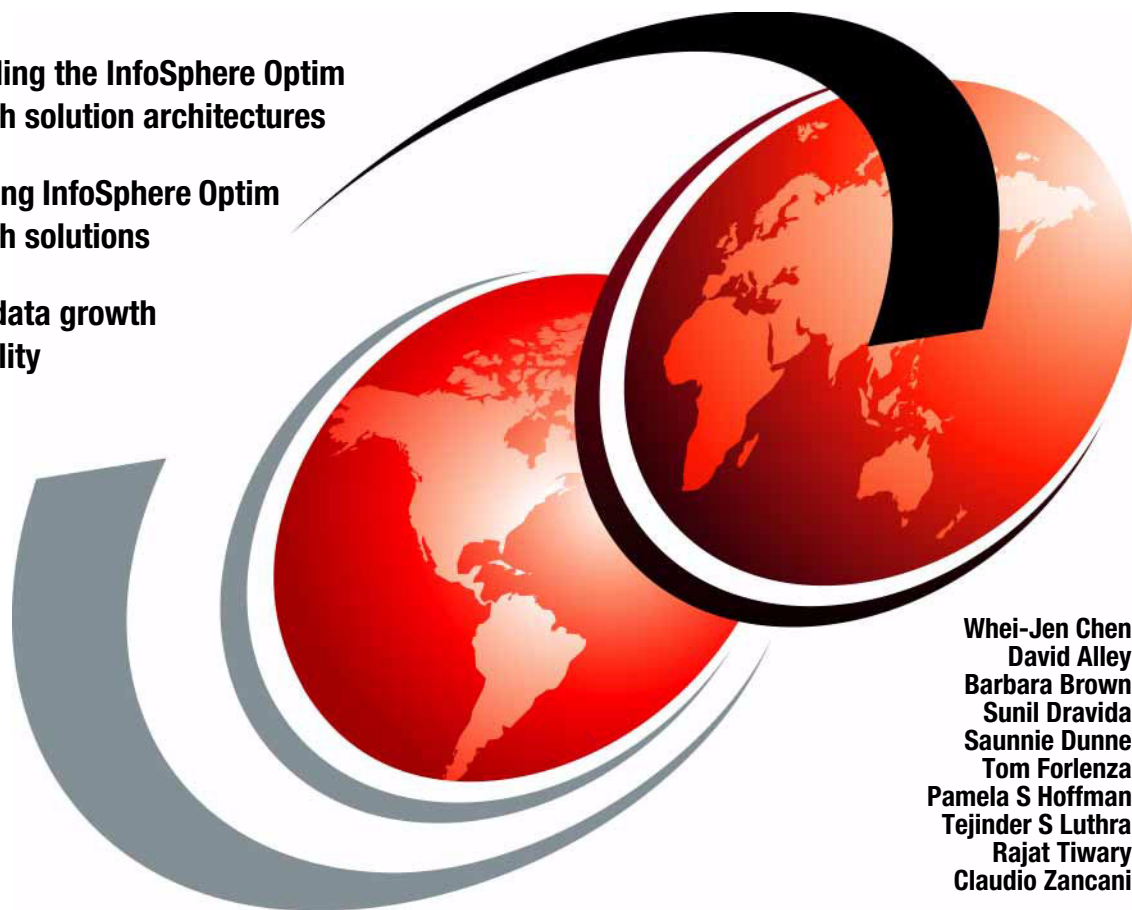


# Implementing an InfoSphere Optim Data Growth Solution

Understanding the InfoSphere Optim  
Data Growth solution architectures

Implementing InfoSphere Optim  
Data Growth solutions

Managing data growth  
with flexibility



Whei-Jen Chen  
David Alley  
Barbara Brown  
Sunil Dravida  
Saunnie Dunne  
Tom Forlenza  
Pamela S Hoffman  
Tejinder S Luthra  
Rajat Tiwary  
Claudio Zancani





International Technical Support Organization

## **Implementing an InfoSphere Optim Data Growth Solution**

November 2011

**Note:** Before using this information and the product it supports, read the information in “Notices” on page xi.

**First Edition (November 2011)**

This edition applies to IBM InfoSphere Optim Data Growth Solution Version 7.3.1.

**© Copyright International Business Machines Corporation 2011. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	xi
Trademarks .....	xii
<b>Preface</b> .....	xiii
The team who wrote this book .....	xiii
Acknowledgements .....	xvi
Now you can become a published author, too! .....	xvii
Comments welcome .....	xvii
Stay connected to IBM Redbooks .....	xviii
<b>Chapter 1. Introduction to IBM InfoSphere Optim</b> .....	1
1.1 Challenges .....	2
1.1.1 Data explosion .....	2
1.1.2 Current approaches .....	3
1.2 Information governance .....	3
1.3 IBM role in information governance .....	4
1.3.1 History .....	4
1.3.2 IBM approach to data governance .....	5
1.3.3 Data governance maturity model .....	7
1.4 Information lifecycle management .....	8
1.4.1 Benefits of implementing the correct ILM strategy .....	11
1.4.2 What is data archiving .....	11
1.5 IBM InfoSphere Optim Data Growth Solution .....	12
1.6 Education support .....	13
1.6.1 IBM Professional Certification .....	14
1.6.2 Information Management Software Services .....	15
1.6.3 Protect your software investment: Ensure that you renew your Software Subscription and Support .....	16
<b>Chapter 2. Project planning and considerations</b> .....	17
2.1 Project management .....	18
2.1.1 Project methodology .....	18
2.1.2 Building the project team .....	18
2.1.3 Establishing the project goals .....	21
2.2 Analyze .....	21
2.2.1 Identify business drivers .....	21
2.2.2 Define requirements .....	23
2.3 Design .....	24
2.3.1 Conceptual design .....	25

2.3.2 Logical design . . . . .	25
2.3.3 Physical design . . . . .	28
2.4 Configure . . . . .	29
2.4.1 Implementation . . . . .	29
2.4.2 Configuration . . . . .	30
2.5 Deploy . . . . .	31
2.5.1 Integration and user acceptance testing . . . . .	31
2.5.2 Deploy to production . . . . .	31
2.5.3 Validation . . . . .	32
2.6 Operate . . . . .	32
<b>Chapter 3. Understanding the IBM InfoSphere Optim Data Growth solutions and software . . . . .</b>	<b>33</b>
3.1 Traditional archiving . . . . .	34
3.2 Optim concepts . . . . .	38
3.2.1 The four pillars of Optim-managed data . . . . .	42
3.3 Enterprise architecture . . . . .	42
3.3.1 Process architecture . . . . .	44
3.3.2 Analyze . . . . .	45
3.3.3 Design . . . . .	46
3.3.4 Deploy and manage . . . . .	49
3.3.5 Source . . . . .	50
3.3.6 Execute . . . . .	52
3.3.7 Target . . . . .	62
3.3.8 Provide user access . . . . .	64
3.3.9 Storage policy . . . . .	71
3.3.10 Security . . . . .	73
3.4 Complete business object . . . . .	74
3.4.1 Elements of a complete business object . . . . .	76
3.4.2 Data-driven processing examples . . . . .	82
3.5 Extract, store, and restore processing . . . . .	85
3.6 Universal access . . . . .	90
<b>Chapter 4. Developing an InfoSphere Optim Data Growth Solution architecture . . . . .</b>	<b>93</b>
4.1 Project management . . . . .	96
4.1.1 Business goals . . . . .	96
4.1.2 Project scope . . . . .	96
4.1.3 Project schedule and phasing . . . . .	97
4.1.4 Methodology . . . . .	98
4.2 Analyze . . . . .	99
4.2.1 Requirements . . . . .	99
4.2.2 Inventory of assets and technologies . . . . .	104

4.2.3 Standards . . . . .	105
4.2.4 Use case models . . . . .	106
4.3 Design . . . . .	107
4.3.1 Optim components . . . . .	107
4.3.2 Software components . . . . .	115
4.3.3 Reference deployment models . . . . .	117
4.3.4 Extensibility of the deployment models . . . . .	123
4.3.5 Hardware components . . . . .	147
4.3.6 Aligning the solution architecture with the Optim design and planned process flow . . . . .	162
4.3.7 Viability assessment . . . . .	163
4.3.8 Preliminary hardware sizing estimate . . . . .	163
4.4 Configure . . . . .	163
4.4.1 Installing and configuring Optim . . . . .	164
4.4.2 Testing, measuring, tuning, and validating . . . . .	165
4.4.3 Hardware sizing . . . . .	165
4.4.4 Modifications as necessary . . . . .	165
4.5 Deploy . . . . .	166
4.5.1 Architecting for future phases . . . . .	166
4.6 Operate . . . . .	166
4.6.1 Review project success criteria . . . . .	166
4.7 Sample Optim solution architecture . . . . .	167
4.7.1 Large Optim deployment . . . . .	167
 <b>Chapter 5. Sizing the Optim infrastructure . . . . .</b>	 173
5.1 Resources for the Optim infrastructure . . . . .	174
5.2 Types of sizing . . . . .	174
5.3 Sizing fundamentals . . . . .	176
5.4 Sizing factors . . . . .	178
5.4.1 Optim installed on Linux, UNIX, and Windows: Data sources . . . . .	178
5.4.2 Optim installed on z/OS: Data source . . . . .	182
5.4.3 Data schema complexity . . . . .	182
5.4.4 How fast can Optim extract the source data . . . . .	183
5.4.5 Hardware sizing considerations . . . . .	185
5.4.6 Align the hardware resource sizing with the deployment model . . . . .	196
5.4.7 Optim design and workflow considerations . . . . .	197
5.5 Sizing metrics, methodology, and process . . . . .	201
5.6 Sizing example . . . . .	203
5.6.1 Sample large, operationally optimized Optim deployment . . . . .	203
5.6.2 Sizing considerations . . . . .	211
 <b>Chapter 6. Data sources . . . . .</b>	 213
6.1 Data source architecture . . . . .	214

6.1.1	Access definition	214
6.1.2	Data-driven engine	215
6.1.3	Example	217
6.1.4	Data-driven engine	224
6.1.5	DB alias	236
6.2	Native data sources	238
6.2.1	Configuration	239
6.2.2	Multiple language considerations	240
6.3	Federated data sources	241
6.3.1	Architecture	241
6.3.2	Using Federation Server	242
6.3.3	Using Oracle	245
6.3.4	Configuring Optim Connect ODBC parameters	252
6.3.5	Optim Connect XML binding parameters	256
6.4	Non-relational data sources	257
6.5	IBM InfoSphere Classic Federation Server	259
6.5.1	Mapping non-relational sources to relational model	263
6.5.2	Import Classic references or run a Database Discovery	265
6.5.3	Array structures	266
6.5.4	Creating views	269
6.5.5	Mapping tables and views for redefined data	270
6.5.6	Creating indexes	271
6.5.7	Generating DDL	272
6.5.8	Granting Classic Federation Server privileges	273
6.5.9	Accessing IMS data	274
6.5.10	Accessing VSAM data	278
6.5.11	Accessing sequential data	281
6.5.12	Accessing SAG-Adabas data	284
6.5.13	Accessing CA-IDMS data	289
6.5.14	Accessing CA-Datcom	293
6.6	Optim Connect	295
6.6.1	Setup procedure	303
6.6.2	Managing metadata	309
<b>Chapter 7. Accessing and viewing archived data</b>		<b>325</b>
7.1	Business requirements for accessing archived data	326
7.1.1	Ad hoc viewing	330
7.1.2	Reporting	331
7.1.3	Federated viewing	333
7.1.4	Optim Data Find	337
7.1.5	Native application access	339
7.1.6	Other viewing from within an application	341
7.2	Managing access to archived data	343

7.2.1 Open data manager . . . . .	343
7.2.2 ODBC drivers for Optim solutions on Microsoft Windows . . . . .	345
7.3 Performance considerations . . . . .	346
7.3.1 Searching archives . . . . .	346
<b>Chapter 8. Design and configuration . . . . .</b>	<b>353</b>
8.1 Design considerations . . . . .	354
8.1.1 Archiving strategy . . . . .	354
8.1.2 Selecting what data to archive . . . . .	355
8.1.3 Restoring archived data . . . . .	355
8.1.4 Making the final design decision . . . . .	355
8.2 A sample federated design . . . . .	356
8.2.1 Sample business requirements . . . . .	356
8.2.2 Sample design . . . . .	358
8.3 A sample z/OS design . . . . .	362
8.3.1 Sample design . . . . .	363
8.4 Miscellaneous considerations . . . . .	365
8.4.1 Optim processing reports . . . . .	365
8.4.2 Optim security . . . . .	366
8.4.3 Change management . . . . .	367
8.4.4 Database schema changes . . . . .	367
8.4.5 Optim naming conventions . . . . .	368
8.4.6 Named objects versus local objects . . . . .	369
8.4.7 Using the export and import utility . . . . .	369
8.4.8 Batch processing . . . . .	370
8.4.9 Matching Optim directories and DBMS code pages . . . . .	371
8.4.10 Handling extreme volumes of data . . . . .	372
8.4.11 Complex data selection requirements . . . . .	372
8.4.12 Special data retention requirements . . . . .	373
<b>Chapter 9. Deployment and operational considerations . . . . .</b>	<b>375</b>
9.1 Data source and operational design . . . . .	376
9.2 Single or multi-instance Optim deployment . . . . .	378
9.2.1 Multi-instance deployment . . . . .	378
9.3 Archival job configuration . . . . .	380
9.4 Restore job configuration . . . . .	381
9.5 Control tables . . . . .	381
9.5.1 Archival job control table . . . . .	382
9.5.2 Restoration job control table . . . . .	385
9.6 Archival job execution . . . . .	387
9.7 Restoration job execution . . . . .	390
9.7.1 Selective restore . . . . .	391
9.7.2 Bulk restore . . . . .	395

9.8 Auditing . . . . .	395
9.9 Monitoring . . . . .	397
9.10 Reporting . . . . .	398
9.11 Tiering . . . . .	399
9.12 Restarting Optim jobs . . . . .	401
9.13 Backup and disaster recovery . . . . .	401
<b>Chapter 10. Optim performance considerations . . . . .</b>	<b>405</b>
10.1 Overview . . . . .	406
10.2 Considerations for processing large data volumes . . . . .	406
10.2.1 Using the current release of Optim software . . . . .	407
10.2.2 Optim archive parallel processes . . . . .	407
10.2.3 Selection criteria . . . . .	408
10.2.4 Archive file indexes . . . . .	408
10.2.5 Optim archive file collections . . . . .	409
10.2.6 Optim process performance reports . . . . .	409
10.2.7 Optim relationship index analysis . . . . .	409
10.2.8 Optim archive processing steps . . . . .	410
10.2.9 Optim table traversal options . . . . .	410
10.2.10 Deleting data after archival . . . . .	411
10.2.11 Partitioned database tables . . . . .	412
10.2.12 Restoring data from archive files . . . . .	412
10.2.13 Optim performance parameter settings . . . . .	413
10.3 Non-relational files . . . . .	415
10.3.1 General performance guidelines . . . . .	416
10.3.2 Processing non-indexed file types . . . . .	416
10.3.3 Trace and log settings . . . . .	417
10.3.4 Performance parameter settings . . . . .	417
10.4 Archive file access . . . . .	417
10.4.1 General performance guidelines . . . . .	418
10.4.2 Archive file indexes . . . . .	418
10.4.3 Archive file collections . . . . .	418
10.4.4 Optim Connect performance parameter settings . . . . .	421
10.5 Analyzing performance with Optim Connect . . . . .	422
10.5.1 Using the Optim Connect explain command . . . . .	422
<b>Chapter 11. Security . . . . .</b>	<b>427</b>
11.1 Basic access security model . . . . .	428
11.2 Optim security architecture . . . . .	431
11.2.1 Definitions . . . . .	431
11.2.2 Standards matrix . . . . .	432
11.2.3 Access flow . . . . .	432
11.3 Authentication and access . . . . .	434

11.3.1	Account management . . . . .	434
11.3.2	Functional security . . . . .	434
11.3.3	Object security . . . . .	435
11.3.4	File access definitions . . . . .	435
11.3.5	Database security . . . . .	436
11.4	Accessing archived data with open data manager . . . . .	436
11.4.1	Prerequisites . . . . .	436
11.4.2	Open data manager security methods . . . . .	437
11.4.3	Open data manager federation configuration . . . . .	441
11.4.4	Implementing open data manager workspace security . . . . .	441
11.4.5	Open data manager and file access definitions . . . . .	449
11.4.6	Open data manager and FAD example UNIX configuration . . . . .	455
11.5	Software . . . . .	456
11.5.1	Program executables and libraries . . . . .	457
11.5.2	Archive, extract, and control files . . . . .	457
11.5.3	Logging, trace, and other temporary files . . . . .	458
11.5.4	Database objects . . . . .	458
11.6	Audit trail . . . . .	459
11.6.1	Optim . . . . .	459
11.7	z/OS . . . . .	460
11.8	Implementation practices for Linux, UNIX, and Microsoft Windows environments . . . . .	460
11.8.1	Initializing Optim security . . . . .	460
11.8.2	Securing your Optim implementation . . . . .	461
11.8.3	Activity classes . . . . .	461
11.8.4	Roles . . . . .	462
11.8.5	Configuration management . . . . .	465
<b>Chapter 12.</b>	<b>Troubleshooting . . . . .</b>	<b>467</b>
12.1	IBM Support . . . . .	468
12.2	Troubleshooting aids . . . . .	469
12.2.1	Environmental information . . . . .	469
12.2.2	Optim processes . . . . .	474
12.2.3	Tracing . . . . .	475
12.3	Issues getting data out of sources . . . . .	482
12.3.1	Troubleshooting your Optim server . . . . .	482
12.3.2	Cannot get to the source data . . . . .	488
12.3.3	Poor archive process performance . . . . .	489
12.3.4	Deleting data from sources . . . . .	492
12.4	Issues accessing data after it has been archived . . . . .	494
12.4.1	Open data manager . . . . .	494
12.4.2	Security . . . . .	496
12.4.3	Poor query performance on archived data . . . . .	496

12.5 Recovery . . . . .	499
12.5.1 Optim objects . . . . .	499
12.5.2 Accessing a duplicate copy of archive files . . . . .	500
<b>Appendix A. Data model . . . . .</b>	<b>501</b>
A.1 Business event processing . . . . .	502
A.1.1 Reference data . . . . .	502
A.1.2 Master data . . . . .	502
A.1.3 Transaction data . . . . .	503
A.2 Order management system . . . . .	503
A.2.1 Ordering products . . . . .	504
A.2.2 Order transactions . . . . .	505
A.2.3 Invoice transactions . . . . .	506
A.2.4 Items master data table . . . . .	507
A.2.5 Customer . . . . .	507
A.2.6 Shipments . . . . .	508
A.3 Optim data model explained . . . . .	509
A.3.1 Logical data model . . . . .	509
A.3.2 Optim sales table . . . . .	510
A.3.3 Optim customers table . . . . .	511
A.3.4 Optim orders table . . . . .	511
A.3.5 Optim details table . . . . .	512
A.3.6 Optim items table . . . . .	512
A.3.7 Optim_Ship_To table . . . . .	512
A.3.8 Optim_Ship_Instr table . . . . .	513
A.3.9 Optim_Male_Rates table . . . . .	513
A.3.10 Optim_Female_Rates table . . . . .	513
A.3.11 Optim_State_Lookup table . . . . .	514
<b>Appendix B. Functions and actions access permissions by roles . . . .</b>	<b>515</b>
<b>Related publications . . . . .</b>	<b>523</b>
IBM Redbooks . . . . .	523
Other publications . . . . .	523
Online resources . . . . .	523
Help from IBM . . . . .	524

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	GPFS™	POWER®
CICS®	IBM®	Rational®
ClearCase®	IMS™	Redbooks®
ClearQuest®	Informix®	Redbooks (logo)  ®
Cognos®	InfoSphere™	System i®
DB2 Connect™	iSeries®	System p®
DB2®	Maximo®	System z®
developerWorks®	MVS™	TCS®
Distributed Relational Database Architecture™	Optim™	Tivoli®
DRDA®	OS/390®	WebSphere®
Dynamic Infrastructure®	POWER7™	z/OS®

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft, Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

SnapLock, NearStore, NetApp, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

Today, organizations face tremendous challenges with data explosion and information governance. InfoSphere™ Optim™ solutions solve the data growth problem at the source by managing the enterprise application data. The Optim Data Growth solutions are consistent, scalable solutions that include comprehensive capabilities for managing enterprise application data across applications, databases, operating systems, and hardware platforms. You can align the management of your enterprise application data with your business objectives to improve application service levels, lower costs, and mitigate risk.

In this IBM® Redbooks® publication, we describe the IBM InfoSphere Optim Data Growth solutions and a methodology that provides implementation guidance from requirements analysis through deployment and administration planning. We also discuss various implementation topics including system architecture design, sizing, scalability, security, performance, and automation.

This book is intended to provide various systems development professionals, Data Solution Architects, Data Administrators, Modelers, Data Analysts, Data Integrators, or anyone who has to analyze or integrate data structures, a broad understanding about IBM InfoSphere Optim Data Growth solutions. By being used in conjunction with the product manuals and online help, this book provides guidance about implementing an optimal solution for managing your enterprise application data.

## The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

**Whei-Jen Chen** is a Project Leader at the International Technical Support Organization, San Jose Center. She has extensive experience in application development, database design and modeling, and DB2® system administration. Whei-Jen is an IBM Certified Solutions Expert in Database Administration and Application Development, and an IBM Certified IT Specialist.



**David Alley** is an Optim Worldwide Solutions Leader. He joined IBM in 2008 bringing 32 years of broad-based IT experience covering both z/OS® and distributed platforms. He has worked with the Optim family of products since 2006 and has implemented Optim solutions in the US, Canada, Europe, and Australia. His IT-related business experience covers both the insurance and banking industries. He has also worked closely with auditors and industry regulators concerning the areas of information privacy and data retention.



**Barbara Brown** is an Information Management Technical Sales Specialist in IBM UK. She has 16 years of experience in IBM working first in the services organization as a DB2 z/OS Systems Programmer. Five years ago, Barbara moved into Software Sales as a DB2 on Linux, UNIX, and Microsoft Windows Specialist and subsequent to the IBM acquisition of Princeton Softech, she has specialized in InfoSphere Optim solutions.



**Sunil Dravida** is an Optim Worldwide Center of Excellence Manager. He began working for IBM in 2007 and has worked with InfoSphere Optim since 2001. He has more than 20 years of broad experience including application development, complex architectures, database administration, Enterprise Resource Planning (ERP) implementations, and executive management. He has a Bachelor's degree in Electrical Engineering and a Master of Science in Computer Engineering with a focus on very-large-scale integration (VLSI) design. He has worked in several industries including financial institutions, insurance, telecommunications, manufacturing, and government. Sunil was involved in one of the largest Optim implementations in the world, architecting an innovative solution for managing the multi-terabyte data growth comprising billions of records generated daily.



**Saunnie Dunne** is the manager of the Information Development team for the Optim solutions. She authored the first user guide for an Optim Data Growth solution, has broad knowledge of all the Optim solutions, and continues to write about these solutions and expand her product knowledge while she carries out her management responsibilities. Saunnie has more than 30 years of experience in writing about technical issues.



**Tom Forlenza** holds a degree in Mathematical Computer Sciences from Kean University. He pursued his degree while working full time starting as a computer operator in the IT department for Casio, Inc. His talents for understanding computing systems were quickly recognized and it was not long before he started programming and engineering software solutions. He eventually ran and managed engineering teams of various sizes and successfully started an IT consulting firm. He has been delivering software solutions successfully that focus on gathering, distributing, and presenting data for over twenty years. For the last nine years, Tom has been part of the Optim team in an engineering and technical sales capacity. His in-depth knowledge of the Optim suite solutions was a valuable asset to this book.



**Pamela S Hoffman** is a Certified IT Architect in the IBM Optim Center of Excellence (CoE). She designs and manages worldwide Optim performance and scalability benchmarks on Microsoft Windows, Linux, UNIX, and z/OS, develops sizing and capacity planning metrics, modeling tools, and methodologies for Optim Data Growth, Test Data Management, Data Privacy, Data Decommissioning, and Application-Aware solutions, and harvests best practices for Optim performance. Previously, she was an I/T Architect in the Oracle International Competency Center, working with Oracle benchmarks and developing sizing and capacity planning methodologies and tools for Oracle applications. She has contributed to three IBM Academy of Technology Workshops on “Capacity Planning, Sizing and Benchmarking”, authored many technical white papers, presented at numerous worldwide technical conferences, and has contributed to several IBM Redbooks publications. She is a member of the IBM I/T Architect Certification Board and a Master Certified Architect with The Open Group.



**Tejinder S Luthra** is an Optim System Architect in the IBM Optim CoE. He is a Certified DBA in Oracle 9i, 10g, and Teradata V2R5, V12. Tej brings over 16 years and a wide range of experience in core database administration, engineering management, software design, and development.

In the CoE, Tej supports and manages worldwide sales activities and driving the proof of concept (PoC) assurance process for Optim. He has been instrumental in producing performance and diagnostic metrics. He conducts strategic workshops and boot camps for clients and IBM Business Partners worldwide. He engages with clients to perform architectural reviews and IDM solution evaluation.

In the past, Tej has served as a lead in IT Operations overseeing extremely large, complex, and diverse implementations of databases and applications, several of them that run in petabytes of data usage. His key contributions have been in monitoring and control, performance tuning, automation, capacity planning, customer relationship management (CRM), data warehousing, backup, high availability, and disaster recovery solutions.

**Rajat Tiwary** is an IT Architect from New Delhi, India. He has 13 years of experience working on various hardware and software platforms, with over 11 years with IBM. His areas of expertise include Enterprise Architecture, Dynamic Infrastructure® Consolidation, Data Center Virtualization, Cloud Computing, IBM InfoSphere Optim Data Growth Solution, UNIX, Linux, IBM Rational® ClearCase® (UCM) and ClearQuest®. He is an Inventor holding a patent in his name with IBM and has been recognized by IBM with a Bravo Award for Demonstrated Technical Leadership and a Client Outstanding Technical Achievement Award for excellent work done in the field of Cloud Computing for IBM clients.



**Claudio Zancani** is an Optimize Chief Architect. He joined IBM in 2008 bringing 33 years of broad-based IT experience as a Development Manager for a mainframe 4-GL at Applied Data Research, the Chief Technology Officer for the Data Access Division at Sterling Software, and the Data Governance Practice Manager at Princeton Softech. His areas of expertise in technology, process, and best practices have helped many clients as well as new IBM Business Partners worldwide establish their own practices and methodologies implementing Optimize. Claudio has written several Optimize technical white papers in conjunction with David, Pam, Rajat, Sunil, and Esli Badenhorst.

## Acknowledgements

The author team wants to recognize and extend a sincere thanks to Esli Badenhorst (deceased) and Robert Amble (deceased) for their many contributions to Optimize and the success of our clients.

A special thanks to **Jeffrey Tuck** for his expertise and written content describing the security around accessing archived data with Optimize open data manager. Jeff is an Optimize Product Manager with IBM Information Management Product Strategy team in Princeton, NJ, US.

A special thanks to **Tim Smith** for his valuable contributions and technical guidance in the areas of performance, security, and open data manager. Tim is an Optimize Development Team Leader, located in Centreville, VA, US.

Thanks also to the following people for their contributions to this project:

Dave Henderson  
Mitesh Shah  
Jim Sinisgalli  
Ricardo Buglio  
Bruce Fischer  
Bob Oakley  
Paula Fricker  
Sam Sampathnathan  
Eric Naiburg  
**IBM Software Group, USA**

Emma Jacob  
**International Technical Support Organization, San Jose Center**

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an email to:  
[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)
- ▶ Mail your comments to:  
IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on Facebook:  
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:  
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>



# Introduction to IBM InfoSphere Optim

Today, organizations face tremendous challenges with data explosion and information governance. The current approaches are not effective anymore. The IBM approach to information governance is unique and helps organizations meet their business objectives in a efficient manner. IBM InfoSphere Optim empowers organizations in managing the life cycle of their enterprise data seamlessly and in accordance with their enterprise policies and industry standard practices. IBM InfoSphere Optim can help organizations with their life-cycle management needs including data growth, test data management, data privacy, and application retirement.

# 1.1 Challenges

Organizations today face with tremendous challenges around the management and governance of data that is siloed across the enterprise. The explosion of data has made it extremely difficult for organizations to meet service-level agreements (SLAs), mitigate spiraling costs, satisfy regulatory compliance requirements, protect the privacy of data, and maintain the high quality of data. Let us look at the challenges of data explosion.

## 1.1.1 Data explosion

The volume and variety of digital information (both structured and unstructured) are exploding as our planet becomes more instrumented, interconnected, and intelligent. Current economic times require corporations and governments to analyze new information faster and make timely decisions for achieving business goals within budget.

There are several factors driving the need for a new way of looking at information and the way that we make decisions based on that information:

- ▶ Volume of digital data

The amount of digital information that organizations create and retain today is staggering. Companies will struggle to deal with this digital expansion as the rate of content generation exceeds corporate storage capacity.

- ▶ Velocity of decision making

The velocity of decision making is about optimizing the speed of insight generated as well as the confidence that the decisions and actions taken will yield the best outcomes based on more proactive planning around the management and use of information sources, and creating far more advanced predictive capabilities.

The resulting explosion of information creates an opportunity for a new kind of intelligence on a smarter planet. The intelligent organizations are looking to use the wealth of information and analytics to reach better, faster decisions and actions, optimized processes, as well as more predictable outcomes. This capability is business optimization.

However, in order for the organizations to take advantage of this information, it must be certified to meet the requirements of the business and effectively managed throughout its life cycle to ensure its relevancy. Systemically knowing what is and is not important and how to connect the information is imperative. The management of these vast amounts of data for business optimization drives the need to govern information so that the organizations can scale and sustain

profitable growth, ensure compliance with policies and regulations, and control costs.

### **1.1.2 Current approaches**

The current approaches taken by clients to manage their information are not working. The most traditional approach is the “keep everything model”. This model has failed. It is too costly and too risky, creates data quality issues, and results in information chaos and overload. Organizations are in a perpetual problem of soaring infrastructure and storage costs, high risks of exposure, and limited visibility to use the knowledge residing across their diverse enterprise environments and to trust that data to drive business optimization. Today, companies still continue to work in silos focusing on departmental immediate needs and really do not know what information content they have, where it exists, or how long they need to keep the information. There is necessary and unnecessary information, and organizations struggle with how to implement a systematic process to manage this growth.

## **1.2 Information governance**

Information governance allows organizations to create order of this information chaos. It provides the means by which organizations can manage this explosion of information. Its about enabling people to do their jobs more effectively by providing them with the decision rights to optimize, secure, and use enterprise data as an asset that can drive business opportunities.

Information governance does not have to be difficult or mysterious. At its simplest, it involves organizing people, processes, and technology to optimize, protect, and use information, both structured and unstructured, as an enterprise asset that must be protected, meet quality standards, and be managed throughout its life cycle.

The management of data through governance is no longer “optional”. The organizations must manage data through governance to be competitive. Your clients expect you to know your data, the executives demand accuracy and expediency, and the regulators and auditors will enforce it. Data custodians are being pushed from all sides to meet these demands.

## 1.3 IBM role in information governance

IBM is not new to the information governance space, as evidenced by Figure 1-1, having pioneered thought leadership in this area since 2005. IBM worked with hundreds of clients, IBM Business Partners, and industry experts to define the requirements for information governance and benchmark that performance for a variety of industries.

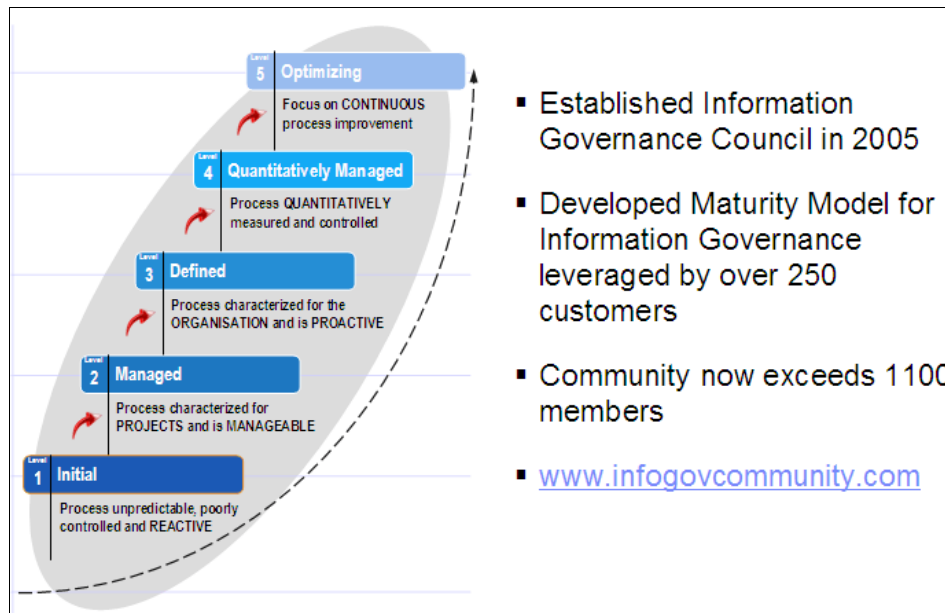


Figure 1-1 Evolution of data governance

### 1.3.1 History

IBM continues to lead the market in the information governance space working with literally thousands of clients to refine the Governance Maturity Model and ensure content and standard practices meet our clients' rigorous demands. A brief look at the major milestones since the Council's inception in 2005 shows this progress:

- ▶ 2005: The Data Governance council was founded under the leadership of Steve Adler from IBM with a focus on governance and continuous process improvement for organizations.
- ▶ 2006: The collaboration of top global companies, IBM Business Partners, and industry experts leads to the development of the Governance Maturity Model. Patterned on the Software Engineering Institute (SEI) Capability Maturity

Model (CMM), the model provides the framework for assessing and evaluating, and defines three core disciplines of governance: Lifecycle, Security & Privacy, and Quality Management. Having an assessment of where you currently are within each category provides a framework for determining where you want to be. For more information about IBM data governance, see this website:

<http://www.ibm.com/ibm/servicemanagement/us/en/data-governance.html>

- ▶ 2007: Over 20 of the council members were using this maturity model, validating its effectiveness and structure.
- ▶ 2008: The council worked with a number of banks and financial institutions, corporations, vendors, and regulators to create a standards-based approach to risk reporting using Extensible Business Reporting Language (XBRL). XBRL is a software language for describing business terms in financial reports. The creation of a risk taxonomy using XBRL provides a vocabulary and a common language allowing everyone to understand what risk means, and that is the first step in making it easier to calculate and report.
- ▶ 2009: The name was expanded from the limitations of “data” governance only to the broader category of “information governance” to keep up with the growing requirements of the council members.
- ▶ 2010: The model’s content is constantly expanded to meet the needs of our clients. The most recent addition was content management.

In 2010, IBM launched <http://www.infogovcommunity.com/>, a private network that is hosted by IBM that has over 1,100 members worldwide. This community is designed to gather, share, and compare ideas about governance approaches among peers.

This year, IBM also published a step-by-step book about how to get started defining a successful governance strategy by industry expert Sunil Soares called *The IBM Data Governance Unified Process: Driving Business Value with IBM Software and Best Practices*.

### 1.3.2 IBM approach to data governance

What is unique about the IBM approach to information governance is that it is not merely another “oversight process” that a company implements, but rather, it is designed with specific objectives in mind to optimize the usage of information across organizational boundaries. Think of the way that your information flows across your organization similarly to the way that goods flow through a physical supply chain. Raw materials are purchased and put together through a series of steps where each step adds value to the next with a common goal to create a finalized product. Any disruptions in that chain of events can disrupt the product creation, the delivery of that product’s schedule, and ultimately, the consumer

waiting for that product. Your organization's data flows through your company in the same concept. However, the physical supply chains are much simpler - usually a 1-to-1 ratio (for example, one windshield for one car). Information is almost always one-to-many, many-to-many (for example, customer data), or systems of systems, integrated across the enterprise. Breakdowns in this supply chain can disrupt business processes and lead to faulty decisions, which can result in poor decisions that cost your organization money.

IBM's information governance approach is about aligning the people, processes, technology, and information to achieve better use of information across an enterprise to truly optimize the usage of information by all participants in that supply chain.

Organizations do not want to simply govern information. They want trusted information that is managed over a strategic period with an end goal in mind. However, they are required to govern information in order to meet legal, regulatory, and business obligations at the same time. Therefore, governance represents the policies and processes associated with defining, creating, and maintaining trusted information over a strategic period, to maximize value across the information supply chain and minimize the cost burden. When deployed this way (that is, tied to business optimization processes), information governance delivers real value rather than being merely another "process" to which to conform.

The organizations that do not govern these supply chains run into a variety of problems all stemming from the same root causes. The problems and breakdowns in information flow manifest themselves in various ways and have a negative impact on the business. As organization data volumes explode, organizations have to change the way that these issues are addressed and address them much earlier. The ability to identify and avoid these symptoms is what enables an enterprise to realize the true potential of the data that it owns.

If we think about a few examples, how many times have you received a report and questioned the numbers on it? Rather than going to the root cause of the problem, you fight your way through it to get that immediate task completed (such as a government reporting requirement or maybe an internal monthly reporting deadline) without going back to the cause of the problem.

How many times has your organization embarked on a new project only to discover that you have disagreement about the fundamental definitions behind the initiative, such as a master data management project to develop a single view of your customer. Who is a customer? Where is the master data stored? Does everyone agree on these fundamentals?

The IBM proven approach to managing these information supply chains proactively is the information governance maturity model. IBM has worked with hundreds of customers as part of the Information Governance Council to develop this model, which was released in 2006. This model is defined this way:

- ▶ The aspects of information governance as reflected by 11 core categories of competencies, as well as the key indicators within each aspect
- ▶ A set of maturity indicators for each of these categories

The 11 categories of governance can be divided into four inter-related groupings:

- ▶ Business outcomes: Focused on reducing risk and increasing value, which is, in turn, driven by reducing costs and increasing revenues
- ▶ Enablers: Including organization structure and awareness, policy, and stewardship
- ▶ Core disciplines: Including data quality management, information lifecycle management, and information security and privacy
- ▶ Supporting disciplines: Including data architecture, classification and metadata and audit information, logging and reporting

IBM software solutions are categorized around these core disciplines of quality, life cycle, and security and privacy. Let us take a look at how this model applies to the information supply chain and what is required for each core discipline category.

### **1.3.3 Data governance maturity model**

Figure 1-2 on page 8 depicts the Information Governance Council maturity model.

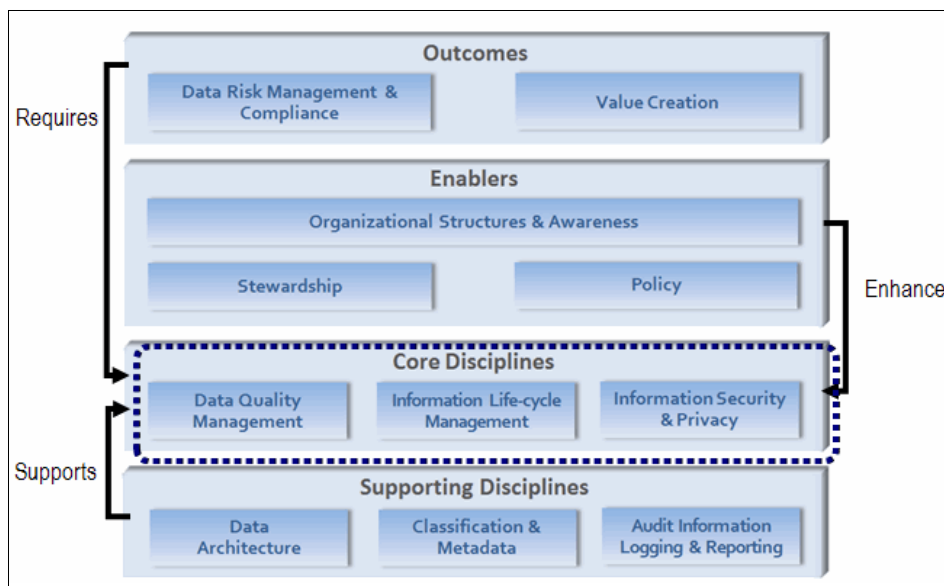


Figure 1-2 Information Governance Council maturity model

The diagram represents the levels of the maturity models. Not everyone attacks all levels at the same time, and measurement is at a category level, not across the board or across all organizational areas. Where you are, and where you want to be, is the choice of your organization, dependent upon your priorities and objectives.

## 1.4 Information lifecycle management

Let us take a look at the core discipline of information lifecycle management (ILM) and the impending challenges the organizations face without a governance program to help the business focus on managing data from its origin to its retirement.

All organizations are challenged by multiple application releases and the delays caused in meeting service-level goals (SLGs) and SLAs. When organizations are fighting to stay competitive, better manage information, as well as improve and support business processes, developing and deploying new and updated applications “on-time” can be critical. There are road blocks along the way.

Test and development environments often multiply (“data multiplier effect”), adding to the complexities of application development, which can impede progress. There are several factors that can feed into this overall problem, including the growing demands for additional disk storage, need for additional database licenses to create the needed non-production environments, and additional time required to provision the environments and refresh them, which affects the performance of the developers waiting for them.

Also, a lack of understanding of the relationships between data objects repeatedly delays projects. More automation of discovery and design tasks is needed to accelerate delivery.

Are you planning to validate test data easily to ensure that errors are caught in the test, development, and quality assurance (QA) processes?

And throughout all this work, what data are you using to create the test and development environments? How are you tracking sensitive data? Are you in compliance with industry regulations? Is the developer upgrading the human resources and payroll system “really” required to see everyone’s salary information? Or can that person use realistic, but fictional data, to complete the work?

Information lifecycle management addresses and provides solutions for these questions. Information lifecycle management is a process for managing information through its life cycle, from conception until disposal, in a manner that optimizes storage and access at the lowest cost.

Figure 1-3 illustrates how the life cycle of data in the information supply chain is managed.

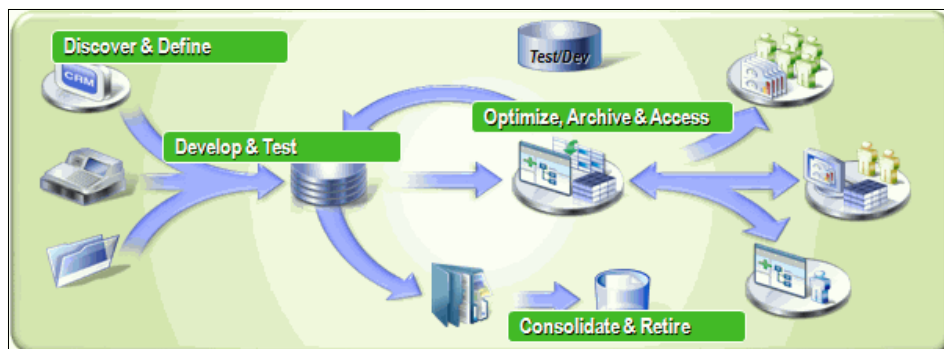


Figure 1-3 Managing the life cycle of data in the information supply chain

Managing the life cycle of the data within your organization's information supply chain involves the following steps:

1. Discover and define

You need to discover the data and define the relationships across information as it comes into the enterprise. The following challenges exist around not knowing your data:

- You might have a lot of data scattered around the organization and you cannot find it.
- Not knowing how certain data relates to other enterprise data.
- No knowledge of where the data resides and how long you are supposed to retain that data.
- Not knowing where all of your organization's sensitive data exists in order to protect it.

2. Develop and test

You must develop applications and functionality that can best maintain your data. You must effectively test those applications. Consider the following best practices for deploying the systems that will manage your enterprise data:

- How do you ensure collaboration across your development ecosystem?
- How can you speed development through common models and policies?
- How can you effectively create and manage development and test environments and protect the data within those environments?

3. Optimize, archive, and access

You have to ensure that the application performance remains optimized and that the data is managed throughout its life cycle and accessible upon request.

Enterprise data management has the following day-to-day activities:

- Ensure that application performance is optimized so that the business users can access the data efficiently, and so that IT staff's time is used effectively.
- Use intelligent archiving techniques so that data retention requirements are adhered to, while still providing access to that data.

4. Consolidate and retire

You have to manage your application portfolio effectively. Over time, the applications managing your data will require upgrading, consolidating, and eventually be retired, but not the data.

Ensure that you can access business-critical data for long-term data retention, long after an application's life expectancy.

### 1.4.1 Benefits of implementing the correct ILM strategy

Implementing the right ILM strategy in your organization offers the following benefits:

- ▶ Understanding the “what and where” of enterprise data. You will have a better understanding of all of your data across the enterprise, which will help you make the right decisions at the right time.
- ▶ Developing models and code to store and access enterprise data, including the configuration of data for test environments. You will be better informed to create your models and metadata for the models for more meaningful test environments.
- ▶ Optimizing the performance of applications through the identification of bottlenecks and building the right strategy for managing data growth. By implementing the right data growth solution, which includes archiving historical data from your production environments, you will work with a smaller footprint of data, and thus, can improve the performance of applications, including the online transaction and batch processing.
- ▶ Implementing a consistent process for retiring or consolidating applications as their usage expires. The additional advantage of implementing the right ILM strategy is a standards-based approach for retiring or consolidating your applications as their business usage expires.

Let us discuss further what archiving your production data means.

### 1.4.2 What is data archiving

Archive in its simplest form is simply the migration of information (data) from an online application to an offline, accessible long-term storage repository. Archiving helps to address these problems or “pain points”:

- ▶ “Save everything” attitude
- ▶ Excessive storage costs (physical and electronic)
- ▶ Litigation (underway or impending)
- ▶ Government audit and investigation (underway or impending)
- ▶ Excess time to locate materials and records
- ▶ Lack of records management policies and procedures
- ▶ System changes (especially to electronic document and records system)
- ▶ Email and filing email
- ▶ Vital records protection
- ▶ Long-term preservation
- ▶ Outdated retention schedules
- ▶ Security of confidential information
- ▶ Meeting regulatory and compliance requirements

However, in order to use the efficiencies of the use of archived information, it is necessary to ensure that the data archiving meets your enterprise requirements.

For the enterprise, the process of information lifecycle management is the optimal approach of data archiving. At its simplest form, information lifecycle management manages the disposition of information/data from its creation to its placement on initial online storage to its migration to longer-term storage devices to its final deletion.

In today's IT environments, most organizations use tiering of online storage to optimize the efficiency of their environment.

## **1.5 IBM InfoSphere Optim Data Growth Solution**

The IBM InfoSphere Optim Data Growth Solution is a single, scalable enterprise data management solution that is designed to meet your evolving business needs around ILM strategies. From small to large organizations, from single applications to global business centers, data management is streamlined using a consistent proven strategy.

With InfoSphere Optim Data Growth Solution, you can simplify enterprise data management to accelerate business-critical projects. It helps you provide open access to current and archived data, makes upgrades easier, supports in implementing cost-effective, tiered storage strategies, and improves application performance and availability.

With capabilities that allow you to manage application data growth across your enterprise, you can improve application service levels, mitigate risk, and control costs. InfoSphere Optim Data Growth Solution also helps organizations control the costs of data growth.

By streamlining critical databases and simplifying complex infrastructures, you can use your existing IT investments. Key IT processes require fewer resources. With InfoSphere Optim Data Growth Solution, you are able to reclaim storage capacity and take advantage of a cost-effective data growth management strategy.

Protecting your company from legal liability and other liabilities is critical. InfoSphere Optim Data Growth Solution enables you to mitigate the risks that are encountered in business today.

The data management capabilities of InfoSphere Optim Data Growth Solution allow you to apply functional policies to govern data retention. You can automate data retention to support compliance initiatives and respond quickly and

accurately to audit and discovery requests. And in the event of a disaster, employing a staged recovery strategy helps ensure the continuity of your business.

InfoSphere Optim Data Growth Solution is a single, scalable solution that supports all major enterprise databases and operating systems, including IBM DB2®, Oracle®, Sybase®, Microsoft® SQL Server®, IBM Informix®, IBM IMS™, IBM VSAM®, Microsoft Windows®, UNIX®, Linux®, and IBM z/OS®. It also supports the key business applications in use today, such as Oracle E-business Suite, PeopleSoft® Enterprise, JD Edwards® EnterpriseOne, Siebel®, and Amdocs® CRM, and all of your custom and packaged applications.

## 1.6 Education support

Available from IBM training are the newest offerings to support your training needs, enhance your skills, and boost your success with IBM software. IBM offers a complete portfolio of training options, including traditional classroom, private onsite, and eLearning courses. Many of our classroom courses are part of the IBM “Guaranteed to run program,” ensuring that your course will never be canceled. We have a robust eLearning portfolio, including instructor-led online (ILO) courses, self-paced virtual courses (SPVC), and traditional web-based training (WBT) courses. A perfect complement to classroom training, our eLearning portfolio offers something for every need and every budget; simply select the style that suits you.

Be sure to take advantage of our custom training plans to map your path to acquiring skills. Enjoy further savings when you purchase training at a discount with an IBM Education Pack - online account, which is a flexible and convenient way to pay, track, and manage your education expenses online.

The key education resources that are listed in Table 1-1 on page 14 have been updated to reflect IBM InfoSphere Optim Data Growth Solution and Test Data Management Solution for DB2 z/OS.

Table 1-1 IBM Optim course

Course title	Classroom	Instructor-led online	Web-based	Self-paced virtual class
Configuring and Using IBM Optim Data Growth for Archiving on z/OS	DT80	3T80	1T80	N/A
Configuring and Using IBM Optim Data Growth for Archiving on Distributed Systems	DT20	3T20	1T20	2T20
Configuring and Using IBM Optim Core Technology on Distributed Systems	DT29	3T29	1T29	2T29
Configuring and Using IBM Optim Designer and Optim Designer	DT15	3T15	1T15	2T15

Check your local Information Management Training and Education website or with your training representative for the most recent training schedule. The descriptions of courses for IT professionals and managers are available at this website:

[http://www.ibm.com/services/learning/ites.wss/tp/en?pageType=tp\\_search](http://www.ibm.com/services/learning/ites.wss/tp/en?pageType=tp_search)

Visit <http://www.ibm.com/training> or call IBM training at 1-800-IBM-TEACH (1-800-426-8322) for scheduling and enrollment.

## 1.6.1 IBM Professional Certification

Information Management Professional Certification is a business solution for skilled IT professionals to demonstrate their expertise to the world. Certification validates skills and demonstrates proficiency with the most recent IBM technology and solutions. Table 1-2 lists the IBM Professional Certification examination that is available for IBM InfoSphere Optim.

Table 1-2 IBM Professional Certification program

Exam number	Exam name	Certification title
000-551	IBM Optim Implementation for Distributed Systems (2009)	IBM Certified Specialist - IBM Optim Implementation for Distributed Systems

For additional information about this exam, see this website:

<http://www-03.ibm.com/certify/tests/ovr551.shtml>

You can learn about additional examinations at the IBM Information Management Certification Page:

[http://www-03.ibm.com/certify/certs/dm\\_index.shtml](http://www-03.ibm.com/certify/certs/dm_index.shtml)

## **1.6.2 Information Management Software Services**

When implementing an Information Management solution, it is critical to have an experienced team involved to ensure that you achieve the results that you want through a proven, low-risk delivery approach. The Information Management Software Services team has the capabilities to meet your needs, and it is ready to deliver your Information Management solution in an efficient and cost-effective manner to accelerate your return on investment (ROI).

The Information Management Software Services team offers a broad range of planning, custom education, design engineering, implementation, and solution support services. Our consultants have deep technical knowledge, industry skills, and delivery experience from thousands of engagements worldwide. With each engagement, our objective is to provide you with a reduced risk and an expedient means of achieving your project goals. Through repeatable services offerings, capabilities, and best practices using our proven methodologies for delivery, our team has been able to achieve these objectives and has demonstrated repeated success on a global basis.

The key services resources listed in Table 1-3 on page 16 are available for InfoSphere Optim Data Growth Solution.

Table 1-3 InfoSphere Optim Data Growth Solution service offerings

Information Management Services offering	Short description
InfoSphere Optim Install Package	Entry-level services to install InfoSphere Optim on a distributed development environment, such as Linux, UNIX, or Windows (LUW) for one system at your site.
InfoSphere Optim Data Growth Offering	Project planning and execution to address data growth. Three levels of services: <ul style="list-style-type: none"> <li>► Gold: Enterprise-level implementation</li> <li>► Silver: End-to-end archiving project from one application</li> <li>► Bronze: Starter kit with limited scope implementation</li> </ul>
InfoSphere Optim Decommissioning Foundation Services	The objective is to support a proper governance process for ongoing decommissioning. The service includes decommissioning one application for one selected database or one z/OS data source to build the foundation of an end-to-end approach.
InfoSphere Optim HealthCheck Package	Assessment of the configuration and usage of the Optim environment to identify exposures that can potentially extend processing or recovery times and to identify opportunities for achieving improved performance, availability, and scalability of the Optim system.

For more information, visit our website:

<http://www.ibm.com/software/data/services>

### 1.6.3 Protect your software investment: Ensure that you renew your Software Subscription and Support

Complementing your software purchases, Software Subscription and Support gets you access to our world-class support community and product upgrades, with every new license. Extend the value of your software solutions and transform your business to be smarter, more innovative, and cost-effective when you renew your Software Subscription and Support. Staying on top of on-time renewals ensures that you maintain uninterrupted access to innovative solutions that can make a real difference to your company's bottom line.

To learn more, visit this website:

<http://www.ibm.com/software/data/support/subscriptionandsupport>



## Project planning and considerations

In this chapter, we discuss the process of planning for your archiving project. We refer to the Optim Data Growth Solution as *Optim*. Optim archiving projects are driven by a number of elements that can include corporate strategy, IT management, and legislation. All these elements have implications for archiving projects and generate a significant community of interested parties. We discuss these drivers and the key participants of an archive project.

We look at a wide variety of requirements of which certain requirements are extrapolated from the initial drivers for the project and other requirements are supplied by the architectural principles within the enterprise and operational restrictions.

Many steps within the archiving project are common to most IT projects. We discuss an Optim project in this context through design, configuration, deployment, and finally into operation against the production data source.

Clearly, deleting data after archive from production systems has an inherent risk associated with it. A project that includes archiving and deletion of data must conform to rigorous testing methods to be sure all the data required is correctly archived before deletion. Risk management forms an integral part of the project from end to end.

## 2.1 Project management

In the project management phase, you decide on your project methodology, the participants within the project, and the goals of the project. We provide a potential methodology and list of team members, but your corporate and project management standards might dictate an alternate approach. The detail that is provided here still is applicable within the framework of your standards.

### 2.1.1 Project methodology

An Optim Data Growth project follows standard project management methods. We look at the following phases and the elements and considerations that influence the project:

- ▶ *Project Management*: Planning for your project is key and will deeply involve the data owners to ensure that data is always held safely with no loss and is accessible in a timely fashion.
- ▶ *Analysis*: Provides an understanding of the relevant data sources and the requirements for the solution. You establish service-level goals (SLGs) and service-level agreements (SLAs).
- ▶ *Design*: A requirements-led design phase ensures integration with enterprise standards. Use standard Optim deployment models incorporating appropriate inputs, such as security and performance.
- ▶ *Configure*: In this phase, you configure the system to provide a physical implementation of the solution and appropriate settings within Optim.
- ▶ *Deploy*: Comprehensive testing is essential before undertaking the deletion of data from production systems.
- ▶ *Operate*: When in production, Optim is essentially another application deployed against the database server. There are operational implications for the archive solution, as well as the database, as a result of archiving and removing data from production.

### 2.1.2 Building the project team

IBM has a strong history of advocating data and information governance as a corporate strategy and we continue to find data governance practices being established in many organizations. If such a practice exists in your organization, the practice members will be key participants in the data growth project being established.

The project team members need to be people who are concerned with application management, database administration, project management, records management, data compliance auditing, and user functions. The information technicians, DBAs, and people in similar functions might be the most compelled to act. The business organizations that will classify the data as “hot”, “warm”, or “cold” are equally essential to the project. They supply the business requirements for archiving, reporting, and accessing the archived data.

Part of the team building exercise must include an understanding of the skill sets available within the organization, planning for education, and bringing in external resources, as required. Planning the project also includes the transition to production so inclusion of the teams that will be responsible for the archiving solution in production will facilitate the project planning and contribute to the success of the project.

Product education, as well as enablement through a services engagement, is available from IBM to support projects.

The following list is by no means exhaustive and the final list of team members will vary depending on the size and scope of your project. Specific roles might overlap in smaller projects and several of the roles will be drawn from operational teams.

- ▶ Project manager:
  - Manages all aspects of the project.
  - Creates and tracks the project plan.
  - Allocates resources.
  - Manages risk.
- ▶ Compliance team:
  - Compliance officer:
    - Ensures that the compliance obligations of the business are included. For example, data retention policies are understood and documented.
    - Ensures that regulatory and audit requirements are enforced.
    - Validates access to archive data for compliance and reporting purposes.
  - Compliance auditor:
    - Audits the validation routines to ensure that the correct data is being archived and only successfully archived data is deleted.
    - Confirms the availability of data within the time that is specified in the SLA that is mandated by regulations.

- ▶ Design authority (DA) or enterprise architect:
  - Ensures integration with enterprise architecture and adherence to architectural principles.
  - Provides framework of work products for design teams.
  - Signs off work products and designs in adherence with architectural and design standards.
- ▶ Data owners (usually arranged by application or business):
  - Interpret the compliance requirements within the business data.
  - Provide business reporting requirements.
  - Validate reporting requirements for archived data access.
- ▶ Optim architect:
  - Responsible for producing a requirements-driven architecture and working within the existing enterprise architecture framework.
- ▶ Optim project administrator:
  - Responsible for Optim software artifact creation and maintenance.
  - Runs and schedules Optim processes.
  - Troubleshoots any issues that arise from a process.
  - Acts as the subject matter expert (SME) for infrastructure.
- ▶ Optim system administrator (platform-specific skills required):
  - Installs and configures Optim servers.
  - Installs Optim and provides system-level support.
  - Responsible for Optim database and environment recoverability.
  - Acts as the SME for infrastructure.
- ▶ Application developers:
  - Subject matter experts concerning application data and how it is used by the business.
  - Subject matter experts for all source data models that are relevant within the archiving project.
- ▶ Network administrator:
  - Connects Optim servers to the network.
  - Analyzes data locations to optimize network traffic for Optim processes.
- ▶ Security administrator:
  - Develops and administers the security matrix (by role).
  - Provides the team members with the appropriate permissions to complete their assigned tasks.

- Secures Optim artifacts and data contained in Optim archive files, which might include Optim's internal security.
- ▶ Storage administrator:
  - Provides disk space and other storage media based on requirements.
- ▶ System Integration team member:
  - Performs system integration test (SIT) to ensure that Optim processes are functioning as intended after they are implemented or changed.
- ▶ Quality assurance team member:
  - Provides user acceptance test (UAT) to ensure that Optim processes are functioning as intended after they are implemented or changed.
- ▶ Change control administrator:
  - Integrates Optim processes into the existing change control environment or establishes a new change control system.

### **2.1.3 Establishing the project goals**

Project goals include a high-level perspective of the expected benefits of the project and enumerate potential applications containing data for archiving. In the analyze and design phases of the project, you validate the proposal of these applications and the data within them. The business drivers and requirements are developed around the candidate applications.

## **2.2 Analyze**

In the analysis phase of the project, you review the data to be archived, the archival and retention rules that apply to the data, and the key benefits you expect to realize as a result of data archival.

### **2.2.1 Identify business drivers**

There are many areas that can instigate a data archiving project and these areas fall into three major categories: performance, compliance, and cost reduction.

Your project might have more than one driver but it is important to prioritize the drivers to balance the implications of your drivers on the project and design of the solution. We look at the most common drivers first and discuss the implications of these drivers later.

## Performance

When looking at performance as a driver, we most typically see systems where “cold” data inhibits the access to “hot” data. Additionally, we might see less direct issues, such as fitting data housekeeping (for example, backups) into available batch windows or lengthy software and hardware upgrades due to data sizes.

## Compliance

There is a growth in stringent legislation around the world determining what you can and cannot do with the data that is held in organizations. Compliance with regulation can be an extremely inflexible driver of an archiving project. Typically, an organization’s obligations will already have been understood and documented, and where this documentation exists, it needs to form an input to the project.

Compliance can consist of several parts. There are potentially complex webs of retention policies and rules. There are also specific requirements for providing that data to legislative bodies both in the form of the data and the rapidity of response to any data request.

## Cost reduction

Cost reduction is a common driver for archive projects and can be prompted by the longevity of a business or by fundamental changes within a business:

- ▶ Growth

Businesses that have physically expanded or been in existence for a long time will have a long history of data located in applications that is no longer required online for core business activities.

- ▶ Merger and acquisition

Where companies have come together, there can be data held in one or another of the organizations that is not required for the consolidation of the two entities.

- ▶ Strategic direction

Companies might have made fundamental strategic decisions that have implications on IT (for example, a new customer care system), which means the existing data must be migrated. “Cold” data can be archived at this point to improve other IT projects (reduce the hardware requirements and length of project).

► Footprint

Environmental considerations are becoming more important. They are driven both by corporate conscience and the inability to physically find enough space to support ever growing data centers.

Archiving data can reduce the footprint that is required for production systems:

- Companies can reduce the physical numbers of servers and CPUs, and lower memory requirements, along with potential software licensing reductions.
- Organizations can reduce the storage size of backups and disaster recovery sites by removing data duplication.

## 2.2.2 Define requirements

After you understand what is driving the instigation of the archiving project and have prioritized the drivers, you must translate these drivers along with other corporate standards into a set of requirements. You must also define what the expected outcome of the project is, the expected results, and what constitutes a successful project. Without these measurements, it is impossible to tell whether the project has been completed successfully.

Your specific project management methodology can define how you treat the requirements gathering phase and if you differentiate between functional and non-functional requirements, as well as business and technical requirements. However, the result needs to be a mixture of prioritized and measurable requirements. These requirements typically form a requirements matrix.

For example, if the business driver is performance in your production system, it is important that you have a set of benchmarks run prior to undertaking the archiving project, which can be used for measurable comparisons when the deployment is underway. This set of benchmarks will facilitate the proper tracking of the improvements in performance and provide proof points for an appropriate sign-off and completion.

We provide more information about defining requirements in Chapter 4, “Developing an InfoSphere Optim Data Growth Solution architecture” on page 93 and 8.2.1, “Sample business requirements” on page 356.

### Functional requirements

Functional requirements define what the system will “do”, for example:

- Validate the applications, database servers, and candidate archive data.
- The business retention and deletion rules.

- ▶ How the data will be archived and managed through its life cycle.
- ▶ External components, such as storage systems.
- ▶ How the data will be accessed after it has been archived.

## Non-functional requirements

Non-functional requirements define what the system will “be”, for example:

- ▶ Size of data to be archived: Performance and throughput
- ▶ Data growth: Capacity and scalability
- ▶ Availability and disaster recovery
- ▶ Security

You might want to perform gap analyses to understand where you are today compared to your expectation, because these analyses will influence the dates within the project plan. Areas where you are already strong will take less effort to complete than those areas with larger gaps.

We spoke earlier about the expected results of the project. We define these goals as the *service-level goals* (SLGs). In order for the project to be deemed successfully completed, you must be able to measure the improvements that are made as part of the project. For example, this measurement might be before and after performance statistics or storage footprints. You must think about the SLGs at this stage, because the measurements must be consistent at each stage of the project.

Also, you must establish the *service-level agreements* (SLAs) early in the project. These SLAs differ from the service-level goals in that the service-level goals are associated with the project and the service-level agreements are the operational agreements that maintain and potentially improve on the service-level goals. You also must establish key performance indicators (KPIs), such as “time taken to complete archiving”, as part of the project and operational considerations.

## 2.3 Design

The design of the solution iterates through several stages to provide the final solution for the project and the operational environment. Depending on the drivers and hence the requirements, you might need to design for both project and operational phases.

For example, if this project is a post-acquisition project, you might want to archive a large proportion of data from a system to be decommissioned while the live data from that system is moved into the new environment. Data from the new environment might then need archiving on an ongoing basis. The data from both

systems, however, is financial data and might need to be reported from a single perspective.

Chapter 4, “Developing an InfoSphere Optim Data Growth Solution architecture” on page 93 covers the design in detail.

### 2.3.1 Conceptual design

The conceptual design focuses on *data* and considers the following elements:

- ▶ From which data sources the data will be archived.
- ▶ What the data model looks like.
- ▶ Classifying the data into hot, warm, and cold. Data that is in use by the business to perform core activities must not be a candidate for archiving.
- ▶ When to dispose of the data and what the data life cycle looks like.
- ▶ How to create the archive both in terms of data content and incremental time slices.
- ▶ How to access data after it has been archived:
  - Through the original application
  - Through a third-party reporting tool
  - Using simple query functions
- ▶ How to delete data from the original source:
  - Through recoverable SQL deletes
  - Using non-recoverable mechanisms, such as dropping partitions, a truncate table, or dropping an entire database

Understanding the data model and identifying candidate data for archiving can be a time-consuming activity within archiving projects. Tools are available to expedite this stage and make the information available for many data-driven projects, including archiving. InfoSphere provides a rich toolset to identify data-driven relationships, including cross-database relationships and sensitive data; to look at ERP application metadata to understand the logical relationships held within the application logic; and to help identify data that is not frequently or regularly used within an enterprise.

### 2.3.2 Logical design

Now that you understand your data and how you have to treat that data within your archiving solution, you translate this information into a logical design. The logical design brings in the component parts of your solution and their relationships to each other.

To convert typical non-functional requirements, such as performance and security into your design, several deployment models are available, which we will begin to describe in this section.

### Deployment models

We describe the typical deployment models used in the solution design for an archiving project further in Chapter 4, “Developing an InfoSphere Optim Data Growth Solution architecture” on page 93. Here, we take a first view of the flexibility of the solution to meet requirements. The use of various deployment models is often driven by non-functional requirements, in particular performance.

### Centralized

A centralized model is the simplest approach with a single, centralized Optim server and Optim directory for accessing the data sources, creating the data archives, and accessing that data after the archive and deletion from the source systems. The Optim server is the engine that performs the work and the Optim directory holds the metadata that is associated with the archiving processes. We explain these elements in more detail in later chapters.

Figure 2-1 depicts the centralized archiving model.

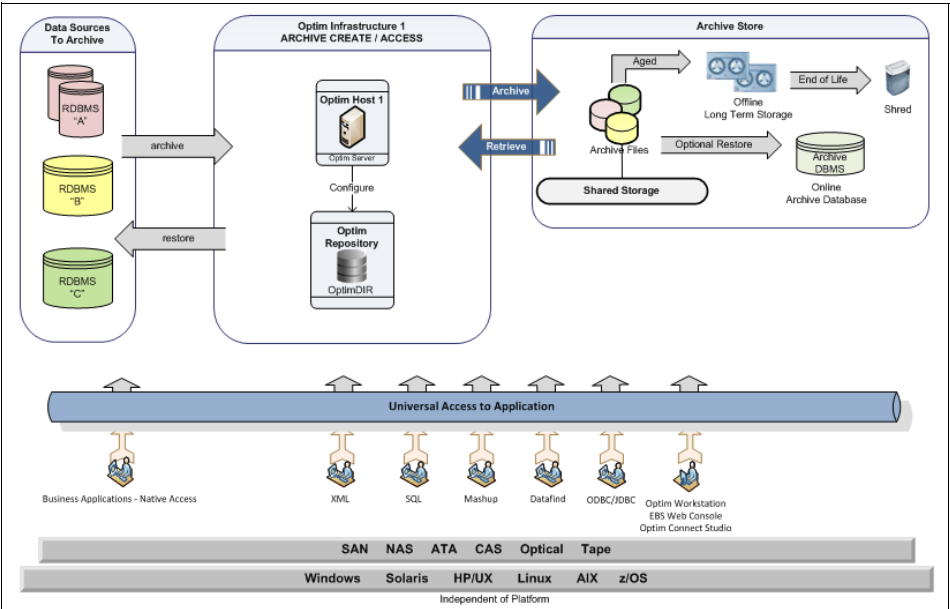


Figure 2-1 Centralized archiving model

## Distributed

The distributed model has two or more Optim servers, which create archives and allow access to the archived data while sharing a single Optim directory to access the metadata. This configuration can occur for many reasons, for example, providing low network latency by placing Optim servers close to the source data.

Figure 2-2 depicts the distributed archiving model.

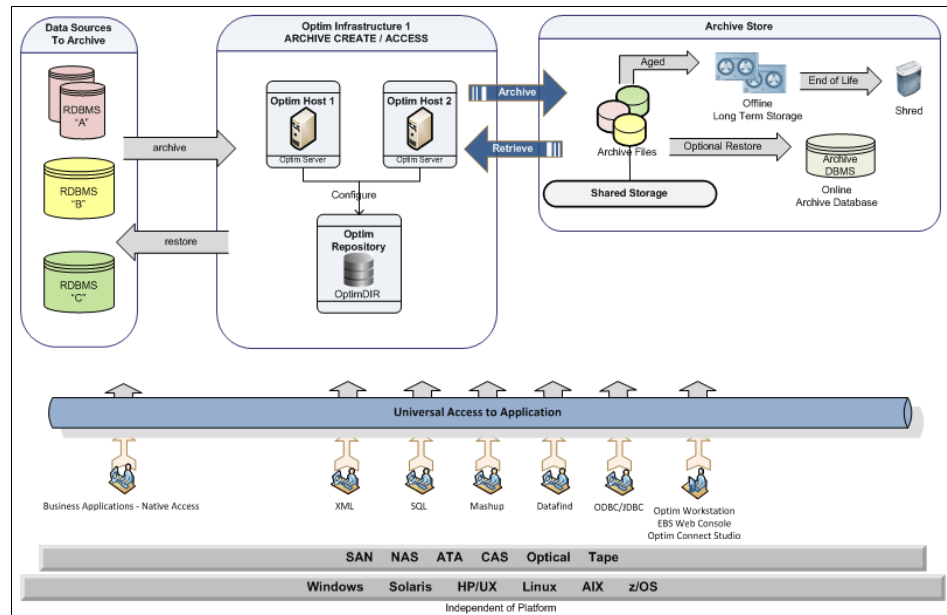


Figure 2-2 Distributed archiving model

## Operationally optimized

In an optimized system, two servers share an Optim directory for the creation of the archives as in the distributed model, but access to the archived data is provided by a separate Optim server and Optim directory.

The major benefit from this configuration is that the two major workloads, the archiving and accessing archived data, do not need to share resources. This optimized system works well in systems with frequent archive processes or with a large number of users querying archived data.

Figure 2-3 on page 28 depicts the optimized archiving model.

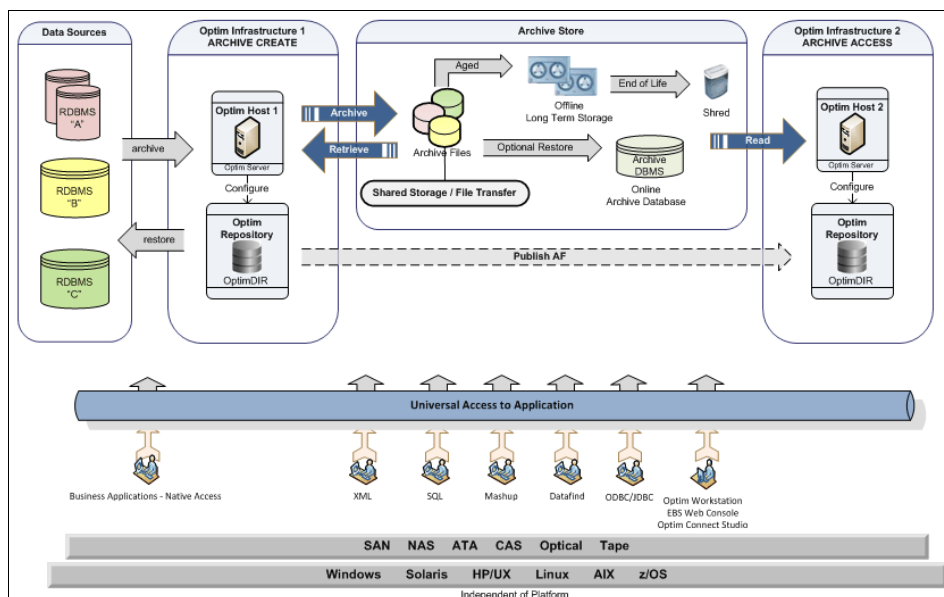


Figure 2-3 Optimized archiving model

## Security

Security usually is driven by corporate standards. Because the data contained within the archive files is production data and potentially sensitive, the security of the data is essential.

Optim helps secure data by restricting the ability of users to use Optim to access the data in the source systems. This level of security is important, because Optim potentially has the authority to not only read data in production, but to copy and delete data also.

The second level of security to consider is access to the data after the data is in the archives. Optim can also protect the data within the archive files. We discuss security in detail in Chapter 11, "Security" on page 427.

### 2.3.3 Physical design

Physical design provides the next level of detail within your design process. It provides the exact physical implementation of your solution. You have a clear view of what you want your solution to provide logically; now you must decide exactly "how" that is to be delivered.

Within the physical design, you consider the sizing of specific components. An understanding of the data sizes and target times for processes is key to this

phase. You also look at where your physical components will be located. Security can be a key influencer here.

The storage sizing is mainly driven by the size of the data to be archived. Access requirements also affect the storage sizing, because the archive indexes will add to the storage needs of the solution as in any normal database. You also have to consider increasing the storage size if you use tiered storage for the life-cycle management of the data.

The physical size and location of the server or servers are influenced by the deployment model you have chosen and the time scales in which data needs to be archived from the data sources. Additionally, your expected growth and capacity planning will influence if you use a scale-up or scale-out model.

Understanding the overall environment where the solution will be deployed also affects the physical design. Network latency coupled with data sizes affects both the sizing and physical location of the servers.

## 2.4 Configure

This project phase encompasses the initial implementation of the solution, including installing software and ensuring connectivity to the data sources, other software, physical components, such as the systems where the archive data will reside (storage, tape, and so on), and the reporting solution (for example, original application or reporting software).

Optim provides a GUI to establish and develop the Optim artifacts that drive the archiving solution. However, when in production, archiving is rarely done through the GUI because we use the batch approach for automation, which is true regardless of the platform.

### 2.4.1 Implementation

After you have established your requirements-based design, you must install all the hardware and software components. Installation is covered comprehensively in the Optim manuals. In this section, we concern ourselves predominantly with the stages after installation.

After you have completed your installation, you must ensure connectivity between all the components. You have to ensure connectivity between Optim and the source data and between Optim and the archived data for reporting. A common project approach is to build and develop the solution on a non-production environment connecting to a suitable data source for both

security and risk reasons. Data sources for this stage can be existing non-production systems or snapshots of an appropriate system specifically for the use of this project. It is easier to develop in a stand-alone environment with fewer security and change control restrictions. You will mitigate the risk of removing production data until the solution is fully tested and the team is comfortable with accessing and removing production data. Typically, to move into production, this initial system is disconnected from the system on which the development and testing occurred and connected to the production system.

For more detailed information about data sources, refer to Chapter 6, “Data sources” on page 213.

## 2.4.2 Configuration

After you set up and connect your environment, you start configuring your archiving solution. We talked about the data model earlier in this chapter, and this task is where that information will be used. You must understand what your data looks like and how you want to slice this data for archive and retrieval.

One of the Optim’s key strengths is its ability to take a referentially intact subset of data out of the production system and place it in an archive. In order for Optim to do this function, it might be necessary to establish relationships between the tables from which you want to extract data. Optim will populate relationships that are enforced through the database or, as mentioned in 2.3.1, “Conceptual design” on page 25, tools are available to populate the relationships within Optim. From these relationships, you begin to build your Optim artifacts, particularly your *access definition* (AD). We explain the access definition in more detail in Chapter 3, “Understanding the IBM InfoSphere Optim Data Growth solutions and software” on page 33. It is sufficient to state at this point that an Optim access definition defines the data model for the data to be archived inclusive of relationships and archive indexes. Iterative testing will be required to ensure that you get the appropriate data into your archive through your archive processes.

Depending on how you intend to access your data after it has been archived, you might also start to think about relevant indexes on archived data. You have the option to access the archived data directly from the compressed archive files or from a reporting database. There are certain instances where it is appropriate to restore to the original production system, but careful consideration must go into this plan and the implications for data duplication within the archive store. See Chapter 7, “Accessing and viewing archived data” on page 325 for further detail on accessing archived data.

If you want to implement Optim security, as discussed in Chapter 11, “Security” on page 427, we suggest that you layer the security over the successful archive

processes at this point. It is better to bring security into the picture after you are sure that your archive process is successful and you have the required data in the archives. A similar approach is useful for implementing life-cycle management, adding this layer after the previous steps are proven to be successful.

## 2.5 Deploy

As you plan to deploy your solution into production, you perform final testing and deployment activities.

### 2.5.1 Integration and user acceptance testing

As expected with any project, you perform integration testing to ensure that all the components operate together. Integration testing is particularly pertinent when using external components to meet requirements.

The following typical elements are areas where corporate standards mean that components, other than core Optim, might be in control:

- ▶ Security
- ▶ Information lifecycle management  
Usually a type of storage solution.
- ▶ Job scheduling

After your Optim processes are placed into batch processes, they must be tested for performance within the batch window.

User acceptance testing (UAT) typically involves the users that are responsible for communicating with auditors regarding compliance requests. The users will confirm that reports run successfully or ad hoc access is appropriate for the requirements.

### 2.5.2 Deploy to production

If you develop your solution on the future production Optim solution but you archive data from non-production systems, deployment to production will consist of redirecting the Optim server to the appropriate production database server.

If you have developed your solution on hardware that is to remain a non-production system, you have to revisit the installation and configuration

steps. Subsequently, you can use Optim's extended capabilities to promote the archive processes from development to test to production.

### 2.5.3 Validation

Finally, the archive processes must be fully validated within the production environment before any deletion of data can be allowed. We discuss methods to check the validity of data within an archive prior to delete in Chapter 12, "Troubleshooting" on page 467.

If you use Optim to delete data in production, Optim optionally can verify the contents of the archived data before deletion. This verification can cause performance overhead, especially in systems with extremely large amounts of data. In these cases or where corporate standards do not allow deletion of the data by third-party applications, deletions can be performed using normal database functions, such as dropping a partition.

## 2.6 Operate

An archiving project is not a stand-alone discrete piece of work. It works alongside the normal management of data within the enterprise on an ongoing basis.

Now that you have successfully designed and implemented your archiving solution, you must operate it on a daily basis and track KPIs to ensure that the SLAs are met.

All the design stages need to have been documented by now. After the Optim artifacts are designed and placed into batch processing, you must produce the usual supporting documentation about the expected behavior and the working instructions for operators (for example, a control book). We describe automating and adjusting the Optim solution for ongoing operations in Chapter 9, "Deployment and operational considerations" on page 375.

Changes to the application and underlying data structures can require updates to the archive solution, and change management for the application and database must reflect these updates.



# Understanding the IBM InfoSphere Optim Data Growth solutions and software

In this chapter, we describe the IBM InfoSphere Optim Data Growth solutions (the Optim solutions) and the Optim software for data growth management. We introduce several optional components that are part of the solution set. The Optim solutions are the cornerstone for archiving structured data. We explain the use of each component as a building block in the solution.

In this part of the book, we introduce terminology and discuss the following topics:

- ▶ Traditional archiving
- ▶ Optim concepts
- ▶ Enterprise architecture
- ▶ Complete business object
- ▶ Extract, store, and restore processing

## 3.1 Traditional archiving

The meaning of archiving differs depending the audience. For production operators, archiving is a physical backup. Production operators perform in an environment where data has a short life span and is deeply connected to a production system. Hardware and software can save data quickly and the backup is a binary representation of the original system. The backup might serve for litigation protection because, for a certain period of time, the backup and the original system are mirrors of one another. If backups are used to archive data, however, a challenge arises when the backup data must be retrieved after the schema in the original system has evolved. If, for example, the size of a database column in production differs from the size of the column in the backup. In an extreme example, an environment change can occur so that Linux is used in production while the backup was created from Microsoft Windows or the production data resides in DB2 but the backup was of data in Oracle. These changes to the production environment make unusable all clones created before the change.

In any case, a backup cannot solve your data growth problem; it adds to it. For example, assuming incremental backups weekly and full backups monthly, a source of one terabyte increases to ten terabytes in a cycle of six months. If you also assume an average of eight clones to cover disaster recovery, testing, quality assurance, and training, all with backups, you end up with many more times the original amount of data.

You might try to manage your data growth by implementing a partitioning schema to separate active data from historical data. After partitioning, you can reduce the size of backups and clones by making selected partitions read-only. By partitioning in this way, you have not reduced the overhead on the database, however, because the indexes remain the same size. For example, assume in a company with 10,000 online users and a one terabyte database:

- ▶ That 95% of the users are active 100% of the time with 20% of the data (hot data)
- ▶ That 5% of the users are active 10% of the time with 100% of the data (hot, warm, and cold data)

In this example, 99.5% of the queries must use an index on a terabyte of data that is 80% unused. There is no formula to determine the amount of additional CPU that is required to keep 80% of the index data for 0.5% of the queries, but there is a definite cost. If your historical data is older than four years (cold data), it is probably accessed only a few times a year and solely by auditors. This data remains 100% important, but 50% to 90% of the index is overhead.

By implementing a history database and, thus reducing the index size, you can address the issue of unused portions of the index. To transfer data to the history partitions, however, you must understand the business rules for that data. It is clear that a Details table without an Orders table is problematic, but an Orders table without a Details is acceptable unless it is in the Backorder table. Rather than move a partition, you must plan to move transactions that are “closed”. Selecting the closed transactions requires navigating tens of the hundreds of tables in the data model. Database referential integrity (if present) is a good starting point, but referential integrity does not apply in all cases; applications enforce certain rules. Only by looking at the data or at the code can you build a successful “move or purge” strategy.

From a business perspective, the idea of purging data is not easily accepted. Applications are interdependent and, with web portals, business users can obtain a complete view of the customer business object without knowing where the applications are. To obtain business user approval to purge historic data from a production database, you must capture information within its business context. In the example discussed here, you fortunately are not dealing with a packaged application, in which thousands of tables have no database management system (DBMS) relationships, and you must interpret the packaged application catalog. Also, obtaining sign-off from business users often translates into archiving data from multiple heterogeneous applications. For example, if archiving an Orders table that is stored in an SQL Server database, you must also capture Customers table data from IMS™. Knowing that an SQL Server DBA is probably not familiar with concepts, such as IMS segment, multi-record layout, COBOL copybook level 88, or EBCDIC data type “packed decimal”, you must add a z/OS expert to your team, or find a method to dynamically map non-relational data to relational data.

As another consideration, it is common for a database to reference external documents, such as contracts, invoices, and signatures. Archiving data from a database with associated documents requires backing up the file system or systems for any external documents, or implementing a method to capture external documents in the archive.

Before you can say that you have conquered data growth, you must purge data. Obviously, only the history data can be purged, and the rest of the business data must remain. Following referential integrity rules, you know that related data (children) are deleted before parent data. What if the data modeler has defined a recursive relationship in which parent and a child are the same table? What if the application expects the sum of all invoices for 2006 to be USD1,289,892.32, and the invoices are purged? You cannot simply delete rows to purge data. You must be concerned about the sequence in which rows are deleted across tens of tables. You must deal with sometimes circular relationships. You must generate data when an application expects it. And, you must provide an audit trail for the data that you purge.

How do you judge if an archiving project is successful? A project is successful if, when the business users ask, “Why did we approve that mortgage in 2008?”, you can obtain the answer from the archive, which contains all pertinent information. In other words, archived data requires an access strategy.

Providing access through an archiving database might be appealing at first. To accomplish this access, you must place the data that you archive in a database and secure the data by granting read-only access to business users. To save disk space, you can also compress the archived data. The backup strategy for the archive database is an important detail. You also must remember to back up the documents that you archived from the original application. The backup frequency depends on the frequency with which you store new archived data, perhaps quarterly, monthly, weekly, or daily.

In many cases, the service-level agreement (SLA) allows you to use network-attached storage (NAS) and a server. It might be less expensive to use an Hierarchical Storage Manager to demote less frequently used data to offline devices, but most databases do not provide support for demoting less frequently used data to offline devices. From time to time, you must be ready to alter the database to reflect minor data model changes, for example, a wider or new column. Because you are altering a read-only database, you also might need an audit process to demonstrate that you are adding new data and not changing existing data.

To access production and archived data from an application, you can join information using federated union views. If you want to access history information, but you archived only transaction data, you must implement distributed queries that require more complex SQL. The business data (such as customer and account) is in the production database, and a federated union view does not provide the correct information. Another consideration is that the archive database grows quickly, so you need a purge strategy to remove data at the end of the retention period.

You must also plan to provide the data for reprocessing by the original application. Of course, the history data is never reprocessed except when a business user tells you, “we forgot that...”, “accounting did...”, or “the law requires...”. Regardless of the reason, out of millions of archived transactions, you might have to locate and restore ten. Imagine having to restore (actually, re-create) items in the partition that you dropped two years before, immediately before IT upgraded the DBMS to Version 12q and switched from Microsoft Windows to AIX®.

Perhaps while reading this example, you identified similar project challenges of your own. We describe the following challenges:

- ▶ Analysis:
  - Discover application relationships
  - Discover referential Integrity
  - Discover data relationships
- ▶ Design:
  - Data in business context
  - How to handle data model changes
- ▶ Deployment:
  - Minimizing skill set requirements
- ▶ Source data:
  - Heterogeneous sources
- ▶ Execution:
  - Enterprise architecture
  - Purging original data
  - Related documents
  - Data comparison
- ▶ Target archive:
  - Binary representation of source data
  - Immutable archived data
  - Compression
  - Archive database
  - Re-created database structure
- ▶ Provide archive access:
  - SQL access
  - Transparent access
- ▶ Project policy:
  - Expiring archived data
  - Separate storage
- ▶ Project security:
  - Security
  - Traceability

## 3.2 Optim concepts

The Optim solutions are consistent, scalable solutions that include comprehensive capabilities for managing enterprise application data across applications, databases, operating systems, and hardware platforms. You can align the management of your enterprise application data with your business objectives to improve application service levels, lower costs, and mitigate risk.

In Figure 3-1, you can see that data growth is part of *information lifecycle management* (ILM). Specifically, to manage explosive data growth, the Optim solutions enable organizations to enhance their use of trusted information by segregating critical business data from history data using a managed data process (see 3.2.1, “The four pillars of Optim-managed data” on page 42).

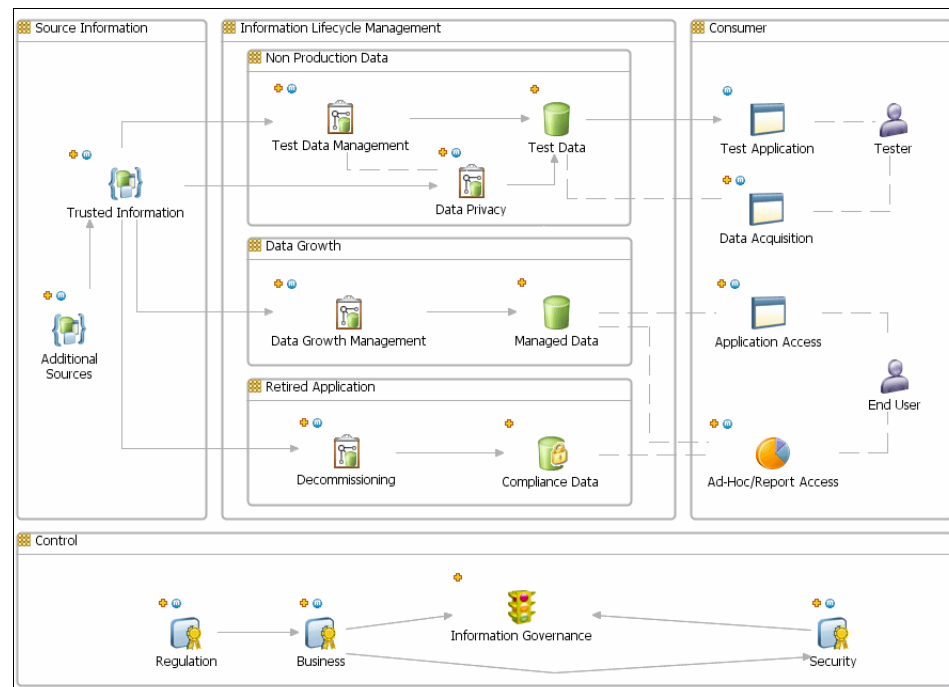


Figure 3-1 Blueprint of Optim-managed data, a component of information lifecycle management

Figure 3-2 on page 39 further defines the high-function data growth management. The use of data discovery functionality identifies data in its business context or as a complete business object. Using the definition for the complete business object, an Optim solution can archive data (copy) to an immutable, compressed archive file (no need to audit) and purge the history data

(warm or cold) from the trusted information source, including the data in VSAM files that the SQL Server DBA does not really understand. The manage archive functionality stores the archived data in active or long-term storage using a storage policy. Additionally, it keeps the archived data for as long as a retention policy indicates. In the case of a litigation hold, the solution prevents the system from deleting the archived data.

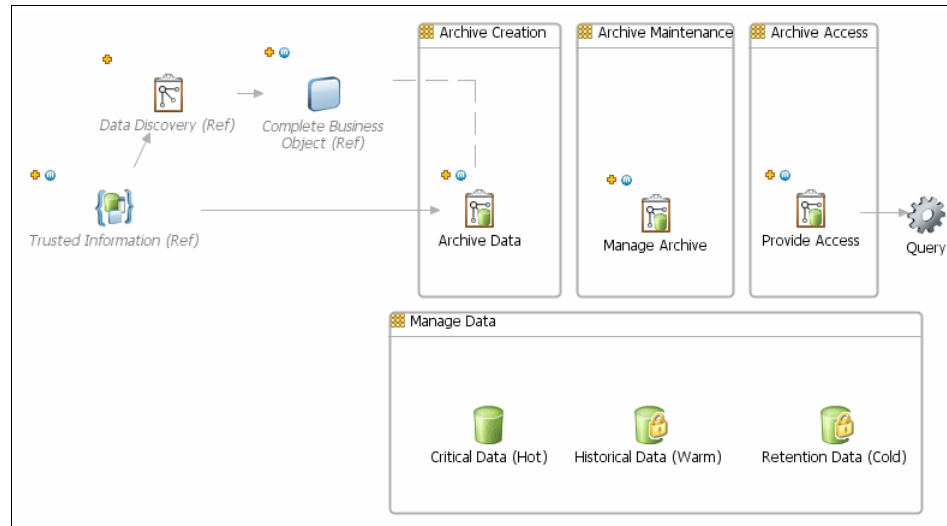


Figure 3-2 Blueprint of Optim-managed data

The Optim solutions *provide access* to the compressed archived data without needing to restore or decompress it. This capability allows you to use Java Database Connectivity (JDBC), Open Database Connectivity (ODBC), or Object Linking and Embedding Database (OLEDB) to perform standard ANSI SQL-92 requests to access archived data. If you must populate an archive database (a reporting, audit, or read-only database whether homogeneous or heterogeneous), the solutions provide both the functionality and the traceability. If your relational environment requires seamless access to both current and historical data, the solutions provide a transparent access layer that does not require application changes. The concept of managed data is the value proposition for the Optim solutions, connecting technology to process and value.

The Optim solutions enable the introduction of data management functionality to new or already existing applications and databases (see Figure 3-3 on page 40). The Optim solutions archive (extract and purge) history data (warm and cold) and related business data into an SQL-accessible, immutable, and compressed container (archive).

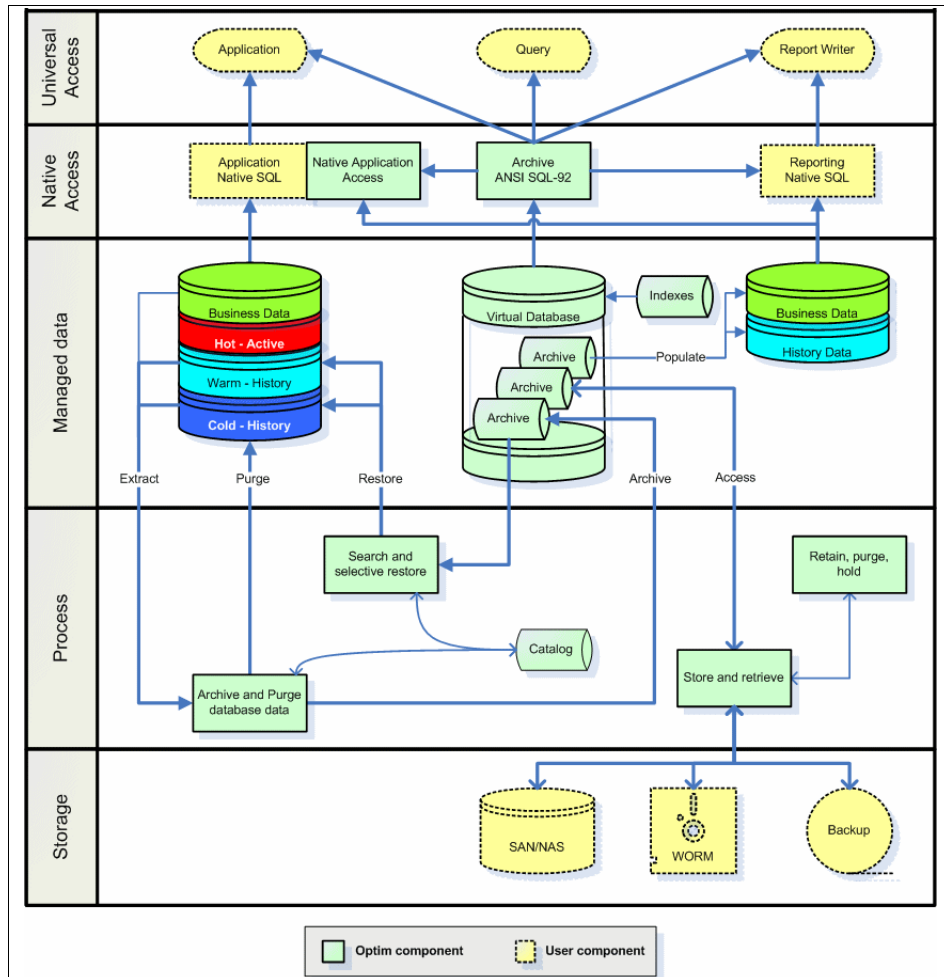


Figure 3-3 Managed data process

Archives are recorded and tracked in the *catalog*, or *Optim directory*, during the life cycle. The source application and the database have the following benefits:

- ▶ Improved DBMS performance as a result of removing data that is accessed less frequently by fewer users
- ▶ Reduced storage requirements for DBMS data and indexes
- ▶ Shorter elapsed times for backup and disaster recovery as a result of segregating data that is subject to updates (update data) from data that is not subject to updates (read-only data)
- ▶ A process that can be audited

- ▶ Creation of a regulatory-compliant and immutable snapshot-in-time of transaction data while retaining the business context of that data

The smaller data footprint that results from archiving accrues to the non-production environments, such as development, test, quality assurance, and training, and offers these benefits:

- ▶ Smaller storage requirements for clones
- ▶ Shorter elapsed time to refresh non-production environments
- ▶ Decreased computer power requirements to privatize or mask data
- ▶ Shorter elapsed time for backup processing
- ▶ Reduced requirements for floor space, power, and cooling

Because compressed archived data can be accessed directly, without further manipulation, you can perform these functions:

- ▶ Access data in a relational manner (*ANSI SQL-92*) using JDBC, ODBC, or OLEDB.
- ▶ Place archives into Optim collections (virtual database) and access them, treating each collection as a partition in a database.
- ▶ Integrate your existing applications (federated access) with active (hot) and history (warm and cold) data directly from the collection (virtual database).

For greater performance or flexibility, you can perform these functions:

- ▶ Restore specific transactions (search and selective restore) to the original database (homogeneous restore).
- ▶ Restore specific transactions to a database that differs in type (heterogeneous restore).
- ▶ Integrate your existing applications (native application access) with active (hot) and historical (warm and cold) data by placing (populating) data in a secondary read-only archive database.

The Optim solutions associate archives with a retention policy to enable purging them from both the catalog and the storage device. This purge policy is overridden in the case of a litigation hold. Additionally, the solutions enable the creation of a sub-archive containing only data pertinent to the litigation hold. The solutions offer the capability to transfer ownership of the litigation-hold archive to a separate catalog while applying new retention and storage policies.

The storage policies associated with archives enable the storage of archives on devices that differ from typical file system mounts. The Optim solution supports multiple classes of storage:

- ▶ Primary: Storage area network (SAN) and NAS
- ▶ WORM: DVD and SnapLock

- ▶ Backup: Tivoli® and tape
- ▶ Hybrid: Information Archive and EMC Centera (behave like a backup and a WORM device)

### 3.2.1 The four pillars of Optim-managed data

Optim-managed data consists of four pillars:

- ▶ Enterprise architecture
- ▶ Complete business object
- ▶ Extract, store, and restore processing (predefined functionality for the Optim solutions)
- ▶ Universal access

## 3.3 Enterprise architecture

Figure 3-4 provides a high-level representation of the extensive support for enterprise technologies provided by the Optim solutions. The four initiatives (*Data Growth*, *Application and Data Retirement*, *Test Data Management*, and *Data Privacy*) support both packaged, predefined applications and custom applications.

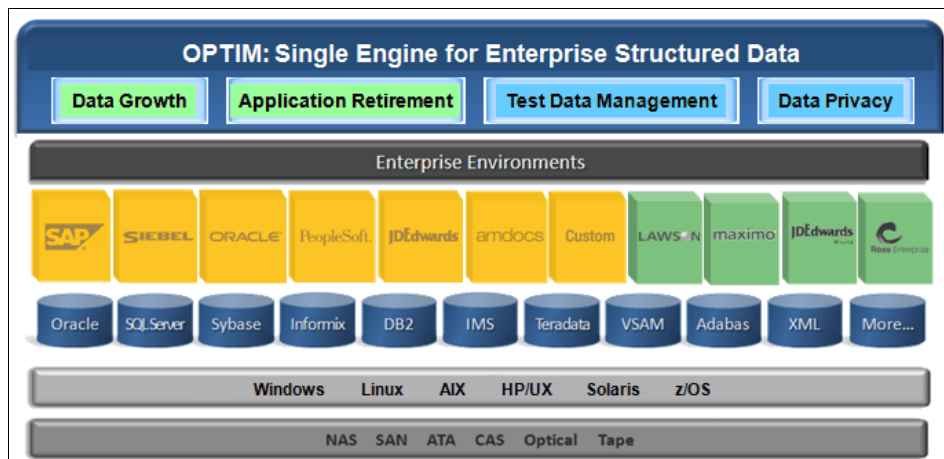


Figure 3-4 Single engine enterprise architecture for Optim solutions

Optim solutions support these packaged applications:

- ▶ SAP Applications

- ▶ Siebel Customer Relationship Management
- ▶ Oracle E-Business Suite
- ▶ PeopleSoft Enterprise
- ▶ JD Edwards EnterpriseOne
- ▶ Amdocs
- ▶ Lawson
- ▶ IBM Maximo® Enterprise Asset Management
- ▶ Ross Enterprise ERP

Supported custom applications have data residing on a variety of databases:

- ▶ Oracle
- ▶ Microsoft SQL Server
- ▶ Sybase
- ▶ Informix®
- ▶ DB2
- ▶ IMS
- ▶ Teradata
- ▶ VSAM
- ▶ Adabas
- ▶ Datacom
- ▶ IDMS

The supported databases reside on many operating systems:

- ▶ Microsoft Windows
- ▶ Solaris
- ▶ HP/UX
- ▶ Linux
- ▶ AIX
- ▶ OS/390®
- ▶ z/OS
- ▶ IBM System i®

Also, the Optim solutions support various of archive storage capabilities:

- ▶ Network-attached storage
- ▶ Storage area network (SAN)
- ▶ Advanced technology attachment (ATA)
- ▶ Content addressed storage (CAS)
- ▶ Optical
- ▶ Tape

### 3.3.1 Process architecture

The enterprise architecture in Figure 3-4 on page 42 is presented from a process point of view in Figure 3-5.

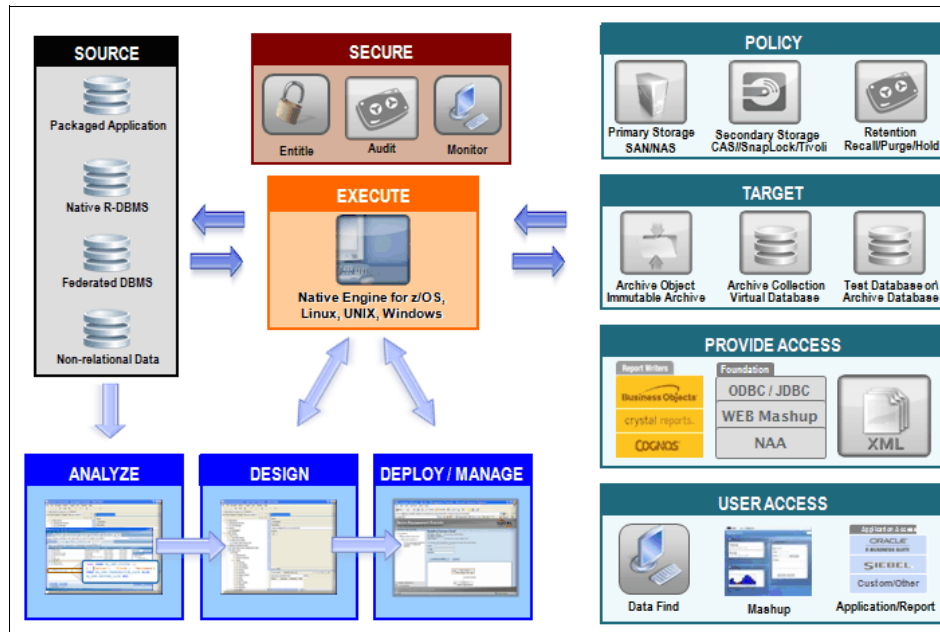


Figure 3-5 Single engine enterprise architecture high-level processes

The Optim solutions enable you to *analyze, design, deploy, and manage* your enterprise solution using engines on Linux, UNIX, and Microsoft Windows platforms and in IBM zEnterprise System (System z®) to archive and restore data from relational database management systems (RDBMSs) natively or through JDBC or ODBC, from federated DBMSs, and non-relational data sources.

Also, archived data can be stored in a variety of targets and is subject to storage and retention policies that safeguard the content during the data life cycle, including litigation hold. Archived data can be accessed (provide access) natively by applications (Native Application Access [NAA]), reporting engines (Cognos®) and portal engines (web mashup). You can universally access (user access) archived data from the native application or using portals and reports (web query), as well as by using unstructured access (search) to mine data or prepare for a litigation hold.

The Optim solutions secure data and process through *entitlements, audits, and active monitoring*.

### 3.3.2 Analyze

Imagine a small database of 100 tables. If each table has 100 columns, the database includes 10,000 columns. How realistic is it to expect a human being to memorize the roles of 10,000 columns and to keep track of any changes to those roles? Now, consider that most applications have thousands of tables and sometimes millions of columns and that large enterprises have thousands of applications.

To address the scale and complexity of data model discovery in modern software systems, IBM offers, as part of the Optim solutions, a full range of capabilities to automate the analysis process. These capabilities go well beyond reverse engineering and data profiling to perform sophisticated analysis that generates actionable results, results that are used during design. Three methods apply when analyzing and documenting a complete business object:

- ▶ Data-driven process
- ▶ Data model-based analysis
- ▶ Packaged application metadata analysis

#### **Data-driven process**

A critical first step in any archiving project is the accurate representation of the business object to be archived. The data-driven discovery component complements the InfoSphere Optim Data Growth Solution by capturing both database-defined and application-managed relationships that make up the business object to be archived. With this component, you have automated capabilities for analyzing data values and data patterns to identify relationships that offer greater accuracy and reliability than manual analysis. Disparate data elements are linked into logical units of information or “business objects”, such as customer, patient, or invoice. A business object represents a group of related attributes (columns and tables) of data from one or more applications, databases, or data sources. You can also identify transformation rules that have been applied to a source system.

#### **Data model-based analysis**

The data model-based analysis component is a collaborative, enterprise data modeling and integration design tool designed to help architects understand information assets and their relationships, design and implement data-driven projects, and standardize and govern heterogeneous data assets. You can perform these tasks:

- ▶ Import an existing logical model, and forward engineering changes to that model to the database.

- ▶ Reverse engineer a model from an existing source database, update the model, and then use compare and sync to generate appropriate change statements to deploy the changes back to the original source database.
- ▶ Publish a web-based report of the physical model for auditing and team review.

### **Packaged application metadata analysis**

The packaged application metadata analysis component is a metadata exploration tool for enterprise applications, such as PeopleSoft Enterprise, JD Edwards EnterpriseOne, Siebel, and Oracle E-Business Suite. This component makes the data definitions from these major ERP and CRM packages available in an easy-to-understand form, allowing you to explore ERP data structures without specialist application knowledge.

This fully automated process allows a quick, accurate, and easy way to find relationships in Oracle ERP/CRM applications, offering you the ability to automatically identify application data models, relationships, and customizations and to analyze data model changes.

Artifacts from this component are easily integrated with other IBM data modeling tools, such as InfoSphere Data Architect, and provide a common model resource for seamless governance across your enterprise application base. You can use this component to support new or custom-built modules or prepare for an application upgrade. You can also make the discovered relationships and customizations available to your Optim solution as a complete business object from which you can build actionable processes for managing data growth for your packaged Oracle ERP and CRM applications.

### **3.3.3 Design**

The design component that is shown in Figure 3-6 on page 47 is a design time workbench that you can use to configure data access and management services for data sources and file system-based XML documents. This component provides a standard visual tool that you can use to design, build, and test data growth services. Optim provides the seamless integration of data access plans with models, policies, and metadata defined at the beginning of the data life cycle to allow a common understanding of the business and technical context of your data across every stage of the life cycle.

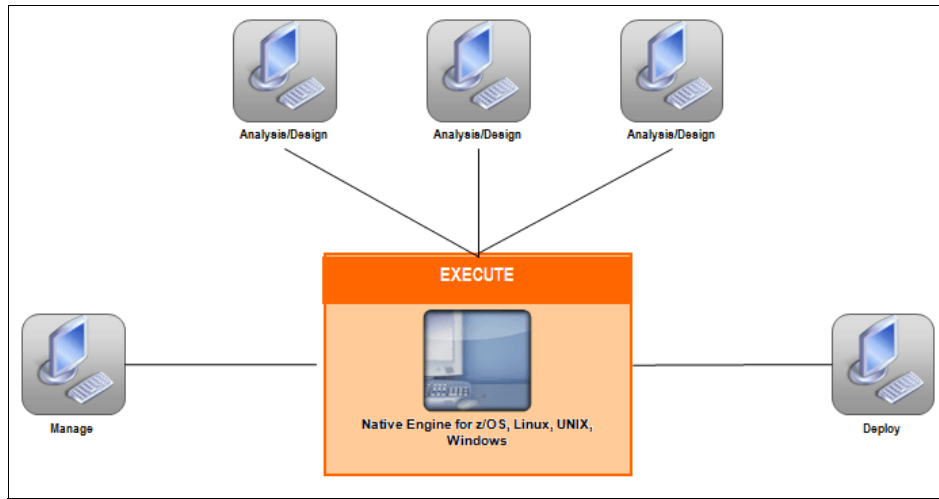


Figure 3-6 Optim components

With data spread over a diverse and sometimes unrelated environment, it is a challenge to manage data across the enterprise. The Optim solutions provide a single scalable integrated data management solution across applications, databases (for example, DB2, IMS, VSAM/SEQ, Adabas, IDS, Oracle, Sybase, SQL Server, and XML), and platforms (for example, Microsoft Windows, UNIX/Linux, z/OS, and IBM system i). The data access strategy of the Optim solutions enables accessing data that resides in any RDBMS and supplies a process to capture a metadata catalog of relational and non-relational data sources.

The Optim data management service enables relational access to archived data using ODBC, JDBC, and SQL-92, as well as XML. The Optim security services provide functional and object security to separate product and data access by roles and responsibilities. Storage management services allow you to manage the life cycle of data in your organization as its value changes over time.

Built upon the foundation of interoperability, integration, and innovation, Optim provides a collaborative design solution that helps you discover, model, relate, and standardize diverse and distributed data assets. Optim solution (Figure 3-7 on page 48) is used to build models and services and provides a common user interface that can be used to support all Optim platforms regardless of the runtime environment. This flexibility allows improved segregation of integrated data management roles and responsibilities.

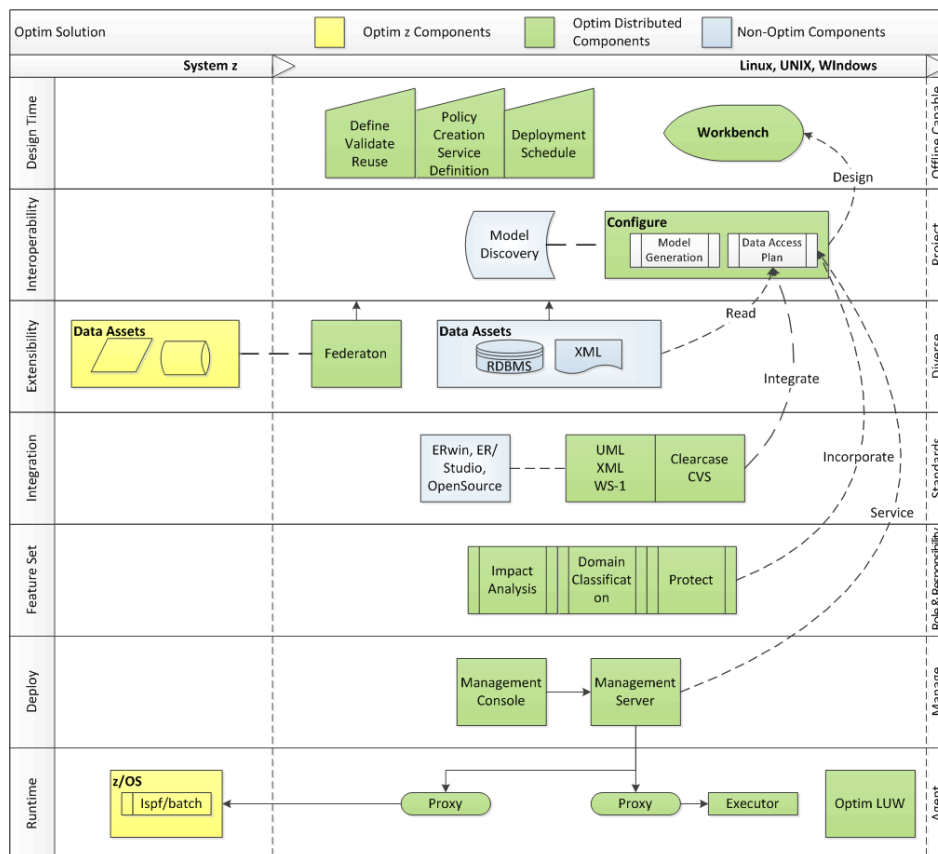


Figure 3-7 Optim solution

With support for data modeling standards, such as Unified Modeling Language (UML), XML, and Web Services Interoperability (WS-I)-compliant web services, Optim solution enables organizations to easily embed data management processes into existing business workflows, such as automation and control of your data growth management solution.

Models, policies, and metadata that are defined using Optim are interchangeable among other InfoSphere products and form the foundation for product integration. This interchangeability provides value at each phase of the data life cycle – from the application and data model design phase, through the development, deployment, and operational management of database applications, through performance and cost optimization, and on to the ultimate retirement of the application and data.

The Optim solutions operate from a common set of “data artifacts” – models, policies, and in certain cases even metadata – that are defined at the beginning of the data life cycle and executed consistently throughout the data life cycle. Therefore, you can establish a common understanding of the business context of your data and the technical context of the data, so that data management practitioners can work consistently across every stage of the life cycle. The resulting fewer errors in interpretation or execution lead to better performance and better business results.

### 3.3.4 Deploy and manage

You can deploy Optim solutions natively on these platforms:

- ▶ System z
- ▶ Linux
- ▶ UNIX
- ▶ Microsoft Windows

Each operating system has its own operational interface (JCL, batch files, and shell scripts) that require separate production automation. To reduce implementation costs, Optim enables you to accelerate the deployment of an Optim solution by removing platform-specific details in a heterogeneous environment. Integration with an enterprise scheduler is possible using a command line, so that you can create a solution that transparently invokes Optim on System z, Microsoft Windows, AIX, Solaris, HP, Linux, Red Hat, and other platforms.

#### Optim manager

Optim manager is a web application that you can use to configure, run, monitor, and manage services. You also use Optim manager to configure the components that are used to run these services.

To test services that you are developing using the designer, access the manager through the designer. You can also publish the service to a registry or export the service to the file system by accessing the manager through the designer.

A *proxy* is a constantly running process that receives service requests from the management server, forwards the service requests for processing, and monitors the running service requests until the service requests are complete.

The component to which the proxy forwards a service request depends on the type of service. The proxy forwards the service request to a service execution component (native Optim engine) on another computer. When the service request is complete, the proxy returns the status of the service request to the manager and the management server.

You can run one or more services by submitting service requests to the management server from the command line (run service tool). Before running a service, you must assign the service to a proxy and a management server using the manager. The proxy and management server must be running to process the service request. You can use the command line to run multiple services deployed to the same management server.

### 3.3.5 Source

The Optim Data Growth solutions on distributed platforms support a variety of heterogeneous data sources, from the most common relational databases to non-relational data. In addition to data sources, the solutions provide a mechanism to include external objects referenced in the database. For example, if a contract database references a PDF file that is stored on the file system, the Optim solutions can capture the PDF (or any other document type) and store it in the archive. Later, during an audit, you can provide the PDF without requiring additional project steps or integrating with an enterprise content management solution.

To support so many software technologies, the Optim solutions have implemented three software stacks that are designed to optimize performance and minimize impact (Figure 3-8 on page 51):

- ▶ Native
- ▶ Federated
- ▶ Non-relational

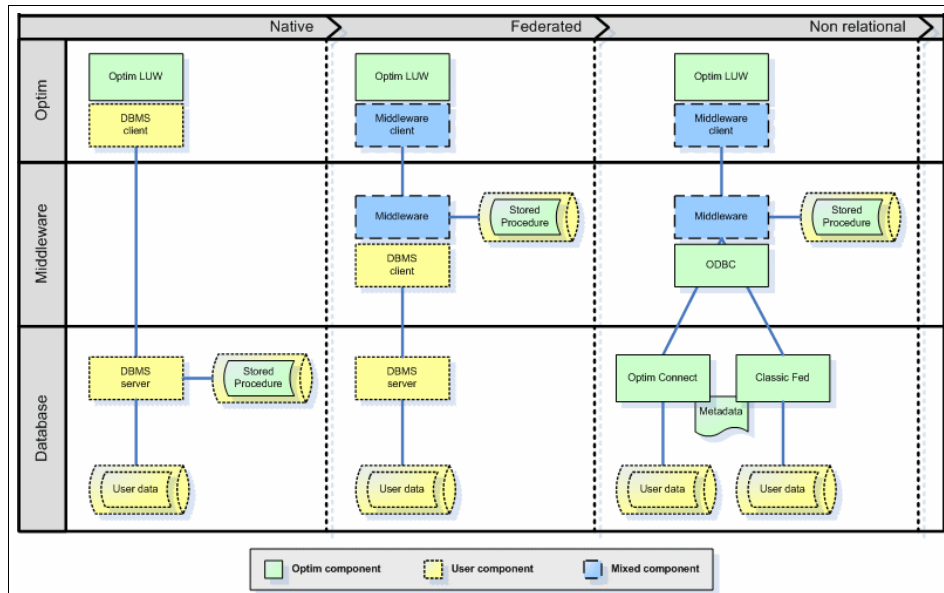


Figure 3-8 Data sources software stack for Optim on distributed platforms

## Native

Native data sources are databases for which the Optim solutions can use the native driver and catalog the required stored procedures. In addition, they can host an instance of the catalog or Optim directory. Through the database client, Optim can access directly a rich set of functionality that it can exploit to attain the highest performance possible.

## Federated

Federated data sources are used when it is not possible to catalog the required stored procedures, in which case, Optim hosts them in middleware or a federation engine (for example, Federation Server or Oracle Database Gateway for ODBC). Because federation involves translation between APIs for the database and the middleware, all native functions might not be exploited. Optim can invoke supported native batch loaders for federated data sources, however.

## Non-relational

Non-relational data sources require a component to manage metadata. This component enables the mapping of non-relational sources to simulate relational sources. For example, a VSAM file with five record types can be translated to simulate five separate tables. Additionally, the data mapping software introduces pseudo keys to ensure that the record types retain a sequencing that is consistent with the original VSAM file. Also, every data source requires a driver to

natively call the non-relational source and map data to relational data. Non-relational data sources must always be accessed using federation.

The Optim solutions support two major middleware configurations:

- ▶ Federation to Optim Connect:
  - Connects to System z data sources, such as VSAM, IMS, SAM, and Adabas
  - Connects to non-relational data sources, such as flat files on distributed platforms
- ▶ Federation to IBM InfoSphere Classic Federation:
  - Connects to System z data sources, such as VSAM, IMS, SAM, Adabas, Datacom, and IDMS

### 3.3.6 Execute

The solution engine performs the archive process so that you can obtain a referentially intact set of data easily from a production database and copy the data to an archive. To streamline your source database, you can delete all or part of the archived data. You can delete data as part of the archive process, or you can defer the deletion and perform it separately, deleting the desired data from the database when your schedule allows. The archived data remains referentially intact and is simple to search or restore, if necessary.

The archive operations performed are table-oriented, which means that all rows for a given table are acquired from the RDBMS and written to the archive file before the next table is processed. The set of tables that participate in the operation are reflected in an access definition created by you. The order of the tables in the requests, as well as information regarding referential integrity constraints (defined in RDBMS or Optim), are contained in the access definition. A single table is identified as the “start table”, which serves as the root of the processing sequence.

The referential integrity constraints are used to determine the order, with respect to the start table, in which tables are visited. During the initial phases of the operation, the validity of the table set is verified by querying the RDBMS catalog. After the validity has been verified, a binary representation of the access definition and all SQL Data Definition Language (DDL) for the archive request is extracted and written to the archive file. In addition to table and index DDL, a variety of related descriptions are written, including elements, such as check constraints, triggers, views, and stored procedures.

The actual archive file is an operating system native file. A storage profile can define another type of persistent store, such as Tivoli Storage Manager or EMC

Centera; therefore, the Optim solutions copy and set retention at the end of the archive operation. After the operation has completed successfully, an entry describing the resulting archive file is placed in the catalog or Optim directory.

The logical sequence for an archive process is to extract all relevant rows from the RDBMS, store them in an archive file, and finally delete the rows from the RDBMS. However, most clients perform these operations as two separate disjointed requests, rather than sequentially under the same controlling request.

During delete processing, the image of the row in the archive file is used to drive the process. By default, the RDBMS row is re-fetched and a comparison is performed. Only RDBMS rows that match the archive are removed. You can optionally bypass this safeguard to expedite the process.

The Optim solutions include the following predefined solutions:

- ▶ Native execution on System z, Linux, UNIX, and Microsoft Windows
- ▶ Functionality required to relationally archive, purge, and restore structured data
- ▶ Derived application-enforced data integrity that augments database referential integrity
- ▶ A data-driven engine that can capture heterogeneous data in the business context
- ▶ Extensive audit trails for purged and restored data
- ▶ A managed and secure catalog of archives
- ▶ Virtual database simulations (collections) to integrate information from multiple archives
- ▶ SQL access to archives and collections using JDBC, ODBC, and OLEDB
- ▶ Templates and workbenches to support packaged applications
- ▶ Retention management
- ▶ Archive storage in SAN, NAS, CAS, WORM, and backup storage devices

Additionally, the Optim solutions simplify the archiving process in these ways:

- ▶ Abstracting data access and navigation
- ▶ Dynamically translating non-relational data into relational data
- ▶ Continuously determining the best access strategy
- ▶ Transparently capitalizing on specific DBMS functionality and APIs
- ▶ Managing multiple cursors
- ▶ Providing restart capability for processes that alter data

- ▶ Automatically capturing the original data model and metadata
- ▶ Dynamically compacting the following information into a compressed and immutable object that is stored on the file system (archive):
  - Metadata
  - Data navigation directives
  - Delete strategy
  - Raw source data (Optim stores data as it has been delivered by the database client software so that type, precision, and code pages information is never lost in the process)
  - Associated documents
  - Security signature
  - Default storage
  - Retention policies
- ▶ Supporting varied reporting requirements by creating indexes that are external to the archives
- ▶ Securing functions, templates, archived data, and the archive access

Optim solutions on distributed platforms and Optim solutions on System z provide the same high-level functionality, but their architectures differ slightly to capitalize on the strengths of each platform. The solutions on System z are architected to use cross-memory services and direct access to DB2.

In contrast, the solutions on distributed platforms use database client software (therefore, the network) to access data. A second difference is that on System z, the solutions capitalize on TSO or the System z job scheduler and dispatcher to allocate and release resources and do not require an internal dispatcher (started task) to archive, purge, and restore data. The solutions on distributed platforms have an internal dispatcher and a daemon to manage and release resources to process multiple requests at the time. Additionally, the distributed platform solutions can fork processes internally and manage sub-requests, network links, and disk I/O asynchronously.

In the case of SQL access to archives, solutions on both platforms share the same multi-threading engine (therefore, solutions on System z require a started task) to handle multiple JDBC, ODBC, or OLEDB connections and user requests.

Both implementations of the solutions share the same core modules. Therefore, the data-driven engine that extracts and restores data produces the same results regardless of the platform. For this reason, the same request in an Optim interchangeability module can be deployed to either distributed platforms or to System z.

## Key components

The InfoSphere Optim Data Growth Solution has the following key components:

- ▶ Native Optim core software in binary format for specific hardware and operating system
- ▶ ODM software in binary format for specific hardware and operating system
- ▶ Packaged application extensions (not covered in this book):
  - Application-specific workbenches to develop and manage solutions
  - Application-specific templates (requests and definitions) to implement solutions
- ▶ Optim catalog (Optim directory), a set of tables in a DBMS used as an extension to the database catalog for relationships, primary keys, and these functions:
  - Record properties on heterogeneous databases (DBAliases)
  - Store project templates (requests and definitions)
  - Provide system configuration (calendars and currencies)
  - Set storage and retention policies (storage profiles)
  - Define virtual databases (collections)
  - Audit data
  - Catalog security (functional, object, and data)
- ▶ Optim database helpers (DBMS stored procedures)
- ▶ Product configuration data
- ▶ Personal configuration data

## Native archive engine for z/OS

The native engine for z/OS is divided in layers and logical sub-components, as shown in Figure 3-9 on page 56.

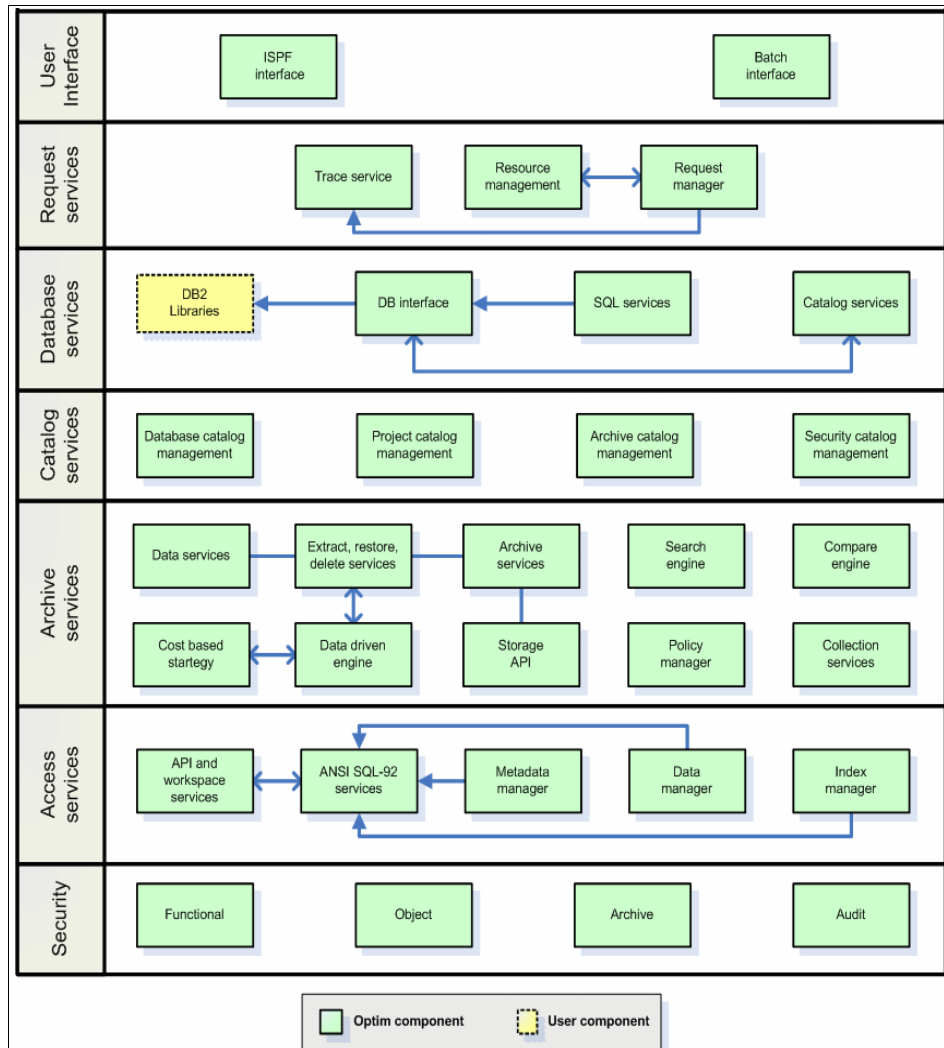


Figure 3-9 Native system z logical sub-components

The following explanations apply to Figure 3-9:

- ▶ **User interface:**
  - *ISPF interface* is used for interactive activities.
  - *Batch interface* is used for scheduled activities.
- ▶ **Request services:**
  - *Request manager* validates and executes a request.
  - *Resource manager* allocates and monitors resource consumption.

- *Trace services* report basic and extended internal information.
- ▶ Database services:
  - *DB2 libraries* (external) interface with the data source.
  - *DB interface* maps to a version-specific DBMS API.
  - *SQL services* format SQL to version-specific SQL.
  - *Catalog services* map to a version-specific DB catalog.
- ▶ Catalog services:
  - *Database catalog management* maps the DB2 catalog to Optim internal format.
  - *Project catalog management* accesses and maintains project catalog integrity.
  - *Archive catalog management* accesses and maintains archive catalog integrity.
  - *Security catalog management* accesses and maintains security catalog integrity.
- ▶ Archive services:
  - *Data services* map internal data requests to database-specific requests.
  - *Data driven engine* provides the unique knowledge, optimization, and data protection required to flawlessly extract, delete, and selectively restore data from thousands of tables across many concurrent heterogeneous databases. It also tracks data across relationship nodes.
  - *Cost-based analysis* is used by the data-driven engine to determine, at run time, the best extract, purge, and restore strategy.
  - *Extract, restore, and delete services* provide process sequencing and timing of the database and system requests required to satisfy the user request and to protect database data from unintended errors. These modules encapsulate many years of best practices and real-life experience on thousands of production databases worldwide.
  - *Archive services* optimize and compact metadata and data into archives. Optionally, archive services create b-tree+ indexes that are external to the archive. Additionally, archive services can split one archive into multiple segments to accommodate storage requirements (for example, DVD size). Additionally, archive services compress and decompress archived data in real time.
  - *Storage API* enables communication with a multitude of storage devices to store, retrieve, and protect data. A litigation hold without an automatic storage lockdown does not prevent operating system-level deletes.

- *Policy manager* implements storage and retention policies for archived data.
- *Retention manager* implements retention policies and provides functionality for Optim and storage litigation holds.
- *Search engine* scans multiple archives to locate data for compliance, audit, or restoration purposes.
- *Compare engine* provides views on data between archives and databases or hybrid data to confirm the existence or changes of the data for compliance, audit, or restoration purposes.
- *Collection services* virtualize multiple archives into a partitioned, database-like environment. Collection services manage metadata differences between data source versions in archives (for example, changed data formats or new columns) and casts data accordingly.
- ▶ Access services:
  - *API and workspace services* provide multi-user support (through the started task), support user access security, and support JDBC, ODBC, and OLEDB servers.
  - *ANSI SQL-92 services* provide direct SQL access to multiple tables in a compressed (or not compressed) archive or collection.
  - *Metadata manager* provides metadata for both single archives and archive collections.
  - *Data manager* provides single-table SQL access to archives and collections.
  - *Index manager* provides statistics and cost analysis to access B-tree+ indexes for single archive or collection.
- ▶ Security:
  - *Functional security* controls access to product features.
  - *Object security* controls access to project templates.
  - *Archive security* controls access to data within an archive (column-level security).
  - *Audit* provides information for processes, requests, and control files to track data purges and restorations.

## **Native archive engine on distributed platforms**

The native archive engine on distributed platforms is divided into layers and logical sub-components, as shown in Figure 3-10 on page 59.

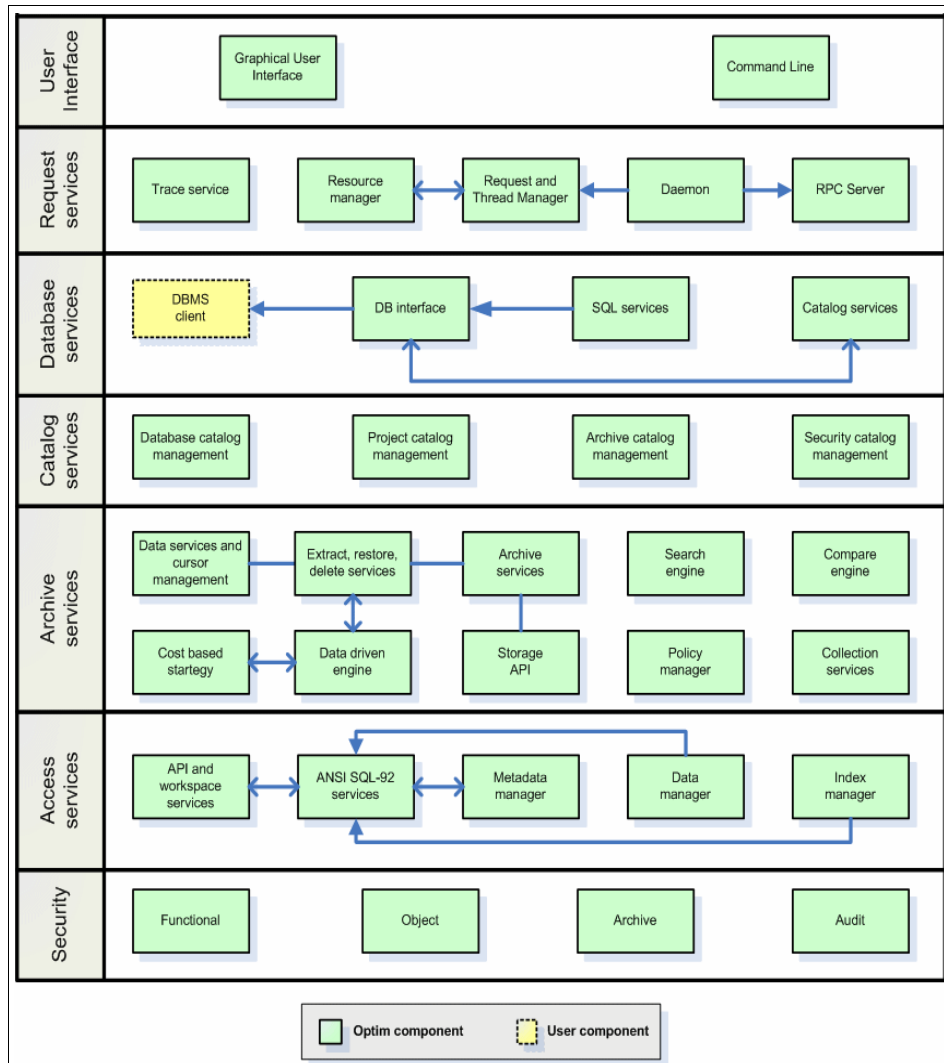


Figure 3-10 Logical sub-components for solutions on distributed platforms

The following explanations apply to Figure 3-10:

- ▶ User interface:
  - *Graphical user interface* is used for interactive activities.
  - *Command line* is used for scheduled activities.
- ▶ Request services:
  - *Daemon* dispatcher is used for a multi-user environment.

- *RPC server* returns various progress status messages to the user interface.
- *Request and thread manager* validates and executes requests in its own thread or task.
- *Resource manager* allocates and monitors resource consumption.
- *Trace services* report basic and extended internal information.
- ▶ Database services:
  - *DBMS client* (external) interfaces with the data source using the native DBMS driver.
  - *DB interface* maps to a version-specific DBMS API.
  - *SQL services* format SQL to version-specific SQL.
  - *Catalog services* map the Optim catalog to the version-specific DBRM catalog.
- ▶ Catalog services:
  - *Database catalog* management maps the DB2 catalog to the Optim internal format.
  - *Project catalog* management accesses and maintains project catalog integrity.
  - *Archive catalog* management accesses and maintains archive catalog integrity.
  - *Security catalog* management accesses and maintains security catalog integrity.
- ▶ Archive services:
  - *Data services and cursor management* map internal data requests to database-specific requests. Additionally, it enables multiple cursors to perform parallel database processing, as well as packing multiple data requests into arrays.
  - *Data driven engine* provides the unique knowledge, optimization, and data protection required to flawlessly extract, delete, and selectively restore data from thousands of tables across many concurrent heterogeneous databases. It also tracks data across relationship nodes.
  - *Cost-based analysis* is used by the data-driven engine to determine, at run time, the best extract, purge, and restoration strategy.
  - *Extract, restore, and delete services* provide process sequencing and timing of the database and system requests that are required to satisfy the user request and protect database data from unintended errors. These

modules encapsulate many years of best practices and real-life experience on thousands of production databases worldwide.

- *Archive services* optimize and compact metadata and data into archives. Optionally, archive services create B-tree+ indexes that are external to the archive. Additionally, archive services can split one archive into multiple segments to accommodate storage requirements (for example, DVD size). Additionally, archive services compress and decompress archived data in real time.
- *Storage API* enables communication with a multitude of storage devices to store, retrieve, and protect data. A litigation hold without automatic storage lockdown does not prevent operating system-level deletes.
- *Policy manager* implements storage and retention policies for archived data.
- *Retention manager* implements retention policies and provides functionality for Optim and the storage litigation hold.
- *Search engine* scans multiple archives to locate data for compliance, audit, or restoration purposes.
- *Compare engine* provides views on data between archives and databases or hybrid data to confirm the existence or changes of the data for compliance, audit, or restoration purpose.
- *Collection services* virtualize multiple archives into a partitioned, database-like environment. Collection services manage metadata differences between data source versions in archives (for example, changed data formats or new columns) and casts data accordingly.
- **Access services:**
  - *API and workspace services* provide multi-user support (through the started task), support user access security, and support JDBC, ODBC, and OLEDB servers.
  - *ANSI SQL-92 services* provide direct SQL access to multiple tables in a compressed (or uncompressed) archive or collection.
  - *Metadata manager* provides metadata for both single archives and archive collections.
  - *Data manager* provides single-table SQL access to archives and collections.
  - *Index manager* provides statistics and cost analysis to access B-tree+ indexes for single archive or collection.
- **Security:**
  - *Functional security* controls access to product features.

- *Object security* controls access to project templates.
- *Archive security* controls access to data within an archive (column-level security).
- *Audit* provides information for processes, requests, and control files to track data purges and restorations.

You can configure your InfoSphere Optim Data Growth Solution to provide all services from one engine. However, typically, organizations separate the creation and restoration of archives from access to archives. Archive creation and restoration are typically batch processes that run on schedules that are defined by operations. Often, the Optim solution is located in the same subnet of the database to minimize latency. Archive access is typically segregated to provide tighter system hardening, security, and monitoring. Optim supports sharing its catalog or Optim directory across the network, so the two engines can share the same archive catalog and security.

### 3.3.7 Target

Archiving data is relatively simpler than choosing where the data will reside. The Optim solutions archive heterogeneous and homogeneous data directly to archive files without intermediate files, tables, or cartesian products. The design enables handling of thousands of tables and relationships in the same way that ten are handled. In one instance, a client extracted, purged, and restored data using nearly 20,000 relationships, and the Optim solution executed the process flawlessly.

#### **Archive file and archive object**

What is an archive file? From the perspective of the operating system, a database and a PDF are simply files. From the point of view of the operator, the database and the PDF differ significantly. An archive file more closely resembles a read-only database than it does a flat file. To demonstrate, you must understand the contents of an archive file.

In an archive file, the Optim solutions store an immutable composite archive object that is logically divided into segments, as shown in Figure 3-11 on page 63.

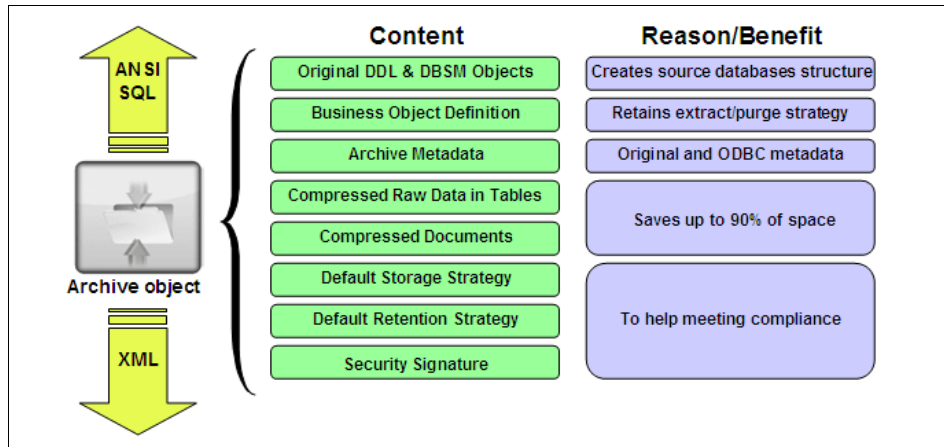


Figure 3-11 Archive object

To further describe the segments in the archive object, as shown in Figure 3-11:

- ▶ **Original DBMS DDL and DBMS objects:** This information enables the solution to re-create the same structure in a homogeneous database or to create a similar structure in an heterogeneous database. Because it is rarely the case that two DBMS support exactly the same data types (for example, DATESTAMP in one database might not be the same data type as in another database), Optim translates from the source format to the target. The end result is that the new structure is similar, but not identical. This information enables the capability, for example, to archive data from VSAM and Teradata and restore it in DB2 and Oracle without requiring advanced DBA skills.
- ▶ **The definition of the complete business object,** including the extract strategy (access definition with tables and relationships) and the purge strategy (delete tables). This information is stored in the archive, both because the data-driven engine requires it for processing and so that Optim can document and re-create the access definition in another Optim directory.
- ▶ **The archive metadata describes both the original data using the original attributes and an ODBC version of the same data,** so that Optim can cast data to a standard format. For example, a VSAM numeric packed field stored as 'x'056C' can be represented in Intel as 'x'3800 or the number 56.
- ▶ **Data that is extracted by the Optim solutions is taken from the DBMS buffer,** without conversion or manipulation (raw data), and is asynchronously compressed. When data is provided to Optim, it is in binary-encrypted format. For compliance issues for which you must perhaps present the data in the original format, the Optim solutions are greatly superior to an XML-based solution.

- ▶ Documents from the file system are automatically retrieved, packed, and compressed like database data. The ODBC metadata is extended and a binary large object (BLOB) is added to the table definition, which enables both restoring the document and accessing from SQL.
- ▶ Default storage strategy at the time that the archive was created. The actual policy is stored in the Optim directory, and a copy is stored permanently with the archive.
- ▶ Default retention strategy at the time that the archive was created. The actual policy is stored in the Optim directory, and a copy is stored permanently with the archive.
- ▶ Security signature indicates whether the archive is protected by a file access definition (FAD) and identifies the corresponding object definition. Additionally, the security signature protects the archive from tampering; a checksum failure will disable the archive.

### **3.3.8 Provide user access**

Archiving might solve your data growth problem by removing history data. Your business continues to need access to that information, perhaps less frequent access than previously; however, the archived data must be accessible. The compressed archives are accessible directly using ANS-SQL-92 through open data manager (ODM). Chapter 7, “Accessing and viewing archived data” on page 325 provides additional information about access to archived data.

#### **Archive access**

Archive access enables reading archive files or collections (virtual database) using ANSI SQL-92 (see Figure 3-12 on page 65) without decompressing or staging. The ODM provides a configurable multi-user security, communication protocol, and administration for this environment.

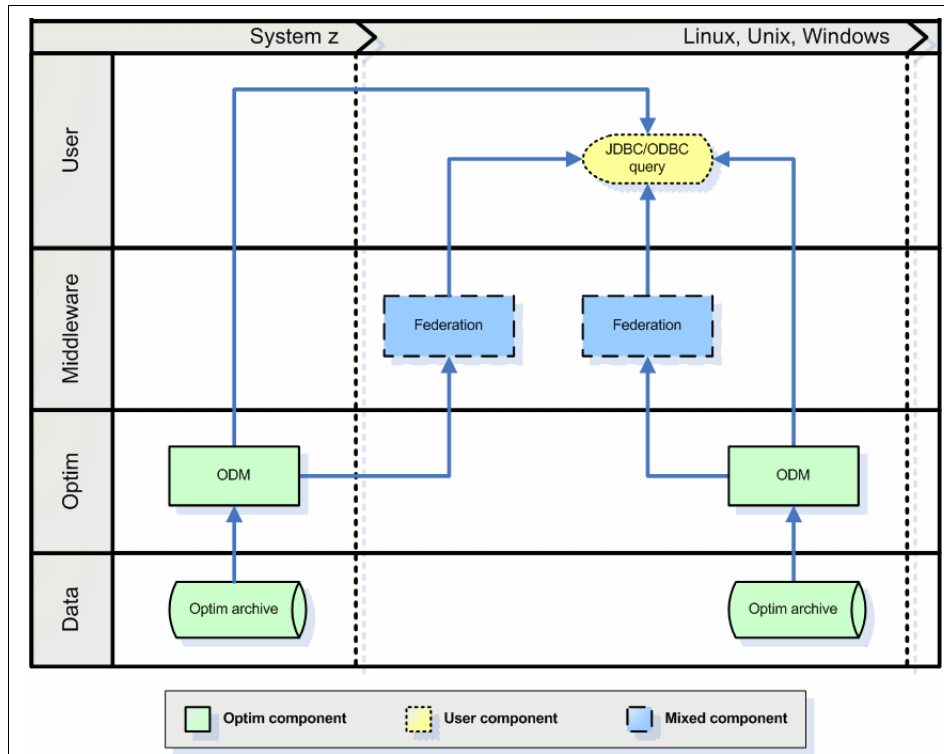


Figure 3-12 Optim archive access

The ODM operates transparently in both System z and distributed platforms using the platform-specific ODM engine. If you want to enable a data growth solution on a distributed platform to store and access archives created on System z (MIPS savings), you can invoke the Optim migration utility and transfer ownership of the archive to the distributed solution.

Regardless of the location of the ODM engine, you can use SQL to access archives from any permissible location in the enterprise. A user on the web can access a System z archive residing on DASD or a distributed archive residing on the SAN.

Most organizations manage data security access using enterprise databases. The ODM can participate as a federated node in distributed queries, so that you can access archives using federation (Federation Server, Oracle Database Gateway, or SQL Server with linked server configuration). In this configuration, you can continue enforcing data access security from your standard DBMS using GRANT and REVOKE statements.

## Open data manager

Two key components, Optim Connect Server and Optim Connect Client, create the foundation for ODM, as shown in Figure 3-13.

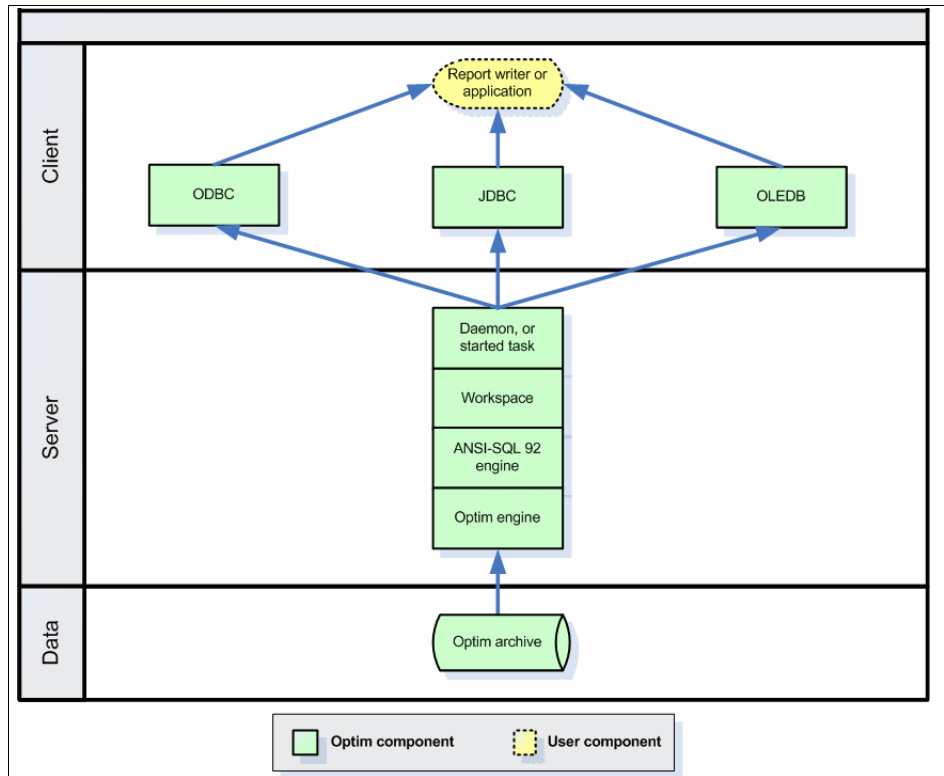


Figure 3-13 Optim open data manager

### Optim Connect Server

Optim Connect Server provides the following functions:

- ▶ Manages communication between client and server components
- ▶ Manages client request threads and the queue on the server side
- ▶ Enforces access-level security through workspace
- ▶ Parses complex SQL statements
- ▶ Optimizes the requests on the basis of query cost statistics
- ▶ Provides archive and collection metadata
- ▶ Provides access to archived data without regard to the storage API
- ▶ Provides query cost statistics to the optimizer for Optim Connect
- ▶ Processes optimized single table access in archives and collections
- ▶ Enforces table and column-level security

## ***Optim Connect Client***

The Optim Connect Client includes these functions:

- ▶ Provides client connectivity (JDBC, ODBC, and OLEDB)
- ▶ Manages protocol between the ODM client and server

## **Open data manager deployment strategy**

To be accessed using ODM, archive files must be registered in an Optim directory and must be accessible from the ODM server on which the data source is defined. A primary ODM server resides on an Optim server with one or more ODM data sources for archive files or archive file collections.

If JDBC is the sole API that is used to access archived data, a direct connection is made to the ODM server. This connection requires the JDBC driver for Optim Connect to access archived data. If ODBC is used to access archived data, the thin ODBC client can connect directly to an ODM server or connect by means of a secondary server.

You can use Optim connect studio, which is included with your ODM installation, to administer the ODM server from a Microsoft Windows machine. You can also perform the same task in all other environments using XML definitions.

## **Archive collection or virtual database**

Archive collections are not physical structures, but logical representations of one or more archives. A collection strategy addresses the issue that archives are immutable, but archiving is a scheduled incremental process. The Optim solutions solve this problem by dynamically assembling archives to resemble database partitions. The partition key is the pseudo column `ARCHIVE_ID`.

The value of `ARCHIVE_ID` (as well as the archive file name) is unique only within an Optim directory. To uniquely identify archives across all Optim directories, you must use the 38-character Globally Unique Identifier (GUID).

Optim does not require that archives have similar structures. For example, if collection “A” represents data archived from 2007 (the year in which a new table was introduced), and collection “B” represents data from 2005 and 2006 (years in which the table did not exist), and the collections are UNIONed together, a single query works without issue. Similarly, if a column in a table, which is in a collection or in multiple collections that are UNIONed together, has various formats (CHAR(30) in certain archive files and CHAR(40) in others), a single query is sufficient.

In Figure 3-14 on page 68, collection “A” represents data that was archived from 2005, 2006, and 2007, and collection “B” represents data from 2005 and 2006. Note that the data from 2005 and 2006 is not replicated; the collections make up a virtual database with logical partitions to describe the same tables in multiple

archives. If in 2007, a new table is introduced, the collection metadata dynamically adds it to the catalog. A more subtle problem arises from adding a column or changing its format between archives. In the first case, the new column is added to the catalog and flagged as null eligible. All other archives (partitions) dynamically show this virtual column. A change to the format of the column is handled by a transparent casting to the appropriate format. For example, in collection “A”, the archive for year 2007 includes a Customer column altered from the original CHAR(30) to CHAR(40). The metadata for collection “A” reflects a CHAR(40) format and data fetched from 2006 and 2005 is automatically recast to the new format. Because collection “B” does not include the year 2007, the metadata still reflects the old CHAR(30). In this example, you can run both versions of the application, in parallel and unchanged (the old version of the application with buffers of 30 bytes and the new version with buffers of 40 bytes), with the versions sharing the same data even though the formats differ.

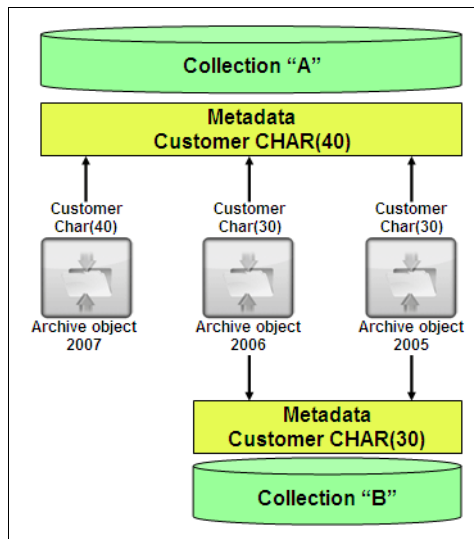


Figure 3-14 Archive collection

## Archive database

The Optim solutions support scenarios for moving historical data to a database. For example, an organization has a requirement to create a reporting database for archived data regarding a medicine that is to be discontinued.

The archiving requirement from the legal department translates into:

*Archive all sales for medicine “z” including all other products sold with that prescription. In addition, archive all other patients and their prescriptions for each doctor prescribing medicine “z.”*

An IBM InfoSphere Optim Data Growth Solution can address the requirement to archive the data in one job, even if business objects for the doctor and related patients are in CA-IDMS and business objects for patient and related sales are in Informix. In turn, storing the data in a reporting database that is neither Informix or CA-IDMS (for example, Sybase) is a simple task for the solution. Attempting the same task in a traditional approach might require temporary tables (not easy in CA-IDMS), file-transferring data from the System z environment, converting EBCDIC to ASCII, and finally, having to extract, transform, and load (ETL) data into Sybase.

In Figure 3-15 on page 70, you see a blueprint of the process and traceability that the Optim solutions offer when populating an archive (or a transaction) database. An archive contains raw data as received from the database. The control files created during purge (if any), from populating a reporting database, as well as Optim compare reports, provide proof that the purged data has been moved to the archive database. The choice between Optim archive and archive database is often about the non-functional aspects of the architecture. For example, the requirement of “DB2, Oracle, and Sybase receive 24x7 operational support, all other databases are supported 8 AM to 8 PM Monday through Friday” dictates an archiving database. A requirement, such as “Our policy is to reduce our storage footprint by 75% or more for any data older than 18 months”, dictates an Optim archive.

An archive database might be more appealing at first because the current support team does not require additional training. Most organizations implement the archiving database for the high-volume “transaction” data; therefore, they capture enough referentially intact data to perform transaction queries in the archiving database. The remaining data is kept in the original system, which means that most queries are federated, because the business transaction data and history transaction data are always in two separate containers.

You also have to add complexity to your project, because you can set your read-only archiving database to be updated when you add new data every week, month, and quarter. So in order to meet governance requirements, such as “data immutability”, you have to introduce controls to ensure that data is not changed during the update cycle. Additionally, you must establish backup procedures for each update cycle. One last consideration is that when data in the archive database expires, you have to purge or move it to another archive mechanism.

When using Optim archives to perform the task of populating the archive database, you have the benefit that data in the archive is in business context (business plus transactions) and it is immutable. So when you populate the archive database, you have all the required information to run queries on the history data without needing the original database.

Backups on the archived data (Optim archive) are automatically performed by Optim using the storage profile. If you suspect that data has been altered in the archive database during the update cycle, you can run an Optim compare to validate and identify changes. When it is time to roll out data from the archive database, you can use the original Optim archive to purge the corresponding data.

If you use Optim archives only, you get all the benefits of the archive database without the added complexity of security, backups, and purge processes.

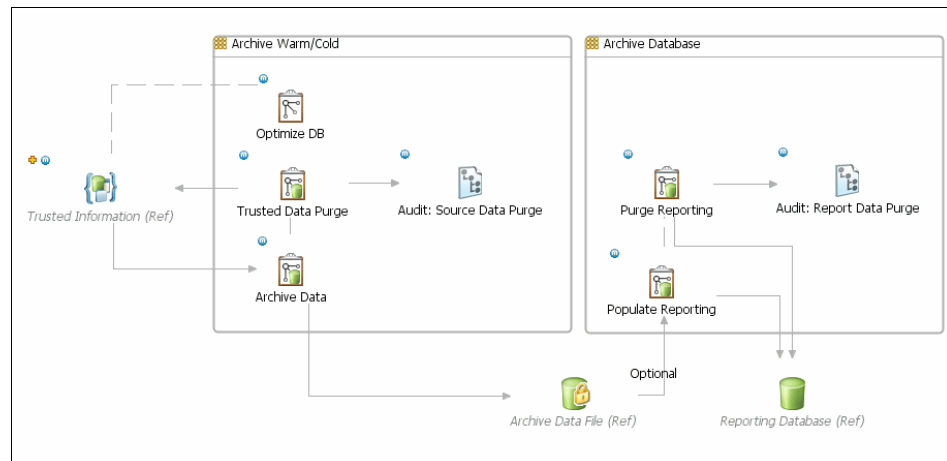


Figure 3-15 Traceability from trusted information to archive database

## Native application access and federation of archived data

The integration of a managed data process with an existing application can be a daunting project. Fortunately, in the relational world, it is possible to integrate two or more data sources without altering the application. *Native application access* (NAA) and one type of federation of archived data implement a shadow schema technology. For native application access, the solution installs and configures an alternate schema that references all objects in the production schema. It also sets up a schema in a remote database to store the tables with archived data. The alternate schema is then configured to look at both the archived tables in the remote schema and the tables in the production schema. For federation of archived data with active data, the solution points to archive files or an archive collection, rather than a database.

This shadow technology is a common practice that is automated with native application access. Figure 3-16 on page 71 illustrates both native application access (the history database) and federation (the collection). Either one is set up with the history data and the schema is created with synonyms and union views

to either the history database (native application access) or the collection (federation). The synonyms transparently inform the DBMS that when referencing production tables with history, the union view must run the query.

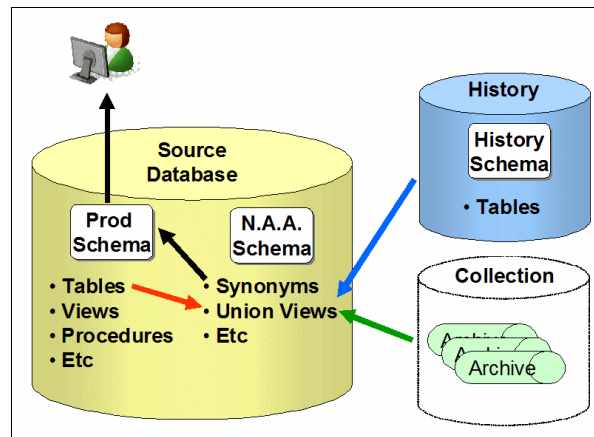


Figure 3-16 Native application and federated access

### 3.3.9 Storage policy

Archives are retained and placed in the appropriate storage class (SAN/NAS, WORM, and BCKUP) according to a defined policy.

Figure 3-17 on page 72 depicts the Optim storage and retention management.

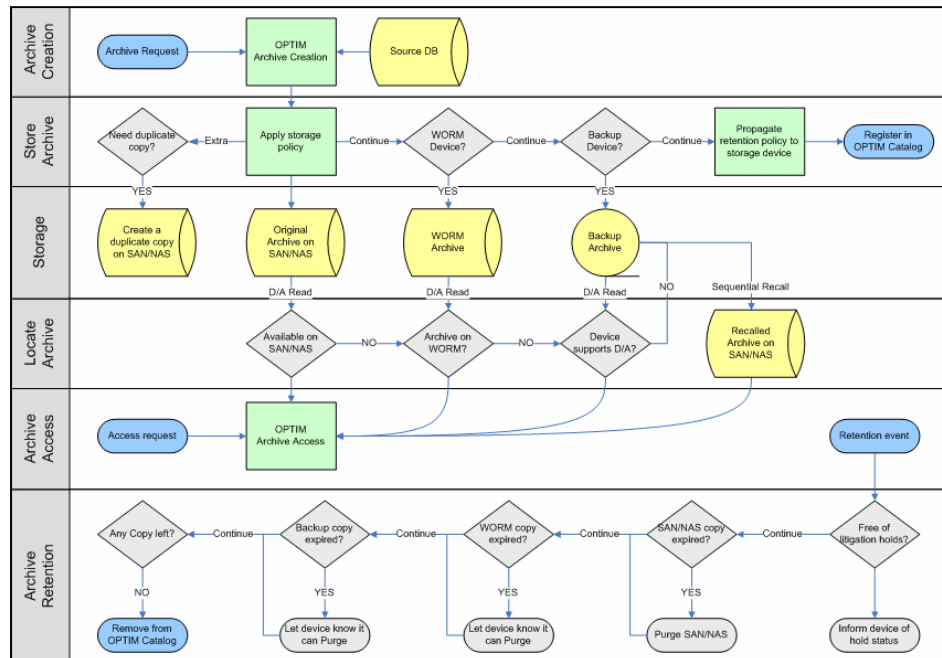


Figure 3-17 Storage and retention policies

You can instruct your Optim solution to automatically create a duplicate copy of the archive to the disaster recovery center, and you can save a secondary copy to a backup device and place it on a WORM device.

Fixed media might be the best choice for storage if archive files must be accessed online or if quick response time is the primary consideration. You can also choose to use cheaper, perhaps less accessible media or a hierarchical storage management (HSM) system to minimize the use of tier-1 storage. When data is stored in HSM (for example, Tivoli/HSM or Symantec Enterprise Vault), the solution recognizes when a stub has replaced the original archive and can mark the archive offline or recall it when needed for processing. If Tivoli is the backup device, you can also specify the pool in which the archive is originally placed.

If the device supports retention and security profiles (for example, EMC Centera), you can request adherence to a specific profile. A retention policy can be set for each of the three storage classes. Additionally, you can set a cache mechanism to optimize an archive that is placed on a removable media.

When a litigation hold is issued, the solution communicates with the device (if APIs are available) and prevents the accidental purge of the archive. Internally, the Optim solutions extend the retention period for as long as the user needs to archived data.

### **3.3.10 Security**

The security that is provided by the Optim solutions is extensive and enables groups of users to be secured so that you can prescribe the activities that they can perform. The Optim solutions can interface externally with Lightweight Directory Access Protocol (LDAP), and they also can enforce their own granular security and limit access to product functions, project objects or templates, and data at table and column levels.

#### **Types of security in solutions**

At the most general level, functional security allows you to control user access to the interface for functions that are provided by the Optim solutions. For example, for a specialized administrator role that is intended to create process requests and objects (templates) required to run these requests, you can grant unlimited access to functions. For members of a role intended only to run the predefined process requests, however, you can grant more limited access to functions.

As a second example, you can use functional security to grant access to the solution-specific editors (archive requests, delete requests, and restore requests), as well as to the archive maintenance utilities, for members of a specialized archive role while denying access to the same editors for developers that use Optim functions to create test data.

Additionally, you might define object security to allow members of a role to read and run, but not edit, a specific archive request (a template). Also, you can configure Optim to secure objects automatically so that all objects receive a default security signature.

Archive file security allows you to control access to data in archive files. For example, you might use archive file security to prevent access to data in a specific table or column for most users while granting access to members of selected roles for the same data.

#### **Audit**

An audit mechanism tracks user activity by creating XML records in a database. This mechanism enables integrating the information that is provided with an external system.

Additionally, key database activities, such as data purge and restoration, generate a log that is used both to audit the process and enable controlling restartability.

### 3.4 Complete business object

A *complete business object* is a set of related data. This concept is interpreted differently based on role. In Figure 3-18, you can see the challenges that confront a data administrator or a database administrator, because business data is often mixed with application data (such as configuration, security, and customizations), projecting a complex data model.

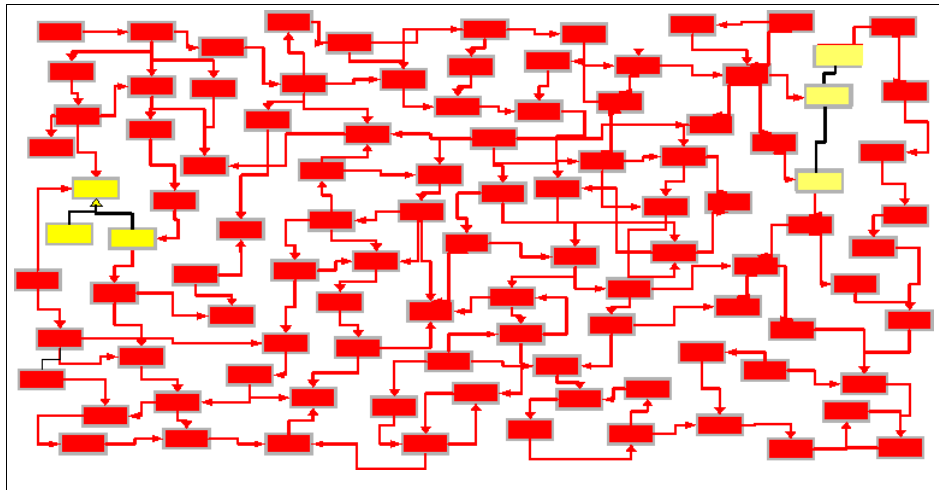


Figure 3-18 Complex data model before the analysis

The data-driven discovery component can help break a complex data model down into manageable data objects that can be assembled in one complete business object.

Figure 3-19 on page 75 illustrates this concept.

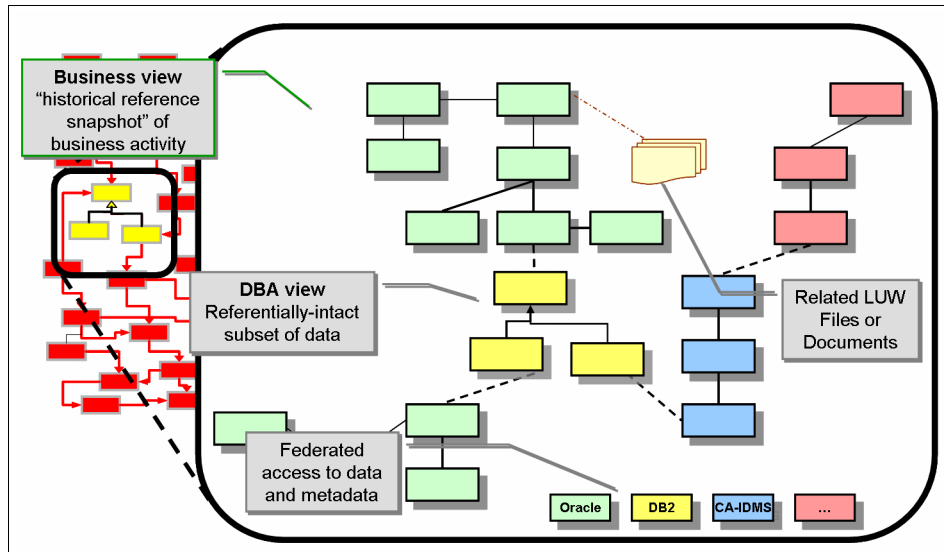


Figure 3-19 Complete business object

## DBA view

A DBA sees a business object as a referentially intact customer, order, or product. Therefore, when a DBA plans to archive, the major concern is to subset data following the data model. Referential integrity rules, if any, help both in the analysis and delete processes.

## Business view

A business person is not concerned with the data model, but with the concept that they support to run the business. The business user has to access (data in business context) a customer order with products and suppliers.

## Federated access

After we expand from business object to complete business object, we must consider that the information resides in many heterogeneous databases, which does not translate into project complexity. This design is a smart infrastructure that enables extending the reach to other systems.

## Related documents

Business data is not limited to databases. Documents that are referenced in the database are integral parts of the complete business object and must be included like any other piece of information.

### 3.4.1 Elements of a complete business object

Optim enables building a complete business object using a series of specialized elements that instruct the data-driven engine when extracting, purging, and restoring user data.

#### Optim directory

The Optim directory is the catalog that the Optim solutions use to catalog and track the archive. It is also where the project template is stored. This directory entry identifies the file by name, group, description, and the tables from which data in the file was archived. The entry also records the user account under which the archive process was initiated and the date that the process was executed. Using one or more filters or patterns for this information, you can search for specific segments of data in two or three steps.

#### Archive file

An *archive file* provides the safe, unmodified, storage of archived data on disk, tape, or optical device. The archive file contains a copy of the segment of data that is described by the access definition that is used in the archive process and a copy of the actual access definition. Although the archive file has no hierarchical structure, it includes all information required to re-create the structure of the original database (copies of object definitions, such as tables, views, relationships, primary keys, and so on). Information about the archived data, including all DBMS and Optim relationship definitions, whether or not used in the archive process, are stored in the archive file. All other DBMS and Optim object definitions that are required to reproduce the database or support applications are also stored in the archive file.

**Relationships:** Only a relationship in the archive file can be used to manipulate data in the file. Thus, a relationship that is defined to the DBMS or Optim directory after data is archived cannot be used to select data for browsing or restoration.

An archive file can be stored on disk or tape or moved to other media using HSM or other means. Copies of archive files can be stored off-site for disaster recovery or exported to other locations, for example, to another subsystem, network, site, or Optim directory. You can search, browse, or restore data in an archive file. (The archive file can also be used with the Optim solutions to insert data from an archive file into a database and to compare archived data to data in a database or in another archive file.)

## Access definition

The principal object that the Optim Data Growth solutions use to archive data is the *access definition* (AD), which governs the overall archive process. An access definition identifies the tables, relationship traversals, and selection criteria for the data that is copied to an archive file, and it can identify tables from which archived data is deleted. It also indicates columns for indexes, which are stored in an index file that is associated with the archive file. It indicates any archive actions, which are custom logic, in the form of an SQL statement or call to a stored procedure, that is executed at a predefined point in an archive or restore process. You use an access definition to perform these tasks:

- ▶ Identify the tables from which data is archived.
- ▶ Identify tables from which data is deleted after archiving.
- ▶ Define criteria for the business objects to be archived using these tools:
  - Selection criteria based on the age of the data or values in one or more columns
  - A manually selected or application-selected list of start table rows (row list or point-and-shoot file)
- ▶ Select relationships to be traversed and the direction of traversal when archiving the data.
- ▶ Set up indexes to be created when data is archived and later used to locate data for restoration or review.
- ▶ Establish archive actions to be executed when data is archived or restored.

## Database alias

The *database alias* (DB alias) represents connectivity to a data source. It defines the database clients that must be invoked, the user account for the connection, and the default size for new database tables and indexes (when needed). You can have multiple database aliases in a complete business object and the Optim solutions take care of invoking the appropriate client, determine database statistics for access strategy, and adjust SQL to the appropriate dialect. If your source is non-relational, the Optim solution stack translates the record layout into a relational model, shielding you from the physical structure of the source data. You might have started with a VSAM file containing five record types, but you will work with a Customers table that is related to the Orders table and so on with virtual referential integrity.

## Start table

The *start table* is the place at which the Optim solutions begin the process of selecting data for processing. The solutions do not impose any particular point in the data model to be the initial table for an archive. You can select the top parent,

the bottom children, or any table in between. The data-driven engine takes care of navigating the databases.

Generally, the table that includes the data that is used as the basis for archiving is designated as the start table. For example, if you want to archive orders shipped before a specific date, you designate the Orders table as the start table. Similarly, if you want to archive all orders for a discontinued item, you designate Items as the start table.

**Start tables:** The start table is also important when searching or restoring archived data. The Optim solutions allow you to change the start table for these processes.

## Selection criteria

Selection criteria define the filtering rules for the business object. You can pick any statement that a WHERE clause in SQL allows you to have. For example, you can pick a date range to archive data for the year 2005.

The criteria that are used vary according to the data and archiving needs. General information about transactional data is the basis for the criteria that are used to select data for archiving. You can use one or a combination of the following types of criteria to select the data in the listed tables for archiving:

- ▶ Use date criteria to select data according to age, as determined by values in one or more date columns. With the Optim solutions, you can specify the date criteria once, indicating an interval in terms of years, months, weeks, and days. Each time that data is archived, this interval is subtracted from the current date to calculate a date in the past. All data older than the calculated date is archived. For example, if the interval is two years, all data older than two years is archived each time that the process is run.
- ▶ Use SQL LIKE syntax to select data according to values in one or more columns. For example, you can select as criteria to archive related rows that have an “inactive” indicator in a column. You can combine selection criteria with date criteria to establish categories of data, for example, to create separate archive files by geographic area or product category to send to remote sites.
- ▶ Select start table rows manually or by using an application. This feature is useful if the data lacks an appropriate date field or active and inactive indicator. For example, an accounting application might generate a file of identifiers for fully amortized equipment. This file can then be used to select related rows from inventory for archiving.

## Related tables

The data-driven engine navigates the related tables using the relationships present in the logical data model. The data that is selected from the start table determines the data from the next related table that is required. The “Show Steps” feature provides a display of how the solutions drive the process.

## Relationships

To select a segment of related data, the Optim solutions use all relevant relationships for tables in the complete business object. Although not all available relationships for a pair of tables are required to be used to select data for archiving, all are copied to the archive file and are available to search, browse, or restore the archived data.

You can create your own relationships at any time with no need to change the database. The solutions also support relationships that are driven by the data content. For these Optim relationships, a number of database restrictions are relaxed:

- ▶ Primary keys and foreign keys are not required.
- ▶ Corresponding columns are not required to be identical, although they must be compatible.
- ▶ Even though at least one of a pair of columns must be referenced by column name, you can use an expression to evaluate or define the value in the second column. Expressions can include string literals, numeric constants, NULL, concatenated values, and substrings of values in a column.

You can use the more flexible or “extended” Optim relationships to emulate implicit, or application-managed, relationships in your database. For example, if your application uses a “National\_Rate” table for all addresses in the US and “International\_Rate” for all other addresses, you can set up an Optim relationship that is based on “Country” to extract data from one table or the other. Using substring and concatenation, you can address even more complex extended relationships.

In addition, an Optim relationship can be stored as a generic relationship, so that it can be used for one or more pairs’ names and attributes that have separate Creator IDs. Generic relationships are useful when several sets of tables differ only by Creator ID, for example, in an organization where each department might use a separate copy of the same production tables. Each set of tables can be distinguished by the Creator ID. Using generic relationships, you define one set of relationships that applies to all sets of tables. Also, when a set of these tables is added, the generic relationships automatically apply.

This flexibility allows you to select sets of data to be archived or restored that are related in the same way as in the production environment, as in a data-driven relationship that exists when a row in the parent table is related to rows in one of several child tables on the basis of the value in a particular column. For example, you can relate sales to foreign rates or national rates based on the content of the data.

One extremely important role of Optim relationships is to connect homogeneous and heterogeneous data sources. The Optim solutions use DB alias information on each end of the relationship to determine at run time the best strategy to physically process related business data.

### **Reference tables**

The Optim solutions allow you to include additional information in the archive tables outside the data-driven process. These tables can have associated selection criteria, but no relationship is required for the reference data to be archived or restored.

### **Navigation rules and governors**

The Optim solutions naturally support waterfall access in relationships. Therefore, using the sample data, when you archive rows from the Sales table, related Customers, Orders, and Details tables are also archived, automatically. To retrieve a complete set of related data, you can also reverse the flow and, for each Details table, archive the related inventory Items table (a parent table).

### **External file association**

When one or more columns are pointers that reference external files (attachments), the Optim solutions can copy the external files and store them in the archive. Additionally, a large object (LOB) column is added to the archive model, so that you can access it without restoring the document to the file system. You can also restore the document back to the same or another path or folder.

### **Delete rules**

You can specify the tables that you want to include in the purge process. For example, you might have archived Sales, Customers, Orders, and Details for year 2005. The customer information must be retained in the database for an indefinite period to accommodate recent and future orders, and item information is required to provide the unit price for current orders. Thus, only the archived rows from the Orders and Details tables will be deleted to manage data growth.

## Archive actions

An *archive action* is a custom SQL statement or call to a stored procedure that can be executed at a selected action phase of an archive, delete, or restore process, for example, at the beginning of an extract, at the end of a delete, before processing a table, or after processing a row. These archive actions enable adjustments to database data to meet specific application requirements, for example, creating a summary record for all sales archived from the year 2006. You can also use an archive action to store archive metadata and results in your own project table. Another use for an archive action is to invoke database functions or a utility to perform maintenance, such as dropping an index or reorganizing a table.

An action for a restore process can be included in the access definition that is used to create an archive file and, if so, used with the table map during restoration. At run time, you can use, override, or disarm the action.

## Data model changes

Over time, changes in the data model for a database are typical. Tables or columns can be added, deleted, or renamed. Information from a column might be divided into two or more columns, or multiple columns combined into one. A column might be required to carry more data than originally contemplated, as when the US Postal Service changed the ZIP Code from 5 digits to the current, 9-digit, 10-character (including the hyphen) ZIP+4 code.

*Data model changes* cause the structure of the current production database to differ from the production database that was in effect at the time that the data was archived. The older the archived data, the more likely it is that the production data model has diverged from that in use at the time the data was archived. This divergence becomes an issue when archived data must be restored, to either the production or a temporary database, for use with application programs written for the current data model. When data is restored, table maps and column maps are used to accommodate these variations in the data model, allowing you to restore data that no longer matches the current data model.

## Table map

A *table map* matches tables in the archive file to tables in the destination database. Using a table map, you can exclude individual tables from restoration or restore to tables with separate names. A table map also provides parameters for archive actions at predefined points in a restore process.

## Column map

A *column map* identifies and matches columns in a pair of tables. Use a column map when column names or attributes are dissimilar or data is to be transformed.

### 3.4.2 Data-driven processing examples

The start table and navigation rules in the access definition determine how the data-driven engine navigates your data sources. Figure 3-20 shows the data model that is used in these examples.

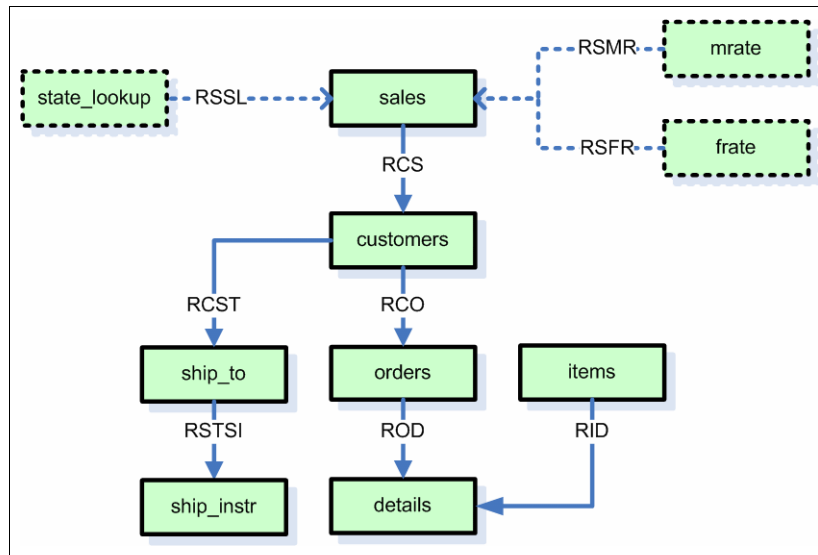


Figure 3-20 Data model

#### Top-down example

You are required to archive all activities in New Jersey for sales people, because retail sales in the state have been discontinued. To do this task, you must extract sales people (Sales) in the New Jersey territory with their related Customers, Orders, and Details. The start table Sales is the top-level parent in the model, so the Optim solution carries this requirement out in a waterfall fashion by performing these steps:

1. Select Sales rows that match the criteria (people in the NJ territory).
2. Following the relationship RCS (Figure 3-20) from parent to child, select Customers rows that are children of the Sales rows selected in step 1.
3. Following the relationship RCO from parent to child, select Orders rows that are children of the Customers rows selected in step 2.
4. Following the relationship ROD from parent to child, select Details rows that are children of the Orders rows selected in step 3.

The steps in this example and the access definition that guides them remain the same whether or not one of these situations occurs:

- ▶ All data is extracted from a single database.
- ▶ Sales are extracted from an Oracle database, Customers are extracted from a DB2 database and Orders, and Details are in a VSAM file with two record types.

### Bottom-up example

To prepare for a recall and possible litigation, you are required to retrieve all archived transactions that include a specific item that is defective. To do this task, you must find the complete business objects for all archived Orders for Item “x” that were placed in the years 2006 through 2008. The business objects were archived monthly, on the basis of data criteria on the Orders table and placed in collections, one for each year. To find the required Orders, the Optim solution searches the collections and extracts the archived Orders, using Details as the start table, by following these steps:

1. Select Details rows that match the criteria on Item\_ID for product “x”.
2. Following the relationship ROD (Figure 3-20 on page 82) from child to parent, select Orders rows that are parents of the Details rows found in Step 1.
3. Following the relationship RCO from child to parent, select Customers rows that are parents of the Orders rows found in Step 2.
4. Following the relationship RCS from child to parent, select Sales rows that are parents of Customers rows found in Step 3.

For a recall, one use of the business objects extracted from the archives in this example might be to insert or load them into a database and update with current Customers and Sales data to have current contact and sales information for customers who purchased the defective item.

**Additional conditions:** Similar to the top-down example, the steps in this example remain the same whether or not any of these conditions are true:

- ▶ All data is extracted from archives or collections of archives.
- ▶ All data is extracted from a single database.
- ▶ Sales are extracted from an Oracle database, Customers are extracted from a DB2 database and Orders, and Details are in a VSAM file with two record types.

## **An example using a data-driven relationship**

You must archive business objects for all Orders completed more than five years ago for which charges were made to a specific credit card vendor. The business object includes Orders, Details, Items, Customers, Sales, and international and US commission rates from the Mrate and Frate tables (Figure 3-20 on page 82). To find the needed Orders, the Optim solution searches and extracts the business objects, using Customers, in the center of the schema, as the start table, by performing these steps:

1. Select the Customers rows that match the criteria on Creditcard\_type for credit card “XX”.
2. Following the relationship RCO from parent to child, select Orders rows that are children of the Customers rows selected in Step 1 and that also match the criteria on Order\_ship\_date for orders older than five years.
3. Following the relationship ROD from parent to child, select Details rows that are children of the Orders rows selected in Step 2.
4. Following the relationship RCS from child to parent, select Sales rows for Customers rows selected in Step 1.
5. Following the relationship RSMR from child to parent, select Mrate rows for Sales rows selected in Step 4 that include a US territory, and following the relationship RSFR from child to parent, select Frate rows for Sales rows selected in Step 4 that include an international territory.

### ***There is no typical access definition***

No access definition can be considered typical. We have seen access definitions for processing one table and access definitions for processing hundreds of tables. Perhaps most access definitions process between 30 and 100 tables; however, it is not uncommon to see access definitions for a thousand or more tables. The largest number of relationships that a customer has placed in production nears 20,000. Now, the 20,000 relationship access definition was a result of limited analysis and understanding of the data model. The customer in question was a large worldwide organization that stored extreme volumes of data and transactions.

If you imagine a representation of a data model that includes 20,000 relationships, you might picture a large wall covered with labels in extremely small fonts. Surprisingly, the actual model was accommodated nicely on a regular computer printer page and you were able to read every label, unaided. Only 30 or so elements were on the page. When asked how the small number of elements mapped to 20,000 relationships, the customer explained that it was a large company with a lot of data. The Customers table is actually 26 instances of the database, one for each letter of the alphabet, with the application choosing an instance on the basis of the first letter of the last name. The access definition

had instructed the Optim solution to use all relationships in all directions, generating 20,000 permutations. After adding several data-driven relationships (using the first letter in the name), the access definition required fewer than 800 relationships. Today, by using the data-driven discovery component, the data-driven relationships can be discovered during the analysis and not in production.

## 3.5 Extract, store, and restore processing

The Optim solutions provide most of the required services to implement a data growth management project:

- ▶ Extract and archive data
- ▶ Purge source data
- ▶ Compare data
- ▶ Selectively restore archived data
- ▶ Maintain archives
- ▶ Access archives

### Extract and archive data

An archive request enables you to extract a subset of data using an access definition that describes the complete business object and to store the extracted data in an archive file. You control the boundaries of the process by defining these elements:

- ▶ Server on which to run the request
- ▶ Variables to be used during processing
- ▶ Types of objects to be captured from the source database
- ▶ Archive and index files to create and compression options for those files
- ▶ Database resource utilization policy (concurrent connections)
- ▶ Access definition with complete business object (a copy is stored in the archive)
- ▶ Storage and retention policies
- ▶ Security for archived tables and columns (file access definition)
- ▶ Collections that are used to virtualize the archive
- ▶ E-mail address for notification in case of success or failure
- ▶ Delete strategy and timing

## **Purge source data**

The delete request enables you to delete data from source databases using instructions in the access definition and data in an archive file. You control the boundaries of the process by defining these elements:

- ▶ Server on which to run the request
- ▶ Control file that is used for restarting the process and for audit purposes
- ▶ Database resource utilization policy (concurrent connections and table locking)
- ▶ Level of validation for the delete process (validate data model and compare data)

## **Compare data**

A compare request enables you to analyze the difference between two sets of data and generate an audit record or control file and a report. You control the boundaries of the process by defining these elements:

- ▶ Server on which to run the request
- ▶ Control file for audit purposes
- ▶ Sources for comparison (tables, access definitions, or archives)
- ▶ Report options

## **Selectively restore data**

The restore request enables you to insert or load a subset of data from one or more archives. When the data model differs from that in the source (perhaps the archive was taken from an earlier version of your application), you can also provide additional instructions in the form of table and column maps. You control the boundaries of the process by defining these elements:

- ▶ Server on which to run the request
- ▶ Archives to be used for the restore
- ▶ Selection criteria for the subset of data to restore
- ▶ Method of restoring (load or insert)

## **Search and restore**

This restore mode enables you to select archives on the basis of data content. The Optim solutions reference the catalog or Optim directory to identify candidate archives and search archive content to list archives before executing the process as in a selective restore process.

## **Archive maintenance**

The Optim solutions have built-in utilities to migrate, subset, register, re-index, alter retention policies, view audit trail records, validate archives, and restart processes.

## **Engine configuration**

You can configure the Optim solutions to provide all services from one engine, but typically, organizations separate the creation and restoration of archives from access to archives. Archive creation and restore processes are typically run in batch on schedules that are defined by operations. Often, the solution is hosted in the same subnet with the database to attain minimum latency. Access to archives is typically segregated to provide tighter system hardening, security, and monitoring. The Optim solutions support sharing a catalog or Optim directory across the network, so that two or more solution engines can share a single archive catalog and security

## **Archive creation and restore**

In a z/OS production environment (see Figure 3-21 on page 88), the Optim engine can be invoked from the z/OS scheduler or from a scheduler on a distributed platform. Regardless of the invocation, the Optim solution can run the archive service (extract and purge or SQL select and delete) by linking to DB2 in the same or, by means of distributed data facility (DDF), a separate logical partition (LPAR). Depending on the user options, the restore process either uses an SQL insert or invokes the DB2 loader.

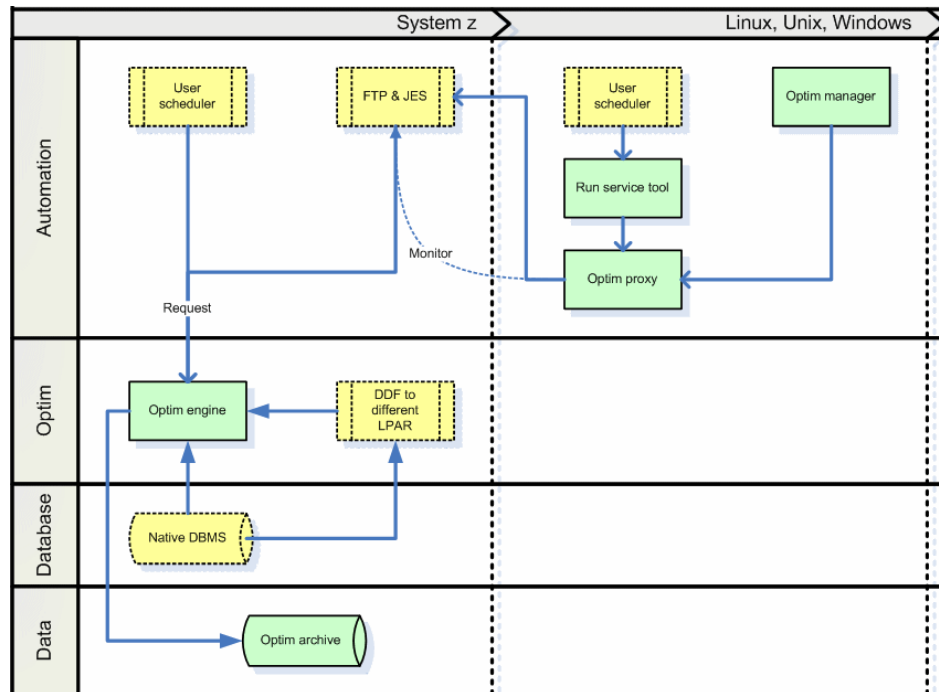


Figure 3-21 Archive creation and restore in System z

A production deployment in a distributed system (Figure 3-22 on page 89) invokes the Optim engine directly from the scheduler using a platform-specific script or invoking an Optim runtime service. The engine can access the heterogeneous data in native, federated, or non-relational mode. The solution can run the archive service (extract and purge or SQL select and delete) by using the DBMS client software. Options permit the restore process to use either SQL insert or to invoke the native database loader.

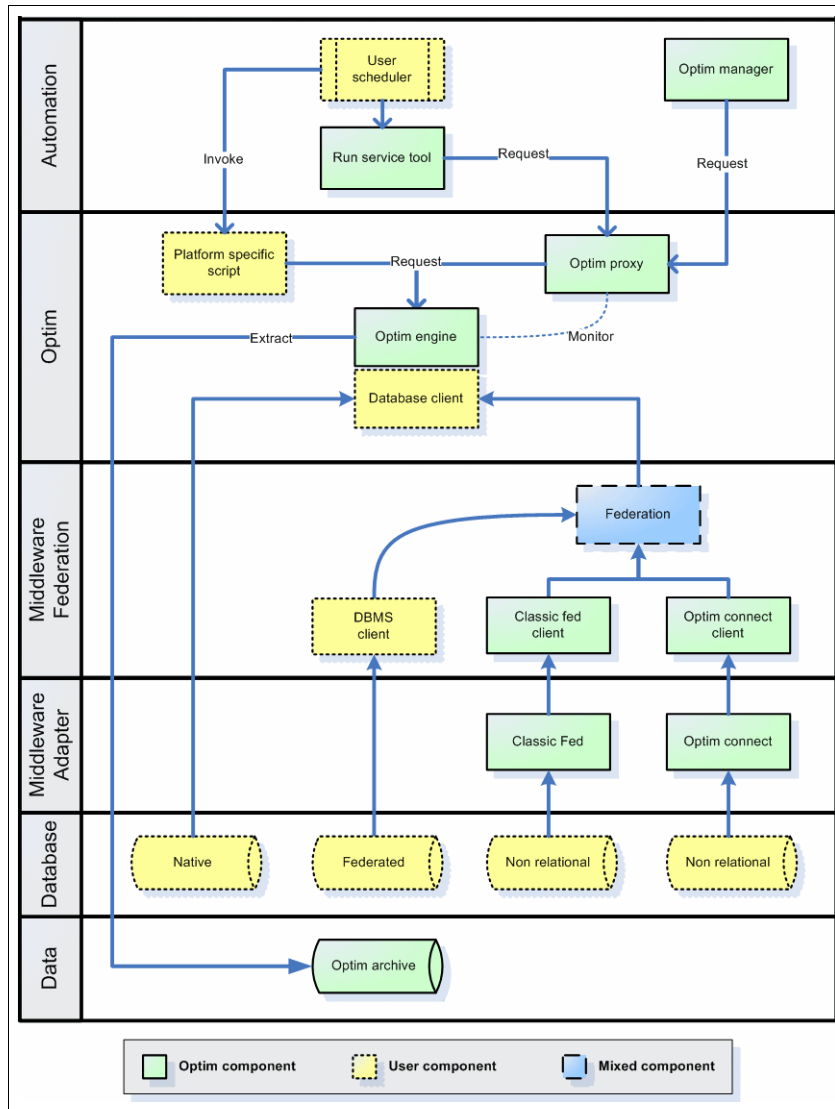


Figure 3-22 Archive creation and restoration in a distributed system

Regardless of the platform, the task of creating archives must be performed by extracting data as quickly as possible. The Optim solutions perform compression in a separate thread so that, in a multi-core environment, it is an asynchronous process. To avoid overloading the servers, the solutions enable the deferral of compression until after all data is extracted. This feature enables earlier disconnection from the source database and continuation of the process while disconnected from the source database.

The primary goal of purge or restore is to protect the database from accidental data inconsistencies and logical errors. The compare engine detects when the original data has been altered and it can either fail the process, ignore the row, or ignore the changes. Additionally, the data-driven engine determines the sequence of deletion based on the complete business object and database referential integrity.

## 3.6 Universal access

Figure 3-23 provides a high-level representation of the extensive support for universal access that is provided by the Optim solutions.

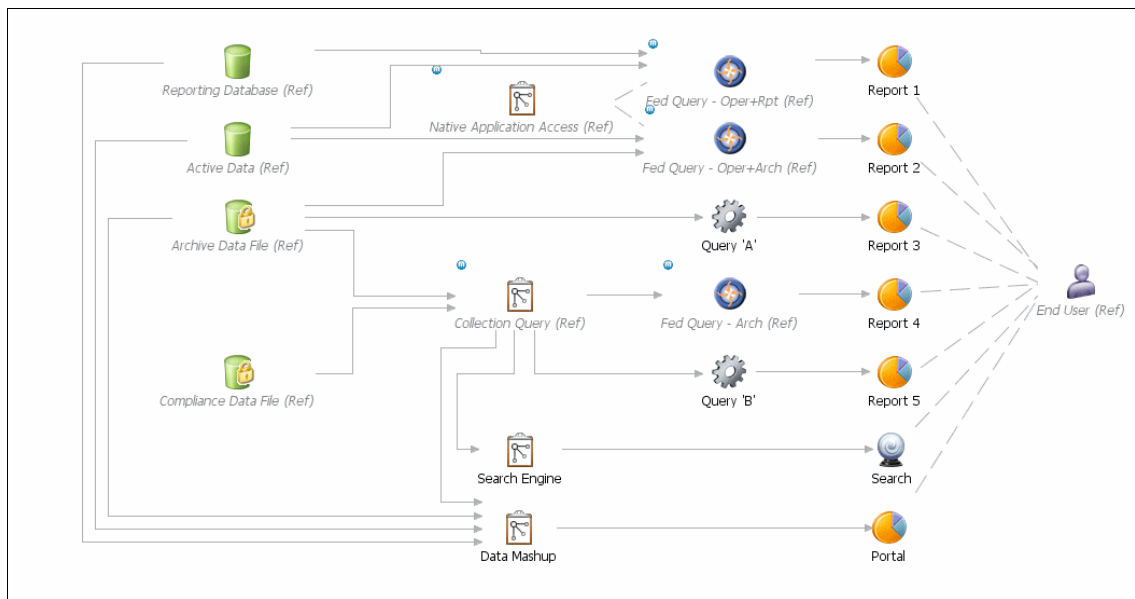


Figure 3-23 Universal access

The universal access infrastructure provides flexibility to integrate information at the enterprise level:

- ▶ Access archived data and virtual database (archive collection) using ODBC, JDBC, or a federated database.
- ▶ Access hot, warm, and cold data either in your application (native application access) or a report writer.
- ▶ Access archived non-relational data as relational data.
- ▶ Access archived data as XML data.

- ▶ Enable mashing up data from multiple sources and presenting it using a web browser.
- ▶ Find data using a search function.

### **Archived data using ODBC, JDBC, or Federated Database**

Archived data is always accessible using ANSI SQL so that you do not need any special process to deliver data to business users. Data can be stored on SAN, NAS, WORM, and backup devices, and Optim still can access it and provide data back to the user without requiring any special processing.

### **Native application access to hot, warm, and cold data**

An application that requires the transparent integration of hot, warm, and cold data can have a universal view of data without changing the application code. Optim builds views that the DBMS can process and transparently provide a union of all the data back to the application.

### **Archived non-relational data as relational data**

Non-relational data is dynamically accessible as relational data, so that report writers and web pages can access data in a similar manner to any other relational database. Business users, auditors, and compliance teams can use standard tools to retrieve information from systems designed 20, 30, or 40 years ago.

### **Enable data mashup**

Organizations present information through portals to Internet and intranet users. A tabular report is not always the best method to navigate data. The Optim mashup engine enables you to click on a piece of information and dynamically (and asynchronously) connect to other information. A customer can see the orders of the last three years. Click one order and retrieve the billing information. Click the billing and see a credit card refund. Three systems feed one web portal that integrates multiple data feeds to display on one window.

### **Find data when you do not know it**

When organizations do not know yet what data is needed, flexibility is the top priority. The *data find* function searches archived data and organizes it in a way that you can search from any angle, similar to a web search engine, and yet, it still understands that it is relational data and not documents. Searching for a transaction with “Mercury”? What if it is also a product, a person, or a company? Data find dynamically brings the data and hints to you where more data is hidden. Auditing data, planning a litigation hold, or getting ready to discontinue a product does not require an army of people to find out what you need.





## **Developing an InfoSphere Optim Data Growth Solution architecture**

In this chapter, we describe the components, considerations, and process of developing a solution architecture to successfully implement and integrate the IBM InfoSphere Optim Data Growth Solution (Optim) into your environment, from project definition through validation. The foundation is built on the knowledge, experience, and proven architectures of successful Optim deployments in both small and large clients across a variety of industries.

Your IT enterprise architecture provides the framework for multiple layers of architecture, including data, application, software, systems, network, security, and storage architectures. Optim is an application that manages data growth by archiving older data while providing access to archived data to users and therefore spans all of those architectural views.

For any given Optim implementation, there might be many options for architecting and deploying the solution. The intent of this chapter is to present not only the “what”, but also the “why” and the “how” of Optim solution architecture by describing the components in the architecture, comparing various options, providing guidance, and discussing the rationale behind key architectural decisions.

We described the Optim enterprise architecture in 3.3, “Enterprise architecture” on page 42. A solution architecture provides a model of the system that suppresses implementation details and concentrates on the analyses and decisions that are most crucial to structuring the system to satisfy your requirements. It includes both software and hardware components, the externally visible properties of those components, and the relationships among them.

Developing the architecture is an iterative process that evolves from a high-level, conceptual architecture in the early phases of the project life cycle to a more specific, physical, and operational architecture during later phases. This chapter follows this progression, from the conceptual to the physical, providing a transition to Chapter 5, “Sizing the Optim infrastructure” on page 173, in which the sizing considerations of the physical components of the architecture are discussed.

In this chapter, we discuss the following topics within the framework of the methodology that was described in Chapter 2, “Project planning and considerations” on page 17:

- ▶ Project scope
- ▶ Methodology
- ▶ Requirements
- ▶ Software and hardware components
- ▶ Reference deployment models
- ▶ Extensibility of the deployment models
- ▶ Optimizing the solution architecture
- ▶ Testing and validating the architecture
- ▶ Sample Optim solution architectures

**Flexible:** InfoSphere Optim Data Growth Solution has been architected to provide a flexible, scalable, and configurable infrastructure spanning these areas:

- ▶ Small to exceptionally large data growth management initiatives
- ▶ Homogeneous or heterogeneous data sources
- ▶ Simple or complex data schemas
- ▶ Relational and hierarchical database systems and file systems
- ▶ A wide variety of operating systems and hardware platforms
- ▶ Multiple data center locations
- ▶ Efficient archive create process and archive data access
- ▶ Effective and cost-saving management of historical data through end of life (purge)

Table 4-1 on page 95 summarizes several of the architectural activities during each project phase, most of which occur in the early phases of the project (analyze, design, and configure). Depending on the scope, duration, and

complexity of your deployment, only a few or many of the items listed need to be addressed.

*Table 4-1 Sample architectural activities by project phase*

Phase	Architectural activities
Project management	<ul style="list-style-type: none"> <li>▶ Identify business goals and drivers</li> <li>▶ Define project scope</li> <li>▶ Set project schedule and phasing</li> <li>▶ Employ methodology</li> </ul>
Analyze	<ul style="list-style-type: none"> <li>▶ Document business, technical, and non-functional requirements</li> <li>▶ Inventory assets and technologies</li> <li>▶ Understand constraints</li> <li>▶ Identify standards</li> <li>▶ Define use case models</li> </ul>
Design	<ul style="list-style-type: none"> <li>▶ Determine required Optim components</li> <li>▶ Utilize reference deployment model</li> <li>▶ Revise and extend deployment model</li> <li>▶ Identify software and hardware components</li> <li>▶ Align architecture with Optim design and process flow</li> <li>▶ Determine preliminary hardware sizing estimate</li> <li>▶ Hold technical delivery assessment (TDA) review</li> <li>▶ Assess viability of architecture</li> <li>▶ Document the architectural decisions</li> </ul>
Configure	<ul style="list-style-type: none"> <li>▶ Configure Optim infrastructure</li> <li>▶ Validate any assumptions</li> <li>▶ Finalize hardware sizing estimates</li> <li>▶ Revise architecture as needed</li> </ul>
Deploy	<ul style="list-style-type: none"> <li>▶ Test and measure results against requirements</li> <li>▶ Perform tuning</li> <li>▶ Adjust as necessary</li> </ul>
Operate	<ul style="list-style-type: none"> <li>▶ Review and validate</li> </ul>

## 4.1 Project management

You need to develop the Optim solution architecture within the context of your larger enterprise IT architecture using a methodology, such as the methodology described in 2.1.1, “Project methodology” on page 18, with project management practices applied.

### 4.1.1 Business goals

There might be multiple business goals driving your data growth management initiative, for example:

- ▶ Manage the growth of structured production data within a complete information life-cycle governance framework, from data creation through end of life (purge).
- ▶ Reduce the size of the production databases to improve online and batch application performance.
- ▶ Reduce the cost of expensive tier one storage and migrate aged data to less expensive storage based on the business value and access requirements.
- ▶ Provide user access to archived data for historical reporting, regulatory compliance, and audit purposes.

### 4.1.2 Project scope

In determining the scope of your InfoSphere Optim Data Growth Solution project, the first step is to decide where you are on your data governance journey:

- ▶ About 10% of Optim clients want to develop a comprehensive data governance practice, along with a Center of Excellence, and create an enterprise architecture to support their Information Life Cycle Management initiatives across a broad range of data sources. Certain clients want to determine the strategic architecture, including the impact on their application portfolio and capacity planning (server, network, and storage) prior to implementing the first project. In this scenario, IT architecture provides the framework within which all of these component parts are integrated to provide a common infrastructure to support the business objectives.
- ▶ About 80-85% of Optim clients start an InfoSphere Optim Data Growth Solution project with a few applications and data sources, gain experience, and extend the solution to other data sources using a repeatable, successful process. One key driver in developing their solution architecture is to build an Optim infrastructure that can both meet the current tactical requirements and also be flexible, extensible, and scalable to include additional data sources

and data volumes in future phases of their strategic corporate data governance initiative. In this situation, there is considerable benefit to understanding and defining a minimum architectural framework for Optim at the enterprise-wide level to ensure that the decisions taken at the individual application or system level are not made in isolation and will conform with and support the overall directions of your corporate data governance strategy.

- ▶ About 5-10% of Optim clients have a more tactical approach and only require an architecture that addresses specific data sources. Frequently, this tactical approach is motivated by a specific business driver, such as the immediate need to reduce the size of a production database for performance reasons.

These three approaches to data governance present separate requirements, and the architectural answers and artifacts need to map appropriately to them.

### 4.1.3 Project schedule and phasing

Project management activities also include setting the schedule and implementation of the various data growth project phases:

- ▶ When can the first phase begin and what are the critical target deadlines?
- ▶ Are there any key business drivers associated with project completion, such as the need to reduce the production database size prior to a seasonal peak or application migration?
- ▶ Will there be a single phase in which a significant percentage of your source data will be archived to “catch up” for all of the years that data has accumulated in your production data store? Can any bulk archiving occur gradually or does it need to be completed quickly?
- ▶ Will you have a second phase of your project in which you will incrementally archive data on a regular basis to keep the size of the production databases at a specific size?
- ▶ Will project phasing be aligned to specific data sources, for example, by application area, database management system (DBMS), time frame, platform, business event, or data center?
- ▶ Do project schedules dictate that multiple data sources need to be archived concurrently, either initially or incrementally?

The scope and phases of your project affect whether the architecture has to meet only your tactical, immediate mandates, or be flexible, scalable, and extendable for future data growth processing workloads within the framework of your strategic IT ecosystem.

## 4.1.4 Methodology

Several of the benefits in following a methodology while developing your Optim solution architecture are that it establishes and documents a repeatable process, helps to mitigate risk, and ensures a successful project that achieves your business objectives. The project phases and associated architectural activities in Table 4-1 on page 95 represent a summary of the methodology.

Figure 4-1 depicts a high-level overview of several of the inputs and considerations that provide the foundation for developing and reviewing a candidate architecture for an Optim Data Growth implementation. We will describe each component and its impact on the architecture in later sections of this chapter.

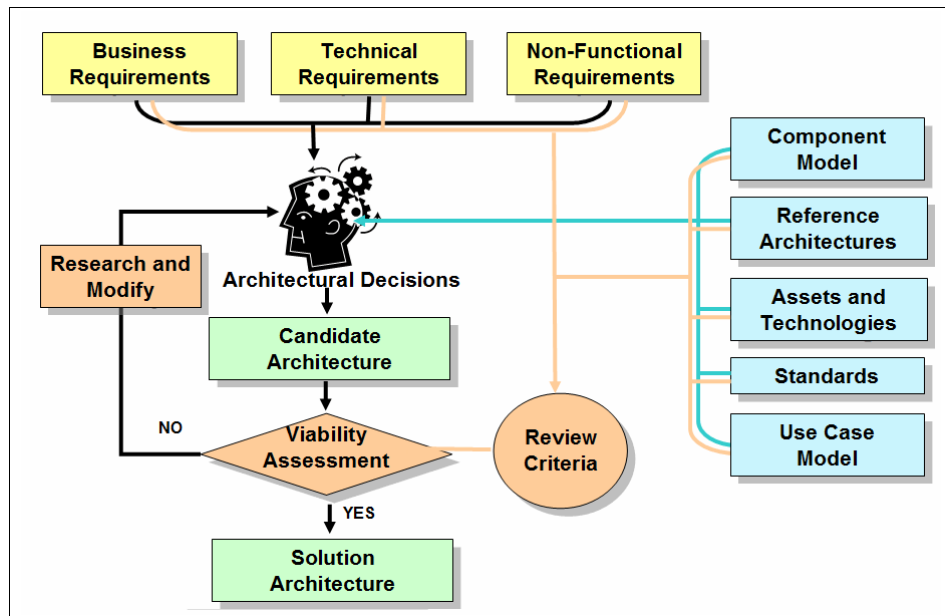


Figure 4-1 High-level view of architecting an InfoSphere Optim Data Growth Solution architecture

Various candidate architectures might evolve during the project as more information becomes known. The design phase begins by exploiting proven, successful reference deployment models, which form the base template for an Optim solution architecture best suited to your environment and objectives. We describe various extensibility options and provide illustrations to expand and customize the deployment models to satisfy your additional business, technical, operational, and non-functional requirements.

The architecture might evolve over time, requirements might change, decisions will be made, constraints might surface, and schedules must be met. In the end, you have to produce a viable architecture that meet all of your business, technical, and non-functional requirements for the current projects, while providing a solid foundation and flexible framework for future implementations of your data growth management initiative.

## 4.2 Analyze

During the analyze phase of the project, critical information and requirements are gathered and documented. The analyze phase consists of these key components:

- ▶ Requirements
- ▶ Inventory of assets and technologies
- ▶ Corporate IT standards
- ▶ Use case models

### 4.2.1 Requirements

Gathering and documenting your data growth and data governance requirements are key to achieving a successful implementation. Good project management of a system design is impossible without explicit validation and governing requirements. Ensure that all stakeholders (business leaders, data center management, auditors, users, database administrators, and so on) are represented. Each stakeholder might have a separate perspective on what is needed, from an abstract view (for example, business process) to a concrete view (for example, network bandwidth). In addition, each stakeholder might have separate priorities that might need to be negotiated. Requirements and constraints are generally categorized into three classifications:

- ▶ Business (functional) requirements
- ▶ Technical and operational requirements
- ▶ Non-functional requirements

In this section, we list sample requirements that drive the solution architecture for the InfoSphere Optim Data Growth Solution.

#### **Business requirements**

Business, or functional, requirements document what the system will “do”, which includes the required behaviors and business rules, as well as features and functions. These requirements are generally addressed in the system design, but they also affect the solution architecture.

Table 4-2 lists key business requirements, constraints, and considerations that affect the solution architecture.

*Table 4-2 Sample key business requirements and constraints*

Sample business requirements and constraints	Sample architectural considerations
Data growth management for an application or data source	<ul style="list-style-type: none"> <li>▶ Reduction of the size of production data</li> <li>▶ Volume of warm and cold data to initially and incrementally archive</li> <li>▶ Time constraints</li> <li>▶ Location of data</li> </ul>
User access and reporting capabilities provided for archived data for “n” number of years	<ul style="list-style-type: none"> <li>▶ Data retention policies for compliance, history, and regulatory compliance</li> <li>▶ Tiered storage strategy</li> <li>▶ Indexes on archive files</li> <li>▶ Restore archived data to reporting database</li> <li>▶ Federation of hot and warm/cold data</li> <li>▶ Service-level agreement (SLA)</li> <li>▶ High availability</li> </ul>
Complete business objects archived with the ability to restore the data, if necessary	<ul style="list-style-type: none"> <li>▶ Data schema and referential constraints</li> <li>▶ Infrastructure to restore archived data back to production database or other DBMS environment</li> <li>▶ Capacity of infrastructure to restore data within the operational window</li> <li>▶ Temporary storage for loader files if you restore using load</li> <li>▶ Operational window</li> </ul>
Inventory of data sources and applications	<ul style="list-style-type: none"> <li>▶ Data sources, operating system (OS), and data volume</li> <li>▶ Native access to relational DBMS (RDBMS ) data</li> <li>▶ Middleware required to access federated and non-relational data</li> <li>▶ Sequential or concurrent processing of multiple data sources</li> <li>▶ Network</li> <li>▶ Placement of Optim components</li> <li>▶ Project phasing</li> <li>▶ Targeted or enterprise-wide solution/architecture</li> </ul>

Sample business requirements and constraints	Sample architectural considerations
Data volume metrics	<ul style="list-style-type: none"> <li>▶ Data volume to be initially and incrementally archived, deleted, and optionally restored</li> <li>▶ Operational window - on source data host (for archive and delete)</li> <li>▶ Operational window on Optim host (for compression, building archive file indexes, archive access, and the creation of the loader file)</li> <li>▶ Frequency of processing</li> <li>▶ Processing concurrency</li> <li>▶ Tiered storage strategy</li> </ul>
Data retention requirements: Regulatory and compliance	<ul style="list-style-type: none"> <li>▶ Length of time to retain archived data on each storage tier</li> <li>▶ Storage capacity and tiered storage</li> <li>▶ Frequency of access</li> <li>▶ Index archive files</li> <li>▶ Define time frame for safe disposal and shredding retired data</li> <li>▶ Disaster recovery</li> </ul>
Use cases	<ul style="list-style-type: none"> <li>▶ Criteria to archive data versus criteria to access archived data</li> <li>▶ Processing workflows</li> <li>▶ Concurrency</li> <li>▶ Scaling</li> </ul>
Budget	<ul style="list-style-type: none"> <li>▶ Hardware resources (cost)</li> <li>▶ Impact on performance and throughput</li> <li>▶ Time constraints</li> </ul>
Risk threshold	<ul style="list-style-type: none"> <li>▶ Proven reference architecture</li> <li>▶ Viability assessment</li> <li>▶ Project schedule and critical target dates</li> </ul>
Project phases	<ul style="list-style-type: none"> <li>▶ Initial versus incremental processing</li> <li>▶ One or multiple data sources</li> <li>▶ Sequential or concurrent use of infrastructure resources</li> </ul>

## Technical and operational requirements

Technical and operational requirements and constraints document the characteristics that you consider critical to deploying and executing Optim within your environment. Table 4-3 on page 102 lists examples of technical requirements.

Table 4-3 Sample technical requirements and constraints

Sample technical requirements and constraints	Sample architectural considerations
Guidelines and corporate IT standards	<ul style="list-style-type: none"> <li>▶ Solution must comply with corporate standard operating environment (SOE)</li> <li>▶ In-house IT skills</li> </ul>
DBMS or middleware technologies	<ul style="list-style-type: none"> <li>▶ Preference for relational DBMS</li> <li>▶ Preference for middleware software</li> </ul>
Server types (vendor, model, and specifications)	<ul style="list-style-type: none"> <li>▶ Preference for server model and specifications</li> <li>▶ Physical or virtualized hosts</li> </ul>
Storage	<ul style="list-style-type: none"> <li>▶ Performance of I/O subsystem</li> <li>▶ Tiered storage strategy</li> <li>▶ Storage management software</li> <li>▶ Increased storage capacity throughout project life cycle</li> </ul>
Network and connectivity	<ul style="list-style-type: none"> <li>▶ Locations of data sources</li> <li>▶ Placement of Optim hosts</li> <li>▶ Capacity of all components in network stack</li> <li>▶ Network bandwidth</li> <li>▶ Concurrent network workload</li> </ul>
Source data	<ul style="list-style-type: none"> <li>▶ Relational, federated, or non-relational source</li> <li>▶ Capacity and utilization of source data host</li> <li>▶ Data volume (current and target)</li> <li>▶ Data growth rate</li> <li>▶ Processing period (off-peak)</li> <li>▶ Federation across data source</li> <li>▶ Database partitioning</li> </ul>
Operational window	<ul style="list-style-type: none"> <li>▶ Critical business operational windows</li> <li>▶ Capacity of Optim host processor</li> <li>▶ Utilization of source data host</li> <li>▶ Performance of I/O subsystem</li> <li>▶ Network capacity</li> <li>▶ Contiguous or non-contiguous operational window</li> <li>▶ Frequency</li> <li>▶ Time of day</li> </ul>
Systems management	<ul style="list-style-type: none"> <li>▶ Automation</li> <li>▶ Consistent design and deploy interface across all platforms</li> <li>▶ High availability and disaster recovery</li> <li>▶ Scalability and flexibility of infrastructure</li> </ul>

## Non-Functional requirements

Non-functional requirements (NFRs) document what the system will “be”, which includes more qualitative and quantitative characteristics and those

requirements, which impose constraints on the design or implementation of the system, such as performance engineering requirements, quality standards, or design constraints. These requirements are detailed in the system architecture and drive many of the architectural decisions. Early in the data growth management project, there is typically a substantial concern for functional requirements. However, failure to satisfy the non-functional requirements can lead to risk and an unacceptable solution for one or more of the stakeholders.

Non-functional requirements describe and judge the operations of a system and can determine the minimum possible criteria for stakeholders to sign off on the deployment. A solution cannot operate successfully if the infrastructure or configurations are done incorrectly, even though it might meet the functional requirements. Table 4-4 lists key non-functional requirements.

*Table 4-4 Key non-functional requirements and constraints*

<b>Sample non-functional requirements and constraints</b>	<b>Sample architectural considerations</b>
Performance and throughput	<ul style="list-style-type: none"> <li>▶ Balanced system: Optim host, source data host, network, and storage</li> <li>▶ Archive create process versus archive access requirements</li> <li>▶ Delete processing: Drop partition versus Optim delete</li> <li>▶ Restore using insert or load</li> </ul>
SLA for users	<ul style="list-style-type: none"> <li>▶ Archive access: Add indexes to archive files and allocate additional storage for indexes</li> <li>▶ Restore archive data to reporting or other database</li> <li>▶ Performance</li> </ul>
Capacity	<ul style="list-style-type: none"> <li>▶ Optim host capacity and scalability</li> <li>▶ Network</li> <li>▶ I/O subsystem performance and capacity (all tiers)</li> <li>▶ Data volumes: Initial and incremental</li> <li>▶ Processing concurrency</li> <li>▶ Timing and key target dates</li> <li>▶ Frequency</li> <li>▶ Sizing of hardware infrastructure</li> </ul>
Scalability and flexibility	<ul style="list-style-type: none"> <li>▶ Project scope and project phase</li> <li>▶ Processing frequency</li> <li>▶ Selection of OS for Optim server</li> <li>▶ Optim host capacity (scale up and scale out)</li> <li>▶ Data center location</li> <li>▶ Physical or virtual hosts</li> <li>▶ Current versus future requirements (data volumes, number of data sources, and so on)</li> </ul>

Sample non-functional requirements and constraints	Sample architectural considerations
Availability	<ul style="list-style-type: none"> <li>▶ Extra capacity for high availability</li> <li>▶ No single point of failure</li> <li>▶ Operational considerations</li> <li>▶ Shared versus dedicated storage</li> </ul>
Recoverability	<ul style="list-style-type: none"> <li>▶ Disaster recovery site</li> <li>▶ Synchronization of Optim directories at production and disaster recovery sites</li> <li>▶ Replication of archive files at disaster recovery site</li> <li>▶ SLA</li> <li>▶ Operational considerations</li> <li>▶ Automation</li> </ul>
Security and privacy	<ul style="list-style-type: none"> <li>▶ Separate infrastructures for archive create process and archive access</li> <li>▶ Separate Optim repositories (by project, data source, business area, location, role, and so on)</li> </ul>

It is important to evaluate the business, technical, and non-functional requirements for their contribution to business goals, for cost, and for schedule impact. Certain clients find it helpful to document, prioritize, and track the requirements in a requirements traceability matrix so that the advantages and disadvantages of various architectural decisions can be evaluated with those objectives in mind, to help mitigate risk, and to ensure all requirements are met.

## 4.2.2 Inventory of assets and technologies

Early in the project, it is important to take an inventory of all of the source data that you want to include in the data growth initiative, both near term and long term, so the scale and robustness of the infrastructure will be prepared to handle the workload. Include these key items in the inventory:

- ▶ Applications that are candidates for data growth management:
  - Current data volume
  - Growth rate
  - Critical thresholds due to data growth
  - Target amount of data to archive and delete initially, and then incrementally
- ▶ Data sources:
  - Relational databases
  - Federated data sources
  - Non-relational data sources

- ▶ Operating systems:
  - OS for each data source
  - Distributed or mainframe
- ▶ Data center locations
- ▶ Servers for host systems:
  - Average and peak utilization
  - Operational window
- ▶ Network capacity
- ▶ Storage hardware and technologies in your enterprise architecture:
  - Classes of storage
  - Storage management systems
- ▶ Middleware technologies

After data sources have been identified, they have to be prioritized and scheduled in the project plan. The number of data sources, data volumes, operational windows, and concurrency will impact the size of the Optim infrastructure.

### 4.2.3 Standards

You must identify your corporate IT standards in the analysis phase, so that appropriate architectural decisions can be made, for example:

- ▶ Standard operating environment:
  - For example, standard OS is Linux (RHEL)
  - For example, preferred DBMS is Oracle
- ▶ Standard hardware vendor or model:
  - For example, quad-core blade servers
  - For example, IBM POWER® architecture
- ▶ Deployment standards:
  - For example, placement of components on servers
  - Application access inside and outside of the firewalls
  - Type of storage used based on SLA
- ▶ Security standards

## 4.2.4 Use case models

For each application and data source that will be included in the Optim Data Growth project, you must define use case models to ensure that the processing and workload can be modeled and the appropriate infrastructure can be deployed. Consider these key points:

- ▶ Archive create process:
  - How to archive the data to include the complete business object (for example, the number of related tables and referential constraints)?
  - What are the business drivers (for instance, performance, data retention, regulatory compliancy, cost)?
  - Define key criteria for archiving source data (for example, date range, region, or status code).
  - Data volume (10 GB or 100 TB), frequency (daily, monthly, or yearly), operational window (three hours or gradually, over three months).
  - Strategy to archive extremely large volumes of data in smaller, more manageable increments (for example, a database with hundreds of gigabytes of data needs to be archived by multiple archive requests).
  - Number of input databases or files into the archive create process. For example, do you need to archive 10 flat files or 1,000 flat files?
- ▶ Archive access:
  - Define key criteria for accessing archived data (for example, date range, department, or status code).
  - Number of concurrent users.
  - Frequency of access.
  - Types of business queries (complexity, data volume, and number of archive files accessed).
  - SLA.
  - Access requirements, such as open data manager (ODM), native application access, report writer, and so forth.

The requirements, constraints, inventory of assets and technologies, standards, and use case models, as shown in Figure 4-1 on page 98, are inputs to the next phase, design, during which many of the solution architectural decisions are made.

## 4.3 Design

Much of the initial solution architecture work is done during the design phase using information and artifacts that were produced in the analysis phase. Based upon the data sources, data volumes, and locations previously identified, you can identify the Optim hardware and software components for your solution. Proven, reference deployment models are used as the basis for developing your customized solution architecture. The models can be extended and adjusted relative to your specific requirements, constraints, assets and technologies, data sources, standards, and use cases.

Considerations for optimal placement of the software components on the hardware are presented, along with the advantages and disadvantages of various alternatives. The candidate architecture is then aligned to key use cases, and the design and workflow of Optim processes (archive, delete, compare, browse, and restore) to ensure that the infrastructure can support the requirements. A preliminary hardware sizing might be done during this phase.

Based upon your IT standards, one or multiple architectural reviews are frequently done at the completion of the design phase for a preliminary validation of the architecture.

### 4.3.1 Optim components

Your data sources and archive access requirements will determine the required Optim software and hardware requirements. These requirements can vary, depending on the data sources and whether Optim is installed on Linux, UNIX, or Microsoft Windows.

#### **Optim on distributed platforms**

There are several software and hardware components in an Optim architecture that is deployed in the distributed platform environment. Figure 4-2 on page 108 shows several of the most common data sources.

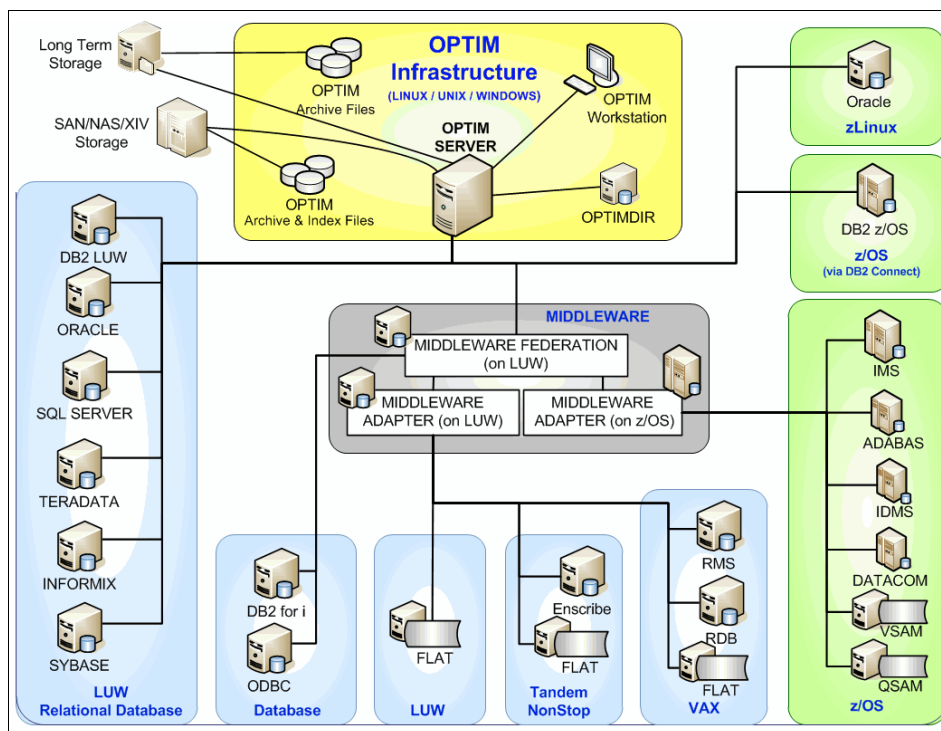


Figure 4-2 Optim Installed on Linux, UNIX, or Microsoft Windows

We define the major Optim distributed components:

► Data source

The data source is the source (for example, a database or a file) from which data is archived. Refer to Chapter 6, “Data sources” on page 213 for a complete description of data sources. Optim in a distributed platform environment supports a variety of data sources on heterogeneous platforms. Data sources fall into three categories, depending on the software stack that is required to access the data: native, federated, or non-relational.

Three software stacks provide support to a variety of data sources:

– Native

Data in relational database management systems (RDBMS) can be accessed by Optim through the database client, using native drivers. The RDBMS can host the required catalog and stored procedures.

– Federated

Certain data sources do not meet Optim requirements for either cataloging stored procedures, SQL level, or API functionality, so middleware

federation software (either IBM InfoSphere Federation Server or Oracle DG4ODBC) is required to host the stored procedures and translate between the APIs. Note that as of Version 11g, Oracle Heterogeneous Services ODBC Generic Connectivity using ODBC (HSODBC) has had its name changed to Oracle Database Gateway for ODBC (DG4ODBC).

- Non-relational

Non-relational data sources require metadata management to map the non-relational source to the simulated relational model. Additionally, a data source-specific adapter converts the federated requests to the native data source API.

- ▶ Source data host

The source data host is an operating system that is installed on a physical server or a logical server in a virtualized environment, such as a logical partition (LPAR) in which the source data host is installed (for example, a DB2 database server). A data source can be installed on a single host or multiple hosts, clustered, and can be located in one or multiple data centers.

- ▶ Optim host

This host is a physical server or a logical partition in a virtualized environment such as a logical partition (LPAR), with a single operating system in which the Optim server instance is installed.

- ▶ Optim infrastructure

This infrastructure is a deployment of InfoSphere Optim Data Growth Solution that is installed on a host, including the repository (Optim directory), user interface workstation, and associated files (archive files, archive file indexes, and so on). The Optim infrastructure is shown in the light yellow area in Figure 4-3 on page 111. The Optim infrastructure is part of the complete hosting environment where the Optim host is provisioned, including network, storage infrastructure, security, backup management, and other supporting software for management and operations.

- ▶ Optim server instance

This instance is an installation of the InfoSphere Optim Data Growth Solution server software that provides services and processes running on Linux, UNIX, or Microsoft Windows that are designed to archive, delete, compare, browse, and restore data. This instance is the server component of the client/server architecture.

- ▶ Optim GUI interface

This interface is the client component and can either be the Optim workstation, Optim designer, or both.

- ▶ Optim repository

The Optim repository contains the Optim directory (`optimdir`), a series of tables residing in a relational DBMS, and is used to store Optim project definitions, configuration information, archive catalogs, audit information, storage profiles, and security definitions. The Optim directory can be installed in a relational DBMS (DB2 for Linux, UNIX, and Microsoft Windows, Oracle, SQL Server, Informix, or Sybase).

- **Middleware federation**

Middleware federation software sits between Optim and both federated and non-relational data sources, providing SQL access to those data sources. Middleware federation software can be installed on Linux, UNIX, or Microsoft Windows, and hosts the stored procedures that are needed by Optim. Options are either IBM InfoSphere Federation Server or Oracle heterogeneous services, DG4ODBC.

- **Middleware adapter**

The middleware adapter software provides access to non-relational data sources by hosting metadata and mapping the non-relational data to a simulated RDBMS. Options are either IBM InfoSphere Classic Federation on z/OS, Optim Connect on z/OS, or Optim Connect on distributed platforms. Middleware adapter software is generally collocated on the source data host.

- **Archive file**

This file contains metadata, Data Definition Language (DDL), tables, and archived data. Archives can be in a contiguous file or in segments, and they can be stored as compressed files or files that are not compressed.

- **Archive file collection**

This logical grouping of one or more archive files is accessible as one unit. A collection can contain multiple years of the same data, archives with separate data, or a combination. Collections do not duplicate data, but they enable dynamic viewing of the combined data. An archive file can participate in multiple collections.

- **Archive file index**

This file contains multiple b-tree structures that are designed to optimize random query access to archive files. Archive file indexes are not compressed.

- **Storage**

Storage is for source data, Optim archive files, and archive file indexes, and temporary storage during processing (for example, control files during delete, insert, or load processing and loader files for restore through load processing).

During archive, delete, browse, restore, and archive access processing, it is important to understand the workload's impact on each component so that an architecture can be developed, which is robust enough to meet the processing throughput requirements.

Optim executes a number of operations and generally engages in I/O intensive tasks that most of the time move large quantities of data. Every Optim process utilizes resources on the Optim host and can also utilize resources on any of the source, target, and intermediate systems. Optim on distributed platforms has a client/server architecture and can transmit large amounts of data over the network between the data source host, the Optim host, and the storage subsystem. Figure 4-3 provides an overview of the various components that are involved in a data growth implementation and classifies them into three categories:

- ▶ Host processors: Optim, middleware, and source data hosts (red circle)
- ▶ Network (green dotted line)
- ▶ Storage (blue circle)

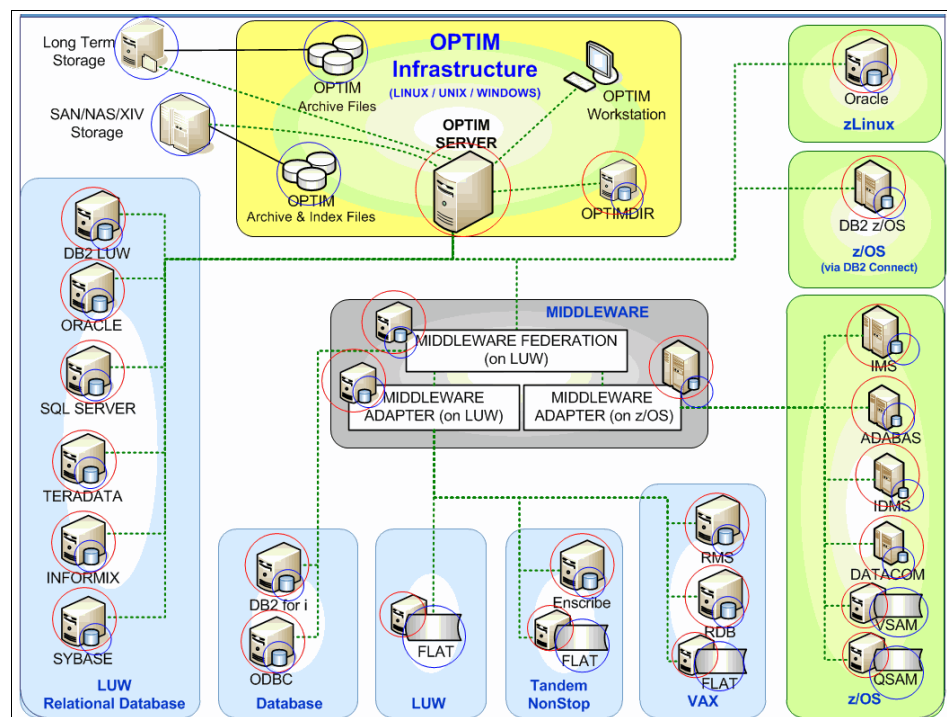


Figure 4-3 Workload during processing: Optim on Linux, UNIX, and Microsoft Windows

For instance, during an archive process, Optim reads large amounts of data from a source data host based on certain selection criteria. For example, read all of

the data for the service request complete business object from the Siebel database on Oracle associated with service requests that were closed in 2009. This Optim archive request results in work on the Oracle database engine that parses the query, executes it, fetches the data from disk storage, processes the results, and sends the data across the network to the Optim host. The Optim host further processes the data, compresses it, optionally creates indexes on the archive file, and writes the resulting archive files to storage. During a delete process, Optim verifies and sends a request to the database, which executes a bulk delete operation for a set of records based on specific criteria. Again, the source database host, Optim host, network, and storage subsystems all run part of the workload. When data is accessed from the archive files, Optim parses the query, reads the indexes, assimilates data from the archive files, and presents it to the user. During a restore process, Optim reads data from one or more archive files, extracts it, and performs either an insert or create loader file and load operations to restore the data to a target database.

For each of the processes involved in your data growth deployment, the hosts, storage, and network are critical components, which must be properly sized and configured to accommodate workloads within your operational window. A bottleneck in any component will affect the overall performance and throughput of the solution. For example, the Optim host might have the adequate capacity to concurrently process data from four separate data sources. However, if your network bandwidth is constrained either due to capacity, bandwidth, or other concurrent network workloads, the Optim server will receive the data at a slower rate and you will not be able to achieve maximum or optimal performance.

### **Components: Optim on z/OS**

Figure 4-4 on page 113 shows the components in an Optim architecture deployed in a mainframe environment on z/OS.

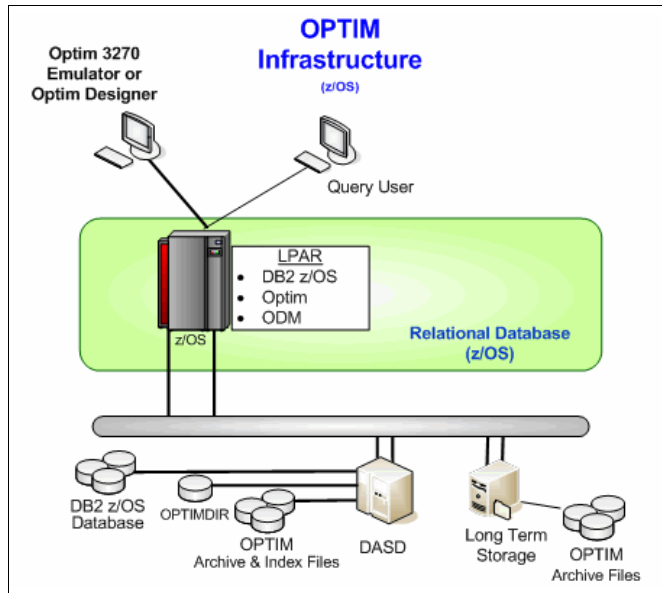


Figure 4-4 Optim installed on z/OS

Optim on z/OS consists of the following components:

- Data source

The data source is the source from which data is archived. Optim on z/OS can archive data from DB2 on z/OS. Other data sources on z/OS can be archived by Optim on distributed OSs (Linux, UNIX, or Microsoft Windows). See Chapter 6, “Data sources” on page 213 for additional information.

- Source data host

This host is an operating system installed on a physical server or a logical server in a virtualized environment, such as an LPAR, in which the source data host instance is installed (for example, database server). DB2 for z/OS is installed on the LPAR that hosts the Optim on a System z (mainframe) processor. A data source can be installed on a single host or multiple hosts and reside in one or multiple data centers.

- Optim host

This host is an operating system that is installed on a physical server or a logical server (LPAR) in which the Optim server instance is installed. Optim is installed in the same LPAR as the source DB2 z/OS system.

- Optim instance

This instance is an installation of the InfoSphere Optim Data Growth Solution server software that provides services and processes running on z/OS that

are designed to archive, delete, compare, browse, and restore DB2 for z/OS data. Only one instance of Optim software is installed per z/OS host LPAR.

- Optim interface

The interface to Optim can either be a 3270 emulator, Optim designer (Microsoft Windows), or both.

- Optim repository

The Optim repository contains the Optim directory, which is a series of tables residing in a DB2 for z/OS database, and is used to store Optim project definitions, configuration information, archive catalogs, audit information, and security definitions.

Optim on z/OS has been optimized for the z/OS platform and does not have a client/server architecture. It is installed on the same LPAR as the source DB2 z/OS database. Figure 4-5 provides an overview of various components that take part in the data growth solution, including host processors and storage. The Optim directory can be installed as a separate database schema in the same DB2 for z/OS instance as the source database or in a separate DB2 instance.

Figure 4-5 classifies the components for Optim on z/OS into two categories:

- Host processors (LPARs) (red circle)
- Storage (blue circle)

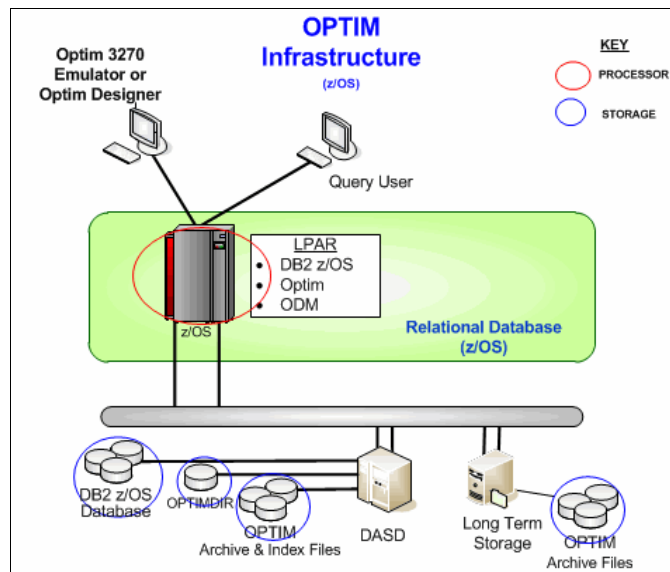


Figure 4-5 Workload during processing: Optim on z/OS

## 4.3.2 Software components

### Key architecture considerations:

- ▶ Data sources
- ▶ Platform
- ▶ Middleware components

It is important to identify the specific software components that are required to support your processing. The software components are installed on a server (host); certain components also have a configuration component, which is installed on a workstation. Optimizing and configuring the installation and configuration of the software components on physical or virtual servers are part of the overall solution architecture.

### Optim and middleware software components (distributed)

All Optim Data Growth implementations when Optim server is installed on Linux, UNIX, or Microsoft Windows require the minimum software components for both the server and client workstation that are shown in Table 4-5.

Table 4-5 Minimum software components required

Software component (server)	Configuration (workstation)
Optim server	Optim workstation Optim designer (recommended)
Optim repository hosted on a relational DBMS on Linux, UNIX, or Microsoft Windows: DB2; Oracle; Sybase; Informix; or SQL Server (Microsoft Windows only for SQL Server)	N/A

For Optim on distributed platforms, part or all of the components that are listed in Table 4-6 might be required, depending on your requirements and data sources.

Table 4-6 Additional software components

Software component (server)	Configuration (workstation)
Open data manager	Optim connect studio
Optim Connect (on Linux, UNIX, Microsoft Windows, or z/OS)	Optim connect studio
InfoSphere Federation Server (includes DB2 for Linux, UNIX, and Microsoft Windows)	Federation Center Control Center

Software component (server)	Configuration (workstation)
Oracle Heterogeneous Services DG4ODBC (includes Oracle)	N/A
Relevant 32-Bit RDBMS Client	Relevant 32-Bit RDBMS Client
Open Database Connectivity (ODBC) driver	N/A
Classic Federation (installed on z/OS)	Classic Data Architect
DB2 Connect™	N/A

## Optim software components (z/OS)

Table 4-7 lists the components for Optim on z/OS.

*Table 4-7 Software components for Optim on z/OS*

Software component (server)	Configuration (workstation)
Optim	3270 Emulator Optim designer (recommended on Microsoft Windows)
Optim repository on DB2 z/OS	N/A

The component listed in Table 4-8 might be required for Optim on z/OS, depending on your requirements and data sources.

*Table 4-8 Additional software components for Optim on z/OS*

Software component (server)	Configuration (workstation)
Open data manager	Optim connect studio

Optim is operating system-agnostic in regard to the operating system on which the source data resides and is able to access data from many heterogeneous sources. The middleware components that are used to access that data might have operating system version limitations and might not support less common operating systems. Check with the middleware provider for specifications. If you have data residing on an unsupported operating system, the data can be migrated to a supported operating system and then archived. This task has sizing implications in that temporary storage on a server with a supported operating system will need to be provisioned.

## RDBMS native drivers

For Optim on distributed platforms, the Optim directory can only be accessed using native relational database drivers. DB2, Oracle, Sybase, and Informix all

provide native drivers so the Optim server can be installed on either Linux, UNIX, or Microsoft Windows.

When the data source is SQL Server, Optim server must be installed on the Microsoft Windows operating system because Microsoft does not provide a driver for Linux or UNIX. The SQL Server Optim directory can only be accessed by Microsoft Windows machines.

### ***Applications***

Optim supports archiving for custom applications, data warehouses, and independent software vendor (ISV)-packaged applications, such as Oracle E-Business Suite, Siebel, PeopleSoft, JD Edwards, Maximo, Lawson, and others. In the requirements phase of the project, certain users might identify a need to have archived data selectively restored to a temporary database for access to historical data from their application through native application access (NAA). Historical (“warm” and “cold”) and “hot” operational data can be viewed together.

Other users might want archived data restored to a reporting database to satisfy certain SLAs for reporting, audit, compliance, or litigation purposes. Still other users might require a “restore on demand” capability for a specific subset of archived data for various purposes. Occasionally, users might want specific archived data restored to their production database.

## **4.3.3 Reference deployment models**

Optim has been architected to provide a multitude of configurations that are flexible, extensible, and scalable to satisfy both your current and future processing requirements. There are three deployment models:

- ▶ Centralized
- ▶ Decentralized
- ▶ Operationally optimized

These deployment models form the framework for all Optim solution architectures. After the reference models have been described, additional factors are applied to demonstrate how the solution architecture can evolve and expand to meet various business, technical, and non-functional requirements. These deployment models lead to the design of a more detailed physical and operational architecture.

We review these models with Optim installed on a distributed platform (Linux, UNIX, or Microsoft Windows) first. Then, we present considerations when Optim is installed on a z/OS platform.

## Centralized deployment model

### Key architectural considerations:

- ▶ Data sources
- ▶ Data volumes
- ▶ Single data center in a location
- ▶ Operational window
- ▶ User archived data access requirements

The centralized deployment model for Optim Data Growth Solution is characterized by having a single Optim server in one data center location. The Optim repository contains the metadata for one or multiple heterogeneous data sources that are stored in the Optim directory (OPTIMDIR). A single Optim server instance executes all data growth processing, including archive create, compare, delete source data after archive, browse, user archive access, and archive restore. Figure 4-6 depicts the architecture of a centralized deployment model.

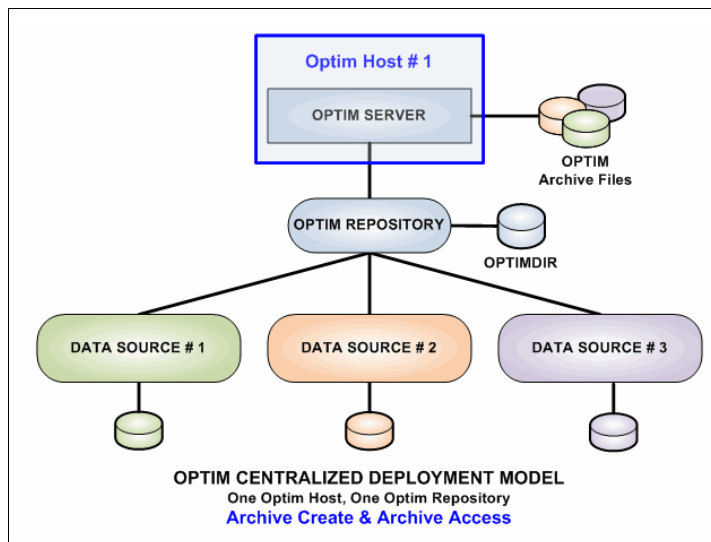


Figure 4-6 Centralized deployment model: One Optim Server and repository

## Decentralized deployment model

### Key architectural considerations:

- ▶ Number of data sources
- ▶ Data volumes
- ▶ Multiple data center locations
- ▶ Operational window
- ▶ Capacity and scalability

The decentralized deployment model is characterized by having two or more Optim hosts across which the workload is distributed. Each Optim host can run archive, delete, compare, restore, and archive access processing. At least one Optim server instance is installed on each Optim host. The Optim hosts might be in the same data center or separate data centers so that the Optim server instance can be close to the source data host. Figure 4-7 illustrates this model with one shared Optim repository with metadata for all three data sources. This model is useful when a single Optim server does not have the capacity to process the required data volume within the operational window. It provides a “scale-out” strategy so that the workload is apportioned to multiple Optim hosts.

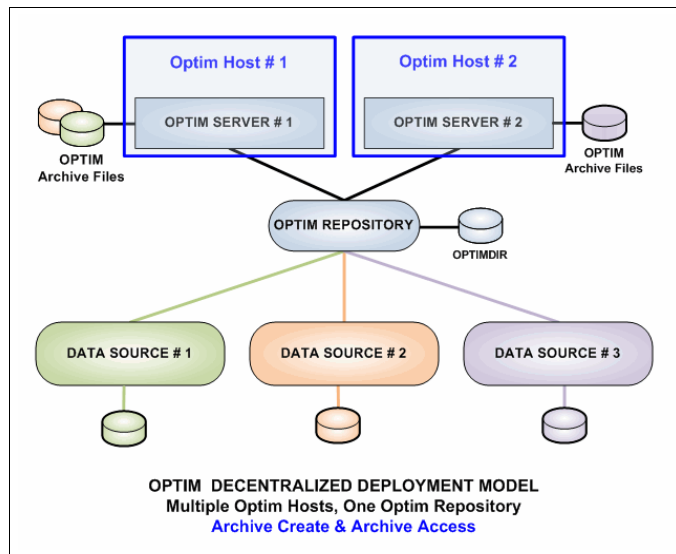


Figure 4-7 Decentralized deployment model: Multiple Optim servers and one repository

A variation of the decentralized deployment model, as shown in Figure 4-8 on page 120, is when each Optim server instance has its own repository, which expands the flexibility and scalability for implementing the decentralized deployment model. Optim server instances #1 and #2 independently run all data

growth functions, each processing data from one or multiple, heterogeneous data sources. Due to the separation of Optim infrastructures, this model offers the following benefits:

- ▶ Optim can be installed in separate data center locations when data sources are geographically distributed.
- ▶ Optim can be installed on separate operating systems (for example, Optim Server #1 on Linux and Optim Server #2 on z/OS) in either a single data center location or multiple data center locations.

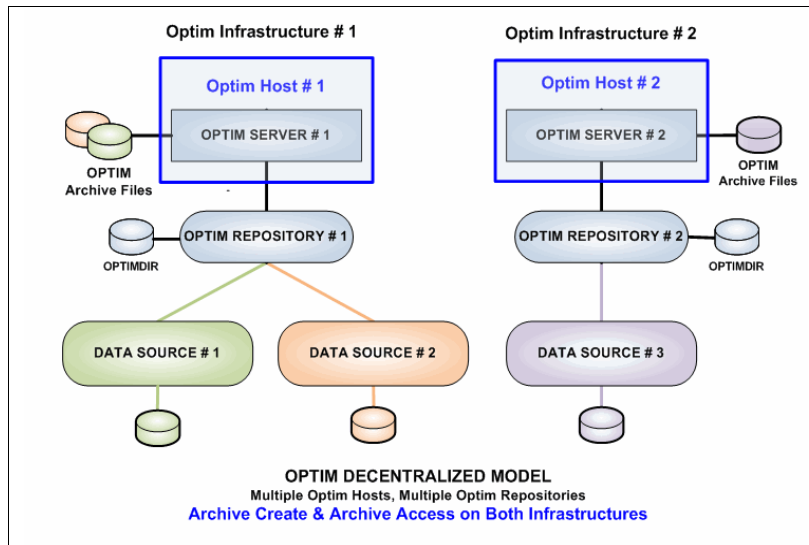


Figure 4-8 Decentralized deployment model with two Optim server instances, each with its own repository

The decentralized deployment model is the preferable approach if you are archiving data from geographically distributed data centers. This approach provides an Optim infrastructure in each data center and avoids the network overhead and latency of transmitting large volumes of data over the network. It is considered a best practice to have the Optim host in the same data center location as the source data host, preferably on the same subnet.

This model is also implemented by clients who want separate Optim infrastructures dedicated to a specific application or data source (for example, Optim server #1 installed on UNIX to archive Oracle, flat files, and Teradata, and Optim server #2 installed on Microsoft Windows to archive SQL Server data.)

Another advantage in using the decentralized deployment model is that it provides the capability to migrate archive files created in one platform to another platform so that all files are centrally managed.

## Operationally optimized deployment model

### Key architectural considerations:

- ▶ Separation of user roles and responsibilities: Archive create and archive access
- ▶ Frequent archive activity
- ▶ Frequent queries against archive files
- ▶ SLA for data retrieval
- ▶ Security
- ▶ Separation of administrative roles and responsibilities
- ▶ Operational window
- ▶ User archive data access requirements

There are two fundamental processing characteristics that you must consider in a data growth project: archive creation and archive access. Do you frequently archive large volumes of data and also have a requirement to provide users frequent and timely access to that archived data? Deploying two Optim infrastructures, which are configured and operationally optimized for a specific purpose, such as archive creation (write) and archive access (read), might provide the best overall throughput and consistent performance.

The operationally optimized deployment model is characterized by having two or more Optim server instances, each with its own repository. Each server is dedicated to either archive create and delete processing, or archive access processing.

In Figure 4-9 on page 122, Optim Server #1 is shown to be dedicated to an archive create role in which it reads data from Client Data Sources #1 and #2. This portion of the infrastructure has large read activity from the data sources and large write activity to either compressed or non-compressed archive files. Archive files that are created in this step are written to a storage network, which might be shared between the two Optim infrastructures. Optim can be tuned and managed to a larger extent for the role it is designed to play. Separate Optim repositories for the two infrastructures provides a physical separation for added security.

Optim Server #2 is shown to be dedicated to an archive access role. In order for this server to read the archive files that are created by Optim Server #1, the metadata for these archive files has to be “published” from repository #1 to repository #2 (refer to “Synchronizing Optim repositories” on page 133). After this operation is complete, Optim Server #2 can access the archive files and perform

other operations on these files, such as creating indexes on the archive files, restoring the data to a reporting or historical database, and archive file access. In our example, we illustrate shared network storage. In another case, for example, where Optim infrastructure #2 is in another data center and requires separate storage for performance reasons, the archive files need to be physically transferred to Optim infrastructure #2, and the files reregistered to repository #2. Optionally, Optim server #2 can also restore all or a portion of the archived data to a reporting database as your requirements might dictate.

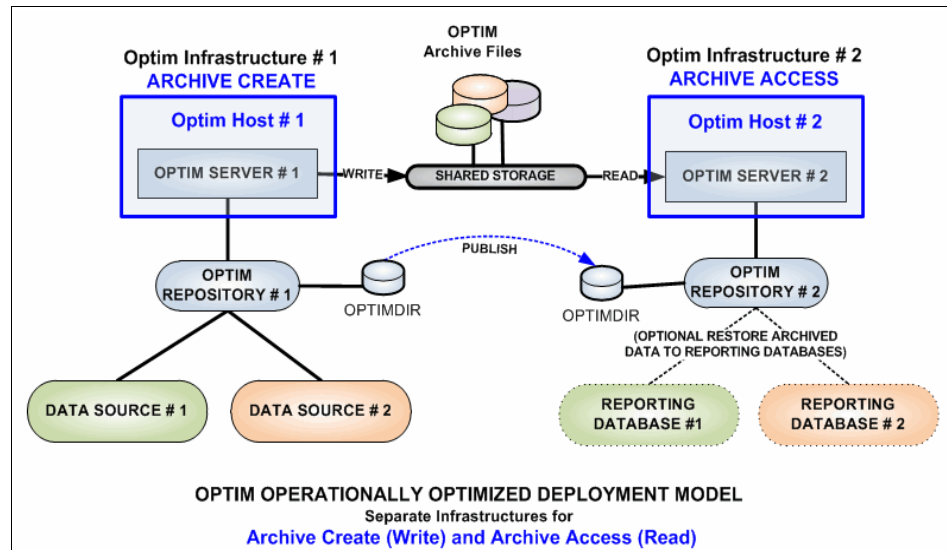


Figure 4-9 Operationally optimized deployment model: Separate infrastructures for archive create and archive access, shared storage

Separating the Optim servers and the repositories in this way allows for better performance, greater flexibility, scalability, redundancy, and separation of roles and responsibilities. The operationally optimized deployment model is a good choice for applications that require frequent archival processing due to significant growth. This model provides archive access to users who need to query and view archived data without affecting the ongoing archive operations. In turn, the performance of user queries is not affected by the heavy archive processing demands. This configuration is operationally optimized to provide more consistent performance for users accessing archived data when requirements dictate SLAs for queries and reporting against archived data.

## Deployment model considerations for Optim on z/OS

The three deployment models are applicable for Optim on z/OS with the following differences:

- ▶ There is no concept of “server” in z/OS, because it is not client/server architecture. The “Optim server” component that is depicted in Figure 4-9 on page 122, for example, is Optim installed on a DB2 z/OS LPAR.
- ▶ Only one instance of Optim is installed on a single z/OS LPAR.
- ▶ Optim software is installed in the same LPAR as the source DB2 z/OS data; therefore, there is no network component between Optim and the data source. Optim on z/OS provides two additional options to minimize or eliminate the impact of reading the production database during an archive process:
  - Installing Optim on an LPAR other than the LPAR that contains the source database
  - Archiving from an image copy of the DB2 for z/OS database

### 4.3.4 Extensibility of the deployment models

The centralized, decentralized, and operationally optimized deployments provide the templates from which you start to develop your solution architecture to satisfy your functional, technical, and non-functional requirements.

Many clients begin their data growth project by archiving a few data sources and then extending the solution to include additional data sources, increased data volumes, perhaps more data centers, and additional query users. We now describe how the three deployment models can be extended to accommodate a variety of requirements, including performance, flexibility, and scalability, to meet both your current and future processing requirements.

There are multiple extensibility strategies, but it is important to first understand any constraints in your environment so that you can focus on the most critical components of the solution architecture. At a high level, constraints fall into one of two categories:

- ▶ Time
- ▶ Money (resources)

Whether you have a short operational window or have to reduce the production database size by 50% within the next month in preparation for an application upgrade, these situations present a shorter time frame in which to run your archives. Also, *time* might be a predominant driver in making architectural decisions. To solve a time constraint, focus on strategies that will increase processing throughput so that the overall elapsed time will be reduced. To

complete the same amount of work in less time, this strategy will generally consume more resources and might cost more. This approach has architectural and resource (processor, network, and storage) sizing implications.

If, however, *money* is your constraint (for example, IT budget, cost of resources, database server capacity, Optim host processor capacity and speed, network bandwidth, and performance of disk subsystem), focus on strategies that distribute the workload over longer periods of time to use fewer resources.

Achieving your desired balance of time and money (resources) is key. The following checklist identifies considerations that might prompt the need to extend and scale your deployment model, or that might cause you to throttle the throughput if resources are limited due to budget constraints:

- ▶ When to consider extending and scaling the deployment model:
  - Time is an issue:
    - A single archive process cannot meet the time constraints.
    - Little or no batch window on production data source.
  - Workload exceeds the capacity of single Optim server instance.
  - Bottleneck on source data host, Optim host, network, or storage.
  - Large data volume:
    - Adding additional data sources or increasing the data volume.
    - Project phase: initial “catch up” bulk archive.
  - Source data in multiple data centers
- ▶ When to consider scaling down or throttling the model:
  - Resource constraints (for example, source data host or network) that cannot be resolved due to budget constraints.

There are multiple options for increasing Optim processing throughput and extending the deployment models, for example:

- ▶ Parallelism
- ▶ Multi-processing: Scaling the number of concurrent processes
- ▶ Scaling the number of Optim repositories
- ▶ Scaling the number of Optim server instances (Linux or UNIX)
- ▶ Scale the Optim host (add processor cores and RAM)
- ▶ Scale the number of Optim hosts
- ▶ High availability
- ▶ Disaster recovery

## Parallelism

### Key architectural considerations:

- ▶ Volume of data
- ▶ Small operational window on source data host
- ▶ Capacity and utilization of source data host during operational window
- ▶ Network capacity between source data host and Optim host
- ▶ Performance and throughput
- ▶ Processor speed of Optim host

One way to scale the solution is to enable parallelism within Optim. By default, when Optim processes a single archive or delete request, one database connection (thread) is created between the Optim host and the source data host. Optim has sophisticated cursor management to enable parallelism and multiple database connections to the source database for archive and delete processing, which increases the throughput between the database and Optim. Parallelism occurs within a single table at a time and processes a group of keys on separate database connections. Refer to *The Optim Archive User Manual* and *The Optim Common Elements Manual* for the specific settings and configuration.

Although increasing the number of database connections is generally a design consideration, it does affect the architecture in that adequate capacity for the network, database host, and Optim host must be architected to support the additional concurrent workload. All data retrieved from multiple database connections is returned to a single Optim mainline processing thread, so the speed of the Optim host is important. Be cognizant of the total workload in which this processing and when it will be scheduled; generally Optim batch processing occurs during database off-peak hours.

Parallelism represents a way to scale an individual process. However, if the source data host, Optim host, I/O, or network capacity is exhausted, increasing parallel Optim processing might not improve and can degrade performance. Test in your environment to validate that performance and throughput gains can be achieved.

Figure 4-10 on page 126 illustrates the parallelism of Optim with four database connections for a single archive process.

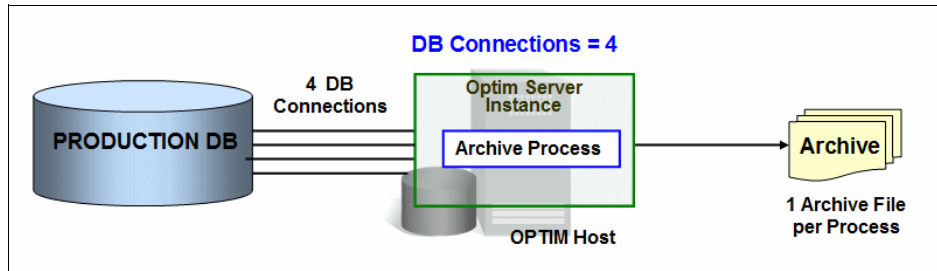


Figure 4-10 Multi-processing: Four database connections for an archive process

Note that during archive processing, parallelism does not apply to the start table or reference tables. The table must have a unique key with a minimum of 10,000 keys in order to enable parallelism. You can specify from two up to 16 database connections for each process, or “max”, which is determined by the number of processor cores in the Optim host. For example, if the Optim host has two cores and you specify four database connections, you can use as many as eight connections per process, which is multiplied by the number of multiple concurrent processes if applicable. In a delete process, only requests with 1,000 or more rows are processed in parallel.

## Multi-processing: Scaling the number of concurrent processes

### Key architectural considerations:

- ▶ Performance and throughput
- ▶ Utilizing multiple-processor Optim hosts
- ▶ Increase the number of concurrent processes
- ▶ Processing workflow of multiple data sources: Concurrent or sequential

One way to scale the data growth solution and process more data in a shorter amount of time is by implementing multi-processing in which multiple concurrent processes (Linux, UNIX, or Microsoft Windows) or jobs (z/OS) process a “slice” of the complete business object based on key range, partition key range, application domain area, or other selection criteria. For example, executing eight archive jobs concurrently might significantly increase the throughput and decrease the elapsed time to complete the work.

Optim must be installed on a host with multiple processor cores to get better throughput and utilize all available capacity when multi-processing. The concurrent workload is distributed across the cores and better utilizes server capacity. Another way to utilize Optim server capacity and improve elapsed time is to increase the number of database connections (parallelism), because the engine can now send more requests to the source database and perform the

same amount of work faster than it can otherwise with a single database connection.

Prior to full production implementation, it is best to test the performance at various loads. As a starting point, provide one processor core for each concurrent process, test the impact on resource utilization and performance, and make adjustments as warranted. Increasing the number of concurrent processes also affects the load on the source data host, Optim host, network, and I/O subsystem. It is best to gradually increase the number of concurrent processes and measure key performance indicators (KPIs). At a certain point, you might hit a bottleneck in one of the components, but after tuning, you might be able to achieve higher concurrency and throughput rates.

The concurrent processes can be executed against the same data source in which each process will, for example, archive a specific portion of the complete business object based on key range, as shown in Figure 4-11. Segmentation of the archive processing against a single table is based on an SQL WHERE clause on columns in the Optim Access Definition.

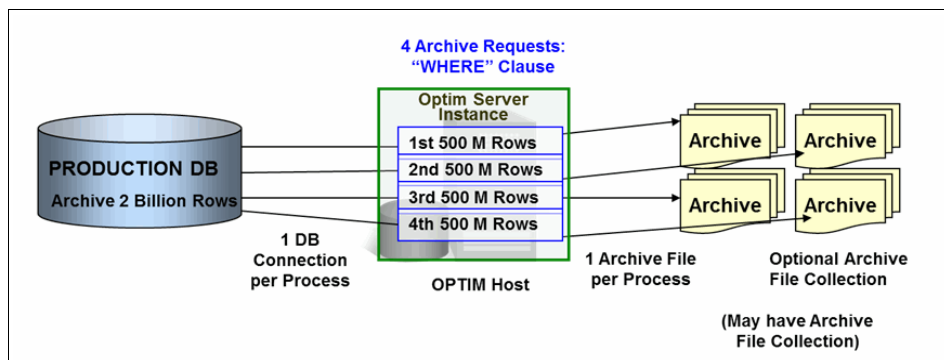


Figure 4-11 Multi-processing: Four concurrent archive requests against a single table based on key range

A variation of this strategy is, for each process, to run against data in a specific partition, as illustrated in Figure 4-12 on page 128.

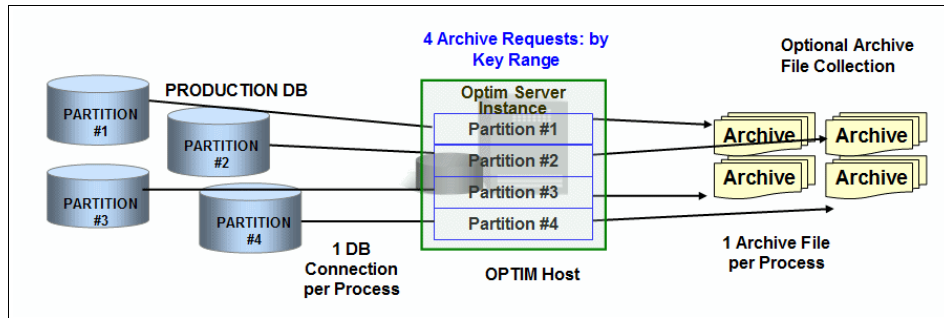


Figure 4-12 Multi-processing: Four concurrent archive requests, each against a separate database partition

The third variation on multi-processing is to run multiple concurrent jobs against separate data sources, as seen in Figure 4-13. In this scenario, Optim is simultaneously archiving data from the Human Resource, Payroll, and Manufacturing applications. This approach is an effective strategy if your processing workflow requires, for example, that multiple data sources are archived on the last day of each month. Again, it is important to provide enough capacity to accommodate the increased workload.

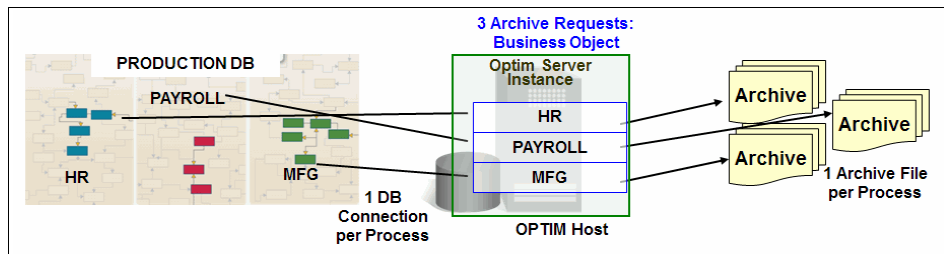


Figure 4-13 Multi-processing: Three concurrent archive requests, each against a separate data source

This approach represents a scale-up strategy of the Optim software. Multiple concurrent processes can execute within a single Optim server instance, providing an effective way to process large amounts of data within a constrained operational window.

### ***Platform considerations: Number of concurrent processes***

There are platform-specific differences in the number of concurrent processes that you can run on an Optim server instance, depending on whether Optim is installed on a Linux, UNIX, Microsoft Windows, or z/OS operating system:

#### ► Microsoft Windows

In a Microsoft Windows environment, there is a theoretical maximum of 48 concurrent processes when you run on “local”. You can only run local when Optim is installed on Microsoft Windows. “Local” means that the Optim client and Optim server are physically installed on the same Microsoft Windows processor, which might be done in development or test. However, it is best to run on “server” and not “local” in a production environment. Running a job on the server ensures that resource-intensive processes and functions are run where they are best suited. This gives better performance, good control of archive file creation, resource utilization, security, and central management of all Optim jobs and processes.

When you run Optim on a “server” (Linux, UNIX, or Microsoft Windows), each process has a corresponding mirror process (the mirror process does not really perform much work), which means that there is a maximum of 24 concurrent processes (pr0sver) per Optim server in Linux, UNIX, or Microsoft Windows.

The actual number of concurrent tasks that can run in a given environment is highly dependent on a number of factors, including the capacity of the Optim host (number of processor cores, clock speed, and memory), performance of the data source host (for instance, database server), the capacity and bandwidth of the network, and the performance of the storage subsystem.

#### ► Linux or UNIX

If Optim is installed on a UNIX or Linux processor, you can run up to 24 concurrent tasks within a single instance of the Optim software, similar to the Microsoft Windows environment. However, if Optim is installed on Linux or UNIX, multiple instances of the Optim server can be installed under a single Linux or UNIX operating system image, and each instance can run up to 24 concurrent tasks. For more details, refer to “Scaling the number of Optim server instances” on page 134.

#### ► z/OS

If Optim is installed on z/OS, the number of concurrent jobs executing is only limited by the capacity of the other resources required to complete the workload (capacity of the host processor, memory, and disk I/O subsystem).

## Scaling the number of Optim repositories

### Key architectural considerations:

- ▶ Segregate the Optim directory metadata by project, security, environment, roles and responsibilities, or other criteria.
- ▶ Budget constraint for hardware but the client requires a logical separation of the environments.
- ▶ Large number of archive files and projects.

The Optim repository contains only a few tables in a relational database, and it is generally small. Typical Optim repositories are 512 KB to 5 GB, depending on the number of projects, access definitions, archive files, and so on. However, when designing the number of repositories and purpose for each repository, consider the ultimate number of archive files and projects that you plan to have when your data growth solution is fully deployed. An Optim repository that contains metadata for 20,000 archives will be faster than an Optim repository that catalogs 300,000 archives. It is best to implement a “business area” strategy (or similar) to ensure that each repository is not overutilized.

In any of the three reference deployment models (centralized, decentralized, or operationally optimized), you can have these configurations:

- ▶ One Optim repository for one Optim server instance (1:1)
- ▶ One Optim repository for multiple Optim server instances (1:*m*)
- ▶ Multiple Optim repositories for one Optim server instance (*m*:1)
- ▶ Multiple Optim repositories for multiple Optim server instances (*m*:*m*)

Multiple repositories provides more flexibility in segregating metadata for various data sources, projects, functions, environments, and roles and responsibilities.

Figure 4-14 on page 131 shows extending the centralized deployment model to have two or more repositories for a single Optim server instance on Microsoft Windows, Linux, UNIX, or z/OS. In this example, metadata for data source #1 and #2 are stored in repository #1, while data source #3's metadata is stored in repository #2. All data archiving, deleting, restoring, and archive access for all three data sources is executed on a single, centralized Optim server instance on a single Optim host.

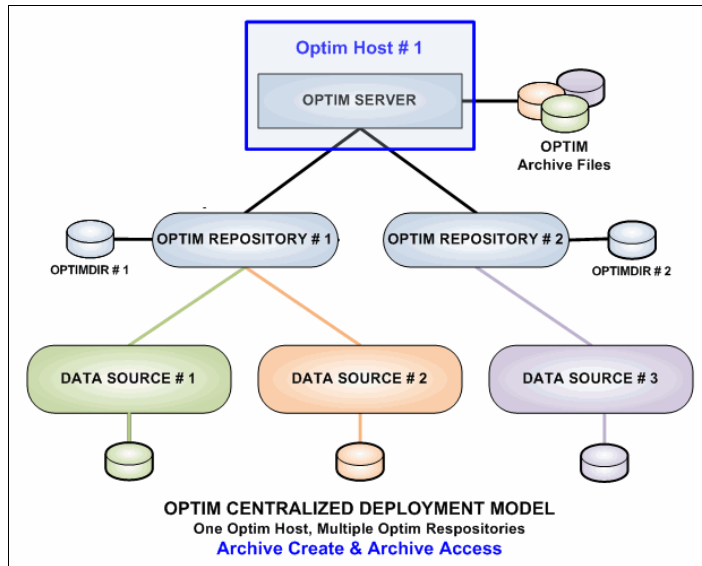


Figure 4-14 Centralized deployment model: One Optim server and multiple repositories

This approach can be adapted for smaller Optim deployments with a limited hardware infrastructure budget to provide a logical separation of environments. For example, development (data source #1) and test (data source #2) share repository #1 (non-production), and production (data source #3) has its own repository #2 (production). All three environments (development, test, and production) share a single Optim server instance.

Extending the decentralized deployment model with additional repositories, such as shown in Figure 4-15 on page 132, provides even more flexibility and scalability. In this scenario, infrastructure #1 is the non-production environment and repository #1 is for development and repository #2 is for test. Infrastructure #2 is the production environment with its own repository #3. The infrastructures in this model can also be segregated by geography or data center, project, data source, access and security, operating system, production and development, roles and responsibilities, or by other criteria.

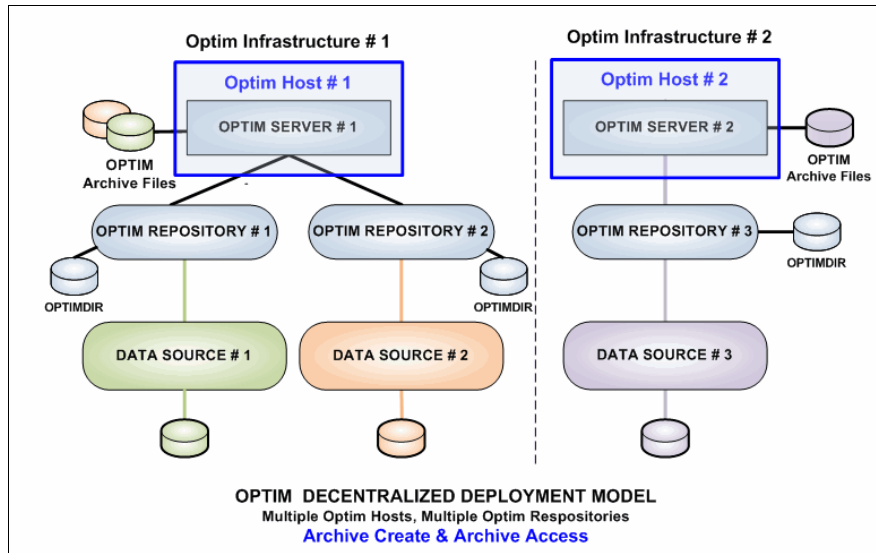


Figure 4-15 Decentralized deployment model: Multiple Optim servers and multiple repositories

Figure 4-16 illustrates an operationally optimized deployment model. In this example, multiple Optim repositories are implemented for archive create processing (one per data source). The metadata from both repositories is combined into a single repository for the archive access infrastructure.

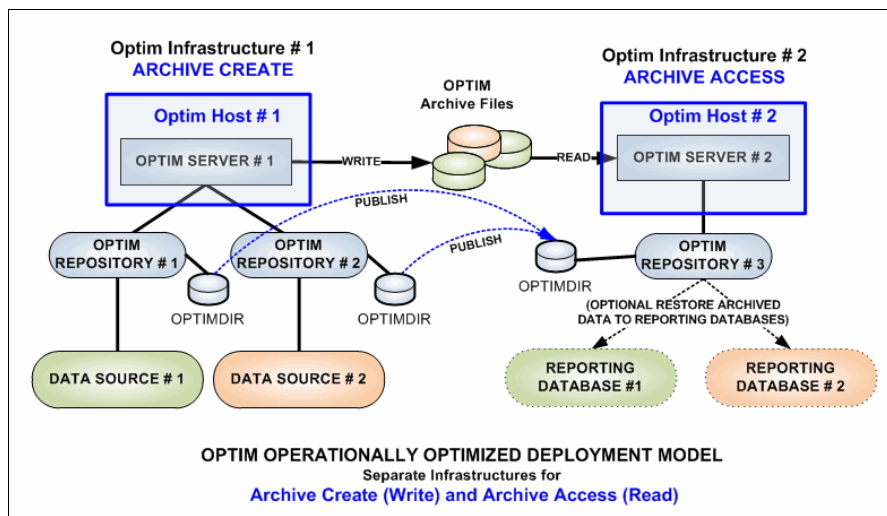


Figure 4-16 Operationally optimized deployment model: Two Optim server instances and separation of Optim repositories

## Synchronizing Optim repositories

You might want to have multiple Optim repositories to keep metadata for various projects or applications segregated. However, if you implement separate repositories for archive create and archive access processing, it might be necessary to keep the repositories and metadata synchronized, depending on your requirements (see Figure 4-16 on page 132).

You must “publish” the metadata between repositories when Optim is installed on Linux, UNIX, or Microsoft Windows, as seen in Figure 4-17, by exporting and importing the metadata from Optim repository #1 into Optim repository #2 for archive access, and reregistering the archive files.

Another scenario is to have source data on both the mainframe and distributed platform environments and the corporate strategy is to migrate the applications and data off of the mainframe onto the distributed platform environment. For the initial archive, your infrastructure might look similar to Figure 4-17 in which infrastructure #1 is Optim on distributed platforms and infrastructure #2 is Optim on z/OS. After archiving the data on z/OS, the compressed archive files can be transferred to the Linux environment where the **/migrate** utility can be executed to convert the files from z/OS to distributed format. The **/migrate** utility also registers the archive files from repository #2 to repository #1 on Linux and changes the three-part z/OS names to two-part distributed names.

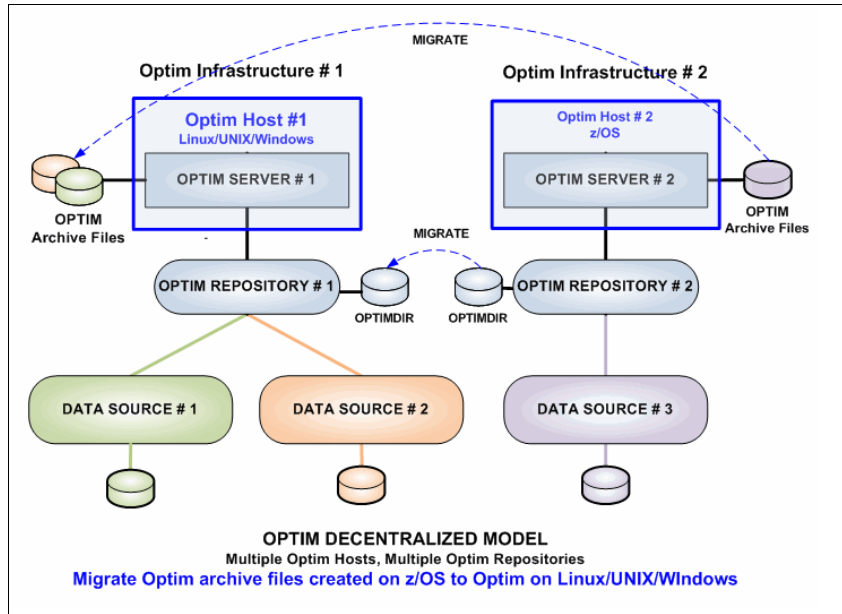


Figure 4-17 Migrating archive files created on z/OS to Optim on Linux, UNIX, and Microsoft Windows

## Scaling the number of Optim server instances

### Key architectural considerations:

- ▶ Performance and throughput
- ▶ Large data volumes
- ▶ Operational window
- ▶ Multiple data sources
- ▶ Concurrency
- ▶ Scalability

### ***Optim on Microsoft Windows***

Optim installed on Microsoft Windows is for departmental environments or to access data in SQL Server. One instance of the Optim server can be installed on the Microsoft Windows operating system<sup>1</sup>, which means that each Optim server can run up to 24 concurrent tasks (provided the host processors, network, and storage have enough capacity). If you require additional concurrent processing to complete the workload within the allotted operational window, you can scale out horizontally to additional hosts (hardware scale-out strategy), each with a single Optim server instance. See “Scale the number of Optim hosts” on page 137.

### ***Optim on Linux or UNIX***

Optim installed on Linux (RHEL or SUSE) provides additional scalability options for enterprise deployments. After you have achieved the maximum throughput that is available on a single Optim server installed on either Linux or UNIX (up to 24 concurrent tasks), you can optionally scale out the software by installing multiple instances of Optim server on a single Linux or UNIX operating system on a single host. Each instance can run up to 24 concurrent tasks, which provides excellent horizontal, scale-out capability within a single Optim host.

For example, you can install four instances of the Optim server on a single Intel server running Linux or on a UNIX server or LPAR running AIX, HP/UX, or Solaris. This option allows you to scale the solution within a single host, assuming that the other resources (processor, network, source data host, and storage) have adequate capacity to handle the additional concurrent workload without hitting any bottlenecks. The Optim server instances can share a common repository, each have a separate repository, or each have multiple repositories, providing you with both exceptional flexibility and scalability.

When scaling the number of Optim server instances and the corresponding number of concurrent tasks, it is important to analyze the impact on the overall infrastructure, including the source data hosts, the Optim host, the network capacity, and the disk subsystem. Ensure that there is adequate additional

---

<sup>1</sup> Single instance due to Microsoft Windows Registry.

resource capacity before scaling the number of Optim server instances so that you have a balanced system across all components. Increasing the workload by adding additional Optim servers without considering the capacity of the other resources can result in queuing and potentially degraded performance.

Figure 4-18 depicts a centralized deployment model with three Optim servers installed on either a single Linux or UNIX operating system on a single Optim host.

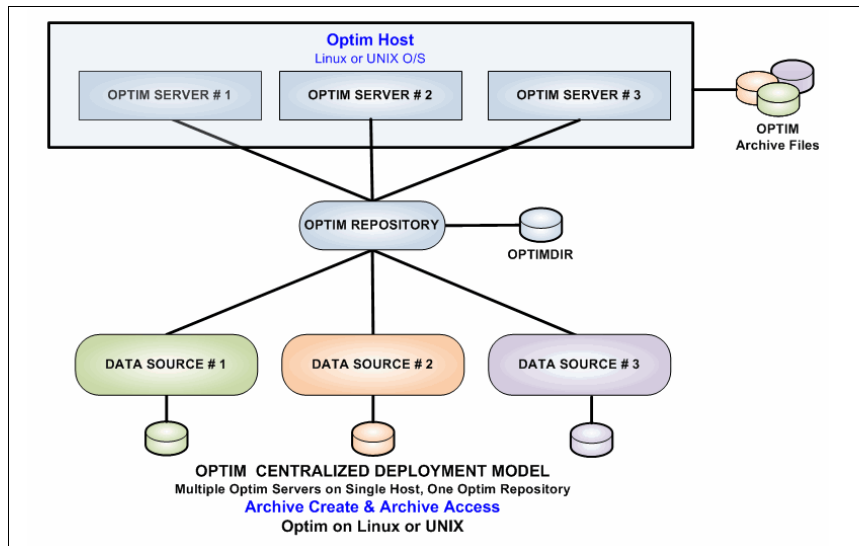


Figure 4-18 Multiple Optim server instances on a single host

Figure 4-19 on page 136 shows a variation of the centralized deployment model in which one or more Optim server instances is installed on a single Optim host. You can still have multiple database owners in the database, and each owner can host a separate Optim repository (for example, one for production, one for test, and one for development, or one for each data source). In this way, the repositories can be separate and secure from each other.

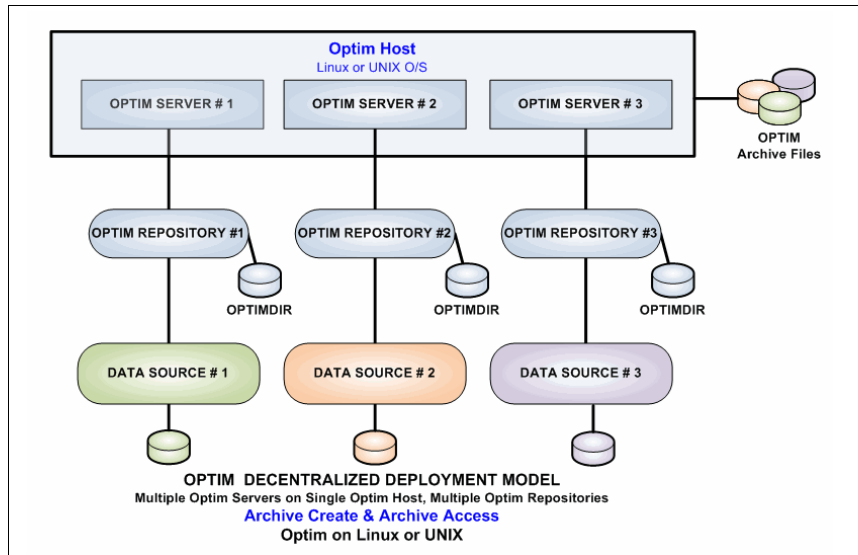


Figure 4-19 One Optim host, multiple Optim server instances, each with its own repository

## Scale the number of cores and RAM on the Optim host

### Key architectural considerations:

- ▶ Vertical scalability within a single Optim host
- ▶ Capacity
- ▶ Performance
- ▶ Multi-processing
- ▶ Fast growth rate of source data
- ▶ Simplified management of single Optim infrastructure

If you increase the concurrent Optim workload by increasing the number of database connections, the number of concurrent processes, or the number of Optim server instances running on a single Optim host, you might have to supplement the capacity of the Optim host and add processor cores and memory. This scale-up strategy adds hardware processor capacity.

The amount of capacity that you need on your Optim host might depend on the phase of your project (initial bulk archive with large data volumes versus incremental archive with smaller data volumes), number of data sources, frequency of processing, data growth over time, and future data archive requirements. If you plan to procure hardware for your Optim host, consider a system that can be upgraded to include more cores and memory if you anticipate archiving more data or expanding the solution to other data sources in the future.

For example, if your initial deployment for archiving one data source is a 1 x six core Intel server (six cores), make sure that it can be upgraded to a 2 x six core (total twelve cores).

Figure 4-20 illustrates adding processor cores and memory to the Optim host as the number of concurrent processes and Optim server instances is increased.

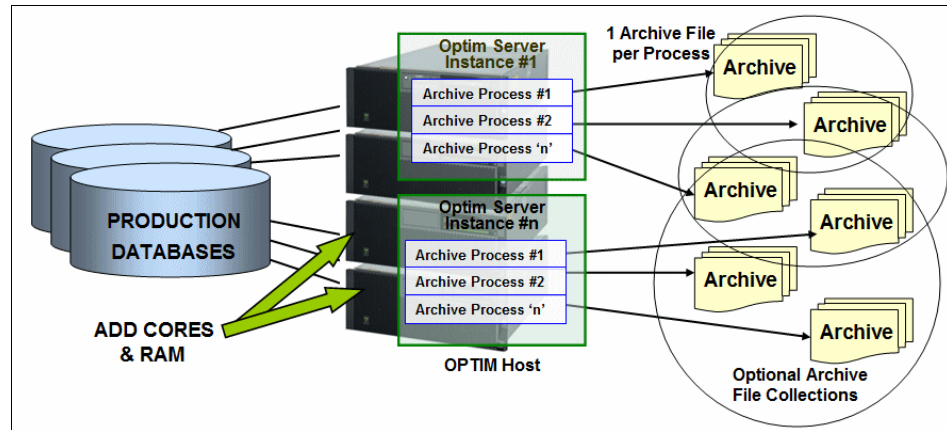


Figure 4-20 Scaling the Optim host

## Scale the number of Optim hosts

### Key architectural considerations:

- ▶ Horizontal scalability across multiple Optim hosts
- ▶ Capacity
- ▶ Performance
- ▶ Data volumes
- ▶ Multiple data centers
- ▶ Enterprise-wide data growth initiative

### ***Optim on Linux, UNIX, and Microsoft Windows***

If further scalability is required, you can scale out to multiple Optim hosts (physical or virtual), providing exceptional, flexible, and scalable configuration options to satisfy your requirements. For example, certain clients have standardized on specific hardware platforms and configurations for their Optim deployment, such as 4-core Intel blades in a server farm. This extensible deployment model satisfies both the need for fast throughput for Optim processing while utilizing the client's standard operating environment hardware.

Figure 4-21 on page 138 shows a decentralized deployment model in which two Optim hosts have a separate instance of Optim server, but both hosts share a

single Optim repository that contains the metadata for both Optim servers and multiple, heterogeneous data sources. Each Optim server instance executes all data growth processing, including archive create, delete source data after archive, archive access, and archive restore. Because this example has a single Optim repository, it typically is utilized when both Optim servers are in a single data center location. This architecture provides additional processor capacity while maintaining a centralized view of the repository data. Query users on either host can read archive files, irrespective of the specific host that archived the data.

This “scale-out” strategy increases the processing capacity and throughput for large data volumes, multiple data sources, and constrained operational windows.

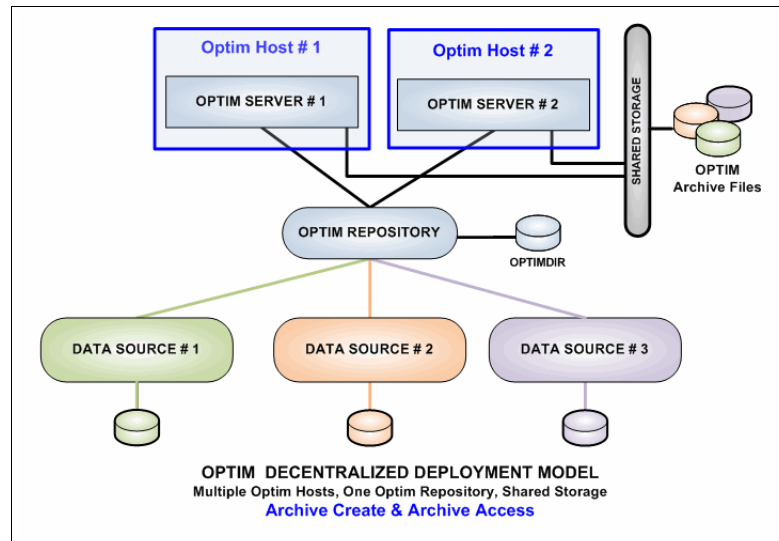


Figure 4-21 Scale the number of Optim hosts, one Optim server per host

Figure 4-22 on page 139 shows a variation and extension of the decentralized deployment model archiving solution for a single, exceptionally large data source within a constrained operational window. The Optim infrastructure is spread across three Linux or UNIX processors, each with two Optim Server instances. Metadata is published and synchronized between the two repositories, and storage is shared across the servers.

This configuration is suitable, for example, for any company that collects voluminous transaction detail records each day. It is important to archive and manage the data growth to ensure the continued performance of their production application systems. Again, this approach provides a scale-out strategy to increase throughput.

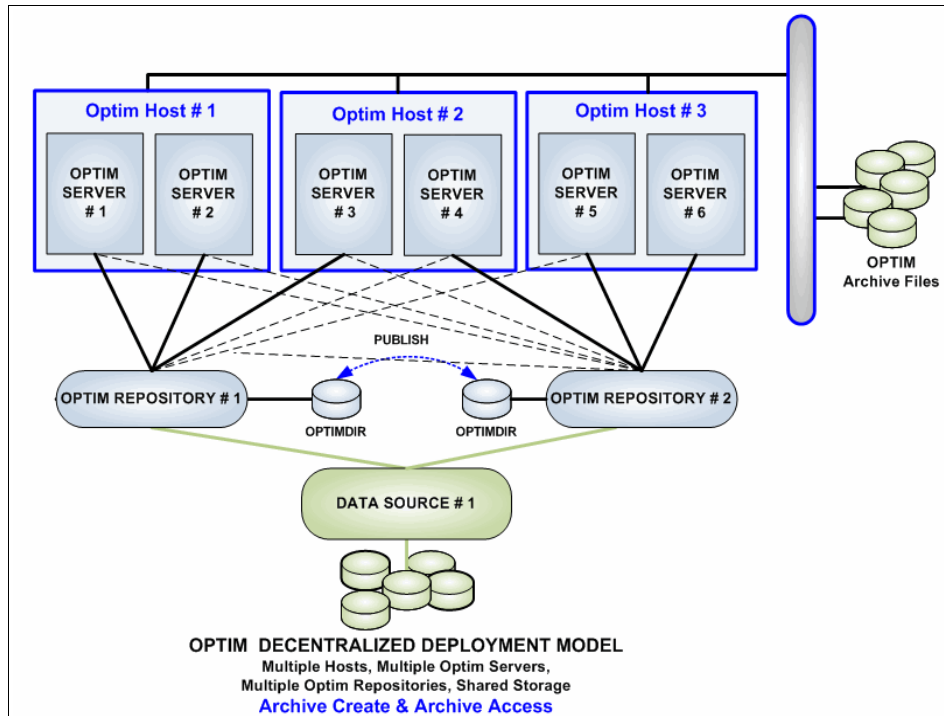


Figure 4-22 Multiple hosts, each with multiple Optim server instances

Figure 4-23 on page 140 illustrates an implementation with multiple types of extensibility:

- ▶ Parallelism: Increasing the number of database connections for each archive or delete process
- ▶ Multi-tasking: Multiple concurrent archive processes on each Optim server instance
- ▶ Multiple Optim server instances
- ▶ Multiple Optim hosts

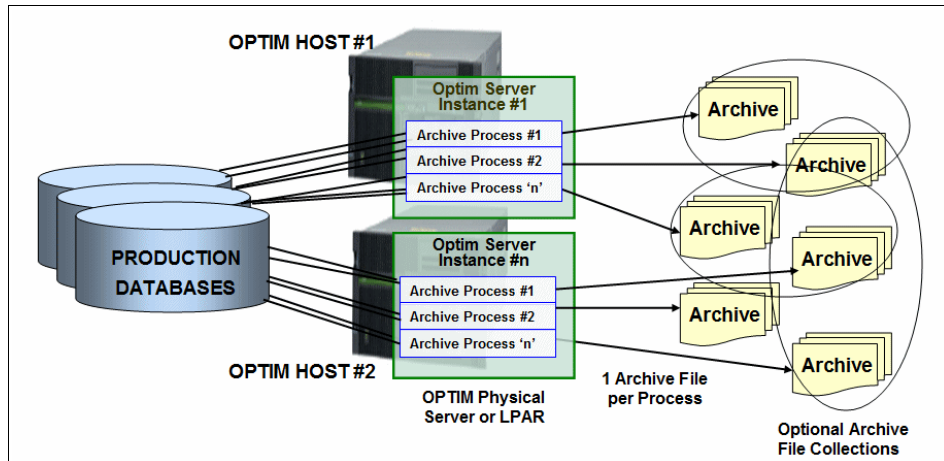


Figure 4-23 Extensibility of reference deployment models

### Optim on z/OS

When Optim is installed on z/OS, scaling out typically occurs on LPARs, instead of on System z physical servers. Optim is generally collocated in the same LPAR as the source DB2 z/OS data. However, if Optim accesses data in another LPAR that does not have Optim, Optim can use the services of DB2 z/OS Distributed Relational Database Architecture™ (DRDA®) and Distributed Data Facility (DDF) to connect.

### Scaling down

So far, we have discussed strategies to increase throughput. However, there might be a scenario in which resources are constrained (source data host, Optim host, network, or storage), and budget constraints preclude hardware upgrades. In this case, you might have to deploy a processing strategy that will not cause a bottleneck in the overall system.

**Key architecture considerations:**

- ▶ Limited Optim host processor capacity (for example, limited number of processor cores or slow clock speed)
- ▶ Budget constraints
- ▶ Incremental archive data volume is significantly less than initial archive
- ▶ Migrating off a platform or retiring an application
- ▶ Requirement to throttle Optim throughput due to constraints in other parts of the system (for example, network, storage, or source data host)
- ▶ Time is not the predominant driver

For example, during the initial archive phase, instead of executing several extremely large archive jobs concurrently, you can break up those archive requests into smaller data volumes and run one or a few over a longer period of time. The advantage is that you will lower the overall resource consumption during execution and decrease resource costs, but it will take longer to complete the workload.

Another consideration is the clock speed of your Optim host. If you use an older, slower server, you might have to scale back on part of the concurrent workload to avoid a CPU bottleneck.

If any resource is constrained, you might want to adjust the number of concurrent Optim processes, number of parallel database connections, and number of Optim server instances. For example, if your source database server is up 24x7, peaking at 90% utilization, and there is no discernible batch processing window, you can break up the data into smaller batches and run them throughout each day to minimize the impact on other constrained resources.

When budget is a concern, you can size the Optim host for the smaller, incremental, ongoing data volume instead of for the initial, bulk archive. If your incremental processing occurs monthly, for example, and takes one day to complete, you can use the Optim host's capacity on the other 29 days of the month to gradually archive the initial bulk data. It might take longer, but you will be able to archive it with fewer resources. We discuss sizing in more detail in Chapter 5, "Sizing the Optim infrastructure" on page 173.

## High availability

### Key architecture considerations:

- ▶ Critical operational windows
- ▶ Resiliency of infrastructure
- ▶ Throughput requirements
- ▶ Business value of archived data
- ▶ Cost
- ▶ Critical time constraint to archive and purge data from source

For clients that require high availability<sup>2</sup> for their data growth implementation, the following extended, decentralized, deployment model provides an architectural template. The business value of the archived data, which is deemed “warm” or “cold”, means that high availability requirements might not be as strict as the high availability requirements for your “hot” operational data. Balance the value of the archived data, the time it takes to access the data if a failure occurs, and the cost of deploying a high-availability environment.

First, determine if you need high availability for the archive create process, archive access functions, or both, because that decision will affect the architecture. Certain data sources grow so rapidly that they must be archived daily, and organizations cannot afford to get behind on that daily archive process due to the negative performance impact on their production applications. Other clients might have to provide high availability for archive data access for user reporting.

Figure 4-24 on page 143 illustrates an Optim high-availability deployment in which there is no single point of failure when the following minimum requirements are implemented:

- ▶ Registered Optim archive files (read-only user data) must be placed on multiple path storage area network (SAN)/network-attached storage (NAS) high-availability devices that provide multiple nodes and RAID types. You can use solutions, such as Tivoli Storage Manager’s High Availability feature.
- ▶ The Optim repository containing the project catalog, archive catalog, and virtualization metadata (a few thousand rows with minimal write and update requirements) must be stored in a high-availability DBMS (database server cluster).
- ▶ At least two Optim server instances are installed, each on a separate host.

---

<sup>2</sup> High availability is a system design approach and associated service implementation that ensures a prearranged level of operational performance will be met during a contractual measurement period.

- Minimum of two network interface cards (NAS, network) and Fibre Channel (FC) cards (SAN) on each Optim host.

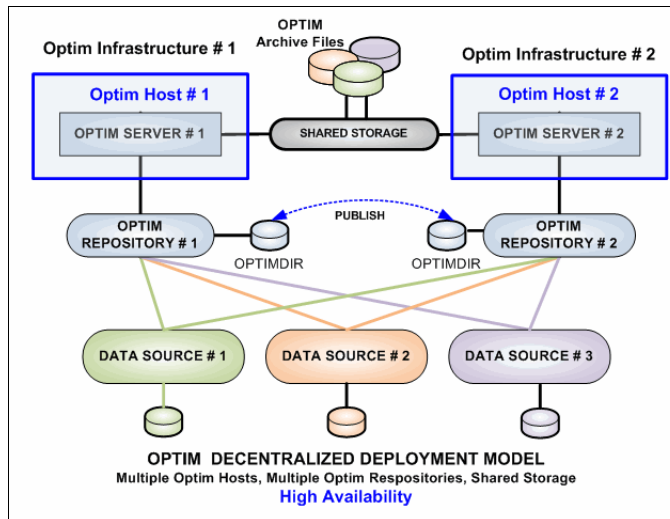


Figure 4-24 High-availability architecture

Optim does not provide the functionality (and associated overhead) for hot-swapping on failover, because many processes are read-only processes. A cold swapping function makes Optim available in seconds, but it does not have automatic restart of active jobs. Delete supports cold restart in addition to standard database checkpoints. Optim is architected to have a short start-up latency (0.05-1.5 seconds is typical).

There are both disruptive and non-disruptive Optim processes, and it is important to differentiate between the two for high availability:

- Non-disruptive processes

A non-disruptive process must run until completion; if something interrupts the execution (for example, system failure), the process must be restarted from the beginning when the system becomes available. Non-disruptive processes include archive creation, archive file index creation, and archive access (reading archive files). In a high-availability architecture, there is a minimum of two Optim servers. If one Optim server fails, Optim can be cold-swapped to the second server, and the requests can be restarted immediately.

► Disruptive processes

Disruptive processes, however, can be restarted based on the last successful commit point, which is stored in a log. You can specify the number of rows to process before committing the changes to the database via the commit frequency parameter in the process options section of the delete, insert, or restore request editor, or in the product options. Disruptive processes include delete (post-archive), insert, and restore. If a delete, insert or restore process ends abnormally, it can be restarted to resume processing from the last commit point. In a high-availability environment in which the primary server failed, the process can be restarted from the last successful commit point on another Optim server.

Deletes are performed following the implied or expressed referential integrity constraints. The process is an operational cooperation between the high-availability DBMS (which maintains the data integrity) and the Optim restart process that completes the interrupted delete and restore process.

## Disaster recovery

### Key architecture considerations:

- Resiliency of infrastructure
- Business value of archived data
- SLAs
- Cost

Providing a disaster recovery architecture for access to archived data includes the following actions:

- The storage management subsystem copies critical archive files to the disaster recovery site with interconnected storage controllers.
- The Optim repository database must be replicated at the disaster recovery site and be part of an overall DBMS disaster recovery solution.
- The Optim storage profile must contain the location of the archive files at the disaster recovery site.
- An Optim host must be provisioned at the disaster recovery site with the following information:
  - Optim server instance
  - Optim repository (synchronized with production repository)
  - ODM
  - DBMS for Optim repository
  - Copy of archive files

Figure 4-25 on page 145 is an example of a disaster recovery architecture.

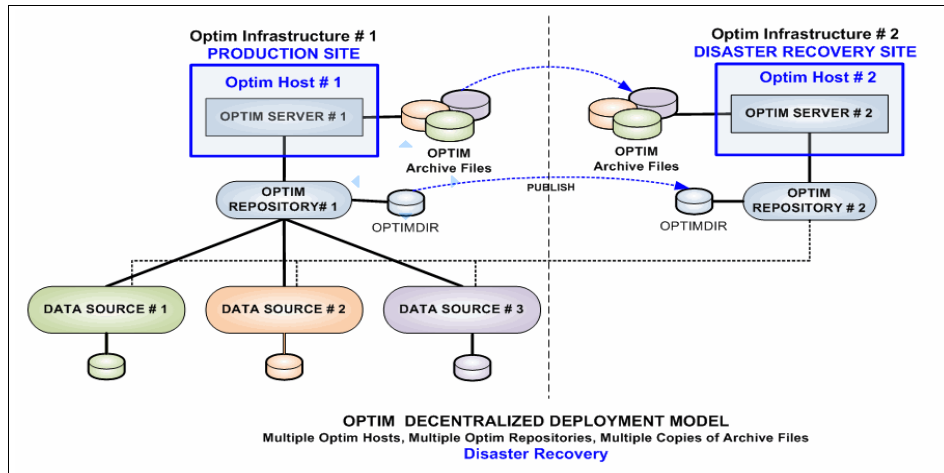


Figure 4-25 Disaster recovery architecture

There are several performance and storage considerations for disaster recovery:

- ▶ Cold failover from the production data center to disaster recovery data center.
- ▶ Unless otherwise required, the Optim host at the disaster recovery data center is generally sized at 50% of the capacity of the production Optim host, unless critical archive processing must continue uninterrupted.
- ▶ Refer to “Storage” on page 153 for information about creating backup copies of your archive files for your disaster recovery data center.
- ▶ If you do not replicate the archive file index at the disaster recovery site, Optim still allows SQL access to archived data, but each query will result in a slower and potentially I/O-intensive table scan. Indexes can be rebuilt from an archive at any time at the disaster recovery data center, if required.

Similar to the high-availability architecture, Optim does not provide functionality (and associated overhead) for hot-swapping on failover, because most processes are read-only processes. A cold swapping makes Optim available in seconds, but it does not have automatic restart of active jobs. Delete supports cold restart in addition to standard database checkpoints. There are both disruptive and non-disruptive Optim processes, as previously described for high availability.

Optional connections from the disaster recovery site can be established to the data source hosts to continue regularly scheduled archive activity if a long outage is anticipated. Network latency will impede throughput, because large volumes of data will be transmitted over the network from the data source to a remote disaster recovery site. However, this capability might not be necessary

due to the nature of archived data (whether it is considered either warm or cold data) and the archived data might not be critical to the daily business processes.

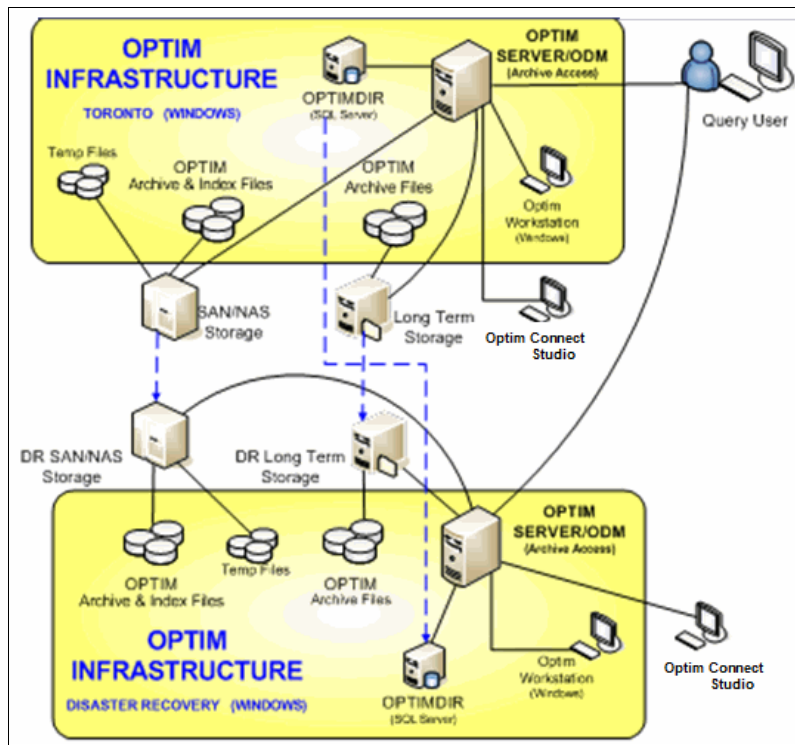


Figure 4-26 Another view of a disaster recovery architecture on Microsoft Windows

Figure 4-27 on page 147 depicts an Optim disaster recovery architecture on POWER7™ servers and AIX.

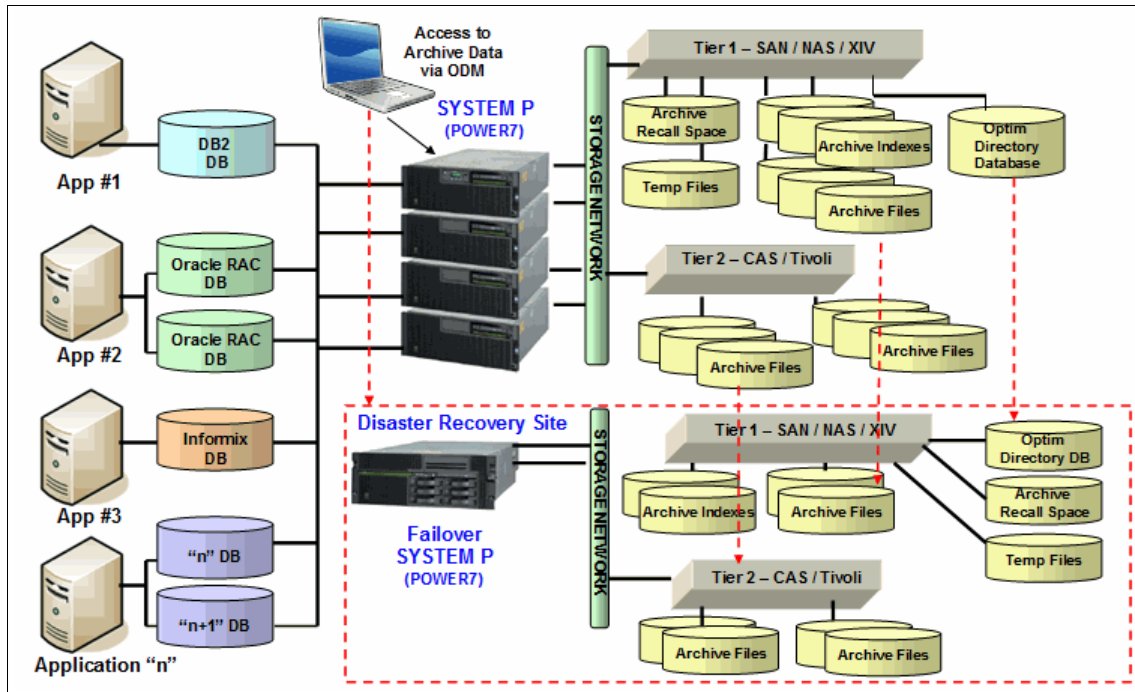


Figure 4-27 Disaster recovery on POWER7 servers and AIX

### 4.3.5 Hardware components

After the software components have been identified (4.3.2, “Software components” on page 115) and the deployment model has been developed (4.3.3, “Reference deployment models” on page 117), the next step is to determine where to install those software components on the hardware in the Optim infrastructure. The deployment model that you have developed provides the framework.

The host on which Optim is installed can be a dedicated physical or a virtualized (for example, VMware, LPAR) server. If a virtual server is deployed, you need to account for normal contention and resource allocation spikes during processing periods.

The Optim infrastructure includes the following hardware components:

- ▶ Optim host
- ▶ Optim workstation
- ▶ Middleware host
- ▶ Source data host (for example, database server)

- ▶ Networks
- ▶ Storage subsystems (for source data, archive files, and temporary storage)

When architecting the solution, all of the hardware components need to be considered so that the entire system, end-to-end, is balanced and without bottlenecks. For example, increasing the capacity of the Optim host might not improve throughput if you have a bottleneck on the network that connects the database server to the Optim host.

#### **Key architectural considerations:**

- ▶ Throughput requirements (volume and time)
- ▶ Processor speed
- ▶ Standard operating environment (SOE)
- ▶ Processing concurrency (scale number of processor cores and memory)
- ▶ One or multiple data centers
- ▶ Dedicated or virtual host
- ▶ Processing strategy: Concurrent or sequential
- ▶ Operational window and processing frequency
- ▶ Where the software components will be installed
- ▶ Scalability and flexibility
- ▶ High availability and disaster recovery
- ▶ Platform and OS for Optim host

### **Optim on distributed platforms**

Table 4-9 on page 149 lists the key software components that are used by Optim that is installed on Linux, UNIX, or Microsoft Windows. It summarizes various hardware options, providing the advantages and disadvantages of each option. Table 4-9 on page 149 enables you to align the solution architecture with your various requirements.

For instance, if your IT standards dictate that applications cannot run on a database server, the Optim repository must not be installed on the Optim host, but on a database server. Alternatively, if time is critical, network latency is high, and there is spare capacity on the source data host, consider installing Optim on the source data host.

Table 4-9 Hardware options for the Optim software components on distributed platforms

Software component	Install software on	Advantages	Disadvantages
Optim server instance	Dedicated Optim host	Can tune for best performance; better ability to meet SLAs.	Server capacity might be idle if archive create and archive access processing is infrequent.
	Multiple Optim hosts	Scale out for extra capacity, faster throughput, high availability, and disaster recovery; recommended if data sources are in multiple data centers with at least one Optim host in each data center.	Potentially higher hardware cost.
	Separate Optim hosts for archive create and archive access	Separation and tuning for specific workload; separation and added security based on roles and responsibilities; better availability. Recommended for large, enterprise deployments.	Potentially higher hardware cost.
	Virtual Optim host	Uses minimum resources if infrequent archive create and archive access processing.	Overhead for virtualization; less control over consistent performance; potential contention for shared resources; not advisable for large volumes or frequent processing.
	Source data host	Avoids network between source data host and Optim host; best performance if network latency is an issue.	Extra capacity required on source data host, might affect application performance if constrained; application and database functions on same server; extra paths to storage devices might be required.
Optim repository (optimdir)	Optim host	Avoids network hop when accessing repository.	Database installed on a server primarily dedicated to Optim application processing.
	Database host	Separation of application and database server hosts; only infrequent and light access by Optim server.	Network hop from Optim server.

Software component	Install software on	Advantages	Disadvantages
Middleware federation on Linux, UNIX, and Microsoft Windows (IBM InfoSphere Federation Server or Oracle DG4ODBC)	Dedicated host	Can tune environment for specific workload.	Add extra network hop and increased network latency.
	Optim host	Avoids extra network hop; generally low processor utilization.	N/A
	Source data host	Avoids extra network hop; generally low processor utilization.	N/A
Middleware adapter on mainframe (Optim Connect on z/OS or InfoSphere Classic Federation)	Collocate with Optim and database LPAR on z/OS	Generally recommended for best performance.	N/A
Middleware adapter on Linux, UNIX, and Microsoft Windows (Optim Connect on Linux, UNIX, and Microsoft Windows)	Dedicated host	Can tune environment for specific workload.	Extra hardware cost; two extra network hops (data source host → middleware federation host → Optim host); network latency.
	Optim host	Avoids extra network hop; generally low processor utilization.	N/A
	Source data host	Avoids extra network hop; generally low processor utilization.	N/A
DB2 Connect	Collocate on DB2 z/OS LPAR	Preferred or required.	N/A

Software component	Install software on	Advantages	Disadvantages
Open data manager	Optim host	Required collocation with Optim server instance.	N/A
	Single Optim host for archive create and archive access	Centralized deployment model; might be less expensive.	More difficult to provide consistent performance for user queries due to concurrent archive activity.
	Separate Optim host for archive access	Optimized deployment model, separation of roles and responsibilities between users and administrators; can tune specifically for query environment; more consistent performance.	At least two Optim hosts; need to publish metadata from archive create host to archive access host.

## Optim on z/OS

When Optim is installed on z/OS, it is generally installed in an LPAR and collocated in the same LPAR as the DB2 for z/OS database. Because there is no middleware involved when archiving DB2 for z/OS data from Optim on z/OS, there are only two software components that have to be installed and configured: Optim and ODM. Refer to Table 4-10 on page 152 for a description of these options and the benefits and the disadvantages of each option.

Table 4-10 Hardware options for the Optim software components on z/OS

Software component	Install on hardware	Advantages	Disadvantages
Optim	Dedicated LPAR	Off-loads Optim work from database LPAR.	Still need Optim stub in DB2 z/OS LPAR.
	Same LPAR as DB2 z/OS data source	Recommended.	N/A
Open data manager	Optim LPAR	Required collocation with Optim.	N/A
	Single LPAR for Optim archive create and archive access	Centralized deployment model.	Might be more difficult to provide consistent performance for user queries due to concurrent archive activity.
	Separate Optim LPAR for archive access	Optimized deployment model, separation of roles and responsibilities between users and administrators; can tune specifically for query environment; more consistent performance.	At least two Optim hosts. You need to publish metadata from archive create host to archive access host.

## Network

Network performance is critical when Optim is installed on Linux, UNIX, or Microsoft Windows due to the client/server architecture, because large amounts of data are transported across the network. Performance of the network is limited by the slowest component in the stack: network interface cards; FC adapters; routers; switches; bandwidth; firewalls; number of network hops; and so on. Even though there might be high bandwidth, any slow component, for example, the speed or number of network interface cards (NICs), on the Optim host will affect the overall throughput. NICs and FC adapters need to be as fast as possible, and you need to provision multiples, depending on the volume of data and operational window.

If Optim server is installed on a separate host than the data source, we prefer collocating them on the same subnet. Frequently, the slowest link in a distributed application is network latency. It is not advisable to archive across a wide area

network (WAN), when the source data is at a remote data center from the Optim host, especially if large amounts of data are being archived.

Allow for extra network bandwidth for peak spikes and to accommodate bidirectional communications. When calculating network requirements, consider the number of database connections, the number of concurrent Optim processes executing, the number of Optim server instances, and other concurrent workload on the network because all of this workload will add to the network load. Include your network specialists in discussions regarding network requirements.

## **Storage**

You need to align the type and quantity of storage that is required for your data growth implementation with your corporate information lifecycle management (ILM) strategy, which assigns business value to various sets of data. Data can be classified based on various criteria, such as age of data, access, availability, recovery, compliance, retention, data mining, security, or discovery requirements. Over time, the business value of the data can change, therefore affecting the type and cost of storage that is deployed.

After your data has been categorized logically, you can establish storage management policies around these classifications to develop a cost-effective storage topology. You can classify data as it relates to business rules and data usage. Service levels determine the execution model, including storage cost, data retention, and disposal, along with data access and recovery objectives. The execution model, in turn, defines the tiered storage services and rule-based implementation of the storage management process. Implementing a tiered strategy allows you to reclaim expensive tier 1 storage capacity and maximize the value of your existing storage infrastructure.

**Key architectural considerations:**

- ▶ Business value of enterprise application data based on business needs (hot/warm/cold data), regulatory compliance, data management policies, and service levels
- ▶ Data retention requirements: Number of years, number of copies, compliance, and end of life (shred)
- ▶ Storage capacity based on data volume and growth rate (including organic business growth, growth rate, mergers and acquisitions, and data multiplier effect), archive file compression, retention strategy, temporary storage during processing, RAID level, and reporting database
- ▶ Data archival and retrieval requirements: Service level and performance required for archive create, access, and restore during the life cycle of the data, and archive file indexing for query performance
- ▶ Information lifecycle management policies: Tiered storage strategy, retention requirements due to business processes or compliance, audit control, and delete strategy
- ▶ Available and installed storage technologies to support your strategy

Optim offers the capability to cost-effectively manage data throughout its life cycle in your organization. It can move large amounts of data from your source database to immutable compressed archive files, has the ability to read data from these files, and if needed, restore the archived data to a database.

Optim can manage data in a heterogeneous storage environment, such as storage area network (SAN), network-attached storage (NAS), content-addressable storage (CAS), such as IBM Information Archive, tape, and optical devices. Optim supports NetApp SnapLock and Symantec Enterprise Vault for all generated file types. Optim supports backup devices for archive files, such as EMC NetWorker, IBM Tivoli Storage Manager, and EMC Centera.

Optim on distributed platforms creates and accesses archive files on SAN or NAS devices. The archive files are tracked in the Optim directory (`optimdir`) until you decide to manually de-register and purge the archive files. Alternatively, you can automate the retention process and thereby automatically de-register and delete the archive files by creating a storage profile for the archive files.

An Optim storage profile enables you to define a storage policy where it is possible to perform these tasks:

- ▶ Create a long-term retention policy for the archive file.
- ▶ Create a copy of an archive file on a device that is classified as Write Once, Read Many (WORM). For example, you can automatically create an archive

file copy on a CD/DVD or on a SnapLock in order to meet your retention and preservation policies. This copy is automatically registered in the current `optimdir`. If the storage device supports a retention policy, Optim sets it accordingly.

- ▶ Create a copy of the archive on a backup device, such as Tivoli Storage Manager, EMC/Centera, Hitachi, and so on. You can use this option, for example, to ensure that a secondary copy exists on a long-term device, thereby reducing IT requirements for additional backup and recovery procedures. This copy is automatically registered in the current `optimdir`. If the device supports a retention policy, Optim sets it accordingly.
- ▶ Create a duplicate or fallback copy of an archive file without registering in the current `optimdir`. You can use this mechanism to create an independent copy that can be managed in another `optimdir`, for example, when you deploy an operationally optimized deployment architecture in which your solution has separate Optim infrastructures (and Optim directories) for archive creation and access processes. This archive has no associated retention policy. You also can use this mechanism to create a copy of the archive file, which can be transported to an off-site disaster recovery data center.

When you set the various long-term policies, Optim checks the `optimdir` for any archive that is past the retention policy, and then, it de-registers and purges the archive. Optim defaults to purging the archives that are present on SAN/NAS every 30 minutes. Optim performs the auto-delete on WORM BACKUP devices only once a day to prevent taxing a device that is not necessarily designed for continuous access. Consider that Optim notifies the device that a delete can be performed, but it does wait until the action is performed.

When a WORM or a backup copy exists, you can also set additional short-term retention policies:

- ▶ How long the original archive file is kept on SAN/NAS after the backup copy has been created.
- ▶ When and where Optim has to recall (copy back from the backup device) an archive file when it is needed for access and the original copy is no longer on SAN/NAS.
- ▶ How long to retain the recalled copy on SAN/NAS after the last access date.

If you need to implement a two-tier storage policy, you can either use a WORM or a BACKUP device and set the long-term retention policy (second storage tier) and use the short-term retention policy to manage how long an archive file can be on SAN/NAS (first storage tier).

If you need a multiple-tier storage strategy, you need to use a BACKUP device that supports Hierarchical Storage Manager (HSM), for example Tivoli Storage

Manager or Hitachi. Optim handles the short-term retention policy on tier one storage, the BACKUP device retains the archive based on the long-term retention policy, and HSM demotes and promotes the archive based on the policies that you set for that storage group. You do not need to manage the archive files manually because the backup, restore, demotion, and promotion of archive files across your storage devices can be automated. Optim can also manage the retention and automatic purge of archive files from the environment, providing end-to-end management capabilities throughout the life of your data. Refer to Chapter 10 of the *IBM Optim Archive User Manual* for more information about storage support and storage profiles.

Figure 4-28 depicts an example of how Optim can manage data archives over its tenure. It is important to understand how the business value and cost of storing that data changes over time in your organization and plan for its storage accordingly so that it aligns with your goals and objectives for retention, compliance, and archive access.

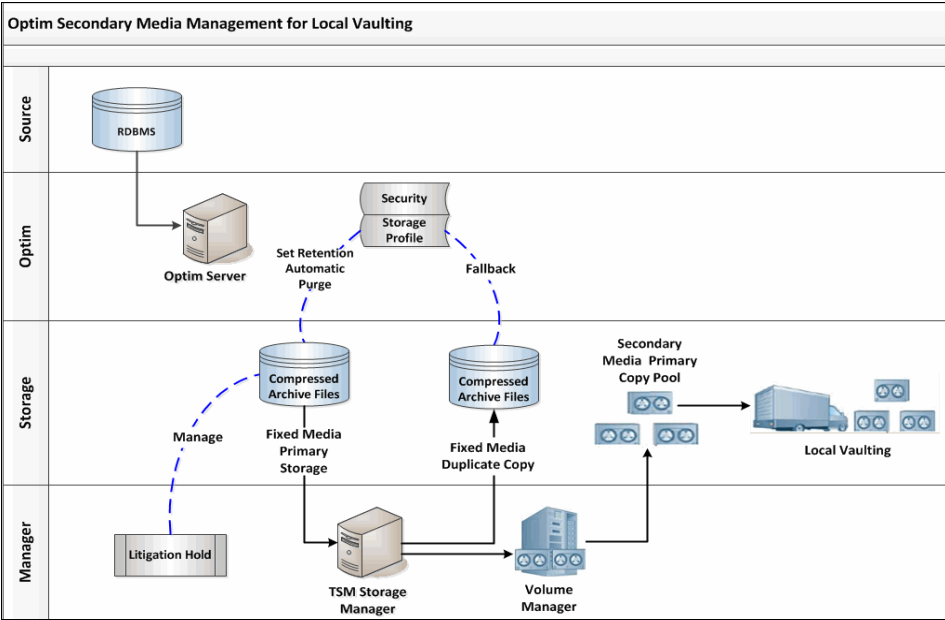


Figure 4-28 Storage management for local vaulting

Figure 4-29 on page 157 illustrates an example of how and where data can be staged over its lifetime. It highlights the important classifications and characteristics of data, storage, and virtualization options within the context of the value of data and cost of data over time. It provides a sample road map to address storage management requirements for your data growth implementation. As archive files age or the value and access requirements of the

data changes over time, you need to move these archives to a slower, cheaper storage option and eventually phase them off to an offline storage device. At the data's end of life, you need to shred the data. Put a suitable policy on data retention and shred in place to complete the life cycle of this data.

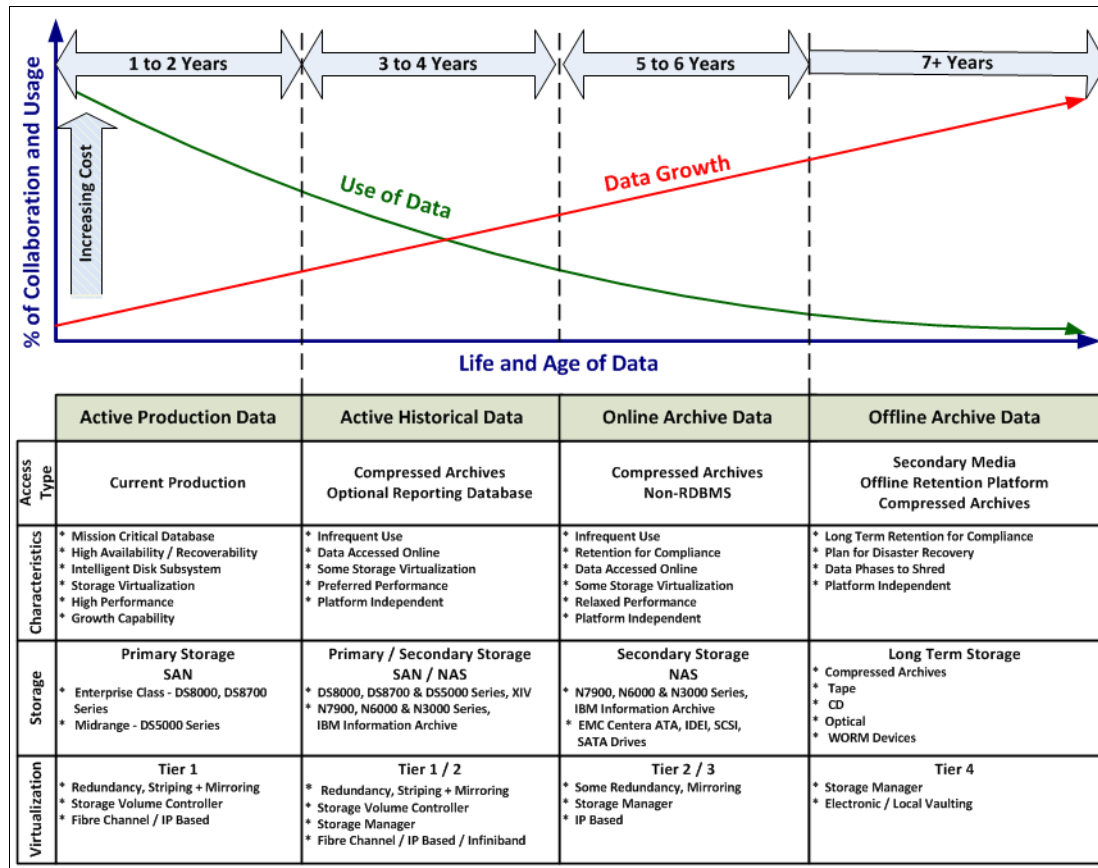


Figure 4-29 Optimize archive files and tiered storage

As you can see in Figure 4-29, data moves from one type of storage to another type of storage depending on access requirements and characteristics. The time frames can vary based on your own objectives. Performance characteristics and storage requirements differ at each stage. So does the storage technology, ranging from most expensive (active production data store) to the cheapest (offline archive).

The type of storage on which your source data resides affects the performance of Optim reads (during archive create) and writes (during archive restore). The storage infrastructure for the Optim files affects the performance for writes

(during archive create) and reads (during archive access and restore). Typically, archive files are written once and read multiple times. A faster storage infrastructure, such as SAN, which is connected over an FC-based or IP-based network, such as Fast/Gigabit Ethernet, will increase overall processing throughput rates.

### ***Active historical data***

When production data is first archived, it generally is considered active historical data, which has specific service levels and business value. When you start your data growth project and initially archive a large amount of data from your application data source, plan to write the archive files to suitable storage based on your requirements. Typically, the fastest storage for archived data is deployed for active historical data.

To improve archive create and access performance, write the compressed archive files to a type of high-performance storage. Use SAN for even better performance gain and segregation from application data network. With NAS, consider the overall impact on the performance of the underlying IP network.

Creating indexes on archive files and placing the indexes on fast storage, such as SAN, can improve archive data access performance.

If archive create and access requirements dictate a strict SLA, high availability, and a constrained operating window, it is best to build high availability, redundancy, and performance in the storage configuration. Various RAID levels, such as RAID 0, 1, 5, or 1+0, can help address performance requirements by striping the data across disks. Various RAID levels can address fault-tolerant requirements by mirroring data across two or more matched disks. A clustered journaling file system, such as General Parallel File System (GPFS™), further improves usability, scalability, resilience, and performance, while hiding the complexity of SAN and disks across multiple sites. The underlying network, FC, or IP forms a major component, which determines the physical I/O path from the CPU to the storage system.

### ***Online archive data***

As the archived data ages, its business value decreases, user access diminishes, and service levels relax, you can move archive files to a more cost-effective storage, such as IBM Information Archive, HP StorageWorks, EMC Centera, or netApp NearStore with SnapLock.

As the data in the archive files ages, two events generally occur:

- ▶ Data in the initial, bulk archive of aged data might not be accessed as frequently. You can move archive files that are identified as aged to a slower storage environment, such as NAS or content-addressable storage (CAS), because performance is not a prime factor. The frequency of access to this

archived data needs to drive the requirement for the type of storage. Consider other requirements to keep this data available on online storage, such as for audit or compliance purposes. If you need access to data that is stored in one of these storage alternatives, you must be able to access the data directly and selectively restore part or all of the data to a reporting database.

At this time, reevaluate the requirements to build high availability, redundancy, and performance in the storage infrastructure, because data access requirements are sometimes more relaxed, along with the service levels. For instance, online archive data might be about five to six years of age and might only be needed for regulatory compliance and occasional queries. These files can potentially be moved to a Tier 2 or Tier 3 storage type or a virtual archive to control cost and provide availability at a reduced throughput rate. Again, a volume controller and a clustered journaling file system provide usability and scalability. RAID 0 can be used for this type of data access, as needed. An IP-based storage network, such as iSCSI or Ethernet, can provide the needed throughput and good use of network and processor resources.

- The process of executing ongoing, incremental archives to keep the size of the production database under control begins. These incremental archive files need to be stored on a high-speed storage network, such as SAN or NAS, which is similar to the storage where the initial archives were created. The storage that you allocated for the initial archives can be reused for this purpose if the initial archive data has been moved to cheaper storage. Create indexes on the archive files for optimal access and better performance. Again, archive index files need to reside on SAN for better performance.

### ***Offline archive data***

Eventually, you can phase the part of the archive files that are rarely accessed, have less business value, or do not have strict service levels to tape or a suitable offline storage device for as long as the data must be retained. You can restore the data to disk if needed.

Suitable storage policies for data retention that are based on business and compliance requirements and shredding at the end of the data's life must be in place to complete the life cycle of this data. A storage solution, such as Tivoli Storage Manager, can help in managing backup, recovery, archive, disaster recovery, space management, and record retention.

### ***Other storage considerations***

Archive file indexes must reside on separate disks than the associated archive file. In addition to storage for archive files and archive file indexes, the Optim engine requires temporary storage for control files, trace files, and log files, which must reside on separate disks from the archive files, preferably on a SAN, for best performance.

The type of data that Optim reads and archives can affect throughput. For example, if there is large object (LOB)-type data in your database or file attachments that are stored outside of the database, data transfer rates might be affected due to the physical nature of I/O.

Archives are immutable, so after they have been backed up, they do not require dynamic disk recover ability (RAID 1 or greater). Striping a disk with RAID-0 is a performance option for archive files and archive file indexes, especially when SLAs are critical.

### ***Storage virtualization***

*Storage virtualization* provides a dynamic pool of storage from which disk space is presented to the operating system as a set of logical volumes that appear to be normal disks as far as the operating system and applications are concerned. Storage virtualization helps address the increasing cost and complexity in data storage management. Virtualization solutions can be implemented in the storage network, the server, or the storage device. Implementing virtualization in your SAN and NAS environment, and therefore in the path of input and output (I/O) activity, helps provide a solid basis for policy-based management. Storage virtualization is good for scalability and manageability and creates a platform where information can be managed as time progresses. It also separates the physical and logical aspect of storage.

A virtualized storage solution supports freedom of choice in storage-device vendor selection. You can use a suitable volume controller to reduce both the complexity and cost of managing your SAN-based and NAS-based storage system, provide simplicity in management, and improve application availability and productivity. You can use a suitable volume controller to provide redundancy in the nodes to avoid a single point of failure. You can virtualize disks further by introducing RAID. A disk subsystem with a RAID controller offers greater functional scope through performance by striping and increasing fault tolerance by introducing redundancy in the disks. The choice of the underlying network and I/O techniques will determine the data transfer between servers and storage devices.

### ***Location of archive files***

When architecting a solution using the operationally optimized deployment model in which there are separate infrastructures for archive create and archive access, it is important to consider where the archive files and indexes are located.

Figure 4-30 on page 161 illustrates an example of an operationally optimized deployment model in which the archive create infrastructure and the archive access infrastructure are located in the same data center, sharing a single copy of the archive files. Metadata from the Optim repositories #1 and #2 in infrastructure #1 (where the data was archived) is published to repository #3 in

infrastructure #2 (where the data will be accessed). Both repositories know where the archive files reside, and the archive file indexes can be created by either Optim server #1 or #2. The benefit of this architecture is that the archive files and indexes are stored centrally and storage capacity is kept to a minimum.

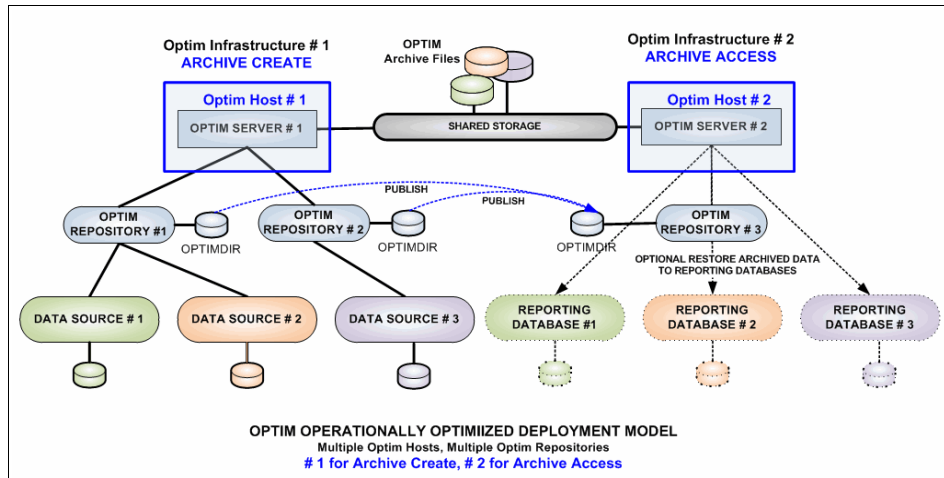


Figure 4-30 Operationally optimized deployment model: Shared storage

Figure 4-31 depicts an operationally optimized deployment model for separate archive create and archive access infrastructures that are completely separate, and potentially in geographically disbursed data centers. Archive files are replicated in infrastructure #2 for archive retrieval.

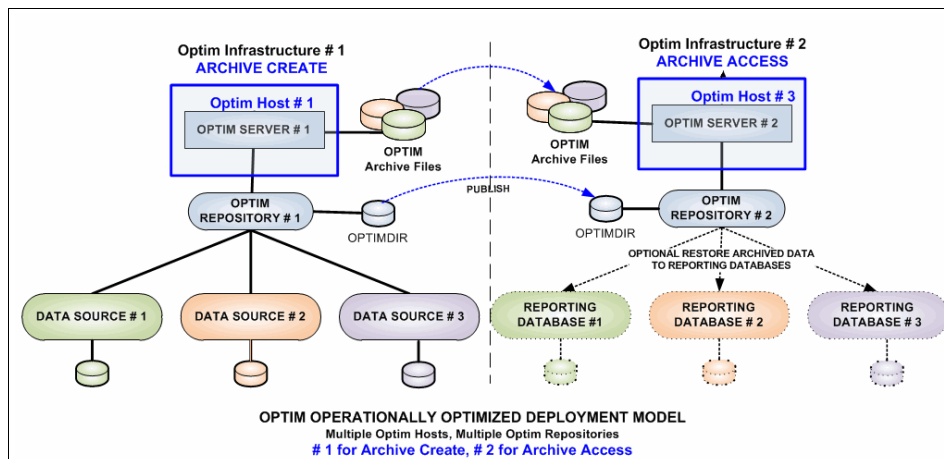


Figure 4-31 Operationally optimized deployment model: Separate storage

This approach offers these advantages:

- ▶ It provides a complete separation of the archive create and archive access infrastructures.
- ▶ You can tune each infrastructure specifically for its intended purpose.
- ▶ More consistent SLAs can be delivered for user query access to the archive files
- ▶ Corporate archive file storage and access can be centralized even if the data sources are distributed across multiple data centers.

After the archive files are transferred to the archive access infrastructure, the indexes can be created. The archive files on infrastructure #1 can be kept (if restore processing is needed), or they optionally can be retained or deleted to reduce storage capacity, depending on your requirements.

If you archive large amounts of data (in the 10 TB to 15 TB range) and Optim is installed on Microsoft Windows, be aware that Microsoft Windows has to see all of the mount points before it will reboot. So if you have a large volume of archive file data, it might elongate server boot time to the point of being impractical and perhaps unmanageable. Optim on Linux or UNIX is a preferable platform for large amounts of archived data.

### **4.3.6 Aligning the solution architecture with the Optim design and planned process flow**

The next step in developing the solution architecture is to ensure that the infrastructure supports the Optim design and the planned processing workflows. How the data is ingested by Optim, the workflow during the archive and delete processing, and user access to the archived data can affect the overall architecture.

Consider the following scenario. A telecommunications company has to archive large amounts of call detail records on a daily basis that are delivered in flat files from several systems. After researching the situation, the company determined that there will be approximately 10,000 files a day to be archived. The Optim infrastructure has to scale to accommodate a large volume of data in a short amount of time. However, another consideration, which affects the design, is that users who will query the archived data will need to access the data on a monthly boundary. In other words, up to 30,000 archive files (average of 30 days in a month times 10,000 files a day) might need to be opened and queried to satisfy a single user query. This design will probably not meet the user SLAs.

To optimize the performance, both from an archive create and an archive access perspective, a staging database is architected between the source data and the Optim host. Archives are executed daily and restored to the staging database,

which keeps accumulating the data for a month. Data is archived again from the staging database and aligned to the way that the users will query the data, for example, by month. Because the archive create processing occurs daily and query activity is anticipated to be moderately heavy, the company created separate Optim hosts for archive create and archive access (operationally optimized deployment model).

Consider another telecommunications company scenario in which Optim has to archive three billion records, around 780 GB, of call detail records from a relational database in three hours. The data is partitioned, so once the data has been archived, the partition can be dropped, thereby eliminating the Optim delete processing and efficiently decreasing the size of the production database. The architecture for this deployment scaled out to multiple quad-core Intel blade servers, each running four instances of Optim server.

If the design and process flow of archive, delete, restore, and access will be scaled by increasing the number of database connections, number of concurrent processing, or number of Optim server instances to archive the throughput required, be sure to consider that increased concurrent workload when developing the architecture.

### **4.3.7 Viability assessment**

Multiple stakeholders, including users, systems administrators, database administrators, application subject matter experts (SMEs), business leaders, data center management, and auditors, need to review the solution architecture. IBM offers architecture reviews, such as the Technical Delivery and Assessment Review, to both ensure the viability of the architecture and your readiness to deploy the solution.

### **4.3.8 Preliminary hardware sizing estimate**

During the design phase, you can develop a high-level, preliminary sizing for the hardware resources to use for planning purposes. Refer to Chapter 5, “Sizing the Optim infrastructure” on page 173 for more information about sizing.

## **4.4 Configure**

During the configure project phase, the solution architecture is installed and configured in your environment.

## 4.4.1 Installing and configuring Optim

Next, you install and configure the software components that were identified in 4.3.2, “Software components” on page 115 on specific hardware components (see 4.3.5, “Hardware components” on page 147) according to the solution architecture. Table 4-11 summarizes the various options that are available when installing and configuring Optim.

Table 4-11 Information to consider when configuring an Optim Data Growth Solution

Summary	Optim on Microsoft Windows	Optim on Linux or UNIX	Optim on z/OS
Data sources	Refer to current Optim documentation for the supported data sources.		
Deployment models	<ul style="list-style-type: none"><li>▶ Centralized</li><li>▶ Decentralized</li><li>▶ Operationally optimized</li></ul>	<ul style="list-style-type: none"><li>▶ Centralized</li><li>▶ Decentralized</li><li>▶ Operationally optimized</li></ul>	<ul style="list-style-type: none"><li>▶ Centralized</li><li>▶ Decentralized</li><li>▶ Operationally optimized</li></ul>
Parallel database connections	Yes	Yes	No
Number of concurrent processes per Optim instance	24 (server) 48 (local)	24	N/A
Number of Optim server instances under single operating system image	1	1 to many	1
Number of Optim repositories per Optim instance	1 to many	1 to many	1 to many
Share Optim repositories across Optim instances	Yes (Linux, UNIX, and Microsoft Windows)	Yes (Linux, UNIX, and Microsoft Windows)	Yes (z/OS)
Move archive files to another Optim server infrastructure	Yes (Linux, UNIX, and Microsoft Windows)	Yes (Linux, UNIX, and Microsoft Windows)	Yes (z/OS to z/OS, or z/OS to Linux, UNIX, and Microsoft Windows using <b>/migrate</b> )
Number of Optim hosts	1 to many	1 to many	1 to many
Virtualization supported	Yes	Yes	Yes
High availability (cold failover)	Yes	Yes	Yes
Disaster recovery (cold failover)	Yes	Yes	Yes

Refer to the *Optim Installation and Configure Guide* for detailed information.

## 4.4.2 Testing, measuring, tuning, and validating

After the Optim infrastructure is set up, prior to the full production rollout, you need to run functional and performance tests using your source data within your data center, so that estimates and assumptions can be validated and adjusted, if necessary. For example, you might have to test the compression ratio of your actual data before determining specific storage capacity requirements, or test the impact on the source database and network when you increase the number of parallel database connections, or measure actual performance and throughput within your IT ecosystem. Based on results, you might need to make certain modifications to the architecture.

### Tuning

In describing the extensible deployment models, it might appear that there is only a trade-off between time and cost (resource). However, tuning is a effective means to achieve faster throughput at a lower cost. Tuning efforts apply to all hardware and software components in the solution architecture and the design and work flow of Optim processes:

- ▶ Source data and Optim hosts (number of cores, processor speed, memory, number of network interface cards, number of FC adapters, and so on)
- ▶ Source database (reorganization, catalog statistics, indexes, buffers, and so on)
- ▶ Storage (paths to storage, speed of storage, placement of data on disk, separation of data and indexes on separate disks, and so on)
- ▶ Network (all components in the network stack)
- ▶ Optim (configuration parameters, buffers, access definitions, and so on)
- ▶ Workload concurrency

## 4.4.3 Hardware sizing

During testing, you need to review the sizing for your production hardware based on specific performance metrics that you have obtained during the test phase.

## 4.4.4 Modifications as necessary

Sometimes, requirements change during the project life cycle. Data volumes can change, operational windows can shrink, or user access requirements can

expand. Review any changes to requirements that might affect the architecture and make the necessary modifications.

## **4.5 Deploy**

After Optim has been deployed in production, continue to monitor and tune the infrastructure as you do for any database application, including the source data host, network, storage, Optim host, user queries, and process design. We discuss deployment and performance in more detail in Chapter 9, “Deployment and operational considerations” on page 375 and Chapter 10, “Optim performance considerations” on page 405.

### **4.5.1 Architecting for future phases**

Optim is architected to provide a flexible, scalable infrastructure that can both meet current needs and also grow to accommodate future data archiving projects. Various options have been presented to scale up, scale out, and scale down both the hardware and software, including capabilities for high availability and disaster recovery. Architecting a flexible and scalable infrastructure that responds to evolving business requirements is key for any data growth management initiative and is a critical component of an enterprise information lifecycle management (ILM) strategy.

## **4.6 Operate**

After the data growth solution is in production, you must validate it to confirm that all business, technical, and non-functional requirements have been met. The project manager needs to obtain sign-offs from all project stakeholders.

### **4.6.1 Review project success criteria**

The Optim solution architecture provides the framework for your data growth project, which allows you to design and configure an infrastructure that will satisfy your requirements while providing a flexible, resilient, and scalable infrastructure. After Optim is in production, review the project success criteria as developed in the project management and analysis phases. The solution must have measurable metrics that demonstrate the positive impact of Optim on your business:

- Production application infrastructure cost reduction

- ▶ Application performance improvements
- ▶ Decreased storage requirements
- ▶ Availability of archived data for query, audit, and regulatory compliance
- ▶ Service levels to deliver information from archived data
- ▶ Positive return on investment (ROI)
- ▶ Managing data growth so that historical data is available to users without affecting the performance of the production application

## 4.7 Sample Optim solution architecture

This following sample solution architecture is provided as high-level example of a large deployment of InfoSphere Optim Data Growth Solution. Most deployments have unique requirements in terms of the number and types of data sources, data volumes, operational windows, and standards. Most deployments also have unique requirements for the concurrency of processing, existing IT infrastructure, throughput requirements, and other factors, as described in 4.2.1, “Requirements” on page 99, that can influence the final architecture.

### 4.7.1 Large Optim deployment

This large Optim deployment example illustrates archiving from both distributed platform and mainframe data sources, various data volumes and processing frequencies, various growth rates and operational windows, and multiple archive data access requirements. Table 4-12 summarizes several of the high-level requirements.

Table 4-12 Sample requirements for a large Optim deployment

Requirements	Data sources					
	DB2 (AIX)	Teradata	Oracle (Solaris)	VSAM (z/OS)	DB2 for i (IBM i)	IMS (z/OS)
Database/File size (data)	8 TB	6 TB	5 TB	500 GB	4 TB	2.5 TB
Database size (index)	4 TB	3.5 TB	1.5 TB	N/A	2 TB	1.5 TB
Production data growth	30%/year	24%/year	15%/year	10%/year	24%/year	28%/year
Initial archive (% of source)	20%	35%	N/A	10%	15%	30%
Archive file index (% of source)	N/A	10%	5%	7%	10%	10%
Initial delete (% of source)	N/A. Drop partition	35%	N/A	10%	15%	30%
Initial restore (% of source)	20%	N/A	N/A	%	%	N/A

Requirements	Data sources					
	DB2 (AIX)	Teradata	Oracle (Solaris)	VSAM (z/OS)	DB2 for i (IBM i)	IMS (z/OS)
Operational window: Initial processing	30 hours	Two weekends	N/A	10 hours	5 Sundays	Eight hours nightly, 2.5 weeks
Frequency of incremental processing	Biweekly	Weekly	Monthly	Quarterly	Weekly	Daily
Incremental archive (% of source)	1%	0.425%	1.25%	2.3%	0.45%	0.07%
Incremental delete (% of source)	N/A. Drop partition	0.425%	1.25%	2.3%	0.45%	0.07%
Incremental restore (% of source)	1%	N/A	N/A	N/A	N/A	N/A
Operational window: Incremental processing	3 hours (years 1-4), 7 hours (years 5-7)	2.5 hours (years 1-4), 5 hours (years 5-7)	5 hours (years 1-4), 8 hours (years 5-7)	3 hours (years 1-4), 4 hours (years 5-7)	7 hours (years 1-4), 13 hours (years 5-7)	1 hour (years 1-4), 2 hours (years 5-7)
Number of query users	100	40	30	10	30	20
User archive access	Reporting database	ODM, ODBC/JDBC appl., and Cognos	ODM, ODBC/JDBC appl., and Cognos	ODM, ODBC/JDBC appl., and Cognos	ODM, ODBC/JDBC appl., and Cognos	ODM, ODBC/JDBC appl., and Cognos
SLA for queries	15 sec.	3 min.	10 min.	5 min.	5 min.	5 min.
Data center	Denver	Denver	Denver	Denver	Denver	Denver
Storage	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred	Years 1-2 on Tier 1; Years 3-5 on Tier 2; Years 6-7 on tape; Year 8 shred
Other requirements	N/A	N/A	N/A	Budget constraint	Budget constraint	Budget constraint

## Deployment model

To meet the new requirement to provide the best and most consistent query performance against archived data, we propose an operationally optimized deployment model in which all archive create functions are on a separate infrastructure from the archive access functions, as seen in Figure 4-32 on page 169. Optim hosts are provisioned to accommodate the various data sources and workload. All data sources (distributed and mainframe) share the same Optim repository. Storage is shared across all hosts via a file system.

Information from

OPTIMDIR #1 is published to OPTIMDIR #2, and the archive files are

reregistered so that the archive access environment has the metadata and location of the archive files.

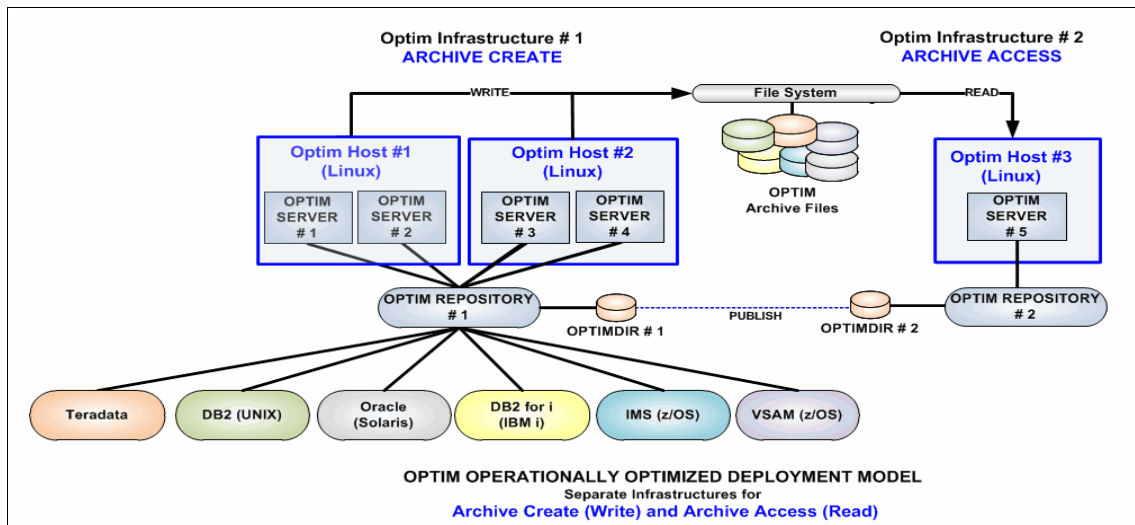


Figure 4-32 Optimized deployment model for a large Optim implementation

Figure 4-33 on page 170 illustrates the architectural layers in the solution and demonstrate the following functions:

- ▶ Scalability of the solution for large workloads
  - Optim hosts #1 and #2 for archive creation are multiple core servers and can accommodate large archive workloads.
- ▶ The physical separation of two Optim infrastructures to support archive creation and archive access activities. Optim host #3 supports the user query workload.
  - Separate Optim repository for archive access, for additional security based on roles and responsibilities, and separation of query users from infrastructures that access production data.
  - The metadata and archive file location from Optim repository #1 are published to #2 so that the archive access infrastructure will know where the archive files and indexes are located (shared storage).
- ▶ All data sources can share a single Optim repository for archive create and a single Optim repository provides access for all archived data.
- ▶ Enhanced options for accessing archiving data, including using ODBC/Java Database Connectivity (JDBC)-compliant applications, report writers, native application access, and federated services.

- ▶ Three-tiered storage implementation to reduce storage costs. The solution provides a choice of online, near-line, and off-line storage for archived data.
- ▶ Middleware federation software is collocated on all Optim server hosts, and middleware adapter software is installed on the same LPARs as the mainframe source data.
- ▶ The user requirement to provide a reporting database for the DB2 (AIX) data is satisfied by selectively restoring data from archive files to a DB2 reporting database.

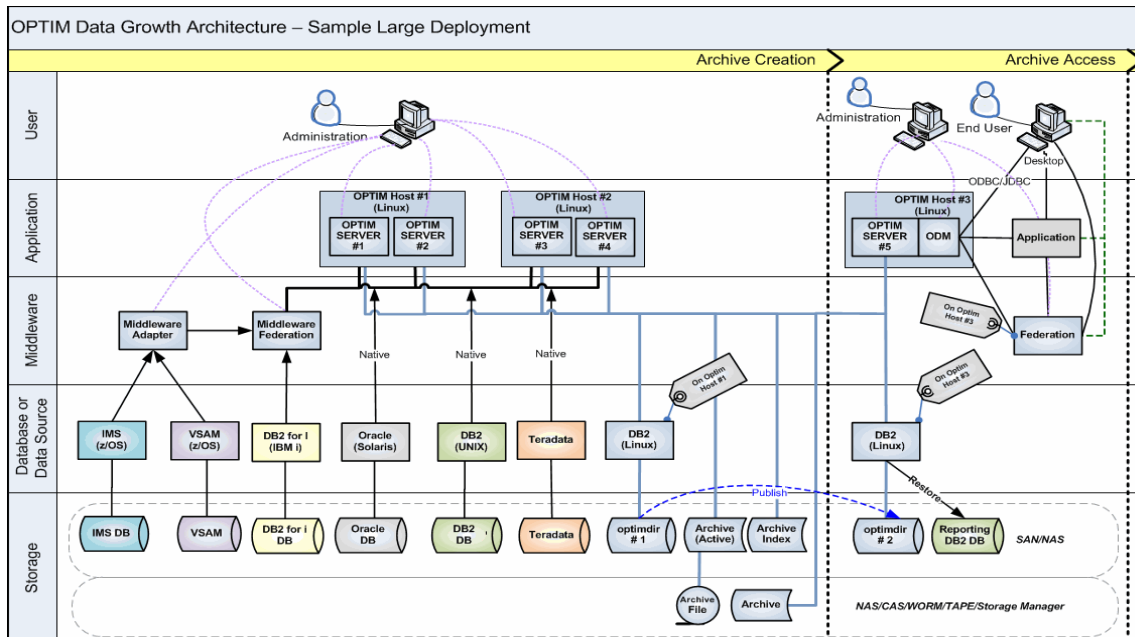


Figure 4-33 Architectural layers for a large Optim deployment

Figure 4-34 on page 171 illustrates the topology of the large deployment, satisfying the business, technical, and non-functional requirements.

All archive create functions (archive create, compress archive, create archive index, compare, delete, browse, and restore) can be executed on Optim servers #1 #2, #3, and #4 (installed on Optim hosts #1 and #2). You can execute all archive access functions (archive access, compress archive, create archive index, and browse) on Optim server #5 (installed on Optim host #3). Depending on scheduling and workload demands, the flexibility to run certain functions, such as compress the archive file and create archive indexes, on either environment is built into the architecture.



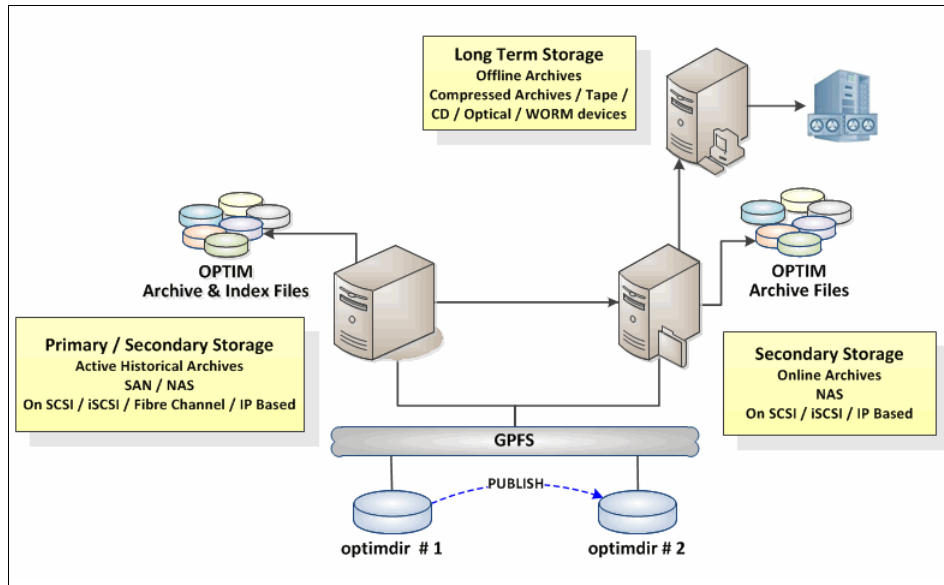


Figure 4-35 Sample, tiered storage for a large deployment model



## Sizing the Optim infrastructure

In this chapter, we introduce the factors and considerations that affect the sizing of hardware resources for your IBM InfoSphere Optim Data Growth Solution infrastructure, which include servers (hosts), workstations, network, and storage.

We discuss the following topics:

- ▶ Resources for the Optim infrastructure
- ▶ Types of sizing
- ▶ Sizing fundamentals
- ▶ Sizing factors:
  - Source data
  - Software considerations
  - Hardware considerations
  - Solution architecture
  - Optim design and workflow
  - Alignment with requirements
- ▶ Optim sizing metrics, methodology, and process
- ▶ Sample infrastructure sizing

## 5.1 Resources for the Optim infrastructure

Determining the hardware resource capacity that is required for your Optim infrastructure is an iterative process. Hardware sizing can be estimated early in the project using various metrics and modeling techniques. However, it is critical to run tests on your specific data within your IT ecosystem to validate and possibly adjust the sizing estimates based on actual performance results. In this chapter, we identify the variable factors that can affect throughput rates and the capacity that is required for your archive workload.

Sizing the Optim infrastructure includes the following hardware resources:

- ▶ Minimum resources:
  - Optim host
  - Optim workstation
  - Storage
  - Network
- ▶ Additional resources, depending on requirements, data sources, and architecture:
  - Additional Optim hosts and configuration workstations
  - Optim designer and Optim manager server and configuration workstation
  - Server for middleware federation and configuration workstation
  - Server for middleware adapter and configuration workstation
  - Open data manager (ODM) and configuration workstation
  - Redundancy of all hardware and software components
  - Additional Optim infrastructure for disaster recovery

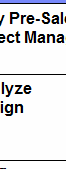
Information and artifacts from the business, technical, operational, and non-functional requirements, solution architecture, performance tuning, and Optim design all converge to influence the hardware sizing.

## 5.2 Types of sizing

Similar to the Optim solution architecture, sizing is an iterative process throughout the data growth project life cycle; it evolves as more detailed information becomes available. Sizing the Optim infrastructure is based on an architecture, whether it is a high-level, conceptual architecture, or a detailed, operational architecture. Therefore, various types or levels of sizing are determined by the specificity of the information that is available at a point in time.

Figure 5-1 on page 175 aligns the type of sizing to the project phases (refer to Chapter 2, “Project planning and considerations” on page 17). During the early

phases, a high-level, “ballpark” hardware sizing is appropriate, which provides generic sizing information using general rules, defaults, and assumptions that are based on similar deployments.

	PROJECT PHASE	SIZING TYPE	PURPOSE
 <p>HIGH LEVEL</p> <p>MORE DETAILED</p>	<ul style="list-style-type: none"> <li>❖ Early Pre-Sales</li> <li>❖ Project Management</li> </ul>	<b>Ballpark Sizing</b>	<ul style="list-style-type: none"> <li>❖ Quick “ballpark” sizing</li> <li>❖ Sample configurations</li> <li>❖ Optim Server &amp; Workstation</li> </ul>
	<ul style="list-style-type: none"> <li>❖ Analyze</li> <li>❖ Design</li> </ul>	<b>Fast Path Sizing</b>	<ul style="list-style-type: none"> <li>❖ High-level sizing when few details are known about design and implementation</li> <li>❖ Uses defaults, assumptions, and Rules of Thumb</li> <li>❖ Optim Server, Workstation, Middleware</li> </ul>
	<ul style="list-style-type: none"> <li>❖ Configure</li> <li>❖ Deploy</li> </ul>	<b>Detailed Sizing</b>	<ul style="list-style-type: none"> <li>❖ More detailed sizing when specific information is known about the implementation</li> <li>❖ Uses general sizing metrics</li> <li>❖ Customizable sizing factors applied</li> <li>❖ Optim Server, Workstation, Middleware, Storage</li> </ul>
	<ul style="list-style-type: none"> <li>❖ Operate</li> </ul>	<b>Capacity Plan</b>	<ul style="list-style-type: none"> <li>❖ Sizing based on client’s performance data from installed Optim (e.g. POC, pilot, production, upgrades)</li> <li>❖ Sizing targeted to specific data sources</li> <li>❖ Based on specific HW and SW infrastructure</li> <li>❖ Multi-year project roll-out and Project Phases</li> <li>❖ Optim Server, Workstation, Middleware, Storage, Network</li> <li>❖ Test, Validate, Adjust Sizing as Needed</li> </ul>

*Figure 5-1 Types of sizing and deliverables align to the project phase*

In later phases, when information about your requirements, data sources, data volumes, operational windows, detailed design, and configuration have been determined, you can apply more specific sizing factors (see 5.4, “Sizing factors” on page 178) to the base sizing metrics and sizing modeling algorithms, providing more detailed infrastructure-sizing guidelines. After Optim is installed in your environment and you have performed the initial testing and performance monitoring, you can use the metrics from your testing to estimate the hardware capacity that is required to support your full production rollout. If information is available, the following additional considerations can be incorporated and factored into the sizing estimate:

- ▶ Source data growth projections
- ▶ Project phasing
- ▶ Expanding the data growth solution to additional data sources
- ▶ Archive data in tiered storage until its end of life (shred)
- ▶ Multiple data centers
- ▶ High availability
- ▶ Disaster recovery
- ▶ Enterprise “archive factory”

## 5.3 Sizing fundamentals

At a high level, sizing is extremely simple:

- ▶ Unit of work: How much work has to be done (for example, how many gigabytes of data need to be archived)?
- ▶ Time: How long will it take (what is the operational window)?
- ▶ Resources (money and budget): How much will the infrastructure to do the work within the time frame cost (processor, storage, network)?

You might hear throughput statistics, such as archiving twenty million rows in five minutes. This type of sizing information, however, becomes merely anecdotal without understanding the business context, design considerations, and technical environment in which that throughput rate was archived. In other words, it depends.

To move from the anecdotal to a more consistent and repeatable process, sizing for the infrastructure is accomplished through the use of Optim sizing metrics and sizing methodology.

There are various ways to measure throughput in an Optim deployment:

- ▶ Rows per second
- ▶ Gigabytes per hour
- ▶ Objects per second

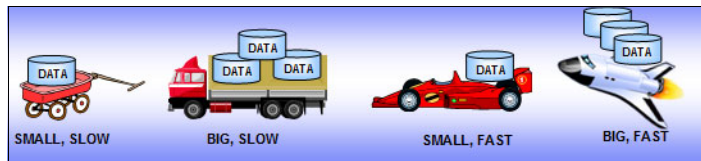
When calculating throughput rates, we typically use either megabytes per second or gigabytes per hour because it is more representative of the actual, overall throughput. Rows per second can be misleading due to varying row size. For example, a row might be 10 bytes or 5,000 bytes (binary large objects (BLOBs), character LOBs (CLOBs), or LOBs). Certain independent software vendor (ISV) packaged applications with well-known data schemas measure throughput in terms of objects archived per second. For example, Siebel CRM throughput might be measured by the number of service requests archived within a time period.

Each Optim process (archive create, delete, browse, compare, restore, and archive access) can have separate throughput rates and hardware resource capacity requirements, both of which can even vary within a single process. For example, a single archive process can archive at a separate throughput rate (number of MB/second) for each table within the same database.

Sizing the solution is an iterative process to find the right balance of time (how fast do you want your workload processed) and money (how much money do you want to spend on the resources) so that the work can complete within the

operational window as specified in the requirements. Just as there are typically multiple ways to architect your Optim solution, there are potentially a variety of hardware resources that can successfully process the workload.

The four examples in Figure 5-2 illustrate the importance of finding an acceptable balance of time and resource to meet your workload requirements. For example, you have a small amount of data that you want to archive and do not have a strict or urgent service level agreement (SLA). You might consider a resource similar to a “wagon,” which can only transport a small amount of data and is fairly slow because you must pull it along. However, a benefit of the wagon is that it is inexpensive. Certain clients only need a wagon to meet their archiving needs.



*Figure 5-2 Identify your key archive requirements*

Other clients might have a large volume of data to archive, but, because the data has been classified as “archivable” and therefore might have less business value than active, hot data, speed is not the compelling driver for the solution. This example is the “truck” approach; it can transport large volumes of data and accomplish the transport faster than a wagon, but the truck speed is limited by a posted highway speed limit.

The third scenario is when a client has a small to medium amount of data and extremely tight operational windows, so speed is a critical requirement of the solution. In this case, a “race car” is appropriate, which is extremely fast, has somewhat limited trunk space, and can be somewhat expensive.

The last scenario is the “cargo jet” that can hold a massive amounts of data and is the fastest way to process and move large amounts of data. Unfortunately, the cargo jet is quite a bit more expensive than the wagon; however, it is able to handle large mounts of data quickly.

Understand where your pain points are and what the more crucial and compelling business drivers are for your data archiving solution. This understanding helps influence your choices during the infrastructure sizing process.

## 5.4 Sizing factors

A variety of factors can influence the sizing of the Optim infrastructure, which is summarized in Figure 5-3. It is important to view the entire ecosystem when determining the hardware size, because it must be balanced. We review the more significant sizing factors in this section.

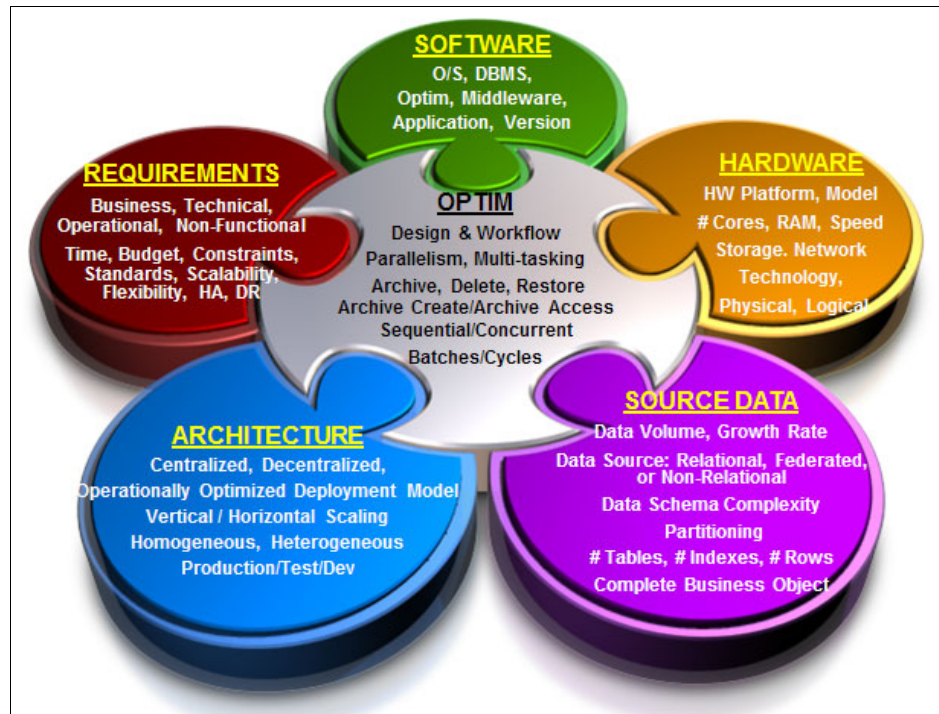


Figure 5-3 Sizing factors

### 5.4.1 Optim installed on Linux, UNIX, and Windows: Data sources

Optim installed on Linux, Unix, or Microsoft Windows can archive data from a wide variety of heterogeneous data sources that are installed on distributed and mainframe platforms. Data sources fall into three categories, and each category has throughput and sizing implications. Refer to 3.3.4, “Deploy and manage” on page 49 and 4.3.1, “Optim components” on page 107 for additional information.

Data sources are classified into three categories, depending on their support for drivers, cataloging of stored procedures, and metadata management: native, federated, and non-relational.

## Native relational database management system data sources

Optim that is installed on Linux, UNIX, and Microsoft Windows has native access to relational data sources:

- ▶ DB2 on Linux, UNIX, and Microsoft Windows
- ▶ Oracle on Linux, UNIX, and Microsoft Windows
- ▶ SQL Server<sup>1</sup>
- ▶ Teradata
- ▶ Informix
- ▶ Sybase
- ▶ DB2 on z/OS<sup>2</sup>
- ▶ Oracle on Linux on System z

The throughput rates for these data sources are typically the fastest, because there is no middleware software stack between the source data and the Optim server. More data is processed in less time, using less of the Optim host resources.

## Federated data sources

Federated data sources require middleware federation software, either IBM InfoSphere Federation Server or Oracle DG4ODBC. The following options to access federated data sources are supported:

- ▶ InfoSphere Federation Server:
  - DB2 for i
  - Open Database Connectivity (ODBC) data sources
- ▶ Oracle DG4ODBC:
  - ODBC data sources

In general, throughput is not quite as fast as native relational database management systems (RDBMS) data sources. And, throughput generally takes slightly more server capacity or a longer elapsed time to process the same amount of data.

You can install the middleware federation software on a separate server or on the source data server, but it is typically collocated on the Optim host for performance reasons. When collocated with the Optim host, you might need to provision a small amount of additional capacity (processor cores and memory), depending on data volumes, time constraints, and processing concurrency.

---

<sup>1</sup> Optim installed on Microsoft Windows accesses SQL Server natively.

<sup>2</sup> The access to DB2 on z/OS from the distributed platform is through DB2 Connect.

## Non-relational data sources

Optim on distributed platforms accesses non-relational data sources on distributed and mainframe environments through the collaboration of the two middleware software components: middleware federation (either IBM InfoSphere Federation Server or Oracle DG4ODBC) and middleware adapter (either IBM InfoSphere Classic Federation or Optim Connect). Optim on distributed platforms supports the following data sources and middleware software stack options:

- ▶ Middleware with Optim Connect (on Linux, UNIX, and Microsoft Windows):
  - Flat file (Linux, UNIX, and Microsoft Windows)
  - RMS on VAX
  - RDB on VAX
  - Flat file on VAX
  - Tandem
  - Enscribe on Tandem
  - Cache
- ▶ InfoSphere Federation Server and Classic Federation:
  - IMS
  - VSAM
  - Sequential
  - Datacom
  - IDMS
  - Adabas
- ▶ InfoSphere Federation Server and Optim Connect (on z/OS):
  - IMS
  - VSAM
  - Sequential
  - Adabas

## ***Placement of software components***

Multiple options are available for the placement and installation of the software components for accessing federated and non-relational data sources. Figure 5-4 on page 181 depicts the three types of data sources with all middleware components on separate servers. This topology results in slightly slower delivery of source data to the Optim host due to additional network hops to the middleware servers.

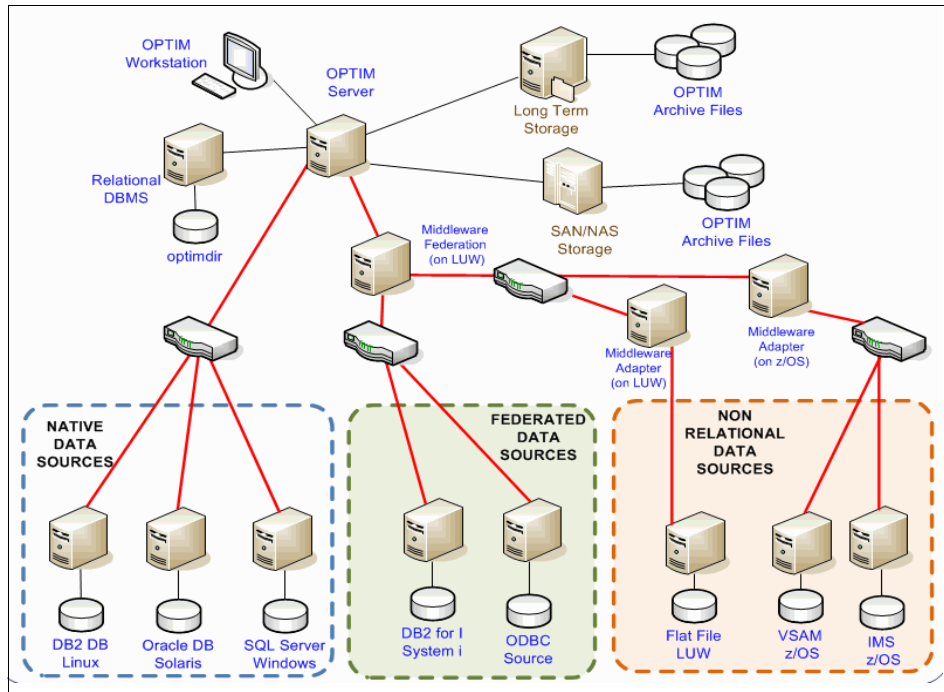


Figure 5-4 Optim distributed platform access to various data sources

Figure 5-5 on page 182 shows a more optimized topology, where the middleware federation software is collocated on the Optim host (on Linux, UNIX or Microsoft Windows), and the middleware adapter server is collocated in the same LPAR as the z/OS data source. This approach removes a network hop and improves the rate at which source data is delivered to Optim.

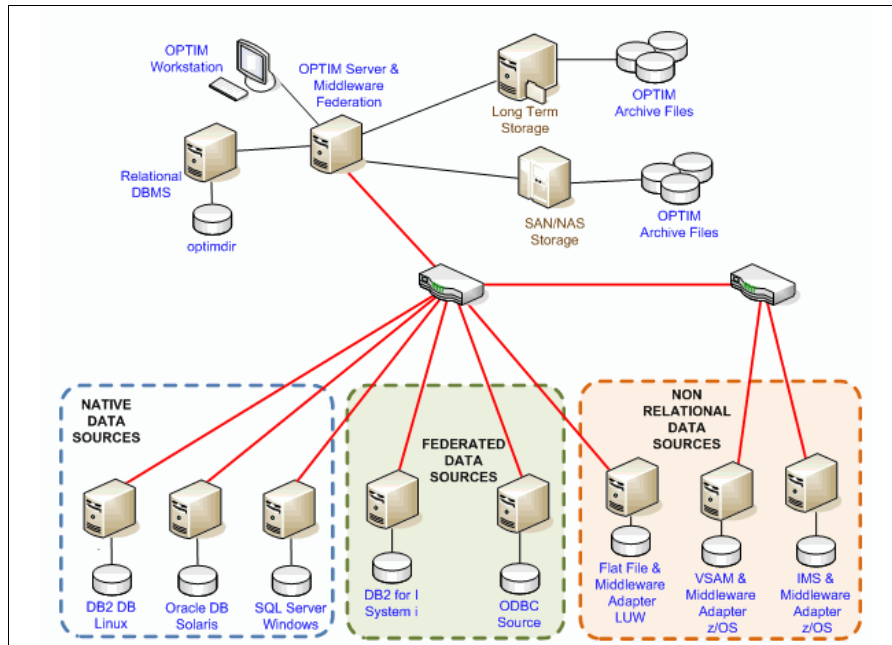


Figure 5-5 Optimized topology for access to various data sources

## 5.4.2 Optim installed on z/OS: Data source

Optim that is installed on z/OS supports the archiving of DB2 for z/OS.

### **Native RDBMS data sources**

Optim on z/OS accesses DB2 z/OS natively. The Optim software is installed in the same LPAR as the DB2 database.

## 5.4.3 Data schema complexity

The following data schema complexity can affect the rate at which data is retrieved and sent to Optim:

- ▶ Number of related tables in the complete business object: Traversing multiple tables to enforce referential integrity constraints can take longer than archiving a few tables or a single table.
- ▶ Partitioning: Partitioned data might improve retrieval if multiple archive jobs access separate partitions concurrently.
- ▶ Indexing source data: Proper indexing improves data retrieval.

## Data types

Data types, such as BLOBs, CLOBs, LOBS, and file attachments that are stored outside of a database, might take longer to process due to their size. Certain data types can add elapsed time to the processing unless the resources are scaled accordingly. Generally, this type of data does not compress well, which also affects the storage capacity that is needed for the archive files.

### 5.4.4 How fast can Optim extract the source data

One key factor in sizing the Optim host is determining how much data has to be processed within a given operational window. However, it is important to understand that the throughput rates on the Optim host also depend on the rate at which data can be extracted from the source. Data delivery, in turn, is limited by the speed at which source data can be retrieved from disk storage and sent across the network to the Optim host. The following key considerations affect the throughput rates of extracting the data:

- ▶ Access to source data using a database index or scan. Non-indexed data sources, such as flat files or VSAM, scan the entire source data. Databases might access the relevant data efficiently by using an index, if available.
- ▶ Disk storage read performance (performance of disk I/O subsystem for source data).
- ▶ Database reorganization and the currency of catalog statistics (for RDBMS sources).
- ▶ Capacity and utilization of data source server (average and peak utilization rates).
- ▶ Priority of the Optim request on the source data server.
- ▶ Number of Optim database connections to the database (for RDBMS sources).
- ▶ Number of concurrent archive requests.
- ▶ Bandwidth and utilization of the network during Optim processing.
- ▶ Time of day for Optim processing (generally scheduled and executed during off-peak periods).

Although an Optim host might have the capacity to archive, for example, 130 GB of data per hour, that capacity will not be fully utilized if the data is only delivered at a rate of 80 GB per hour due to bottlenecks on the database server, slow disk reads, multiple network hops, firewalls, and inadequate network bandwidth. Increasing the size of the Optim host or trying to tune Optim by increasing the number of database connections and increasing the number of concurrent jobs in this scenario only exacerbates the performance problem because the bottleneck

lies in the data delivery, not in the capacity of the Optim host. During testing, you can start analyzing where potential bottlenecks might occur within your IT infrastructure and make necessary adjustments.

If you have to process large volumes of data from one or many sources concurrently within a constrained operational window and there are no bottlenecks, you can scale and size the Optim infrastructure appropriately to accommodate the desired throughput rates. For example, if data is being delivered to the Optim server at 130 GB per hour, you must size the Optim host with an adequate number of processor cores, memory, and network cards to handle a minimum of 130 GB per hour. You must align and balance the Optim server capacity with the overall IT ecosystem.

## Data volume

The amount of data to be archived and deleted from your data sources affects both Optim host sizing, network capacity, and storage sizing. Consider the following information:

- ▶ Amount of data *initially* to be archived, deleted, and optionally restored to a reporting or historical database, production, or other target. Generally, a large amount of data is archived and deleted at the beginning of the project to reduce the size of the production source data:
  - If you have a tight operational window for the initial archive and delete, you might have to allow for extra server and network capacity during that time (more resources, but less time).
  - If budget is the larger concern, you can size the Optim host to accommodate the volume of data that is associated with incremental archive and delete processing. The initial, bulk processing can be done in multiple batches. Make each batch about the size of the incremental archive and delete, but execute the batches consecutively and more frequently until all the initial bulk data has been processed gradually. This approach is less expensive, because it requires a smaller Optim host. However, it takes longer.
  - An alternative approach is to have, for example, two Optim servers perform the initial bulk archive and delete processing. After the initial bulk archive and delete processing is complete, one of the Optim hosts can continue to process incremental archive create activities. You can repurpose the other Optim host for archive access processing (operationally optimized deployment model).
- ▶ Amount of data to be archived, deleted, and possibly restored *incrementally* (amount of data and frequency).
- ▶ Operational windows for initial and incremental processing. The amount of data to be processed for archive, delete, and restore processing and the

operational window determine the necessary throughput rates. Then, you can configure the Optim host to deliver that throughput rate, assuming there are no other bottlenecks in the system.

- ▶ Large databases are frequently partitioned. If archiving and deleting all the data in one or more partitions, you can improve the delete processing significantly if the partitions are dropped after the successful completion of the archive processing. This step eliminates delete processing and reduces the overall resource capacity that is required for the Optim host and network.

## **Number of data sources**

When your data growth solution includes multiple data sources, you must determine if business requirements specify that you have to process part or all of the data sources concurrently or sequentially, for both the initial and the incremental workloads. This choice influences the total capacity that you will require, in terms of the Optim host, temporary storage, and network bandwidth. Consider the following information:

- ▶ Number of data sources and data volume for both initial and incremental processing
- ▶ Processing frequency and time constraints
- ▶ Concurrency and parallelism
- ▶ Impact of application workflows and data dependencies on archiving processing (for example, both the application workflows and data dependencies might need to be processed together, on the same day, or sequentially for business reasons)
- ▶ Capacity of the network
- ▶ Performance of the I/O subsystem for archive files
- ▶ Location of the data sources

## **5.4.5 Hardware sizing considerations**

After the solution architecture has been developed, you can start determining the necessary capacity to support your planned workload within your operational window. You must consider the hardware components: the Optim host, Optim workstation, network, and storage.

### **Optim host**

The Optim host is the hardware server, either physical or logical (virtual), on which the Optim server instance (software) is installed.

### ***Minimum hardware requirements***

The minimum hardware requirements for an extremely small Optim host are available at the following website:

<https://www-304.ibm.com/support/docview.wss?uid=swg27019453>

The Optim host requires the following key sizing information:

- ▶ Number of processor cores (physical or virtual)
- ▶ Memory
- ▶ Processor speed
- ▶ Number and capacity of Fibre Channel (FC) cards for storage area network (SAN)
- ▶ Number and capacity of Ethernet network interface cards (NICs) for network-attached storage (NAS) and TCP/IP connectivity

It is important to consider the sizing factors that are summarized in this chapter. Start with several baseline performance metrics and employ modeling techniques to estimate the server capacity requirements. Refer to 5.5, “Sizing metrics, methodology, and process” on page 201 for information about the Optim sizing metrics and the sizing process.

Optim performs best on servers with fast clock speeds. Deploying an older, slower server for the Optim host negatively affects your overall throughput. Specific processing, such as compression, is processor intensive, so overall performance and throughput will benefit from faster servers. We suggest that you size the Optim host with additional capacity as a buffer to handle the peak processing periods. Typically, Optim on distributed platforms is not memory intensive. We generally suggest starting with at least two GB of RAM for each processor core.

### ***Data growth***

Most clients that implement Optim experience large increases in production data source growth rates. When sizing for the Optim host, consider the following information:

- ▶ Size the server so that it can achieve target throughput rates with the given data growth rate for an estimated three to four years, which is about the same length of time at which many clients refresh their servers.

For example, 1,000 GB at a compounded growth rate of 2% per month grows to 2,040 GB by the end of the third year. Do not forget to perform the math when projecting server requirements for longer than the immediate, tactical, archive processing.

- ▶ When selecting a server, make sure that it is expandable so that you can add processor cores, memory, and potentially network cards, as necessary:
  - For example, a 4-core server can be upgraded to an 8-core server, or a 6-core server can be upgraded to a 12-core sever.
  - Certain servers offer “capacity on demand” in which additional processing cores and memory can be allocated to the server during peak processing periods, such as during the initial archive processing.

Information that can affect hardware sizing might not be available until later phases of the project, so it is critical to build a flexible, scalable, and extendable Optim infrastructure to accommodate additional data growth workloads.

### ***Virtualization considerations***

Many clients use virtualization environments to consolidate multiple workloads onto fewer systems. Optim can run on these virtualized environments, which are powered by supported hypervisors, such as VMware and IBM PowerVM™.

### ***Impact of multi-processing***

*Multi-processing* is the execution of multiple Optim processes concurrently. The concurrent processes can be either the same request (for example, six archive processes) or separate requests (for example, three archives, two deletes, and one restore). Concurrent processing can occur against a single or multiple data sources. If resources are unconstrained, multi-processing is an effective strategy to improve performance and throughput.

In Figure 5-6 on page 188, you can see the implications of archiving 600,000,000 rows from a single, large DB2 table using one, two, three, and four concurrent archive processes. Figure 5-6 on page 188 compares the percent of processor cores used (Y axis<sup>3</sup>) to the amount of elapsed time that it took to archive the data (X axis). The Optim host was an 8-core Microsoft Windows server.

Use case #1 in Figure 5-6 on page 188 shows that a single archive job archived 600 million rows in about 48 minutes and used on average 1.5 processor cores (150%). This use case shows that when a single archive process runs on an 8-core processor, about 80% - 85% of the Optim host is idle. Therefore, you pay more for your resource (server) without fully using the server capacity or gaining any improvement in the elapsed processing time.

Use case #2 illustrates two concurrent archive jobs, each archiving 300 million rows. More resource is used, about 2.85 processor cores, and elapsed time was reduced to 29 minutes. You can see further improvements by executing three concurrent archives (200 million rows each), which is shown by use case #3, and

---

<sup>3</sup> The Optim host was an 8-core Microsoft Windows processor. Each processor core shows on the Y-axis as 100%, so the total capacity for the server is 800%.

then four concurrent archive processes (150 million rows each), which is shown in use case #4.

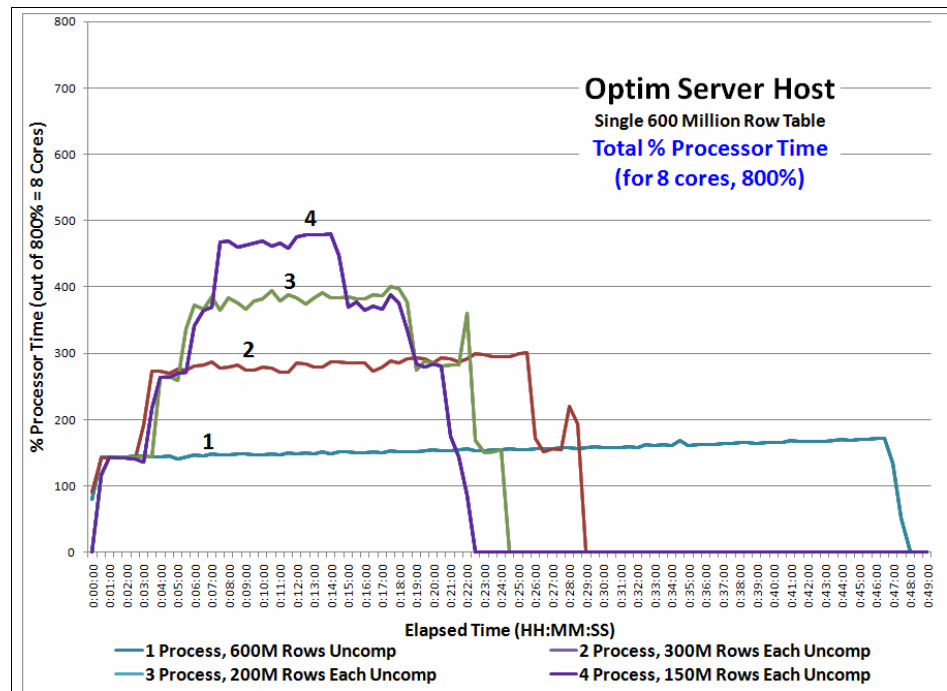


Figure 5-6 Impact of scaling the number of concurrent archive jobs on Optim host utilization

By scaling the number of concurrent processes from one to four in this benchmark, we were able to archive the same 600 million rows in less than half of the elapsed time using only 2.75 times more processor capacity (around five cores instead of 1.5 cores; more resource, less time).

If you can achieve your desired throughput rate using a single process and only have to process your data sources one at a time, you might not need to scale the Optim server to more than four processor cores. However, if you are unable to achieve the required throughput rates with a single process, you might need to scale the infrastructure resources to support the additional concurrent workload.

During the testing phase of your deployment, it is important to test throughput rates and scalability using your specific data within your infrastructure. Your results might vary, depending on many factors.

## Scalability

At a certain point, when you keep increasing the number of database connections, concurrent jobs, and Optim server instances, you will see a point of diminishing returns when throughput gradually starts to degrade. It is important to determine where the bottleneck resides. Is it the database, database host, network, storage, the Optim host, or Optim access definition? If no bottlenecks are observed or identified during performance monitoring, you can continue to scale the Optim host by performing these tasks:

- ▶ Scaling the number of concurrent jobs in each Optim instance (“scale-up” software)
- ▶ Installing additional Optim server instances when Optim is installed on Linux or UNIX (“scale-out” software)
- ▶ Increasing the capacity of the Optim host by adding processor cores and memory (“scale-up” hardware)
- ▶ Deploying one or more physical or virtual Optim hosts with a separate instance of Optim (“scale-out” hardware) and distributing the workload across the servers

You need to balance all the components in the solution architecture to achieve the maximum throughput with the least cost.

## Archive file compression

If you want your archive files compressed, Optim provides flexible choices in terms of processing workflow to suit your specific objectives. Table 5-1 lists the compression processing options and their benefits and usage considerations.

Table 5-1 Compression processing options

	Description	Benefits	Considerations
Inline compression	Data is compressed as it is extracted and before it is written to the archive file.	Read/write data only once, fewer I/Os, and shorter elapsed time.	Connection to source database is longer.
Post compression	Data is compressed after it has been extracted and written to the archive file.	Connection to source database is shorter.	Read/write data twice, more I/Os, and longer total elapsed time.

## Optim workstation

Each Optim implementation requires that at least one workstation is installed on Microsoft Windows. The following website shows the minimum hardware requirements for the Optim workstation:

<https://www-304.ibm.com/support/docview.wss?uid=swg27019453>

You can deploy one or multiple Optim workstation clients, depending on the number of users deploying and configuring the InfoSphere Optim Data Growth Solution.

## Network

Network throughput is limited by the slowest element in the network stack:

- ▶ Bandwidth
- ▶ Routers and switches
- ▶ Number and capacity of network interface cards
- ▶ Number and capacity of FC interface cards
- ▶ Number of paths

Also, these factors affect the network throughput:

- ▶ Network hops between the source data host and the Optim host
- ▶ Firewalls
- ▶ Concurrent network activity
- ▶ Utilization of the network (average and peak)

For example, if your network is 100 MB/second, you can transmit around 8 MB/second at 65% utilization, assuming that no other network workload contends for bandwidth. At that rate, it takes around 1.8 hours to transmit 50 GB of data, or 73 hours to transmit two TB of data. If your requirement is to archive two TB of data in 48 hours, you know that you will need to scale the network at a minimum, because merely the transmission time with that limited bandwidth will take over three days.

Network capacity must support the throughput rates that are required by your archive SLAs, which are affected by these factors:

- ▶ Location of the data source hosts and the Optim host
- ▶ Number of database connections per process
- ▶ Number of concurrent processes
- ▶ Array fetch (key lookup limit)
- ▶ Number of Optim server instances
- ▶ Number of Optim hosts
- ▶ Throughput rate of each process
- ▶ Other concurrent network activity
- ▶ Time of day (peak or off-peak)

Similar to the Optim host, plan to size the network with spare capacity to accommodate the peak processing periods. If the network is saturated, adding additional workload from the Optim server can result in further network bottlenecks, resulting in degraded overall performance.

It is considered a best practice to have the Optim host collocated in the same data center location as the source data, preferably on the same subnet. It is not advisable to archive or delete the source data that resides in a separate data center than the Optim host due to network latency, especially for large volumes of data. If you have source data in multiple locations, it is best to install an Optim host at each of those locations to minimize network latency, as exemplified by the decentralized deployment that is shown in Figure 4-7 on page 119.

If critical time constraints are imposed on archiving data, consider installing Optim on the same host as the source data. The advantage of this approach is that it eliminates the network and increases the throughput rate. However, it imposes additional workload and resource requirements on your production database server. Consider your production database server within the context of other concurrent database workloads, the execution time frame (peak versus non-peak hours), and source data host capacity.

Figure 5-7 illustrates the impact on network capacity in terms of total bytes sent and received per second, when the same amount of data is processed using one process (600 million rows) (see line #2) versus four concurrent processes (150 million rows each) (see line #1). Bytes per second transmitted over the network are significantly higher with four concurrent archive processes, compared to a single archive process, and the four concurrent processes completed in less than half of the elapsed time. This example, once again, shows the balance of time versus cost (resource).

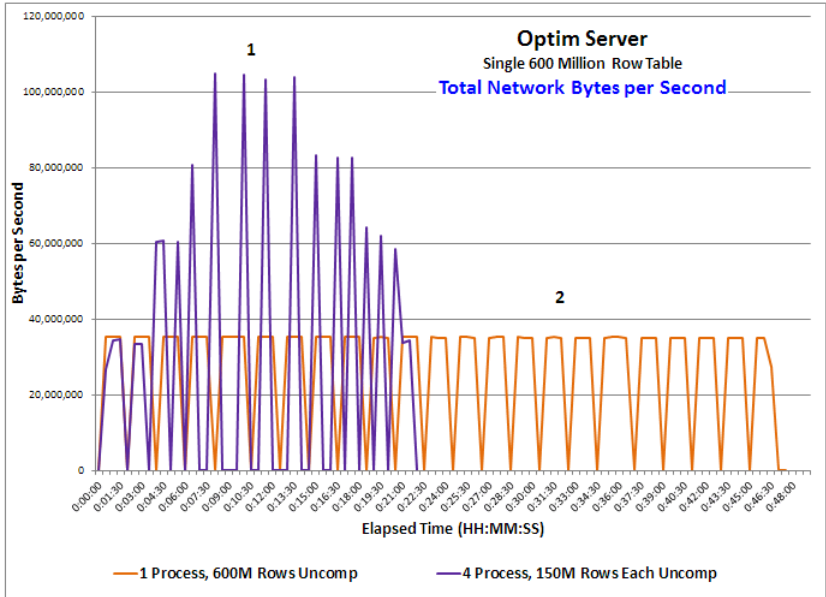


Figure 5-7 Network impact of one versus four concurrent archive processes

## Storage

Estimating the storage capacity that will be needed for your data growth project evolves as you obtain more specific information about the processing workflows, concurrency rates, number of copies of the archive files that you might need, compression ratios, archive file index requirements, and so on. Optim provides capabilities to manage archive files automatically using a *storage profile*, which can create a duplicate copy of an archive file, save a copy of the archive file to a backup device, and provide a date when the primary archive file is deleted. You can also create a retention policy for archive files, after which the primary archive file will be purged.

The purpose of storage capacity estimation in the early phases of the project is to provide initial information to begin the ongoing process of storage subsystem sizing. You will need to refine the actual storage capacity required after the testing begins on your specific data and the processing workflows have been designed.

Refer to Chapter 10, “Manage Archive Media”, in the *IBM Optim Archive User Manual* for additional information about how to manage archive files on mixed and secondary media.

### **Storage requirements**

Storage is required for the following components in an Optim Data Growth solution:

- ▶ Archive files (compressed or non-compressed)
- ▶ Archive file indexes (optional, but recommended if querying the archive files) (non-compressed)
- ▶ Any secondary or backup copies of the archive files and indexes (for example, for an off-site disaster recovery data center)
- ▶ Optim directory
- ▶ Optim binaries
- ▶ Temporary storage, as needed for certain processing
- ▶ Optional historical or reporting database (data that is restored from the archive file)

### **Optim directory**

The Optim directory is usually small in size, around 512 KB to 5 GB, depending on the number of projects. Store it on a similar storage type to the storage type that is used for your production databases, with a level of RAID.

### ***Temporary storage***

The amount of temporary storage depends on the process that is running (archive, delete, compare, restore, and so on), volume of data, and number of concurrent jobs. Temporary storage is only required until the processing completes and then it can be reclaimed. Temporary storage is required for the following functions:

- ▶ Control file: Similar to a transaction log, the control file tracks the success or failure for each row during certain processing. Optim creates a control file after any action that modifies the source database, for example, during delete and restore processing, which allows a restart of the process if it is interrupted.
- ▶ Loader file for restoring archived data using the **load** utility.
- ▶ Temporary storage for compare processing.
- ▶ Temporary storage to recall archived data from tape to disk for queries when a tiered storage strategy has been deployed.

### ***Storage performance***

The performance of the disk I/O subsystem can affect the throughput rates on the Optim host significantly, especially when processing large volumes of data. The performance of the disk I/O subsystem can affect the efficient reads from the source data, writes to the archive files, deletes to the source data, writes during restore, and reads from the archive files for query access.

The benchmark results in Figure 5-8 on page 194 show the impact of disk read performance on the source database host during Optim archive processing. In these tests, the database server was installed on an 8-core Intel server and the Optim host was installed on a second 8-core Intel server. In both tests, 600 million rows were archived from a single, large table in a DB2 database and the archive files were written to SAN. The result for use case #1 shows an average of 32 milliseconds per read from the source database, and the archive job took two hours, 14 minutes, and two seconds. Use case #2 utilized faster SAN storage with an average of six milliseconds per read, and it completed in 47 minutes and 31 seconds. Merely by changing the disk subsystem on the database host, we were able to archive the data 2.8 times faster.

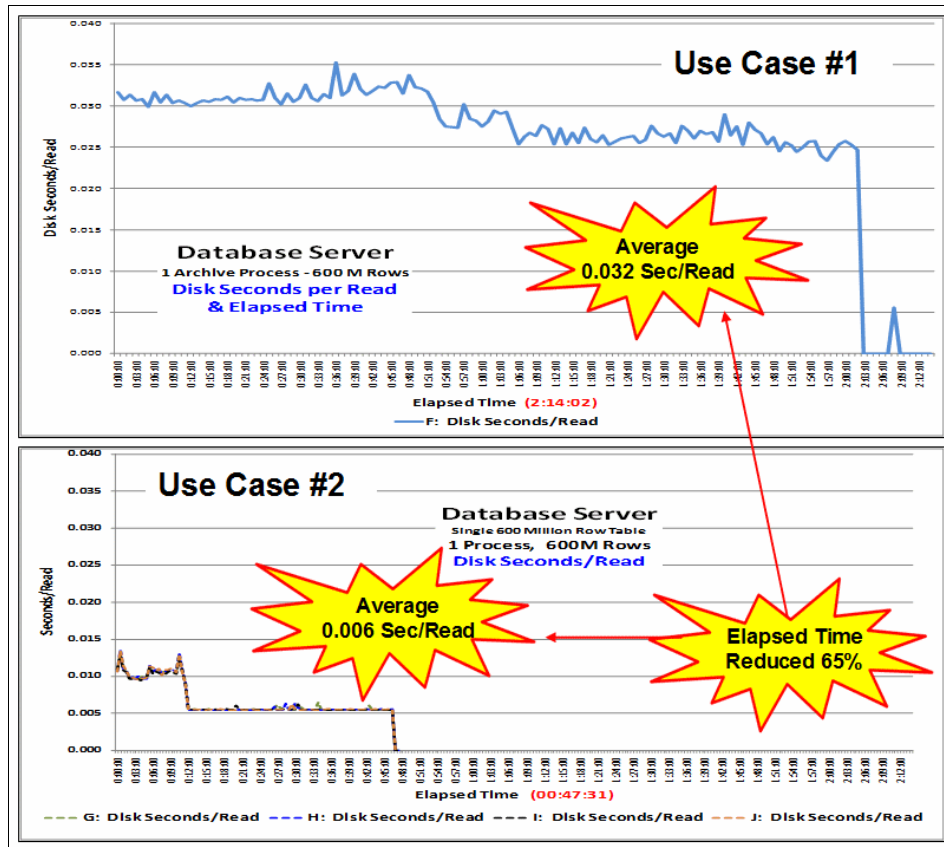


Figure 5-8 Impact of storage performance on Optim throughput

Storage performance affects the throughput rate of writing the archive files and reading the archive files for delete, restore, and query processing. Archive files can be on slower tier-2 or tier-3 storage, compared to your production data on tier-1 storage. However, users accessing archived data cannot expect it to perform in the same way as a production database system.

Generally, for the best storage performance, use a SAN, which can run at a sustained rate of around 300+ MB/second<sup>4</sup> per FC. NAS, however, can run around 50 to 70 MB/second<sup>4</sup>. For example, Table 5-2 on page 195 compares the estimated time to write one TB of archive data to SAN and NAS storage.

<sup>4</sup> Performance can vary; this situation is used as an example.

Table 5-2 Sample estimates of time to write 1 TB of archive data to storage

Storage type	Average throughput	Hours
SAN	300 MB/second	1 hour
NAS	70 MB/second	4+ hours

In this scenario, it takes more than four times longer to write the files to disk if you use NAS instead of SAN.

Refer to your non-functional requirements to see if storage performance will impede your SLA. However, if cost is a compelling driver for your data growth solution, the cheaper, slower storage might be appropriate. Typically, archive files are written once and read occasionally. You must align the speed of the archive file writes (and therefore archive create throughput) and archive file reads (and, therefore, the SLA for query users) with the selection of storage hardware and technology. Refer to “Storage” on page 153 for more information about using tiered storage to reduce storage costs.

Optim archive files must be spread across multiple disk volumes to reduce and minimize the I/O contention when multi-processing is implemented. Archive file indexes must be on separate disks from the associated archive files. Critical files, such as the Optim directory, control files, and logs, must reside on separate physical disks, if possible.

You need to hold periodic capacity planning reviews as additional specific implementation data and historic data become available. As with any system, you need to establish ongoing monitoring and tuning procedures to track disk growth to provide accurate sizing information.

### ***Storage savings return on investment***

A major benefit of archiving data using Optim is that aged data can be compressed and stored on less expensive tiered storage, depending on the business value and access requirements of the data, providing a positive return on investment (ROI). There is significant cost savings in reducing the amount of data that is stored in your production databases and data sources on expensive tier 1 storage, after data has been archived and deleted from the source. For database systems, the database has to be reorganized and catalog statistics updated to reclaim the space that is gained as a result of archive and delete processing.

Optim for distributed platforms provides a *delete impact analysis* report for DB2 for Linux, UNIX, and Microsoft Windows and Oracle data sources, which estimates the amount of storage that will be saved as a result of the Optim delete process. If DBMS statistics are enabled at the database level, Optim calculates

the savings for each deleted object. These statistics are included in the delete report, regardless of the setting for statistics in the delete request. The delete report shows statistics for each table and index. Index and table statistics for LOB data are not included in the estimate.

## 5.4.6 Align the hardware resource sizing with the deployment model

In 4.3.3, “Reference deployment models” on page 117, we describe the three reference deployment models (centralized, decentralized, and operationally optimized) and how they can be scaled and extended to satisfy your requirements. It is during the sizing process that you determine the capacity of various hardware components, the number of Optim hosts, and the optimal placement of software components on hardware. The resulting solution architecture affects sizing:

- Centralized deployment model

The centralized deployment model assumes a single Optim host in a single location. If you anticipate any additional workload (additional data sources or larger data volumes), it is important to select a server that can scale by adding additional processor cores and memory.

With a centralized deployment model, you have to determine the maximum capacity that you will require during an operational window, based on the total workload, taking into consideration the potential growth of that workload over time. If that maximum capacity exceeds the capacity of a single Optim server instance, consider multiple Optim server instances on the Optim host (when Optim is installed on Linux or UNIX). If additional capacity is required, you might need to migrate to a decentralized deployment model and add an additional Optim host. Testing helps determine throughput rates on a specific server in your environment.

- Decentralized deployment model

If you require the capacity of more than a single Optim host, you must size for at least two Optim hosts and scale the hardware based on maximum concurrent capacity requirements.

The decentralized deployment model is more flexible in that there can be multiple Optim hosts in a single data center or in multiple data centers. If you have data sources in multiple data centers, it is best to deploy an Optim host in each data center. You must size each Optim host based on the workload that it will perform at that location. You can obtain additional flexibility and scalability by adding physical or logical servers. The goal is to size the various components in the infrastructure (hosts, storage, and network capacity) based on the workload in each data center.

- ▶ **Operationally optimized deployment model**  
 If the operationally optimized deployment model, which is designed to segregate archive create processing from archive access processing, is the basis of your solution architecture, you have to size for a minimum of two Optim hosts. With two hosts, you can scale and tune each environment specifically for the workload. For example, if you archive data once a month and it takes 24 hours, the archive create Optim host can be virtualized. Then, the excess server capacity that is available the other 29 days in the month can be utilized for other workloads. However, if you have users querying the archive files daily, consider providing a more dedicated host environment for archive access to ensure that SLAs can be met.
- ▶ **High availability**  
 If you require high availability, you must size for an additional Optim host and configure redundant hardware and software components for everything. Using certain RAID levels for storage can increase the availability; however, it will increase the overall storage volume.
- ▶ **Disaster recovery**  
 You must deploy a separate Optim host at the disaster recovery site, along with a duplicate copy of the archive files and the Optim directory. Depending on your requirements, the Optim host at the disaster recovery site is typically sized at 50% of the capacity of the production Optim host.
- ▶ **Middleware:**
  - If your solution architecture includes any middleware federation software (InfoSphere Federation Server or Oracle DG4ODBC), you must size for additional capacity to accommodate both the middleware and requisite RDBMS. You can install this middleware on any of the following hosts:
    - Separate host
    - Source data host
    - Optim host (generally recommended)
  - If your solution architecture includes any middleware adapter software (Classic Federation or Optim Connect), you will need a small amount of additional capacity. This component is typically installed on the data source host.

### 5.4.7 Optim design and workflow considerations

Within a single Optim process, such as archive, there are a number of processing workflow variations and design considerations that will affect not only the throughput, but also resource consumption and hardware sizing. For example, certain processing can be accomplished as a single step (archive

create + compress + create archive file index); in which case, the data is read once and written once. Therefore, there is more demand on the Optim host resources due to more processing, but less demand on the I/O subsystem during the processing period. Throughput is usually fastest.

If you have constrained operational windows on the data source server, you can alternatively separate that same processing into three separate steps (archive create, then compress, and then create archive file index). One benefit is that the process will disconnect from the source data server faster (after the archive completes), thus decreasing the elapsed time in the source data server. However, the data will need to be read and written three times instead of once, affecting the I/O subsystem on the Optim server host and lengthening the overall elapsed time. Table 5-3 summarizes the processing and workflow options for archive and the effect on the resource consumption.

*Table 5-3 Examples of effect of Optim archive processing workflow on hardware resource requirements*

Process	Source data host	Network	Optim host	Storage
Archive create, archive file is not compressed	Fastest way to extract data from source and disconnect from source. Heavy read activity.	All data transmitted from source data host to Optim host.	Least capacity needed to archive data. Heavy write activity to non-compressed archive files.	Non-compressed files are larger than source data and much larger than compressed files.
Compress archive file (input is not compressed archive file)	N/A	N/A	Heavy CPU	Heavy read from non-compressed archive file and less heavy write to compressed archive file. More storage required temporarily for both non-compressed and compressed archive files.
Create archive file index (after extract from source)	N/A	N/A	Heavy CPU	Heavy read from archive file and write to index file. Indexes are not compressed.

Process	Source data host	Network	Optim host	Storage
Archive create + archive file compression	No extra CPU overhead, but longer elapsed time connected to source data host due to time for Optim to compress data.	All data transmitted from source data host to Optim host.	Compression is CPU intensive, with a longer elapsed time than if using a non-compressed archive file.	Significantly less storage than a non-compressed archive files, less time to write to disk.
Archive create + archive file compression + index create	No extra CPU overhead, but longer elapsed time connected to source data host due to Optim compressing data and index creation.	All data transmitted from source data host to Optim host.	Compression and index creation are CPU intensive, longer elapsed time than if not compressed or no index.	Least amount of storage required for archive files, faster write to disk because less data when compressed. Indexes are not compressed.
Multiple number of concurrent processes (“n”)	Up to “n” times overhead for one archive.	Up to “n” times overhead for one archive.	Up to “n” times overhead for one archive.	Up to “n” times concurrent disk writes.
Multiple database connections (“n”) (parallelism)	Up to “n” times more concurrent requests from source database server.	Up to “n” times overhead for one DB connection.	Up to “n” times more source data can be delivered concurrently, might take more CPU.	Up to “n” times concurrent disk writes to the Optim host.
Single archive process	Requires longer operational window on source data host.	Longer sustained throughput rate needed.	Longer sustained throughput rate needed.	One large file.
Multiple smaller archive processes	Can spread workload across multiple jobs over a longer time frame.	Shorter bursts of activity, spread over a longer time frame.	Shorter bursts of activity, spread over a longer time frame.	Multiple files can be logically combined using an archive collection.

As described in the table, certain design decisions will affect the capacity that is required for various hardware components of the infrastructure. Your choices need to reflect the requirements.

Consider these examples:

- ▶ If the operational window on your production source data host is a constraint, you might consider strategies that minimize the time that an Optim process is connected to your production source data host. Archive file compression or index creation can be done afterwards on the Optim host, so that it does not have an impact on the source data host.
- ▶ If one of your business requirements is to minimize the cost of storage for archived data, plan for processing strategies that minimize storage, such as compression.
- ▶ If you need the fastest end-to-end processing (archive create + compression + index create), you can run all of those functions in a single request and avoid the extra overhead of reading and writing the data multiple times if the steps are done sequentially.
- ▶ You have determined through testing that you can achieve a maximum throughput of “ $n$ ” on a 4-core server with two Optim server instances and four concurrent processes (two per Optim server instance). However, you need “ $2 \times n$ ” throughput to meet your operational window. Therefore, in the sizing phase, you need to either scale up the Optim host (for example, an eight core) or scale out to another Optim host (for example, a second four core). Your standard operating environment requirements can also influence the choice of servers.
- ▶ If business requirements dictate that you must archive data daily (which yields an average of at least 30 archive files per month), but users will access the archived data based on month, the queries will, on average, need to access data from at least 30 archive files. Although these 30 archive files can be logically combined in an archive collection, it is more efficient if the archived data is stored in a manner that more closely aligns to the query requests. You can use a staging database to repackage the incoming data to align with the query usage. Then, daily archives are restored to the staging database each day, and at the end of the month, a “monthly” archive is performed against the staging database. Archive access performance is thus improved.
- ▶ During the later phases of the sizing process, when information about the design of the Optim processes and workflows is known, you can perform more detailed sizing estimates based on planned concurrent workloads.

Refer to *The IBM Optim Archive User Manual* for additional information regarding options for other Optim processes.

## 5.5 Sizing metrics, methodology, and process

It is best to base the sizing of the production hardware components in your Optim infrastructure on the performance metrics obtained during the testing phase using processes against your data sources within your IT ecosystem. However, in the early phases of the project and for hardware planning purposes, this method might not always be feasible.

When specific, customized performance metrics for your implementation are unavailable, we use generalized metrics from various sources that have been obtained, analyzed, and synthesized. As seen in Figure 5-9 on page 202, we derive Optim sizing and performance metrics from a variety of sources, including the following sources:

- ▶ Over 300 benchmarks that have been conducted at the following IBM worldwide labs: IBM Innovation Center in Waltham, Massachusetts; Princeton Lab in Princeton, New Jersey; Silicon Valley Lab in San Jose, California; and the IBM China Development Lab in Beijing, China
- ▶ Proof of concept (POC) and pilot engagements
- ▶ Production implementations
- ▶ Regression and quality certification testing
- ▶ Server metrics
- ▶ Empirical knowledge based on experience

When IBM conducts a sizing and solution architecture engagement (Figure 5-9 on page 202), IBM uses a methodology to ensure that requirements are being satisfied and to mitigate risk. The process begins with your data growth requirements, which, as an initial step, are documented in a sizing questionnaire. There are typically follow-on meetings and reviews to obtain missing information, clarify items, and start a dialog regarding available options. The precision of a sizing estimate depends on the project phase and the accuracy and specificity of the information provided.

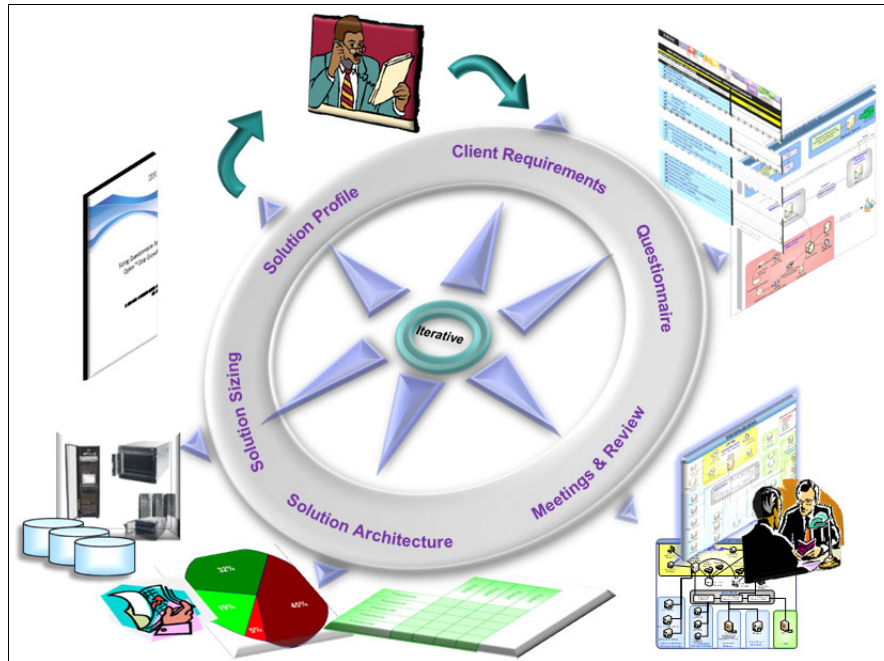


Figure 5-9 The Optim sizing process is iterative

Next, the solution architecture is developed, which forms the basis of the resource sizing. Using various sizing models that can take into account several of the sizing factors described in 5.4, “Sizing factors” on page 178, sizing guidelines for the hardware components are developed for your specific implementation. The Optim sizing metrics and sizing methodology are continually reviewed and revised to provide the best possible estimate of hardware capacity requirements for deployments of Optim.

An InfoSphere Optim Data Growth Solution Sizing Guidelines and Preliminary Solution Architecture report, which summarizes the solution architecture and resource sizing estimates, is developed for your implementation.

Finalizing your hardware resource requirements and deployment architecture is an iterative process that evolves as more information is accumulated throughout the implementation project life cycle. The final production hardware sizing (server capacity, storage, and network) and solution architecture must ultimately be determined by your project implementation team after the detailed design and testing phases are complete.

## 5.6 Sizing example

The purpose of the following IBM InfoSphere Optim Data Growth Solution sizing example is to provide high-level sample sizing guidelines that are representative of a large deployment. Your results can vary, depending on many factors. Sizing is an iterative process that is refined during the project and must be aligned to your specific data sources and requirements.

### 5.6.1 Sample large, operationally optimized Optim deployment

In this section, we describe the sizing guidelines for a sample large, operationally optimized Optim deployment scenario, which was described in 4.7.1, “Large Optim deployment” on page 167. The infrastructure has been sized to process the data volumes within the specified operational windows. Figure 5-10 shows the topology for easy reference.

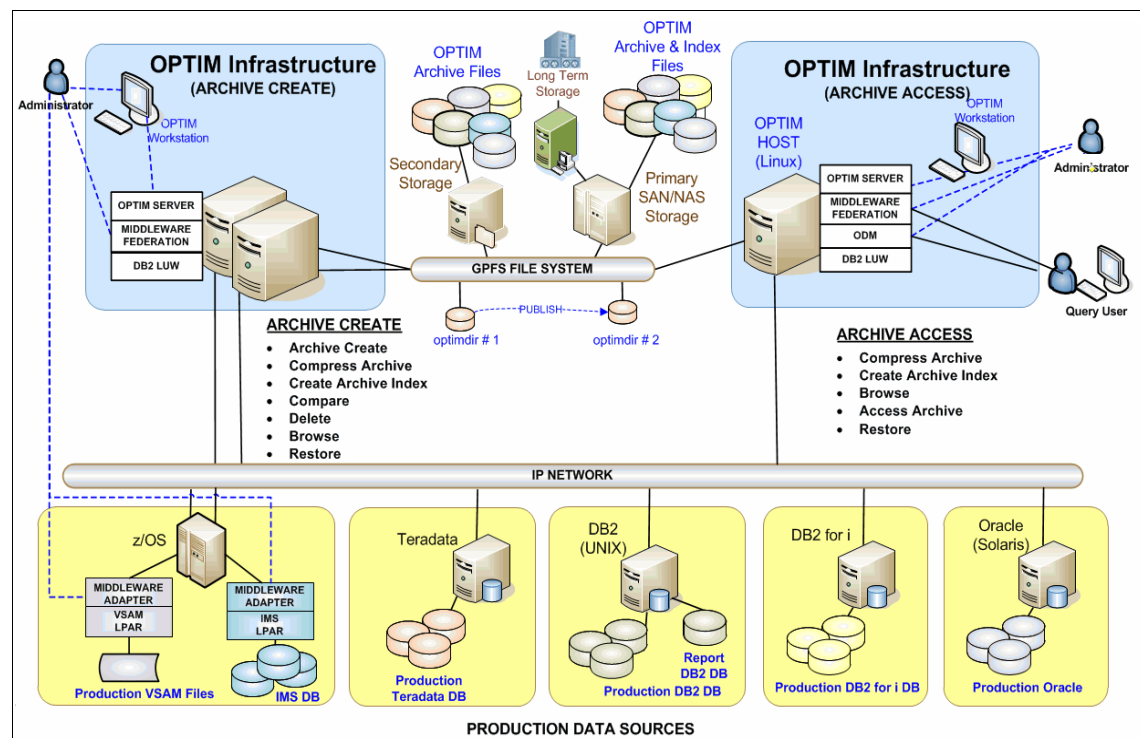


Figure 5-10 Topology for sample large, operationally optimized Optim deployment

## Assumptions

This deployment example includes the following assumptions:

- ▶ The servers are sized for approximately 75% utilization.
- ▶ Multi-processing must be employed when archiving and deleting data.
- ▶ We estimate a 70% compression ratio for archived data (100 GB compresses down to 30 GB). The actual compression rates can vary, depending on actual data.
- ▶ We estimate that the archive files' indexes are 7% to 15% of the source data size and are non-compressed (refer to specifications in Table 4-12 on page 167). The size of the compressed archive files is driven by the compression ratio, which is data dependent. Optim compression can be compared to the results of the Lempel-Ziv-Welch compression algorithm, which yields better results on text data than on binary objects.
- ▶ There are no bottlenecks in the network, storage subsystem, or database server.
- ▶ For efficient access to the archived data, the source database has been recently reorganized, and the catalog statistics are current.
- ▶ No BLOBs, LOBs, CLOBs, or file attachments. The rate of processing will vary widely based on the number and complexity of relationships. The rate of processing also varies based on the size of the data set being archived (actually the number of keys) and the presence of LOBs (LOB, CLOB, BLOB, RAW, or external file attachments).
- ▶ Data growth factors have been included in the analysis.
- ▶ No compare processing with deletes.
- ▶ The final hardware (CPU capacity, storage, and network) sizing and solution architecture must be provided by the implementation team after the detailed design and testing phases.
- ▶ No I/O, network, or database bottlenecks.
- ▶ The DB2 data will be restored via load to a reporting database for queries.
- ▶ I/O reads and writes are less than six milliseconds each (reads from database and writes to archive files).
- ▶ Assume that all production database servers are in the same geographical data center as the Optim server host. For optimal performance, the Optim host must be on the same subnet as the database servers.
- ▶ If more concurrency or faster throughput is needed, the size and number of the Optim hosts might need to be increased. If the elapsed time can be extended, fewer resources will be needed. It is a balance of throughput (desired elapsed time) versus cost (hardware resources).

- ▶ Depending on the actual throughput and growth rates of the data, specific servers might need to be upgraded in year four or five of the deployment based on the projected data growth and operational windows.
- ▶ The access to archived data might be provided by ODM, native application access, or another reporting tool.

## **Optim workstations**

The workstations include the following hardware and software components:

- ▶ Hardware:
  - Minimum: Intel, 2.0 GHz or faster, 1 - 2 cores, 2 GB RAM (can be mobile computer or desk-side computer)
  - One or multiple Optim workstations, depending on the number of users deploying and configuring the InfoSphere Optim Data Growth Solution
- ▶ Software:
  - Microsoft Windows
  - DB2 for Linux, UNIX, and Windows 32-bit client
  - Oracle 32-bit client
  - Teradata 32-bit client
  - ODBC Driver
  - Optim

## **Optim connect studio workstation**

The ODM configuration includes the following hardware and software:

- ▶ Hardware:
  - Minimum: Intel, 2.0 GHz or faster, 1 - 2 cores, 2 GB RAM (can be mobile or desk-side computer)
  - One or multiple workstations, depending on the number of users (minimum two cores, 3 - 4 GB RAM)
  - Can be collocated with Optim workstation
- ▶ Software:
  - Microsoft Windows
  - Optim connect studio
  - DB2 for Linux, UNIX, and Windows 32-bit client
  - Oracle 32-bit client
  - Teradata 32-bit client
  - ODBC Driver

## Federation control center workstation

The InfoSphere Federation Server configuration workstation includes the following hardware and software:

- ▶ Hardware:
  - Minimum: Intel, 2.0 GHz or faster, 1 - 2 cores, 2 GB RAM (can be mobile computer or desk-side computer)
  - One or multiple workstations, depending on the number of users (minimum two cores, 3 - 4 GB RAM)
  - Can be collocated with Optim workstation
- ▶ Software:
  - Federation control center
  - Microsoft Windows
  - DB2 for Linux, UNIX, and Windows 32-bit client
  - ODBC Driver

## Classic data architect workstation

Classic Data Architect is the software component to configure Classic Federation.

## Optim hosts

The Optim hosts have the following configurations:

- ▶ Hosts #1 and #2 (archive create):
  - Hardware:
    - Two 2xQuad-core (8 cores each), 16 GB RAM each, Intel server, *minimum* 2.0 GHz (recommend fastest clock speed)
    - Minimum 2 Dual Port Fibre Channel Card, 4 Gb or 8 Gb for SAN
    - Minimum 2 - 4 Ethernet Network Interface Cards (NICs) 1 Gb or 10 Gb
      - TPC/IP connectivity and NAS
  - Software:
    - IBM InfoSphere Optim Data Growth Solution
    - DB2 for Linux, UNIX, and Windows 32-bit client
    - Oracle 32-bit client
    - Teradata 32-bit client
    - ODBC Driver
    - InfoSphere Federation Server

- ▶ Host #3 (archive access):
  - Hardware:
    - One 2xQuad-core (8 cores), 16 GB RAM, Intel server, *minimum* 2.0 GHz (recommend fastest clock speed)
    - Minimum 2 Dual Port Fibre Channel Card, 4 Gb or 8 Gb for SAN
    - Minimum 2 - 4 Ethernet Network Interface Cards (NICs) 1 Gb or 10 Gb
      - TPC/IP connectivity and NAS
  - Software:
    - IBM InfoSphere Optim Data Growth Solution
    - ODM
    - DB2 for Linux, UNIX, and Windows 32-bit client
    - Oracle 32-bit client
    - Teradata 32-bit client
    - ODBC Driver
    - InfoSphere Federation Server

### **Mainframe LPAR for VSAM**

You install Classic Federation on the VSAM mainframe LPAR.

### **Mainframe LPAR for IMS**

You install Classic Federation on the IMS mainframe LPAR.

### **Estimated storage**

Storage projections for the various data sources are shown in Figure 5-11 on page 208 to Figure 5-16 on page 210:

- ▶ The red line is the projected size of the data source (data and indexes) by the end of year seven without the use of Optim Data Growth Solution.
- ▶ The green line is the projected size of the data source after Optim archiving has been deployed, based on initial and incremental archive and delete processing as specified in 4.7.1, “Large Optim deployment” on page 167.
- ▶ The blue line shows the projected size of the compressed archive files and non-compressed indexes.
- ▶ The purple line is the projected size of the temporary storage that is needed during processing.

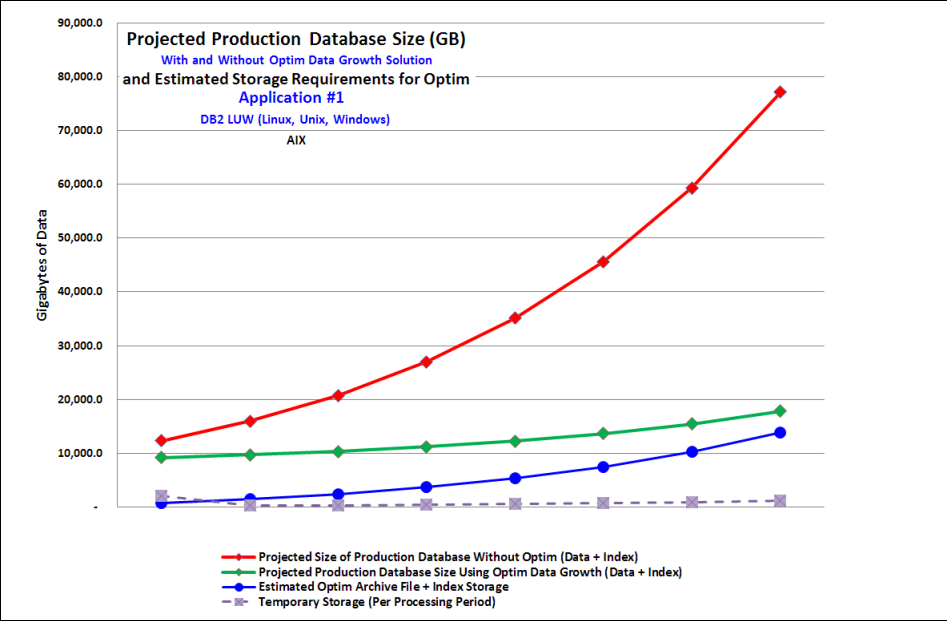


Figure 5-11 Storage estimates for the DB2 on AIX data

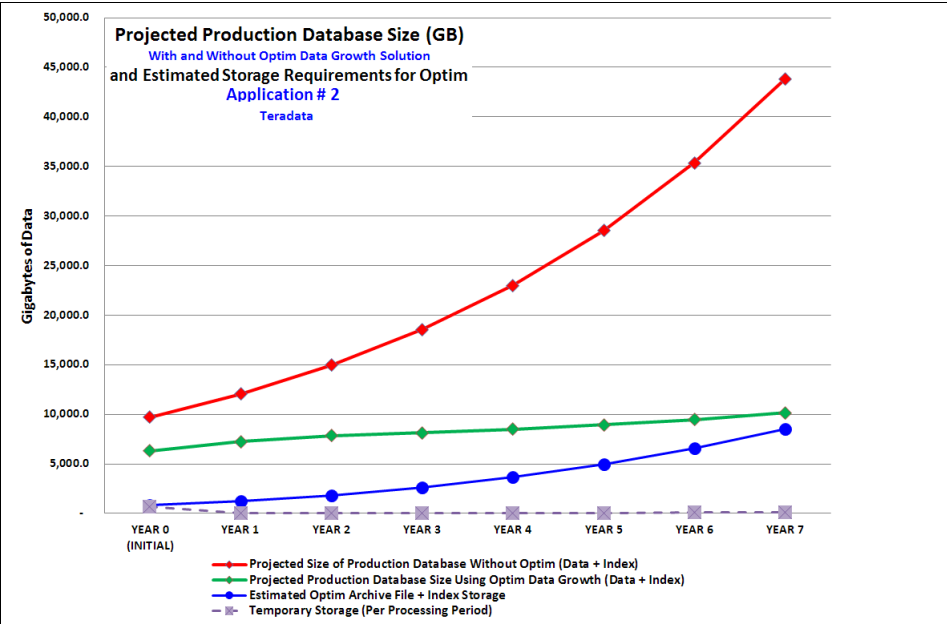


Figure 5-12 Storage estimates for Teradata

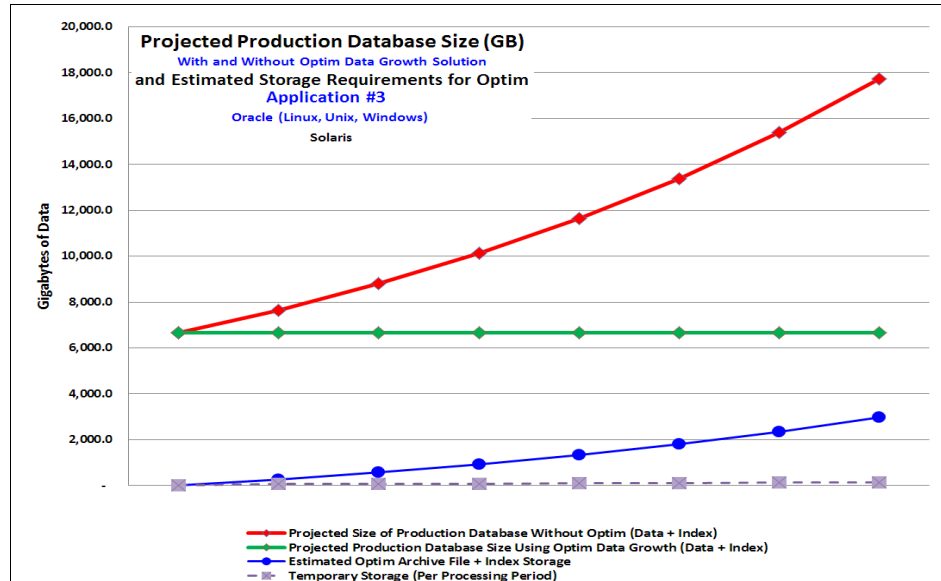


Figure 5-13 Storage estimates for Oracle on Solaris data

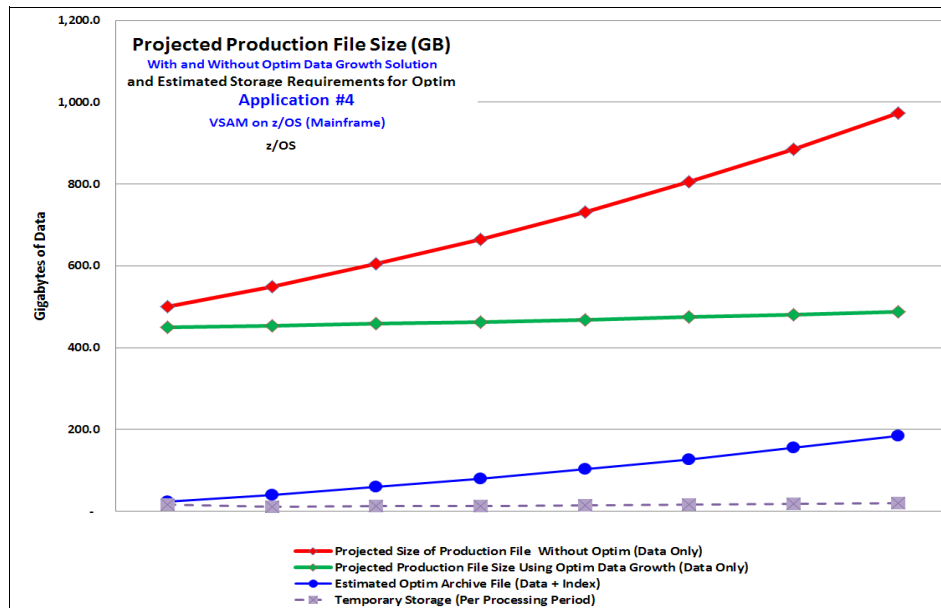


Figure 5-14 Storage estimates for VSAM data

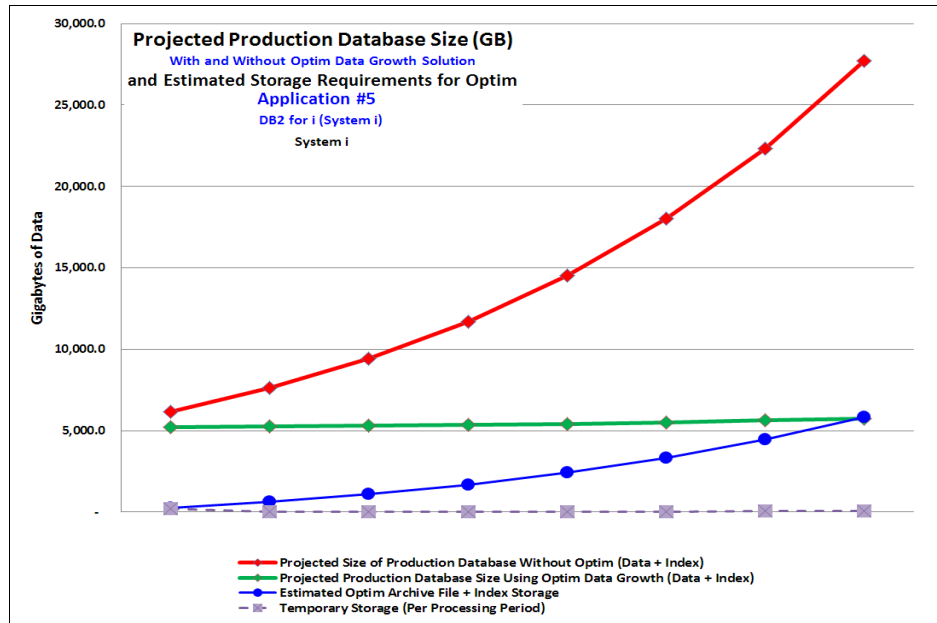


Figure 5-15 Storage estimates for DB2 for i data

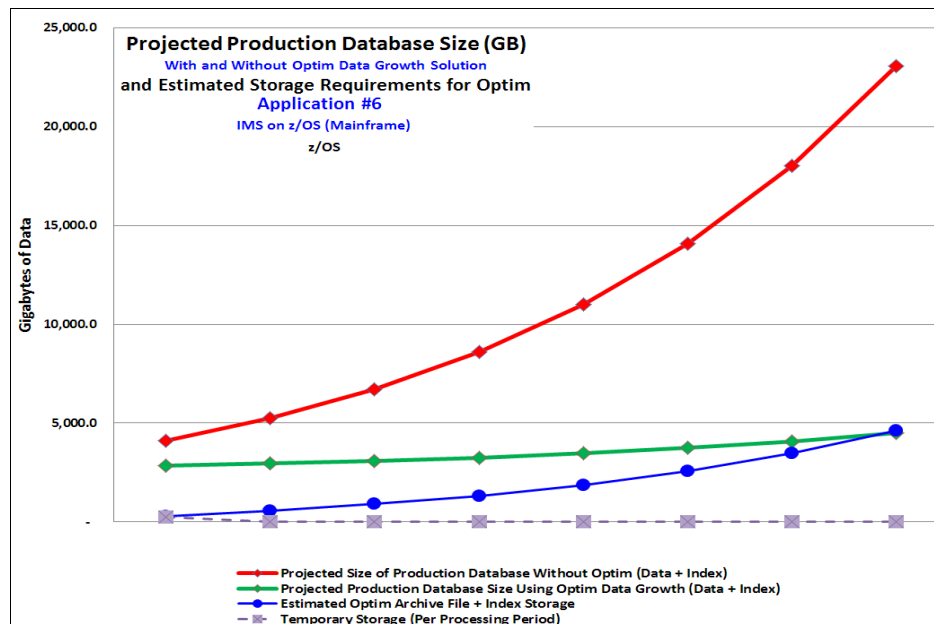


Figure 5-16 Storage estimates for IMS data

## 5.6.2 Sizing considerations

We have provide this sizing example for reference only. Do not use it to size your specific Optim capacity requirements. Your results can vary due to a variety of factors, as outlined in this chapter.

It is important to continually test, monitor, measure, tune, and analyze the performance of the overall infrastructure and make adjustments as necessary. Modeling techniques can provide a starting point for sizing guidelines; you need to determine your actual capacity requirements after you test your Optim solution.

Continually evaluate your requirements to determine if you need more resources to complete the workload in the allocated operational window, or if you can process the workload with fewer resources and take a longer amount of time. Your Optim solution architecture has the flexibility to scale up, scale out, and scale down, based upon your varying requirements.





## Data sources

In this chapter, we investigate the data source stack of Optim for distributed platforms. We cover the following topics:

- ▶ Data source architecture
- ▶ Native data sources
- ▶ Federated data sources
- ▶ Non-relational data sources
- ▶ IBM InfoSphere Classic Federation Server
- ▶ Optim Connect

## 6.1 Data source architecture

The data source architecture is built around the data-driven engine (DDE) in the Optim solutions. The engine for Optim solutions on System z and the engine for Optim solutions on distributed platforms are identical in most cases; however, to capitalize on the native platform, each engine has a specialized data source architecture.

To understand the details of the architecture, it is important to understand the data-driven engine process and how the Optim data-driven engine sees and understands data.

To demonstrate the workings of the Optim solutions, the Agent's native interface for distributed platforms is used in the examples in this chapter. For additional details regarding the native interfaces, reference the *IBM Optim Archive User Manual* for the native solutions.

The access definition (AD) is the interface to the data-driven engine, and the database alias (DB alias) is the interface to heterogeneous data sources.

### 6.1.1 Access definition

The best way to describe an access definition and its functions is to start with a data model. We provide several examples using our standard data model in 3.4.2, "Data-driven processing examples" on page 82. We continue these examples in this discussion. Chapter 4 "Access Definitions", in the *IBM Optim Common Elements Manual* for your solution, provides additional detail.

The access definition consists of many objects, which are grouped in tabs on the *access definition editor*. The two key objects in an access definition are the lists of tables and relationships:

- Tables

Tables that participate in the archive process. Each table is associated with a data source through a defined DB alias. You can determine whether data from a table is extracted as part of the data-driven process or is extracted in its entirety as a reference table. Reference tables are extracted after data-driven processing completes. The list does not imply priority or order of execution. However, within the data-driven list, you must designate one table as the start table. Using overrides, you can change the start table designation at run time (see Chapter 12. "Command Line-Interface" in the *IBM Optim Archive User Manual*).

- Relationships

Relationships among the tables in the list of tables. The Optim solutions automatically populate this list from the database catalog and the Optim directory. The Optim solutions connect to the database catalog using the DB alias for listed tables. If you are processing tables in more than one database, you have a DB alias for each data source and the Optim solutions search the database management system (DBMS) catalogs for each data source to identify the relationships. A relationship between tables in separate data sources cannot be in a DBMS catalog and must be defined in the Optim directory. Note that you can choose to exclude any relationship that does not meet the requirements for your complete business object.

## 6.1.2 Data-driven engine

The data-driven engine builds two types of models from the tables and relationships in the access definition:

- A model of the data flow (the schema), which is used to extract data.
- A referentially intact model, which is used to restore and purge data. The referentially intact model is used to ensure referential integrity when archived data is applied against a database, for example, as when the data is restored (inserted) or purged.

For extract processing, you can display a data flow model by using the Show Steps feature before processing. After extract processing, the data flow model is described in the statistical report. In the statistical report for a delete process, however, you can see that the process maintains referential integrity by first deleting rows from the lowest-level child table, followed by deletions of the parent rows, and so on. The statistical report for an insert process shows that processing occurs in the opposite direction; the process inserts rows into the highest-level parent table first, followed by insertions of the next child table rows, and so on. The discussion of these models and examples in this chapter uses the sample schema that is shown in Figure A-7 on page 510, which is reproduced in Figure 6-1 on page 216.

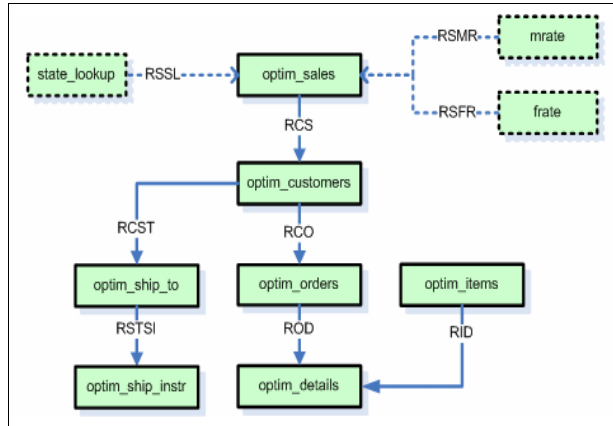


Figure 6-1 Data model

The normal notation for a data model is not adequate to explain the flow of data in a data-driven process. The Optim solutions offer four ways in which to navigate relationships during processing:

- Waterfall (top-down)

This method follows relationships automatically from parent to child. For example, from Products (Optim\_Items), the data-driven engine follows the relationship RID to select rows from Sales Order Line Details (Optim\_Details).

- Reverse waterfall

This method follows relationships optionally from child to select parent rows. For example, to extract Sales Order Details (Optim\_Details) with related Sales Orders (Optim\_Orders), Customers (Optim\_Customers), and Sales Person (Optim\_Sales), you must enable the reverse waterfall option for relationships ROD (Optim\_Items → Optim\_Orders) and RCO (Optim\_Orders → Optim\_Customers).

- More data

This method follows relationships optionally from parent rows that are selected in a *reverse waterfall* flow to select child rows that have not been selected previously. *More data* also triggers a *new waterfall* that automatically engages all related child tables. For example, enable the “more data” option for the relationship RCTS to extract Shipment (Optim\_Ship\_To) rows after extracting Sales Order Line details with related Orders, Customers, and Sales. The new waterfall also propagates to relationship RSTSI to select rows from shipment instructions (Optim\_Ship\_Instr).

► Cycle

This method optionally follows additional rows for an existing relationship. For example, if we need to extract the remaining Sales Order Line Details for the Orders, we extract in the reverse waterfall example.

### 6.1.3 Example

The user interface for the Optim solutions, whether on distributed platforms or System z, presents the four methods as options. Figure 6-2 shows a data flow model that indicates these options. For this example, extract processing begins with Optim\_Items and waterfalls to Optim\_Details. Related Optim\_Orders, Optim\_Customers, and Optim\_Sales rows are selected using reverse waterfall traversal. More data traversal is used to select Optim\_Ship\_To, and automatically the new waterfall selects Optim\_Ship\_Instr rows. A cycle is needed to select additional Optim\_Details rows that relate to previously selected Orders.

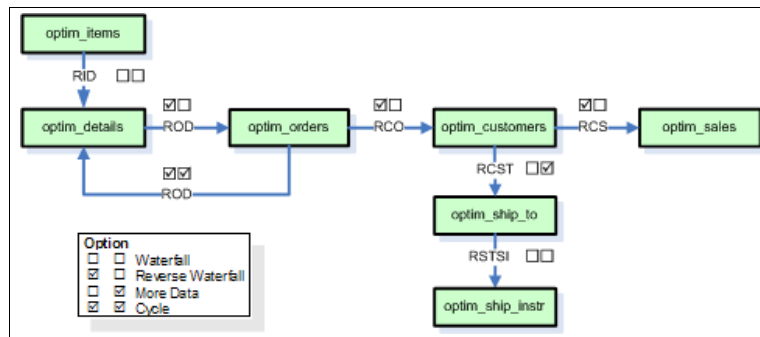


Figure 6-2 Data flow model

### Business requirements

For this example, we use these business requirements:

- Archive all business objects that are defined as sales activities that relate to the product “Tiny Bubbles” with related sales and shipment information.
- Delete archived orders from the data source.
- As required, restore Orders with its related data to the original source or to a new data source.

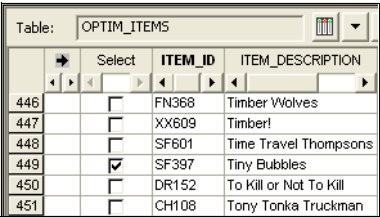
### Technical approach

Here, we illustrate the data flow model and the use of options to define traversal rules. The Indent and Show Steps features are used to build the complete business object and to illustrate aspects of the data-driven engine.

Additionally, you can use the statistical report to illustrate how the Optim solutions run the requirements for archiving, delete, and restore.

**Starting the project**

Because a product, “Tiny Bubbles”, is the criteria for identifying the business objects to be archived, we select Items as the start table. This selection drives the archive process from the Items row for “Tiny Bubbles”. Because a single row defines the business objects to be archived, we do not use selection criteria to select the row but, instead, use a *point and shoot* technique that is more frequently used in Optim test data management solutions. Figure 6-3 shows the point and shoot selection of the Items row for “Tiny Bubbles”.



The screenshot shows a table named 'OPTIM\_ITEMS' with columns 'ITEM\_ID' and 'ITEM\_DESCRIPTION'. Row 449 is selected, showing 'SF397' and 'Tiny Bubbles'.

	Select	ITEM_ID	ITEM_DESCRIPTION
446	<input type="checkbox"/>	FN368	Timber Wolves
447	<input type="checkbox"/>	XX609	Timber!
448	<input type="checkbox"/>	SF601	Time Travel Thompsons
449	<input checked="" type="checkbox"/>	SF397	Tiny Bubbles
450	<input type="checkbox"/>	DR152	To Kill or Not To Kill
451	<input type="checkbox"/>	CH108	Tony Tonka Truckman

Figure 6-3 Point and shoot selection

**Data flow step 1: Waterfall only**

After we have listed the tables required for the business object in the access definition editor and selected the start table, we can use a command to display the organization of the tables and analyze the relationships among them, as the data-driven engine does. Note that to create a printout that is similar to the printout that is used in this example using your Optim solution and the sample database, you must open the access definition editor, select **Optim\_Items** as the start table, and use the solution to add all related tables automatically. Go to the **Relationships** tab and remove all Option 1 and Option 2 default settings. Finally, click **Indent** to create the display and print the results. The result is a printout similar to the printout that is used in this exercise.

Figure 6-4 shows the relationship among the tables.

	Table	Relation	Option		Method
1	OPTIM_ITEMS	[Start Table]			
2	C.OPTIM_DETAILS	RID	<input type="checkbox"/>	<input type="checkbox"/>	Waterfall
3	P.OPTIM_ORDERS	ROD	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]
4	P.OPTIM_CUSTOMERS	RCO	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]
5	P.OPTIM_SALES	RSC	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]
6	C.OPTIM_SHIP_TO	RCST	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]
7	C.OPTIM_SHIP_INSTR	RSTSI	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]

Figure 6-4 Table relationship

Using the Show Steps feature enables us to validate that indeed, rows from Optim\_Items (line 1 and step 1) and Optim\_Details (line 2) are processed with the waterfall setting. As you can see in Figure 6-5, five tables and five relationships are not processed using a waterfall setting.

**EXTRACT STEPS for RBOOK.ADOPT1N2**

**Step 1:**

Extract Rows from Start Table DB.TEST.OPTIM\_ITEMS. No Point and Shoot List, Selection Criteria or Statistical Controls used, therefore Start Table does not need to be revisited, even if part of a Cycle.

**Step 2:**

Extract Rows from DB.TEST.OPTIM\_DETAILS which are Children of Rows Previously Extracted from DB.TEST.OPTIM\_ITEMS in Step 1 using Relationship RID.

**Untraversed Table(s):**

DB.TEST.OPTIM\_ORDERS  
DB.TEST.OPTIM\_CUSTOMERS  
DB.TEST.OPTIM\_SHIP\_TO  
DB.TEST.OPTIM\_SALES  
DB.TEST.OPTIM\_SHIP\_INSTR

**Untraversed Relationship(s):**

RCST  
RCO  
ROD  
RSC  
RSTSI

Figure 6-5 Step 1: Show Steps with only waterfall traversal

### **Data flow step 2: Waterfall with reverse waterfall**

To process business object rows from the parent tables (tables listed with a “P:” prefix), we must use reverse waterfall. Lines 3, 4, and 5 are parent tables, so we select the reverse waterfall option for each line, as shown in Figure 6-6.

	Table	Relation	Option		Method
1	OPTIM_ITEMS	[Start Table]			
2	C:OPTIM_DETAILS	RID	<input type="checkbox"/>	<input type="checkbox"/>	Waterfall
3	P:OPTIM_ORDERS	ROD	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
4	P:OPTIM_CUSTOMERS	RCO	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
5	P:OPTIM_SALES	RSC	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
6	C:OPTIM_SHIP_TO	RCST	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]
7	C:OPTIM_SHIP_INSTR	RSTSI	<input type="checkbox"/>	<input type="checkbox"/>	[untraversed]

Figure 6-6 Selecting the reverse waterfall option for the parent tables

The Show Steps feature (Figure 6-7 on page 220) validates that rows from the parent tables, Optim\_Orders, Optim\_Customers, and Optim\_Sales, are processed. Note that Optim\_Ship\_To (line 6) and Optim\_Ship\_Instr are children (prefix “C:”) of parents that were picked up in the reverse waterfall traversal and are not included.

EXTRACT STEPS for RBOOK.ADOPT1N2	
<b>Step 1:</b>	Extract Rows from Start Table DB.TEST.OPTIM_ITEMS. No Point and Shoot List Selection Criteria or Statistical Controls used, therefore Start Table does not need to be revisited, even if part of a Cycle.
<b>Step 2:</b>	Extract Rows from DB.TEST.OPTIM_DETAILS which are Children of Rows Previously Extracted from DB.TEST.OPTIM_ITEMS in Step 1 using Relationship RID.
<b>Step 3:</b>	Extract Rows from DB.TEST.OPTIM_ORDERS which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_DETAILS in Step 2 to satisfy an RI rule using Relationship ROD.
<b>Step 4:</b>	Extract Rows from DB.TEST.OPTIM_CUSTOMERS which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_ORDERS in Step 3 to satisfy an RI rule using Relationship RCO.
<b>Step 5:</b>	Extract Rows from DB.TEST.OPTIM_SALES which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_CUSTOMERS in Step 4 to satisfy an RI rule using Relationship RSC.
<b>Untraversed Table(s):</b>	DB.TEST.OPTIM_SHIP_TO DB.TEST.OPTIM_SHIP_INSTR
<b>Untraversed Relationship(s):</b>	RCST RSTSI

Figure 6-7 Step 2: Show Steps with waterfall and reverse waterfall traversal

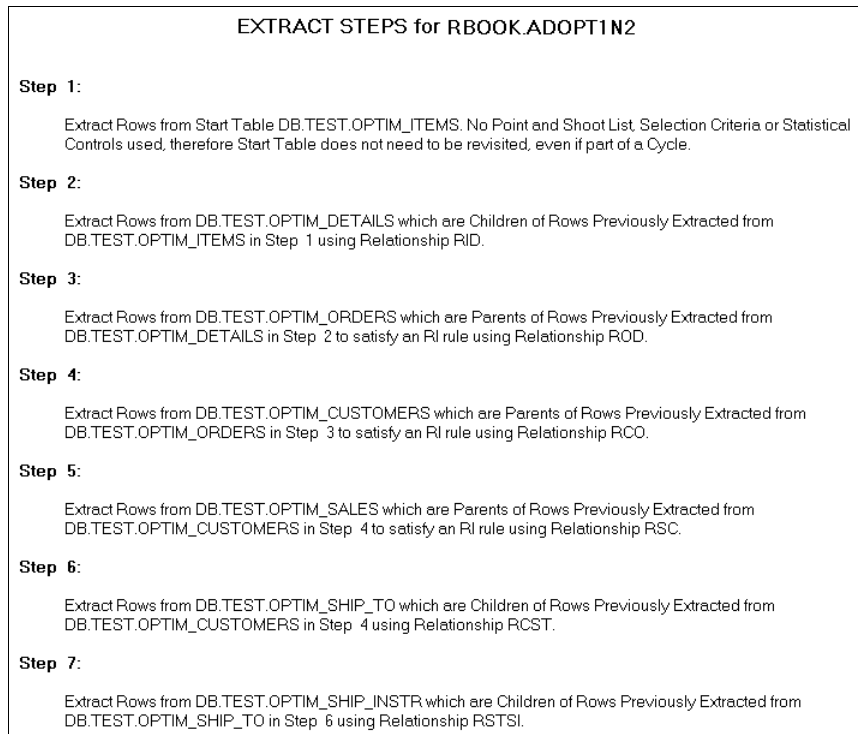
### Data flow step 3: Waterfall, reverse waterfall, and more data

To include rows for Optim\_Ship\_To, you must select More Data for line 6. Note that you can choose to include all related child rows or only the first Optim\_Ship\_To row because you are not interested in knowing the previous addresses. The Optim solution automatically starts a new waterfall from Optim\_Ship\_To and picks up all related rows from the related child tables, in this example, only Optim\_Ship\_Instr (line 7). See Figure 6-8.

	Table	Relation	Option		Method
1	OPTIM_ITEMS	[Start Table]			
2	C:OPTIM_DETAILS	RID	<input type="checkbox"/>	<input type="checkbox"/>	Waterfall
3	P:OPTIM_ORDERS	ROD	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
4	P:OPTIM_CUSTOMERS	RCO	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
5	P:OPTIM_SALES	RSC	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
6	C:OPTIM_SHIP_TO	RCST	<input type="checkbox"/>	<input checked="" type="checkbox"/>	More
7	C:OPTIM_SHIP_INSTR	RSTSI	<input type="checkbox"/>	<input type="checkbox"/>	New Waterfall

Figure 6-8 Select new waterfall option to include the Optim\_Ship\_to child table

The Show Steps (Figure 6-9) validates that Optim\_Ship\_To (line 6) and Optim\_Ship\_Instr (line 7) have been included, leaving all relationships traversed and no tables as orphans.



*Figure 6-9 Step 3: Show Steps with waterfall, new waterfall, and reverse waterfall traversal*

### **Data flow step 4: Add cycles**

At this point, you have almost finished mapping the data flow. Intuitively, you know that an order can include many items. Also, using the data-driven discovery component or relying on direct knowledge, the application subject matter expert (SME) has pointed out that an order can include many items during the analysis phase of your data growth project. To add a cycle that directs your Optim solution to select additional details from the orders that were selected previously, you must change the reverse waterfall on line 3 into a cycle. See Figure 6-10 on page 222.

	Table	Relation	Option		Method
1	OPTIM_ITEMS	[Start Table]			
2	C:OPTIM_DETAILS	RID	<input type="checkbox"/>	<input type="checkbox"/>	Waterfall
3	P:OPTIM_ORDERS	ROD	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Cycle
4	P:OPTIM_CUSTOMERS	RCO	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
5	P:OPTIM_SALES	RSC	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reverse Waterfall
6	C:OPTIM_SHIP_TO	RCST	<input type="checkbox"/>	<input checked="" type="checkbox"/>	More
7	C:OPTIM_SHIP_INSTR	RSTSI	<input type="checkbox"/>	<input type="checkbox"/>	New Waterfall

Figure 6-10 Adding a cycle

The Show Steps (Figure 6-11) shows that, rather than initiate a full cycle, the optimizer for the data driver has added a step (step 4) between Optim\_Orders and Optim\_Details, using the entire set of orders extracted in step 3.

EXTRACT STEPS for RBOOK.ADOPT1N2	
<b>Step 1:</b>	Extract Rows from Start Table DB.TEST.OPTIM_ITEMS. Selection Criteria and/or Statistical Controls are used, these determine rows selected.
<b>Step 2:</b>	Extract Rows from DB.TEST.OPTIM_DETAILS which are Children of Rows Previously Extracted from DB.TEST.OPTIM_ITEMS in Step 1 using Relationship RID.
<b>Step 3:</b>	Extract Rows from DB.TEST.OPTIM_ORDERS which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_DETAILS in Step 2 to satisfy an RI rule using Relationship ROD.
<b>Step 4:</b>	Extract Rows from DB.TEST.OPTIM_DETAILS which are Children of Rows Previously Extracted from DB.TEST.OPTIM_ORDERS in Step 3 using Relationship ROD.
<b>Step 5:</b>	Extract Rows from DB.TEST.OPTIM_CUSTOMERS which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_ORDERS in Step 3 to satisfy an RI rule using Relationship RCO.
<b>Step 6:</b>	Extract Rows from DB.TEST.OPTIM_SALES which are Parents of Rows Previously Extracted from DB.TEST.OPTIM_CUSTOMERS in Step 5 to satisfy an RI rule using Relationship RSC.
<b>Step 7:</b>	Extract Rows from DB.TEST.OPTIM_SHIP_TO which are Children of Rows Previously Extracted from DB.TEST.OPTIM_CUSTOMERS in Step 5 using Relationship RCST.
<b>Step 8:</b>	Extract Rows from DB.TEST.OPTIM_SHIP_INSTR which are Children of Rows Previously Extracted from DB.TEST.OPTIM_SHIP_TO in Step 7 using Relationship RSTSI.

Figure 6-11 Step 4: Show Steps with an optimized cycle

To help you understand true cycles, Figure 6-12 on page 223 shows the processing steps from selecting cycle navigation for all relationships, which is not the best strategy for actual client implementations.

EXTRACT STEPS for RBOOK.CYCLE	
<b>Step 1:</b>	Extract Rows from Start Table DB.TEST.OPTIM_ITEMS. Selection Criteria and/or Statistical Controls are used, these determine rows selected.
<b>Step 2:</b>	Tables DB.TEST.OPTIM_ITEMS, DB.TEST.OPTIM_SALES, DB.TEST.OPTIM_DETAILS, DB.TEST.OPTIM_ORDERS, DB.TEST.OPTIM_CUSTOMERS, Using Relationships RCO, RID, ROD, RSC, Form a Cycle. This Cycle includes the Start Table, DB.TEST.OPTIM_ITEMS.
<b>Step 3:</b>	Extract Rows from DB.TEST.OPTIM_SHIP_TO which are Children of Rows Previously Extracted from DB.TEST.OPTIM_CUSTOMERS in Step 2 using Relationship RCST.
<b>Step 4:</b>	Extract Rows from DB.TEST.OPTIM_SHIP_INSTR which are Children of Rows Previously Extracted from DB.TEST.OPTIM_SHIP_TO in Step 3 using Relationship RSTSI.

*Figure 6-12 Show Steps with actual cycle*

At run time, the tables belonging to the cycle are accessed multiple times in step 2. The cycle continues until the cyclical parent-child relationship is satisfied. Specifically, when no rows remain that satisfy the criteria, whether because the DBMS retrieves no more rows for a given SQL statement or all the rows returned have been processed, the cycle ends.

### ***Indexes and primary keys***

The Optim solutions require a primary key (PK) to uniquely identify a row. The primary key can be DBMS-enforced (with a unique index), informational (for example, an Oracle view in a federated database), or defined in the Optim directory. To accommodate the processing of non-relational data sources and flat files, the Optim solutions support primary keys with unique indexes, non-unique indexes, and no indexes and issue warnings when processing a primary key that has no real enforceable uniqueness. The warnings allow you to determine whether a strategy using primary keys with non-unique indexes or with no indexing is appropriate and acceptable.

For native data sources, you can use the Optim configuration program to create missing primary keys on the basis of unique indexes. Also, you can limit the relationships so that no more than a specified number of related rows are retrieved, for example, to satisfy a requirement, such as “one address”.

## 6.1.4 Data-driven engine

The data-driven engine has a set of tools for processing. We group these tools this way:

- ▶ Process sequencing
- ▶ Process control
- ▶ DB alias
- ▶ DBMS catalog information
- ▶ Independent data access strategies
- ▶ Dependent data access methods
- ▶ Multi-connection key lookup
- ▶ Multi-connection data delete

### Process sequencing

Optim has three process sequences:

- ▶ Extract

The extract process sequence extracts data from the data source by following the data flow sequencing that was established in the access definition. Data from the start table is always the first data to be extracted, followed by the data from the related tables, as described in the Show Steps display. Reference tables are extracted at the end of the process. When fetching multiple rows in one process, the Optim solutions batch the archive actions and file attachments.

- ▶ Delete

Delete follows the data model rules rather than the sequencing that is used during extract. The data-driven engine selects all the tables (including reference tables) in the access definition and builds a top-down data model tree. Note that the start table is not necessarily the top table in the tree, which might have multiple branches and sub-branches. The data-driven engine follows referential integrity rules, deleting from a child table before processing the parent table. After identifying the lowest-level child tables for every branch, the data-driven engine deletes data from the lowest child first and progresses to the next lowest child to delete rows and so on until the delete request is fully satisfied. So that the delete process can be restarted, if necessary, the Optim solution creates a control file, which also provides an audit trail for the process. When deleting multiple rows in one process, the Optim solutions batch the archive actions and file attachments.

- ▶ Insert

Restore uses SQL insert and builds a data model tree, similar to the delete process sequence; however, the restore processing progresses from the top table and down the tree. So that the insert process can be restarted, if

necessary, the Optim solution creates a control file, which also provides an audit trail for the process.

## Process control

During processing, the Optim solutions create several structures to manage data:

- ▶ Control file

To support process restarts and auditing, Optim creates a physical control file (CF) for delete and insert processes. The control file contains row-level Optim actions and return codes from the database.

- ▶ Table control

Optim creates, in memory, table control structures (TCS®) that track tables by primary key. The table control structures are created for each table and reside in memory for as long as the table must be visited. During extract, table control structures ensure that no duplicate data is stored in an archive. For this reason, the statistical report sometimes indicates that the number of “Rows Fetched” is greater than the number of “Rows Written”. This discrepancy can be a result of data deduplication that is performed by table control structures (or the row did not meet the selection criteria). If a primary key does not have a unique index, the Optim solution indicates when more than one row with the same primary key has been deleted.

- ▶ Relationship control

Optim creates, in memory, relationship control structures (RCS) that track rows by relationship. The relationship control structures are created for each relationship, and they reside in memory for as long as data in a relationship must be processed. The relationship control structures identify key values to be used in a relationship and track the cycle in which a row has been extracted so that rows can be identified by cycle.

## DB alias

DB aliases (DBALIAS) are a reference to a DBMS client (and its application programming interface (API)), connectivity information, and services. The Optim solutions use the DB alias as an abstraction layer to enable data access. For example, a fully qualified table name has three parts:

*Dbalias.Creatorid.Tablename*. The data-driven engine is generally DBMS-agnostic and operates under predefined referential constraints.

## DBMS catalog information

The Optim solutions obtain the following information from the DBMS catalog:

- **Primary keys**

The Optim solutions determine whether tables in the business object that is defined in the access definition have primary keys and identify the type of supporting index, if any. In the standard relational model, a primary key must have a unique index. Certain databases support informational primary keys that are used to describe the unique way to identify a row. For example, in Oracle, you can define:

```
ALTER VIEW xyz ADD CONSTRAINT pk1 PRIMARY KEY ("_PARENT") RELY DISABLE NOVALIDATE;
```

An Optim primary key is an informational primary key that does not require a unique index. Optim can find a primary key with a non-unique index or with no index at all.

- **Indexes**

The Optim solutions identify all visible indexes and map them to the primary keys that are required for the process. Because the lack of indexes translates to lower performance, the solutions include a Relationship Index Analysis tool that you can use to identify (and, optionally, create) the appropriate indexes.

- **Relationships**

The Optim solutions identify relationships between tables, including relationships that have no corresponding database constraints. Additionally, the solutions recognize enforced relationships (primary key and foreign key constraints) or scoped relationships (as a reference). Optim relationships are always scoped relationships.

- **Database statistics**

The Optim solutions check whether database statistics have been run. The database statistics provide key distribution statistics to the DBMS query optimizer for an SQL cost analysis. Running statistics might seem to be an expensive operation; however, not running them can be more expensive in the long run.

## Independent data access strategies

Optim uses one of two methods to read data:

- **Table scan**

The Optim solution opens a single cursor and runs a single SQL statement (with or without a WHERE clause) to retrieve data from the data source. Data might be filtered when SQL WHERE is not sufficiently limiting or data has been already archived in a prior step.

- ▶ **Lookup key**

The Optim solution opens a single cursor for each SQL statement and packs from one to 100 key values into a single request. For example, if you retrieve 30 customer rows and set the key lookup limit to three, the solution opens and closes 10 cursors. Each cursor has SQL with multiple ANDs to satisfy the multiple key values, for example, “SELECT A, B C WHERE (B=1 AND B = 2 and B=239) FROM...”

## **Dependent data access methods**

Drivers for certain DBMS clients support high-performance processing. The following features in the Optim solutions are enabled only for the DBMS that supports them:

- ▶ **Array fetch**

DB2 and Oracle provide high-efficiency data retrieval processing through an array fetch. A simplified description of this mechanism is to envision one buffer with dynamic SQL predicates that are stored in a separate buffer. For example, if “ORDER\_ID = ?” is associated with a buffer with 100 values, the database parsing and execution are much faster than 100 SQL statements.

- ▶ **Array delete**

DB2 and Oracle provide high-efficiency data delete processing using array delete, which is similar to array fetch. DB2 can also provide a row-level response code that enables tracking singular failures and makes the process traceable and auditable. When the Oracle array delete feature is used during a delete process, the rows not found are listed as deleted in the delete process report. To turn off the array delete feature, clear “Use Oracle Array Delete” on the Database tab on the Product Options dialog. For most cases in which array delete is suitable but not available, Optim has implemented a function to perform “Multi Key Delete” that packs multiple row deletes into one SQL statement.

## **Multi-connection key lookup**

The Optim solutions have a mechanism to increase parallelism by allowing multiple connections during archive (extract) processing. When certain conditions are met, Optim determines the number of concurrent connections that are allowed and divides the fetch buffer pool and driving keys into equal portions before starting to process independent asynchronous threads.

Before you can use this facility, you must enable it by using the Product Options and selecting the Database tab to allow parallel processing and set the Maximum Database Connections to greater than 1 (Figure 6-13 on page 228).

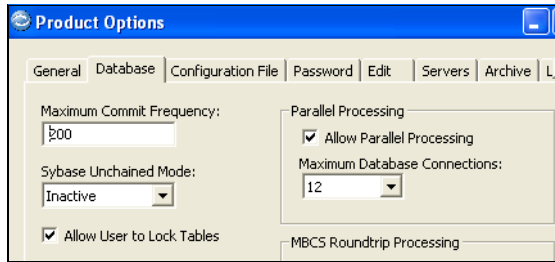


Figure 6-13 Product Options: Enabling parallel processing

Also, you can enable multiple database connections at the archive request level (Figure 6-14). Therefore, you can tune each archive request differently when extracting data using relationships. The start table and reference tables cannot participate in parallel processing, because the Optim solution does not have any data in memory from which to drive the process.

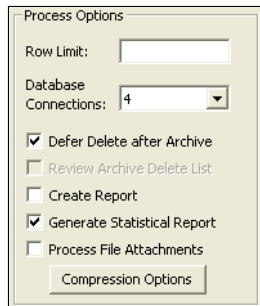


Figure 6-14 Multiple connections

The Optim solutions set an internal limit that is equal to twice the number of CPU cores in the server, so that a single core CPU can have a maximum of two concurrent connections for an archive process. A 4-core CPU can have up to eight concurrent connections and so on.

Because threads reduce only the wait time between database requests, multiple connections typically produce a 4% - 50% improvement. The real advantage in using multiple connections is to reduce the opportunities for Optim to be idle while processing a table. To illustrate, Figure 6-15 on page 229 shows the main thread (*Optim*, in blue) starting two asynchronous threads (*DBMS*, in yellow). Each *DBMS* thread opens the cursor and executes the SQL.

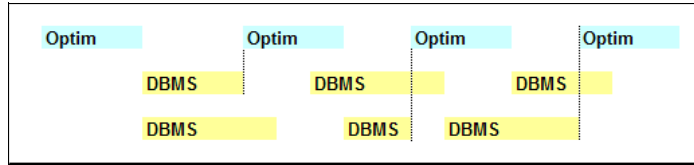


Figure 6-15 Optim main thread reactivated when more data is available

Until a DBMS thread returns control, the Optim solution has no data to process. After the Optim thread processes data, the asynchronous thread is dispatched again and either continues fetching data for the current cursor or processes the next cursor request. The Optim thread checks the DBMS threads and if data is available, the Optim thread continues processing. If data is not available, the Optim thread waits.

In most databases, large objects (LOBs) are not stored in-line with the row, and additional calls are required to fetch LOB content. Certain database management systems restrict the number of LOBs that you can have in process for a given connection. It is common to see the Optim solutions reduce throughput to meet a DBMS restriction to process the large amount of data coming from the source database.

**Row-level archive actions:** If you implement archive actions at a row level, the Optim solutions batch all “Before {action} row” archive actions in the buffer and run them before fetching the data and performing the same for “After {action} row” archive actions. For best performance, use row-level archive actions sparingly, if at all.

Note that each SQL request is in two parts:

- ▶ Part 1 processes the SQL and prepares the record set.
- ▶ Part 2 fetches data.

The first request for a cursor normally takes longer (create record set and fetch) than the subsequent fetch requests. A process that has many cursor requests, which are satisfied in one round-trip because all data fits in the Optim buffers, produces the highest DBMS wait as percent of process. In this case, multiple connections provide the greatest improvement. A process with fewer cursor requests and larger sets of data will fill the Optim buffers and require more fetch requests per cursor, resulting in a lower performance improvement from multiple connections.

## Multi-connection delete

Delete is not performed by following the data flow model, but by interpreting the data model. The delete process is always from the lowest-level table (child), including reference tables, to the higher (parent).

A good way to decide the optimal number of concurrent delete connections needed is to look at the number of branches in your data model or schema. A data model of 200 tables and six branches benefits from approximately six database connections, while a data model of 3,000 tables and two branches benefits from only two database connections. The data model for this example appears to have three branches: Optim\_Ship\_Instr, Optim\_Details, and Optim\_Items. In reality, Optim\_Items is a parent until Optim\_Details is processed, and two connections will suffice. If Optim\_Items has additional parents then, after Optim\_Details is processed, the remaining tables in the tree are processed independently.

If deleting from both branches, you use two connections, which the data-driven engine uses to process in parallel. For this example, however, while Optim\_Details and Optim\_Ship\_Instr are the lowest-level tables, Optim\_Ship\_Instr rows are not candidates for deletion because shipping instructions must be retained for future orders. Only Optim\_Details and Optim\_Orders are deleted. Because Optim processes one table delete per thread, multiple database connections cannot expedite the process.

## Archive and extract processes

The data-driven engine has been architected to support archiving heterogeneous business objects in a production environment. The data-driven engine uses a single SQL FROM clause per table (single-table SQL). To avoid Cartesian products that duplicate data and heavy SQL statements, such as JOIN with ORDER BY, the data-driven engine tracks data as it is retrieved.

Again, using the sample data that is described in Appendix A, “Data model” on page 501, consider a project in which you want to archive Optim\_Customers with related Optim\_Orders. In a traditional SQL statement, you JOIN Optim\_Customers with Optim\_Orders and Optim\_Orders with Optim\_Details. For a customer that has 100 orders of ten items each, the Cartesian product of the join retrieves the customer row 1,000 times, requiring the removal of 999 duplicates of the customer row. The same principle applies to the orders that, on average, are duplicated 10 times (one per item). One approach is to sort the data to simplify the process for removal of duplicates. This method, however, expends DBMS, network, and CPU time to process and remove the unnecessary duplicates (999 customers and 900 orders).

The Optim solutions, however, archive one Customer row, and using that information, extract the 100 related orders. For each order, the solution

interrogates the database ten times to get a total of 1,000 details. This architecture provides the unique capability of treating extracts from a single database and those extracts from heterogeneous data sources exactly the same. Additionally, the architecture enables you to start an extract at any point in the data model and navigate to other tables in any direction. This unique capability and flexibility has made the Optim solutions a strategic component of many information lifecycle management (ILM) projects worldwide.

Optim does not store duplicated data, so it must filter out duplicated data using table control structures. In our sample project, the table, Optim\_Items, is processed using the relationships ROD and RID. Data from both Optim\_Items and Optim\_Orders drives the selection of Optim\_Details.

In the statistical report (Figure 6-16) for “Data flow step 2: Waterfall with reverse waterfall” on page 219, you can see that, for Optim\_Details, the data-driven engine retrieves (rows fetched) and stores (rows written) 32 rows.

```
Step 2: Table Name: DB.OPTIM.OPTIM_DETAILS
DBMS: DB2 LUW      Version: 09.07.0000 Columns: 4
Cycle: No          Lobs: No          Est. Rows: 21427
Row Length: 38     DB Connections: 1 Select with UR: No
Fetch Buffer Size per Connection: 512K
PK w/Index: 0      PK wO/Index: 0
FK w/Index: 0      FK wO/Index: 1
Parent Strategy: No Keys
Dependent Strategy: only one Key

Relationship: DB.OPTIM.OPTIM_DETAILS.RID

Lookup Keys: 1      Direction: dependent Indexed: No
Keys Per Cursor: 50 DB2 Lookup Cost: 0.000000 p
Key Length: 5       Access: Not Forcing
Lookup SQL:          ITEM_ID = ?

DBMS Access: RID

Access Type: FK Lookup Keys Per Cursor: 50
Open Cursor: 1      Rows Fetched: 32
Rows Written: 32     Time in DBMS: 100
                    Time in DBMS Archive Actions: 0
Process Time: 0:00:01 Rows Per Sec: 32
```

*Figure 6-16 Step 2: Data retrieval using relationship RID*

For “Data flow step 4: Add cycles” on page 221, the Optim data-driven engine traverses the relationship ROD and revisits Optim\_Items. Several of the items are already archived, so the data-driven engine filters out duplicate rows as they are fetched. Looking at the report for step 4 in Figure 6-17 on page 232, you see that 90 rows are fetched and only 58 rows are written. The Optim solutions do not attempt to include an SQL AND and NOT for rows that are already tracked the benefit does not outweigh the performance cost.

```

Step 4: Table Name: DB.OPTIM.OPTIM_DETAILS
DBMS: DB2 LUW      Version: 09.07.0000 Columns: 4
Cycle: No          Lobs: No      Est. Rows: 21427
Row Length: 38     DB Connections: 1 Select with UR: No
Fetch Buffer Size per Connection: 512K
PK w/Index: 0      PK wo/Index: 0
FK w/Index: 32     FK wo/Index: 0
Parent Strategy: No Keys
Dependent Strategy: Key Lookup due to small number of keys

Relationship: DB.OPTIM.OPTIM_DETAILS.ROD

Lookup Keys: 32     Direction: dependent Indexed: Yes
Keys Per Cursor: 50 DB2 Lookup Cost: 0.000000
Key Length: 13     Access: Not Forcing
Lookup SQL:        ORDER_ID = ?

DBMS Access: ROD

Access Type: FK Lookup Keys Per Cursor: 50
Open Cursor: 1      Rows Fetched: 90
Rows Written: 58    Time in DBMS: 100
                  Time in DBMS Archive Actions: 0
Process Time: 0:00:01 Rows Per Sec: 90

```

Figure 6-17 Step 4: Data retrieval using relationship ROD

To demonstrate the optimizing strategy that is used when the data-driven engine identifies a better method to fetch data, the second step in Figure 6-12 on page 223 visits `Optim_Items` eight times. As you can see in Figure 6-18, on the third occurrence, the data-driven engine automatically switches from one to four connections (DB connections: 4), while in the other seven visits, the data-driven engine decided that one connection was the best strategy. This example shows that running database statistics allows your Optim solution to perform at its best.

```

Step 2: Table Name: DB.OPTIM.OPTIM_DETAILS
DBMS: DB2 LUW      Version: 09.07.0000 Columns: 4
Cycle: Yes         Lobs: No      Est. Rows: 21427
Row Length: 38     DB Connections: 4 Select with UR: No
Fetch Buffer Size per Connection: 512K
PK w/Index: 0      PK wo/Index: 0
FK w/Index: 2579   FK wo/Index: 0
Parent Strategy: No Keys
Dependent Strategy: Key Lookup due to small number of keys

Relationship: DB.OPTIM.OPTIM_DETAILS.ROD

Lookup Keys: 2579   Direction: dependent Indexed: Yes
Keys Per Cursor: 50 DB2 Lookup Cost: 0.000000
Key Length: 13     Access: Not Forcing
Lookup SQL:        ORDER_ID = ?

DBMS Access: ROD

Access Type: FK Lookup Keys Per Cursor: 50
Open Cursor: 52     Rows Fetched: 5839
Rows Written: 5807  Time in DBMS: 25
Process Time: 0:00:01 Rows Per Sec: 5839

```

Figure 6-18 Access with multiple connections

## Data purge process

The Optim data-driven engine delete functionality is geared to protect data and its integrity. We have already explained that Optim deletes from the lowest-level table and follows the upward branches, thereby ensuring that referential integrity

rules are followed. When multiple connections and multiple branches are used, the process is actually running in parallel.

The delete process can be restarted because the Optim solutions create a control file that augments the database checkpoint mechanism. For example, assume that an Optim solution is deleting 1,000,000 rows and that the database has a checkpoint interval of 1,000. The operator cancels the delete process while Optim is issuing a delete on row 830,709. The database rolls back the delete for the range 830,000 - 830,709. When you restart the delete process, the solution automatically restarts deleting the correct table at row 830,000.

The control file ensures the restart capability. In addition, there are decreasing levels of protection to help you ensure that you have deleted what you archived. A primary key alone does not necessarily protect against errors in deleting:

- Before deleting rows, comparing the index with a primary key that is supported by a primary key index provides the highest level of certainty and might be the best option for legal compliance. Because the content of a database row (including any LOBs) is compared to the row with the same primary key in the archive, before it is deleted, you validate that the archive data is an identical copy of the deleted source data (SQL fetch followed by cursor delete). You can trade this level of security for performance by eliminating certain comparisons of LOBs and even tables, which might be acceptable to you under certain conditions.

Deleting without comparing on a primary key supported by a unique index deletes only one row; however, it provides no assurance that the deleted row contains the same data as the archive. If you must meet compliance requirements and must ensure that the archive is the new system of record, we do not recommend this option.

- Comparing before deleting rows with a primary key where a foreign key index allows duplicate entries might be needed in certain cases in which the application enforces referential integrity (for example, in certain modules of Oracle E-Business Suite). The Optim solution fetches a row matching the primary key and, based on the results of a comparison of the row content with the archived row, the Optim solution executes, fails, or duplicates the delete.

If you delete rows with a primary key supported by a foreign key index without comparing, the delete all rows that match primary key values are deleted and the rows are noted as duplicates in the control file. If using array delete with Oracle, there is a further issue in that the “extra” purge is not recorded in the control file. You can avoid this issue by disabling the Oracle array delete in Optim.

- The results differ slightly when comparing before deleting rows with a primary key that is not supported by a primary key index. In this case, the Optim solution must scan the database to locate data and compare every row

fetches with the table control structures. Upon finding a row with a matching primary key, the solution compares the rest of the content and, depending on the results of the comparison, notes the row as “compare failed” in both the control file and table control structures, or as deleted and performs a cursor delete. If succeeding rows with the same primary key (PK) are retrieved, Optim notes those rows as “row already deleted” in the control file. If you are deleting without comparing before deleting rows with a primary key that is not supported by a primary key index, duplicate rows are also marked as “row already deleted”.

## Offline delete on System z

Optim solutions on System z offer an offline batch delete process that allows you to avoid the overhead of DB2 logging, which is an important consideration when deleting large quantities of data from your database. This offline delete process has the same effect as deleting “delete after archive” data directly from the database.

In the offline delete process, Optim treats one or more compatible unload files like the results of a table scan with no index. It compares archive file data to create a new file that excludes the archived data marked “delete after archive.” You can then load this new file, which lacks the deleted data, into the database. This process does not offer traceability due to the fact that the DB2 log file is not used during the process.

Figure 6-19 depicts the System z offline delete process.

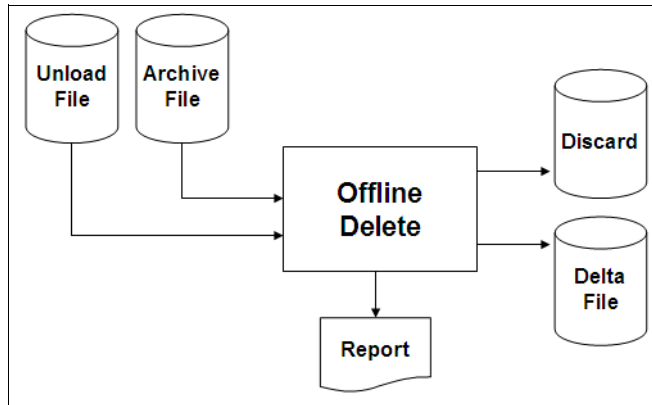


Figure 6-19 Offline delete

## Data insert process

The Optim data-driven engine restoration through SQL insert functionality is designed to manage referential integrity and possible data changes. Unlike a

delete process, the Optim insert process navigates the model from the top down. To maintain referential integrity, the highest-level parent is restored first, followed by child tables.

A restore using insert is a process that can be restarted that requires a control file to ensure the same process consistency and traceability that are required for a delete request. In certain cases, database constraints create the parent relationship as a cycle as, for example, when an account must be part of a portfolio, and a portfolio must include, at least, an account.

Optim processing and referential database constraints can be in conflict. If so, the Optim solutions support the suspension of DBMS constraints and triggers to enable restoring data.

Similar to delete processing, the Optim solutions support comparing rows before restoration. The same general discussion regarding primary keys and indexes applies here.

Optim insert logic changes depending on your process options:

- ▶ Insert:
  - If a row exists in the source and does not exist in the destination, the process inserts the new row.
  - If a row exists in both the source and destination tables, the process bypasses that row. These rows are discarded and noted in the control file.
- ▶ Mixed:
  - The mixed (selective update and insert) process option is valid only for the Mixed and No Tables Delete Options. You cannot update rows from tables that you have just deleted. The data-driven engine assumes that rows will be added without conflict, so it performs a blind SQL insert.
- ▶ Update only:
  - If a row exists in both the source and destination tables, the process updates that row.
  - If a row exists in the source and does not exist in the destination, the process bypasses that row. These rows are discarded and noted in the control file.
- ▶ Update/Insert:
  - If a row exists in the source and it does not exist in the destination, the process inserts the new row.
  - If a row exists in the source and destination tables, the process updates that row.

## 6.1.5 DB alias

Optim can access several databases simultaneously; however, each database must have a unique DB alias that is stored in the current Optim directory. A DB alias provides the required level of abstraction to the Optim data-driven engine. The DB alias is used as a high-level qualifier for a database table name to provide a single-name association for parameters that are required to connect to the database.

Optim solutions on System z interface with DB2 using these tools:

- *Call attachment facility (CAF)*, which is a part of the DB2 code and allows other programs to connect to and use DB2 to process SQL statements, commands, or instrumentation facility interface (IFI) calls. With the CAF, the Optim solution establishes and controls a connection to a local DB2 database.
- *Distributed Relational Database Architecture (DRDA)*, which enables applications and database systems on disparate, connected systems to communicate and interoperate. With DRDA, Optim can establish and control a connection to a remote DB2.

On Linux, UNIX, and Microsoft Windows platforms, the Optim solution can connect to several categories of data sources, as shown in Figure 6-20.

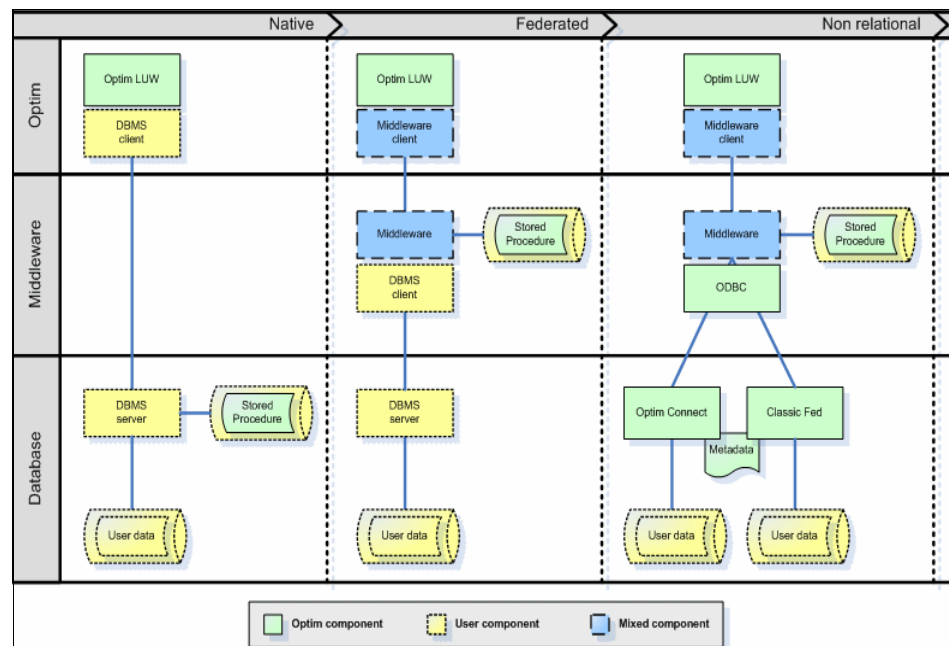


Figure 6-20 Data source categories

Figure 6-20 on page 236 shows the following data source categories:

- ▶ **Native**  
Data sources are accessed using DBMS client software and a set of stored procedures.
- ▶ **Federated**  
Data sources are accessed using native DBMS federated (middleware) to your target database either using Open Database Connectivity (ODBC) or native driver.
- ▶ **Non-relational**  
Data sources are accessed using a Federated data source and a module that provides relational metadata and translates SQL into native calls (middleware).

A DB alias requires DBMS client software to communicate with the source database. Currently, the Optim solutions support seven database management systems natively:

- ▶ DB2 for z/OS
- ▶ DB2 for Linux, UNIX, and Windows (or Federation Server)
- ▶ Informix
- ▶ Oracle
- ▶ SQL Server (Microsoft Windows platform only)
- ▶ Sybase
- ▶ Teradata

An Optim directory can reside on any of these databases. Although the practice occurs more often in development than in production environments, the Optim solutions can use a single DB alias to access data and an Optim directory residing in the database. However, using a single DB alias to access data and an Optim directory residing in the database is not an advisable practice. If you share the same connection, you share the same security level; therefore, you lose granularity in your separation of roles.

**Important:** For Optim solutions on distributed platforms, all DB aliases must be created from a Microsoft Windows machine, using the Optim configuration program. The DB alias information is not verified until a connection is required by the Optim solution.

## 6.2 Native data sources

The discussion of Optim data-driven engine has already provided you with the required knowledge to understand how native data source support is implemented. If you have the proper indexes and statistics, the process self-adjusts in the vast majority of the cases.

Your checklist for a native data source must include the following components:

- ▶ Complete business object:
  - Logical data model with relationships with data cardinality (1:1, 1: $M$ , 1:[0-1], 1:[0- $M$ ])
  - Physical data model with database type, indexes, constraints, and approximate table size
  - Data rules (for example, an order can have zero or more details, and the order must be associated to a customer) and enforcement method (database or application)
  - Source-data code pages
  - LOB specifications
  - File attachment specifications
- ▶ DB alias:
  - Source database type and version
  - Path for source database 32-bit client (configured)
  - Path and name of native database loader
  - Connection string inclusive of connect user (and password)
  - DB loader parameters
  - Maximum concurrency allowed (lookup key and delete)
- ▶ Access definition:
  - List of related tables
  - List of reference tables
  - Active relationships
  - Show Steps report
  - Optim index analysis report
- ▶ Optim product configuration:
  - Required configurable options have been included or excluded (for example, Oracle array delete)
  - Fetch buffer size
  - Delete buffer size

- ▶ Statistical report (requires running an archive or delete request):
  - Validate report warnings.
  - For each DBALIAS, validate that the DB version matches the specifications.
  - Validate if and why the following messages have been issued:
    - Scan - No Index
    - User Forced Scan
    - User Forced Key Lookup
    - User Forced Scan
    - User Forced Key Lookup
  - For each key lookup access, validate how many connections have been used.
  - For each lookup key strategy, validate that “Keys Per Cursor” is set correctly.

## 6.2.1 Configuration

The Optim configuration program in Microsoft Windows creates the DB alias definition (Chapter 7, “DB Aliases” of *IBM Optim Installation and Configuration Guide*). The user interface displays the process as one step. However, in reality, three separate actions occur:

- ▶ Packages, plans, or stored procedures (SPs) are installed into the database. The user account that is used to connect must have the authority, through system privileges or roles, to catalog the packages, plans, or stored procedures under the appropriate table identifier. Only the configuration utility uses this user ID.
- ▶ DB alias connections and defaults are configured. During this step, Optim stores in the Optim directory the connection information, default user without password, database unique identifier, default values for table and index creation, and more. Optim uses the user account and password for the Optim directory to perform this task. Note that Optim dynamically determines the code pages. For further information, see Character Formats in Chapter 1 of the *IBM Optim Installation and Configuration Guide* for your Optim solution. In certain instances, however, the code page is not recognized (ODBC) and the DB alias code page is used.
- ▶ Connection information is set up in the Microsoft Windows registry. In Linux and UNIX, you must add this connection configuration by following the instructions in Appendix A, “Install and Configure the Server under UNIX or Linux” in the *IBM Optim Installation and Configuration Guide*.

## 6.2.2 Multiple language considerations

When embracing a project dealing with structured data, you must be sure that any data manipulation is consistent with the language with which the data is associated.

The Latin alphabet and numbers can be represented in single-byte ASCII using seven bits (0x00-0x7F). Certain national languages can also be represented in single-byte ASCII using the remaining characters (0x80-0xFF). They also require a code page (CP) to represent them. For example, code page 874 represents Thai, and code page 861 represents the Icelandic language. Archiving data requires capturing code page information for later display of the archived data.

Specific Asian languages require character sets that expand beyond the few hundred characters that are supported by single-byte ASCII. For this reason, Unicode is a computing industry standard for the consistent encoding of 109,000+ characters that use a two-byte descriptor. Unfortunately, 50% or more of data remains in single-byte format, so the use of Unicode doubles the space requirements without providing any real advantage. The Unicode Transformation Format (UTF) addresses this issue.

UTF-8 encodes all the characters between 0x00 and 0x7F in one byte, and all the remaining (theoretical) four billions in two to six bytes. This capability enables compatibility with the old 7-bit ASCII without wasting space. UTF-8 requires libraries or code to interpret each character (using more CPU) and ensures that the characters are not truncated mid-character.

Certain implementations, such as Oracle JA16SJIS, have multiple characters that are mapped to a single Unicode character. When these characters are converted from Unicode back to multi-byte (a round-trip), the original character might not be returned. Your Optim solution can detect round-trip conditions so that you can customize the appropriate behavior.

UTF-16 encodes 1,112,064 characters in two or four bytes. UTF-16 is more common in the Microsoft Windows and Java environments. In the original implementation of Microsoft Windows NT 4.0, the Universal Character Set (UCS) had two-byte encoding and a set limit of 63,488 characters, but UTF-16 is UCS upward compatible. UTF-16 also requires libraries or code to interpret each character and to ensure that the characters are not truncated mid-character.

A DB alias and the Optim directory must be compatible to operate in a multiple code page environment. Figure 6-21 on page 241 shows the compatibility matrix.

Optim Directory Setting	DB Alias Setting Requirement		
	Single-byte	UTF-8	MBCS
Single-byte Format	<input checked="" type="checkbox"/>		
UTF-8	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
MBCS			<input checked="" type="checkbox"/>
DB Alias Setting	Optim Directory Setting Requirement		
	Single-byte	UTF-8	MBCS
Single-byte Format	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
UTF-8		<input checked="" type="checkbox"/>	
MBCS			<input checked="" type="checkbox"/>

Figure 6-21 Compatibility matrix of DB alias and Optim directory

At run time, the Optim solution identifies the environment in which the request is running and validates that the product license is valid. Optim also logs this information.

## 6.3 Federated data sources

You manage access to a data source that is not a native data source by implementing a federated data source (FDS).

You accomplish the federation of data sources with a federation engine, which can be one of these engines:

- ▶ IBM InfoSphere Federation Server
- ▶ Oracle Database Gateway for ODBC

The use of these federation engines to join information across databases for archiving data is not supported. You must implement heterogeneous data support natively, using multiple DB aliases.

### 6.3.1 Architecture

For non-native data sources to be processed by the Optim solutions, the Federation Server client is collocated with the Optim server. Also, the architecture is implemented as a layer between Optim and the data source, as shown in Figure 6-22 on page 242.

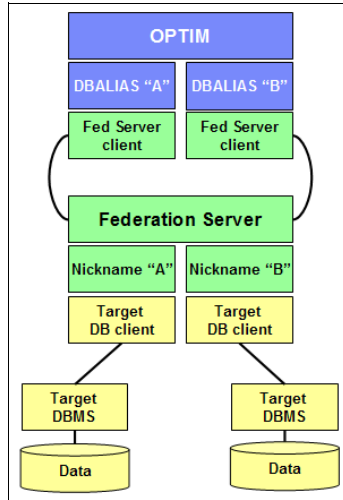


Figure 6-22 Federated data source using InfoSphere Federation Server

Oracle federation architecture (Figure 6-23) is implemented similarly as an addition to the architecture of the Optim solution.

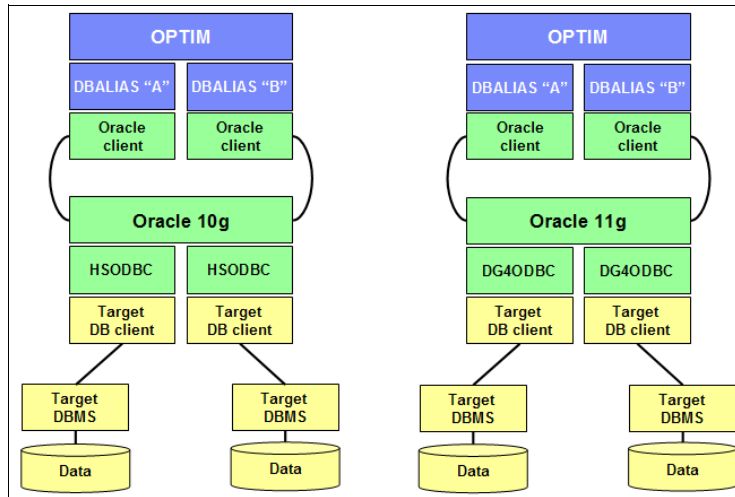


Figure 6-23 Federated data source using Oracle

## 6.3.2 Using Federation Server

In the Federation Server configuration, the Optim solution optimizes requests to the Federation Server engine. The Federation Server optimizer converts the SQL to the SQL of the target database and identifies the best access strategy with the

source database. The wrapper module is responsible for invoking the appropriate target database client API.

## Setting up Federation Server

To use Federation Server, you must perform these steps:

1. Install and configure Federation Server.
2. Create a wrapper for ODBC and define the server.
3. Discover the federated data sources.
4. Create a DB alias for Federation Server.

### *Installing and configuring Federation Server*

To benefit from the latest features and programming fixes, install the most recent version of Federation Server. You can find complete information about the installation of Federation Server V9.7 (the most recent version at the time of publishing this book) at this website:

[http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.swg.im.iis.db.prod.fed.nav.doc/topics/iivfed\\_installing\\_l1.html](http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.swg.im.iis.db.prod.fed.nav.doc/topics/iivfed_installing_l1.html)

For later versions of Federation Server, you can search the web for “Installing IBM Federation Server” without quotation marks.

### *Creating a wrapper and defining the server*

You can find details about how to configure an ODBC wrapper in the DB2 for Linux, UNIX, and Windows Information Center. For Version 9.7 of Federation Server, this information is at this website:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.swg.im.iis.found.conn.fw.odbc.doc/topics/tlsodb01.html>

You can obtain additional configuration steps for supported data sources in the same information center.

## Discovering the federated data sources

You can discover a data source from the Federation Server Control Center by expanding the Object view tree under the name of the ODBC wrapper, right-clicking **Server Definitions** and selecting **Create**. On the Server Definitions dialog, click **Discover** to display the ODBC data sources that are available to the server. Because certain information might not be captured when a source is “auto discovered”, validate the following information:

- ▶ Primary Keys are included or are not included.
- ▶ Indexes are or are not included.
- ▶ Relationships are or are not included.
- ▶ Type mapping is correct.

### ***Index specifications***

When you create a nickname for a table in a data source, information about any indexes on the table is added to the global catalog. The query optimizer uses this information to expedite the processing of distributed requests. The catalog information about a data source index is a set of metadata, and it is called an index specification. The query optimizer uses this information to expedite the processing of distributed requests.

A federated server does not create an index specification when you create a nickname for these objects:

- ▶ A table that has no indexes
- ▶ A view, which typically does not have any index information stored in the remote catalog
- ▶ A data source object that does not have a remote catalog from which the federated server can obtain the index information

If a table acquires a new index, in addition to the indexes that it had when the nickname was created, the federated server is unaware of the new index. Similarly, when a nickname is created for a view, the federated server is unaware of the underlying table (and its indexes) from which the view was generated. In these circumstances, you can supply the necessary index information to the global catalog. You can create an index specification for tables that have no indexes. The index specification tells the query optimizer on which column or columns in the table to search to find data quickly.

### ***Database statistics***

Database statistics are visible to Federation Server and not to the Optim solution. Federation Server translates the DB2 SQL into SQL for the source database and, on the basis of database statistics, might reorder (optimize) the SQL statement. For this reason, run database statistics to ensure Federation Server optimization. Because Optim cannot see the statistics, it issues a warning. It is important to validate that the Optim solution selects the correct strategy (key lookup or table scan) and to consider overriding Optim when the strategy is not optimized.

### **Creating a DB alias to Federation Server**

Create an Optim DB alias to Federation Server to provide a communication channel to the source data. Creating at least one DBALIAS per federated data source provides you traceability (the DB alias maps to a data source, such as Teradata or DB2 for System i). If you want flexibility to relocate the processing of multiple federated data sources, dedicate DB aliases to specific data sources (for example, Teradata application x).

**Multi-key delete:** Federation Server converts the DB2 array delete into a series of SQL delete statements. Optim support can provide you with a switch (ZAP) to use multi-key delete instead of array delete.

### 6.3.3 Using Oracle

In this configuration, the Optim solution optimizes the requests to the Oracle engine. The Oracle optimizer is responsible for converting the SQL to the SQL for the database and identifying the best access strategy. The wrapper module invokes the appropriate database client API.

Optim Connect, when configured using Oracle federation (see Figure 6-24), has two possible configurations:

- ▶ Oracle 10g and HSODBC
- ▶ Oracle 11g+ and DG4ODBC

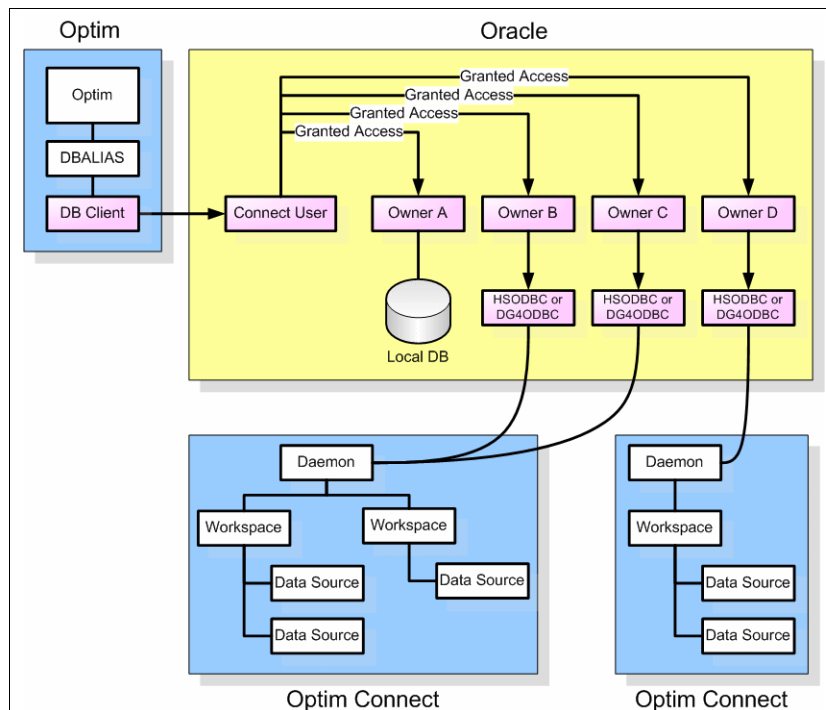


Figure 6-24 Optim Connect through Oracle federation

From a high-level point of view, the two configurations are similar; however, in Oracle 11g and DG4ODBC, you have to configure additional components. We point out these differences during the process.

**Oracle:** Use the information that is described in this chapter to facilitate your Oracle configuration. Your primary source of information is the Oracle documentation and technical bulletins. Databases are mission-critical systems, and you must identify and follow your internal best practices and procedures.

For both HSODBC and DG4ODBC (see Figure 6-25), you must configure `listener.ora`, `tnsnames.ora`, and add a new configuration file (`init{SID}.ora`).

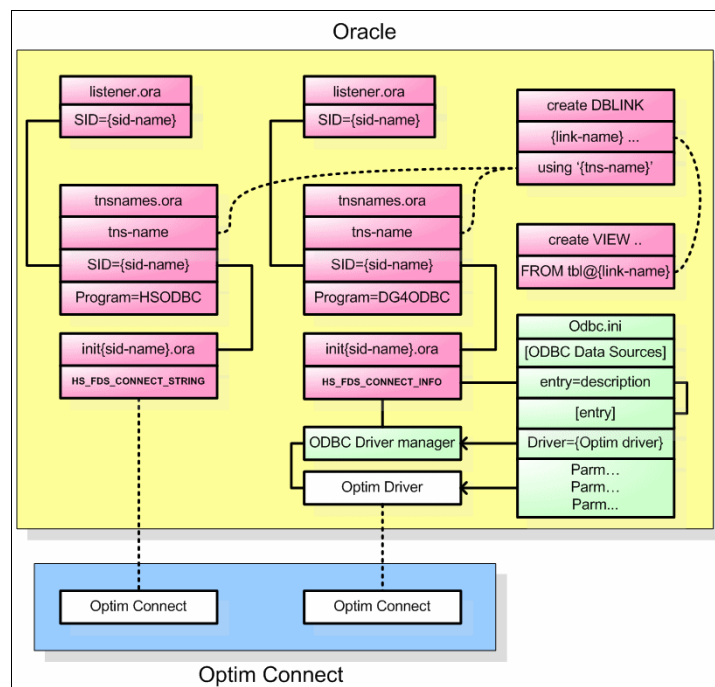


Figure 6-25 Oracle configuration map

In the case of 11g and DB4ODBC, you also have to configure `odbc.ini` for the ODBC driver manager (a prerequisite). As common steps, you also create a DBLINK and a few VIEWS.

You have to choose four configuration names:

► *{sid-name}*

To uniquely identify your new federated connection information. The name has no association with the real data source; however, consider selecting a name that can facilitate traceability. In our example, the name is OPTIMFLAT.

► *{tns-name}*

To uniquely identify this gateway configuration to Oracle. In our example, the name is OPTIMFLATWIN. This name is used when creating the DBLINK.

► *{dblink-name}*

To uniquely identify your data source to the DB users within an Oracle instance. In our example, the name is OPTIMCONNFLAT.

► *{odbc-name}*

To uniquely identify an Optim Connect configuration in one instance of `odbc.ini` (applies only for DG4ODBC). In our example, the name is OPTIMODBCFLAT.

**Important:** You must make copies of all configuration files before any changes are made so that any unexpected errors can be easily reversed. For example, if the syntax in the `listener.ora` file is not correct, the service will fail when it is refreshed.

### ***Configuring the listener.ora file***

Open the configuration file `$ORACLE_HOME/network/admin/listener.ora` and add the SID entry for the Optim Connect source:

```
(SID_DESC=
  (SID_NAME= {sid-name})
  (ORACLE_HOME= {oracle-home-dir}
  (PROGRAM=hsodbc|dg4odbc)
  ....
)
```

In this file, *{sid-name}* is the name that you chose at the beginning of this process (we chose FLATDATA). The parameter *{oracle-home-dir}* is the content of the system variable `$ORACLE_HOME`.

Locate the following statement for the current Oracle instance:

```
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      ...
      (ADDRESS = (PROTOCOL = TCP)
```

```

        (HOST = {oracle-listener-address})
        (PORT = {oracle-listener-port})
    )
)

```

Write down the value of *{oracle-listener-address}* and *{oracle-listener-port}*, and then save the configuration file.

For HSODBC, the final configuration looks like this example:

```

(SID_DESC=
  (SID_NAME= OPTIMFLAT)
  (ORACLE_HOME= /users/app/oracle/product/ora10g)
  (PROGRAM=hsodbc)
  ....
)

```

For DG4ODBC, the final configuration looks like this example:

```

(SID_DESC=
  (SID_NAME= OPTIMFLAT)
  (ORACLE_HOME= /users/app/oracle/product/ora11g)
  (PROGRAM=dg4odbc)
  (ENVS="ODBCINI=/opt/users/YourDriverManager/odbc.ini" )
  ....
)

```

### ***Configuring tnsnames.ora file***

Open the configuration file *\$ORACLE\_HOME/network/admin/tnsnames.ora* file, and add the TNS entry for the Optim Connect source:

```

{tns-name} = (DESCRIPTION=
  (ADDRESS=
    (PROTOCOL=tcp)
    (HOST={oracle-listener-address} )
    (PORT={oracle-listener-port} )
  )
  (CONNECT_DATA=
    (SID={sid-name})
  )
  (HS=OK)
)

```

In this example, *{oracle-listener-address}*, *{oracle-listener-port}*, and *{sid-name}* are the values that are used in the *listener.ora* file.

The final configuration looks like this example:

```

OPTIMFLATWIN = (DESCRIPTION=

```

```

        (ADDRESS=
          (PROTOCOL=tcp)
          (HOST=192.168.30.111)
          (PORT=1521)
        )
        (CONNECT_DATA=
          (SID=OPTIMFLAT)
        )
        (HS=OK)
      )

```

Save the configuration file.

### ***Configuring the gateway initialization parameter file***

You need create an Oracle gateway configuration file in \$ORACLE\_HOME/hs/admin for each federated source. The name must adhere to the standard:

init{sid-name}.ora

In our case, the file name is:

initOPTIMFLAT.ora

Enter the required configuration parameters:

```

HS_FDS_TRACE_LEVEL={trace-level}
HS_FDS_SHAREABLE_NAME={driver-path}
HS_FDS_CONNECT_INFO={dns-name}
HS_FDS_CONNECT_STRING="{XML-connect-string}"

```

The following list describes the configuration parameters:

- ▶ HS\_FDS\_TRACE\_LEVEL={trace-level}
 

Must be set to OFF. In the case of connection errors, you can change the value to ON (basic tracing) or DEBUG (detailed tracing). The trace is written in the LOG directory where the gateway is installed.
- ▶ HS\_FDS\_SHAREABLE\_NAME={driver-path}
 

For HSODBC, this value must be set to DUMMY.  
For DG4ODBC, this value must be set to the full path to the ODBC driver manager.
- ▶ HS\_FDS\_CONNECT\_INFO={dns-name}
 

For HSODBC, this value must be set to REMOTE.  
For DG4ODBC, this value must be set to the odbc.ini entry that is configured to access your Optim Connect.
- ▶ HS\_FDS\_CONNECT\_STRING={connect-string}

For HSODBC, this value must be set to the Optim Connect XML string enclosed between a pair of double quotation marks (“”). Reference 6.3.5, “Optim Connect XML binding parameters” on page 256 for the appropriate procedure.

For HSODBC, our `initOPTIMFLAT.ora` looks like this example:

```
HS_FDS_TRACE_LEVEL=OFF
HS_FDS_SHAREABLE_NAME=DUMMY
HS_FDS_CONNECT_INFO=REMOTE
HS_FDS_CONNECT_STRING="<NAVOBJ> ... </NAVOBJ>"
```

For DG4ODBC, our `initOPTIMFLAT.ora` looks like this example:

```
HS_FDS_TRACE_LEVEL=OFF
HS_FDS_SHAREABLE_NAME=/opt/odbc/lib/myDriverManager/odbc.so
HS_FDS_CONNECT_INFO=OPTIMODBCFLAT
```

Note that the `HS_FDS_CONNECT_STRING` is not included.

Save the file.

### ***Configuring odbc.ini for DG4ODBC***

Open the file `odbc.ini`, which is located in `$ODBCHOME` and enter the generic parameters that are required by the ODBC Driver Manager:

```
[ODBC Data Sources]
{Optim-Connect-entry}=Optim Connect Server Driver for ODBC 3.5

[{Optim-Connect-entry}]
Driver={Optim-Connect-driver}
Description={description}
.... Optim Connect Parameters ...
```

The following descriptions refer to the previous example:

- ▶ *Optim-Connect-entry* is the value of `HS_FDS_SHAREABLE_NAME` that is used in the DG4ODBC configuration.
- ▶ *Optim-Connect-driver* is the name of the Optim Connect ODBC driver. You must specify the full path.
- ▶ *Description* is a mnemonic value to help with traceability.

```
[ODBC Data Sources]
OPTIMODBCFLAT=Optim Connect Driver for ODBC 3.5

[OPTIMODBCFLAT]
Driver=/dsk/users/optim/connect/lib/navcli.so
Description=Optim Connect ODBC access to W_FLAT.Customer
.... Optim Connect Parameters ...
```

We discuss additional (and required) Optim Connect parameters in 6.3.4, “Configuring Optim Connect ODBC parameters” on page 252.

### ***Recycling the Oracle listener***

After making changes to `listener.ora`, it is necessary to refresh the previous values that are cached. Use the following command:

```
LSNRCTL RELOAD
```

Stopping and starting the listener also refreshes the cache; however, it is disruptive to any other processes that use this service. The following commands stop and start the Oracle listener:

```
LSNRCTL STOP  
LSNRCTL START
```

### ***Creating the database link***

Establish an SQL\*PLUS command-line session, connecting to the database as SYSTEM.

Create a database link using the *{tns-name}* entry name that you used in the `tnsnames.ora` file.

The command must be typed with no line breaks:

```
CREATE PUBLIC DATABASE LINK "{dblink-name}"  
    CONNECT TO "{optimconnect-user-name}"  
    IDENTIFIED BY "{optimconnect-password}"  
    USING '{tns-name}';
```

The following descriptions refer to the previous command:

- ▶ *{dblink-name}* must be a mnemonic value. It does not need to be any of the prior values that have been used in the configuration.
- ▶ *{optimconnect-user-name}* must be one of the authorized users that you have defined for the workspace (`machines::daemons::IRPCD::workspace`).
- ▶ *{optimconnect-user-password}* must be one of the authorized user passwords that you have defined for the workspace (`machines::daemons::IRPCD::workspace`).

Consider the following example:

```
CREATE PUBLIC DATABASE LINK "OPTIMCONNFLAT"  
    CONNECT TO "FLATUSER"  
    IDENTIFIED BY "FLATPWD"  
    USING 'OPTIMFLATWIN';
```

Then, you can select from a table with the following query:

```
SELECT * FROM customers@OPTIMCONNFLAT;
```

### ***Creating database views***

Optim cannot access the DBLINK data directly. Therefore, you must create views for the tables that are required in your project, for example:

```
CREATE VIEW customers
  AS SELECT custid, custname, maxaddr, maxaccount, address, account,
         rfa, xml
  FROM customers@optimconnflat;
```

You also have to identify the primary keys to Optim, for example:

```
ALTER VIEW customers
  ADD CONSTRAINT pk1 PRIMARY KEY (custid) RELY DISABLE NOVALIDATE;
```

Optionally, you can add virtual indexes, if they are available, to map real indexes on the source database. Unfortunately, in this case, we access a flat file, so no index exists. Assume that this file is an Adabas file, in which case, the Oracle command is:

```
CREATE INDEX vix_custid ON emp(emp_id) NOSEGMENT;
ALTER SESSION SET "_use_nosegment_indexes" = true;
```

This command helps the Oracle optimizer somewhat during the process of rewriting the SQL for the remote data source.

## **6.3.4 Configuring Optim Connect ODBC parameters**

You have already set up the generic parameters that are required by the ODBC Driver Manager:

```
[ODBC Data Sources]
{Optim-Connect-entry}=Optim Connect Driver for ODBC 3.5

[{Optim-Connect-entry}]
Driver={Optim-Connect-driver}
Description={description}
.... Optim Connect Parameters ...
```

The `odbc.ini` configuration continues with Optim-specific parameters. For reference, read “ODBC Client Interface Part XI” of *IBM Optim Connect User Guide and Reference*. The following list explains the parameters:

- `Binding=name|XML_format`  
Specifies the data source connection information:
  - `name`

The name of the binding settings in the local repository. This name provides access to all data sources that are defined in this binding configuration.

- XML format

The binding settings in XML format (see the “Optim Connect XML parameters for ODBC” chapter in the *IBM Optim Connect User Guide and Reference* manual). This version of the parameter defines specific data sources either locally or on a remote machine and eliminates the need to define local binding settings in the repository. Only the data sources that are specified for the binding are accessed. If you want to access the data sources in all the binding settings on a remote machine, use the BindURL parameter.

- BindURL=[connectprefix://] [username:password@]host[:port] [/workspace] [ | ...]

Specifies an Optim Connect Server to which to connect and whose data sources, which are defined in the binding settings on this server, are available. This parameter eliminates the need to define a local binding with entries for data sources on a server. If you want to access only a subset of the data sources on the server, use the XML Binding parameter. BindURL uses these parameters:

- connectprefix

An optional prefix to make the URL unique when the context is ambiguous.

- username:password@

An optional user ID and password pair for accessing the Optim Connect server.

- host

The TCP/IP host where the daemon resides. Both the numeric form and the symbolic form are accepted.

:port

The daemon port number.

workspace

The workspace name. The default name is Navigator.

**BindURLs:** You can specify multiple BindURLs, using an OR symbol (|) as a separator. Spaces between the BindURLs are not allowed. The connect string succeeds as long as one of the machines listed is accessible. Optim does not currently support high-availability hot-swap, but you can implement data access cold-swap by using multiple BindURLs. In this mode, if a new query connection is established after the first BindURL (your primary server) goes offline, the second BindURL (your standby server) can provide data access services.

You can apply this concept to open data manager configurations.

- A data source name can appear multiple times (for example, in the local and remote bindings). Optim Connect resolves this ambiguity by using the first definition of any default subsystem name (DSN) and disregarding any subsequent definitions. Thus, if a DSN called SALES appears in the local binding and uses the BindURL parameter, the local definition is used.
- When using BindURL, Optim Connect binds upon initialization to all of the DSNs that are defined for the binding, regardless of which DSNs are actually used (this function can result in decreased performance).
- For each server that is specified in the BindURL connect string item, Optim Connect automatically adds a remote machine (dynamically in memory) called BindURL $n$  with  $n=1,2,\dots$ , according to the order of the elements in the BindURL value.
- For multiple BindURLs, use the following syntax to specify a remote query processor to use:

BindURL $n$ =[attconnect://].... where  $n$  is the number of the BindURL specifying the remote machine whose query processor you want to use.

► Database=databasename

The name of a virtual database that this connection accesses. (The virtual database presents a limited view to the user of the available data, such as only selected tables). For more information, see “Using a Virtual Database” in *IBM Optim Connect User Guide and Reference*.

► DefTdpName=datasource

The name of the single data source that you want to access as the default using this connection. The tables, which are specified in SQL statements, are assumed to be from this data source. If this parameter is not specified, SYS is the default. For tables from any other data source, you must prefix each table name with the name of the data source, using the format: *data source:tablename*. Optim Connect opens the connection in single data source mode (unless explicitly overridden by setting OneTdpMode=0).

► DSNPasswords=source|machine=user/password[&source|machine=user/password [&...]]

User profile information (*username* and *password*) granting access to a data source or remote machine using this connection. As an alternative to storing *usernames* and *passwords* in the user profile, this parameter allows you to dynamically supply one or more pairs of *username* and *password* values, with each pair assigned to a particular data source or remote machine:

- *source*: A data source name that is defined in the binding configuration
- *machine*: A machine alias that is defined in the binding configuration
- *user/password*: A pair of user ID and password values needed to access the indicated data source

► LocalQp=1|0

Specifies that the ODBC client works in local query processor (QP) mode (1) and not remote QP (0, which is the default).

► OneTdpMode=1|0

Specifies whether you are working in single (1) or multiple (0) data source mode. You explicitly must set a value for OneTdpMode, as well as set a value for DefTdpName, for Optim Connect to work in single data source mode. Otherwise, the connection is opened to allow access to multiple data sources.

► Passthru=0

Do not enable (by setting to 1) SQL Passthru, because it creates unpredictable results in your Optim project.

► PWD=password

Specifies the password that is required to access the user profile. For details, see “Managing a User Profile” in *Optim Connect User Guide and Reference*.

► QpTdpName=machine

Specifies the remote machine where query processing will take place. The name of this remote machine is defined in the binding configuration.

- UID=userID

Specifies the name of a user profile in the repository. If the user profile is not specified, the default user profile (NAV) is used.

### 6.3.5 Optim Connect XML binding parameters

The {XML-connect-string} must adhere to following template:

```
<NAVOBJ>
  <BINDING>
    <DATASOURCES>
      <DATASOURCE type='REMOTE'
        connect='REMOTE_MACHINE'
        name='{source-name}'
      />
    </DATASOURCES>
    <REMOTEMACHINES>
      <REMOTEMACHINE name='REMOTE_MACHINE'
        workspace='{workspace-name}'
        address='{daemon-address}'
        port='{daemon-port}'
      />
    </REMOTEMACHINES>
  </BINDING>
</NAVOBJ>"
```

The following explanations refer to the preceding template:

- {source-name} represents the name that you specified in Optim Connect Studio in machines::bindings::data source, for example, Accounting, Customer.
- {workspace-name} represents the workspace that you specified in Optim Connect Studio in machines::daemons::IRPCD, for example, ONLY\_FLAT.
- {daemon-address} represents the server address, for example, 192.168.10.173, localhost, or \\myserver.
- {daemon-port} represent the port that is used where Optim Connect listens for requests, for example, the default port 2551.

You can locate the binding syntax in Chapter 3 of *IBM Optim Connect User Guide and Reference*.

## 6.4 Non-relational data sources

Access to non-relational data sources, such as IMS, VSAM, Adabas, CA-IDMS, and other data sources, with Optim requires adding either InfoSphere Classic Federation Server for z/OS or Optim Connect to the Federated Database stack.

This discussion about the specific implementation of non-relational data sources requires knowledge of the underlying data sources and it is not in the scope of this book.

There are only few valid software stacks:

- Federation Server to Optim Connect
- Federation Server to Classic Federation Server
- Oracle federation to Optim Connect

The choice between Optim Connect for z/OS and Classic Federation Server is not simple. Both products are comprehensive integration platforms for the on-demand access and integration of the enterprise data sources and the existing applications. Technically speaking, both middleware products solve the same problem. There are slight implementation differences when mapping non-relational data sources as well as separate restrictions. Therefore, to decide which stack to use, ask three technical questions:

- What data source do I need to support?

If there is only one middleware product that supports the data source, the decision is easy. Figure 6-26 represents a high-level matrix of the data sources that are supported by Optim middleware.

	Optim Connect	Classic Federation
Flat files on LUW	■	
SEQ	■	■
IMS	■	■
ADABAS	■	■
IDMS		■
DATA COM		■

Figure 6-26 Middleware data source support matrix

- What are the mapping mechanism and restrictions?

The most common divider is the COBOL structure, such as OCCURS *n* TIMES or OCCURS DEPENDING ON, which is located in the record (beginning, middle, or end) and how it is mapped to relational data sources (subtable or multiple columns).

- What are the specific data source version and features that are supported by the middleware?

The most common dividers are CICS® support, FastPath, and so on.

When the choice is still unclear, determine which federation to select.

## Non-relational data sources using Federation Server

Optim can gain access to non-relational data sources through Federation Server (Figure 6-27) using one of these products:

- Optim Connect
- Classic Federation Server

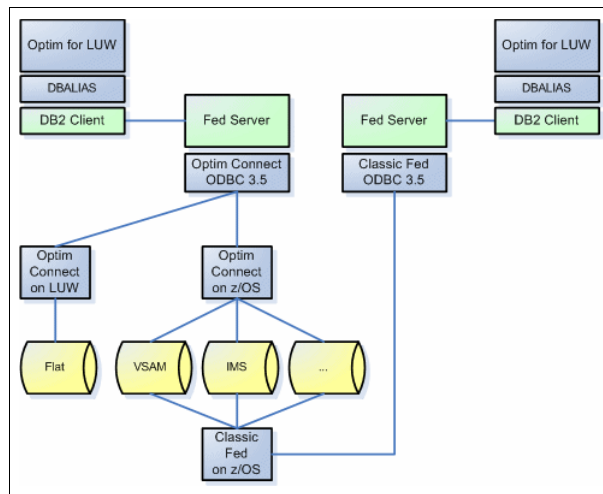


Figure 6-27 Non-relational data sources through Federation Server

## Non-relational data sources using Oracle federation

Optim can also gain access to non-relational data sources through the Oracle Federation Server (Figure 6-28 on page 259) using Optim Connect.

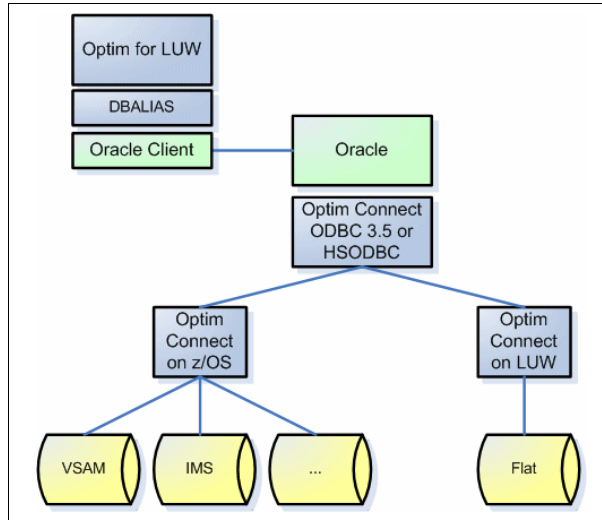


Figure 6-28 Non-relational data sources through Oracle federation

## Classic Federation Server

The installation and configuration of Classic Federation and Federation Server require a thorough understanding of these components and how they are used with an Optim solution. You can obtain detailed installation and configuration information for Classic Federation and Federation Server in the *Installing and Configuring your Optim Solution for z/OS with Classic Federation* manual.

## Optim Connect

Optim Connect is a comprehensive integration platform for the on-demand access and integration of enterprise data sources and existing applications. Optim Connect runs on many platforms, such as Microsoft Windows, UNIX, and z/OS, and provides middleware services. You can obtain detailed documentation in the *IBM Optim Connect User Guide and Reference* manual.

## 6.5 IBM InfoSphere Classic Federation Server

The Classic Data Architect is used to create relational tables and views that map to data sources in supported non-relational database management systems. With IBM InfoSphere Classic Federation Server for z/OS, client applications can issue SQL queries against these tables to access data in the non-relational databases. The client applications can also issue INSERT, DELETE, and UPDATE requests against the tables to modify the data in the non-relational databases.

IBM InfoSphere Classic Federation Server for z/OS is a complete, high-powered solution that provides SQL access to mainframe databases and files without mainframe programming.

Using the key product features, you can perform these functions:

- ▶ Read from and write to mainframe data sources using SQL.
- ▶ Map logical relational table structures to existing physical mainframe databases and files. Use the Classic Data Architect graphical user interface (GUI) to issue standard SQL commands to the logical tables.
- ▶ Use standards-based access with ODBC or Java Database Connectivity (JDBC) interfaces.
- ▶ Take advantage of multi-threading with native drivers for scalable performance.

The architecture of InfoSphere Classic Federation Server for z/OS consists of the following major components:

- ▶ Data server

Data servers perform all data access. The architecture of the data server is service-based. The data server consists of several components, or services. A major service that is embedded in the data server is the query processor that acts as the relational engine for InfoSphere Classic Federation. These services are defined by adding a service definition to the data server configuration. A *service definition* contains a number of configuration parameters in which you set values to define a service and the behavior of that service. Data servers perform the following functions:

- Accept SQL queries from clients
- Determine the type of data to access
- Transform SQL queries into the native file or database access language
- Optimize queries
- Create relational result sets from native database records
- Process post-query result sets as needed

A data server accepts connection requests from client applications. Client applications can access a data server using ODBC or the JDBC client that InfoSphere Classic Federation Server for z/OS provides. The major service that runs within the data server is the region controller. In addition, the following subservices run in the data server:

- Connection handlers
- Query processors
- Initialization services
- System exits

► Data connectors

The query processor dynamically loads one or more data connectors to access the target database or file system that is referenced in an SQL request. Optim supports these connectors:

- Adabas provides access to Adabas files.
- CA-Datcom provides access to CA-Datcom files.
- CA-IDMS provides access to CA-IDMS files.
- IMS provides access to IMS data using the IMS database resource adapter (DRA) interface or the IMS bean-managed persistence (BMP)/database management batch (DBB) interface.
- Sequential provides access to sequential files or members.
- VSAM provides access to native VSAM files, VSAM files under the control of CICS, and VSAM files under the control of DFSMSvts.

► Classic Data Architect

The purpose of the Classic Data Architect is to administer the logical table definitions, views, and SQL security information that are stored in the metadata catalog. You must map the data definitions to logical tables, so that they can be processed through SQL requests. The Classic Data Architect tool makes it easier for you to perform the following tasks:

- Define tables, columns, primary keys, indexes, stored procedures, and views.
- Specify user authorization for all objects.
- Import existing physical definitions from copybooks, CA-IDMS schemas, and IMS database descriptors (DBDs).
- Generate Data Definition Language (DDL) for the objects that you create that can be run directly on a server or saved to a script file.
- Generate DDL script from objects that are already defined in the catalog and export DDL scripts to a data set on the server for use with the **metadata** utility.
- Connect directly to a Federated Classic data source and view the objects in the system catalog.

► Metadata catalog

The information that you generate from the Classic Data Architect is stored in metadata catalogs. A *metadata catalog* is a set of relational tables that contain information about how to convert data from non-relational to relational formats. The data server accesses the information that is stored in these catalogs. Metadata catalogs emulate relational database catalogs and define business-oriented relational mappings. You can use the catalog initialization

and maintenance utility to create or perform operations on a metadata catalog. In addition, the **metadata** utility accepts DDL that is generated from the Classic Data Architect. The **metadata** utility must connect successfully to a data server before running any DDL statements. The utility updates the system catalogs with the contents of the DDL statements.

► DB client

InfoSphere Classic Federation Server for z/OS provides the ODBC and JDBC clients. The clients enable client applications or tools to submit SQL queries to the data server. InfoSphere Classic Federation Server for z/OS provides the ODBC and JDBC clients. The ODBC client is based on the ODBC 3.5 standard and connects to Federation Server on Linux, UNIX, or Windows. The JDBC client is based a JDBC 3.0 driver and connects with the designer component of Optim.

**Note:** There are two types of query processors:

- The *Single-phase commit* query processor (CACQP) accesses and joins information from multiple data sources and performs updates to a single data source.
- The *Two-phase commit* query processor (CACQPRRS) accesses and joins information from multiple data sources and performs updates to multiple data sources. The two-phase commit query processor uses z/OS Resource Recovery Services to coordinate the data source updates. The two-phase commit query processor supports the CA-Datcom, DB2 for z/OS, IMS, and VSAM data sources.

Generally, you cannot mix these two types of query processors within a single data server. The data server configuration file must include a service definition for a query processor. The query processor invokes one or more connectors to access the target database or file system that is referenced in an SQL request. The query processor supports SQL processing, two-phase commit processing, and calls to stored procedures. By using native database and file facilities, the query processor maintains the structural integrity and the performance characteristics of the data source.

For information about system requirements for InfoSphere Classic Federation Server, see this website:

[http://publib.boulder.ibm.com/infocenter/iisc1zos/v9r1/index.jsp?topic=/com.ibm.websphere.ii.federation.classic.overview.doc/prod\\_overview/iifyfcstocfed.html](http://publib.boulder.ibm.com/infocenter/iisc1zos/v9r1/index.jsp?topic=/com.ibm.websphere.ii.federation.classic.overview.doc/prod_overview/iifyfcstocfed.html)

For information about system requirements for InfoSphere Classic Federation Server V9.5, see this website:

<http://www.ibm.com/software/data/infosphere/classic-federation-server-z/requirements.html>

For information about supported data sources, see this website:

<https://www-304.ibm.com/support/docview.wss?uid=swg27011950>

For information about the installation of Classic Data Architect, see this website:

<http://publib.boulder.ibm.com/infocenter/iisc1zos/v9r1/index.jsp?topic=/com.ibm.websphere.ii.product.install.clas.doc/topics/iipicac-instcda.html>

## 6.5.1 Mapping non-relational sources to relational model

Before you begin, you must perform the following tasks on the data server where the query processor will run:

1. Install InfoSphere Classic Data Architect. *Do not* configure it yet.
2. Install InfoSphere Classic Federation Server on z/OS.
3. Create and initialize a metadata catalog.
4. Set up the configuration file.
5. Start the data server.
6. Configure and use InfoSphere Classic Data Architect.
7. Configure InfoSphere Classic Federation Server ODBC for Federation Server.

**Important:** It is essential that you install Classic Data Architect before you install IBM InfoSphere Classic Federation Server for z/OS. Failure to do so will result in the ODBC drivers for z/OS not being installed if you happen to use them in your scenario. For examples of configuring ODBC data sources on the z/OS platform, refer to *IBM WebSphere® Information Analyzer and Data Quality Assessment*, SG24-7508.

To map non-relational sources to the relational model, use Classic Data Architect. You need to know these terms before you start:

### ► Workspaces

When you first open Classic Data Architect, you create a workspace that will contain your work. This workspace is located on your local workstation, and you do not share it with others. All users who work in Classic Data Architect have their own workspaces and their own projects within those workspaces. Within a workspace, there are two types of objects that you must create: data design projects and physical data models.

► Data design projects

Within a project, you design the tables and views that you eventually will create in the metadata catalog on the data server. The type of project that you create in Classic Data Architect is called a *data design project*. You can create any number of data design projects, using them to organize your physical data models, or you can put all of your physical data models into one project.

► Physical data models

A *physical data model* is a collection of schemas that contain the tables that you create to map to the data in your data sources. Schemas can also contain views on those tables and stored procedures that you might want to use to perform operations on the result sets for queries. Instead of creating your tables directly in the metadata catalog on the data server and modifying them there, you create and modify your tables in the model and then promote them to a metadata catalog. For example, you might have one data server for a test environment and another data server for a production environment. In a model, you can create a table and then run the DDL to create the table in the test data server to test the table. If you need to modify the table, you can drop the table from the metadata catalog in the test data server, modify the table in Classic Data Architect, then re-create the table in the metadata catalog, and test it again. When you have a version of the table that you want to put into production, you can create the table in the production data server.

The configuration of the Classic Data Architect tool is minimal. You must create and configure the following items before you can map any tables:

1. Create a data design project:
  - Open the New Data Design Project wizard by selecting **File** → **New** → **Data Design Project** from the menu bar at the top of Classic Data Architect.
  - The data design project appears in the Data Project Explorer.
2. Create a physical data model with the database type “Classic Integration”:
  - Open the New Physical Data Model wizard by right-clicking the data design project folder and selecting **New** → **Physical Data Model**.
  - A new physical data model appears in the Data Models folder of your project. Close the **Diagram1** tab, because diagramming is a function in Eclipse but it is not a function in Classic Data Architect.
3. Create a connection to the InfoSphere Classic Federation data server running on the mainframe:
  - Open the New Connection wizard by right-clicking the **Connections** folder in the Database Explorer and selecting **New Connection**.

4. Set various global values that Classic Data Architect can use as default values in its wizards:
  - Open the Preferences window by selecting **Window → Preferences**.
  - Set these preferences in the Classic Data Architect pane, as needed:
    - Discovering Adabas files and mapping Adabas fields to columns in relational tables.
    - Accessing CICS VSAM.
    - Setting an FTP subcommand for double-byte character set (DBCS) data when you import Classic Data Architect References or export files that contain SQL statements.
    - Mapping PIC9(n) USAGE DISPLAY data to the character or decimal SQL data type.
    - Specifying the locale that the COBOL parser uses when validating COBOL copybooks on which you want to base tables. Select COBOL in the Preferences window, and then select the More COBOL options tab. Set the locale in the Compile time locale name field.

**Tip:** For more information about COBOL and PL/I import preferences that are supported by Eclipse, see the preference pages that are associated with the importer.

5. Create the logical relational table under the Data Design Project and Physical Data Model.
6. Define access privileges to the table. This process creates a DB2-like GRANT statement.
7. Generate the DDL. After the table definition is created, you must create and populate the DDL in the metadata catalog that is read by the InfoSphere Classic Federation data server. Also, you can generate the GRANT statements in this step.
8. Verify that the table mapping was successful by running the Sample Contents option under the table definition in the Database Explorer. You can visually validate that the extracted data meets your specifications.
9. After the table is created, you can create a Federation Server nickname against the non-relational source.

## 6.5.2 Import Classic references or run a Database Discovery

After these steps are done, you can start to import data definition files into a data design project.

For CA-Datcom, CICS VSAM, sequential, native VSAM, and IMS, you can import COBOL copybooks, PL/I include files, and DBDs. For CA-IDMS, you can import schema and subschema reports.

If you want to create tables that map to Adabas databases, you do not need to import files that describe the data structures to which you want to map. When you create an Adabas table, you provide information that tells Classic Data Architect how to locate the data structures in the Adabas database that you want to use.

If you want to create tables that map to CA-IDMS databases, you can either import schema and subschema reports into Classic Data Architect or you can tell Classic Data Architect how to locate the records and sets that you want to use in the CA-IDMS database.

When you import files into a project from a local or remote server (using FTP when accessing z/OS), the original file extension is replaced and the files are organized into subfolders for that project according to the file types:

- ▶ COBOL copybooks (extension .cpy) are placed into a subfolder that is called COBOL Copybooks.
- ▶ PL/I include files (extension .inc) are placed into a subfolder that is called PL/I Include Files.
- ▶ CA-IDMS schema (extension .sch) and subschema (extension .sub) reports are placed into a subfolder that is called CA-IDMS Schemas/Subschemas. You can also tell Classic Data Architect to discover subschema information directly from CA-IDMS.
- ▶ DBD files (extension .dbd) are placed into a subfolder called IMS DBDs.

### 6.5.3 Array structures

A *record array* is a group of data items in a database that have multiple occurrences within a single record in the database.

Typically, you map tables and views by importing data definitions in COBOL copybooks or PL/I include files. The copybooks or include files can contain array definitions. Classic Data Architect maps array definitions that you have not flattened into columns by converting the definitions to statements within BEGINLEVEL and ENDLEVEL blocks. The application then generates the Data Definition Language (DDL) that contains the BEGINLEVEL statements. You run the DDL on the data server to create a user table.

You can optimize performance when you query array data. If you want to insert, update, delete, or perform change capture on array data, you must flatten the structure.

## Fixed-length arrays

Fixed-length array constructs define an array in which the number of instances does not change. For example, an employee record can include the employee's dependent information (spouse and children). Because an employee can have multiple dependents, you can declare an array of dependent information within an employee record by specifying a COBOL OCCURS clause or a PL/I DIMENSION (DIM) attribute. The following example of a COBOL OCCURS clause defines a fixed-length array:

```
05 DEPENDENTS-ARRAY OCCURS 20 TIMES
```

The array appears 20 times in the source database record, regardless of how many dependents the employee has.

## NULL IS processing

The query processor on the data server skips null array instances as SQL ROW candidates at run time. In the example of the dependents array, if an employee has three dependents and the array occurs 20 times, 17 null instances of the array do not appear as a row in a result set.

You can include a NULL IS value in DDL to identify a given array instance as null, based on a comparison value. A comparison value can identify a null array instance in the following ways:

- ▶ The start of the first column in the array instance matches the comparison value (NULL IS null-value).
- ▶ Each character in the array instance matches a single-character comparison value (NULL IS ALL null-value).
- ▶ The start of a specified column in the array matches the comparison value (NULL IS column-name EQUAL null-value).

In the following example of NULL IS ALL DDL, the single character X is the comparison value. If each character in the array instance is X, that instance of the array is null:

```
MAXOCCURS 20 NULL IS ALL X
```

Null instances of a record array are not returned as a row in the result set unless ALL instances of the array are NULL. If all instances of the array are NULL, Classic Federation returns a single row for the non-array information in the record and sets the array data items to NULL. In the dependents array example, the employee has no dependents.

## Variable-length arrays

Another common record array construct defines variable-length data. The number of array instances depends on the value of a data item that precedes the array in the structure, such as NUMBER-OF-DEPENDENTS. In COBOL, the array declaration is:

```
05 NUMBER-OF-DEPENDENTS PIC 9(4) COMP.  
05 DEPENDENTS-ARRAY OCCURS 1 TO 20 TIMES DEPENDING ON NUMBER-OF-DEPENDENTS.
```

The next example shows a similar array declaration in PL/I that uses the REFER attribute to point to the variable containing the value for the number of dependents:

```
5 NUMBER_OF_DEPENDENTS BIN FIXED(15),  
5 DEPENDENTS_ARRAY DIM(N1 REFER(NUMBER_OF_DEPENDENTS)),
```

PL/I has no equivalent syntax for the COBOL DEPENDING ON clause, so the data server calculates the number of array instances automatically based upon array offset, array size, and record or segment length. The formula is:

$$\text{<Number-of-array-instances>} = (\text{<record-length>} - 1 - \text{<array-offset>}) / \text{<array-size>}$$

The number of instances does not appear in the DDL that is generated by the wizard. The number is calculated when the data server processes the DDL and validates the table information prior to creating the table definition in the catalog.

## Creating a separate table for each record array in a table definition

You can improve the performance of federated queries that read record array data if you map a separate table for each record array in the table definition.

Before you run the New Table wizard, import a data definition file into your data design project that describes the structure of the data in the source database, such as a COBOL copybook, a PL/I include file, or a CA-IDMS schema with subschemas. If your source is CA-IDMS or Adabas, you can connect to the server to retrieve the information directly from the source database.

Each table that you map separately consists of a single record array definition that contains the column definitions that are unique to a single array instance. Any given column appears in each instance of the array. You can map a separate table for each array by running the New Table wizard in Classic Data Architect once for each array, or you can copy and edit table objects as described in the following steps.

To map a separate table for each array in a base table by using the Data Project Explorer view, perform the following steps:

1. Run the New Table wizard once to create a parent table object using the array processing option **Create record array**.
2. Make one copy of the table object for each record array.
3. Delete columns in each copied table object until the columns that remain contain the key information and array data that you want.
4. Optional: Delete the columns that contain array definitions from the parent table if you want to use the non-array data for queries.

## 6.5.4 Creating views

To create a view on a table that already exists either only in your project or also in a metadata catalog, create an empty view in your project. Then, use the Properties view to create the SELECT statement.

### Project guidelines

The view must not reference more than one table. Use multiple DBALIAS to access heterogeneous data.

### Procedure

Use this procedure to create views:

1. In the Data Project Explorer, expand the physical data model in which you are working. Expand the database in which you are working. Right-click the schema in which you want to create the view and select **Add Classic Object** → **View**. In the Data Project Explorer, a view is created under the schema.
2. Name the view.
3. Select the view, and on the SQL page of the Properties view, type the SELECT statement.
4. On the Privileges page of the Properties view, grant privileges on the view.
5. Optional: Generate the DDL for the view. Right-click the view and select **Generate DDL** to open the Generate DDL wizard. With this wizard, you can generate the SQL DDL to define the view, and you can choose to run the DDL on a data server so that the view is created in the metadata catalog for that data server. You can also edit the generated DDL before you run it.

After you run the DDL, the view appears on the data server in the Database Explorer. To see the view, expand the data server and then expand **Schemas** → **the schema of the view** → **Views**.

If you want to generate and run the DDL for more than one object at a time, you can right-click a schema and select **Generate DDL**. The Generate DDL wizard will generate the DDL for all of the objects in the schema.

6. Optional: If you created the view on the data server, run a test query on the view:
  - a. In the Database Explorer, right-click the view and select **Data → Sample Contents**.
  - b. Look in the Data Output view to see the results of the test query.

### 6.5.5 Mapping tables and views for redefined data

Redefined data uses alternate record layouts for the same storage area, based on record types.

To read redefined data, define a table with an associated view for each type of record. To insert, update, or delete, define a separate table for each record type.

Each table that you map for redefined data must contain columns that identify common key information and a column for the type code field. These columns are followed by type-specific columns.

For example, a single VSAM file stores both employee and address records. The correct record interpretation is managed by associating the value of a record type field with a record layout. The COBOL definition in Example 6-1 shows how a REDEFINES clause specifies an alternate record type for ADDRESS-INFORMATION. If RECORD-TYPE = "A", Classic Data Architect uses the layout for address data.

#### *Example 6-1 COBOL definition*

---

```
01 EMPLOYEE-ADDRESS-RECORD.
   05 EMP-ID                PIC X(6).
   05 RECORD-TYPE           PIC X.
       88 RECORD-IS-EMPLOYEE VALUE 'E'.
       88 RECORD-IS-ADDRESS VALUE 'A'.
   05 EMPLOYEE-INFORMATION.
       10 LAST-NAME         PIC X(20).
       10 FIRST-NAME        PIC X(20).
       10 DATE-OF-BIRTH     PIC 9(8).
       10 MONTHLY-SALARY    PIC S9(5)V99 COMP-3.
       10 FILLER            PIC X(48).
   05 ADDRESS-INFORMATION REDEFINES EMPLOYEE-INFORMATION.
       10 ADDRESS-LINE-1    PIC X(30).
       10 ADDRESS-LINE-2    PIC X(30).
       10 ADDRESS-CITY      PIC X(20).
```

```
10 ADDRESS-STATE PIC XX.  
10 ADDRESS-ZIP   PIC 9(5).
```

---

Follow this procedure:

1. Map two tables, each with an associated view.  
For example, you define one base table and view for employee information, and another base table and view for address information.
2. Use the view or the table, depending on whether you query or update the data:
  - a. To query redefined data, supply filtering information in the view definition when you map the table. This approach simplifies queries in client applications, because the queries do not require WHERE clause filtering.
  - b. Supply a WHERE clause and use the type code value to filter the records.

**Important:** You cannot update a view.

To insert, update, or delete redefined data, your application must use the base table name and provide WHERE filtering.

## 6.5.6 Creating indexes

After you define a table, you can define indexes on the table to map existing indexes on the underlying data. An *index* identifies columns in a table that correspond to a physical index that is defined against a data source. Indexes are supported for Classic Federation, but not for Classic event publishing or Classic replication.

Follow this procedure to create an index:

1. In the Data Project Explorer, right-click the table on which you want to define an index and select **Add Classic Object** → **Index**. The New Index wizard opens.
2. Name the index, specify whether it will be unique, and select the columns on which to base the index.
3. If the index is for an IMS table, specify the method that Classic federation can use to select a program communication block (PCB) to access your IMS database.
4. If the index is for a CICS VSAM or native VSAM table, specify the directory service (DS) or data definition (DD) name for the index.

## 6.5.7 Generating DDL

When you finish designing your objects, you generate the DDL that you use to promote those objects to a metadata catalog on a data server.

When the DDL is generated, you can choose to run it on a data server. You can also choose to open the DDL in an editor.

If you do not choose to run the DDL immediately after it is generated, you can run it later by opening the SQL Scripts folder, right-clicking the file with the DDL, and selecting **Run SQL**.

### Before you begin

If you choose to run the DDL after it is generated, you must perform the following tasks first:

1. Open a connection to a data server.
2. Create a set of metadata catalogs on the data server.
3. Set up connectivity from the data server to your data sources.

### Procedure

Before you generate DDL, follow these steps:

1. Open the Generate DDL wizard in either of these ways:
  - Right-click the schema in which the objects are located and select **Generate DDL**. You can choose for which objects in the schema you want to generate DDL.
  - Right-click the object for which you want to generate DDL and select **Generate DDL**.
2. Follow the pages of the wizard to make these selections:
  - **Which DDL statements to generate**  
You can generate ALTER, COMMENT ON, CREATE, DROP, and GRANT statements. You can also choose whether to use fully qualified names and quoted identifiers.
  - **Which objects to generate DDL for**  
The available objects depend on which object you right-clicked to open the Generate DDL wizard.

- **Where to create the file, which statement terminator to use, whether to run the DDL on a data server, and whether to open the DDL file for editing**

The page on which you make these choices displays the DDL that will be generated.

3. After reviewing your settings, click **Finish**.

## 6.5.8 Granting Classic Federation Server privileges

Next, we describe granting privileges and privileges for performing actions on data servers. To grant a privilege, you must create the privilege in the Privileges page, generate the GRANT statement for that privilege, and then run the GRANT statement on a data server.

Follow this procedure to grant privileges:

1. Select the database in your data design project.
2. In the Properties view, click the **Privileges** tab. The table on the Privileges page lists the users who have one or more privileges.
3. Create a new privilege. Click the yellow (**New privilege**) icon in the Grant System or Database Privilege window, and specify Grantee, Type, and Privilege. For type, select either **SYSTEM** or one of the following types:
  - **\$CFI**: Allows grantees to create, drop, and view a metadata catalog on a data server.
  - **\$SP**: Allows grantees to create, drop, and view stored procedures on a data server.
  - **\$ADABAS**: Allows grantees to create, drop, and view Adabas tables and views on a data server.
  - **\$DATACOM**: Allows grantees to create, drop, and view CA-Datcom tables and views on a data server.
  - **\$IDMS**: Allows grantees to create, drop, and view CA-IDMS tables and views on a data server.
  - **\$IMS**: Allows grantees to create, drop, and view IMS tables and views on a data server.
  - **\$SEQUENT**: Allows grantees to create, drop, and view sequential tables and views on a data server.
  - **\$VSAM**: Allows grantees to create, drop, and view CICS VSAM and native VSAM tables and views on a data server.

- **SYSADM**: Grants all privileges on all objects that are in a metadata catalog. Users with this privilege can grant privileges to other users.
  - **SYSOPR**: Grants remote operator privileges to display reports and manage a data server.
  - **DISPLAY**: Grants remote operator privileges for displaying reports for a data server.
4. When you want to promote the privileges to a data server, right-click the database and select **Generate DDL**. In the Generate DDL wizard, follow these steps:
    - a. On the Options page of the wizard, clear all check boxes except **Fully qualified names**, **Quoted identifiers**, and **GRANT statements**.
    - b. On the Objects page, clear all of the check boxes.
    - c. On the Save and Run DDL page, specify the name of the SQL file that the wizard will create. Verify that the GRANT statements are correct. Select the **Run DDL on server** check box.
    - d. On the Select Connection page, select the connection to the data server, or create a new connection to a data server.
    - e. On the Summary page, verify the actions that the wizard will perform and click **Finish**.
    - f. In the Data Output view (which is by the Properties view by default), verify that the SQL statements ran successfully on the data server.

## 6.5.9 Accessing IMS data

IMS is a hierarchical database that consists of segments within a database. Mapping these segments within IBM InfoSphere Classic Federation Server for z/OS is driven by the IMS path from the root (uppermost) segment to the specific child or leaf segment. Understanding the data and mapping the segments correctly for privatization or archiving purposes is crucial for a successful deployment.

**IMS segment data compression constraint:** If IMS data is compressed using an IMS database compression exit, the compression is not seen by an application. This form of data compression is supported by Classic Federation. However, if IMS segment data is application-compressed at the segment level, Classic Federation does not support an exit point to compress or decompress the segment data. If your IMS segments are compressed, consider using OPTIM Connect to perform the task.

## Project guideline

The following guidelines apply to mapping tables for data growth archiving purposes:

- ▶ Classic Federation tables that are created from OCCURS clauses (see 6.5.3, “Array structures” on page 266), which were moved into their own table and “flattened out”, must have a parent and child relationship that is defined in the Optim access definition between the base table and the OCCURS table. An alternative method is to map multiple OCCURS clauses in a single table for INSERT, UPDATE, and DELETE operations.
- ▶ To map an IMS table, begin with a root segment and follow the path down to a specific child or leaf segment. The leaf segment is the data to be retrieved in the table mapping. In a multi-segment table (other than the root segment), map only the key fields in the parent segments to optimize Classic Federation access to the leaf segment data.
- ▶ All of the necessary segments for the test database must be mapped as separate tables.
- ▶ When working with tables from OCCURS clauses that were moved into their own table and “flattened out”, restoring the data requires an INSERT or UPDATE operation. Key values are inserted into the parent segment for the first row in the child segment. Every subsequent row will fail on a duplicate key error if the INSERT operation is used.
- ▶ OCCURS clauses and Redefines also must be mapped as separate tables.
- ▶ OCCURS tables must map all occurrences to restore the data to the IMS database.
- ▶ For OCCURS tables in a root segment table, the DELETE function deletes the full record because the table includes the key of the record.
- ▶ When a parent segment (root or non-root) is deleted in IMS during the deletion phase of archiving, all descendants of all segment types are also deleted. This approach is also used in a relational database if there are referential integrity relationships with a cascade delete rule.

Few relational models enforce referential integrity with cascade delete; however, this approach is normal for IMS data. Optim allows you to select which Optim tables (segment types) are deleted, although you must perform this selection with considerable care and planning.

## Before you begin

You must have performed the following steps *before* you start defining IMS data sources:

- ▶ Configure the data server where you plan to run the correlation service that will process change data from your IMS database.
- ▶ Create a metadata catalog.
- ▶ Decide which segment you want to map and the required path for navigating to the segment from a physical root or index.
- ▶ Configure a connection between the data server and your IMS database.
- ▶ Ensure that the IMS DBDs folder in your project has a database definition file (DBD) that lists the segments from which you want to select the fields to map to columns.
- ▶ Ensure that a COBOL copybook or (on the Microsoft Windows platform) PL/I include file for each IMS segment to which you want to map is in the appropriate folder in your data design project.

## Procedure

Use this setup procedure for accessing IMS data:

1. Map your IMS database to a relational table and optionally a view by using the New IMS Table wizard:
  - a. Open this wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object** → **IMS table**.
  - b. Select the DBD file on which you want to base the table.
  - c. Choose whether to use the table for queries, updates, or both.
  - d. Choose whether to create a view on the table.
  - e. Provide information about how to access the IMS database.
  - f. For each segment that is in the path, specify a COBOL copybook or include file, select the desired 01 level if there are more than one level, and then select the elements that you want to map as columns.

If you are creating a view, specify the criteria for the WHERE clause.

When you finish the wizard, the new table appears under the selected schema. If you created a view, the view also appears under the selected schema.

2. Optional: Modify the table properties or add privileges. Select the table and make any changes in the Properties view.
3. Optional: Create one or more indexes on the table.
4. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema:

- a. Right-click the table and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.
    - ii. Choose to generate DDL for tables. You can also choose to generate DDL for indexes.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine whether the DDL ran successfully.
    - v. Choose whether to open the DDL for editing.
5. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your IMS database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema in which you created the table. Expand the schema and expand the **Tables** folder.
  - b. Right-click the table and select **Data → Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.
6. Optional: If you have created a view, generate the DDL for the view. You can generate the DDL later. You can also generate the DDL for all of the objects within the same schema.
  - a. Right-click the view and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE and ALTER statements.
    - ii. Choose to generate DDL for views.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine whether the DDL ran successfully.
    - v. Choose whether you want to open the DDL for editing.
7. Optional: If you ran the DDL successfully on a data server, validate the view by running a test query against your IMS database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the **Views** folder.
  - b. Right-click the view and select **Data → Sample Contents**.

- c. Check the Data Output view to determine whether the test query ran successfully.

## 6.5.10 Accessing VSAM data

You can use IBM InfoSphere Classic Federation Server for z/OS to access VSAM files directly or use the Customer Information Control System (CICS) to access the files. For more information about CICS, see this website:

<http://www.ibm.com/software/http/cics>

### Project guidelines

The following guidelines apply to accessing VSAM data for data growth archiving purposes:

- ▶ VSAM files are relatively simple to work with compared to other data sources. Each VSAM file can be mapped as a logical table with INSERT, UPDATE, and DELETE (IUD) capabilities through CICS. Transactional VSAM Services are needed for IUD capabilities with native VSAM access.

**IUD:** Classic Federation currently does not support IUD on views, so you can create the view on the nickname in IBM InfoSphere Federation Server.

- ▶ You can only map to the following types of VSAM files:
  - KSDS, ESDS, and RRDS files
  - AM files
- ▶ If a VSAM file has multiple record types, each record type must be mapped as a View in Classic Federation.
- ▶ If the VSAM file has a secondary index, the index can be used only if a corresponding index definition is created in Classic Federation. Normally, a secondary index is not required for a parent table. A secondary index is required for child table, however, if the join column on the child table is not part of the primary index.
- ▶ When creating a physical VSAM file, the file must be initialized with at least one record or the first insert operation will fail.
- ▶ If a VSAM record contains an OCCURS clause, mapping those fields varies based on the Optim operation and the target database tables or files:
  - If the target is a relational table, the OCCUR fields can be mapped as a long character, or a separate normalized table with the same structure as the target table.

- If there is an OCCUR field, it must be expanded if a restore is required; otherwise, it can be mapped as an array table. Deletion of an array table, however, might result in multiple warning messages.
- ▶ Classic Federation supports VSAM file I/O compression and decompression routines. For further information, see this website:

<http://publib.boulder.ibm.com/infocenter/iisclzos/v9r5/index.jsp?topic=/com.ibm.swg.im.iis.fed.classic.exits.doc/topics/iifcsyxrpxit.html>

## Before you begin

You must have performed the following steps *before* you start defining VSAM data sources:

- ▶ Configure the data server where you plan to run the query processor that will accept requests from client applications.
- ▶ Create a metadata catalog.
- ▶ Decide which record elements you want to map in the CICS VSAM file and plan the indexes that you will need. To optimize performance, you must map the underlying data correctly. This task includes ensuring that any indexes, keys, or units of data that are defined in your file are defined to the data server when you map the data.
- ▶ Configure a connection between the data server and your CICS VSAM file.
- ▶ Ensure that a COBOL copybook or (on the Microsoft Windows platform) PL/I include file that references the database is in the appropriate folder in your data design project.

## Procedure

Use this procedure to define VSAM data source:

1. Map your CICS VSAM file to a relational table and optionally a view by using the New CICS VSAM Table wizard:
  - a. Open the wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object** → **CICS VSAM table**.
  - b. Select the copybook or include file on which to base the table.
  - c. Choose whether to use the table for queries, updates, or both.
  - d. Choose whether to create a view on the table.
  - e. Provide information about which CICS file control table to use and how to access the CICS VSAM file.
  - f. Select the elements that you want to map to columns in your relational table.

g. If you are creating a view, specify the criteria for the WHERE clause.

When you finish the wizard, the new table appears under the selected schema. If you created a view, the view also appears under the selected schema.

2. Optional: Select the table and, in the Properties view, modify any of its properties or add privileges.
3. Optional: Create one or more indexes on the table. See 6.5.6, “Creating indexes” on page 271.
4. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema. See 6.5.7, “Generating DDL” on page 272.
  - a. Right-click the table and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.
    - ii. Choose to generate DDL for tables. You can also choose to generate DDL for indexes.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine whether the DDL ran successfully.
    - v. Choose whether you want to open the DDL for editing.
5. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your CICS VSAM file. Be sure that the data server is connected to the system where the file is located.
  - a. In the Database Explorer, search your data server for the schema in which you created the table. Expand the schema and expand the Tables folder.
  - b. Right-click the table and select **Data → Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.
6. Optional: If you created a view, generate the DDL for the view. You can generate the DDL later. You can also generate the DDL for all of the objects within the same schema.
  - a. Right-click the view and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE and ALTER statements.

- ii. Choose to generate DDL for views.
  - iii. Name the file in which you want to save the DDL within your project.
  - iv. Choose whether to run the DDL on a data server. After running the DDL, check the Data Output view to see if the DDL ran successfully.
  - v. Choose whether you want to open the DDL for editing.
7. Optional: If you ran the DDL successfully on a data server, validate the view by running a test query against your CICS VSAM file. Be sure that the data server is connected to the system where the file is located.
- a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the Views folder.
  - b. Right-click the view and select **Data** → **Sample Contents**.
  - c. Check the Data Output view to see whether the test query ran successfully.

### 6.5.11 Accessing sequential data

To query or update data in a sequential file, you must create a relational table that maps to that file. You can also create a view on the table to filter record types or filter rows and columns. You use the Sequential Table wizard to create the table and, optionally, the view.

#### Project guidelines

The following guidelines apply to accessing the sequential file for data growth archiving purposes:

- ▶ The table definition can refer to the data set name. This method requires the data server to issue dynamic allocation requests before the file is opened physically. For Classic Federation to use dynamic allocation, the file must be cataloged.
- ▶ The table definition can reference the file by DD (statement) name. Accessing the file by DD name requires that the file is allocated statically and permanently to the server address space. The referenced DD statement must be added to the server JCL, and the DSN parameter on the DD statement identifies the physical file to be accessed.
- ▶ The recommended technique is to use dynamic allocation to access a sequential file. When a file is allocated dynamically, the file disposition is share mode, which allows other applications to access the file concurrently, if the applications are not attempting to access the file in exclusive mode.

## Before you begin

You must have performed the following steps *before* you start defining sequential file data sources:

- ▶ Configure the data server where you plan to run the query processor that will accept requests from client applications.
- ▶ Create a metadata catalog.
- ▶ Decide on the sequential files to which you want to map.
- ▶ Ensure that a COBOL copybook or (on the Microsoft Windows platform) PL/I include file that references the database is in the appropriate folder in your data design project.

## Project guidelines

The following guidelines apply to accessing the sequential file for data growth archiving purposes:

- ▶ Sequential data sets can be created, but they cannot be updated. Because sequential data sets do not have any native index definitions or keys, any request to access a sequential data set causes a table scan. You cannot use Data Architect to create indexes for tables that are mapped to sequential data sets.
- ▶ If you are mapping partitioned sequential data sets, a table must map to a single member within a partitioned data set.
- ▶ SQL access to extended partitioned data sets is not supported.
- ▶ When a table references a direct access data set, these data sets are referred to as Basic Direct Access Method (BDAM) data sets. BDAM data sets can be accessed using “keys” that consist of track addresses, block numbers, or a combination of the two. Classic Data Architect does not access a direct access data set using any of these techniques, but Classic Data Architect can retrieve the records sequentially that are stored in one of these direct access data sets.

## Procedure

Follow this setup procedure for accessing sequential file:

1. Map your sequential file to a relational table and optionally a view by using the Sequential Table wizard:
  - a. Open this wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object → Sequential table**.
  - b. Select the copybook or include file on which to base the table.
  - c. Choose whether to create a view on the table.

- d. Provide information about how to access the sequential file.
- e. Select the elements to map to columns in your relational table.
- f. If you are creating a view, specify the criteria for the WHERE clause.

When you finish the wizard, the new table appears under the selected schema. If you created a view, the view also appears under the selected schema.

- 2. Optional: Modify the table properties or add privileges. Select the table and make any changes in the Properties view.
- 3. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema:
  - a. Right-click the table and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.
    - ii. Choose to generate DDL for tables.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After running the DDL, check the Data Output view to determine whether the DDL ran successfully.
    - v. Choose whether to open the DDL for editing.
- 4. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your sequential file. Be sure that the data server is connected to the system where the sequential file is located:
  - a. In the Database Explorer, search your data server for the schema in which you created the table. Expand the schema and expand the Tables folder.
  - b. Right-click the table and select **Data → Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.
- 5. Optional: If you created a view, generate the DDL for the view. You can generate the DDL later. You can also generate the DDL for all of the objects within the same schema:
  - a. Right-click the view and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.
    - ii. Choose to generate DDL for views.
    - iii. Name the file in which you want to save the DDL within your project.

- iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to find out whether the DDL ran successfully.
  - v. Choose whether you want to open the DDL for editing.
6. Optional: If you ran the DDL successfully on a data server, validate the view by running a test query against your sequential file. Be sure that the data server is connected to the system where the sequential file is located:
- a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the **Views** folder.
  - b. Right-click the view and select **Data** → **Sample Contents**.
  - c. Check the Data Output view to find out whether the test query ran successfully.

## 6.5.12 Accessing SAG-Adabas data

The limitations and guidelines that are associated with accessing Adabas files for Optim processing fall into three categories:

- ▶ Limitations on mapping Adabas files
- ▶ Descriptors and the need for primary keys
- ▶ OCCURS handling and repeating fields

### Project guidelines

The following guidelines apply to accessing Adabas files for data growth archiving purposes:

- ▶ Each column in the table that you create must be associated with a field in the file, a superdescriptor, or a subdescriptor:
  - If Predict formatting is available, the following Predict formats are supported:
    - Character (A, AL, or AV)
    - Binary (B) with length of 2 or 4
    - Date (D, DS, or DT)
    - Floating point (F)
    - Integer (I)
    - Logical (L)
    - Numeric packed and unpacked (N, NS, P, PS, U, or US)
    - Time (T or TS)

- If only Adabas field formatting is available, the following formats are supported:
  - Alphanumeric (A)
  - Binary (B) with length of 2 or 4
  - Fixed point (F)
  - Floating point (G)
  - Packed decimal (P) and unpacked decimal (U)
- ▶ Because Adabas can support an undefined number of repeating fields, Classic Data Architect has a parameter to set the maximum number of occurrences of these fields. Make sure that this parameter is large enough to cover the maximum number of occurrences. This parameter affects the number of columns generated for the repeating fields in a mapped table. If this parameter is set too high, it will affect performance and might create memory issues.
- ▶ Descriptors and superdescriptors automatically generate Create Index DDL statements. After a nickname is created for a table in IBM InfoSphere Federation Server, a primary key is required for update purposes. If a primary key is not defined, updates will not be allowed by your Optim solution.
- ▶ Multiple Value fields (MU) and Periodic Groups (PE) have similar properties as OCCURS. For each repeating field in Adabas, there is an occurrence count field that determines the number of repeating fields. This field is mandatory if you want to map the corresponding repeating field.
- ▶ Adabas allows a user to have an unlimited number of occurrences of a field, with the logical maximum number of occurrences determined by your DBA.
- ▶ If the files are to be restored using SQL insert, the same columns cannot be mapped more than one time. This rule applies to superdescriptors and subdescriptors. If these fields are mapped more than one time, the updates will fail.
- ▶ Like other data sources, repeating fields can only be updated when they are “flattened” out. Array mappings cannot be updated.
- ▶ If the restore target is another Adabas file, the repeating fields can be mapped as individual fields. Classic Data Architect will append a number to each repeating field (for example, Col\_1, Col\_2, and so on). Because the target has a similar structure, it typically fits.
- ▶ If the restore target is a relational table, the repeating fields can be mapped as a normalized table or as multiple fields. Perform the mapping based on the matching structure in the target table.
- ▶ If the target is a relational table, the repeating fields can be mapped as a separate normalized table with the same structure as the target table.

- ▶ If there is an OCCURS construct, it must be expanded if a restore is required; otherwise, it can be mapped as an array table. The deletion of an array table, however, might result in multiple warning messages.

## Before you begin

You must have performed the following steps *before* you start defining Adabas file data sources:

- ▶ Configure the data server where you plan to run the query processor that will accept requests from client applications.
- ▶ Create a metadata catalog.
- ▶ Decide which data structures to map in your database and plan the indexes that you will need. To ensure optimal performance, you must map the underlying data correctly. This task includes ensuring that any indexes, keys, or units of data that are defined in your database are defined to the data server when you map the data. You can use almost any column that maps to a field definition table (FDT) or special descriptor table (SDT) definition for an index.
- ▶ Configure a connection between the data server and your Adabas database.
- ▶ If you want to use Predict, know the Adabas file number of the Predict dictionary and the name of the view to which you want to map. If you are not using Predict, know the number of the Adabas file to which you want to map.

## Procedure

Follow this setup procedure for accessing the Adabas file:

1. Optional: Use the Adabas page of the Preferences window to set these default values:
  - The name of the Predict dictionary that you want to use
  - The date format to which Classic Data Architect will convert dates
  - The time format to which Classic Data Architect will convert times
  - The maximum length for VARCHAR data
  - The maximum length for LVARCHAR data
  - The maximum number of occurrences for all fields that occur multiple times in an Adabas file
  - Whether to use the User Synonym field from the Predict Dictionary when this field is defined

2. Map your Adabas database to a relational table by using the New Adabas Table wizard:
  - a. Open this wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object** → **Adabas table**.
  - b. Select the model and schema in which you want to create the table.
  - c. Choose whether to create a view on the table.
  - d. Choose whether to connect to your Adabas database through an existing connection to a data server or whether you want to create a new connection to a data server. Either data server must be configured to access the Adabas database.
  - e. Specify the format of dates and times, the lengths of VARCHAR and LVARCHAR data types, and the maximum number of occurs. The default values that appear are either the global defaults that are set for the Adabas database or the defaults that are set in the Adabas page of the Preferences window.
  - f. Specify whether you plan to use the table (and view, if you are creating one) for queries, updates, or both.
  - g. Provide either the Predict or Adabas information that is necessary for the discovery process.
  - h. Select the Adabas fields that you want to map to columns in your relational table.
  - i. Optional: In the ISN name field, provide a name for the column that maps to the Adabas Internal Sequence Number (ISN). The default column name is ISN.
  - j. If you create a view, specify the criteria for the WHERE clause.
  - k. Modify the names of columns and provide null values.

When you finish the wizard, the new table appears under the selected schema. If you created a view, it also appears under the selected schema.
3. Optional: Modify the table properties or add privileges. Select the table, and make any changes in the Properties view.
4. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema.
  - a. Right-click the table and select Generate DDL.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.

- ii. Choose to generate DDL for tables.
  - iii. Name the file in which to save the DDL within your project.
  - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine if the DDL ran successfully.
  - v. Choose whether to open the DDL for editing.
- 5. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your Adabas database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema in which you created the table. Expand the schema and expand the **Tables** folder.
  - b. Right-click the table and select **Data** → **Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.
- 6. Optional: If you created a view, you can generate the DDL for the view now or later. You can also generate a DDL for all of the objects within the same schema.
  - a. Right-click the view and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE and ALTER statements.
    - ii. Choose to generate DDL for views.
    - iii. Name the file in which you want to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After running the DDL, check the Data Output view to find out whether the DDL ran successfully.
    - v. Choose whether you want to open the DDL for editing.
- 7. Optional: If you ran the DDL successfully on a data server, validate the view by running a test query against your Adabas database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the **Views** folder.
  - b. Right-click the view and select **Data** → **Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.

### 6.5.13 Accessing CA-IDMS data

CA-IDMS is a network database and is more complicated to map than other non-relational databases, such as IMS and Adabas. It is imperative to understand user requirements and the user's current record structure before any mapping is done.

In the CA-IDMS Table wizard, you can map a single record or a specific path to as many as 10 records. You define a path by starting with a single record and then navigating sets to additional records that are defined in the subschema. The subschema information that you use for mapping determines which records and sets are available. You can import the subschema information from a combination of CA-IDMS schema and subschema report files or directly from the CA-IDMS database by using Classic Data Architect's discovery process.

You produce CA-IDMS schema and subschema reports by running the CA-IDMS schema and subschema compilers and capturing the punched output into a z/OS data set. The JCL to punch these reports is in the SCACSAMP library with the member name CACIDPCH.

When the data server returns SQL rows for a logical table that is mapped to a path, the data server returns an instance of the first record type that is mapped with each instance of related records down the defined path. See the following example section.

#### Project guidelines

We divided the following three client scenarios into three categories to help you determine how much effort is involved in each engagement:

► Category 1: SELECT only

In this category, a client wants to extract data only from its CA-IDMS database; the client does not want to purge or delete the data. Because the source data is not affected, the tables can be mapped in various ways to meet client requirements. To enhance performance, it is best to include as many IDMS tables in the mapping as possible.

► Category 2: SELECT with DELETE

In this category, a client wants to extract data from its CA-IDMS database, store the extracted data in an archive file, and delete that data from the source database. The mapping in this category is more complicated. You must know the client's IDMS record structure and SET definitions, because the source data will be deleted.

In IBM InfoSphere Classic Federation Server for z/OS, a delete from a CA-IDMS table results in an @ERASE PERMANENT of the last record type that is mapped in the table. CA-IDMS automatically cascades the ERASE

statement to members of mandatory sets and disconnects members of optional sets when CA-IDMS issues this type of ERASE.

To delete multiple IDMS records (files) in CA-IDMS, you must map one table for each record that you want to delete. In addition, the member SET option must be set to AUTOMATIC. Otherwise, the record might not be deleted, or data integrity problems might occur.

Here is a typical client scenario for this category:

- A client has too much old data in CA-IDMS. The extra data causes performance problems and extra maintenance. The client wants to delete several of those records and save them in an archive file for future reference. There is no need to reinsert the records in IDMS.

**Delete:** If an IDMS record has a child with a MANDATORY member SET option, a delete with cascade effect will delete the child, even if the child record is not part of the archive table.

Updates can only be done on the last mapped record in the path.

► Category 3: SELECT with UPDATE/INSERT

In this category, there is a requirement to restore records after they are archived. If these record types are members of multiple member sets, a special INSERT-only table must be used in most cases. This INSERT-only table must be mapped with Classic Data Architect and cannot be used for SELECT, DELETE, or UPDATE purposes. This is a limitation of the IDMS SET updating function.

To create an archive file, another table with the same set of columns must be mapped within Classic Data Architect. In the Optim solution, you can archive data under one table name and insert the record back into the database under another table name.

Here is a typical client scenario for this category:

- A client wants to archive its customer file but wants the flexibility to restore the records to CA-IDMS. The Customer file has few parent records, and all record SET options are set to AUTOMATIC.

### Hints and tips:

- ▶ If an INSERT-only record has only one parent, it can be used in a SELECT statement.
- ▶ To perform an insert operation on an INSERT-only record with multiple parent records, all of the parent records must already exist or the insert will fail. In addition, the mapped table cannot be used in a SELECT statement.
- ▶ If the RECORDS SET option is set to MANUAL, instead of AUTOMATIC, an INSERT/DELETE operation might cause data integrity problems.

Consider this additional information:

- ▶ If a restore is not required, each record can be mapped as one logical table, and the parent/child relationship can be defined in your Optim solution.
- ▶ If the archive data will be restored, each record must have one table for SELECT and one table for DELETE.
- ▶ If the target database is relational, use the target database structure in most cases to map the data.
- ▶ If the target database is IDMS, map one table for INSERT, UPDATE, and DELETE operations for each IDMS record.
- ▶ A separate table must be matched for insert operations only.

### Procedure

Follow this setup procedure for accessing CA-IDMS data:

1. Map your CA-IDMS database to a relational table and optionally a view by using the New CA-IDMS Table wizard:
  - a. Open the wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object** → **CA-IDMS table**.
  - b. Select the CA-IDMS schema and subschema on which to base the table.
  - c. Choose whether to use the table for queries, updates, or both.
  - d. Choose whether to create a view on the table.
  - e. Provide information about how to access the CA-IDMS database.
  - f. For each record in the path, specify a COBOL copybook, select an 01 level if there are more than one 01 level, and then select the elements that you want to map as columns in your relational table.

- g. Select the elements that you want to map to columns in your relational table.
  - h. If you are creating a view, specify the criteria for the WHERE clause.
- When you finish the wizard, the new table appears under the selected schema. If you created a view, the view also appears under the selected schema.
2. Optional: Modify the table properties or add privileges. Select the table and make any changes in the Properties view.
  3. Optional: Create one or more indexes on the table.
  4. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema:
    - a. Right-click the table and select **Generate DDL**.
    - b. In the Generate DDL wizard, follow these steps:
      - i. Choose to generate CREATE statements.
      - ii. Choose to generate DDL for tables. You can also choose to generate DDL for indexes.
      - iii. Name the file in which to save the DDL within your project.
      - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine if the DDL ran successfully.
      - v. Choose whether you want to open the DDL for editing.
  5. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your CA-IDMS database. Be sure that the data server is connected to that database.
    - a. In the Database Explorer, search your data server for the schema in which you created the table. Expand the schema and expand the **Tables** folder.
    - b. Right-click the table and select **Data → Sample Contents**.
    - c. Check the Data Output view to determine whether the test query ran successfully.
  6. Optional: If you created a view, you can generate the DDL for the view now or later. You can also generate the DDL for all of the objects within the same schema.
    - a. Right-click the view and select **Generate DDL**.
    - b. In the Generate DDL wizard, follow these steps:
      - i. Choose to generate CREATE and ALTER statements.

- ii. Choose to generate DDL for views.
  - iii. Name the file in which to save the DDL within your project.
  - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine whether the DDL ran successfully.
  - v. Choose whether to open the DDL for editing.
7. Optional: If you ran the DDL successfully on a data server, validate the view by running a test query against your CA-IDMS database. Be sure that the data server is connected to that database.
- a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the **Views** folder.
  - b. Right-click the view and select **Data** → **Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.

## 6.5.14 Accessing CA-Datcom

To query or update data in a CA-Datcom database, you must create a relational table that maps to that database. You can also create a view on the table to filter record types or to filter rows and columns.

One or more FIELD entities are associated with a table in a CA-Datcom database. These FIELD definitions describe the contents of the table. In the CA-Datcom documentation, FIELD entities are also referred to as *columns* when the SQL option is used to access the CA-Datcom table. The online help does not use the term “column” to refer to a CA-Datcom FIELD entity.

The table that you map to must have one or more CA-Datcom ELEMENT definitions associated with it. A CA-Datcom ELEMENT definition refers to one or more contiguous CA-Datcom FIELD entities and is the unit of data transfer between Classic federation and a CA-Datcom database.

Use the CA-Datcom Table wizard to create the table and, optionally, the view.

**CA-Datcom null indicator:** Fields that use the CA-Datcom null indicator are not supported.

Follow the setup procedure to access CA-Datacom data:

1. Map your CA-Datacom database to a relational table and, optionally, a view by using the New CA-Datacom Table wizard:
  - a. Open the wizard by right-clicking either the database in your data design project or one of the schemas within the database. Select **Add Classic Object** → **CA-Datacom table**.
  - b. Select the copybook or include file on which you want to base the table.
  - c. Choose whether to use the table for queries, updates, or both.
  - d. Choose whether to create a view on the table.
  - e. Provide information about the record entity to which you want to map and the user requirement table (URT) for accessing the corresponding CA-Datacom table.
  - f. Select the fields that you want to map to columns in your relational table.
  - g. If you create a view, specify the criteria for the WHERE clause.

When you finish the wizard, the new table appears under the selected schema. If you created a view, the view also appears under the selected schema.
2. Optional: Modify the table properties or add privileges. Select the table and make any changes in the Properties view.
3. Optional: Create one or more indexes on the table.
4. Optional: Generate the DDL for the table. You can generate the DDL later, if you do not want to generate it now. You can also generate the DDL for all of the objects within the same schema.
  - a. Right-click the table and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE statements.
    - ii. Choose to generate DDL for tables. You can also choose to generate DDL for indexes.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine if the DDL ran successfully.
    - v. Choose whether to open the DDL for editing.

5. Optional: If you ran the DDL successfully on a data server, validate the table by running a test query against your CA-Datacom database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema that you created the table in. Expand the schema and expand the **Tables** folder.
  - b. Right-click the table and select **Data → Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.
6. Optional: If you created a view, you can generate the DDL for the view now or later. You can also generate the DDL for all of the objects within the same schema.
  - a. Right-click the view and select **Generate DDL**.
  - b. In the Generate DDL wizard, follow these steps:
    - i. Choose to generate CREATE and ALTER statements.
    - ii. Choose to generate DDL for views.
    - iii. Name the file in which to save the DDL within your project.
    - iv. Choose whether to run the DDL on a data server. After you run the DDL, check the Data Output view to determine if the DDL ran successfully.
    - v. Choose whether to open the DDL for editing.
7. Optional: If you ran the DDL successfully on a data server, you can validate the view by running a test query against your CA-Datacom database. Be sure that the data server is connected to that database.
  - a. In the Database Explorer, search your data server for the schema in which you created the view. Expand the schema and expand the **Views** folder.
  - b. Right-click the view and select **Data → Sample Contents**.
  - c. Check the Data Output view to determine whether the test query ran successfully.

## 6.6 Optim Connect

Optim Connect is a comprehensive integration platform for the on-demand access and integration of enterprise data sources and your existing applications. Optim Connect runs on many platforms, such as Microsoft Windows, UNIX, VMS, and mainframes, and provides non-relational data access.

Optim Connect provides a modular solution that allows organizations to address multiple tactical requirements quickly, while relying on a comprehensive platform that allows reusability and addresses many needs.

You use Optim connect studio to configure and manage access to applications, data, and events on all machines running Optim Connect. You make configurations in the Design Perspective. This perspective has tabs for configuration and metadata. These tabs enable the following configuration:

- ▶ Setting up access to machines running Optim Connect
- ▶ Configuring the daemon, which manages the communication between Optim Connect machines
- ▶ Configuring metadata

Optim connect studio displays in the configuration pane (Figure 6-29) most of the elements that are required for your non-relational data source configuration.

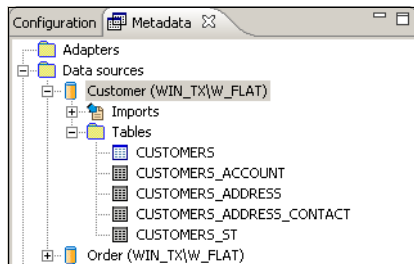


Figure 6-29 Optim Connect data access project definition

At the root level, Optim Connect has “machines” and each machine represents a unique computer with a TCP/IP address. Each machine requires an Optim Connect license key. In our case, we have chosen UNIX\_SD, WIN\_TX, ZOS\_CA, and ZOS\_CA. Every machine can be configured already and active or an offline design machine (all others). The offline design mode enables you to define the resources to Optim Connect in Optim connect studio without having to connect to the actual machine where the definitions are implemented. For example, you can set up a machine even when the actual server machine is down, or you can set up a number of definitions for separate machines on the same design machine. After the resources are defined on the design machine, you can drag and drop each definition to a server machine to implement the resources.

Every machine has a folder called *bindings*. Each binding represents a project with environmental properties and source information. Optim Connect always creates a default binding called *NAV*. You need to create your own binding (project), because default names are a weak link in securing an environment. You can have as many bindings as required. In our case, WIN\_TX has a binding

called W\_FLAT. The ZOS\_CA has two bindings: Z\_ADABAS and Z\_VSAM. By now, you probably understand our name conventions.

Every binding has three project types (or folders):

- ▶ Adapters
- ▶ Data sources
- ▶ Events

Your license enables only data sources, so that you can only run Data Source type projects. There is no limit to the number of projects that you can define. In our example, we have Customers and Sales flat files in the Texas data center running in a Microsoft Windows machine and Customers VSAM files running in a z/OS in the California data center.

If bindings represent the project definition, then *daemons* represent its implementation. A daemon is the listener for an Optim Connect service and has associated timing, logging, and security parameters to govern its execution.

Your data access project is executed in a *workspace*. A daemon can run one or more workspaces and each workspace can service one or more data sources. Normally, you create workspaces to satisfy the separation of environments and roles. A workspace can be executed as single client (a single use development or test execution), reusable (multiple use sandbox execution), and multi-client (shared execution). You normally test in single client or reusable mode and run production in reusable or multi-client mode. A workspace has its own user security and logging. Optim Connect defines Navigator, a default workspace, but in our case we have WIN\_TX daemon providing service to the workspace ONLY\_FLAT and ZOA\_CA to JUST\_ADABAS, JUST\_VSAM and OTHER\_STUFF.

The binding definition that was created in this example is not stored locally. It is translated into XML, shipped, and stored by the actual execution engine (in this case, WIN\_TX or \\WINSERVER:2551). Example 6-2 shows the example binding definition in XML.

---

*Example 6-2 XML binding definition*

---

```
<?xml version="1.0" encoding="UTF-8"?>
<binding name="W_FLAT">
  <remoteMachines name="W_FLAT">
    <remoteMachine name="WIN_TX"
      address="WINSERVER" port="2551"
      workspace="ONLY_FLAT"/>
  </remoteMachines>
  <datasources name="W_FLAT">
    <datasource name="Accounting" type="ADD-FLAT">
      <config newFileLocation="C:\\FlatFiles\\Accounting"/>
    </datasource>
  </datasources>
</binding>
```

```

        </datasource>
        <datasource name="Sales" type="ADD-DELIMITED">
            <config newFileLocation="C:\FlatFiles\Sales"/>
        </datasource>
    </datasources>
    <adapters name="W_FLAT"/>
</binding>

```

---

To further understand, install, and configure this infrastructure, reference the following Optim Connect documentation:

- ▶ *IBM Optim Connect User Guide and Reference*
- ▶ *Data Source Extensions for z/OS Solution Guide*
- ▶ *IBM Optim Connect Installation Guide for Windows*
- ▶ *IBM Optim Connect Installation Guide for UNIX*
- ▶ *IBM Optim Connect Installation Guide for Mainframe Systems*

Here is the list of key terminology that is used in conjunction with Optim Connect:

- ▶ **ADD**  
Optim Connect data dictionary, which includes metadata for data sources and applications.
- ▶ **Binding**  
Every workspace has a definition of which binding it works with. Bindings include data sources and environments.
- ▶ **Connection pooling**  
A cache of connections maintained in memory so that the connections can be reused when future requests are received. Connection pooling is handled by the daemon, by setting a number of parameters, including the server mode (making the server process reusable) and the number of available servers.
- ▶ **Daemon**  
The daemon's major function is to house the assemblage of workspaces. A series of daemons can be made available. The default is called IRPCD. Workspaces are underneath daemons in the Optim connect studio tree.
- ▶ **Data source**  
Data access within Optim Connect is divided into data sources. A data source can be of the type Adabas, VSAM, and so on.
- ▶ **Database adapter**  
A special application adapter in which the underlying application is the Optim Connect query processor.

► Extended ADD

An ADD repository that is used in conjunction with a non-ADD-based data source. It does not store the metadata, but rather it augments information. For example, The Extended ADD commonly is used for storing cardinality information for NonStop SQL and Adabas-Predict. Another example is virtual tables or virtual views for arrays within Adabas-Predict.

► File pool

Caching of file handles in a pool to reduce the amount of open and close operations on files. File pools are used in conjunction with file system data sources, such as VSAM, flat files, and so on.

► File system data source

In Optim Connect terms, any data source that does not accept SQL commands is considered a file-system data source. Examples include Adabas, flat file, VSAM, and so on.

► IRPCD

The default daemon.

► Machine

A computer system that is being used as either the client or server in a data access or application access scenario.

► Metadata

The term *metadata* is used both in conjunction with data access and application access scenarios. In data access, metadata consists of table definitions, including columns, data types, indexes, cardinality, file locations, and so on. In application access, metadata consists of interactions, input records, output records, and their respective columns and data types. Metadata can either be generated and stored within Optim Connect, or it can be retrieved dynamically from any back-end system that provides metadata. The Oracle database is an example of a back-end system that holds its own metadata. The VSAM data source is an example of a back-end system that uses the Optim Connect data dictionary (ADD) to store metadata.

► NAV\_UTIL

The Optim Connect command-line interface, with its own collection of commands. The commands include troubleshooting utilities and metadata utilities. All of the commands run from the **NAV\_UTIL** command-line utility (or **NAVCMDB** on IBM z/OS systems).

► Navigator

The name of the default workspace.

- ▶ Native object store

The native object store is the physical persistent storage for all Optim Connect definitions, including bindings, daemons, ADD metadata, and adapter definitions. Every Optim Connect installation has at least one repository, which is called SYS. The SYS stores all definitions with the exception of ADD metadata, which is stored in separate repositories.

- ▶ Prestarted server

A prestarted server is a process, which is started when the daemon starts and kept in a pool. Prestarted servers are immediately available for use by new client processes, saving initialization time. Instead of starting a new server process each time that a new server process is requested by a client, the client receives a process from the pool of available processes. When the client finishes processing, this server process either dies or, if reusable servers have been set, is returned to the pool of available servers.

- ▶ Process

In Optim terminology, a *process* is an execution context in which the program code runs. You can have more than one process running the same program. In MVS™ terminology, a process is equivalent to a task.

- ▶ Query processor

An Optim Connect software component when executing a data access solution. The query processor accepts SQL requests from client applications and tools, and works with the query optimizer to devise and implement an access strategy for executing the SQL request. The query processor is distributed by design so that an SQL request can be made by a group of query processors on several machines working at the same time.

- ▶ Query optimizer

An Optim Connect software component that is used in choosing an efficient access strategy for a given client SQL request. The Optim Connect query optimizer is a cost-based optimizer. It attaches a cost to every potential access strategy according to indexes, such as cardinality. It then chooses the access strategy with the lowest cost.

- ▶ Repository

Optim Connect holds information internally in the repository. There are two types of repository:

- General repository: Also referred to as the SYS repository. The general repository stores all Optim Connect definitions, such as bindings, daemons, adapter definitions, user profiles, remote machines, and so on.

- Data source repository: A data source repository can exist for every data source. For ADD-based data sources, this repository includes full metadata describing the tables, columns, indexes, and so forth. For non-ADD-based data sources (for example, relational data sources), the repository can include extended information, such as cardinality. We refer to this repository as extended ADD.
- ▶ Reusable server

After the client processing finishes, the server process does not die and can be used by another client, reducing start-up times and application start-up overhead. This mode does not have the high overhead of single-client mode, because the servers are initialized already. However, this server mode might use a lot of server resources, because it requires as many server processes as concurrent clients.
- ▶ Server machine

A machine that can supply data to other machines is called a server. The query processor and the inter-machine communications components are generally present on every one of the machines in an Optim Connect network. The Optim Connect server interface programs to the specific data sources and applications generally reside on the same machine as the respective data sources and applications.
- ▶ Single client

Each client receives a dedicated server process. The account in which a server process runs is determined either by the client login information or by the specific server workspace. This mode enables servers to run under a particular user account and isolates clients from each other (because each receives its own process). However, this server mode incurs a high overhead due to process start-up times and can use a lot of server resources, because it requires as many server processes as concurrent clients.
- ▶ Two-phase commit

A protocol for reliably committing changes across multiple systems, even in the case of network failures or node failures. In the protocol, a transaction manager calls each participating resource manager, first to prepare to commit the changes, and then to actually commit them. These two phases allow the transaction manager to get a commitment from all participating resource managers that they will be able to actually commit the changes. After all participating resource managers agree to commit, the transaction manager asks each resource manager to commit the changes.
- ▶ Virtual table

One of the ways that the Optim Connect server exposes an array for SQL access. Using virtual tables, one can view arrays within a table as if they were separate tables. Virtual tables have full SQL functionality.

► Virtual view

One of the ways that the Optim Connect server exposes an array for SQL access. Virtual views enhance performance on joins between a table and an array within it by implementing the join at retrieval time rather than at query processing time.

► Workspace

A server is the smallest service unit that is managed by the daemon. Every server belongs to a specific workspace. A workspace's function depends on its binding. A workspace defines the server processes and the environment that is used for the communication between the client and the server machine for the duration of the client request. A workspace definition includes the data sources that can be accessed as well as various environment variables.

To identify the design components of Optim Connect and the runtime design components, reference Figure 6-30.

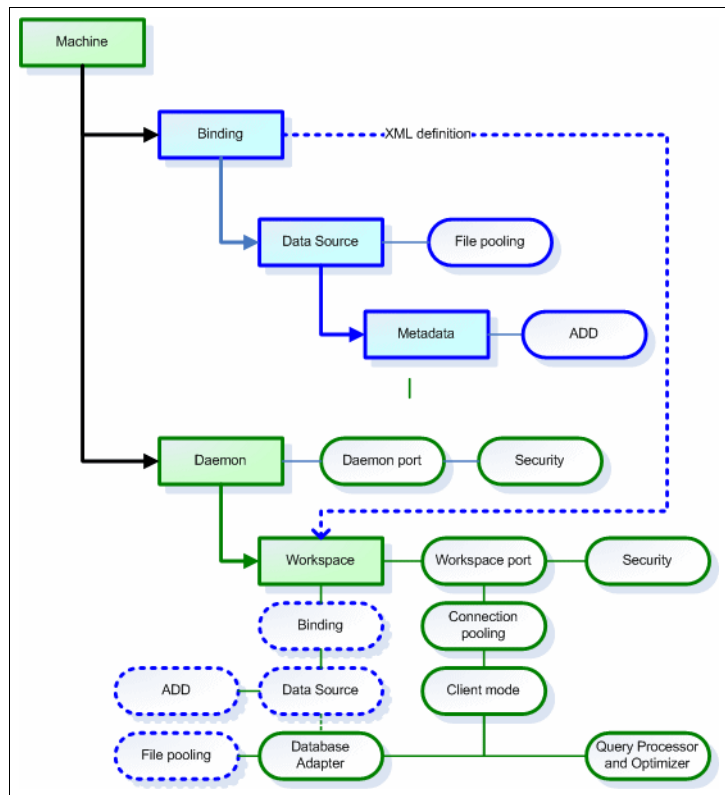


Figure 6-30 Optim Connect design and runtime components

The blue objects (Binding, Data Source, and Metadata) represent design definitions that are used to create data access objects. The green objects (Machine, Daemon, and Workspace) represent runtime objects. The white bubbles represent the key configurations of the related objects. The dotted objects and lines represent a design object that is deployed to the runtime engine. Every aspect of Optim Connect data access projects ID is defined using Optim connect studio.

## 6.6.1 Setup procedure

In this section, we provide the procedure for setting up bindings, daemons, and workspace.

### Setting up bindings

You use the Design perspective Configuration view to configure Optim Connect machines and applications, data, and events on those machines.

#### ***Process to add a machine***

Perform these two steps to add a machine:

1. Open Optim Connect Studio.
2. In the Design perspective Configuration view, right-click the **Machines** folder and select **Add Machines**. The Add machine window opens.

After you add a machine, it is displayed in the Configuration view under the Machine folder. You can edit the machine's login information, or you can configure bindings, daemons, and users for each machine.

#### ***Process to define an offline machine***

Perform these steps to define an offline machine:

1. In the Configuration view of the Design perspective, right-click the **Machines** folder and select **Add Offline Design Machine**. The Add offline design machine window opens.
2. Enter a name for the design machine.
3. Click **Finish**.

You can define all available resources on this machine. You also can set up metadata using a metadata import utility. Every resource is available on the design machine, no matter on which platform the resource must exist. For example, both Microsoft Windows data sources and z/OS data sources are available, even though on completion, you can only drag and drop the z/OS definitions (such as the IMS data source) to a z/OS machine.

## Binding configuration

The information that Optim Connect needs to access data sources is defined in a binding configuration.

A binding configuration always exists on a server machine, where the data sources to be accessed using Optim Connect reside. Additionally, You can define a binding configuration on a client machine (which is not covered in this book) to point to data sources on a server machine.

Use Optim connect studio to configure one or more bindings, each with a set of application adapters, data sources, and events. Each binding configuration has its own environment that defines the binding (such as cache sizes for storing information in memory during a session). The following sections describe the required tasks to define a binding.

You can set up a number of separate bindings in which each binding is for a separate set of data sources or projects. You can also create separate binding configurations for the same data source, with each binding configuration having a separate set of requirements, including a separate set of users.

### ***Server binding***

A binding configuration on a server includes the following content:

- ▶ Definitions for data sources that are accessed using Optim Connect
- ▶ Pre-loading commands that let you prepare the server to be executed when starting up, which provides you the ability to carry out multiple actions, such as server warm-up
- ▶ Environment properties that apply to all the data sources, adapters, and machines that are listed in the binding configuration

### ***Server warm-up***

Optim Connect supports warming up for pre-started servers by executing one or more adapter interactions when Optim Connect servers start. The benefit of warming up servers is the elimination of start-up delays when client requests arrive at a server for the first time. A warmed-up server can provide a fast response from the first request it handles.

To set up server warm-up, you must edit the binding by performing the following steps in the *<adapter>* element:

- ▶ Add an attribute called *preloaded* with a value true (*preloaded='true'*). Add a child element called *<onServerPreload>*.
- ▶ Add the interactions or queries that you want to run when the server starts up to the *<onServerPreload>* element.

Example 6-3 shows an example of editing the `<adapter>` element to define server warm-up.

*Example 6-3 <Adapter> element for defining server warm-up*

```
<adapter name='query' type='query' preloaded='true'>
  <onServerPreload actionType='interactions'>
    <query>select * from SomeTable_1 limit to 1 rows</query>
    <query>select * from SomeTable_2 limit to 1 rows</query>
    <query>select * from SomeTable_n limit to 1 rows</query>
    ...
  </onServerPreload>
</adapter>
```

**Important:** Add the `<onServerPreload>` element after the `<config>` element. Ensure that you limit the number of rows that are selected by using the LIMIT TO 1 ROWS syntax.

### ***Process to add a new binding***

Use these steps to add a new binding:

1. Open **Optim connect studio**.
2. In the Design perspective, Configuration view, expand the **Machines** folder.
3. Expand the machine where you want to add the binding.

You can add a new binding configuration in a design machine in the offline design mode and later drag and drop the binding to this machine. For more information, see Chapter 2, “Using an Offline Design Machine to Create Optim Connect Definitions” in *IBM Optim Connect User Guide and Reference*.

4. Right-click the **Bindings** folder and select **New Binding**.
5. Enter a name for the binding in the New Binding window.
6. Click **Finish**. The new binding editor opens. For information about entering information in the binding editor, see “Editing Bindings” in Chapter 3 of *IBM Optim Connect User Guide and Reference*.

### **Setting up daemons**

Daemons manage the communication between machines running Optim Connect. The daemon is responsible for allocating Optim Connect Server processes to clients. A daemon runs on every machine running Optim Connect.

The daemon authenticates clients, authorizes requests for a server process within a certain server workspace, and provides the clients with the required

servers. When a client requests a connection, the daemon allocates a server process to handle this connection and refers the client to the allocated process.

**Default IRPCD daemon:** The configuration that is supplied with the product installation includes the default IRPCD daemon. This configuration is used when no other daemon is configured to access the machine that is requested by a client machine.

### ***Server daemon***

When you want to add a new daemon in the Design perspective, you use the standard configuration information or copy the configuration information from another daemon. When you edit the daemon, you can make custom changes to its configuration.

A machine can have more than one daemon running at the same time, each on its own port. You can add a new daemon configuration in off-line design mode in a design machine, and later, you can drag and drop the daemon configuration to this machine. The daemon editor can contain four additional tabs for workspace information. To display both the daemon and workspace configuration, right-click a workspace under the daemon and select **Edit Workspace**.

### ***Process to add a new daemon***

Use these steps to add a new daemon:

1. Open **Optim connect studio**.
2. In the Design perspective Configuration view, expand the Machines folder and then expand the machine where you want to add the daemon.
3. Right-click the **Daemons** folder and select **New Daemon**. Complete these tasks in the New Daemon dialog box:
  - a. Enter a name for the new daemon.
  - b. Select one of the following options:
    - Create an empty daemon with default values.
    - Copy the properties from another daemon. If you choose to copy the properties of an existing daemon, click **Browse** and select the daemon from which you want to copy the properties.
4. Click **Finish**. The Daemon editor opens on the right side of the workbench. This editor contains three tabs.

### **Setting up workspaces**

A daemon must have one or more workspaces. *Workspaces* define the server processes and environment that is used for the communication between the

client and server throughout a client request. You set the workspace definition in the Optim Connect Studio Design perspective Configuration view.

When you define a new workspace, you can copy the values of an existing workspace on the same daemon or have Optim Connect set its default values.

### ***Process to add a new workspace***

Use these steps to add a new workspace:

1. Open **Optim connect studio**.
2. In the Design perspective Configuration view, expand the **Machine** folder and then expand the machine with the daemon where you want to add the workspace.
3. Expand the daemon folder.
4. Right-click the daemon where you want to add the workspace and select **New Workspace**.
5. In the New Daemon Workspace window, enter the following information:
  - a. Name: The name that is used to identify the workspace. The workspace name is made up of letters, digits, underscores (\_), or hyphens (-).
  - b. Description: A short description of the workspace.
6. From the Workspace data section, select one of the following options:
  - Create an empty workspace with default values.
  - Copy the properties from another workspace: If you copy the properties from another workspace, the fields beneath the selection become active. You must indicate the workspace from where you want to copy the properties. Enter the following information: *<name of the workspace>* in the *<name of the daemon where the workspace is located>* on the *<name of machine where the daemon is located>*. Or, you can click **Browse** to select the workspace that you want to use, in which case, the information is added automatically.
7. Click **Next** to open the Select Scenario window. Select the type of applications with which the daemon works from the following options:
  - Application server using connection pooling.
  - Stand-alone applications that connect and disconnect frequently.
  - Applications that require long connections, such as reporting programs and bulk extractors.

- Custom (configure manually). If you select this option, the Workspace editor opens. Click **Next** to open the next window. Select one of the following options. The available options depend on the scenario that is selected:
  - The minimum number of server instances available at any time  
This option is the minimum number of connections that is available at any time. If the number of available connections drops beneath this number, the system will create new connections (available if you select Stand-alone applications that connect and disconnect frequently).
  - The maximum number of server instances available at any time  
This option is the maximum number of connections that is available at any time. If the number of connections that is used reaches this number, no additional server connections can be made (available if you select Stand-alone applications that connect and disconnect frequently).
  - The average number of expected concurrent connections  
This option lets the system know how much the average load will be. This option helps to distribute the resources correctly (available if you select Application server using connection pooling or if you select Stand-alone applications that connect and disconnect frequently).
  - The maximum number of connections  
This option is the greatest number of connections that will be available. If the number of requests exceeds this number, an error message is displayed that informs the user to try again when a connection becomes available (available if you select Application server using connection pooling or if you select Stand-alone applications that connect and disconnect frequently).
  - How many connections you want to run concurrently  
This option sets the number of connections that will run at the same time (available if you select Applications that require long connections, such as reporting programs and bulk extractors).
- 8. Click **Next** and enter the amount of wait time for the following parameters. If your system is not too overloaded, you can leave the default times.
  - How long to wait for a new connection  
Enter the amount of time (in seconds) to wait for a connection to be established before the system times out. For example, if you want a wait time of one minute, enter 60 (the default). If you enter 0, the time is unlimited.

- How long to wait for a response that is usually fast  
Enter the time (in seconds) to wait for a response from the system before the system times out. For example, if you want to wait for one minute, enter 60. The default is 0, which indicates unlimited wait time.
- 9. Click **Next** to open and enter the workspace security information in this window. You can determine which users or groups can access the workspace that you are defining.
- 10. Click **Next** to open the summary window. Review the summary to be sure that all the information that you have entered is correct. If you need to make any changes, click **Back** to go back to previous steps and change the information.
- 11. Click **Finish** to close the wizard and add the new workspace to the Configuration view.

## 6.6.2 Managing metadata

Metadata defines the structure of the data and where it is located. Optim Connect server relies on the native metadata of the data source when connecting to certain file-system data sources (such as Adabas, when using Predict). For other data sources whose metadata is not readable by the Optim Connect server or which do not have metadata, Optim Connect server requires its own metadata.

Optim Connect metadata is stored in a proprietary data dictionary called the Optim Connect data dictionary (ADD). You need Optim Connect metadata for each record, which is referred to as a table, in the data source. The Optim Connect metadata definition for a record is viewable and can be updated by using the Design perspective Metadata tab of Optim connect studio or using an XML file. The Optim Connect metadata for each data source is stored in a repository for the data source, on the machine where the data resides.

The following objects require Optim Connect metadata:

- ▶ IMS/DB data sources (z/OS only)
- ▶ Text-delimited file data source
- ▶ VSAM data source (z/OS)
- ▶ Adabas C data source without Predict

You can import, save, and manage metadata in the Optim connect studio Design perspective's Metadata tab.

### Importing metadata

You can use the Optim connect studio import wizards or stand-alone import utilities to generate metadata. If an import wizard or stand-alone utility is not

available, you can create the metadata manually. You can import metadata using an Optim connect studio import wizard for IMS/DB data sources.

For other data source drivers, if a COBOL copybook is available, metadata can be generated from the COBOL in Optim connect studio. Otherwise, you must define the metadata manually in the Optim Connect Studio Design perspective Metadata tab.

If COBOL copybooks describing the data source records are available, you can import the metadata by running the metadata import in the Optim Connect Studio Design perspective Metadata tab. If the metadata is provided in a number of COBOL copybooks, with separate filter settings (such as whether the first six columns are ignored), you import the metadata from copybooks with the same settings and later import the metadata from the other copybooks. You can save an import procedure and use it again.

Each data source has separate import requirements. You might define amounts differently and supply separate information in certain data sources.

## Process to import metadata

Use these steps to import metadata:

1. Open **Optim connect studio**.
2. In the Design perspective Configuration view, expand the **Machines** folder and then expand the machine with the data source where you are importing the metadata.
3. Expand the **Bindings** folder and expand the binding with the data source metadata with which you are working.
4. Expand the **Data Source** folder.
5. Right-click the data source with which that you are working with and select **Show Metadata View**. The Metadata view opens with the selected data source displayed.
6. Right-click **Imports** under the data source and select **New Import**. The New Import dialog box is displayed.
7. Enter a name for the import. The name can contain letters, numbers, and the underscore character.
8. Select **COBOL Import Manager for Databases**.
9. Click **Finish**.

## Process to manage metadata

You manage metadata in Optim connect studio. Click the **Metadata** tab in the Design perspective to view and modify Optim Connect metadata for data sources.

To view and modify metadata for data sources, perform these steps:

1. In the Design perspective Configuration view, right-click the data source for which you want to manage the metadata.
2. Select **Edit metadata** from the shortcut menu. The Metadata tab opens with the selected data source displayed in the tree. (You can also open the Metadata tab and right-click the Data sources folder in the Metadata view to add the data source for which you want to import metadata to the tree.)
3. Right-click the resource, such as the data source table, in the Metadata view and select **Edit**. You edit the data source tables by using the following tabs, which are at the bottom of the window:
  - General tab: Defines general information about the table, such as the table name, the way that the table is organized, and the location of the table.
  - Columns tab: Specifies the table columns and their properties, for example, the column data type, size, and scale.
  - Indexes tab: Enables you to specify the indexes of a table. The indexes are described by the order of the rows that they retrieve, the data source commands that are used, and the index type.
  - Statistics tab: Enables you to specify statistics for the table, including the number of rows and blocks of the table.

**Relational model terminology:** Optim Connect server provides a relational model for all data sources that are defined to it. Thus, relational terminology is used, even when referring to non-relational data sources. For example, the metadata for a IMS record is referred to as the metadata for an IMS table.

## Extended native data source metadata

When native metadata lacks certain features that are provided by Optim Connect metadata, you can improve performance by creating or extending the native metadata with ADD metadata.

For example, you can use the Optim Connect metadata to specify the number of rows and blocks in a VSAM table. When accessing the data, the Optim Connect server uses this extended metadata.

### ***Process to assign extended metadata for a data source***

To assign extended metadata for a data source, perform these steps:

1. Display the metadata for the data source in the Design perspective Metadata tab of Optim connect studio.
2. Change the relevant values for the table to be extended in the Statistics tab. The table symbol in the tree is marked with an asterisk to show that the metadata has been extended.

Note that the information in other tabs is for reference only and cannot be edited. The noExtendedMetadata property in the data source definition in the binding configuration is set to false.

Sometimes, you can improve performance by using Optim Connect metadata instead of the native metadata. In this case, you can export a snapshot of the native metadata to the Optim Connect server and use this local copy of the native metadata when accessing the data source.

### ***Process to enable metadata caching***

When access to a data source through its native metadata is slow but the metadata is static, you can improve performance by creating a local copy (snapshot) of the data source metadata. Then, you can run queries using this metadata instead of the data source metadata.

This approach is beneficial, for example, when native metadata is not up-to-date or information, such as statistics, is not available. You can see a snapshot of the metadata in Optim connect studio.

Follow these steps to make a copy of the data source metadata:

1. Display the metadata for the data source in the Design perspective Metadata tab of Optim connect studio.
2. Right-click the data source and select **Manage Cached Metadata** from the pop-up menu.
3. Select the tables that you want to use for a local copy and move them to the right pane.
4. Click **Finish**. The tables are displayed under the data source.

Only the tables that have cached metadata are displayed in the tree. There are two ways to revert back to using non-cached metadata. You can right-click individual tables and choose **Delete Cached Table** from the pop-up menu. Or, for all the tables, right-click the data source, choose **Set Metadata**, and choose **Native Metadata** from the pop-up menu.

## Importing COBOL copybooks

You can use the COBOL Import wizard with most of the Optim Connect data sources. The following procedure describes how to use the COBOL import:

1. Open **Optim connect studio**.
2. In the Design perspective Configuration view, expand the **Machines** folder and then expand the machine with the data source where you are importing the metadata.
3. Expand the **Bindings** folder and expand the binding with the data source metadata with which you are working.
4. Expand the **Data Source** folder.
5. Right-click the data source with which you are working and select **Show Metadata View**. The Metadata view opens with the selected data source displayed.
6. Right-click **Imports** under the data source and select **New Import**. The New Import dialog box is displayed.
7. Enter a name for the import. The name can contain letters, numbers, and the underscore character.
8. Select **COBOL Import Manager for Datasources** as the Import type from the list. Certain data sources might have additional import types available.
9. Click **Finish**.

## Metadata model selection

You can generate virtual and sequential views for imported tables containing arrays (which we discuss later in this chapter). In addition, you can configure the properties of the generated views, which allows you to flatten tables that contain arrays.

In the metadata model selection step, you can select configuration values that apply to all the tables in the import or set specific settings for each table.

To configure the metadata model, select one of the following options:

- Default values for all tables  
Select this option if you want to configure the same values for all the tables in the import. Make the following selections when using this option:
  - Generate sequential view: Select this option to map non-relational files to a single table.
  - Generate virtual views: Select this option to have individual tables created for each array in the non-relational file.

- Include row number column. Select one of the following options:
  - True: Select true to include a column that specifies the row number in the virtual or sequential view. This option is true for this table only, even if the data source is not configured to include the row number column.
  - False: Select false to not include a column that specifies the row number in the virtual or sequential view for this table even if the data source is configured to include the row number column.
  - Default: Select default to use the default data source behavior for this parameter.
- Inherit all parent columns. Select one of the following options:
  - True: Select true for virtual views to include all the columns in the parent record. This option is true for this table only, even if the data source is not configured to include all of the parent record columns.
  - False: Select false so that virtual views do not include the columns in the parent record for this table even if the data source is configured to include all of the parent record columns.
  - Default: Select default to use the default data source behavior for this parameter.
- Specific virtual array view settings per table
 

Select this option to set separate values for each table in the import. This option will override the data source default for that table. Make the selections in the table under this selection.

## Handling arrays

The Optim Connect server exposes a purely relational front end through APIs, such as Oracle HSODBC, ODBC, or JDBC. However, the Optim Connect server connects to non-relational data sources, which include non-relational data models. Therefore, Optim Connect server provides a logical mapping that exposes the non-relational constructs in a relational manner. The most prevalent problem in this domain is the issue of arrays, or OCCURS constructs, which is described in this section.

## Representing metadata

Before looking at the various methods of handling arrays, you must understand how metadata is represented in Optim connect studio.

Example 6-4 on page 315 shows an example in COBOL that illustrates arrays and nested arrays. When you import this metadata into Optim connect studio, the import process creates an ADD definition that is equivalent to the original structure, usually mapping the fields one-to-one.

#### Example 6-4 COBOL arrays

```

01 CUSTOMERS.
  05 CUST-ID                PIC X(5).
  05 CUSTNAME               PIC X(20).
  05 MAXADDR                PIC 9.
  05 MAXACCOUNT             PIC 9.
  05 ADDRESS OCCURS 5 TIMES DEPENDING ON MAXADDR.
    10 STREET               PIC X(50).
    10 CITY                 PIC X(15).
    10 STATE                PIC X(2).
    10 ZIP                  PIC X(5).
    10 MAXCONTACT           PIC 9.
    10 CONTACT OCCURS 3 TIMES DEPENDING ON MAXCONTACT.
      15 LOCATION           PIC X(24).
      15 EMAIL              PIC X(50).
      15 PHONE              PIC X(15).
  05 ACCOUNT OCCURS 1 TO 3 TIMES DEPENDING ON MAXACCOUNT.
    10 YTD-SALES            PIC S9(5)V99 COMP-3.
    10 ACCOUNT-ID           PIC X(6).
    10 ACCOUNT-NUMBER       PIC X(10).

```

The record CUSTOMERS has three fields (ADDRESS, CONTACT, and ACCOUNT) that are arrays. Each of them has a counter (MAXADDR, MAXCONTACT, and MAXACCOUNT) that drives the number of repeated entries. CONTACT is a nested array that can translate in 15 distinct values:

$$(\text{ADDRESS} * \text{MAXADDR}) * (\text{CONTACT} * \text{MAXCONTACT}) = 5 * 3 = 15$$

After the COBOL copybook import operation completes (Figure 6-31), you can validate that the definition has three arrays (data type = group) with a dimension that is greater than 0.

Column name	Data type	Size	Scale	Dimension	Offset	Fixed offset
CUST_ID	string	5		0	0	<input type="checkbox"/>
CUSTNAME	string	20		0	0	<input type="checkbox"/>
MAXADDR	numstr_u	1	0	0	0	<input type="checkbox"/>
MAXACCOUNT	numstr_u	1	0	0	0	<input type="checkbox"/>
ADDRESS	group			5	5	<input type="checkbox"/>
STREET	string	50		0	0	<input type="checkbox"/>
CITY	string	15		0	0	<input type="checkbox"/>
STATE	string	2		0	0	<input type="checkbox"/>
ZIP	string	5		0	0	<input type="checkbox"/>
MAXCONTACT	numstr_u	1	0	0	0	<input type="checkbox"/>
CONTACT	group			3	3	<input type="checkbox"/>
LOCATION	string	24		0	0	<input type="checkbox"/>
EMAIL	string	50		0	0	<input type="checkbox"/>
PHONE	string	15		0	0	<input type="checkbox"/>
ACCOUNT	group			3	3	<input type="checkbox"/>
YTD_SALES	decimal	7	2	0	0	<input type="checkbox"/>
ACCOUNT_ID	string	6		0	0	<input type="checkbox"/>
ACCOUNT_NUMBER	string	10		0	0	<input type="checkbox"/>

Figure 6-31 Optim Connect metadata window

Optim Connect has added a few virtual tables to the root table CUSTOMERS dynamically (the darker gray icons in Figure 6-32):

- ▶ CUSTOMERS\_ACCOUNT
- ▶ CUSTOMERS\_ADDRESS
- ▶ CUSTOMERS\_ADDRESS\_CONTACT

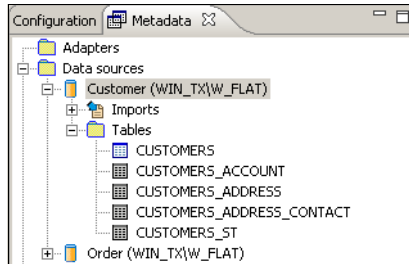


Figure 6-32 List of tables in metadata catalog (ADD)

CUSTOMERS\_ST (ST means single table) is the flattened version of the same table, and it is not accessible in this current configuration.

The same information is represented in XML format. To access it, we have clicked the metadata panel. Then, we selected the first table and clicked the **Source** tab. Example 6-5 shows the output that was generated by Optim Connect.

#### Example 6-5 Metadata in XML format

```
<?xml version="1.0" encoding="UTF-8"?>
<table description="" metadataArrayPolicy="parentPrimaryKey"
  nBlocks="0" nRows="0" name="CUSTOMERS" organization="index"
  recordFormat="variable" size="1787">
  <fields>
    <field datatype="string" name="CUST_ID" size="5"/>
    <field datatype="string" name="CUSTNAME" size="20"/>
    <field datatype="numstr_u" name="MAXADDR" size="1"/>
    <field datatype="numstr_u" name="MAXACCOUNT" size="1"/>
    <group counterName="MAXADDR" dimension1="5" name="ADDRESS">
      <fields>
        <field datatype="string" name="STREET" size="50"/>
        <field datatype="string" name="CITY" size="15"/>
        <field datatype="string" name="STATE" size="2"/>
        <field datatype="string" name="ZIP" size="5"/>
        <field datatype="numstr_u" name="MAXCONTACT" size="1"/>
        <group counterName="MAXCONTACT" dimension1="3" name="CONTACT">
          <fields>
            <field datatype="string" name="LOCATION" size="24"/>
            <field datatype="string" name="EMAIL" size="50"/>
            <field datatype="string" name="PHONE" size="15"/>
          </fields>
        </group>
      </fields>
    </group>
  </fields>
</table>
```

```

        </fields>
    </group>
</fields>
</group>
<group counterName="MAXACCOUNT" dimension1="3" name="ACCOUNT">
    <fields>
        <field datatype="decimal" name="YTD_SALES" scale="2" size="7"/>
        <field datatype="string" name="ACCOUNT_ID" size="6"/>
        <field datatype="string" name="ACCOUNT_NUMBER" size="10"/>
    </fields>
</group>
</fields>
<keys/>
<primaryKey name="CUSTOMERS_PK">
    <pKeySegments>
        <pKeySegment segment="CUST_ID"/>
    </pKeySegments>
</primaryKey>
</table>

```

---

Optim connect studio always generates XML statements, because generating XML statements is the only method to customize any of the Optim connect studio components. To see what the server presents to the application, we have to switch to the server machine (WIN\_TX) and run the utility **NAV\_UTIL**. Reference Chapter 30, “Using the NAV\_UTIL Utility,” in the manual *IBM Optim Connect User Guide and Reference*. If you have to check a z/OS machine, the utility name is **NAVROOT.USERLIB(NAVCMD)**.

To connect to the workspace W\_FLAT, access the data source Customer, and display the table Customers, we enter the commands:

```

nav_util execute -wW_FLAT customer
native customers

```

These commands display the metadata information that is available to the server, starting with the list of columns, followed by the indexes as shown in Example 6-6.

---

#### Example 6-6 Metadata

---

Table CUSTOMERS of application Customer (ADD-FLAT)

```

Table length is 1795
File organization is INDEX
Table format is 'VARIABLE'

```

There are 21 fields in the table:

N	LVL	Name	Datatype	Size	Width	Sc1	Ofst	Grp	Cnt	Dm1	Dm2
---	-----	------	----------	------	-------	-----	------	-----	-----	-----	-----

```

-----
0  1 CUST_ID          string          5      5  0    0 -1 -1  0  0
1  1 CUSTNAME         string          20    20  0    5 -1 -1  0  0
2  1 MAXADDR          numstr_u          1      1  0   25 -1 -1  0  0
3  1 MAXACCOUNT       numstr_u          1      1  0   26 -1 -1  0  0
4  1 ADDRESS          Group          340    0  0   27 -1  2  4  0
5  2 STREET           string          50    50  0    0  4 -1  0  0
6  2 CITY             string          15    15  0   50  4 -1  0  0
7  2 STATE            string          2      2  0   65  4 -1  0  0
8  2 ZIP              string          5      5  0   67  4 -1  0  0
9  2 MAXCONTACT       numstr_u          1      1  0   72  4 -1  0  0
10 2 CONTACT          Group          89     0  0   73  4  9  2  0
11 3 LOCATION         string          24    24  0    0 10 -1  0  0
12 3 EMAIL            string          50    50  0   24 10 -1  0  0
13 3 PHONE            string          15    15  0   74 10 -1  0  0
14 1 ACCOUNT          Group          20     0  0 1727 -1  3  2  0
15 2 YTD_SALES        decimal          4      7 -2    0 14 -1  0  0
16 2 ACCOUNT_ID       string          6      6  0    4 14 -1  0  0
17 2 ACCOUNT_NUMBER   string          10    10  0   10 14 -1  0  0
18 1 RFA              uint4           4      0  0 1787 -1 -1  0  0
19 1 XML              string          0      0  0 1791 -1 -1  0  0
20 1 ###BMK           UNSPECIFIE       4      4  0 1791 -1 -1  0  0

```

(1 index specified)

```

Index 1: length is 4, Unique, Best Unique
name is RFA
segments are:
RFA

```

(no connections specified)

(primary key specified)

```

Name is CUSTOMERS_PK
Segments are:
CUST_ID

```

<END OF TABLE>

---

**Note:** Inserting whole records of data, for which the model is hierarchical, requires the use of a special XML column in the parent table and, if selectively returning data to the non-relational data source, the use of an archive action. For this reason, only archived records can be selectively restored to a non-relational data source when the data model is hierarchical.

Within the Optim Connect binding, the “exposeXmlFieldsetting” value in the “Misc” property group must be set to true. When this change is saved, a new column with the name XML (see line 19 in Example 6-6 on page 317) becomes part of the underlying schema for the table or tables that were defined earlier in the metadata mappings. This column is required for any insert activity on non-relational files with hierarchical data models.

You can obtain more information about this topic in Chapter 6, section 6.5, “Returning Data to the non-Relational Data Source” in the *Data Source Extensions for z/OS Solution Guide*.

This definition is usable internally, but it is not usable by an ODBC connection. While we are in the **NAV\_UTIL** utility output, we can browse the table catalog:

```
NavSQL > list tables;
TABLE_NAME          TABLE_TYPE
CUSTOMERS            TABLE
CUSTOMERS_ACCOUNT    TABLE
CUSTOMERS_ADDRESS    TABLE
CUSTOMERS_ADDRESS_CONTACT TABLE
CUSTOMERS_ST         TABLE
```

5 rows returned5 rows returned

This example shows that our original table named Customers has been renamed to CUSTOMERS\_ST, and it is not accessible. Now, it has a few new virtual tables (the darker gray icons in Figure 6-33 on page 320):

- ▶ CUSTOMERS
- ▶ CUSTOMERS\_ACCOUNT
- ▶ CUSTOMERS\_ADDRESS
- ▶ CUSTOMERS\_ADDRESS\_CONTACT

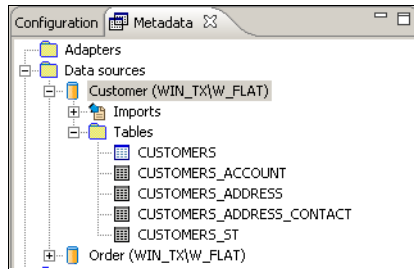


Figure 6-33 List of tables in metadata catalog (ADD)

Example 6-7 shows the output if we describe the Customers table.

#### Example 6-7 Customers table definition

NavSQL > **describe customers;**

Table CUSTOMERS

There are 9 fields in the table:

#	Name	Datatype	Size	Width	Sc1	Nullable
0	CUST_ID	string	5	5	0	no
1	CUSTNAME	string	20	20	0	no
2	MAXADDR	numstr_u	1	1	0	no
3	MAXACCOUNT	numstr_u	1	1	0	no
4	ADDRESS	cstring	65	64	0	no
5	ACCOUNT	cstring	65	64	0	no
6	RFA	uint4	4	0	0	no
7	XML	string	65	64	0	no
8	###BMK	UNSPECIFIED	4	4	0	no

(1 index specified)

Index 1: length is 4, Unique, Best Unique  
name is RFA  
segments are:  
RFA

<END OF TABLE DESCRIPTION>

Example 6-7 lists the original columns CUST\_ID, CUSTNAME, MAXADDR, and MAXACCOUNT. The arrays ADDRESS and ACCOUNT have been transformed in foreign keys with length 64. Because we are processing a flat file, Optim Connect has no natural way to identify this row from any other row in the file, so it adds the column RFA (Relative File Address) that points to the offset in the file,

and it is also displayed as an index. The last two columns are additional controls that were added by Optim Connect.

Example 6-8 shows how the first array has been translated.

*Example 6-8 First array definition*

---

```
NavSQL > describe customers_address;
```

```
-----  
Table CUSTOMERS_ADDRESS
```

```
Based on is 'CUSTOMERS::ADDRESS'
```

```
There are 9 fields in the table:
```

#	Name	Datatype	Size	Width	Scl	Nullable
0	STREET	string	50	50	0	no
1	CITY	string	15	15	0	no
2	STATE	string	2	2	0	no
3	ZIP	string	5	5	0	no
4	MAXCONTACT	numstr_u	1	1	0	no
5	CONTACT	cstring	65	64	0	no
6	_PARENT	cstring	65	64	0	no
7	_ROWNUM	int4	4	0	0	no
8	###BMK	UNSPECIFIED	8	8	0	no

```
(2 indexes specified)
```

```
Index 1: length is 65
```

```
name is P$CUSTOMERS$ADDRESS
```

```
segments are:
```

```
_PARENT
```

```
Index 2: length is 69, Unique, Hashed
```

```
name is U$CUSTOMERS$ADDRESS
```

```
segments are:
```

```
_PARENT
```

```
_ROWNUM
```

```
<END OF TABLE DESCRIPTION>
```

---

The five original columns are at the beginning of the table, and they are followed by two generated columns:

- **\_PARENT**: Identifies the parent record. This column is a string representation of the parent record's bookmark (###BMK).
- **\_ROWNUM**: The ordinal that identifies the array record.

Also, Optim Connect has generated the foreign key “\_PARENT” (matched by index 1 - P\$CUSTOMERS\$ADDRESS) and the primary key “\_PARENT + \_ROWNUM” (matched by index 2 - U\$CUSTOMERS\$ADDRESS). So, Optim

Connect can access (and maintain referential integrity) by locating the flat file record using “\_PARENT” and its location in the array through “\_ROWNUM”.

Example 6-9 shows how the nested array has been translated.

*Example 6-9 Nested array definition*

```
NavSQL > describe customers_address_contact;
-----

Table CUSTOMERS_ADDRESS_CONTACT
  Based on is 'CUSTOMERS_ADDRESS::CONTACT'
  There are 6 fields in the table:

#  Name                      Datatype      Size Width Scl Nullable
-----
0  LOCATION                  string         24   24   0       no
1  EMAIL                     string         50   50   0       no
2  PHONE                     string         15   15   0       no
3  _PARENT                   cstring        65   64   0       no
4  _ROWNUM                   int4           4     0   0       no
5  ###BMK                    UNSPECIFIED    12   12   0       no

(2 indexes specified)

  Index 1: length is 65
           name is P$CUSTOMERS_ADDRESS$CONTACT
           segments are:
             _PARENT

  Index 2: length is 69, Unique, Hashed
           name is U$CUSTOMERS_ADDRESS$CONTACT
           segments are:
             _PARENT
             _ROWNUM

<END OF TABLE DESCRIPTION>
```

Like in the prior table, we can see the two navigation and referential integrity columns and indexes. The same applies for the table Customers\_Account.

So, Optim Connect has mapped records to file offsets and translated the COBOL copybook into related tables with referential integrity.

**Special considerations for restore**

You can use Optim insert and restore processing to return data to non-relational data sources. When you work with a record layout that has OCCURS that have been translated into related tables, there are special considerations. The complete set of information has been converted into relational data; therefore,

restoring the parent without the related child data (OCCURS clause) will violate the non-relational referential integrity.

To restore data that has OCCURS, you have to stage data in the federation engine (Oracle or Federation Server) and rejoin the parent and child data (original record with occur data) before inserting it back in the original non-relational source.

Note that if your archive has 100 tables and only three were created as a consequence of an OCCURS clause, project consideration applies only to these three tables.

The process that is described in Figure 6-34 shows the original record layout with two OCCURS (address and account). The OCCURS are translated into corresponding tables. During archive, Optim splits the original record in three tables and stores them independently in the archive. During restore, the parent table is restored in a temporary relational table. Then, the next child table is restored to a separate temporary relational table and so on. At the end-of-restore process, the data from the parent table must be connected dynamically (in memory) to the child data using the XML specifications. For each reassembled row, you now can invoke Optim Connect to process the XML and store the data in the original non-relational data source.

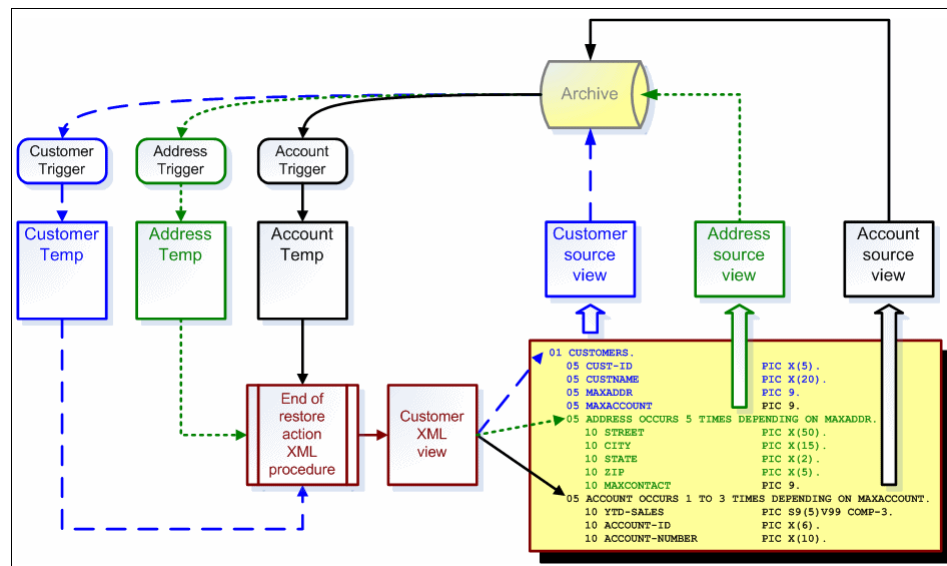


Figure 6-34 Restore process for records with OCCURS clause

This process uses triggers to divert the Optim insert process from the original non-relational data source to the temporary tables. The triggers ensure that data is not accidentally inserted before it is fully assembled.

You can obtain more documentation in Chapter 6.5, “Returning Data to the non-Relational Data Source,” in the *Data Source Extensions for z/OS Solution Guide*.



## Accessing and viewing archived data

Even though you have concluded that your databases include a significant volume of data that is rarely touched by applications, you cannot simply purge this data. You must have a convenient way to access it when needed. Your infrequently used data is a valuable asset that provides a historic reference to business activity. For example, your organization might be required to answer questions from clients, provide data for business trend analysis, or present documentation for tax or other regulatory purposes. Understanding the situations in which archived data is needed and providing a way to deliver the data to business users is a key element in the data archiving strategy.

In the context of accessing and viewing archived data, this chapter discusses these topics:

- ▶ The business requirements for accessing or viewing archived data
- ▶ How to provide access to archived data in ways that support the business requirements
- ▶ How to provide the performance that is indicated by the business requirements

We describe the data model that is used in the examples in this chapter in Appendix A, “Data model” on page 501.

## 7.1 Business requirements for accessing archived data

Archived data continues to provide business value for analysis and reporting purposes even after the data is purged from the production database. It is important that you anticipate and analyze each requirement or use case for accessing archived data as an integral part of your archive strategy. You must consider the estimates and projections of the frequency of the needed access to data purged from your production environment, the form in which the data is best presented, the quantity of data to be provided, and the maximum acceptable time lag in presenting it when deciding what to archive. On the basis of this analysis, you can decide when and how to present the information when needed and design your archiving environment accordingly.

You can access archived data as a stand-alone source of data for reporting purposes or combine it with active production data for analysis. If archived data must be federated with active data or temporarily restored to a database for reporting or other processing, you must determine the best way to accomplish the federation or restoration, the applications that will be affected, and any necessary modifications to those applications.

To evaluate the requirements for access to archived data, you must review each set of data, as categorized by function, criteria, and retention:

- ▶ Identify the data to be accessed.
- ▶ Examine the reason for the access.
- ▶ Establish the frequency with which archived data will be accessed, the duration for which access must be provided, and the volume of concurrent access.
- ▶ Estimate the maximum response time when a business event presents the need for access.
- ▶ Estimate the volume of data that must be accessed at one time.
- ▶ Determine the best method for access.

Armed with this information, you can develop the technical processes and administrative procedures that are required for timely responses to user requirements and initializing data retrieval and select the best storage media for the archived data.

### Data for access

Your user experts and business analysts must identify data thoughtfully to be queried, reviewed, or reported after it is archived and the context in which access is required.

Context is necessary to identify the proper criteria for access. The criteria that is used to identify archived data for access purposes differs from the criteria that is used to select the data for archiving. For example, a manufacturing company that archives orders on the basis of a date in the Orders table knows that, in the event of a product recall, it must obtain information about customers related to orders for a specific item. In this case, you must obtain the customer information on the basis of a value in the archived reference table, Items.

### **Reasons for access**

You must also examine and record the reasons, or potential causes, for access to archived data. Are accounts re-opened? If so, which accounts? Is inactive business data a resource for anticipated special projects or reports? What data is subject to legal requirements for data retention? What data is subject to audit? Do possible recalls of, or product liability claims for, manufactured items make it necessary to find the addresses to which items are shipped and the parties that receive them?

### **Frequency of access and duration of accessibility**

Requirements from user experts and business analysts determine how to divide transactional data into bands by age, validate estimates of the frequency with which data in each band is referenced or restored, and determine the period for which data must be accessible. For example, in discussing the requirements of business users, you might learn that material between 12 and 18 months old is referenced frequently, and data that is more than two years old is needed rarely, if ever. Also, requirements might indicate that access to statistics that are derived from data that is older than five years must be retained but the actual data can be discarded.

Note that categories of data from a single application can have a variety of retention and access requirements. For example, a utility company that archives billing information that is older than two years decides at first to include the underlying meter readings and rate information in the archive. The company must retain these readings and rate classifications indefinitely, but the company will access them rarely, if ever. The actual billing information, however, will not be retained after five years but they expect to be access it frequently over that period. The company needs procedures or methods to access that billing information. Based on these requirements, the company chooses to archive the data in two related steps. In the first step, it archives the readings and rate classifications data, using an archive action to insert an identifier for the archive file into the billing data, which is not archived. The billing data is then archived in the second step. When the archived billing data is accessed, the identifier for the file containing the archived reading and rate classification details is readily available in case it is needed.

It is important to make a realistic assessment of the frequency of access to data before it is archived. If archived data is more active than anticipated, you might need to restore it to your production environment or, perhaps, suffer much lower performance than your business users need.

One method for validating assumptions about the levels of activity of data is to track database activity against the data that you plan to archive. Accumulating this information for several months can give you a realistic estimate of the frequency of access to live data, helping you to make sound business decisions regarding data for archiving and access to it after it is archived.

### **Acceptable response times**

Requirements from user experts and business analysts also determine how quickly archived data must be accessed. Certain information must be available immediately or nearly so, while other data does not need be immediately available. For example, a call center manager might need immediate access to information about previous calls when handling a customer complaint on the telephone. Alternatively, customer service personnel in a distribution center can accumulate customer requests for order history to be processed weekly, in batch. Other situations will generally fall between these extremes.

### **Volume of data accessed**

The volume of archived data can be an important factor when designing your archiving system. Searching large volumes of archived data presents special challenges to timely retrieval.

### **Mode of access**

In most cases, the best method for access to archived data is determined by the design of the production system, the quantity of data to be searched and retrieved, the purpose and expected frequency of access, and the acceptable response time. A batch process might be best for retrieving a large quantity of data that is not needed immediately. An application may require modification to check for the presence of archived data and to trigger a restoration when certain conditions are met. In some cases, an application may generate the criteria used to locate and restore archived data, although the restoration is completed offline.

Categorizing sets of data to be archived according to all of your business requirements for access simplifies your decisions regarding the best media for storing your archive files and the optimal process for retrieving it. Table 7-1 provides examples of information you might gather regarding your requirements for accessing archived data.

*Table 7-1 Information gathered to determine access requirements*

<b>Data and criteria</b>	<b>Retention period</b>	<b>Access criteria</b>	<b>Reason for access</b>	<b>Frequency of access</b>	<b>Acceptable response time</b>	<b>Mode of access</b>
Orders older than 2 years	12 years (Life expectancy of product)	Item number	Recall activity	Almost never	2 days	Batch creation of temporary database
		Order ID or Date	Customer query	3 – 5 per month	Immediate	Browse or generate report
		Customer name or ID	Customer query	3 – 5 per month	Immediate	Browse or generate report
Invoices older than 1 year	1 year	Customer name or ID	Customer query	80 – 120 per year	2 days	Browse, generate report, or restore for re-invoice
Invoices older than 2 years	6 years	Billing date	Audit	Almost never	1 week or more	Batch creation of temporary database
		Customer name or ID	Re-establish account	3 – 5 per month	5 days	Restore payment history
Service Orders older than 1 year	Indefinite	Order number	Planning	Annually	1 month or more	Generate report
		Customer ID	Customer complaint	Almost never	1 week	Generate report

After this information is gathered, you have several options for presenting the data. Next, we discuss these methods, including ad hoc viewing, report generation, federated viewing, and native application access and other viewing from within an application.

### 7.1.1 Ad hoc viewing

For infrequent, unrepeatable, and unpredictable instances that require access to a small amount of archived data, a manual browse of archived data or small targeted restorations might be the only option. For example, to call a customer and ask a question, a customer service representative might need to locate the segment of data that relates to a specific order. By examining the related customer information, the representative can obtain the telephone number. This browse capacity can provide a means of satisfying simple *ad hoc* queries, or it can be used in more complicated situations, if needed.

The Optim Data Growth solutions support this type of access by providing a **browse** utility that allows you to review data stored in an archive file quickly and easily. For example, you can browse an archive file for these purposes:

- ▶ Satisfy a customer inquiry, without having to restore a single row to a database.
- ▶ Ensure that the archived data is as expected, before deleting any data from the database.
- ▶ Determine if an archive file contains appropriate or expected data before restoring it.

Using the **browse** utility, you can browse data in any table in an archive file and join to related rows in other tables in the same file, as if using a table editor in a relational database.

No special objects are required to search archived data, when browsing it. However, you must locate the proper files when searching for archived data and, having located the files, you want to search efficiently for specific data in them. Use file names, group designations, and descriptions to classify archive files according to application, data model, or other significant feature. Also, you can organize archive files into collections for access using Optim open data manager (ODM). You must consider these designations carefully and apply them systematically to give you maximum advantage in searching an accumulation of archive files. In addition to name, group, and description, the Optim Data Growth solutions allow you to search for files that were created within a range of dates, or that include data from a specific table or column.

After you narrow the search to a group or collection of candidate archive files, you can provide criteria to locate specific data within the files. You can apply the criteria directly to data in an archive file or to an index of values in one or more columns.

Generally, you can search the indexes more quickly than you can search an entire archive file. An archive file might have several indexes that are stored in an

associated file. When you browse or restore data, the Optim solutions determine if the use of an available index can expedite the search.

## 7.1.2 Reporting

You can choose among several alternatives to provide reports that meet your organizational business requirements:

- ▶ Using the browse facility in your Optim solution to locate and print a segment of related data from an archive file.
- ▶ Using the solution restore capability to create a staging database of selected data from one or more archive files. After it is created, you can process the staging database to create a report, using an application that is designed for the production database.
- ▶ Developing or modifying an application (for example, a custom application, Crystal Reports, Microsoft Access, or Microsoft Visual Basic) to use an application programming interface (API) or industry standard Java Database Connectivity (JDBC)/Object Database Connectivity (ODBC)-compliant access to locate and retrieve data in archive files for further processing and reporting.
- ▶ Generating a report from federated data (see 7.1.3, “Federated viewing” on page 333) or from within an application that has been configured for viewing (see 7.1.5, “Native application access” on page 339 or 7.1.6, “Other viewing from within an application” on page 341).

If a simple report showing a segment of archived data is needed, business users can rely on the browse facility that is described in 7.1.1, “Ad hoc viewing” on page 330. The browse facility, when coupled with the search capacity of the solutions, provides a way to locate and review a segment of related data in an archive file quickly and easily and print it in a simple report.

Note that using an InfoSphere Optim Data Growth Solution to browse archived information and obtain printed reports requires expertise that might be difficult to sustain at a business-user level. If users browse or otherwise access archived data infrequently, or if the browse interface differs markedly from that used in day-to-day transactions, business users might be challenged when searching for archived data. In addition, certain archived information might be confidential; for example, a business user that needs the name of the next-of-kin for a deceased former employee must not necessarily be privy to salary or stock options information. Also, your organization might need reports that are more comprehensive or complex than those reports that are provided using only the Optim solutions.

Restoration to a temporary database is perhaps the best solution if you are running one or more production applications against a large segment of archived data, for example, to apply a report application to archived data. However, it might be enough to convert archived data to a comma-separated value (CSV) or other format that can be imported into a commercial spreadsheet application. It is seldom advisable to restore data to the production database, especially if the restored data will be picked up by a subsequent archive process.

## Cognos reports

It is possible to configure your solution on distributed platforms so that Cognos report writers can access archived application data, such as JD Edwards, Oracle Enterprise Applications, or Siebel CRM, or custom applications. This approach might provide greater flexibility and save time and cost on the resources that are needed to restore the data to a database. Figure 7-1 illustrates an architecture that uses Cognos Reports with Optim archive files.

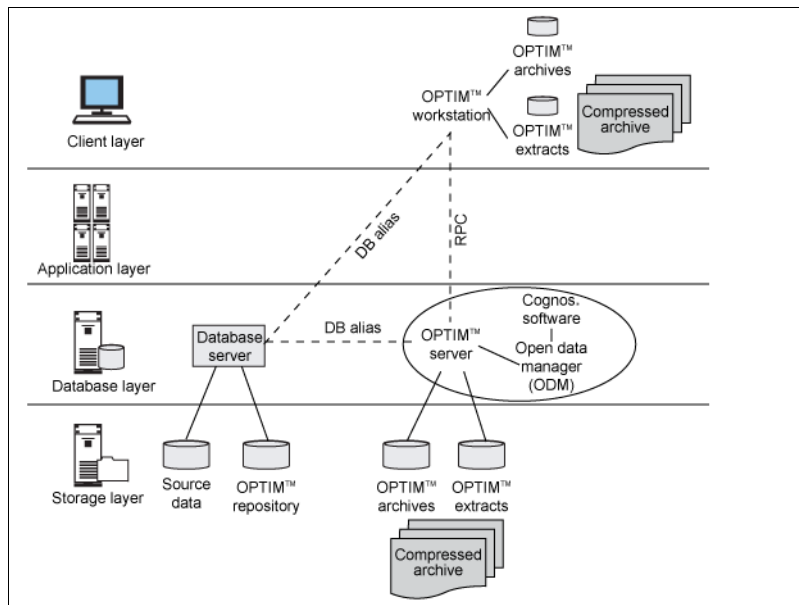


Figure 7-1 Architecture of Cognos reports with Optim archive files

The article, “Access Optim archive file data using Cognos”, in developerWorks®, describes the required steps to configure the Optim solution and Cognos 8 (including versions 8.1, 8.2, and 8.3) so that Cognos report writers can access archived application data.

You can read access this article at the following website:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-0905cognosoptim/index.htm>

### 7.1.3 Federated viewing

You can accomplish the viewing of active or production data that is federated with archived data in at least two ways. The first method trades greater disk space consumption for faster access in retrieving archived data and is perhaps the best choice when both the high frequency of access and faster speed of retrieval are required. In this case, you achieve federated access to archived data by restoring deleted data to a separate database or to separate tables in the production database. This way, users can access the archived data as needed but they do not interfere with performance during normal processing. As this near-line data ages and becomes less active, a second process is necessary to move it to cheaper and less easily accessible media.

The combined viewing of data in an archive file is also possible. This method does not require restoring the data. This method federates the archived data in files with active data using Optim ODM, which is a component of your data growth solution, along with IBM InfoSphere Federation Server with an ODBC wrapper, Microsoft SQL Server with a linked server configuration, or Oracle Database Gateway for ODBC.

#### **Database to database federation of archived data**

Federated access of active data with archived data that is restored to database tables requires a database view. A view is unlike an ordinary table in a relational database in that it does not form part of the physical schema. Instead, a view is a stored query that is accessible as a virtual table, which is composed of the result set of the query. For federated access, you use a view to UNION data from two data sources: the production tables and archive tables. In Figure 7-2 on page 334, View VB (Select \* from y.B Union all Select \* from B@ArchData) appends the data retrieved from B@ArchData to the set of data retrieved from y.B.

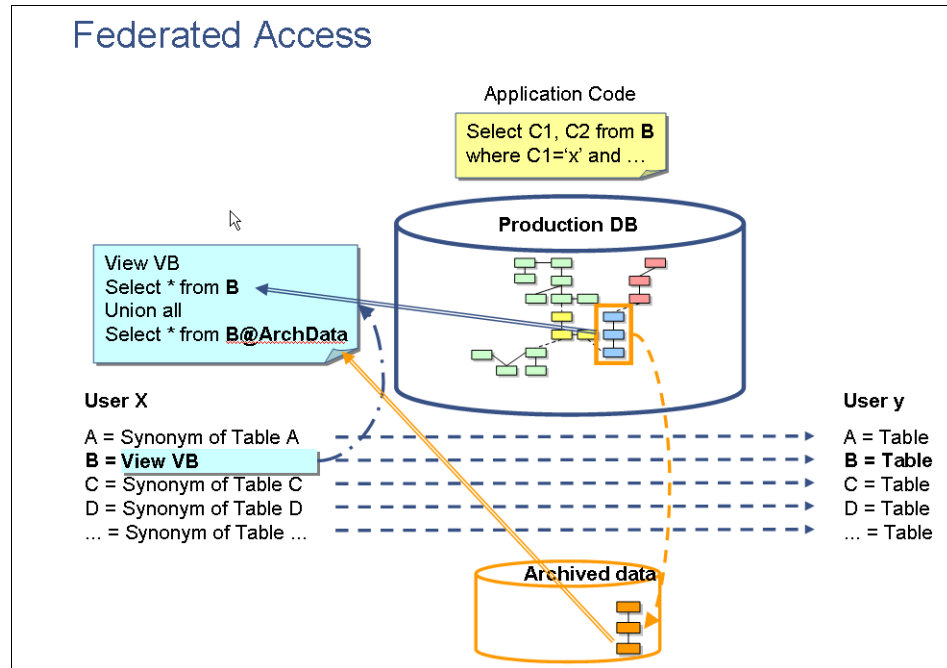


Figure 7-2 Database to database federation

Federated viewing requires synonyms for database objects, such as tables, views, sequences, and stored procedures. In Figure 7-2, User x does not have direct access to table y.B, which was created by User y. Defining B as a global synonym for table y.B makes it possible for User x to access the data in y.B without knowing where the data is or even that the data is archived. By substituting the matching synonym for the view, the application is either directed to a local table (data that is not archived) or to the view that UNIONS the primary data to the active archive data so that accessing that data is accomplished without changing the application.

Because, in federated viewing, the archived data is restored to a separate, rather than the same database, you are able to retain performance improvements in your production database that result from archiving. Depending on the class of database, service-level agreements (SLAs), recoverability, network, and disk performance, federated viewing offers good performance between databases that are managed by the same database management systems (DBMS).

A database view of the active data and the archived data provides federated viewing of the data. If users of a business application are able to access the federated view from within the application interface, viewing of this type is referred to as native application access (NAA). See 7.1.5, “Native application access” on page 339 for an additional discussion of federated access and NAA.

### **Combined views with archived data in a file**

When viewing and searching both active and archived data, it is sometimes inconvenient to first restore the archived data into database tables. Combined viewing of data in an archive file with active data uses certain mechanisms and techniques that treat ODM (see 7.2.1, “Open data manager” on page 343) and, by extension, data in archive files as part of the active database. Several mechanisms are available to accomplish this type of federation:

- ▶ IBM InfoSphere Federation Server with an ODBC wrapper
- ▶ Microsoft SQL Server with a linked server configuration
- ▶ Oracle Database Gateway for ODBC

When federating data from archive files, you can obtain the best performance by unioning the data with active data, using UNION ALL (UNION forces a costly sort to remove duplicates). If joining, you must ensure that all table references in the query can be satisfied by ODM, that is, that all tables are in the ODM collection.

When data in archive files is federated with active data, it is important that as much of the query (JOIN, WHERE, GROUP BY, and so forth) as possible is pushed down to ODM to avoid queries for excess data. Queries for excess data must be buffered and subjected to post-processing, and result in longer run times. Figure 7-3 on page 336 shows an example of this type of configuration.

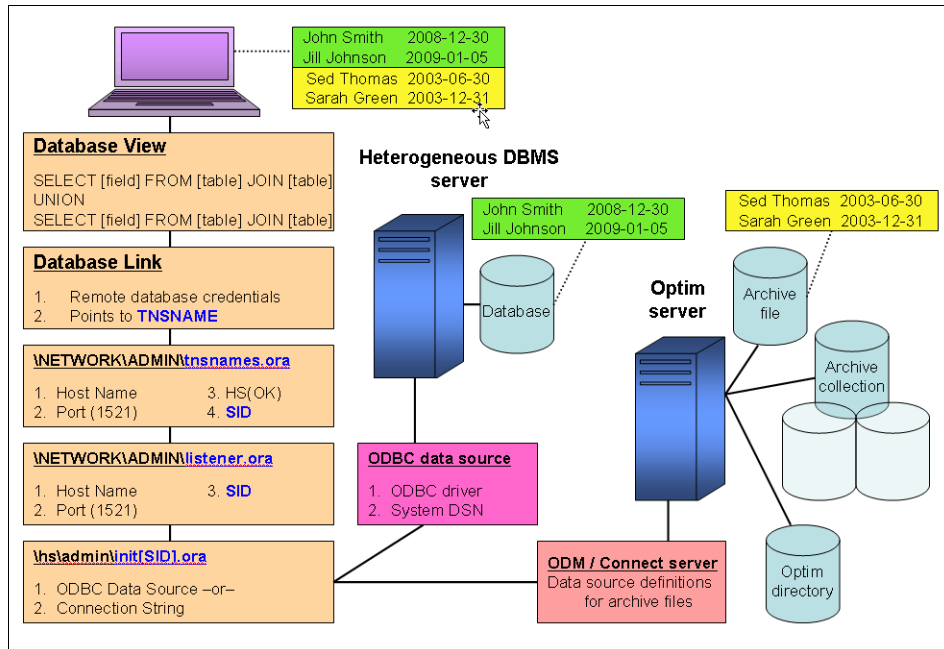


Figure 7-3 Combined view with archived data in files

## Mashups

*Mashups* present a method for viewing and reporting on archived data with active data that involves fetching both the online and the archived data from native repositories and combining them. A mashup lets you access active enterprise data and combine it with archived data when the need arises.

The developerWorks article, “Mash active and archived data using IBM Mashup Center and Optim,” shows how to extend a mashup application that was built to track active order information on DB2 to also include archived order information, as shown in Figure 7-4 on page 337. You can read the article at the website:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-1001mashuptoptim/index.html>

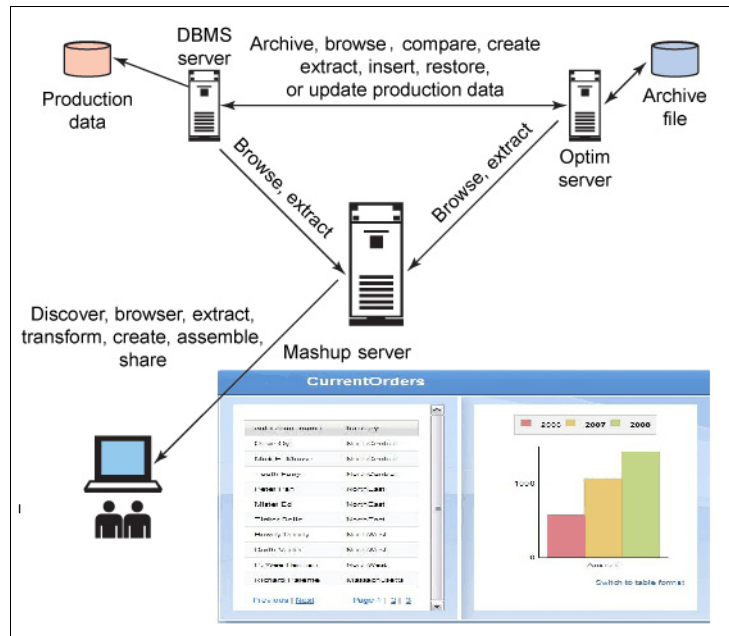


Figure 7-4 Mashup overview

## 7.1.4 Optim Data Find

Optim Data Find provides a rich set of query capabilities to support flexible and powerful searches across data in multiple Optim archives and collections. Data Find uses ODM to access archived data, as shown in Figure 7-5 on page 338.

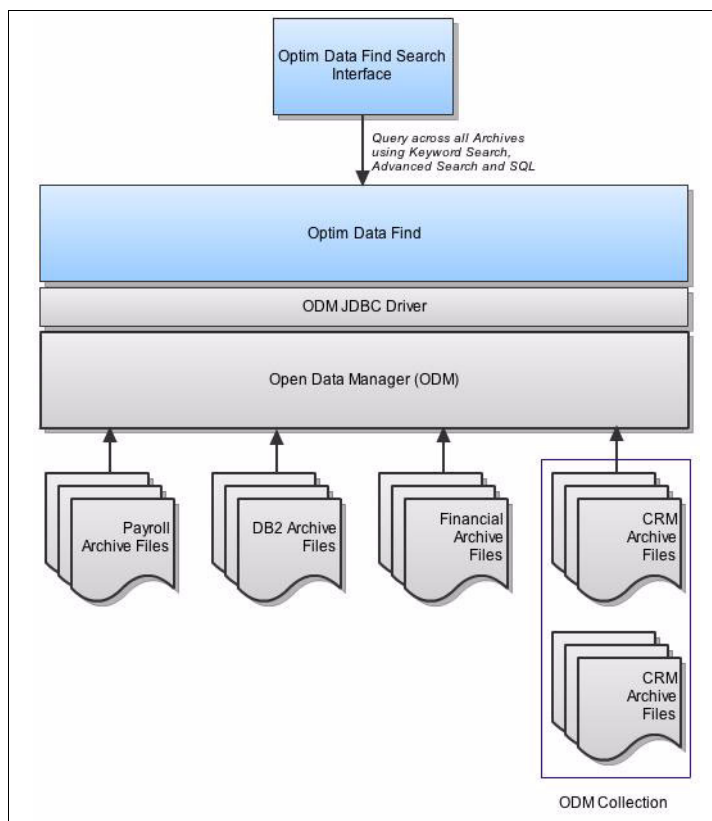


Figure 7-5 Optim Data Find environment

Using an array of options for finding and understanding data, you can search your archived data. The simple query language of Optim Data Find, which is illustrated in Figure 7-6 on page 339, is familiar to anyone who has used a web search engine. The Advanced Query Language of Optim Data Find provides expert users and automated systems with a more structured, functionally rich query syntax. One unique feature of the Advanced Query Language is that it supports a JOIN operator. This operator allows for SQL-style JOIN queries with the performance and fuzziness of full text search.

fowler

Search

Simple

Advanced

SQL

Advanced Options

Your search for **fowler** took 2,924 milliseconds.

Showing 1-10 of 29 results.

view results as list

<< first < previous 1 2 3 next

company.PRIVACYOPTIM\_COMPANY

#	SEQ	PST_AF_NAME	NAME	PST_ARCHIVE_ID
1	9159	C:\Optim\archive\company.AF	RALPH FOWLER DDS	24
2	5020	C:\Optim\archive\company.AF	FOWLER APTS	24

names.PRIVACYOPTIM\_UK\_PERSON

#	COMPANY	SEX	LASTNAME	NATIONALID	PST_ARCHIVE_ID	FIRSTNAME	PST_AF_NAME	ID	BIRTHDATE	EMAIL_ADDRESS	PHONE	SEQ
3		m	Fowler		26	Riley	C:\Optim\names.AF		Thu Sep 09 00:00:00 EDT 1965	Riley.F.Fowler@pookmail.com	070 2197 8566	8467
4		f	Fowler		26	Ruby	C:\Optim\names.AF		Thu Nov 16 00:00:00 EST 1950	Ruby.D.Fowler@spambob.com	078 6514 3805	8584
5		m	Fowler		26	Owen	C:\Optim\names.AF		Sun Dec 16 00:00:00 EST 1962	Owen.L.Fowler@pookmail.com	079 6301 9133	8044
6		f	Fowler		26	Mollie	C:\Optim\names.AF		Sun Jul 24 00:00:00 EDT 1949	Mollie.A.Fowler@mailinator.com	078 3413 6926	7378
7		m	Fowler		26	Luca	C:\Optim\names.AF		Sat May 27 00:00:00 EDT 1967	Luca.P.Fowler@pookmail.com	077 6025 7308	6438
8		f	Fowler		26	Nicole	C:\Optim\names.AF		Sun Nov 07	Nicole.L.Fowler@mailinator.com	078 5105	7803

Figure 7-6 Simple Data Find query

## 7.1.5 Native application access

Displays of archived data that is federated with active data are sometimes desirable if the application users need to view archived data in the course of their daily tasks. When federated viewing is provided from within the application interface, it is referred to as *native application access* (NAA). For performance reasons, it is better generally for the application to manage NAA than the relational database management system. You implement NAA by creating a shadow schema in the production database and an archive schema in a separate archive database.

The shadow schema is primarily a pseudo copy of the production schema that holds synonyms for objects in the production database. For tables from which data is deleted, however, the shadow schema contains union views. Also, any objects in the production database that refer to tables from which data is deleted are re-created in the shadow schema as reference objects or tables.

Certain objects in the production schema must be replicated in the shadow schema even though they are not related to the deleted data or seemingly have nothing to do with objects that are related to the deleted data. An example of this type of exception object is one or more temporary tables that are used by the application to generate a report. Because it is likely that the same report will be run on the archived data, it is important that the shadow schema retain the ability to manage this temporary information. Re-creating the temporary tables in the shadow schema permits the report to be run.

The archive schema replicates, in name and structure, the production tables and indexes from which data is deleted in the archive process. For NAA, the application references the shadow schema rather than the production schema to provide user access (typically read-only access) to both archived and production data through the union views.

After you set up NAA and all objects are in place, your NAA configuration will function seamlessly. The setup includes these objects and steps:

- ▶ Establishing user accounts and schemas
- ▶ Installing the system procedures and objects for NAA
- ▶ Replicating the production schema by creating synonyms in the shadow schema
- ▶ Copying exception objects
- ▶ Performing all necessary grants and permissions

Then, you must create the tables and indexes for the business object in the archive schema, create the union views in the shadow schema, create reference objects, perform exceptions, and set the appropriate grants and permissions:

- ▶ For each production table from which data is deleted, create a table in the archive schema (use the link to production to copy the table structure).
- ▶ Replicate production indexes in the archive schema.
- ▶ Drop synonyms for the production tables from which data is deleted into the shadow schema.
- ▶ In the shadow schema, create union views that point to the production tables from which data is deleted and the corresponding archive table, giving the views the same name as the production tables.
- ▶ Identify all reference objects (that is, objects that refer to, use, or query the delete tables) and for each reference object, drop the synonym in the shadow schema that points to the reference object.
- ▶ Re-create the reference object in the shadow schema, giving it the same name as the production table.

## 7.1.6 Other viewing from within an application

In addition to native application access, a configuration, which uses the business application interface to provide a view of archived and active data to business application users, is a type of guided viewing provided to business users from within the interface of an application. For this type of viewing, business users are presented with a summary of archived information and are given the means to view the actual data in the archive file, if circumstances call for it.

Providing guided viewing of archived data from within an application interface relies upon a “look aside” or summary table as an intermediary between the application and the archive file. During archive processing, an archive action populates a row in the table with summary information about each business object that is archived. Included in the summary row is identifying information, including object IDs for related information and the name and globally unique identifier (GUID) for the archive file in which the transaction is archived.

Another archive action adds a flag to the undeleted, non-transactional data for the archived object that remains in the production database after the transactional data is deleted.

Guided viewing requires modifications to the application interface to display the archived data or generate a report from it. In addition, a web service that is distributed with your Optim solution is required to retrieve data from ODM and return it to the calling client, which is an integration object on the application side. Figure 7-7 on page 342 shows a guided viewing architecture in a Siebel CRM implementation.

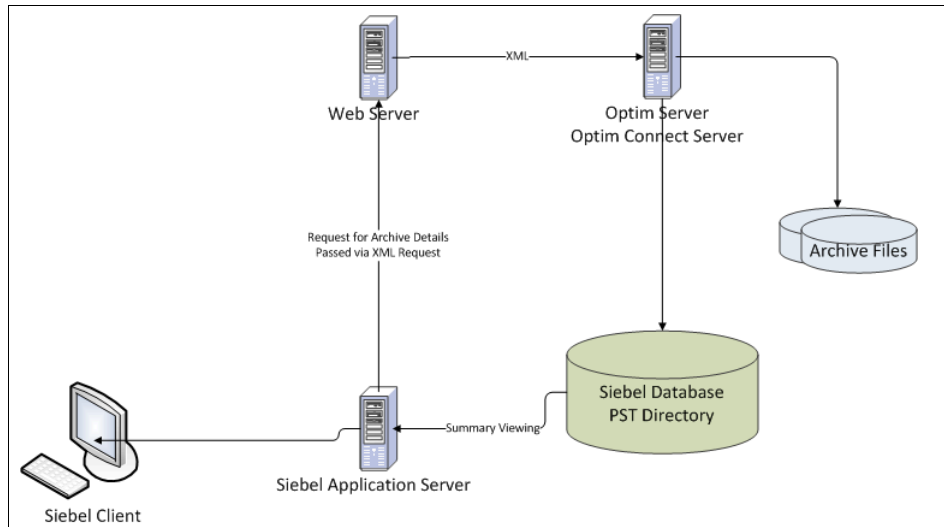


Figure 7-7 Guided viewing architecture

Performance in retrieving data from an archive file is generally lower than in retrieving data from a database. At the same time, using the summary table in the database avoids overhead from searching for the appropriate archive file. Also, the summary table has a smaller footprint than the archived data if restored to database tables as required for federated viewing and NAA.

For example, assume that a business archives all orders older than two years, using its InfoSphere Optim Data Growth Solution. During the archive process, the complete business object, including context such as customer name and address with the old order information, is copied to the archive file while only the order information is deleted from the original database. During this processing, an archive action populates the look aside table with summary information and a second archive action flags the context information in the database to alert the application that orders are archived.

If a business user uses the application interface to access the customer information in the production database, a notice indicates that orders for the customer have been archived and, from the look aside table, provides summary information about the archived orders. If the business user desires additional information, the application also can reference the summary table for information that is needed to identify the archived orders and, using Optim ODM, retrieve it for restoration or more detailed viewing of the data.

## 7.2 Managing access to archived data

To help manage access to archive files, your InfoSphere Optim Data Growth Solution includes ODM, as well as an ODBC driver. The following discussion provides information about both methods.

### 7.2.1 Open data manager

Optim ODM brings a standards-based interface, ANSI SQL-92, to access data in archive files for programs that use ODBC or JDBC APIs. The ODM supports the use of tools, such as Microsoft Excel and DBVisualizer with archived data. It also facilitates the federation of archived data with relational databases using IBM InfoSphere Federation Server with an ODBC wrapper, Microsoft SQL Server with a linked server configuration, and Oracle Database Gateway for ODBC. You implement the ODM by using Optim Connect, a rich, peer-to-peer networking component of the data growth solutions, in concert with a custom driver that provides access to archive files and archive file collections.

To be accessed using ODM, you must register the archive files and they must be accessible from the ODM server on which the data source is defined. An ODM server resides on an Optim server with one or more ODM data source definitions for archive files or archive file collections.

#### **Archive file collections**

A collection of archive files can be unioned together logically as a single data source for ODM access. The ODM uses a collection to obtain access to data in a group of archive files, which might not have the same schema. For example, all files in the group might have a similar schema, but certain files do not include a specific table or column, or the attributes of data in a column vary from file to file. The ODM processes archive files in the order that is listed in the Archive File Collection Editor.

#### ***Unioned tables***

Tables with matching creator IDs and names are unioned across archive files without regard to DB alias; however, an archive file cannot have two tables with matching creator IDs and names but separate DBAliases. The union includes all columns in all matching tables. A table that does not include a column that is found in matching tables in the collection is considered to have a default value such as NULL, a default date provided in the Archive File Collection Editor, or an appropriate value for the data type, such as spaces or zero.

Each unioned or composite table includes a pseudo-column that records the archive file identifier. In addition, a pseudo-table contains a row for each file in the

collection that records the archive file identifier and GUID, file and group names, and the timestamp of the archive process generating the file.

**Column compatibility**

All columns with the same name that are in matching tables must have compatible attributes. If columns have separate but compatible attributes, a compatible attribute will be used for those columns. The column compatibility rules for the Optim compare process apply to archive file collections. For information about comparison compatibility rules, refer to the *IBM Common Elements Manual* for your Optim solution.

If the size of a column in a table varies by file because of differing, but compatible, data types (for example, char, varchar, binary, and numeric), the size in the composite table will accommodate all types. If a column occurs in certain files but not others, it is included in the composite table with default values supplied for rows that do not include the column.

For example, columns COLX DECIMAL(8,2) and COLX DECIMAL(10,0) are searched as COLX DECIMAL(10,2).

**Unioned indexes**

Indexes for tables in the collection are also generally unioned if each archive file that includes the table has the same index. A composite index is built for each table that has the same indexes in each archive file (only the set of columns, their data type, and order must match exactly; the index name is immaterial).

**Collection example**

Consider the hypothetical collection that is represented in Table 7-2, noting the difference in schemas for files in the collection.

Table 7-2 Collection example

Archive files	Tables in file		
Arch1.af	Cid1.Tbl1(10rows)	Cid2.Tbl2	N/A
Arch2.af	N/A	Cid2.Tbl2(drop col)	Cid3.Tbl3
Arch3.af	Cid1.Tbl1(30rows)	Cid2.Tbl2(add col)	Cid3.Tbl3(alter col)
Arch4.af	Cid1.Tbl1(20rows)	Cid2.Tbl2	N/A

The resulting collection content is a composite of tables from separate archive files, as shown in Table 7-3.

*Table 7-3 Resulting collection content*

Col_id.Col_name	Cid1.Table1(60 rows)	Cid2.Table2(all columns)	Cid3.Table3 (new column)
-----------------	----------------------	--------------------------	--------------------------

## 7.2.2 ODBC drivers for Optim solutions on Microsoft Windows

The Optim solutions in a Microsoft Windows environment have an ODBC driver that allows you to access data in an archive file programmatically. Using the ODBC interface, you can integrate your archiving strategy into your production environment, including the ability to browse and restore archived data, as necessary. You can process archive files that are registered in an accessible archive directory by using ODBC. As you develop your archiving strategy, it is important to consider these key points:

- ▶ The ODBC interface connects to, and can retrieve data from, a single archive file at a time.
- ▶ A well-conceived set of archive indexes greatly improves performance when retrieving archived data using ODBC. An indexing scheme needs to anticipate and support searches that will occur with any regularity. Optim accommodates an evolving system; as search requirements change, you can add, update, or delete indexes to meet current search requirements.

To use the ODBC interface, you must define a data source to identify the archive file that you will access. Using the Microsoft Windows ODBC Data Source Administrator to create a data source, you enter specifications on the ODBC Source Data Default Specification dialog. Criteria provide the necessary information to select an archive file as identified by group, server, or creator. Also, you can identify an archive file by using more specific criteria (for example, the file creation date, or the name of an archived table that is included in the archive file) by referencing a parameter file. Also, you can use connection string parameters in conjunction with a data source definition, depending on the application interface.

You might define several data sources for the archive file driver. For example, you can create a data source definition to use for creating reports, saving it with all the required information to create the desired report, except the specific identifier for the archive file, which is provided programmatically at run time.

At the same time, you might create another data source definition to provide supplementary information to the user of an application during daily transactions. The parameters that are saved for this type of data source are less specific,

relying on connection call parameters or parameter file overrides to retrieve the desired data at run time.

You can report on retrieved rows using your application software, or commercial software, including, but not limited to, Business Objects Crystal Reports, Microsoft Access, and Microsoft Visual Basic.

## **7.3 Performance considerations**

You must give careful thought to your strategy for accessing archived data to achieve the optimum balance among retrieval performance, business user need, and security.

### **7.3.1 Searching archives**

The Optim Data Growth solutions support several features that you can use to facilitate searches of your archive files. A robust and well-considered indexing strategy, as well as judicious use of file organization elements and globally unique identifiers, can help you find archived data quickly and efficiently.

#### **Indexing**

As with other elements in the Optim Data Growth solutions, you must carefully consider your indexing strategy to achieve maximum benefit and efficiency in archive file searches. The indexing scheme must anticipate and support searches that will occur with any regularity. At the same time, the Optim solutions accommodate evolving search requirements. As search requirements change, you can add, update, delete indexes to meet your current search requirements. By judicious selection of the columns for indexing, you ensure that archived data can be located efficiently and accurately for browsing or restoring.

A systematic approach to indexing archive files enhances the ability to locate the appropriate archive files to browse or to restore archived data. In combination with well-designed naming conventions for archive files, well-planned indexes allow you to eliminate files quickly for which a search is unproductive. Searching an index is usually more efficient than searching the file. For this reason, it is important to research carefully the selection of columns for indexing to provide the maximum leverage for searches of archived data. Note, however, that if a previously unforeseen need for additional indexes occurs, you can add or update indexes for previously archived data. You can also delete unnecessary indexes at any time.

### ***Indexing parameters for Optim Data Growth solutions***

Business use cases might require several indexes of archived data. For example, many queries that are received by personnel in the Customer Service Department require them to look for data that is based on customer name and order number or date of transaction. Another, substantial portion of customers that query Customer Service provide incomplete information – a customer knows the approximate time of the order and can describe the item ordered but does not remember the order number or item identifier. Also, in the event of a product recall, the Legal Department and Manufacturing must be able to locate all customers that ordered a specific item. Access requirements, which are similar to those requirements in Table 7-1 on page 329, are the basis for selecting columns for indexing.

Based on these requirements, you can decide to define several indexes for archived Orders data:

- ▶ An index on values in Customers.Custname for searches based on the customer name
- ▶ An index on values in Orders.Order\_ID for searches based on the order number and an index on values in Orders.Order\_Date for searches based on the date of the order
- ▶ An index on values in Items.Item\_ID for searches based on the item identifier

Indexes for archive files are generally on disk, although the archive files can be on disk, tape, or optical devices. Typically, all index files that are needed for a search are accessible at one time and can be searched without regard to storage media or the direct accessibility characteristics of the associated archive files. For the solution on distributed platforms, you can transfer archive files to tape (using external software) and use an index-only search to determine the appropriate tape for external transfer back to disk. To ready the transferred file for use, you must use the **registration** utility in the Optim solution to create a new archive entry for the file and the archive directory maintenance utility to create a new index file. The original archive entry and index file are discarded.

You must give careful consideration to indexing to achieve the maximum benefit and efficiency in archive file searches. To provide the maximum leverage for searches of archive files, the indexing scheme must anticipate and support searches that will occur with any regularity. The Optim Data Growth solutions accommodate an evolving development of search requirements. As search requirements change, you can add, update, or delete indexes to meet current search requirements.

## Search elements

When browsing or restoring data, searching archived data does not require any special objects. However, you must locate the proper archive files when searching for archived data and, having located the files, you want to search efficiently for specific data in them. You can use file names, group designations, descriptions, and collections to classify archive files according to application, data model, or another significant feature. You must carefully consider these designations and systematically apply them to give you maximum advantage in searching an accumulation of archive files. In addition to name, group, and description, Optim Data Growth solutions allow you to search for archive files that were created within a range of dates, or that include data from a specific table or column.

After you narrow the search to a group of candidate archive files, you can provide criteria to locate specific data within the files. You can apply the criteria directly to data in an archive file or to an index of values in one or more columns. Indexes for each archive file are stored in an associated index file and, generally, the Optim solution can search the indexes more quickly than it can search an entire archive file. When you browse or restore data, the Optim solution determines whether it can use an index to expedite the search.

### ***Archive directory entries***

To access archived data, you must be able to find it. The Optim Data Growth solutions provide many features that allow you to organize archived information so that you can locate it efficiently. With careful planning, you can establish a purposeful system of organization for archive files and implement procedures to maintain this organization. When combined with well-conceived indexing procedures, this planning lays the groundwork for the automated or manual searches of archive files, regardless of the media on which the files are stored. After the desired segment of archived data is located, you can browse or restore it without being hampered by files or rows in which you have no interest.

The Optim Data Growth solutions create an entry in the archive directory for each archive file. This directory entry identifies the file by name, group, description, and the tables from which data in the file is archived. The entry also records the user account initiating the archive process and the date that the process is executed. Using one or more filters or patterns for this information, you can search for specific segments of data in two or three steps:

1. Search the archive directory to identify the archive files that are most likely to contain the specific data in which you are interested.
2. Search the index files that are associated with the archive files that were identified in the first step to eliminate the files that cannot contain the data in which you are interested.

3. Search the archive files remaining from the second step to locate the precise data in which you are interested.

### **File organization example**

As an example of the use of file organization features in the data growth solutions, consider a company that archives data for three applications: the order entry application, the payroll application, and the inventory application. Archive files from the three applications are maintained in common storage and all archive directory entries are in the same database.

#### ***Archive file name***

In this example, the archive file name is prefixed with a 2-character identifier for the application that pertains to the data in the file, that is, OD, PY, and IV for the Order Entry, Payroll, and Inventory applications:

- ▶ Because orders more than two years old are archived at the end of each month, a two-digit number (01 to 12) represents both the current month and the month in which the archived orders were shipped. A four-digit suffix represents the year in which the archived orders were shipped. Thus, the unique name of each archive file for the Order Entry application is in the form ODmmYYYY.
- ▶ Payroll records more than seven years old are archived every year. The unique name for each archive file containing payroll data is in the form PYYYYY.
- ▶ Inventory records more than seven years old are also archived yearly. The unique name for each of those archive files is in the form IVYYYY.

For both the Payroll and Inventory applications, YYYY is the year to which the data applies, not the year that it is archived.

**File names:** For automated processes, the file names can be generated automatically.

#### ***Group designation***

An application evolves to support and accommodate changing business needs, and this evolution might entail changes to the data model. For this reason, it might be helpful to track the version of the application that applies when data is archived. In this example, the company uses the optional archive file group designation to record the name of the application and its version number.

#### ***Description***

Optional description values can provide a third level of granularity in the organization of archive files. The company in this example creates order entry

archives for three geographical areas and retains a composite archive file for disaster recovery. Archive files for each area are identified by a value in the Description field, such as “East”, “South”, and “West”, while the description for the composite is “US”.

Depending on the information that is available at the time of the search, this company can use criteria for archive file name, group, and description, with the appropriate wildcard characters, to locate the archive files that are most likely to contain archived data of interest. After the search is narrowed to the fewest possible archive files, search criteria are applied to associated index files or to data in the files.

### **Globally unique identifiers**

The Optim Data Growth solutions assign a globally unique identifier (GUID) to each archive file as it is created. Unlike the archive file ID, this identifier never changes. The GUID is recorded in both the archive file and the archive directory entry. By using an archive action statement, you also can save the GUID externally (perhaps to a look aside table) when the archive file is created. In an automated process, a customized application can supply the externally saved GUID to retrieve archived data.

### **Alternative to Optim indexes and use of other search elements**

Certain search tools can provide an alternative to the indexing features of the Optim Data Growth solutions by providing a flexible and powerful high-speed method to query data across multiple archive files and collections from a search interface. After the results are obtained, they are displayed. A good tool for this purpose supports search engine-like keyword searches, as well as a more advanced keyword language and an extended version of SQL.

Figure 7-8 on page 351 shows the architecture for using a search tool with your Optim solution.

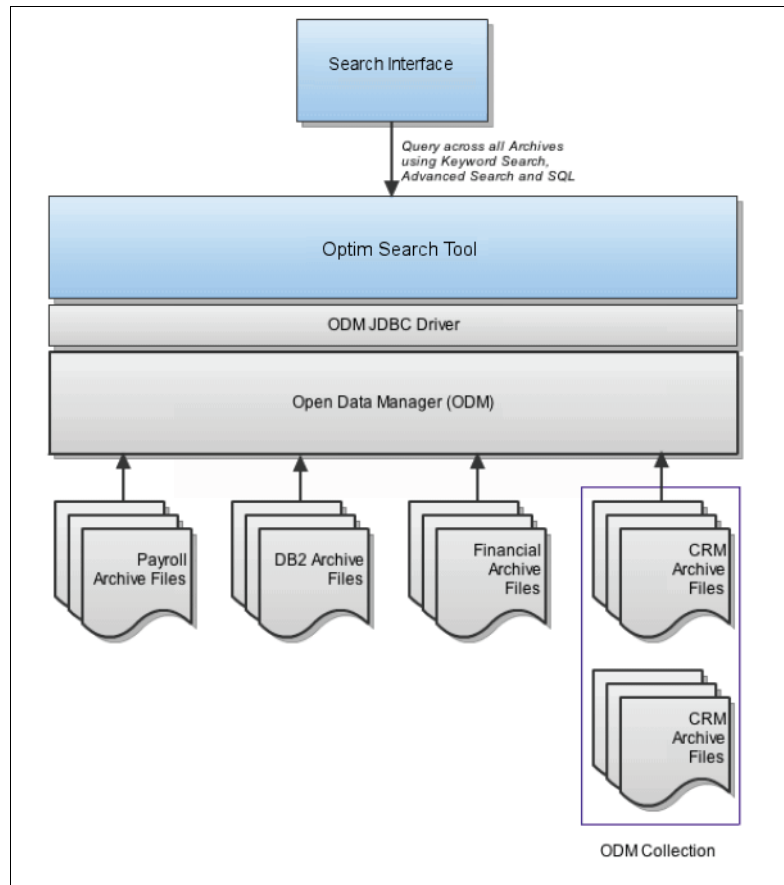


Figure 7-8 Architecture of search tool with Optim





## Design and configuration

In this chapter, we cover the major considerations for implementing an IBM InfoSphere Optim Data Growth Solution (Optim). We describe and illustrate two complete Optim Data Growth designs. We provide a clear picture of a multi-data source solution and illustrate several of the many available options for effective data archival, deletion, and access. We also emphasize the importance of completing all prerequisite tasks before beginning a design and configuration.

## 8.1 Design considerations

There are numerous things to consider in the design of an archive solution:

- ▶ Amount of *initial* data for archival
- ▶ Time frame for archiving *initial* data
- ▶ Amount of data archived for ongoing maintenance
- ▶ Frequency of ongoing maintenance archives
- ▶ Inclusion of non-relational data
- ▶ Service-level agreements (SLA) for data archival and retrieval
- ▶ Frequency and volume of data retrieval
- ▶ Number of concurrent retrieval requests
- ▶ Anticipated increases or decreases in data volume

These considerations affect both architecture and sizing. Therefore, it is extremely important to develop clear and universally understood business requirements so that an appropriate architecture is created.

### 8.1.1 Archiving strategy

A key consideration is whether to archive data by *complete business object* or by table. Archiving by complete business object requires a clear understanding of the relationships between tables. Rows are archived with all of their related rows from all related tables and grouped in archive files by date range. The initial effort is more labor-intensive than simply archiving all of the rows in tables that meet a specified criteria. However, what might seem to be an unnecessary effort in the beginning clearly becomes beneficial when the data in those tables is accessed or is ready for deletion.

If data is archived by table without regard to relationships, the probability of missing rows in tables and creating orphan rows is extremely high. Additionally, Data deletion is complicated when data is not stored in complete business objects.

We strongly suggest that data is archived by complete business object and grouped by expiration date. Using this method allows for the deletion of expired data by simply deleting individual archive files after their expiration date has passed.

## 8.1.2 Selecting what data to archive

The success of an archive design and configuration is highly dependent on clearly understood and achievable business requirements. You need to perform analysis against data that is a candidate for archival to validate that the strategy will indeed satisfy those requirements.

If the actual usage patterns of candidate archive data are not fully understood, the end results might be undesirable. For example, if large volumes of data are archived that are still frequently used by their associated applications, the level of archive file access demand might exceed the capabilities of the design.

## 8.1.3 Restoring archived data

The Optim Data Growth Solution's *restore* function can restore any data that was archived and deleted back to its original location or to a separate location. Restoring data to a separate location is a viable method of accessing archived data, because the original application can be utilized. However, you must not use the restore function to restore data back to its original production database unless it was removed in error. Doing so creates a situation where the same data can be archived more than once and exist on more than one archive file, which can cause confusion. Which archive version is correct? Possible answers are that they are all the same, or they all differ and all are correct. We suggest that a simple solution is best: data is archived only once; deleted; and accessed by queries to the archive files or restored to a reporting database. If data is ever restored back to production and archived multiple times, it might be necessary to incorporate a method to track the archival history. The capability to track the archival history ensures that business questions are answered using the correct data and questions from auditors about multiple archives of the same data are satisfied.

## 8.1.4 Making the final design decision

Before final design decisions are made, we strongly suggest that you prove that the design works in the actual environment where it will be implemented. You need to create and execute a limited archive process. This approach is an excellent method of uncovering issues early and at a point where you can correct the issues more easily. Running a limited archive process provides the opportunity to measure run times, prove that data throughput is as expected, and validate that the data is correct. This same philosophy is also true for validating the process for accessing data that is archived.

## 8.2 A sample federated design

In this section, we present a design sample to illustrate the thought process of designing an Optim Data Growth solution.

The sample environment contains 1.81 terabytes (TB) of data stored on three separate operating system platforms, two database types, and two non-relational files (VSAM files on z/OS). Appendix A, “Data model” on page 501 contains the data model for this example. We describe the requirements, prerequisites, and considerations that are incorporated into the sample design.

### 8.2.1 Sample business requirements

The first task in designing an InfoSphere Optim Data Growth Solution is to understand the business requirements. Here, we list specific information about the application and tables from which data is archived. The information includes the processing window, table names, archiving rules, access requirements, disk storage rules, and other pertinent information.

#### **Processing window**

The available time for archive and delete processes operating against production databases is a six-hour window that is available on Monday mornings between 12:00 am and 6:00 am. This period might not be available on month-end weekends and when system maintenance is scheduled. The archive process for any given month can be scheduled in any available time slot. The initial archive of the Orders and Details tables is completed within 30 calendar days.

#### **Orders and Details tables’ archiving rules and considerations**

The following requirements are necessary for processing the Orders and Details tables:

- ▶ The primary reason for archiving and deleting data in these tables is to conserve space and improve performance. Compliance and cost are secondary.
- ▶ The Orders table contains 8.6 billion rows and the Details table contains 20.6 billion rows.
- ▶ All data from the Orders table and the related records in the Details table that are three years old and older are archived and deleted from the DB2 database. This data constitutes 21% of the rows in the Orders table and reduces the total size of those two tables by approximately 388 GB.
- ▶ The data is archived by month beginning with the oldest data and proceeding to the most recent data.

- ▶ The data retention period is seven years.

### **Customers tables' archiving rules and considerations**

The following requirements are necessary for processing the Customers tables:

- ▶ The primary reason for archiving and deleting data in these tables is for audit and compliance. Performance and cost are secondary.
- ▶ The Customers table in DB2 contains 25.3 million rows. The Customers table in Oracle contains 11.3 million rows.
- ▶ Any customers in the Customers tables who have not placed an order in more than three years are archived and deleted from both the DB2 and Oracle databases. This data constitutes 12% of the rows in the Customers tables in both databases and reduces the size of those Customers tables by approximately 1 GB.
- ▶ Before deleting rows from the Customers tables (DB2 and Oracle), it must be confirmed that no children rows exist in the Orders tables in their respective databases.
- ▶ The data is archived by year beginning with the oldest data and proceeding to the most recent data.
- ▶ The data retention period is seven years.

### **Sales tables' archiving rules and considerations**

The following requirements are necessary for processing the Sales tables:

- ▶ The primary reason for archiving and deleting data in these tables and files is for audit and compliance. Performance and cost are secondary.
- ▶ The Sales table in DB2 contains 251,000 rows. The Sales table in Oracle contains 95,000 rows.
- ▶ Any salespersons who have not recruited any new customers in more than two years are archived and deleted from their associated database. This data constitutes 5% of the rows in the Sales tables in both databases and will reduce the size of those Sales tables by 0.01 GB.
- ▶ Each salesperson's rate history and state information is managed by an application on z/OS. The archive files from the Sales tables in both DB2 and Oracle also contain the related rows in the Rates and State\_Lkup VSAM files on the z/OS system. Combined, these files contain one million rows of which 50,000 rows are archived. Space is reduced by 0.01 GB from a total file size of 0.2 GB.
- ▶ Before deleting rows from the Sales tables (DB2 and Oracle), it must be confirmed that no children rows exist in the Customers tables in their respective databases.

- ▶ Related rows are also deleted from the Rates and State\_Lkup VSAM files on the z/OS system.
- ▶ The data is archived by year beginning with the oldest data and proceeding to the most recent data.
- ▶ The data retention period is seven years.

### **Archive file access requirements**

Data restore requests to the reporting database are expected to complete within 24 hours. Direct queries to archive files are expected to complete within three minutes for data that is stored on disk. It is expected that there are approximately 75 concurrent users accessing archived data at any given moment.

### **Storage media requirements**

All archives that have been created are kept on fast-access disk storage for a period of one year from the date of creation. After that period, they are moved to a tape backup server where they reside until expiration. Any archive files that have been restored from the tape backup system will remain on disk for seven days and are then deleted from disk.

## **8.2.2 Sample design**

The number of parallel processes is determined by the size of the processing window and the amount of data that must be archived within that window as dictated by the business requirements. An estimate of the number of parallel processes is made during architecture and sizing so that the necessary hardware that is required to support the expected number of processes is available. Additional processes can be added or removed as required. In our example, we decide to use up to four parallel archive processes on two Optim servers.

This design ensures that an alignment exists between the grouping of data in each archive file and the retention period of that data. In this example, data that is archived monthly from the Orders and Details tables aligns with the requirement to delete that data after it is seven years old. Likewise, the yearly archival of data from the Sales and Customers tables also aligns with annual deletion after seven years of retention.

The amount of data in each archive file is distributed in the following manner:

- ▶ Each of the four monthly archive files for the Orders and Details tables contains approximately 43 million rows and 3 GB of data (uncompressed and exclusive of indexes).
- ▶ Each of the archive files for the Customers tables contains approximately 1.5 million rows and 0.3 GB of data (uncompressed and exclusive of indexes).
- ▶ Each of the archive files for the Sales tables contains approximately 5,767 rows and 0.003 GB of data (uncompressed and exclusive of indexes).

Four archive processes running together for the Orders and Details tables provide enough parallelism to archive the volume of data that is defined by the business requirements within the six-hour window. This archive includes the requirement to process the initial archive (21% of the data in the Orders and Details tables) within 30 calendar days. This example assumes that the hardware infrastructure can support the transfer of approximately 388 GB in about 4.5 hours. Having this 1.5 hour buffer of time between the expected execution time and the processing window SLA allows for unforeseen operational delays and unexpected process failures.

All processes are run in batch and are controlled by the client's existing enterprise scheduling system. This approach is the typical method for running archive processes after they are tested and ready for production transition.

The data is archived by complete business object. Referential Integrity (RI) is not enforced by any of the databases. The relationships between tables are known only by the application. Therefore, it is necessary to build Optim relationships between the tables for the Optim archive processes to access the data.

The archiving of the Customers tables is performed by selecting data from both the DB2 and Oracle database instances. The design includes verification that any selected customer rows do not have matching keys in the related Orders table. If a matching key exists in the Orders table, that customer row is not archived. The volume of data for the Customers tables does not indicate the need for more than a single Optim archive process.

To prevent a situation where a client is actually inactive, but the client is not archived because there are entries for that client in the Orders table, all archiving for the Orders and Details tables must complete before archiving any data from the Customers tables.

The Sales tables are archived from both DB2 and Oracle along with records that have matching keys in the Rates and State\_Lkup VSAM files on z/OS. The design includes verification that any salesperson rows that are selected do not have matching keys in the related Customers table. If a matching key exists in the

Customers table, that salesperson is not archived. DB2 Federation Server and Optim Connect are used to access the VSAM files. The data volume for the Sales-related tables does not indicate the need for more than a single Optim archive process.

To prevent a situation where a salesperson is actually inactive, but the salesperson is not archived because there are entries for that salesperson in the Customers tables, all archiving for the Customers tables must complete before archiving any data from the Sales tables.

Indexes were created for the tables in the archive files based on the individual keys necessary to access the tables efficiently in those files directly using relational style queries. The queries are run using Open Database Connectivity (ODBC)/Java Database Connectivity (JDBC)-compliant software.

All Optim archive files for Sales, Customers, Orders, and Details tables are grouped into collections. There is a collection for the Orders and Details tables, and a collection for the Sales and Customers tables. Each collection contains seven years of data. The number of files in each collection is controlled by the requirement to delete archive files that contain data that is more than seven years old.

Collections can affect performance. After the archive project is implemented, further design decisions can be made for whether additional subgroups of collections are required to improve the archive file access speed.

Because of the large number of rows in the two largest tables, Orders and Details, the data is deleted from the DB2 database by dropping partitions.

The volume of data that is deleted from the Sales and Customers tables in DB2 is relatively low and the tables are not partitioned by date. Therefore, deletion is accomplished using the archive delete function. The archive delete function is also used to process the Sales and Customers tables in the Oracle database.

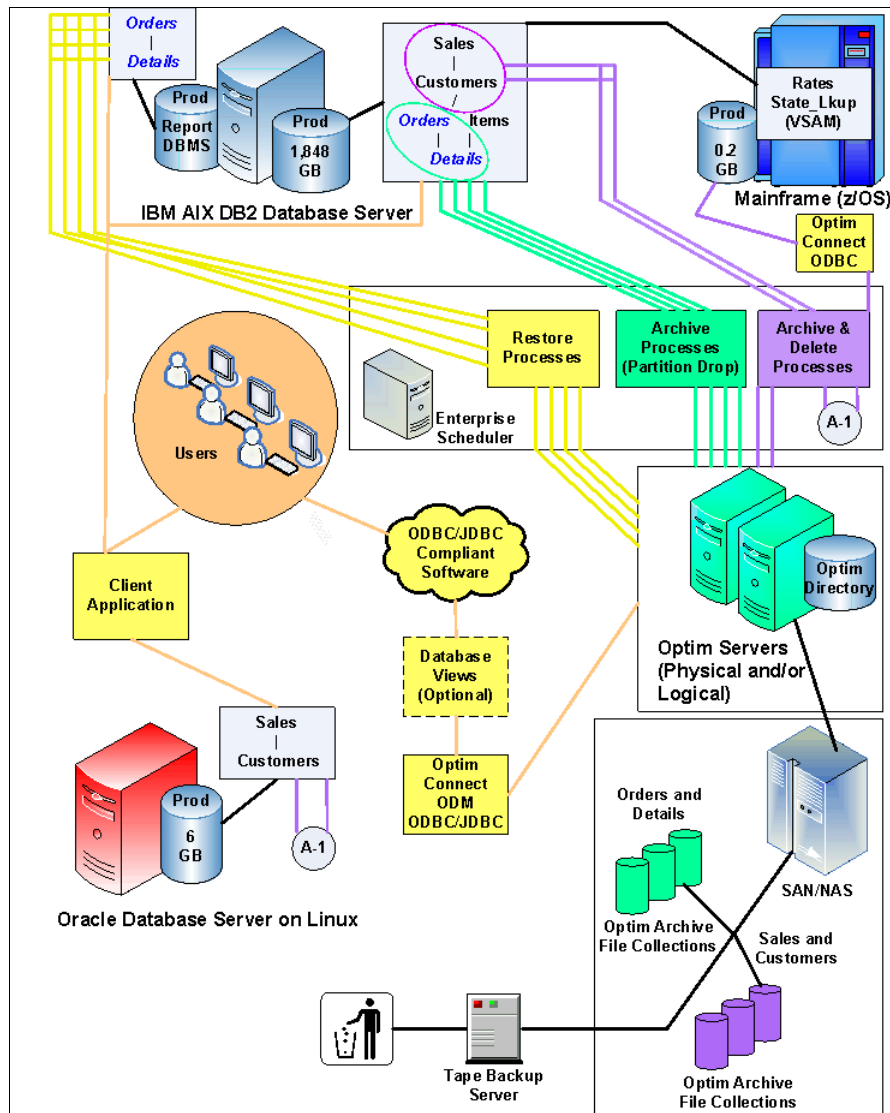
To achieve the most efficient use of storage resources, we decide to migrate the archived data to tape after one year. After all the data on an archive file passes its required retention period, the archive file is deleted.

For data retrieval, we use a combination of methods for the purpose of illustrating that there are several options available for viewing data after archival. Business requirements, application complexity, and personal preference determine which methods are used. For example, if data is archived from a complex data model that contains a large number of tables, queries to the archived data might be extremely complex and time-consuming to write. In those cases, it might be beneficial to establish a reporting database that is connected to an instance of the original application.

Retrieval of data from the DB2 database for the Orders and Details tables is achieved by restoring the data to a reporting database that is connected to an instance of the client's application.

For the Sales and Customers tables, which were archived from DB2, Oracle, and z/OS, data retrieval is accomplished by queries from ODBC/JDBC-compliant software. Access to the archive files and collections is directly from ODBC, ODBC using database management system (DBMS) views, JDBC, or any combination. Optim Connect, together with open data manager (ODM), enables archive files to appear to users and applications as relational tables.

Figure 8-1 on page 362 shows the design of our sample Optim Data Growth solution.



### 8.3 A sample z/OS design

If all application data to be archived is stored in DB2 on z/OS, Optim can be installed and executed directly from z/OS. All archive files will be written to that platform. The archive files can be restored to a reporting database, connected

directly to an application, or accessed using a GUI. It is also possible to migrate archive files that were created on z/OS to a Linux, UNIX, or Microsoft Windows platform and access them from there. Access to archive files that are stored on z/OS is accomplished using Optim Connect and ODM installed on z/OS.

### 8.3.1 Sample design

Figure 8-2 on page 364 shows a sample configuration for z/OS. It uses the same tables, business requirements, design, and configuration that were described in section 8.2, “A sample federated design” on page 356. The differences are that all of the data is stored in DB2 on z/OS and there are no network connections. Therefore, the data transfer rate is extremely fast. It is likely that fewer parallel processes are necessary to move an equal volume of data as compared to network-attached servers. Results will vary depending on the specific channel connections and the hardware that is used.

Optim is typically installed on each logical partition (LPAR) from which data is archived. Optim z/OS can be configured for *remote location* access. This configuration requires that the same release level of Optim z/OS is installed on every LPAR that contains a DB2 instance to which Optim must connect. Optim's remote location setup uses Distributed Data Facility (DDF) and Distributed Relational Database Architecture (DRDA). The advantage is that access credential authentication occurs on a single LPAR.

When connecting to remote data, network latency must be considered during the design phase of the project.

Figure 8-2 on page 364 depicts the sample design for z/OS.

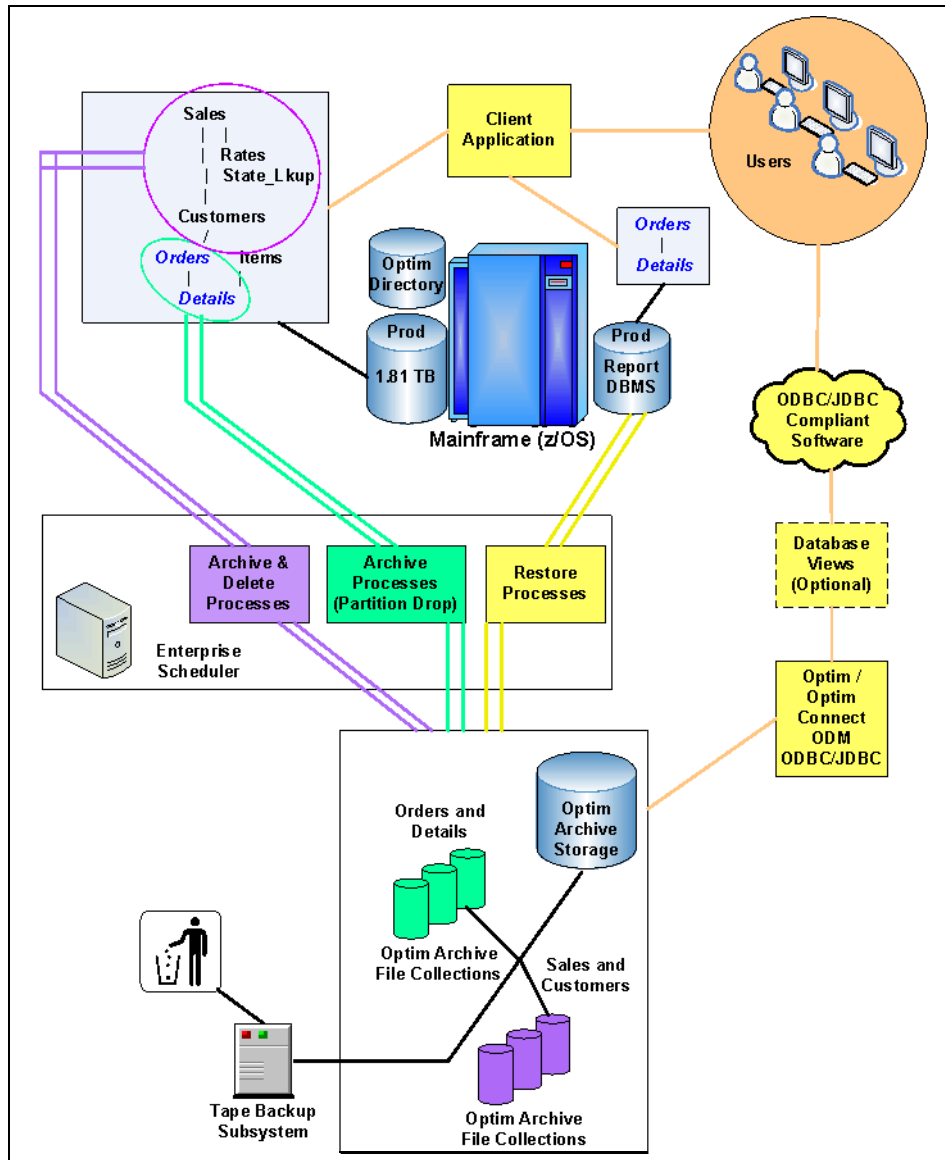


Figure 8-2 An Optim z/OS design

For more information about DDF and DRDA, see *DB2 9 for z/OS: Distributed Functions*, SG24-6952. The *IBM Optim for z/OS Installation and Administration Guide* discusses Optim remote location configuration and ODM data access.

## 8.4 Miscellaneous considerations

When implementing an InfoSphere Optim Data Growth Solution, there are many decisions to make. With proper planning, most of those decisions occur at the outset of the project as part of the design. However, certain changes can be required based on the test results of validating the initial design.

### 8.4.1 Optim processing reports

Optim produces reports from archive, delete, and restore processes that show detailed information about database connections, access, data throughput, and other relevant statistics. This information is useful for design considerations and troubleshooting. For example, if an archive process runs for an unexpectedly long time, the report can be used to identify the tables that process at slower rates. Then, the information that is displayed concerning index usage, access methods, and other data about those tables can provide clues to the cause of the issue. The information is also valuable when considering performance settings.

Figure 8-3 on page 366 shows a sample Archive Process Report with statistical information.

<b>Statistical Information:</b>		
<b>Settings During Extract:</b>		
File System Compressed: No		
<b>Step 1: Table Name: OPTIM.PSTPIADM.OPTIM_ORDERS</b>		
DBMS: DB2 LUW	Version: 09.07.0003	Columns: 8
Cycle: No	Lobs: No	Est. Rows: 9321
Row Length: 94	DB Connections: 1	Select With UR: No
Fetch Buffer Size per Connection: 2048K		
PK W/Index: N/A	PK WO/Index: N/A	
FK W/Index: N/A	FK WO/Index: N/A	
Parent Strategy: N/A		
Dependent Strategy: N/A		
<b>DBMS Access:</b>		
Access Type: Cursor Scan	Keys Per Cursor: 0	
Open Cursor: 1	Rows Fetched: 9321	
Rows Written: 9321	Time in DBMS: 100%	Time in DBMS Archive
Actions: 0%		
Process Time: 0:00:01	Rows Per Sec: 9321	
<b>Step 2: Table Name: OPTIM.PSTPIADM.OPTIM_DETAILS</b>		
DBMS: DB2 LUW	Version: 09.07.0003	Columns: 4
Cycle: No	Lobs: No	Est. Rows: 21427
Row Length: 38	DB Connections: 1	Select With UR: No
Fetch Buffer Size per Connection: 512K		
PK W/Index: 0	PK WO/Index: 0	
FK W/Index: 9321	FK WO/Index: 0	
Parent Strategy: No Keys		
Dependent Strategy: Scanning due to large number of keys		
<b>Relationship: OPTIM.PSTPIADM.OPTIM_DETAILS.ROD</b>		
Lookup Keys: 9321	Direction: dependent	Indexed: Yes
Keys Per Cursor: 1	DB2 Lookup Cost: 0.000000	DB2 Scan Cost:
0.000000		
Key Length: 13	Access: Not Forcing	
Lookup SQL:	ORDER_ID = ?	
<b>DBMS Access:</b>		
Access Type: Cursor Scan	Keys Per Cursor: 0	
Open Cursor: 1	Rows Fetched: 21427	
Rows Written: 21427	Time in DBMS: 100%	Time in DBMS Archive
Actions: 0%		
Process Time: 0:00:01	Rows Per Sec: 21427	

Figure 8-3 Optim archive process report

## 8.4.2 Optim security

We suggest that you install Optim initially in an environment that is only accessible by the users whose role is to set up and configure the solution. All lower-level security, such as Optim internal security and file-level permissions, needs to be unrestricted. After initial testing is done and it is proven that all of the components and functions work as expected, a systematic implementation of security related settings can begin. Similar to many platform-agnostic and database-agnostic tools, if Optim is initially installed in an environment with stringent security settings, issues can occur that are nothing more than permissions issues. Those issues might not manifest themselves as being

security-related and result in an unnecessary expenditure of resources. For more detail about security and how archived data is secured, refer to Chapter 11, “Security” on page 427.

### **8.4.3 Change management**

As with any new application that is introduced into a business environment, the InfoSphere Optim Data Growth Solution must be integrated into the clients’ change management process. The export and import utility that is described in 8.4.7, “Using the export and import utility” on page 369 can be used to facilitate this requirement.

Most change management policies mandate promotion through a development-level environment and at least one test-level environment prior to promoting changes to production. Development-level and test-level Optim environments might be required to create and update existing Optim objects and test the effects of the changes prior to their promotion to production. These development and test environments, or a separate system testing environment, are also used to test upgrades to Optim and related software prior to implementing upgrades into production.

With respect to change management processes, Optim must have the same controls in place as all other enterprise-level software. Because it accesses production data, special care is necessary to ensure that unauthorized changes cannot be made.

### **8.4.4 Database schema changes**

The chances are that the databases and tables from which Optim archives data will eventually be affected by schema changes. No changes to any archive processes are required when fields are added or deleted within tables. However, the addition or removal of tables does require changes to the Optim access definitions that contain the names and relationships between the tables that they process.

For accessing data, changes might be necessary depending on how the data is accessed. No changes are required for accessing data using ODM regardless of the type of schema change (provided that the data types remain compatible). However, if data is loaded to a reporting database, changes to Optim table maps and the reporting database are probably required. The components that are mentioned in this section are documented in 3.4.1, “Elements of a complete business object” on page 76.

## 8.4.5 Optim naming conventions

Naming conventions for all Optim objects must be established. Naming conventions are necessary for archive requests, access definitions, archive files, indexes, collections, delete requests, table maps, relationships, and so on. Having naming conventions reduces confusion because each Optim component can be readily identified by its name and associated with the business application that it supports.

The names for most Optim objects consist of two parts. The first node is eight characters long and can be an abbreviation of the application name, the database owner, or another meaningful identifier. The second node is 12 characters and is used to identify the object. Using an abbreviation of the *start table* name followed by a two-digit number is suggested. For example, payroll.employemst01 is used for an application named “payroll” and the start table is “employee\_master” (abbreviated and numbered).

When all of the various types of Optim objects for an application are named the same, they are easy to identify and associate at a glance using Optim’s GUI tree structure. No detailed cross-reference documents are required. The concept is similar to having multiple folders in a file system and each folder has a file with the same name. Even though the names of the files in each folder are the same, they have separate contents and functions based on the folder in which they are stored.

Well-constructed naming conventions also facilitate the integration of Optim into the client’s change management process. Refer to 11.8.5, “Configuration management” on page 465 for more information about configuration management.

If deemed useful, *table maps* can use an additional identifier in their names to show what type of restore process they support. The process types are: insert, update, insert or update, and load. For example, the table map that is used to insert data archived by the archive request named payroll.employemst01 might be named payroll.employemst1i where “i” indicates that it is an insert process versus a load or another of the process types.

For more information about start tables, table maps, and how Optim works, refer to Chapter 3, “Understanding the IBM InfoSphere Optim Data Growth solutions and software” on page 33.

Standards must include file naming conventions. Optim macros allow unique identifiers to be included as part of file names. For example, a date and timestamp can be included as part of an archive file name and its associated indexes. It is also beneficial to include the Optim object name (in part, or in whole) as part of the file name.

We suggest that project planning include a review of all Optim objects that are intended for use. Then, align the naming of Optim objects as closely as practical with any existing naming conventions.

### 8.4.6 Named objects versus local objects

There are several types of Optim objects that have the option to either be named and stored as independent, reusable objects in the Optim directory, or saved as *local*. When saved as local, an object is stored as part of another object. For example, an Optim archive request uses an access definition to determine what tables to process and how those tables are traversed. If the access definition is stored as a *named* object, it can be used by other Optim archive requests. If it is saved as local, it is stored as part of the archive request and cannot be shared with other requests.

The object types that have this option are access definitions, table maps, and column maps.

Objects saved as local objects are extremely helpful for facilitating ad hoc executions, proof-of-concept, and independent unit testing. However, we suggest that only named objects are part of any formal testing process and only named objects are promoted to production.

A good example of why named objects are important can be taken from the sample design that is used in 8.2, “A sample federated design” on page 356. For the Orders and Details tables, the archive process occurs monthly. The monthly archive is performed by four independent processes that are run in batch. Each process uses selection criteria that divides the month into four near equal parts. If the access definition is saved as local in each archive request, any change to the access definition requires modifications to each of the four archive requests. Conversely, If the access definition is stored as a named object, all four archive requests can reference that one object. Any changes can be performed by making those changes to a single access definition.

If Optim objects are saved as local, and Optim object security is used, it is possible that certain security restrictions can be circumvented. For example, if an administrator is allowed to update archive requests, but not allowed to update access definitions, that level of security cannot be enforced.

### 8.4.7 Using the export and import utility

All of the metadata that is used by Optim is stored in Optim directories. The Optim *export* and *import* utility accesses those directories providing additional backup protection for individual objects. It can facilitate the implementation of an

Optim design and be used to integrate Optim into an existing change management process by maintaining the exports of individual objects using a change management tool that supports source code versioning.

To illustrate how this utility can be used to assist in implementing a design, we refer to the example of the application named *payroll* that is used in 8.4.5, “Optim naming conventions” on page 368. For this example, we assume that the *payroll* application has been developed, tested, and is ready to archive and delete data from its production tables.

Typically, the instance of InfoSphere Optim Data Growth Solution that is transitioned to production is the actual development instance because it is already configured correctly and tested rigorously. After it is transitioned to production, there might be no instances of Optim that can be used to test changes to Optim at the development and test levels. The export and import utility can be used to help create these missing environments by exporting all of the Optim objects with a first node of “payroll” and importing them to other Optim directories.

Because the Optim export is in text format, the identical objects that are exported can be imported or global changes can be applied. Global changes are made either using the change functions that are built into the utility or the *find and replace* function of any commercially available, text-editing software. Any parameter can be modified, including the actual names of the objects.

## 8.4.8 Batch processing

When the initial Optim design is developed for a database, Optim objects usually are developed using Optim’s GUI. Sometimes, early production executions are also performed using the GUI. This approach allows the team that is assigned to implement Optim to have complete control over the process. While this approach is highly effective during development and testing, it is a manual process that does not fit well into production environments.

Optim has its own scheduler that can be beneficial for scheduling initial test runs. However, when the design is ready for formal testing and production, most clients require that their existing enterprise-level scheduling system is used. We suggest that running Optim from batch using an enterprise-level scheduling system is an approach that minimizes manual intervention, and reduces the audit-related and compliance-related issues that are associated with manual processing and intervention.

Usually, a series of batch processes make up a complete archive execution cycle. For example, parallel archive requests run to archive data. Upon their successful completion, delete requests or commands to drop partitions are

executed. Other activities either before, in between, or after these processes might be necessary. For instance, a process to create control tables to control archive data selection might be required before the execution of any archive requests.

When using an enterprise scheduling system, each process has prerequisites that are monitored. After each process completes, return codes are passed back to the scheduling system to confirm success before successor jobs are released. As a result, the entire process is tightly controlled and managed. The responsibility of executing and monitoring Optim-related processes is removed from the Optim implementation team and placed with the client's operations staff, where it belongs.

**Return code:** Most Optim processes, especially archive and delete requests, return a code of “0” when they are successful.

Optim batch processing (the command-line interface) is described in the *IBM Archive Users Manual* for Linux, UNIX, and Microsoft Windows platforms and in the *Batch Utilities Guide* for z/OS. These manuals are delivered with the product.

### 8.4.9 Matching Optim directories and DBMS code pages

Generally speaking, the Optim directory must be created in a DBMS that is code page-compatible with any DBMS that is accessed from that directory using database aliases (DB aliases), for example:

- ▶ Single-byte format (ASCII/EBCDIC)
- ▶ Unicode format (UTF-8)
- ▶ Multibyte format (MBCS)

Figure 8-4 shows the valid settings for all the possible combinations for Optim directories and DB aliases.

Optim Directory Setting	DB Alias Setting Requirement		
	Single-byte	UTF-8	MBCS
Single-byte	☑		
UTF-8	☑	☑	
MBCS			☑
DB Alias Setting	Optim Directory Setting Requirement		
	Single-byte	UTF-8	MBCS
Single-byte	☑	☑	
UTF-8		☑	
MBCS			☑

Figure 8-4 Optim DB alias compatibility table

**Important:** For DB2 on Linux, UNIX, and Windows, if the Optim directory is in Unicode format, the DB alias must also use Unicode.

When MBCS is used, the code page for the database where the Optim directory is stored must match exactly the code page for the database to which the DB alias connects. For example, if the database code page for the Optim directory is configured to support Simplified Chinese, the database code page to which a connection is made must be configured identically.

Another consideration when using MBCS is the issue of round-trip conversions. Round-trip conversions mean that there are characters that do not convert from Unicode to native code pages and back correctly.

When using MBCS, Optim uses the native code page. Most characters convert correctly from Unicode to the native code page and vice versa. However, depending on the DBMS and the specific code page that is used by that DBMS, there are characters that convert differently. This issue is limited to the Chinese, Japanese, and Korean regions. Those characters that have conversion issues are well known in the regions where they are used. The best solution is to avoid the use of characters that do not convert correctly.

Optim has settings under both product and personal options that determine how round-trip conversion errors are handled. The *IBM Optim Installation and Configuration Guide* describes these settings.

## 8.4.10 Handling extreme volumes of data

Certain organizations deal with huge volumes of transactional data. Often, that data must be kept for regulatory compliance; however, the volume can make it costly and impractical to maintain that data in a database. Telecom service providers are a good example where this situation can occur. The examples in this chapter provide enough detail to help you understand techniques to manage medium to large data volumes. However, when extreme amounts of data must be archived, extra attention must be given to design and configuration. Chapter 4, “Developing an InfoSphere Optim Data Growth Solution architecture” on page 93 contains additional information and diagrams of configurations describing how to scale the Optim design for processing extreme volumes.

## 8.4.11 Complex data selection requirements

Optim provides numerous methods of supplying criteria to select data for archival. These methods include options that range from simple data selection criteria to robust SQL query support.

Specific data selection requirements are extremely complex. They can require access to multiple fields in multiple tables to determine which rows are eligible for archiving. In these cases, it might be more efficient to build a repeatable process that loads *control tables* that contain the keys of the rows that are eligible for archival. Control tables are used as *start tables* for archive processes.

Creating control tables uses more overall resources, but it offers the potential advantage of being able to create them in advance and thereby reduce the amount of time that actual archive requests run. Control tables can be as simple as one column tables that contain a list of all of the keys associated with data that meets the criteria for archival. They can also be robust and additionally store all of the metadata necessary to rapidly identify which archive files contain the specific data requested for restore or viewing.

For more information about start tables and how Optim works, refer to Chapter 3, “Understanding the IBM InfoSphere Optim Data Growth solutions and software” on page 33 and Chapter 9, “Deployment and operational considerations” on page 375.

## 8.4.12 Special data retention requirements

We have described the design for the “normal” processing of archived data. Data meets the specified archive criteria; it is archived; it is deleted from the DBMS; it is accessed from archive files as necessary; and ultimately, archive files are deleted after their retention requirements are satisfied. However, normal is not always the case. For reasons of civil litigation and regulatory investigation, sometimes data cannot be deleted after the published retention requirements have passed. Optim has a *litigation hold* feature that allows clients to comply with these requirements.

While litigation hold is a feature of Optim, it also requires a business process. Litigation hold is accomplished by preventing the deletion of archive files and their associated directory entries. Therefore, when the retention period of archive files have passed, if they are on litigation hold, they are not deleted. Typically, only a small percentage of the data on archive files is the data that is affected. If that is the case, there is data on the archive files that needs to be deleted. Not deleting that data violates the business’ retention policies and potentially introduces legal risk. Therefore, a business process is necessary.

We suggest that ownership of any data placed on litigation hold is transferred to the business’ legal department. Therefore, the data affected must be copied to the legal department’s own set of archive files and restored to special litigation hold databases. The legal department then manages those copies until the end of the litigation hold, at which time those copies of the data are deleted.

After the legal department has a copy of the data and it is certified by the business to be complete and accurate, the litigation hold can be removed from the original archive files and they can expire according to their normal retention policies.

Optim provides features and functions that create subsets of archive files that can facilitate building a business process to support compliance with litigation hold requests. Chapter 3, “Understanding the IBM InfoSphere Optim Data Growth solutions and software” on page 33 contains additional information concerning litigation holds.



# Deployment and operational considerations

In this chapter, we provide information about deployment and operational considerations for IBM InfoSphere Optim Data Growth Solution (Optim). The topics that are covered in this chapter consider databases as source systems. We discuss the following topics in this chapter:

- ▶ Data source and operational design
- ▶ Single and multi-instance Optim deployments
- ▶ Archival and restoration job configurations
- ▶ Control tables
- ▶ Monitoring
- ▶ Archival and restoration job executions
- ▶ Auditing, monitoring and reporting
- ▶ Storage tiering
- ▶ Restarting Optim jobs
- ▶ Backup and disaster recovery

## 9.1 Data source and operational design

The Optim Data Growth Solution on distributed platforms is a client/server application that archives and restores structured data. Related data is archived from all associated database tables according to selection criteria that is derived from your business requirements. Archived data is retrieved by relational queries against the archive files or after restoration to a database.

The start table and selection criteria for archiving are the starting points for the automation of the operational process. Minimal manual intervention is required after automation is complete. The necessary logic to automate an archival process is also the key to automating the restoration process. Consider these key points:

- ▶ Your data retention policy, which helps determine automation requirements for your data storage tiering strategy
- ▶ The categorization of data in database tables
- ▶ The deployment strategy (single or multi-instance) for your solution

Categorizing the data is important in the development of an efficient, archival solution design.

Here, we continue with the example that we described in Chapter 8, “Design and configuration” on page 353, which provided the solution design considerations using Orders, Details, Customers, and Items tables. The data in the Orders table can be categorized based on customer and order types. The intent is to create four parallel archival job queues that are based on customer and order types. As an example, consider the Orders table containing values GENERAL and PRIORITY for customer type column CUST\_TYPE whereas values for order type column ORDER\_TYPE are BUSINESS and PERSONAL. Figure 9-1 shows a logical relationship between these values.

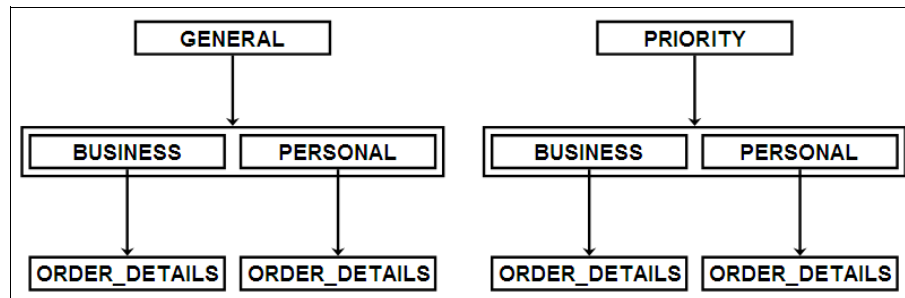
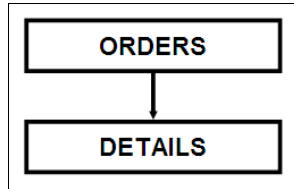


Figure 9-1 Logical relationship between the Orders and the Details tables

A customer type GENERAL can place an order of type BUSINESS or PERSONAL. The associated details of these orders are kept in the Details table with the primary key on ORDER\_ID column. The archived business object, which is shown in Figure 9-2, consists of the Orders and Details tables. The Orders and Details tables are archived monthly to remove data that is older than three years. The data in these tables can be a combination of online (data that is needed on a daily basis by the application) and historical.



*Figure 9-2 Archived business object*

The data in the Customers and Items tables is considered master records. Part of the data in these tables can become inactive over time. It might be necessary to schedule regular archive processes to remove any inactive data from these tables for performance reasons and to store it for a period for compliance. In our example, the Customers table is archived yearly to remove inactive customers.

Table partitioning is a powerful aid to performance. Both the Orders and Details tables are partitioned by calendar date range in this example, with a calendar month taking up a single partition. Thus, archiving Orders and Details tables monthly makes it easier to remove archived data from these tables by simply dropping the corresponding partitions.

Analysis of the most common queries that are anticipated for restoration jobs drives the creation of the indexes on archived data. The column indexes might also become the filter condition for restoration jobs, which can be captured as part of metadata of the operational design.

In this example, the source database contains indexes on the CUST\_ID and ORDER\_ID columns of the Orders table and the ORDER\_ID column of the Details table. This information provides us with clues on what columns to create archive indexes.

Now, let us look at the options to consider for deploying Optim.

## 9.2 Single or multi-instance Optim deployment

You can deploy the InfoSphere Optim Data Growth Solution in a single or multi-instance configuration on a single physical server or logical partition (limitations might exist because of the choice of underlying infrastructure and operating system type).

An *Optim instance* refers to a complete software deployment of the InfoSphere Optim Data Growth Solution. A single instance deployment is the default installation as described in the *Installation and Configuration Guide* for the solution. A *host* is an operating system that is installed on a bare metal physical server or a logical partition in a virtualized infrastructure environment, such as logical partitions (LPARs) on IBM System p® using Power VM Hypervisor.

A multi-instance deployment requires multiple installations of the InfoSphere Optim Data Growth Solution on one or more physical servers or LPARs. Consider a multi-instance deployment topology for the following reasons:

- ▶ Scale up at the application (InfoSphere Optim Data Growth Solution) layer and, at the same time, scale out across multiple physical servers.
- ▶ Ensure that the solution is highly available.
- ▶ Distribute workload across multiple instances of Optim server.
- ▶ Invoke parallel archival and restoration jobs.

The following sections in this chapter use a multi-instance deployment topology to explain automated operational procedures for the InfoSphere Optim Data Growth Solution.

### 9.2.1 Multi-instance deployment

The installation and configuration of your solution require a unique user ID on the operating system. Multiple installations of the InfoSphere Optim Data Growth Solution on the same operating system are possible. Consider these suggestions for a multi-instance deployment, from an operational design perspective:

- ▶ Use a general parallel file system (GPFS) across all installations to be able to retrieve archived data from any instance of the Optim server.
- ▶ Use a common User ID (UID) and Group ID (GID) for installations across multiple physical servers.
- ▶ Point each instance of the InfoSphere Optim Data Growth Solution to the same Optim directory.

- For every Optim installation, create endpoints of all the other installations so that each installation can retrieve data from archive files that were created by other installations.

Refer to Chapter 4, “Developing an InfoSphere Optim Data Growth Solution architecture” on page 93 for more details about scale-up and scale-out considerations.

Consider the multi-instance deployment in Figure 9-3 in which two physical servers, Host A and Host B, have two instances of the InfoSphere Optim Data Growth Solution installed on them. GPFS is installed, configured, and mounted on both physical servers.

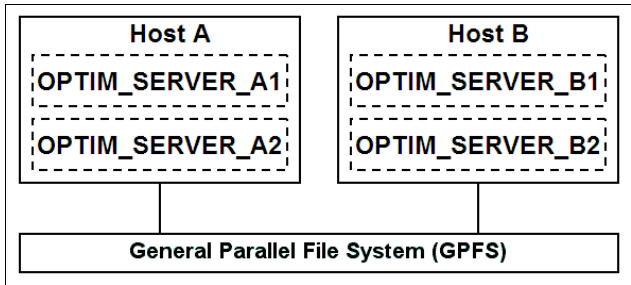


Figure 9-3 Multi-instance deployment layout

In this example, we use GPFS mount points for storing the archive files, and corresponding index files have been created. Table 9-1 depicts a scheme to create these mount points.

Table 9-1 GPFS mount points

Host (physical server)	Optim Server installation	Mount point	Path
A	1	/HOSTA1	/HOSTA1/GENERAL/BUSINESS/
A	2	/HOSTA2	/HOSTA2/GENERAL/PERSONAL/
B	1	/HOSTB1	/HOSTB1/PRIORITY/BUSINESS/
B	2	/HOSTB2	/HOSTB2/PRIORITY/PERSONAL/

The path name that is depicted in this scheme can be generalized, as shown in Example 9-1 on page 380.

---

```
/<hostname><Optim server name>/<customer type>/<order type>/
```

---

The intent is to create four parallel archival job queues that are based on customer and order types. There are also a total of four combinations of customer and order type values. We archive each combination of customer and order type value per Optim server. Each Optim server creates archive files in a unique path. Thus, the scheme is to create a directory structure that is based on host name, Optim server name, customer type, and order type. It becomes easy to identify the archive files per the customer-order type combination at the file system level by system administrators.

Ensure that all GPFS mount points are visible on both hosts and have appropriate permissions. This sample configuration allows you to separate archive files by customer and order type. You can also further create subdirectories under order type subdirectories that are based on monthly archival operations.

Having a physical file system layout aligned to your logical solution design helps system administration activities on physical archive and corresponding index files.

## 9.3 Archival job configuration

The specifications for archiving a business object are made using an Optim Access Definition (AD). For this example, the business object is shown in Figure 9-2 on page 377. The access definition has these functions:

- ▶ Identifies the start table for archival job execution
- ▶ Lists details of the additional tables to be archived and provides criteria in the form of a WHERE clause to select the data to be archived
- ▶ Requires archive indexes on the CUST\_ID and ORDER\_ID columns in the Orders table and the ORDER\_ID column in the Details table
- ▶ Specifies the archive actions that are used to run custom queries during various stages of the archival job execution

In the absence of referential integrity in the database, you can define Optim relationships between tables in the business object. In this example, we create an Optim relationship between the Orders and Details tables with the relationship column as ORDER\_ID, as shown in Figure 9-4 on page 381.

Parent:		Base.Creator.Id:		Type:
OPTIMDIR.ADMINISTRATOR.ORDERS				Optim
Child:		Description:		
OPTIMDIR.ADMINISTRATOR.DETAILS				

	Parent Expression	Data Type	Child Expression	Data Type	Status
1	ORDER_ID	DECIMAL(10,0)	ORDER_ID	DECIMAL(10,0)	OK
2					

Figure 9-4 Optim relationship

## 9.4 Restore job configuration

The restoration job is configured to retrieve data from archive files, either selectively or in bulk. The bulk restoration extracts all data from the set of archive files that you choose for this type of restoration, whereas, a selective restore extracts only those records that match the selection criterion that you specify for the request.

You can find the detailed information that is needed to create your restore request template in Chapter 8 of the *IBM Archive User Manual*. At a minimum, a restore request requires this information:

- ▶ A list of archive files
- ▶ The filter condition for data to be restored
- ▶ The type of database Data Manipulation Language (DML), such as LOAD or INSERT, that is used to restore data

Depending on the volume of data to be restored and service-level agreements (SLAs) for data restoration, you can choose to use one or several instances of Optim server for processing.

## 9.5 Control tables

Scheduling automated archival job executions requires access to information pertaining to the configuration of these archive jobs and corresponding Optim server objects. Using a combination of the host environment and the details of the pertinent Optim objects, it becomes possible to run archival jobs with minimal manual intervention. In this section, we explain how to keep and access this kind of information in database tables.

You can choose to create these tables in your choice of database type, for example, IBM DB2 or Oracle, in any of the following choices of database instances:

- ▶ Optim directory database, which is your Optim metadata database
- ▶ Source database from which Optim solution extracts data to create archives
- ▶ A stand-alone database instance

### 9.5.1 Archival job control table

For the example that is considered in this chapter, monthly scheduled and automated archival jobs are required. The core design for this automation process consists of maintaining a control table, which allows you to perform the following tasks:

- ▶ Run archival jobs according to schedule.
- ▶ Maintain the status of archival job execution.
- ▶ Maintain the metadata relationship among the data categorization, access definition, Optim server, physical server, archive file location, storage media, archive file size, rows archived, and so on.
- ▶ Act as a lookup table for retrieval requests.

With the appropriate set of database privileges and access, you must create a new table, for example, Control\_Driver. This table can reside in the database that hosts the Optim directory or in a separate database. The metadata for automating the archiving process can vary, as required.

For this example, consider the columns in Table 9-2 for the Control\_Driver table.

Table 9-2 Control table columns

Column name	Description
HOST_NAME	Host name of the physical server
OPTIM_SERVER	Optim server installation name as configured in <code>pstserv.cfg</code> configuration file for the <i>rtservername</i> parameter
MOUNT_POINT	GPFS mount point with path details for ARCHIVE_FILE
ARCHIVE_FILE	Archive file name with complete path
ARCHIVE_FILE_SIZE_COMPRESSED	Compressed size of archive file as reported by the InfoSphere Optim Data Growth Solution
ARCHIVE_FILE_SIZE_UNCOMPRESSED	Non-compressed size of archive file as reported by the InfoSphere Optim Data Growth Solution
ARCHIVE_FILE_SIZE_DISK	Size of archive file as reported by the operating system
COMPRESSION_RATIO	Compression percentage as reported by the InfoSphere Optim Data Growth Solution
ARCHIVE_FILE_RECORDS_TOTAL	Total number of rows in the archive file
ARCHIVE_FILE_RECORDS_PARENT	Number of rows in the start table, in this example, Orders table
ARCHIVE_FILE_RECORDS_CHILD	Number of rows in the child table, in this example, Details table
ARCHIVE_REQ_NAME	Optim archive request object name
AD_NAME	Access definition name
CUST_TYPE	Customer data category
ORDER_TYPE	Order data category
YEAR	Calendar year of data to be archived
MONTH	Calendar month of YEAR of data to be archived
STATUS	Successful (S) or not successful (N)

The data in the control table `Control_Driver` helps automate and run the archival jobs. There is also a need to record configuration information that maps the information about the source tables and identifies the instance of the InfoSphere Optim Data Growth Solution to archive a particular category of data from source tables.

This mapping information can be stored in another table in the database hosting the Optim directory. For example, you can create a table called `Map_Table` that contains information about the mapping among the Optim server, physical server host name, and data category. This step is a one-time activity that is required to predefine the Optim server that is used for each data category archival request.

Table 9-3 shows an example of the mapping table.

*Table 9-3 Mapping table columns*

Column name	Description
HOST_NAME	Host name of the physical server
MOUNT_POINT	GPFS mount point with path details for <code>ARCHIVE_FILE</code>
OPTIM_SERVER	Optim Server installation name as configured in the <code>pstserv.cfg</code> configuration file for the <i>rtservername</i> parameter
CUST_TYPE	Customer data type
ORDER_TYPE	Order data type
ARCHIVE_REQ_NAME	Optim archive request object name
RESTORE_REQ_NAME	Optim restore request object name
AD_NAME	Access definition name
START_TABLE	Start table object from archive request Optim object
CHILD_TABLE	Child table object from archive request Optim object
FILTER_COL	Comma-separated list of columns of <code>START_TABLE</code> , which is used in the WHERE clause of the SQL parameter of the override file

The `Map_Table` that is shown in Table 9-3 contains information correlating Optim objects, source data categorization, host, and Optim server details. The data categorization in the example is done on types of customers and orders and the intent is to drive parallel archival jobs per customer-order combination. There can be four types of combinations for this example, as shown in Table 9-4 on page 385.

Table 9-4 Data categorization

Customer type	Order type
GENERAL	BUSINESS
GENERAL	PERSONAL
PRIORITY	BUSINESS
PRIORITY	PERSONAL

For each combination that is listed in Table 9-4, an entry in Map\_Table is assigned to a unique Optim server instance. This way, you create parallel archival job queues per Optim server instance that is deployed across two physical host machines. Map\_Table also captures the location of the corresponding archive file to be kept on the GPFS file system per Optim server instance. The corresponding restore request Optim object name is also captured in this table, to automate restoration jobs.

### 9.5.2 Restoration job control table

Having controlled the execution of archival jobs, it is also important to be able to control the restoration jobs and retain the execution information in an audit table. The Control\_Driver table is also the driver table for restoration, because it contains the complete information about each archive file.

Additionally, we also need a table to capture the details of the restoration processes. This table can track the details of restoration processes. After a process has completed successfully, the output logs from Optim server are used to update the corresponding restore request details. Thus, you can track the attempts to restore data and the number of records restored. Essentially, you can use this table to audit the restoration processes.

For this example, consider using the columns in Table 9-5 on page 386 to create a table called Restore\_Info.

Table 9-5 Restoration information table columns

Column name	Description
RESTORE_REQ_NUM	Unique serial number to track restore processes.
RESTORE_REQ_DATE	Date on which restore process was attempted.
RESTORE_TYPE	BULK or SELECTIVE.
CUST_TYPE	Customer data type: GENERAL, PRIORITY, or BOTH.
CUST_ID	A comma-separated list of unique customer IDs.
ORDER_TYPE	Order data type: BUSINESS, PERSONAL, or BOTH.
ORDER_START_DATE	Start date filter specification for orders.
ORDER_END_DATE	End date filter specification for orders.
RESTORE_TABLE	Start table from Optim restore request. Also available as START_TABLE in Map_Table.
RESTORE_FILTER_COL	Comma-separated list of columns of start table that is used in WHERE clause of LOCSQLTEXT parameter value.
OPTIM_SERVER	Optim server installation name; Optim server instance for restoration process.
HOST_NAME	Physical server host name on which OPTIM_SERVER for this process is running.
OPTIM_USER_ID	System user ID for restoration process.
IS_RESTORE_TO_DB	Is the restoration process required to load data to a target database - Yes (Y) or No (N).
TARGET_DB_ALIAS	Database alias for target, if any.
TOTAL_ROWS_EXTRACTED	Total number of rows extracted in the restore process.
TOTAL_RESTORE_START_TIME	Start date and timestamp of restore process.
TOTAL_RESTORE_END_TIME	End date and timestamp of restore process.

Column name	Description
PARAMETER_FILE	File name with complete path used for executing restore process. This file name is created dynamically during restore job execution using archive file path name, customer type, order type, and unique request number.
LOG_FILE	Log file with complete path that is used to capture log output. This file name is created dynamically during the restore job execution using archive file path name, customer type, order type, and unique request number.
STATUS	Successful (S) or not successful (N).

## 9.6 Archival job execution

After the metadata is configured, you can use the steps that are outlined in this section to create a script in any programming language and run it using a standard scheduler, such as *cron* on UNIX. Figure 9-5 shows the logical workflow of these steps.

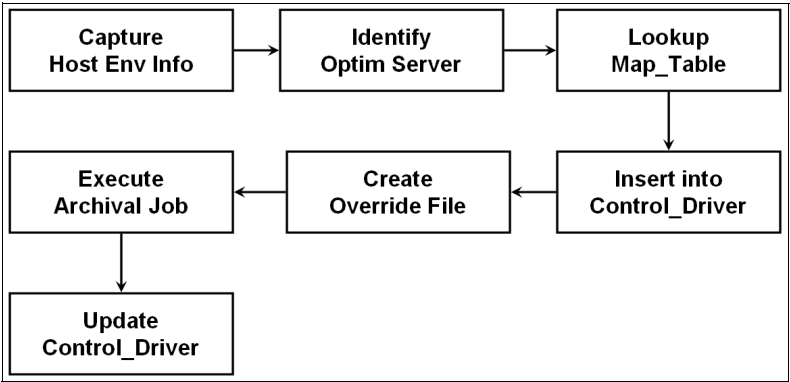


Figure 9-5 Archive script logical flow

Use the following steps to create your automated archival job process:

1. The archival process starts by gathering data on the execution environment and updating the Control\_Driver:
  - a. Capture system environment details:
    - Host system name
    - Host system date (year and month)
    - Host system user ID
  - b. Identify the Optim Server name using these system environment details.
  - c. Look up Map\_Table for the entry matching the system environment details with HOST\_NAME, MOUNT\_POINT, and OPTIM\_SERVER. Obtain the details of the corresponding columns for the matched record to build the automated archive job.
  - d. Insert a new entry into Control\_Driver for this archival job and keep the STATUS column value as "N". The YEAR and MONTH for this archival job are calculated using logic that is based on the retention policy and system date.
  - e. Determine the start and end calendar dates for YEAR and MONTH.
  - f. Reconcile within Control\_Driver that this job was not executed previously.
2. The override file for the archival job is created dynamically using the information in Control\_Driver and the information that was captured in step 1 as shown in Example 9-2.

*Example 9-2 Override file*

---

```
AFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2007-05_<$SEQ>.AF
AFXFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2007-05_<$SEQ>.AFX
ADNAME DEMO.ORDERSAD
DEFERDAA Y
SQL DEMODB.ORDERS ORDER_DATE BETWEEN '2007-05-01' AND '2007-05-31' AND
CUST_TYPE='GENERAL' AND ORDER_TYPE='BUSINESS'
```

---

**Important:** The archive file-naming convention is extremely important. A proper template allows a system administrator to identify the type of data in an archive file easily. The breakdown of the archive file-naming convention that is used for this example is:

*<Customer Type>\_<Order Type>\_Year - Month \_ \$SEQ*

We explain the values in this example:

- ▶ *<Customer Type>* = Value from CUST\_TYPE
- ▶ *<Order Type>* = Value from ORDER\_TYPE
- ▶ Year = YEAR to which archived data belongs
- ▶ Month = Value of MONTH of YEAR to which archived data belongs
- ▶ \$SEQ - Sequential file name macro

Macros in file names are especially useful in a process that is scheduled to run automatically at specified intervals, so that a new name is generated each time that the process is executed.

Using this Optim macro, the sequence number is increased by a single digit each time that Optim generates a file name. The generated number causes the file name for each run to be unique. Refer to the *IBM Common Elements Manual* for a complete list of Optim file name macros.

### 3. The archival job is run, as shown in Example 9-3.

#### *Example 9-3 Running an archive job*

---

```
pr0cmd /R TYPE=ARCHIVE REQUEST=$optim_arch_request_obj OUTPUT=$log_file  
SERVER=$optim_server OV=$ov_file
```

---

In this example, the variables have the following meanings:

- *\$optim\_arch\_request\_obj* is the archive request template that was created using the first Optim workstation. This value was captured from Map\_Table.
- *\$log\_file* is the file name with the complete path relative to MOUNT\_POINT from Map\_Table to keep the output of the **pr0cmd** command.
- *\$optim\_server* is the Optim Server instance name that is captured from the OPTIM\_SERVER column value of Control\_Driver.
- *\$ov\_file* is the override file name with the complete path relative to MOUNT\_POINT from Map\_Table.

4. After the archival job completes successfully, update the Control\_Driver with following information that was captured in *\$log\_file* and the information that was captured in a custom audit table using archive actions (see 9.8, “Auditing” on page 395 for more information about how to create a custom audit table):
  - ARCHIVE\_FILE
  - ARCHIVE\_FILE\_SIZE\_COMPRESSED
  - ARCHIVE\_FILE\_SIZE\_UNCOMPRESSED
  - COMPRESSION\_RATIO
  - ARCHIVE\_FILE\_RECORDS\_TOTAL
  - ARCHIVE\_FILE\_RECORDS\_PARENT
  - ARCHIVE\_FILE\_RECORDS\_CHILD
  - STATUS

You can delete the override files, because all the information that is required to create these files is retained in Control\_Driver. Ensure that you retain *\$log\_file* for auditing purposes.

## 9.7 Restoration job execution

Executing a restoration job requires that a restore request is created, using the first Optim workstation. It is possible that for a given restore process, there are multiple archive files involved. In this scenario, a single entry in Restore\_Info will not suffice to drive and track a restore process.

Therefore, for every unique RESTORE\_REQ\_NUM in the Restore\_Info table, create multiple entries for each archive file that is identified for restoration into a table called Restore\_Info\_Detail. Consider using the columns in Table 9-6 when creating the Restore\_Info\_Detail table.

*Table 9-6 Restoration information detail table columns*

Column name	Description
RESTORE_REQ_NUM	Unique serial number to track restore requests
ARCH_FILE_NAME	ORDERS table archive file name with complete file path details
ROWS_EXTRACTED	Number of rows extracted

Using the combination of the Restore\_Info and Restore\_Info\_Detail tables, consider a logical restore job workflow, as shown in Figure 9-6 on page 391.

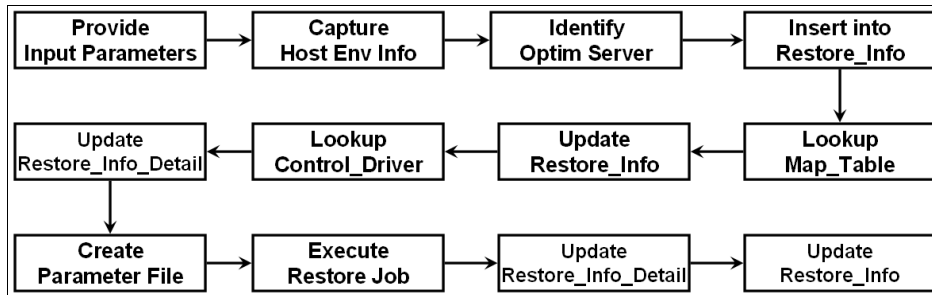


Figure 9-6 Restore job workflow

We now describe how to run selective and bulk restorations based on this logical workflow.

### 9.7.1 Selective restore

A restoration process might not be completely automated, because it requires user input for information about filter conditions.

Consider a scenario in which orders of type BUSINESS and associated details are to be retrieved for customer IDs 10208, 11458, and 11257 belonging to customer type GENERAL and between the order dates 2008-12-15 and 2009-4-15. Also, data must be retrieved to flat files and not restored into a database.

You can choose to run the restoration request from any Optim server instance in your deployed environment. However, it is best to restore from the same server instance that was used to archive that data type.

The following steps are the logical sequence of steps for automating this selective restore request operation:

1. Provide the following parameters to the restoration script:
  - RESTORE\_TYPE = SELECTIVE
  - CUST\_ID = '10208,11458,11257'
  - CUST\_TYPE = GENERAL
  - ORDER\_TYPE = BOTH
  - ORDER\_START\_DATE = '2008-12-15'
  - ORDER\_END\_DATE = '2009-4-15'
  - IS\_RESTORE\_TO\_DB = N
  - RESTORE\_REQ\_DATE

2. Capture host system environment details:
  - Host system name
  - Host system date
  - User credentials with which you are logged on to the host system
  - Identify the Optim Server name using these system environment details and Map\_Table
3. Generate a unique request number of your choice when you insert the system environment details that were captured in step 2 and the input values that were provided in step 1. For this example, we use request number 61.
4. Using the customer and order type combination information that was given for this restoration request, select the correct entry in Map\_Table and update the Restore\_Info table for request number 61 with this information:
  - RESTORE\_TABLE
  - RESTORE\_FILTER\_COL
    - If CUST\_ID is not NULL and not ALL, add CUST\_ID to RESTORE\_FILTER\_COL comma-separated list.
  - PARAMETER\_FILE
  - LOG\_FILE
5. Because IS\_RESTORE\_TO\_DB = N, update TARGET\_DB\_ALIAS = NULL for request number 61.
6. Set STATUS = N.
7. For request number 61, split the values of ORDER\_START\_DATE and ORDER\_END\_DATE and keep in system variables:
  - START\_YEAR = 2008
  - END\_YEAR = 2009
  - START\_MONTH = 12
  - END\_MONTH = 4
8. For request number 61, identify all ARCHIVE\_FILE from Control\_Driver using following information:
  - CUST\_TYPE = "GENERAL"
  - ORDER\_TYPE = "BUSINESS"
  - YEAR = 2008 and 2009
  - MONTH = 12 and 4
9. For each ARCHIVE\_FILE that was identified in step 8, create a unique entry in Restore\_Info\_Detail and tag it to request number 61.
10. If all or several archive files that were identified for restoration are on tape media, ensure that the vendor software interface is used to restore these files to their original disk file locations before executing the restore request.

11. For each unique ARCHIVE\_FILE in Restore\_Info\_Detail, create an [AFn] entry in the PARAMETER\_FILE. For this example, we assume that there are three archive files that meet our selection criterion. Example 9-4 shows the parameter file.

*Example 9-4 Optim parameter file*

---

```
TYPE=RESTORE
REQUEST=$optim_restore_request_obj
OUTPUT=/HOSTA1/GENERAL/BUSINESS/RESTORED/LOGS/GENERAL_BUSINESS_61_LOG.txt
OV=*
```

```
[AF1]
SERVER OPTIM_SERVER_A
AFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2008-12_19.AF
SUBXFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2008-12_19.XF
LOCSQLTEXT DEMODB.DEMO_USER.ORDERS CUST_ID IN (10208,11458,11257) AND
ORDER_DATE BETWEEN '2008-12-15' AND '2009-4-15'
GLOSELSQL N
TMNAME DEMO.ORDERSTM
PERFLOAD N
MODE DEMODB INSERT
DELONSUCCESS DEMODB N
DELONFAILURE DEMODB N
WORKPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
SERVPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
DESTQUAL DEMODB.DEMO_USER
```

```
[AF2]
SERVER OPTIM_SERVER_A
AFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2009-03_20.AF
SUBXFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2009-03_20.XF
LOCSQLTEXT DEMODB.DEMO_USER.ORDERS CUST_ID IN (10208,11458,11257) AND
ORDER_DATE BETWEEN '2008-12-15' AND '2009-4-15'
GLOSELSQL N
TMNAME DEMO.ORDERSTM
PERFLOAD N
MODE DEMODB INSERT
DELONSUCCESS DEMODB N
DELONFAILURE DEMODB N
WORKPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
SERVPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
DESTQUAL DEMODB.DEMO_USER
```

```
[AF3]
SERVER OPTIM_SERVER_A
AFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2009-04_21.AF
SUBXFFILE /HOSTA1/GENERAL/BUSINESS/GENERAL_BUSINESS_2009-04_21.XF
```

```

LOCSQLTEXT DEMODB.DEMO_USER.ORDERS CUST_ID IN (10208,11458,11257) AND
ORDER_DATE BETWEEN '2008-12-15' AND '2009-4-15'
GLOSELSQL N
TMNAME DEMO.ORDERSTM
PERFLOAD N
MODE DEMODB INSERT
DELONSUCCESS DEMODB N
DELONFAILURE DEMODB N
WORKPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
SERVPATH DEMODB /HOSTA1/GENERAL/BUSINESS/RESTORED
DESTQUAL DEMODB.DEMO_USER

END

```

---

**Important:** In this example, we restore the data to flat files so it is important to ensure that the load request uses Insert mode, and that the File Type is set to 'ASCII Delimited' with your choice of delimiter.

Also, note that we do not provide the CUST\_TYPE and ORDER\_TYPE filters in the LOCSQLTEXT parameter, because we have already identified the specific set of archive files per the restoration criteria of the customer and order types. We do not need the filters, because the archive files were created using the customer type and order type data categorization scheme.

12. The restoration job is then executed, as shown in Example 9-5.

---

*Example 9-5 Running restore job*

---

```
pr0cmd /R @GENERAL_BUSINESS_61_PAR.txt
```

---

13. When the restoration completes, use the LOG\_FILE to update the Restore\_Info\_Detail table with the total number of rows (ROWS\_EXTRACTED) extracted for every row entry with RESTORE\_REQ\_NUM = 61 and corresponding ARCHIVE\_FILE.
14. Use the information that was captured in step 13 to update the Restore\_Info table for TOTAL\_ROWS\_EXTRACTED for the request number 61 record and also use LOG\_FILE to update following values in the Restore\_Info table:
  - TOTAL\_RESTORE\_START\_TIME
  - TOTAL\_RESTORE\_END\_TIME
15. Update the STATUS column in the Restore\_Info table for request number 61 to successful (S).

## 9.7.2 Bulk restore

Consider a scenario where you must retrieve data from archive files based on a filter condition but the entire set of archive files must be restored. In this scenario, the requirement is to restore all orders and associated details for all customers of type GENERAL from months January to March for year 2009. Again, the requirement is to restore data to flat files.

The procedure that was outlined in 9.7.1, “Selective restore” on page 391 also applies in this case with the following changes:

1. Provide these parameters to the restoration script:
  - RESTORE\_TYPE = 'BULK'
  - CUST\_ID = ALL
  - CUST\_TYPE = 'GENERAL'
  - ORDER\_TYPE = 'ALL'
  - ORDER\_START\_DATE = 2009-1-1
  - ORDER\_END\_DATE = 2009-3-31
  - IS\_RESTORE\_TO\_DB = 'N'
2. Follow steps 2 on page 392 to 10 as outlined in 9.7.1, “Selective restore” on page 391, but with a new request number generated and stored in the RESTORE\_REQ\_NUM field of the Restore\_Info table.

Create the parameter file in exactly the same way as shown in step 11 of 9.7.1, “Selective restore” on page 391, but change the LOCSQLTEXT per the bulk restore requirement filter condition. See Example 9-6.

### *Example 9-6 Parameter file*

---

```
LOCSQLTEXT DEMODB.DEMO_USER.ORDERS ORDER_DATE BETWEEN '2009-1-1' AND  
'2009-3-31'
```

---

**Tip:** You also can also calculate and provide the exact calendar date range for each month.

3. Complete steps 12 to 14, as described in 9.7.1, “Selective restore” on page 391 for the new request number.

## 9.8 Auditing

Archival processing requires the maintenance of audit records for reconciliation. The InfoSphere Optim Data Growth Solution has a feature called archive actions that supports the ability to capture additional information during the execution of

an archival. The InfoSphere Optim Data Growth Solution provides built-in variables for creating the archive actions. For a complete list of the built-in variables, refer to the *IBM Optim Common Elements Manual*.

For this example, we use the built-in variables in Table 9-7 to create archive actions to run at the start and end of extract processes.

*Table 9-7 Built-in variables*

Built-in variable	Description
PST_ARCHIVE_ID	Archive process ID.
PST_ARCHIVE_FILE_NAME	Name of archive file with absolute path.
PST_INDEX_FILE_NAME	Name of corresponding index file name with absolute path.
PST_ACTION_TEXT	Archive action - Before or After extraction.
PST_USER_ID	System user ID for execution of archival job.
PST_SERVER_NAME	Name of Optim server instance on which the archival job is executed.
PST_MACHINE_NAME	System host name on which the archival job is executed.
PST_START_DATETIME	Archive process start time.
PST_CURRENT_DATETIME_CHAR	Archive process end time after the archival job has finished with a status code (See PST_PROCESS_STATUS variable) or this variable can be used to monitor the archival job.
PST_TOTAL_ROWS_EXTRACTED	Total number of records extracted by this archival job.
PST_PROCESS_STATUS	Process status. See Table 9-8 for possible values.

The extract action phases enable a built-in variable that is labeled PST\_PROCESS\_STATUS. Table 9-8 lists the possible returned status values.

*Table 9-8 Values for PST\_PROCESS\_STATUS*

Value	Description
0	Success.
1	The process was cancelled.
2	A DBMS error occurred that aborted the archive.

Value	Description
3	An archive action aborted the archive.
4	An error occurred that aborted the archive.

The archive action uses these built-in variables to populate a custom table called `Archive_Audit` that has been created in the database hosting the Optim directory. The column names for this table are same as the PST variable names that are shown in Table 9-7 on page 396.

The `Restore_Info` and `Restore_Info_Detail` tables provide information to perform the following types of audits:

- ▶ When was the restoration request raised?
- ▶ Who executed the restoration request?
- ▶ What was the restoration request type and its associated filter conditions?

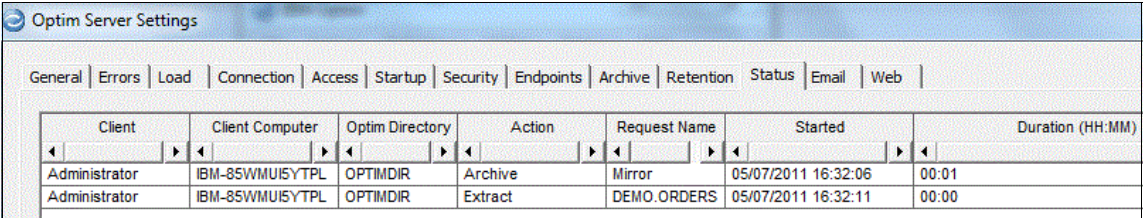
## 9.9 Monitoring

With multiple Optim server instances, it is important to keep track of archival jobs. Use the `rtserver list` command or the Status tab of the Optim control panel to list jobs that are currently running on the instance of the Optim server. Example 9-7 shows a sample of the output from this command on a UNIX terminal.

*Example 9-7 rtserver list command output*

Active	Client	Computer	Pst Dir	Action	Request
3629	optim1	OptimSVRA	OPTIMDIR	Scheduling	Mirror
3635	optim1	OptimSVRA	OPTIMDIR	Extract	DEMO.ORDERS

Similarly, you can track the running archival jobs on a Microsoft Windows terminal using the Status tab of the Optim Server Settings window, as shown in Figure 9-7.



Client	Client Computer	Optim Directory	Action	Request Name	Started	Duration (HH:MM)
Administrator	IBM-85WMUISYTPL	OPTIMDIR	Archive	Mirror	05/07/2011 16:32:06	00:01
Administrator	IBM-85WMUISYTPL	OPTIMDIR	Extract	DEMO.ORDERS	05/07/2011 16:32:11	00:00

*Figure 9-7 Tracking archival job status on Microsoft Windows*

In 9.7, “Restoration job execution” on page 390, the Archive\_Audit table was created to track archive jobs that were executed. Also, you can use this table to get the status of archival jobs that are currently running and finished for a given date and time. Run a simple SELECT query on the Archive\_Audit table with a filter on PST\_START\_DATETIME and PST\_ACTION\_TEXT to obtain the desired status of archival job executions in progress or finished. See Example 9-8.

*Example 9-8 Obtaining archival job status from Archive\_Audit*

```
$ db2 "select
pst_archive_id,pst_start_datetime,pst_current_datetime_char,pst_total_rows_extr
acted,pst_process_status from archive_audit where
PST_ACTION_TEXT = 'AftExtProcess' and
date(PST_START_DATETIME)=date('2011-05-05')"
```

PST_ ARCHIVE_ ID	PST_ START_ DATETIME	PST_ CURENT_ DATETIME_CHAR	PST_ TOTAL_ROWS_ EXTRACTED	PST_ STATUS
-----	-----	-----	-----	-----
31	2011-05-05 19:23:17	2011-05-05 19:23:17	471	0
32	2011-05-05 19:23:20	2011-05-05 19:23:20	66	0
33	2011-05-05 19:23:31	2011-05-05 19:23:31	290	0
34	2011-05-05 19:23:31	2011-05-05 19:23:32	450	0

## 9.10 Reporting

You can run a simple query on the Archive\_Audit table with filters on PST\_ACTION\_TEXT (AFTEXTPROCESS and BEFEXTPROCESS) and PST\_CURRENT\_DATETIME\_CHAR to obtain the desired report on the archival jobs that were executed. When an archival job finishes, PST\_CURRENT\_DATETIME\_CHAR contains the time stamp for when the job finished. The sample query that is shown in Example 9-8 is also an example that was used to generate reports on finished archival jobs.

Similarly, to create reports on restoration jobs, use the Restore\_Info and Restore\_Info\_Detail tables to generate reports on restoration jobs. For example, if you want to get the list of archive files and the corresponding number of records that were extracted for all selective restore request types, Example 9-9 shows a query on Restore\_Info and Restore\_Info\_Detail tables to generate this type of a report.

*Example 9-9 List archive files and rows that were extracted from a restore request*

```
$ db2 "SELECT substr(A.RESTORE_REQ_NUM,1,3) as RESTORE_REQ_NUM,
substr(B.ARCHIVE_FILE,1,73) as ARCHIVE_FILE, substr(B.ROWS_EXTRACTED,1,3) as
```

ROWS\_EXTRACTED from RESTORE\_INFO A LEFT JOIN RESTORE\_INFO RESTORE\_INFO\_DETAIL B  
ON A.RESTORE\_REQ\_NUM=B.RESTORE\_REQ\_NUM AND A.RESTORE\_TYPE='SELECTIVE'"

RESTORE_REQ_NUM	ARCHIVE_FILE	ROWS_EXTRACTED
61	e:\temp\book\archives\GENERAL\BUSINESS\GENERAL_BUSINESS_2008-12_19.AF	0
61	e:\temp\book\archives\GENERAL\BUSINESS\GENERAL_BUSINESS_2009-03_20.AF	4
61	e:\temp\book\archives\GENERAL\BUSINESS\GENERAL_BUSINESS_2009-04_21.AF	0

## 9.11 Tiering

At the beginning of this chapter, we referred to the importance of knowing the length of time that the data must be retained after it is archived from the source database. In this example, the business requirement is to store archived data on disk for one year and, after a year, to move it to tapes and retain it for another three years.

The InfoSphere Optim Data Growth Solution on distributed platforms has the capability to integrate with IBM Tivoli Storage Manager. Although the InfoSphere Optim Data Growth Solution treats Tivoli Storage Manager as a backup device, you can use this integration to tier archive files from disk to tapes.

The integration requires following minimum configuration:

- ▶ File recall path
- ▶ File space prefix
- ▶ Management class
- ▶ Retention after archive
- ▶ Retention after recall
- ▶ Retention policy: Automatically delete file

For more information about configuring this integration, refer to Chapter 10 of the *IBM Optim Archive User Manual*.

When using a tape backup device other than Tivoli Storage Manager or NetWorker, you can move archive files from disk to tape using appropriate vendor software and remove the files from disk. In this case, you must return the file to its original disk location, as registered by the InfoSphere Optim Data Growth Solution, for restoration.

For this example, consider a logical workflow, as outlined in Figure 9-8 on page 400, where vendor software has been used to move archive files from disk to tape media.

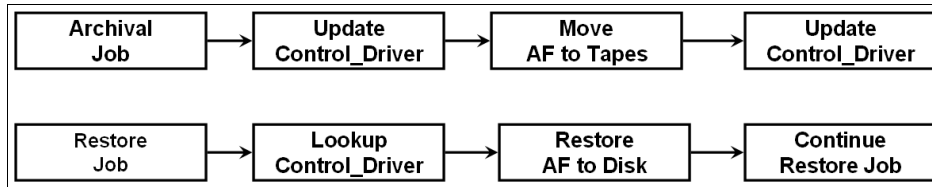


Figure 9-8 Workflow for moving archive files with vendor software

It is imperative to control the storage tiering information in the Control\_Driver table. Doing this helps to control the restoration of data from archive files by ensuring that it is brought to its original path location before executing a restoration process. This process requires that you add the columns in Table 9-9 to the Control\_Driver.

Table 9-9 Additional columns in the control table for using vendor archiving software

Column name	Description
STORAGE_TIER	DISK or TAPE
ARCHIVE_FILE_SIZE_TAPE	Size of archive file as reported by tape storage management system
TAPE_MOVE_DATE	Date on which archive file was moved from disk to tape
TAPE_LABEL	Label of the tape volume on which the archive file resides

In step 4 on page 390 of 9.6, “Archival job execution” on page 387, after successful completion of the archival job, update Control\_Driver with the additional value STORAGE\_TIER=DISK.

Based on your retention policy for archive files that are aged on storage disk media, you can either automate the movement to tape media or execute the movement manually. In either case, follow these steps:

1. Identify the set of archive files that have aged on the disk tier according to the retention policy.
2. Use the vendor software to move the listed archive files to tape and, for each archive file, update Control\_Driver with the following information:
  - a. Size of the archive file as reported by the vendor software managing the tape subsystem
  - b. Volume label for the tape to which the archive file was moved
  - c. Date on which the archive file was moved to tape

Capturing the size of the archive file on disk as reported by the file system and the InfoSphere Optim Data Growth Solution ensures the file size reconciliation. Recording the size of archive files on the tape subsystem allows you to create a report on the total size of the archive files on the tape subsystem.

It is important to restore the archive files to their original location on disk before you run the restoration job, because the InfoSphere Optim Data Growth Solution will look for archive files according to their registry entries, which point only to a disk location. A restoration job that is executed before the files are restored to their original disk location will fail.

## 9.12 Restarting Optim jobs

In the event of any failure during archival and restoration processes, check the error logs for messages that were reported by the Optim server. Fix the errors and rerun the archival or restore process manually using filter conditions for that particular job. The filter conditions are captured in the control tables and logs that were generated by the earlier failure run.

## 9.13 Backup and disaster recovery

When running operations, it is extremely important to back up data and configuration artifacts to recover from failure events. The Recovery Time Objective (RTO) and Recovery Point Objective (RPO) must be agreed to and defined in the service-level agreement (SLA) to achieve the necessary operating conditions in disaster events at the primary site.

Consider backing up the following important Optim server configuration information that will allow you to redeploy Optim servers in the event of crash and failure events:

- ▶ The `pstserv.cfg` and `pstlocal.cfg` for every instance of Optim server on UNIX platform.
- ▶ The `pstrt.cfg` of the Optim server on the Microsoft Windows platform.
- ▶ The `rtsetenv` for every instance of Optim server on the UNIX platform.
- ▶ Use the **export** utility from the Optim workstation to regularly back up all the objects in the metadata repository (Optim directory).

Follow these recommendations to back up archive files:

- ▶ The InfoSphere Optim Data Growth Solution integrates with Tivoli Storage Manager to provide archive files that can be moved from disk to tape media and also to create a duplicate copy of the same archive file on the tape media. You can use this feature to create backup copies of archive files. The duplicate copy can be sent to an off-site tape vault for disaster recovery purpose.
- ▶ You can also choose to use Tivoli Storage Manager features to duplicate tape volumes so that when Optim requests that Tivoli Storage Manager restore an archive file and the tape volume is corrupted, Tivoli Storage Manager automatically uses the duplicate tape volume to restore the data.
- ▶ If you have vendor software other than Tivoli Storage Manager or NetWorker, use it to create backups of archive files on both disk and tape media.

In the case of disaster events that require archival operations to continue from a disaster recovery (DR) site, it is important that the source database is online at the DR site and available according to agreed-to RTO and RPO parameters. The same parameters must apply to the InfoSphere Optim Data Growth Solution also. A DR plan must be in place to ensure the successful start-up of Optim servers on the DR site and continued operations.

Consider the following points when using the InfoSphere Optim Data Growth Solution in a DR situation:

- ▶ Have a policy in place for tape vaulting of duplicate archive files at an off-site location.
- ▶ Also, you might choose to have an exact deployment of InfoSphere Optim Data Growth Solution in the DR site that is identical to the deployment in the primary site:
  - The DR site stays in a passive state because of business conditions considered for DR site planning.
  - Any changes that are made to the deployment configuration must be replicated to the DR site.
- ▶ You can also choose to take a storage flash copy backup of the GPFS mount points holding the archive files and ship it to the DR site deployment.
- ▶ Send the backup copies of the Optim server configuration files to your DR site regularly so that both deployments are in sync.
- ▶ Any new archives that are created using the DR site implementation must be moved back to the primary site and registered appropriately with the Optim servers.

Refer to *Useful Disaster Recovery Definitions and Concepts* on the IBM Redbooks website for more details about DR concepts:

<http://www.redbooks.ibm.com/abstracts/tips0047.html>





## Optim performance considerations

In this chapter, we discuss both the settings and strategies that are used to achieve the best possible data throughput using the InfoSphere Optim Data Growth Solution (Optim). We assume that an appropriately sized architecture that is capable of satisfying previously defined business requirements exists.

Most of the settings and strategies described herein are valid for Optim in Linux, UNIX, Microsoft Windows, and z/OS platforms. Depending on the software component that is being discussed, the method that is used to make changes to specific settings might differ between platforms. To understand how to change settings on a specific platform, refer to the manuals for that platform.

## 10.1 Overview

It is often necessary for Optim to access tremendous amounts of data that might be stored on multiple servers, on multiple platforms, and in separate database types. Non-relational data is often part of the data model. Geographically distributed data centers also might be involved.

Archiving data from large databases across heterogeneous data sources requires close attention to the design and scrutiny of the Optim software settings that affect performance.

Generally speaking, the following factors determine Optim performance:

- ▶ Client database server capacity and load
- ▶ Client network speed, capacity, and load
- ▶ Properly maintained and tuned client databases
- ▶ Proper use of primary keys
- ▶ Properly indexed client tables
- ▶ Disk subsystem performance
- ▶ Properly architected and sized Optim servers
- ▶ Properly designed and configured Optim software solution
- ▶ Optim software performance-related settings
- ▶ Optim Connect software performance settings

For more details about the information and settings that are discussed in this chapter, refer to the *IBM Optim Common Elements Manual* and the *IBM Optim Connect* online manual, which are delivered with the product.

## 10.2 Considerations for processing large data volumes

Getting the performance that is expected from InfoSphere Optim Data Growth Solution is analogous to building a house. A solid foundation is required with strong walls that are bolted to the foundation and a solid roof that is strapped to the walls. After those three areas are covered, decisions about insulation, windows, heating and air conditioning, and so on can be adjusted based on requirements. In the case of a house, the remaining requirements are driven by climate. For data archival and retrieval, the requirements are driven by data volume and latency.

This chapter assumes that the following components are already in place:

- ▶ An appropriately sized Optim architecture
- ▶ Properly tuned source databases
- ▶ Appropriately indexed source database tables

- ▶ Sufficient network bandwidth
- ▶ Adequate disk subsystem performance

## 10.2.1 Using the current release of Optim software

It is important to run on reasonably current releases of all enterprise-level software. Optim is not an exception. Newer releases offer new and enhanced features, including performance enhancements. For example, Optim Version 7.1.1 for Linux, UNIX, and Windows platforms demonstrated a significant improvement in archive performance over previous releases. The current release of InfoSphere Optim Data Growth Solution at this writing is 7.3.1. Specific information about releases and updates is located at this website:

<http://www-01.ibm.com/software/howtobuy/passportadvantage/index.html>

The current release of InfoSphere Optim Data Growth Solution for z/OS at this writing is 7.1.0. Specific information about releases and updates is located at this website:

<https://steamboat.boulder.ibm.com/webapp/ShopzSeries/ShopzSeries.jsp?action=signin>

## 10.2.2 Optim archive parallel processes

The single biggest factor that affects the volume of data that Optim can archive in a given period of time is the number of efficient parallel archive requests executed. During the early phases of a project, business requirements are written. Those requirements are considered during analysis of the systems, network, and databases. Based on the business requirements and analysis, an architecture is decided and sizing estimates are made.

The architecture and sizing process includes growth estimates. The architecture is then designed to scale appropriately as the client's business data grows and ages.

During the design phase, testing is performed to determine if the expected amount of throughput is within expectations. The results of that testing determined the number of parallel processes required. In the example that was used throughout Chapter 8, "Design and configuration" on page 353, the number of processes needed for the Orders and Details tables is four. The Customers and Sales tables require one process each. Parallel processes can be added or removed as necessary to satisfy business requirements. The only limitations are that the number of processes must not exceed the capacity of the servers on which they run, and that the source database is able to satisfy the queries from those processes efficiently.

Optim uses selection criteria or more complex SQL to determine what data is archived by each parallel process. Using the aforementioned example of the Orders and Details tables, data is archived by month. To process a single month's worth of data using four parallel processes, data selection must use criteria that subdivides each month into four near-equal parts. In the case of the Orders and Details tables, each parallel process can cover approximately seven days of each month. Other possibilities include account number ranges, transaction type, and so on.

**Timing:** Optim archive requests can be processed during normal business hours without an adverse effect to production databases provided that the databases can absorb the workload. Assuming that the candidate archive data is in a static state, the risk of archiving data in a state of change is extremely low. Running processes during, or on the edges of, prime business hours can significantly increase the amount of data that can be archived over a given period of time.

Running delete requests during normal business hours requires additional evaluation, because delete requests are more resource intensive and might result in resource locks.

### 10.2.3 Selection criteria

Optim uses selection criteria or more complex SQL to determine what data to select for archiving. This feature is powerful and supports numerous advanced SQL capabilities, including subselects. Subselects and other complex queries can be expensive to run. Therefore, you must analyze subselects and other complex queries for performance-related implications and tune them before you use them.

Depending on the circumstances and requirements, it might be beneficial to create *control tables* to facilitate archive data selection. For more information about this topic, refer to 8.4.11, "Complex data selection requirements" on page 372.

### 10.2.4 Archive file indexes

Optim has the option of creating indexes for archive files. You need to build these indexes based on how the data will be accessed after it is archived. For more information, refer to 10.4.2, "Archive file indexes" on page 418.

## 10.2.5 Optim archive file collections

Optim uses *collections* to organize archive files into logical groups for easy identification and reporting. Collections also play a role in the performance of accessing archive files. For more details about archive file collections and how they affect performance, refer to 10.4.3, “Archive file collections” on page 418.

## 10.2.6 Optim process performance reports

Optim provides reports that contain detailed information about archive and delete processes that help tremendously with identifying bottlenecks and determining the appropriate performance settings. The statistics are shown by table and include detailed information, such as the rows per second, number of database connections, usage of keys, time spent in the database management system (DBMS), and other relevant information. You can see a sample archive report in 8.4.1, “Optim processing reports” on page 365.

## 10.2.7 Optim relationship index analysis

Optim has a built-in function called *relationship index analysis*. It analyzes the relationships between tables (both DBMS and Optim relationships). Based on the intended access, Optim determines the index requirements. The relationship index analysis function displays the index requirements for each relationship and indicates whether those requirements are satisfied and to what degree. You can use this information to add or adjust indexes as necessary to improve performance.

Optim automatically selects the strategies for accessing data based on indexes, relationships, SQL commands, and other criteria. Optim usually makes the best possible selection for efficiently accessing data. However, the table access methods that are chosen by Optim can be overridden if necessary.

When viewing Optim process performance reports, it is important to review the data access strategies that are used and ensure that those strategies are the best choices for performance. For example, if a table is accessed using a key lookup, but the majority of the data is selected for archival, manually forcing a table scan might be a better choice for performance.

For more details about the information and settings that have been discussed in this section, refer to the *IBM Optim Common Elements Manual* that is delivered with the product.

## 10.2.8 Optim archive processing steps

Optim has a feature called *Show Steps*. You can use this function to view the exact steps that Optim will take for an archive process, the reasons for taking those steps, and whether selection-related criteria are used. This tool is excellent for seeing exactly what will happen before running an archive request. Any instances where tables are revisited (processing cycles) are revealed. If they are inappropriate, those cycles can be removed to improve performance and prevent you from archiving more data than was intended. Conversely, any tables that will not be traversed are also identified. Figure 10-1 shows a sample report.

```
EXTRACT STEPS

Step 1:
Extract Rows from Start Table OPTIM.PSTPIADM.OPTIM_CUSTOMERS. Selection Criteria and/or Statistical
Controls are used, these determine rows selected.

Step 2:
Extract Rows from OPTIM.PSTPIADM.OPTIM_SALES which are Parents of Rows Previously Extracted from
OPTIM.PSTPIADM.OPTIM_CUSTOMERS in Step 1 to satisfy an RI rule using Relationship RSC.

Step 3:
Extract Rows from OPTIM.PSTPIADM.OPTIM_ORDERS which are Children of Rows Previously Extracted from
OPTIM.PSTPIADM.OPTIM_CUSTOMERS in Step 1 using Relationship RCO.

Step 4:
Extract Rows from OPTIM.PSTPIADM.OPTIM_DETAILS which are Children of Rows Previously Extracted from
OPTIM.PSTPIADM.OPTIM_ORDERS in Step 3 using Relationship ROD.

Untraversed Table(s):
OPTIM.PSTPIADM.OPTIM_ITEMS

Untraversed Relationship(s):
RID
```

Figure 10-1 Option Show Steps report

## 10.2.9 Optim table traversal options

Optim controls how tables are traversed based on their relationships by using traversal path options, which we described in 6.1.2, “Data-driven engine” on page 215. By default, the option is set to allow traversal from children tables up to any parent tables. Optim automatically traverses from parent to child without setting any options. To improve performance, we suggest that the options to traverse from child to parent are only set for those relationships where that behavior is required. For more information, refer to the *IBM Optim Common Elements Manual* that is delivered with the product.

## 10.2.10 Deleting data after archival

Deleting data from database tables after archival is performed by one of two methods:

- ▶ Run Optim archive delete requests.
- ▶ Drop the partitions from the tables from which the data was archived.

Archive delete requests use archive files to delete data from database tables. The tables that are targeted by the delete request must be in the same database from which they were archived.

Optim delete requests run DBMS SQL DELETE statements to remove the rows from the tables that are eligible for deletion. Therefore, Optim cannot delete data from databases any faster than the database can process the SQL DELETE statements that are submitted to them.

Often, data is archived as part of *complete business objects*, but specific tables might be set to *not* have their data deleted, for example:

- ▶ The *start table* might be a master table from which data must not be deleted, or deletion occurs on a separate schedule and uses separate rules.
- ▶ For data retrieval purposes, it might be helpful to archive static tables that contain reference data that is not intended for deletion, for example, tables that contain actuarial information.

The following list shows the most significant considerations that affect the performance of delete processes:

- ▶ The tables from which data is deleted needs to have primary keys that, by definition, are unique to each row. Optim can delete data in tables where no primary keys are defined or where duplicate keys exist. However, there are significant performance implications.
- ▶ Delete processes need to be run during periods of low activity to prevent DBMS resource contention and locks. You can see the specific settings that cause contention and affect locks in 10.2.13, “Optim performance parameter settings” on page 413.
- ▶ Delete requests can be run in parallel in the same manner as archive requests. Like archive processes, delete processes also produce detailed performance reports. You can use these reports to tune the delete process.
- ▶ The commit frequency, number of database connections, and delete buffer size affect the performance of delete processes. We discuss these areas in 10.2.13, “Optim performance parameter settings” on page 413.

- ▶ Certain databases support array deletes. Optim automatically takes advantage of this feature, provided that the tables are defined with primary keys.
- ▶ Dropping partitions is the most efficient method for deleting large volumes of data from the tables.
- ▶ Optim compares each row in the archive files with its matching row in the database to ensure that they match before deletion. Large objects (LOBs) are also compared. These settings increase run time. Using selection criteria that guarantees that only static data is archived might permit you to turn off one or both of these compare settings. Turning off one or both of these compare settings results in a significant performance increase.

### 10.2.11 Partitioned database tables

The most efficient way to delete extremely large volumes of data is to archive from tables that are partitioned by date. The data in any given partition contains only that data that will be archived by the current archive requests. After the archive processes complete, the partitions are dropped. The prerequisite of using this approach is that the table must be designed as a partitioned table.

In our example data model, the Orders and Details tables contain transactional data. If they are partitioned by date range, the partitions that contain the previously archived data can be dropped after the archive process completes.

### 10.2.12 Restoring data from archive files

Optim has robust features that allow the restoration of data back to database tables after the database tables have been archived and deleted. Typically, only relatively small quantities of data are restored back to reporting databases. However, if archive files are used to restore large volumes of data, performance can be a concern. In those cases, all of the factors that were discussed in 10.1, “Overview” on page 406 apply. Additionally, all generally accepted database practices that relate to restoring data apply, for example, making the decision about whether to drop indexes before loading data and then rebuilding the indexes after the load is complete.

The following specific Optim restore settings and options affect performance:

- ▶ Loading versus inserting data:
  - For database loads, Optim first converts archive files into a format that the target DBMS uses to perform the load. This process takes time, but after it completes, the DBMS' native loader utility is given control. When using Optim, the load option is faster than the insert option for extremely large volumes of data.
  - Optim uses SQL to insert data into the target DBMS. Unlike the load, there is no front-end conversion process. Therefore, an insert process might be faster than a load process, depending on the number of rows. Typically, an insert of 250,000 to 500,000 rows is the range where using a load process becomes more efficient.
- ▶ Compression of subset extract files

If a subset of data in an archive file is selected for restore, Optim creates the extract files to store the subset. Compressing those files increases run time, but the compression saves disk space, and vice versa.
- ▶ Commit frequency

We discuss this setting in 10.2.13, "Optim performance parameter settings" on page 413.
- ▶ Lock tables

We discuss this setting in 10.2.13, "Optim performance parameter settings" on page 413.
- ▶ Process file attachments

This setting determines whether file attachments that are stored in the archive file are restored. If file attachments are not needed, resources can be saved by not restoring attachments.

### 10.2.13 Optim performance parameter settings

Optim has numerous settings that have a direct impact on performance. The settings that are described in this section are located in separate areas within Optim and can vary between OS platforms. Several of the settings are Optim software configuration settings, several of the settings are part of Optim requests, and other settings belong to Optim objects that are used by the requests. You can see the locations of the settings in the *IBM Optim Installation and Configuration Guide* and the *IBM Optim Common Elements Manual*.

The following list shows the Optim performance parameter settings:

► Archival of database objects:

- By default, Optim archives database objects by capturing the Data Definition Language (DDL), or the actual object is placed in the archive files. These objects include primary keys, relationships, indexes, partition functions, procedures, and so on.
- Several of these objects can affect performance significantly. We suggest that you do not select these objects when archiving data. If the objects are needed to create tables in the future, run a special archive that only archives the objects and not the data.

► Compression of archive files

The archive files can be compressed or left uncompressed:

- Compress the archive files to save disk space, which increases the run time.
- Leave them uncompressed, which uses more disk space, but shortens run time.
- Compress selected tables in an archive file, which shortens run time and saves disk space when compression is only applied to tables that are highly compressible.
- Compress archive files after the archive process is complete, which shortens the archive run time, but uses more disk space temporarily and uses more resources overall.

► Maximum database connections:

- This setting determines the number of parallel database connections. The number of connections that are actually used can differ from the setting. For more details, see the *IBM Optim Common Elements Manual*, which is delivered with the product.
- Start with two database connections and test by increasing in multiples of two until performance improvements diminish.
- This setting affects each parallel process that is actively running.
- Monitor the source DBMS server for over-utilization.

► Fetch buffer size:

- This setting determines the size of the buffer that is returned to Optim from a DBMS fetch.
- The default of 512K is usually the best setting. The initial setting is multiplied by the maximum database connections. If the setting is 512K and four database connections are specified, the fetch buffer size is automatically set to 2048K (512K assigned to each connection).

- ▶ Delete buffer size:
  - This setting determines the delete buffer size and only applies to array deletes. The setting is also affected by commit frequency. For more details, see the *IBM Optim Common Elements Manual*, which is delivered with the product.
  - The default of 512K is usually the best setting. The initial setting is multiplied by the maximum database connections. If the setting is 512K and four database connects are specified, the fetch buffer size is automatically set to 2048K (512K assigned to each connection).
- ▶ Parent and child key lookup limits:
  - This value determines the number of keys that are searched for during a single DBMS request. The possible values are 1 to 100.
  - This setting requires experimentation in the actual environment.
  - Typically, if more keys are returned from a single request, performance improves.
- ▶ DBMS commit frequency:
  - This setting affects database deletes and restores. The possible values are 1 to 999,999.
  - More frequent commits reduce the chance of failed transactions because of resource locking, but increase run time.
  - Less frequent commits increase the chance of failed transactions due to resource locking, but improve performance.
  - This setting must be evaluated on an application-by-application basis.
- ▶ Delete process *lock tables* option
 

Locking is requested at the table level. This option prevents all access to a table while that table is being processed.

## 10.3 Non-relational files

To understand the performance implications of accessing non-relational files, it is important to understand the components that are involved and the purpose of each component.

Files that Optim cannot recognize as DBMS relational tables are classified as non-relational or *extended data sources*. These files include flat files, text files, VSAM, IMS, ADABAS, and others. There are relational tables that also require either DB2 Connect or Federation Server for access. They include DB2 on

iSeries® and DB2 on z/OS when accessed from Linux, UNIX, and Microsoft Windows platforms.

Optim cannot connect directly to non-relational files. Therefore, middleware in the form of Optim Connect or Classic Federation Server for z/OS is used to make that connection. The purpose of these software components is to make the non-relational files appear as relational data sources and to expose the data to Open Database Connectivity (ODBC). Then, the native ODBC connectivity that is supported by either DB2, Oracle, or SQL Server can connect to those data sources and access the data by means of database views.

Compared to accessing relational data from a DBMS where advanced buffering, multi-threading, and a host of other performance features are available, non-relational file access has slower data transfer rates. Additional latency is introduced because of the middleware layers. You can, however, still achieve acceptable performance. Note that the data is not in a native database and cannot be expected to perform as though it were in a native database. Therefore, it is important to understand this difference and make allowances during the project's design phase.

**Optim Connect:** Optim Connect together with open data manager (ODM) is required for accessing archive files using ODBC and Java Database Connectivity (JDBC). It also has robust functionality for accessing non-relational files with the exception of Integrated Data Base Management System (IDMS) and CA Datacom.

Because Optim Connect together with ODM is required to access Optim archive files and it can also process most types of non-relational files, it is the suggested middleware of choice unless connections to IDMS or Datacom are needed.

### 10.3.1 General performance guidelines

All of the expected considerations for efficient data access also apply here. The aforementioned middleware must be installed on a server that is not constrained by CPU, memory, I/O, or network speed. When there is a significant constraint, it negatively affects performance. It is not uncommon for processes that were running for hours to be reduced to minutes after eliminating the constraints.

### 10.3.2 Processing non-indexed file types

If non-relational data is not indexed, such as flat files for example, all of the data must be read from the first record to the last. Therefore, running parallel archive

processes for those types of files might not offer significant performance gains unless buffering is supported. In the case of sequential files on z/OS, buffering is supported and parallel processes may improve overall data throughput.

For file types that support indexes, such as VSAM KSDS files on z/OS for example, parallel archive processes can significantly increase throughput when processing large amounts of data.

### 10.3.3 Trace and log settings

It is important to ensure that tracing and unneeded logging options are turned off for all the various processing components. While trace and certain log settings are typically set to “off” as the default, it is not uncommon that, during initial development and testing, certain tracing and unneeded logging options are turned on and forgotten. Later, those settings can be propagated to production. Trace settings are located at the native database level, ODBC, and the middleware. In certain cases, there might be multiple types. A good example is Optim Connect, which has four locations where various traces and logs are set. We suggest that you review all of the various settings and control files for all middleware components to ensure that no unnecessary traces or logs are produced.

### 10.3.4 Performance parameter settings

Specific performance settings for Optim Connect and Classic Federation Server vary based on the environment. We suggest starting with the default settings. If necessary, make modifications and test the results in a controlled environment. 10.4.4, “Optim Connect performance parameter settings” on page 421 shows the Optim Connect settings that affect performance.

**Performance settings:** The performance settings that are shown in 10.4.4, “Optim Connect performance parameter settings” on page 421 are settings for accessing data from the Optim archive files. Those specific settings might not produce performance improvements for accessing the non-relational files that are described in this section.

## 10.4 Archive file access

The final result of an Optim archive process is that all data that is infrequently used is removed from the databases and stored in the archive files. Access is infrequent, but still required to answer business questions and respond to audits.

Access is accomplished using a combination of ODBC and JDBC, Optim Connect, and ODM. There is also the option to restore data back to a database.

### 10.4.1 General performance guidelines

Similar to access to non-relational files, remember that the data is no longer in a DBMS and cannot perform that way. The goal is to return data at a rate that meets or exceeds the established SLA.

You must correct any constraints that relate to CPU, memory, I/O, or network speed. The guidelines that apply for trace and log file settings as described in 10.3.3, “Trace and log settings” on page 417 also apply for the archive file access.

### 10.4.2 Archive file indexes

The individual tables that are stored in archive files can be indexed. Any indexes that are created for archived tables need to be based on how the data is accessed. It is likely that the index requirements for archived data are the same as the index requirements for the production database, but not necessarily.

To achieve the desired level of performance, it might be necessary to create indexes on fields that normally are not considered as fields requiring an index. In certain cases, composite indexes are required. The scenario that is described in 10.5, “Analyzing performance with Optim Connect” on page 422 describes how to use the *explain* function in Optim Connect to maximize performance and clearly illustrates this concept.

The time that is required to determine what indexes are necessary to satisfy reporting requirements and to eliminate any indexes that are not needed is worth the investment. The potential results are significant space savings (archive indexes are not compressed) and improved archive access performance.

You can build indexes during the archive process or as independent steps after archive processes complete.

### 10.4.3 Archive file collections

*Archive file collections* are used to organize archive files into a logical order. All archive files in a collection are viewed as if all of the data for a table is a single table. For example, if Table\_A is archived monthly and the resulting archive files are placed in collections by year, all twelve of the Table\_A archive files appear as a single table when queried. Figure 10-2 on page 419 shows this example.

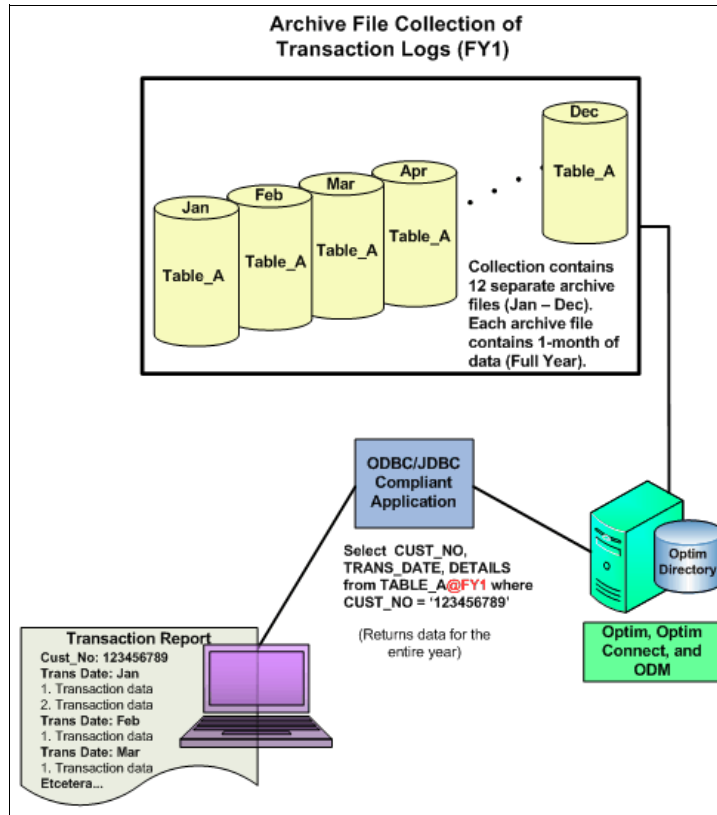


Figure 10-2 Archive file collections

You must control the maximum number of files in an archive file collection. All environments differ; however, a good rule is to maintain the total number of archive files in a collection at around 200. Having large numbers of archive files in a collection increases the number of files that must be opened and read. Memory requirements also increase.

In terms of resource utilization and performance, it is better to have fewer, larger archive files than many small archive files. The maximum number of rows in a single archive file is limited to  $10^9 - 1$ . However, do not attempt to place as many rows as possible into each archive file. Base the organization of archive files by collections on how the business accesses the data that is combined with these performance guidelines.

Archive files can be members of multiple collections. This capability provides the opportunity to create subcategories of collections that might improve performance by limiting the number of archive files that are read. Using the

previous example of Table\_A, the archive files in a yearly collection also can be placed in quarterly collections to produce quarterly reports. There are two benefits:

- ▶ The number of archive files that must be opened to process the quarterly data is reduced by 75%.
- ▶ The SQL logic does not require the execution of any code to limit the rows that are selected by date. All of the data in the selected collection is already in the correct data range.

Figure 10-3 depicts this concept.

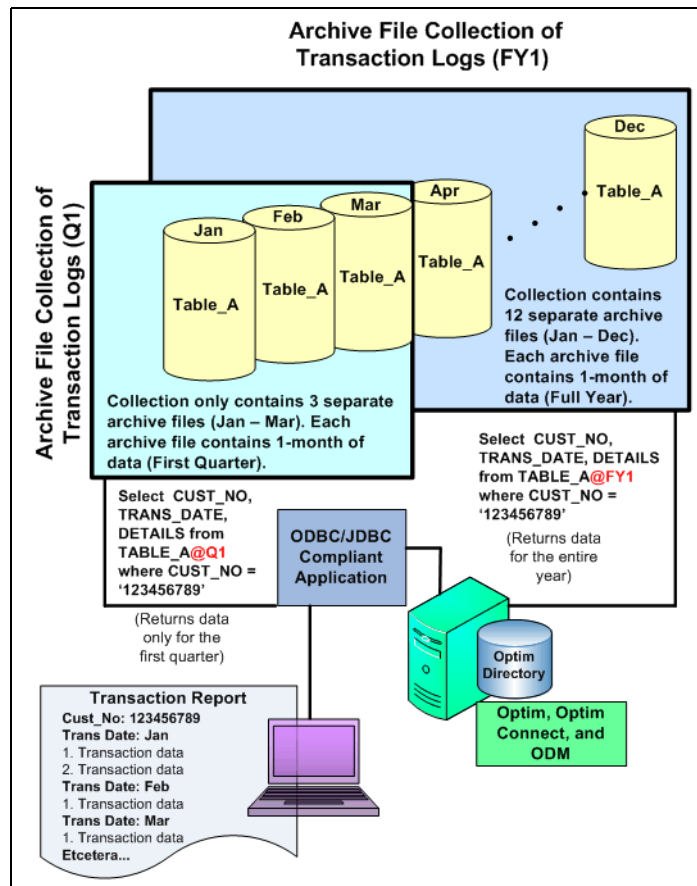


Figure 10-3 Archive file collections with subgrouping

**No storage impact:** Archive files are only stored one time. Therefore, there is no impact on storage when placing archive files in multiple collections.

## 10.4.4 Optim Connect performance parameter settings

Most of the default settings in Optim Connect are considered optimal. There are specific settings that must be changed to improve the performance of accessing archive files using Optim Connect queries and ODBC/JDBC connections.

Figure 10-4 shows the settings and the values that we suggest. These settings are binding-level settings and they only apply to accessing Optim archive files. You can locate these settings under the Tuning tab for each binding. For more information, refer to the *Optim Connect* online manual that is delivered with the product.

Property	Value
DSM max buffer size	100000000
DSM max hash file size	2000000000
DSM max sort buffer size	100000000
DSM mid buffer size	100000000
File pool size	10
File pool size per file	3
File Close on transaction	<input type="checkbox"/>
Use global file pool	<input type="checkbox"/>
Hash buffer size	100000000
Hash max open files	500
Hash primary extent size	0
Hash secondary extent size	0
Hash max extents	16
Hash enable RO	<input type="checkbox"/>
Hash refresh EOF	<input type="checkbox"/>

Figure 10-4 Optim Connect parameter settings

**Accessing data from Optim archive files:** The performance settings that are shown in this section are settings for accessing data from Optim archive files. Those specific settings might not produce the performance improvements for accessing the non-relational files that are described in 10.3, “Non-relational files” on page 415.

## 10.5 Analyzing performance with Optim Connect

In this section, we describe how to use the Optim Connect **explain** command to understand how the tables in archive files are accessed and determine which indexes are not being used and whether indexes must be added to improve query performance. You can use the same commands and functions to analyze any Optim Connect data source, including access to non-relational files.

### 10.5.1 Using the Optim Connect explain command

The **explain** command works similarly to any DBMS explain facility.

The **explain** command is executed from the Optim Connect query processor **nav\_util**. **Nav\_util** (**navcmd** on z/OS) is invoked from a server command line or from TSO under ISPF on z/OS. There are numerous start-up parameters and options. The following command syntax is its simplest form:

```
nav_util execute <data source name>
```

For more detail about all the features and options, refer to the *Optim Connect* online manual that is delivered with the product.

The use case that we describe here accesses two tables, Customers and Orders, that are stored in archive files. The query that is used provides a list of all the clients in the state of New Jersey who spent more than USD10 to have their purchases delivered. The following query is the actual query:

```
SELECT c.cust_id, c.state, o.freight_charges FROM optim_customers c,  
optim_orders o WHERE c.state = 'NJ' AND c.cust_id = o.cust_id AND  
o.freight_charges > 10.00
```

The Customers and Orders tables are implicitly joined on *cust\_id*. Both the Customers table and the Orders table are indexed on that field. To see how efficiently the query is running, we run an explain on the query:

```
EXPLAIN SELECT c.cust_id, c.state, o.freight_charges FROM optim_customers c,  
optim_orders o WHERE c.state = 'NJ' AND c.cust_id = o.cust_id AND  
o.freight_charges > 10.00
```

To simplify the examples, we show the consolidated views of the results. The full results are in XML format and list all tables, columns, row counts, and other pertinent information.

In the first execution (Figure 10-5 on page 423), none of the archive indexes are used. Even though an index exists on the Orders table for the field that participates in the join (*o.cust\_id*), that index is not used because of the

expression that is coded for o.freight\_charges. This query is the most expensive of the examples that are used in this section. It has a cost of 32.147500.

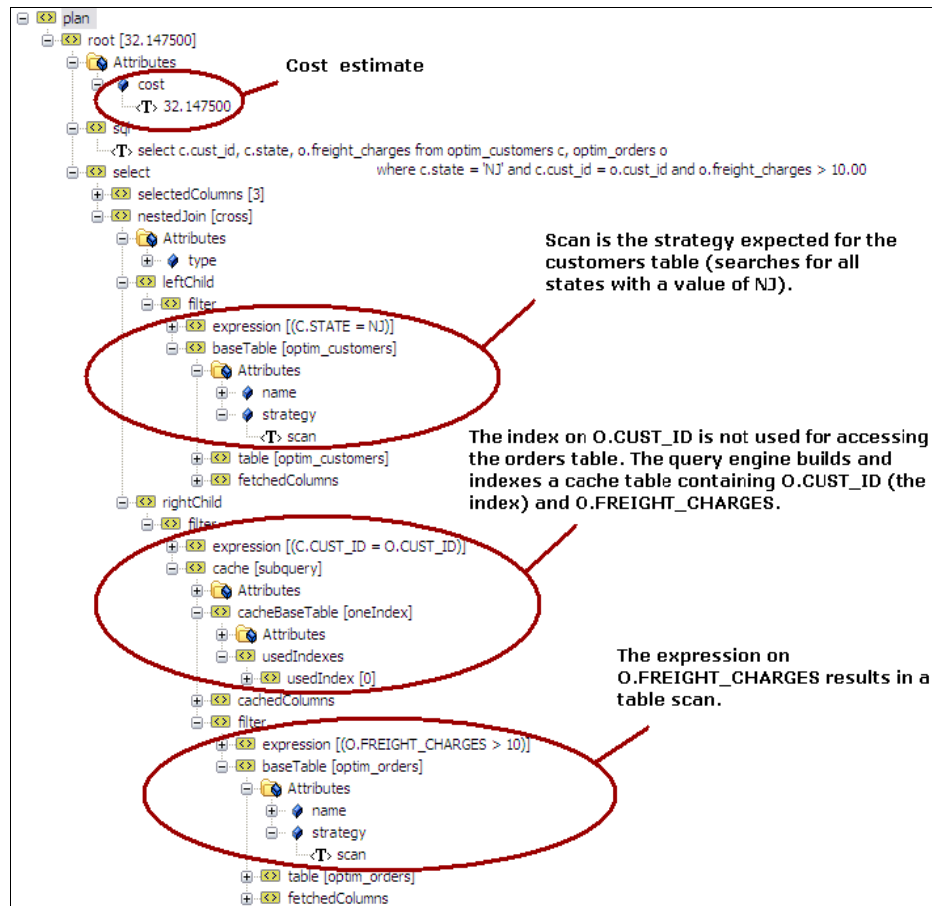


Figure 10-5 Explain result: Index not used

For the second execution, we make changes to improve the performance from the first run. It appears that the table scan resulting from the expression `O.FREIGHT_CHARGES > 10` is the culprit. Therefore, we add an index to the `o.freight_charges` field. We then rerun the same **explain** command. Figure 10-6 on page 424 shows the result.

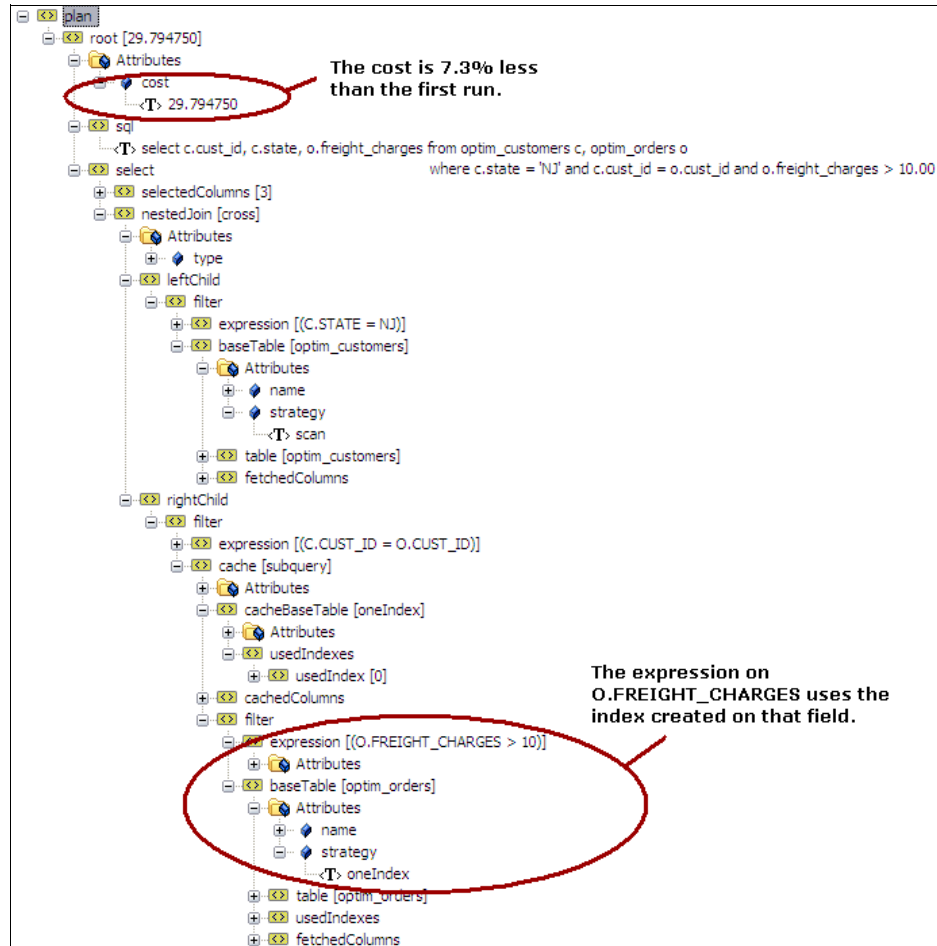


Figure 10-6 Explain result: Index added

The difference between the results of the first execution (Figure 10-5 on page 423) and this execution is that the cost is 7.3% less. This reduction in cost is the direct result of the index that is placed on o.freight\_charges. Although there is an improvement, it is less than desired.

The Optim Connect query engine can only make use of one index per table, which is the reason that the existing index on o.cust\_id is not used. However, this limitation can be overcome by creating a composite index.

In the third, and final, execution, a single composite index is created using the o.cust\_id and o.freight\_charges fields. Figure 10-7 on page 425 shows the results.

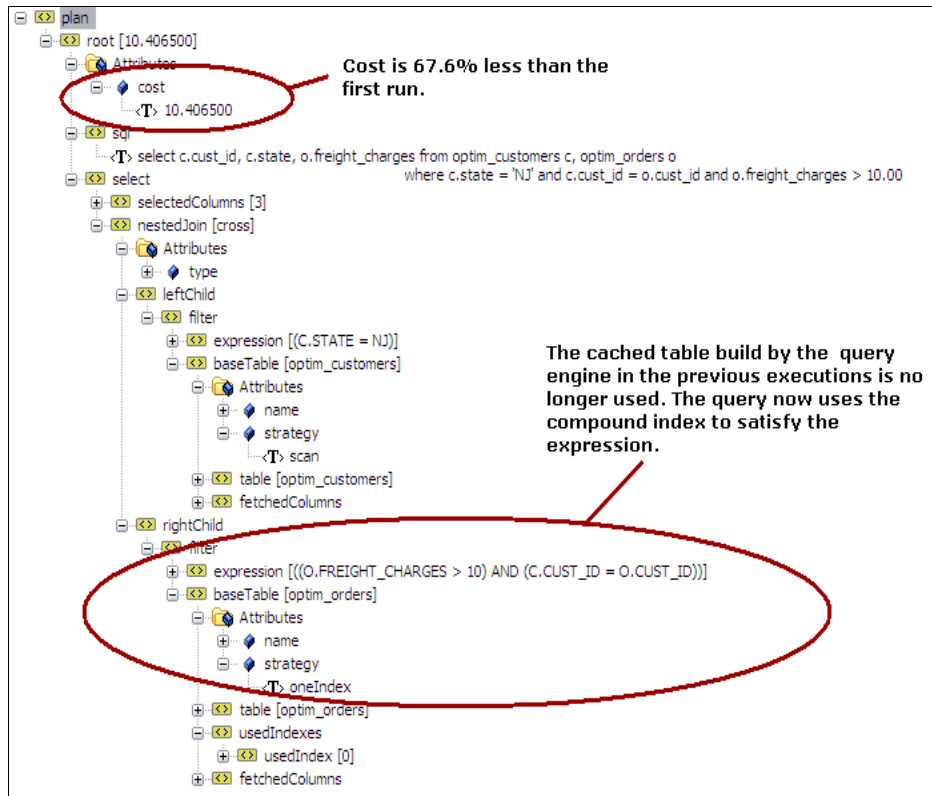


Figure 10-7 Explain result with composite index

Using the composite index produced excellent results. The cost of execution is 67.6% less than the first run and 65.1% less than the second run. Because the Customers table is not indexed on the State column, the cost of this transaction can be reduced significantly more by creating an index on that column. Decisions on indexes are based on whether the business priority is transaction speed and machine resource efficiency or saving disk space. The answer might be a balance between the two resources.

While not part of this example, note that the use of certain functions on an indexed column will prevent its index from being used, for example, **AVG()**, **SUM()**, and **UPPER()**.

The differences between the index strategies that are described here can determine whether queries complete within the established SLA or run seemingly forever.

These are the types of performance improvements that are required when running queries against tables that contain large amounts of data. Queries that are properly tuned return results faster, and they consume fewer machine resources. When balanced with disk space usage (the number of indexes used), queries that are tuned use overall resources more efficiently and might reduce the business operational costs when compared to untuned queries.



# Security

Providing security for its systems and data without sacrificing usability is a challenge that every organization faces. In this chapter, we discuss the IBM InfoSphere Optim Data Growth Solution (Optim) security and certain common and optimal ways to implement practical and functional security controls.

## 11.1 Basic access security model

The model in Figure 11-1 on page 429 provides the common method of implementing security around Optim. It does not require any knowledge of Optim-specific security features to implement it, and it uses the security that is already known and used in your existing environment.

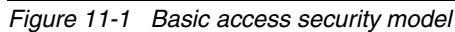
When implementing security with Optim, there are two modes in which Optim is used:

- ▶ Archive creation
- ▶ Access to archived data

Archive creation is typically performed by operations users. These users must not have any requirements to access the data. In many implementations, these users are actually automated batch scripts that execute archive jobs periodically. This method requires no human intervention unless there is a problem with the run.

Data access must be restricted. Users must only be able to access the specific data for which they have access authorization.

The basic security model addresses both of these modes.



Using a basic security model, Optim administrators have access to Optim configurations and archived data that is governed solely by the permissions that have been granted to their individual user IDs.

Chapter 11. Security 429

to production data and archive files, special client and database user IDs might need to be created.

These special user IDs can be available only during defined periods of testing and their usage can be audited by special processes that are defined and controlled by the client's security administrators and implemented in the OS.

User access must also be controlled by user ID-based permissions. Users must not be able to log on directly to any Optim server. Their user IDs grant them appropriate access to applications and database management systems (DBMS) that return the required data. That data is obtained by an application using the authority of functional IDs that are not accessible to a user. Access to data can then be managed by the application while presenting the data to the user.

In this model, an Open Database Connectivity (ODBC) and Java Database Connectivity (JDBC)-compliant application or a database schema is set up to access the archived data using a functional ID that has the authority. Users of the applications or database are then granted restricted access to the application data or schema data model. Therefore, the security to archive data can be managed by existing application or database security mechanisms.

Individual users must have read-only accounts to the archive DBMS and no administrator privileges, because the archived data cannot be modified. Individual users must not be allowed to log on to the server and they must have no direct access to the directories where the archive files are stored. Also, these users must have no access to Optim software files.

If archive file folders are maintained and organized by line of business (LOB), access permissions to each LOB can be controlled separately. This approach allows a more granular level of control.

The following list summarizes the requirements for this model and is intended as guidance for you to implement this security model in a generic environment:

- ▶ Use automated scripts for running archive jobs.
- ▶ Administrators only need access to servers and server software to maintain and monitor the archive jobs.
- ▶ Administration access to Optim software must be monitored and logged.
- ▶ ODBC/JDBC access to archived data must be restricted to only those databases and applications that require the data. You can use a firewall that blocks the IP addresses and ports and only open them to those servers running the applications and database.
- ▶ The applications are configured to access the archived data via JDBC/ODBC.

- ▶ The databases define views that can access the archived data via JDBC/ODBC.
- ▶ The application security limits and restricts access to the data that exists in the archives.
- ▶ The database security limits and restricts access to the archived data views.
- ▶ The functional users access archived data via the applications or database where the security layer manages the access.
- ▶ The functional users must *not* have any access to Optim, Optim servers, database servers, or application servers.
- ▶ Administrator access to applications and database must be non-existent or restricted.

Even though this model is the common method for implementing security around Optim, no company is limited to only this technique. Optim is rich in features and its security can be extended beyond this model in numerous ways. The remainder of this chapter provides more detail about Optim security features for those environments with more complex security requirements.

## 11.2 Optim security architecture

In this section, we provide an overview of the security structure of the InfoSphere Optim Data Growth Solution.

### 11.2.1 Definitions

We define the terminology that is used in this chapter:

- ▶ User: The entity that requires system access.
- ▶ Owner: Same as user (terms are used interchangeably).
- ▶ Privilege: A level of authorized access.
- ▶ Authentication: The method by which a user's privileges are validated.
- ▶ Roles: A group containing one or more users that require specific access privileges.
- ▶ Separation of duties: The division of roles between users. Typically, users are given separate roles so that no single user has full authority over the environment.
- ▶ Access control list (ACL): A list that controls the privileges for the specified users to certain resources.

- Access control domain (ACD): A grouping of security definitions and objects (ACLs, roles, and so on).

## 11.2.2 Standards matrix

There are many standards that provide guidelines about how an organization must implement security. This book is not a security standards document. However, we have identified a few of the more commonly asked about security controls in the standards and listed them in Table 11-1. This chapter discusses security around each of these controls.

Table 11-1 Security controls

Control	Description
Account management	The information system manages the user accounts.
Access enforcement	The information system enforces approved authorizations for the logical access to the system.
Separation of duties	The information system enforces the separation of duties through assigned information system access authorizations.
Audit events	The information system audits the events that are determined by an organization.
Content of audit records	The information system produces audit records that contain required information.
Configuration management	Physical and process monitoring and controls to securely manage the changes to information systems.
User identification and authentication	The information system uniquely identifies and authenticates for all users.
Encryption	The process of transforming information using an algorithm (called a <i>cipher</i> ) to make it unreadable to anyone except those users possessing a key code.

## 11.2.3 Access flow

The implementation of Optim security uses callback methods. When a user initiates a process within Optim, that process first validates the user's privileges. If successful, the process is executed. If validation fails, the process displays the reason for failure as "insufficient privileges".

Optim employs a “deny first” security model, which means that when security is enabled, all users will be denied access unless explicitly allowed.

Additionally, any privilege that is explicitly denied supersedes any privileges that are explicitly allowed. If a user belongs to one or more roles, and any of those roles deny a privilege, that user is denied the privilege.

Figure 11-2 is a flowchart of how Optim handles this access. For every role of which a user is a member, a security flag is checked for “allow”, “deny”, and “none (neutral)”.

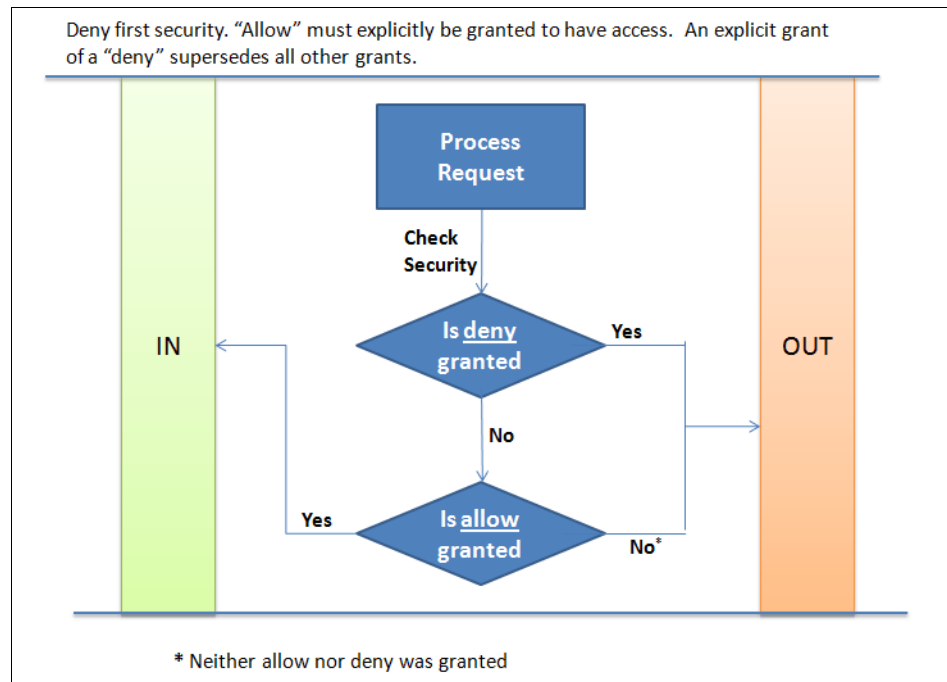


Figure 11-2 Optim access flowchart

This chapter describes the default behavior. You can modify the default behavior by using a custom user-written exit that allows you to apply an additional layer of security to Optim. A custom user-written exit goes beyond the extensive security that we describe in this chapter, to meet any security requirements that are mandated by your company or government regulations. The *Optim Initialization Exit Programmer's Guide* describes writing custom user-written exits.

## 11.3 Authentication and access

There are several areas of security within Optim, including user, functional, object, file, database, and open data manager (ODM). Optim provides security at these levels for all of its components.

### 11.3.1 Account management

Optim uses the existing authentication mechanism that is provided by the operating system of the system where the program is installed. You are not required to maintain any additional user management functions in order to use the Optim software. All references to “users” within Optim are from the perspective of the operating system.

By default, Optim checks the operating system for the owner and *host name* of the user that started the program. On the Microsoft Windows platform, the Microsoft Windows *domain* is used instead of host name, if it is available.

You can modify this default behavior by providing a custom security exit, which is a compiled C program. Using a custom security exit, Optim can authenticate users with an authentication server, for example, Lightweight Directory Access Protocol (LDAP).

### 11.3.2 Functional security

This level of security restricts access to the activities that can be initiated in Optim, such as create archive request or access definition. Both runtime and design-time activities are secured. All activities are controlled individually by *role*. Users that are assigned to a given role have permission to perform any of the activities that are allowed by that role.

Each defined role sets access to each of its related functions to either allow access, deny access, or be neutral (none). “Allow” grants access to users in a role. “Deny” prevents access to users in a role. And, “none” is neutral. By default, neutral denies access, unless the user is a member in another role that is allowed access to a given function.

**Deny function:** Any explicit “Deny” of a function supersedes all explicit allows. If a user is part of more than one role and any of those roles denies access to a function, that user is denied access to that function even if other roles grant access.

### 11.3.3 Object security

This level of security restricts access to the definitions and objects that are created by Optim at design time. Objects are reusable components that are saved in an Optim directory. Objects include access definitions, delete requests, archive requests, and so on. For more information about Optim objects, refer to the *IBM Optim Common Elements Manual*, which is delivered with the product.

Objects are used during run time and design time and can be secured for both. All objects are secured individually as part of a role. Users of a given role have access granted to any objects that are identified by the role.

For every Optim object, you create an ACL. The ACL can contain one or more roles. Each role explicitly allows, denies, or is neutral (specified as “none”) for read, update, delete, and execute for that object. It also allows, denies, or is neutral to modifying the ACL. Users that can modify the ACL are in control of the privileges for other users.

**Deny in a role of an ACL:** Any explicit “deny” in a role of an ACL supersedes all explicit allows for any user of that role. For example, if an ACL defines security for more than one role and any of those roles deny access, a user who is a member of those roles is denied access to those functions even if other roles allow access.

### 11.3.4 File access definitions

This level of security restricts access to data that is stored in archive files. The *file access definition* uses roles to authenticate and validate user privileges. Access can be limited to specific tables and specific columns within those tables. For example, if a table in an archive file contains client names and client salary information, users that do not have a business reason to view salary information can be denied access to that field in the table.

An access level of “deny” supersedes all allows. Therefore, if a user is a member in more than one role and any of those roles denies access to a table or column, that user is denied access.

Managing the security of data in archive files can also include the security-related settings in the Optim Connect software. We provide more detailed explanation in 11.4, “Accessing archived data with open data manager” on page 436.

### 11.3.5 Database security

Optim conforms to the security that is defined by the data source by requiring a user to be authenticated against the data source to which it connects. All data source connections are configured to use authentication tokens that are created at design time and used at run time. You can configure Optim to force users to enter a password every time that a connection is made. Optim can store and reuse the password. The passwords that are stored by Optim can be encrypted. The *Optim Installation and Configuration Guide* describes the process for encrypting passwords in the configuration files.

## 11.4 Accessing archived data with open data manager

Optim's ODM feature enables standards-based access to archived data without the need to restore to a relational database management system (RDBMS). While the archived data is considered near line, Optim provides robust, standards-based client interfaces to make access to archived data as seamless as possible. The ODBC, JDBC, and ANSI SQL 92 standards are supported.

ODM provides facilities to secure access to data. This section is intended to convey the methods and best practices that can be used to secure an ODM endpoint and alleviate security concerns.

**Important:** Consider your deployment environment when you select security options from this book. You can combine the security methods that are described in this book to meet the specific requirements of your operational environment.

This section is not intended to be a tutorial on the configuration of ODM. We intend to communicate the specific concepts and configurations for securing an ODM endpoint. See the following prerequisites section for the required skill sets to use this guide effectively.

### 11.4.1 Prerequisites

Users of ODM security must be familiar with the configuration and operation of the following products and components:

- ▶ Optim for distributed platforms
- ▶ Optim Connect and ODM

- ▶ Operating system administration and security that are associated with your target platform
- ▶ RDBMS user management and federation (For ODM federation only)
- ▶ Optim file access definitions (For ODM and file access definition only)
- ▶ Optim access control domains (For ODM and file access definition only)

## 11.4.2 Open data manager security methods

Three suggested methods exist for securing ODM endpoints:

- ▶ ODM federation with an RDBMS
- ▶ ODM workspace security
- ▶ ODM and file access definitions (FADs)

Each method provides for secured access to archived data. However, the methods differ in the required complexity to configure, deploy, and maintain them, as well as the level of granularity that is available to control access. Evaluate each security requirement use case to determine the best method, or combination of methods, to address the security use case.

In the following sections, we provide a summary of the features and benefits of each security method, as well as provide advice to aid you in selecting the best security model to meet your specific security requirement.

### Federation of open data manager into an RDBMS

For many enterprise organizations, the data access policies and related infrastructure are already in place and managed through the use of an existing RDBMS. Today's modern RDBMSs provide for the federation of external data sources into one or more schema objects within the database. You can use these federation capabilities to expose archived data from within the RDBMS, allowing for the reuse of the existing security models that are in place to access ODM data sources.

In addition to exposing ODM from within an already established secured environment, you simplify the implementation of ODM within the enterprise because access is achieved by using clients that are already deployed and in place. You are not required to install any additional clients, because the native database client is used to connect to the data source.

Access to the data using this technique is presented through the database. Because this approach provides security through the database, the users can expect the performance to be, at most, the performance of native table access.

**Important:** It is imperative to highlight that security model and client reuse are the fundamental benefits of using this technique to secure and roll out ODM. You must educate the users that near-line data is being accessed through their online data repository.

You must manage this expectation up front and educate the user community as to which data is the near-line data that is accessed through federation.

Figure 11-3 shows the advantages and disadvantages of using the existing security model of RDMBS for accessing archive data through ODM.

PROS	CONS
<ul style="list-style-type: none"><li>✓ Leverages existing expertise within the organization for securing data access</li><li>✓ Leverages existing security models already in place within the organization</li><li>✓ Client roll out is greatly simplified as existing database clients are leveraged</li><li>✓ Central management of exposed data through the use of federation techniques</li><li>✓ Ability to provide fine grained security access control through the use of views.</li></ul>	<ul style="list-style-type: none"><li>✓ Performance expectations must be managed</li><li>✓ May require additional security consideration if the endpoint is exposed on a network where unauthorized access could compromise data. (Solved with ODM workspace security and a functional ID)</li></ul>

Figure 11-3 Pros and cons of using the existing security model of RDMBS

Open data manager workspace security

The ODM workspace security provides the ability to secure ODM data sources using credentials that are presented during a connection. The endpoint is secured. Because ODM uses ODBC and JDBC standards, any application that supports these standards can present credentials to ODM. ODM grants or denies access based on how the access control list (ACL) that is associated with the credential and the data source is defined.

Ideally, you use the workspace security in scenarios where the ODM endpoint has to be secured. Common use cases include when the ODM endpoint is exposed outside of a firewall and is not contained within a protected data center. Another use case is when ad hoc reporting access is required within an organization and access control has to be enforced.

**Important:** Note that the ODM workspace security feature is not only useful for enforcing access control for users, but also for applications.

It is common to find use cases where ODM is nothing more than a resource to a front-end application, and all user authentications are provided by the application. In this instance, a functional ID can be granted access to ODM data sources to serve the application. Federation of ODM into an RDBMS is an excellent example of this method. Federation of ODM into an RDBMS also is an excellent example of using a combination of security methods to lock down archived data that is exposed using ODM.

This option introduces a new security model and methods and requires that existing security personnel incorporate them into the existing security policies.

Figure 11-4 lists the advantages and disadvantages of using workspace security to secure an ODM endpoint.

PROS	CONS
<ul style="list-style-type: none"><li>✓ Secures the endpoint directly</li><li>✓ Leverages standards to convey user credentials</li><li>✓ Leverages the Operating System user repository</li></ul>	<ul style="list-style-type: none"><li>✓ Introduces a new security model</li></ul>

Figure 11-4 The pros and cons of using workspace security

Open data manager and file access definitions

ODM and file access definitions provide the ability to use Optim’s file access definitions (FADs) security model in securing ODM. This method of securing ODM brings with it all of the features that are inherent with file access definitions, as well as these features:

- ▶ Parallel access control to archived data for both the Optim Browse Tool and ODM
- ▶ Fine-grained security access control down to the table and column level per archive file

In use cases where this kind of access control is required, ODM and FADs are ideal for securing ODM data sources.

**FADs vs. views:** You also can achieve fine-grained access control to columns and tables by using the ODM federation method by building views within the RDBMS and restricting access to the federated table objects and these views. The FADs differ in that the FAD control is at the archive file level whereas the views filter data at the data source level.

Figure 11-5 on page 440 lists the advantages and disadvantages of using FADs to secure the archive data access.

PROS	CONS
<ul style="list-style-type: none"> <li>✓ Leverages standards to convey user credentials</li> <li>✓ Leverages the Operating System user repository</li> <li>✓ Leverages the Optim FAD security model</li> <li>✓ Aligns access control to archived data regardless of access method</li> <li>✓ Provides fine grained security access at the archive file level.</li> <li>✓ Consistently applies Access Control Domains across archive files</li> </ul>	<ul style="list-style-type: none"> <li>✓ Introduces a new security model</li> <li>✓ Requires the FAD to be applied at the time of Archive file creation</li> <li>✓ The most complicated to set up</li> </ul>

Figure 11-5 Pros and cons of using FADs to secure ODM

## Selecting the best open data manager security method

Selecting the best method to secure ODM is a subjective decision. You must define any and all security requirements for archived data access before you can make any assessment of security deployments.

Consider these questions when selecting a method to secure ODM:

- ▶ Will ODM be exposed on a network segment that is outside the protection of a firewall? For example, will ODM be deployed outside of a firewalled data center?
  - If yes, ODM will be exposed to access by a wide range of users on the immediate network segment. A form of protection might be required.
  - If no, If the firewall filters packets for specific services and IP addresses, ODM can be protected by the firewall. This option assumes that direct access from outside the firewall is *not* required. (This option is useful when ODM federation is a desired security method and the RDBMS resides in the same protected network segment.)
- ▶ Does the enterprise currently have data access security models in place that can be extended to ODM, such as models in an existing RDBMS that support the federation of external sources?
  - If yes, this environment is an opportunity to reuse existing security models within the enterprise and simplify the overall rollout of ODM by reusing already deployed clients.
  - If no, a security model that is specific to ODM is required to secure the endpoint.
- ▶ Is the organization averse to implementing a security model that is specific to ODM?
  - If yes, an existing security model within the enterprise can be used, as long as that security model lends itself to the access of an ODBC/JDBC

data source. An existing application or RDBMS federation is a prime example.

- If no, all ODM security methods are available for consideration.
- ▶ Are FADs currently in use or are there specific plans to use this feature?
  - If yes, ODM can be configured (but not mandated) to take advantage of these security definitions.
  - If no. If there is no requirement for fine-grained security access, ODM and FADs are not an appropriate security model to consider at this time.

### 11.4.3 Open data manager federation configuration

Optim ODM provides access to data through ODBC and JDBC standards-based interfaces. You can use any RDBMS system, which can federate data sources based on these standards, to access ODM. Any existing security access controls in place for the RDBMS lend themselves to accessing the federated ODM source.

Most commonly, Optim solutions have used IBM Federation Server's ODBC wrapper in conjunction with nicknames. In addition, companies have used Oracle Heterogeneous Services in conjunction with DBLinks. Consult the vendor-specific documentation for details about how to configure a specific RDBMS to federate an external data source.

### 11.4.4 Implementing open data manager workspace security

ODM workspace security is fairly straightforward to implement. It essentially requires that you define an access control list for a given workspace to determine who has access to that particular workspace. You can use Optim connect studio to define this access control list.

**User accounts:** You do not manage user accounts within the Optim Connect in which the ODM is implemented. Optim Connect uses the underlying user repository for the system and domain on which it resides for user account definition. When an access control list is built, the user names are those names that are reflected in the underlying operating system user repository.

Before discussing how to add users for workspace access, it is important that we describe what a workspace is. Also, we describe how a workspace relates to defined ODM data sources, which will build the foundation that is required to define access to a particular data source.

## Introduction to bindings and workspaces

There are two fundamental concepts, bindings and workspaces, that are required to understand how workspace security works:

- ▶ Binding

A *binding* within Optim Connect is a logical grouping of defined data sources.

- ▶ Workspace

A *workspace* within Optim Connect is a process that services client requests. A workspace is configured to service a binding. Therefore, any data sources that are defined in a binding are bound by the configuration of the workspace.

By default, Optim Connect is configured with a default binding called NAV and a default workspace called Navigator. By default, the Navigator workspace services the binding NAV. Let us examine the Navigator workspace definition using Optim connect studio. Here, we assumed that the Optim connect studio is configured to manage an Optim Connect server. Expand the server of your choice, and if the default binding and workspaces have not been removed, you will see a NAV binding and a NAVIGATOR workspace, as shown in Figure 11-6.

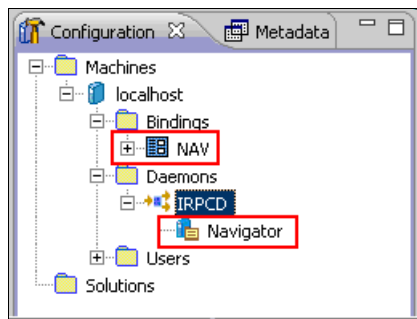


Figure 11-6 Default binding and workspace

Right-click the **Navigator** node, and select **Open**. The Navigator panel opens a dialog with the Navigator tab selected, displaying the Navigator workspace configuration. Note the setting for “Workspace binding name”. By default, it is empty. When this setting is empty, it defaults to the NAV binding at run time. See Figure 11-7 on page 443.

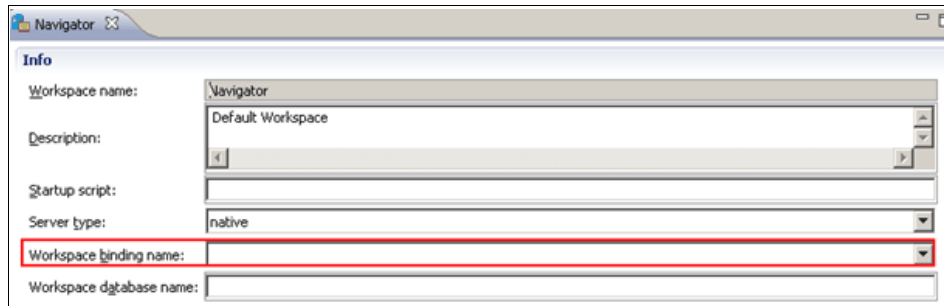


Figure 11-7 Navigator workspace

Selecting the drop-down box for the Workspace binding name setting displays all of the available bindings that are defined for this Optim Connect server. If no new bindings have been added to the server configuration, only the NAV binding is available. Selecting a specific binding name here invokes this workspace to service the data sources that are defined within that binding.

## Creating a new binding and workspace

It is best to use the same name for the binding name as the workspace name, which allows for immediate identification of which binding is serviced by which workspace.

Using this guideline, let us create a new binding and an associated workspace to service the data sources that are defined within the binding using the following steps:

1. Right-click the **Bindings** folder in the Optim connect studio tree and select **New Binding**.
2. For the binding name, type Application1 and select **Finish**. This action results in the child named Application1 being added to the Bindings node, as shown in Figure 11-8.

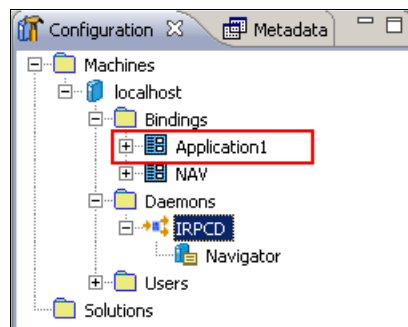


Figure 11-8 Adding a new binding

3. Next, right-click the **IRPCD** node under the Daemons folder within the tree and select **New Workspace**.
4. For the workspace name, enter `Application1`. Select all of the defaults (or configure based on your specific need) for the rest of the wizard. This action results in the child named `Application1` being added to the Daemons node, as shown in Figure 11-9.

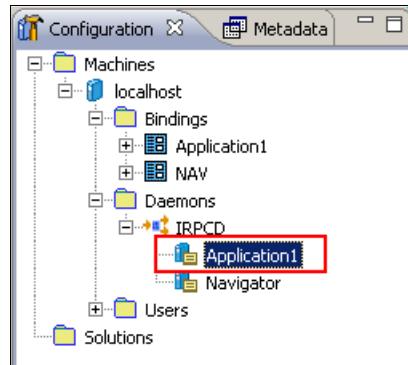


Figure 11-9 Adding workspace

5. Right-click the **Application1** workspace that you have added and select **Open**. The main panel opens a dialog with the General tab selected displaying the Application1 workspace configuration.
6. Select the “Workspace binding name” drop-down list, and select the workspace binding name **Application1**. Your configuration now looks like Figure 11-10.

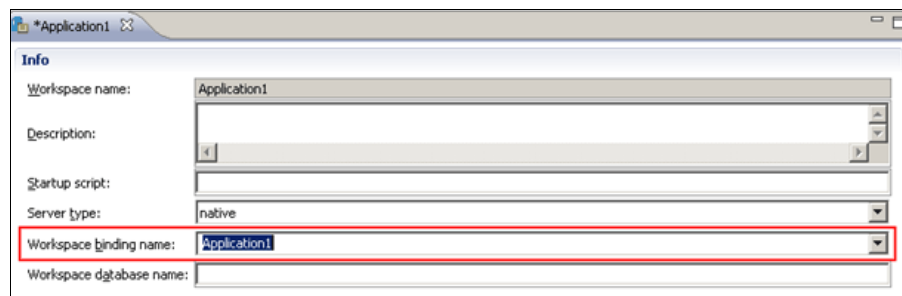


Figure 11-10 View workspace binding name

7. Click the **Server Mode** tab at the bottom of the main panel. Ensure that the “Workspace server mode” setting is set to **singleClient**, as shown in Figure 11-11 on page 445.

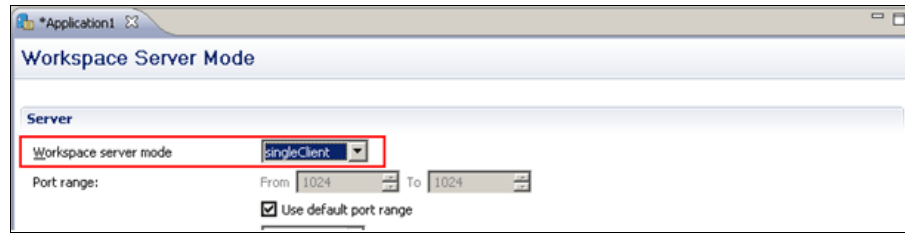


Figure 11-11 Workspace Server Mode

8. Click **Save** to save the workspace configuration.

You have now created a new binding and a workspace to service that binding.

## Setting the access control list for workspace access

In the previous section, we created a new workspace called Application1, and we configured that workspace to use the new binding that we created called Application1. In this section, we secure this workspace for access by authorized users only.

Perform these steps to secure the workspace:

1. In Optim connect studio, right-click the workspace **Application1** (Figure 11-12) and select **Open**.

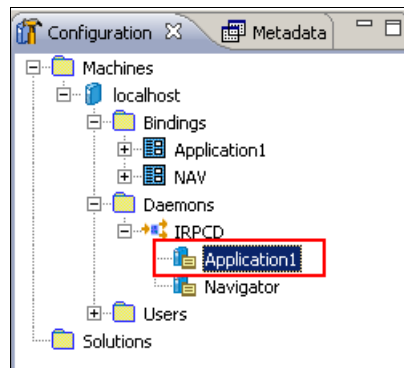


Figure 11-12 Open workspace configuration

2. The main panel opens with the workspace configuration. At the bottom of the main panel, select the **Security** tab. This action takes you to the Workspace Security configuration window (Figure 11-13 on page 446).

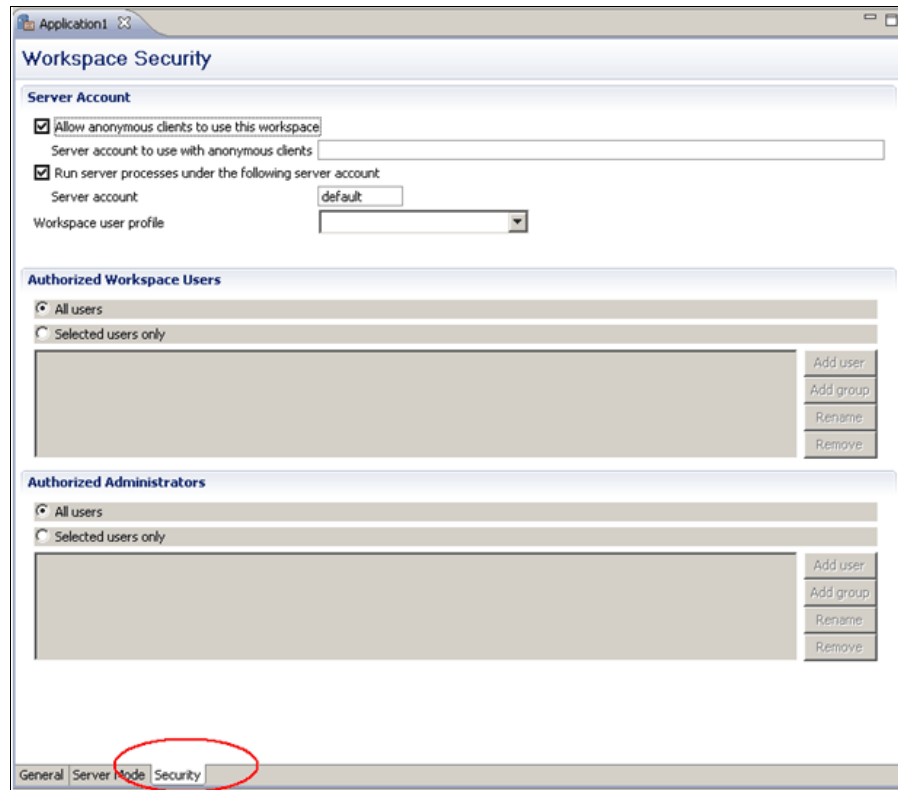


Figure 11-13 Workspace Security configuration window

3. In the Workspace Security configuration window, in the Server Account section, clear **Allow Anonymous clients to use this workspace**.
4. In the “Authorized Workspace Users” section, click **Selected users only**. You now have a workspace security configuration that looks like Figure 11-14 on page 447.

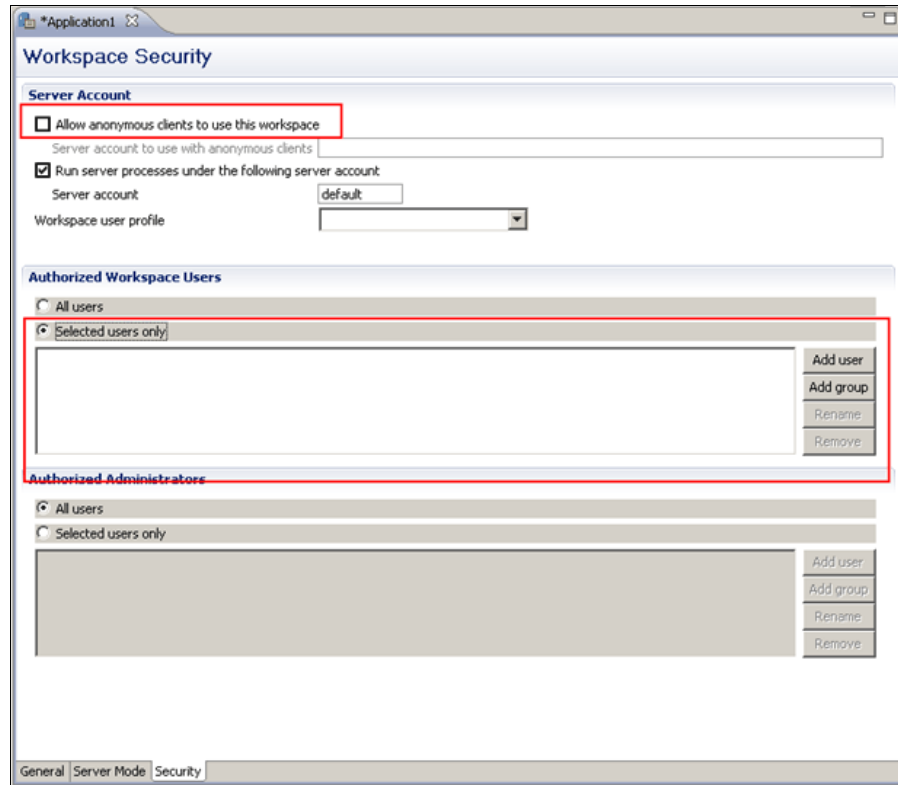


Figure 11-14 Workspace Security configuration

You now need to add users to this workspace. Note that the users that you add are the account names for the underlying user repository. For example, on a Microsoft Windows machine that is not part of a domain, these users are the local users.

5. On the right side of the window, click **Add user** for the Selected users only option. In this example, we type user1 to add the user named user1, which is configured as a local user. See Figure 11-15 on page 448.

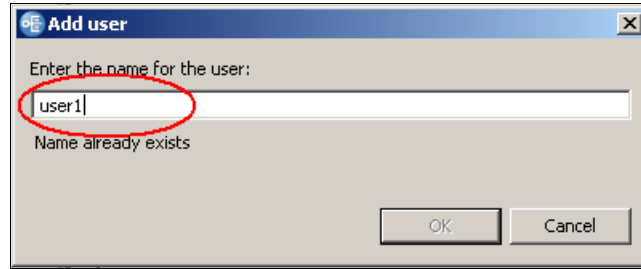


Figure 11-15 Adding a user

6. Click **OK** when done. Your Selected users only box now contains user1, as shown in Figure 11-16.

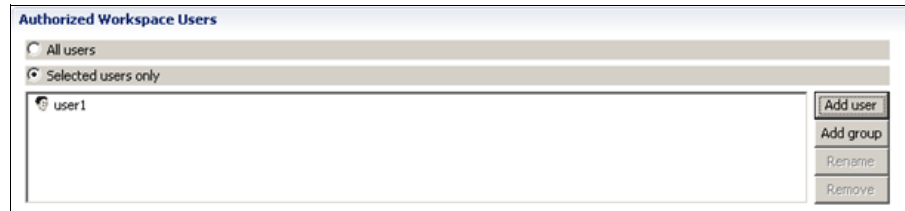


Figure 11-16 User added

7. You have now secured access to this workspace, and any of the data sources that are defined in the binding Application1. In this case, only user1 can access the workspace.

Figure 11-17 on page 449 shows the Microsoft Windows 2003 Local Users and Groups where user1 has been defined. In this example, user1 is listed as an Authorized workspace user, but the user account is defined in the local user repository.

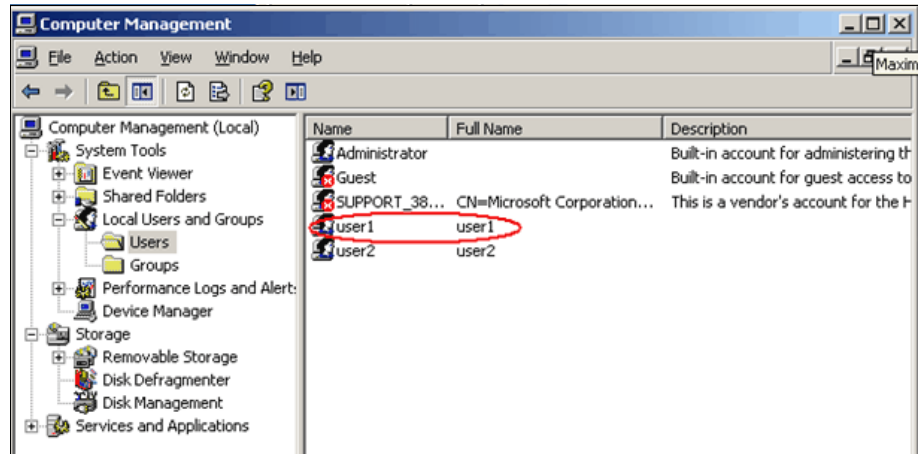


Figure 11-17 Local Users and Groups

### 11.4.5 Open data manager and file access definitions

To use file access definitions using ODM, operating system impersonation is used to represent client identity. As an example, when an ODBC connection parameter presents user credentials for John Doe, a process starts on the server that impersonates the user John Doe. In order to facilitate this impersonation, the main ODM process (Optim Connect server daemon process) first attempts to validate the user credentials that are provided for John Doe. If the process is unable to validate the user credentials (for any reason), the connection will be rejected.

ODM consumes the underlying operating system security model to validate users. This function requires that the ODM user credentials must be valid credentials from the operating system's perspective. The credentials are authenticated based on the following criteria:

- ▶ If the ODM server resides in a domain, the user credentials are validated against that domain.
- ▶ If a domain is not available, the user credentials are validated against the local user repository of the ODM server.

**Exception:** In Microsoft Windows environments, if a workgroup is defined on the machine where the ODM server is running, the workgroup name must be used in the FAD definition. The local user repository for which the ODM server is running is used to validate the user.

After the user is validated, a subprocess is started, which impersonates the user. As the impersonated user process attempts to access archive files and collections, the impersonated user credentials are used to determine the level of access that is allowed to these resources.

**Important:** It is important to note that these interactions require the configuration of both Optim and the underlying operating system. From an Optim perspective, the configuration involves allowing user impersonation. From an operating system perspective, the configuration involves user management and access control. These separate perspectives allow you to take advantage of the underlying security model that is deployed within an enterprise.

### **File access definitions and ODM configuration**

Three primary configurations are required to enable file access definitions through ODM:

- ▶ Operating system credentials
- ▶ Operating system file permissions
- ▶ Configuration of the ODM server to impersonate users

### ***Operating system and user credential requirements***

All users accessing an ODM data source, which has file access definitions secured, must have a valid account from the underlying operating system's perspective. The information, which is provided in "Open data manager and file access definitions" on page 439, indicates one of the following requirements for user credentials:

- ▶ Microsoft Windows:
  - If the ODM server runs within a Microsoft Windows domain, users are validated against that domain.
  - If the ODM server runs outside of a Microsoft Windows domain, users are validated against the local server's user repository.
- ▶ UNIX variants

Users are validated against the local user repository.

**User validation:** Calls to the underlying operating system to validate users are the same whether a domain is present or not. Optim prefixes user names with a domain or machine name. It is up to the underlying operating system to validate the user credentials properly.

If the operating system on which the ODM server runs has the ability to validate users against a central user repository, there is no need to create local user credentials. In cases where the underlying operating system cannot validate against a central user repository, you must create local user credentials for ODM users.

***Open data manager user credential privileges: Operating system file permissions***

After the ODM server has validated the user credentials, those credentials require the appropriate file access to perform the operation of accessing Optim archives and collections. Table 11-2 lists the base set of file access permissions required by all ODM users for data sources that are secured by file access definitions.

*Table 11-2 Required file access permissions*

Directory	Access required
<OPTIM_INSTALL_ROOT>/rt (Including all objects under this directory)	Read, Write, and Execute
<NAVRROOT>/ (Including all objects under this directory)	Read, Write, and Execute
<DATA_BASE_CLIENT_FOR_OPTIMDIR>/	Requires default client privileges

***Optim Connect server configuration for impersonation***

Configuration of the Optim Connect server to support impersonation is a simple and straightforward process. Follow these steps:

1. Open Optim connect studio and expand the machine tree to expose the workspaces that are associated with the data sources for which you want to configure impersonation.

2. Right-click the workspace that you want to configure and select **Open**. Then, you see the workspace configuration editor with the General tab (tab at the bottom of the window). The General configuration window, as shown in Figure 11-18, requires no changes.

The screenshot shows the 'Navigator' workspace configuration editor. The window has a title bar with the text 'Navigator'. Inside, there are several sections:

- Info**: Contains fields for 'Workspace name' (Navigator), 'Description' (Default Workspace), 'Startup script' (/opt/IBM/Optim/rt/navroot/bin/nav\_server), 'Server type' (native), 'Workspace binding name', and 'Workspace database name'.
- Timeout Parameters**: Contains three spinners: 'Client idle timeout' (0), 'Connect timeout' (60), and 'Call timeout' (0).
- Logging and Trace Options**: Contains several checkboxes: 'Specific log file format', 'No timeout', 'Call trace', 'RPC trace', 'Sockets', 'Extended RPC trace', 'System trace', and 'Timing'. All are currently unchecked.
- Query governing restrictions**: Contains two spinners: 'Max number of rows in a table that can be read' (0) and 'Max number of rows allowed in a table before scan is rejected' (0).

At the bottom, there is a tabbed interface with three tabs: 'General' (selected), 'Server Mode', and 'Security'.

Figure 11-18 Workspace general information

- Next, select the **Server Mode** tab in the workspace configuration editor. Under the Server section, ensure that the Workspace server mode is set to **singleClient**. In the Server Provisioning section, ensure that all values are set to **0**. Your configuration looks similar to Figure 11-19.

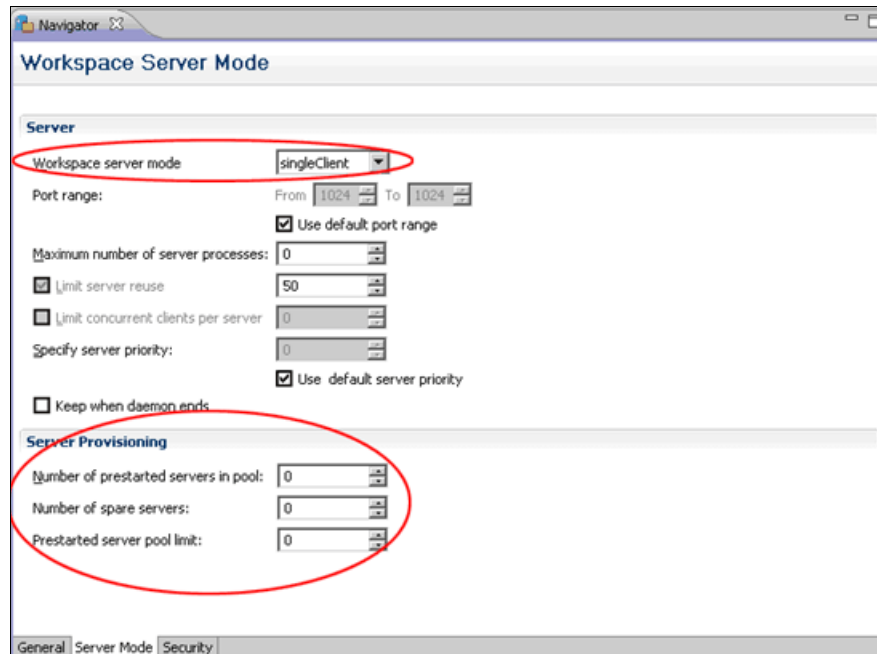


Figure 11-19 Setting the workspace server mode

4. Next, select the **Security** tab in the workspace configuration editor. In the Server Account section, perform the following steps:
  - a. Clear **Allow anonymous clients to use this workspace**.
  - b. Check **Run server processes under the following account**:
    - For the Server Account input box, ensure that it is empty. (*Do not* specify a server account.)
    - For the Workspace user profile drop-down box, ensure that it is empty. (*Do not* specify a workspace user profile.)

Your configuration needs to look similar to Figure 11-20.

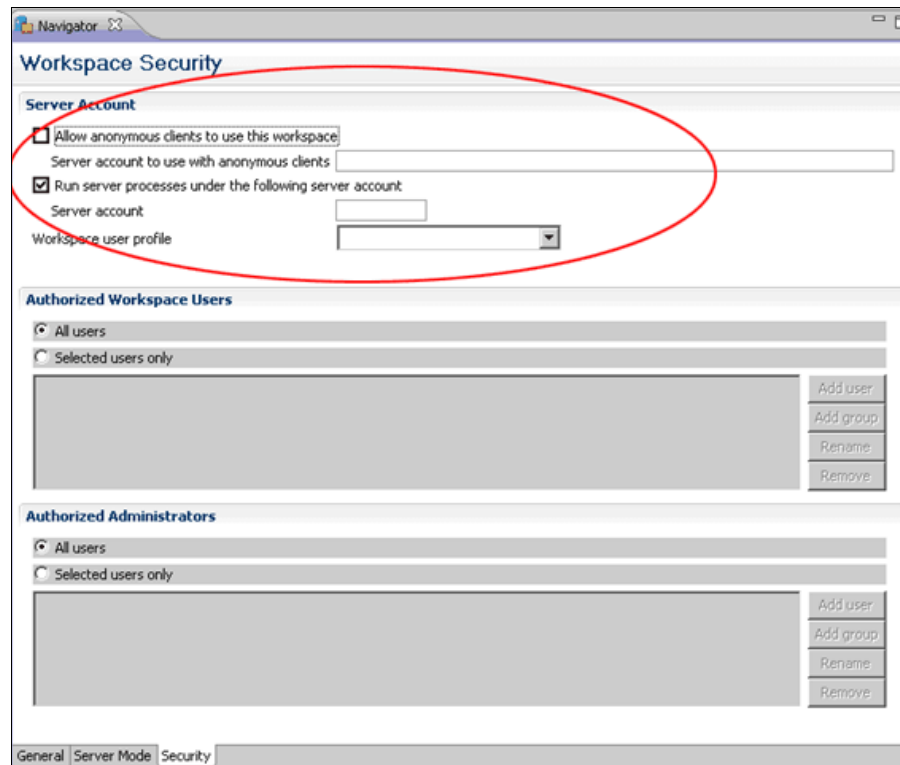


Figure 11-20 Setting server account information

5. Now, save the workspace configuration and reload the configuration on the server.

## Runtime requirements for impersonation

After you have completed the configuration of a workspace to use impersonation, it is important to run the Optim Connect server under a user ID that has authority

to validate user credentials against the user repository in use. If the main Optim Connect server daemon process ID does not have authority to validate users, all connections will be rejected.

In addition, all users that connect to the server process require the proper authority to read, write, and execute various objects within the file system. Ensure that *all* users who will connect to the server and be impersonated have the proper access control writes, as defined in the configuration section.

User credentials must be presented in the form of the simple user ID and password. User IDs must *not* include any domain qualification, because the user can only be impersonated within the domain for which the Optim Connect daemon is running. *Only a simple user ID can be presented.*

## 11.4.6 Open data manager and FAD example UNIX configuration

In this section, we show an example of the ODM and file access definitions configuration for use with UNIX variants. We assume that file access definitions and ODM are already installed and configured for normal user access.

### Users and groups

All users to be impersonated must have local system accounts on the ODM server. It is typical to use LDAP to manage local users on UNIX variants within an enterprise. Note if LDAP is deployed within your enterprise, because the creation of local users will be overwritten by LDAP if they are created locally.

After you have determined that the local user accounts are in place, create a group called “optim”. Add all local user accounts to this group, which will access ODM.

### File permissions

Locate the root installation directories for Optim on the distributed platforms and Optim Connect server:

- ▶ The default location for Optim on Linux, UNIX, or Microsoft Windows is /opt/IBM/Optim.
- ▶ The default location for Optim Connect server is /opt/IBM/Optim/rt/navroot.

### *Optim permissions*

Set the following permissions for Optim:

- ▶ Change the group recursively for the directory <OPTIM\_INSTALL\_ROOT>/rt to optim, which you previously created, for example:

```
chgrp -R optim rt
```

- Change the group permissions recursively for the directory `<OPTIM_INSTALL_ROOT>/rt` to READ, WRITE, and EXECUTE, for example:  

```
chmod -R 755 rt
```

### ***Optim Connect permissions***

If Optim Connect is installed to the default location that we have specified, the previous step has already provided the group with the required permissions. If it is not installed in the default location, perform the following steps:

1. Change the group recursively for the directory `<OPTIM_CONNECT_INSTALL_ROOT>/` to `optim`.
2. Change the group permissions recursively for the directory `<OPTIM_CONNECT_INSTALL_ROOT>/` to READ, WRITE, and EXECUTE.

### ***Optim directory database client***

Locate the database client and change the group to `optim`. The default privileges might or might not be sufficient, depending on your client.

## **Runtime connections**

Start the Optim Connect server and verify the security settings.

### ***Start the Optim Connect server***

Start the Optim Connect server with authority to validate users against the local user repository (for example, `root`) using the following command:

```
irpcd start
```

### ***Connect with an ODBC and JDBC client***

Using an ODBC client, connect to the ODM data source and provide valid user credentials, for example:

```
Username: user1  
Password: *****
```

### ***Access control domains and file access definitions***

Depending on the UNIX variant, the domain name that is listed in the access control domain users list has to specify either the host's domain name or the host name as the domain name, for example, `localdomain` or `localhost`.

## **11.5 Software**

In this section, we describe several practices for securing Optim software and the files that are generated by the Optim processes and activities.

## 11.5.1 Program executables and libraries

There are multiple components in the Optim software. Like any other software package, it is important to keep certain files secure from tampering. Optim was designed to function in the user space of the operating system. Therefore, it is not necessary for it to run as a super user. There is a client server stack. Hence, the server can run in a secure location and its only point of access is the definable port on which it listens. Authorized users are allowed to access and run processes on the server. A client/server configuration is the ideal method of securing the Optim software and access to its functions.

In this configuration, if the database uses the operating system authentication, Optim must run as a user that has access to the databases. Also, any file shares on which you plan to store the Optim output files must be accessible by the user under which Optim runs.

You also can configure Optim to require each client to authenticate using the credentials of a user that is local or available to the resident operating system. The advantages of this security approach include a more defined audit trail and individual security for files that are created and used by Optim. However, this method requires Optim to run as a super user so that it can create processes as a separate user. For more detailed information, see the appendix in the *Optim Installation and Configuration Guide*.

## 11.5.2 Archive, extract, and control files

Executing Optim processes generates several types of files. The archive, archive index, and control files are flat files that contain the data that is selected from one or more data sources. In this section, we refer to these files as archive files. These files are in a machine-independent format that is only interpreted by Optim functions and services. You can compress archive files by using native Optim functions, or you can encrypt them while they are at rest using your choice of encryption software.

The archive files can be secured from other Optim users by using file access definitions. File access definitions prevent anyone from copying an archive file to another machine where that user is a super user and accessing it with another copy of Optim.

The extract files cannot be secured in this way. The extract files are created during selective restores and must be removed after the restoration is completed. Also, any data privacy or data loading activities must be removed after the restoration is completed. Temporary files are created during these processes and they must be removed after they are used.

The archive files are created in a default data directory that is defined during product installation and configuration. This default location can be overridden in each archive process.

The Optim files exist as flat files in the directories and folders of the host system. You also must secure these directories and folders from unauthorized users at the operating system, file system level. The Optim users must be able to read and write to the directories and folders that contain these files. Using file naming conventions and macros (predefined system variables, which are described in the *IBM Optim Common Elements Manual*), you can set up separate directories for various users or user groups. Therefore, you can limit the access to these files to only authorized users.

### 11.5.3 Logging, trace, and other temporary files

Optim creates temporary files as part of its normal activities. The most common temporary files are trace files. Optim uses these files as diagnostic tools when there is an error. Also, you can use trace files as an audit trail of activity.

Trace files are created in a temporary directory that is defined during installation and configuration. Its location can be changed at any time.

These files are stored in the file system of the operating system. It is the responsibility of the operating system security administrator to keep them safe. The permissions need to be set at the directory level to prevent unauthorized access. However, the Optim users must have permission to create and update them. Typically, after trace files are created and reviewed if necessary, they are no longer needed. Trace files are in clear text, but no sensitive information exists in these files.

### 11.5.4 Database objects

Optim requires a database to store its repository. This repository retains information about Optim's configurations and all the objects that are created during design time. This repository also stores the procedures that are used during program execution. Optim relies on the operating system and DBMS settings to keep these objects and procedures secure.

Optim also retains information about user accounts for defined DBAliases. A DBAlias is an Optim definition that describes a connection to a database and contains user credentials.

Optim stores certain procedures in the user or owner space of the DBAlias. These DBAliases must be kept safe by the underlying operating system and database system.

When using Oracle as a repository or a data source, it is important to know that the owner of the DBAliases and repository is required to grant certain permissions that are not allowed by certain clients' security policies. The most notable grant is SELECT ANY DICTIONARY. Optim will not function properly without this grant. However, after you fully understand the security policy, you can use work-arounds. For example, a possible work-around is to create a database user specifically for and used by Optim. The Optim DBAlias can be created for that user. Optim users that use this DBAlias can then supply their own database user credentials for their own database user to perform Optim activities.

## 11.6 Audit trail

Auditing activities is a common practice for any organization implementing a security policy. This section describes several of the auditing capabilities within Optim.

### 11.6.1 Optim

Optim comes standard with several simple auditing capabilities.

Optim Process Audit is a facility that enables a site to monitor Optim processes and obtain information about their execution. When a process is started, Optim collects information, such as the type of request and the user who initiated it. It then writes audit records in XML format. Audit records are produced whether an Optim process is run from a menu, editor, command-line interface (CLI), or a scheduled job. When the Process Audit is enabled for an Optim directory, auditing is active for all users of the following processes: *archive*, *browse*, *compare*, *convert*, *create*, *delete*, *edit*, *extract*, *export*, *import*, *insert*, *load*, *ODM*, *report*, and *restore*. Auditing can be activated or deactivated at any time.

The XML that is produced by Process Audit is written to the Optim directory audit tables. An audit record is written at the completion of an Archive, Browse, Compare, Convert, Create, Delete, Edit, Extract, Export, Import, Insert, Load, ODM, Report, and Restore process. Refer to the *Optim Installation and Configuration Guide* for more information about Optim Process Audit.

The trace files that Optim produces can also provide audit information. However, this information is intended for diagnostics by IBM Support and is not easy to understand.

Optim is configured to use a default security exit. This exit is called whenever a function in Optim is activated. By default, it validates the current operating system user against any ACLs to validate that they have the appropriate privileges to access the requested functions or objects.

You can use a customized exit with Optim instead of the default exit. You can use customized exits to provide a more thorough audit of activities, if necessary. You can read the details that are associated with customizing an exit in the *Optim Initialization Exit Programmer's Guide*.

## 11.7 z/OS

You administer security differently on z/OS, and the functionality varies. However, the same level of security that is available on the distributed versions of Optim is available on z/OS.

For more details about Optim security on z/OS, refer to the *IBM InfoSphere Optim for DB2 for z/OS Customization Guide*, which is delivered with the product.

## 11.8 Implementation practices for Linux, UNIX, and Microsoft Windows environments

In this section, we discuss several of the common uses of Optim security with a focus on the Linux, UNIX, and Microsoft Windows environments.

### 11.8.1 Initializing Optim security

Implementing Optim security is fairly straightforward. You use the Optim Configuration tool to start a wizard that guides you through the process. The wizard first asks if you want to initialize. If you do, it then asks you to enable function, object, and file access security. You can enable any or all of these types of security at any time after security has been initialized.

The *IBM Optim Installation and Configuration Guide* describes the process for performing these tasks. However, it is important to remember that Optim security, by default, denies access. Therefore, when initializing for the first time, you must not enable function, object, or file access security until after you have configured an access control domain.

At a minimum, the access control domain must have a system administrator role with an assigned user ID that has privileges to administer Optim's security features and objects.

After the access control domain is defined with a role that has access to Optim's security features and objects, then go back to the Optim Configuration tool and enable the security subsystems.

## 11.8.2 Securing your Optim implementation

Before enabling Optim security, you need an implementation plan. Use the information in the following sections as a guide to aid you in this planning.

### 11.8.3 Activity classes

Plan the Optim security implementation around five classes of activities:

- ▶ Administration
- ▶ Configuration
- ▶ Operation
- ▶ Auditing
- ▶ Viewing

Each class can map to one or more roles. In certain rare cases, roles can span classes. However, to maintain the proper segregation of duties, it is best to avoid creating roles that perform activities in multiple classes.

Segregate the duties by business function. For example, a customer services representative needs to be able to view the contents of the allowed archived data but not have access to any Optim administrative or configuration functions.

#### **Administration**

This class includes the day-to-day management activities. They include functions, such as software maintenance, security management, software upgrades, and so on. Users that are designated as administrators must not access privileges in any other classes.

#### **Configuration**

This class encompasses the activities for those users who design archiving strategies. These users are responsible for designing and building Optim objects to satisfy business, audit, and regulatory requirements.

These users must perform this type of work in non-production environments. Where possible, all changes must be promoted to production using

industry-accepted change control protocols. These users must not make changes directly to production environments.

### **Operation**

This class consists of the activities that are performed by the users that are Optim operators. These users are responsible for the execution of Optim processes, such as running archives and extractions, privatizing and accessing the data.

In a non-production environment, these users are typically the Quality Assurance (QA) team. In production environments, these users are the actual users of the Optim system. The operational class even includes users that are automated processes.

### **Auditing**

This class includes activities for monitoring and viewing the activities occurring within the Optim solution. This class also includes activities for maintaining the audit logs. Only the users of the auditing class need to be able to clean up and change audit logs. Even administrators must not be able to manage audit logs, because they too are being audited. You do not want them to be able to hide their activities in the system.

### **Viewing**

This class includes those activities that access the data in the archive files. Viewers are typically the consumers of the data that is contained in the databases. After the data is archived, the viewers must use the archive files to access the data.

## **11.8.4 Roles**

Roles are subsets of any of the previously mentioned classes. You implement a class in Optim by using roles. You use roles to define a group of people with like access to Optim. Roles are the mechanism that Optim uses to implement the separation of duties.

You separate duties to prevent any single person from having full authority. This separation of duties creates an environment where multiple people are required to make a system work as a whole and, at the same time, prevents any one person from being able to take down the whole environment.

If we seem redundant in discussing the separation of duties, it is because we think it is the most significant part of implementing a solid security structure.

Giving all access to a single person makes it possible for that one person to act maliciously, either by accident or on purpose.

We suggest that, at a minimum, you create the following roles and responsibilities:

- ▶ **System manager:** Responsible for the Optim software components and the enabling and disabling of security.
- ▶ **Security manager:** Responsible for maintaining roles, grants, and the access control domain.
- ▶ **Optim manager:** Responsible for managing the Optim directory and DBAliases.
- ▶ **Designer:** Has the ability to create objects and definitions within Optim.
- ▶ **Operator:** Has the ability to execute actions.
- ▶ **Auditor:** Has the ability to review and manage security reports.
- ▶ **User:** Has the ability to review data after it has been archived. These users do not need any access to functional or object security, but they need access in the file access definitions.

Table 11-3 breaks down the roles and suggests to which Optim functions the roles need access. Appendix B, “Functions and actions access permissions by roles” on page 515 lists the actions in each function and the suggested access that a role might have.

*Table 11-3 Roles and functions access*

<b>Optim functions</b>	<b>System manager</b>	<b>Security manager</b>	<b>Optim manager</b>	<b>Designer</b>	<b>Operator</b>	<b>Auditor</b>
Create new actions	N/A	N/A	N/A	Allow	N/A	N/A
Create new definitions	N/A	N/A	N/A	Allow	N/A	N/A
Create security definitions	Deny	Allow	Deny	Deny	Deny	Deny
Create utility definitions	Allow	N/A	N/A	N/A	N/A	N/A
Editor options	N/A	N/A	Allow	N/A	N/A	N/A
File maintenance	N/A	N/A	Allow	N/A	N/A	N/A
Invoke action editors	N/A	N/A	N/A	Allow	N/A	N/A
Invoke command-line actions	Allow <sup>a</sup>	N/A	Allow <sup>a</sup>	Allow <sup>a</sup>	Allow <sup>a</sup>	N/A
Invoke configuration actions	Allow <sup>a</sup>	N/A	Allow <sup>a</sup>	N/A	N/A	N/A

Optim functions	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Invoke definition editors	N/A	N/A	N/A	Allow	N/A	N/A
Invoke options	Deny <sup>b</sup>	Allow <sup>a</sup>	Deny <sup>b</sup>	Deny <sup>a</sup>	N/A	N/A
Invoke utilities	Allow <sup>a</sup>	N/A	Allow <sup>a</sup>	Allow <sup>a</sup>	Allow <sup>a</sup>	N/A
Run untitled actions	N/A	N/A	N/A	N/A	N/A	Allow <sup>a</sup>
Security tasks	Allow <sup>a</sup>	Allow	Allow <sup>a</sup>	N/A	N/A	Allow <sup>a</sup>

a. Only on certain actions. See Appendix B, “Functions and actions access permissions by roles” on page 515 for details.

b. Production options are allowed. See Appendix B, “Functions and actions access permissions by roles” on page 515 for details.

Table 11-4 and Table 11-5 on page 465 show suggested access permissions for each role for Optim objects and the access control list of the objects. We use the following notation used in the table:

- R: Read
- U: Update
- D: Delete
- E: Execute

*Table 11-4 Object security for each role*

Object access	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Access definition	R	N/A	R	ERUD	R	N/A
Archive request	N/A	N/A	R	ERUD	R	N/A
Compare request	N/A	N/A	R	ERUD	R	N/A
Convert request	N/A	N/A	R	ERUD	R	N/A
Delete request	N/A	N/A	R	ERUD	R	N/A
Extract request	N/A	N/A	R	ERUD	R	N/A
Insert request	N/A	N/A	R	ERUD	R	N/A
Load request	N/A	N/A	R	ERUD	R	N/A
Report request	N/A	N/A	R	ERUD	R	ERUD
Restore request	N/A	N/A	R	ERUD	R	N/A

Table 11-5 Object ACL access for each role

Object ACL access	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Access definition	N/A	N/A	N/A	R	R	N/A
Archive request	N/A	N/A	N/A	R	R	N/A
Compare request	N/A	N/A	N/A	R	R	N/A
Convert request	N/A	N/A	N/A	R	R	N/A
Delete request	N/A	N/A	N/A	R	R	N/A
Extract request	N/A	N/A	N/A	R	R	N/A
Insert request	N/A	N/A	N/A	R	R	N/A
Load request	N/A	N/A	N/A	R	R	N/A
Report request	N/A	N/A	N/A	R	R	R
Restore request	N/A	N/A	N/A	R	R	N/A

## 11.8.5 Configuration management

Managing the growth and change of your Optim implementation is not only for change control, but for security as well. Having more control over the changes to objects and changes to your production system helps to keep them safe from bad or malicious code.

Optim is flexible enough to allow you to use any change control software to manage objects that have been created and changed by the designers. The Optim import and export facility allows these objects to be copied from Optim to a source code repository and back again.

During the design, development, and maintenance phases, an Optim object goes through several statuses. Identify the appropriate states and use them as points for control and propagation. The following list suggests certain states:

- ▶ New: A newly created object.
- ▶ Working: An object on which someone is working. It has been returned from testing or several days might pass before anything new is added to it. This stage allows the object to be stored in the source code repository until it can be finished.
- ▶ Ready for test: The object is moved to the source code repository and marked as ready for test. All ready for test objects are imported into the test

environments as testing commences. Objects that fail the tests go back to a working state.

- **Production ready:** The object passed the tests and is ready for production. All objects that are marked as Production Ready are imported into the production system.

## The process

A typical process using these states might look like this example.

A developer is assigned to create a procedure to archive customer orders using specific criteria. The developer starts by accessing Optim and creating an access definition and archive request (at a minimum). At the end of the day, the developer is not finished, so the developer exports the object and adds it to the source code repository, changing its status to “Working”.

The next day, the developer continues the work on the object. When the developer finishes and tests the object satisfactorily, the developer exports and updates the object in the repository. The developer now marks the object as “Ready for test”.

At the beginning of the following week, the testers are ready to test. They pull out all objects that are marked as “Ready for Test”, import these objects to the test system, and start testing.

For those objects that fail, the object is marked as “Working” again and the designer is given a message that it needs to be fixed. For those objects that pass the tests, they are marked “Production ready”.

When it is time to start the user acceptance test (UAT) system, all the “Production Ready” objects, which typically is all objects at this point, are imported into the UAT system. The users perform acceptance testing. If the acceptance testing succeeds, the objects are imported into production and go live.

Of course, this example is an extremely simplified version of the process, but it is enough to demonstrate how to use Optim within the confines of a configuration system to help manage control and security.

Even without a source code repository, you can use Optim to manage the change of objects but you need to institute a naming convention on the object to allow for versioning control.

How you choose to implement configuration management for Optim is up to you, but implementing configuration control is a must for a strong, secure InfoSphere Optim Data Growth Solution.



# Troubleshooting

Up to this point, we have looked at the planning, design, and implementation work that you must undertake to realize your IBM InfoSphere Optim Data Growth Solution. In this chapter, we give you insight into problem and issue resolution.

We start by looking at how you can get the available support and diagnostic tools. Then, we move on to several of the problem areas you might have to explore. We work with these areas in the order that they might appear:

- ▶ Validating your installation and solution
- ▶ Dealing with issues getting data out of your sources
- ▶ Deleting data from the sources after a successful archive
- ▶ Addressing issues accessing your data from the archive
- ▶ Implementing disaster recovery

Where troubleshooting tips vary between platforms, we have endeavoured to supply you with the options. However, due to the broad nature of the platforms and sources that Optim supports, we were not able to provide all the options within the confines of this book. The principles for each issue are transferable and the product manuals will supply the pertinent details.

# 12.1 IBM Support

To obtain IBM Support for Optim Data Growth solutions, contact your software support team. If you are unsure how to make contact with your software support team, use the contact information that is published on the internet:

<http://www14.software.ibm.com/webapp/set2/sas/f/handbook/home.html>

Initially, you have to supply your customer number, summary of the problem, and the severity of the problem for your business. When you are put in touch with the correct product support team, think about your problem in the terms (not all terms will apply) that are shown in Table 12-1. This information can facilitate IBM Support in timely problem solving.

Table 12-1 Problem-related questions

Relevance	Question
General	What release or build of the product are you running?
	Do you have any maintenance applied?
	What operating systems is Optim deployed on and on which versions?
	What database management system (DBMS) is being accessed and what version of the database is being used?
	What is the complete text of the error message?
	Can you reproduce this problem?
	Was the process running online or in batch?
	Is this the first time that this problem has occurred?
	Has anything changed in your environment prior to getting this error (database upgrade, product upgrade, and so on)?
	What operation was being performed at the time of the error?

Relevance	Question
Archive	Were you using the Optim Demo tables when the error occurred?
	Can you re-create the error with the Optim Demo tables?
	Are Selection or SQL criteria being used?
	Are any row limits being used?
	How many tables are in the access definition?
	Are any tables marked as reference tables?
	Are any tables marked for delete after archive (DAA)?
	Are there any archive actions being used?
	Is there date criteria (DC) specified?
	Is the archive file being written to disk or to tape (mainframe only)?
Restore	Is this a full restore or a selective restore?
	If running batch, is a single archive file specified or are multiple archive files specified for searching?

## 12.2 Troubleshooting aids

In this section, we look at the troubleshooting aids that are available in Optim to help you answer several of the preceding questions and diagnose your problems. You can use several of these aids in conjunction with guidance from the support teams. Other aids are useful to you in ensuring consistent behavior from your Optim solution with regard to the goals of your archiving project, service-level agreements (SLAs), and performance targets.

### 12.2.1 Environmental information

It is important to have accurate information about your solution and installation. This information supports your change management and problem management processes. It helps establish if change is responsible for any issues and it helps the support teams validate if your issue has been seen and resolved previously. In this section, we detail how to establish your environmental information.

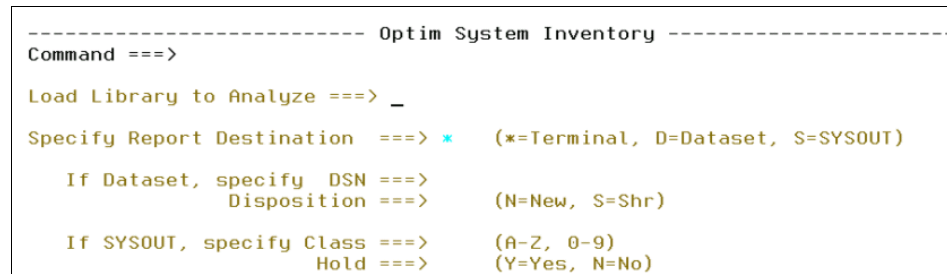
## z/OS

As of Optim for z/OS V7.1, Optim has been SMP/E-enabled. You can obtain information about the product's function modification identifiers (FMIDs) and maintenance that has been applied through SMP/E as usual. If you do not have responsibility for the SMP/E environment, your systems programmers will be able to help you supply this information with advice from the support team.

On Optim for z/OS V6.1.1 and earlier, you can use the **PSTINVN** and **PSGETINF** utilities to obtain your environmental information.

To determine what product fixes and accumulated maintenance have been applied, use the **PSTINVN** utility.

While in the product, on the command line, enter TSO PSTINVN. The Optim System Inventory panel is shown (Figure 12-1). Enter the name of the load library to analyze and specify the report destination.



```
----- Optim System Inventory -----
Command ===>

Load Library to Analyze ===> _

Specify Report Destination ===> *    (*=Terminal, D=Dataset, S=SYSOUT)

  If Dataset, specify DSN ===>
    Disposition ===>          (N=New, S=Shr)

  If SYSOUT, specify Class ===>      (A-Z, 0-9)
    Hold ===>                    (Y=Yes, N=No)
```

Figure 12-1 Input load library

Or, you can run the following JCL:

```
//PSTINVN JOB ( ),CLASS=A,MSGCLASS=H
// EXEC PGM=PIPIMAIN
//STEPLIB DD DSN=PSTRT.SUP54.ISPLLIB,DISP=SHR
//PSIFRPT DD SYSOUT=*
//PSIFLIB DD DSN=PSTRT.SUP54.ISPLLIB,DISP=SHR
```

Change *PSTRT.SUP54.ISPLLIB* to your Optim ISPLLIB.

To determine the z/OS environment, use **PSGETINF**. Figure 12-2 on page 471 shows the output.

```

07/01/16 - 15:28:53  PRINCETON SOFTECH - SITE INFORMATION REPORT

Hardware Configuration:
-----
Number of CPUs : 2
CPU 00       : Serial(0000F2) Version(0A) Model(1247) PCCA Address(00F51008)
CPU 01       : Serial(1000F2) Version(0A) Model(1247) PCCA Address(00F4e238)
Total MIPS   : 105.07
MSU          : 18
Service Units : 2548  service units per second

Software Configuration:
-----
Operating System : z/OS          Version: 01.07.00  SP Level: SP7.0.7 H887720
Job Entry Subsys : JES2         Type: JES2 Level: z/OS 1.7
System Name      : ZDEV
Sysplex Name     : ZDEVPLEX
DFSMS/HVS        : 01.07.00
VTAM Level       : 6.1.7
TSO/E Level      : 3.06.0
ISPF Level       : 5.7  PDF 5.7
Security System  : RACF          Level: 7.72.0
DB2 Subsystem   : DBSC          Feature Code: 5740XVR01 Status: Active
DB2 Subsystem   : DBSF          Feature Code: 5740XVR01 Status: Active
DB2 Subsystem   : DSE           Feature Code: 5740XVR01 Status: Inactive
DB2 Subsystem   : DS7           Feature Code: 5740XVR01 Status: Active
DB2 Subsystem   : DSNB          Feature Code: 5740XVR01 Status: Active
DB2 Subsystem   : DSNL          Feature Code: 5740XVR01 Status: Active
DB2 Subsystem   : DSNV          Feature Code: 5740XVR01 Status: Active
CICS Level      : 6.4.0
Virtual Storage :
  User SPLS(Below) 6064K 1768K 4296K 2008K
  Auth SPLS(Below) 2104K 464K 1640K 964K
  User SPLS(Above) 32768K 14000K 17868K 18248K
  Auth SPLS(Above) 1875968K 12400K 1863568K 8168K

```

Figure 12-2 PSGETINF utility output

## Linux, UNIX, and Microsoft Windows

From the Optim workstation, you can learn about your Optim installation by clicking the Help option from the main panel, as shown in Figure 12-3.

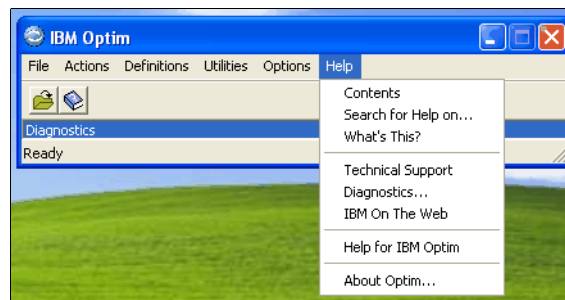


Figure 12-3 About Optim installation

Look at both About Optim and Diagnostics. Figure 12-4 on page 472 shows the Optim information in About Optim.

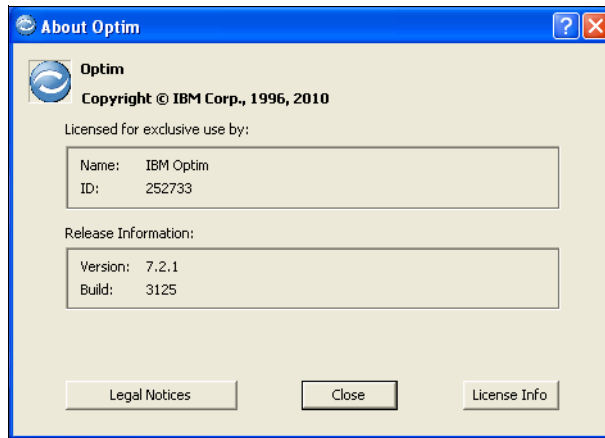


Figure 12-4 About Optim output

The Diagnostics window provides many elements of information, including PROTOOL, which shows build information. See Figure 12-5.

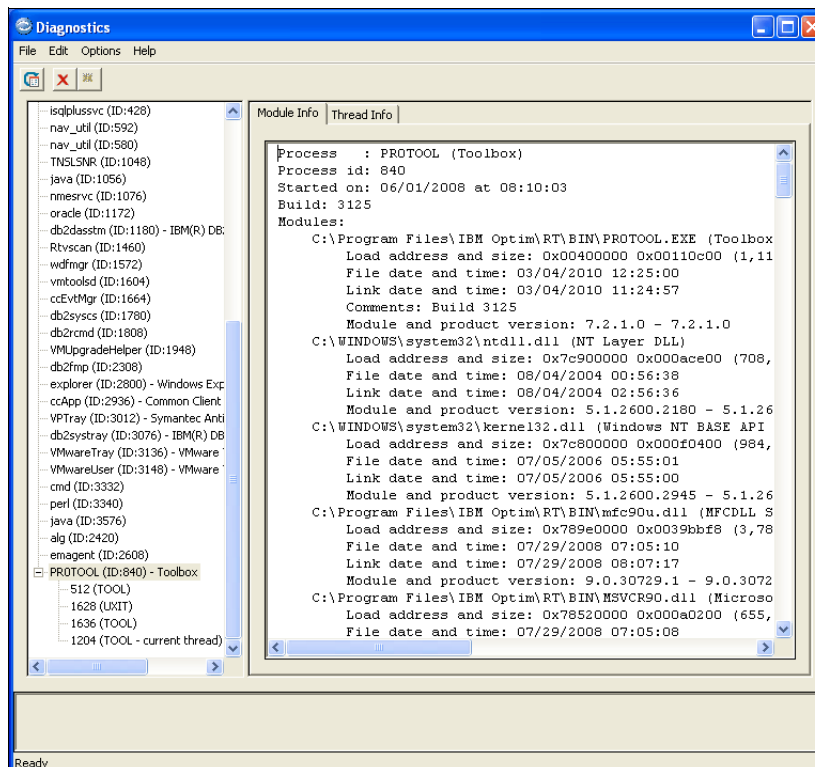


Figure 12-5 Diagnostics information

On Linux and UNIX, you can obtain the environmental information in the `PROTOCOL.xxx` file that is held in the temporary directory setup in your personal options.

In addition, the setup information is held in the configuration files and shell scripts, which are identified in the *IBM Optim Install and Configuration Guide*, which is supplied with your product code.

During your server setup, you typically deal with these configuration files and scripts in the following order:

- ▶ **RTSETENV**

This command invokes the shell script of the same name, which in turn defines the operating environment for the Optim server. RTSETENV is installed in the PSTHOME directory and designed to be included in a user `.profile` or `.login` script to set up the environment for the Optim server or command line on login.

- ▶ **RTSERVER**

This command invokes the shell script of the same name. The arguments that are supplied with the command define the operation to be performed. The script is installed in `/sbin`, which is subordinate to the PSTHOME directory.

- ▶ **pstserv.cfg**

The pstserv configuration file `pstserv.cfg` holds the configuration parameters for running `pr0svce`, the server daemon.

- ▶ **pstlocal.cfg**

This file is optional, but it is almost always used. It is the configuration file that you have to edit if you want to use the command line that is either local to the server or on the workstation to be passed to the server. The `pstlocal.cfg` file includes a more comprehensive set of configuration parameters than the `pstserv.cfg` file, because elements from the client are included.

- ▶ **RT4S**

This script is a less commonly used shell script whose purpose is to have the Optim server started as part of the server booting up.

## **Optim connect studio**

You can find the information about the installation level of Optim connect studio through the Help option on the main panel. Click **About Optim Connect Studio** to obtain the information (Figure 12-6 on page 474).

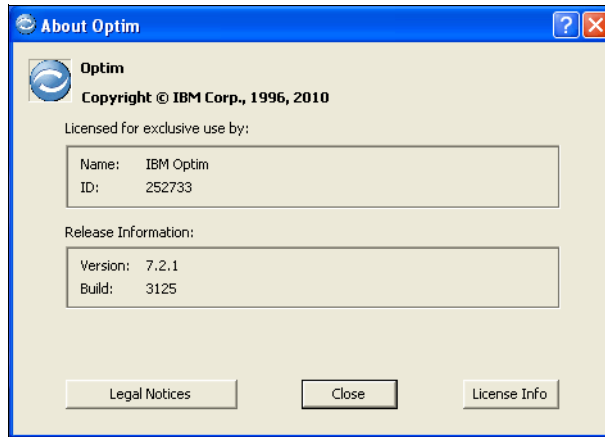


Figure 12-6 Installation level of Optim connect studio

## 12.2.2 Optim processes

Optim provides several capabilities, which have been discussed throughout this book, to help you with your archiving project:

- ▶ Reporting output from your archive processes:
  - Archive reports give you information, such as run times, source tables extracted from, and row counts.
  - Statistics reports, which are generated by clicking the Generate Statistical Report box in the archive request, provide detailed performance information.
- ▶ Show Steps
 

Use this feature to view the exact steps that the archive process follows and to understand which tables are traversed and in what order. It can also help you identify cyclical relationships and if tables are traversed more than one time.
- ▶ Relationship Index Analysis
 

You can perform the equivalent of a database EXPLAIN within Optim to assess if additional indexing is required.
- ▶ Optim Tool Help
 

In addition to the product manuals, extensive help is available within the Optim tooling on Microsoft Windows and z/OS. The product help can help you develop your archive solution and resolve syntax issues.

Show Steps and Explain are particularly useful with respect to performance problem resolution. You can read detailed information about using these functions in Chapter 10, “Optim performance considerations” on page 405.

### 12.2.3 Tracing

Every Optim executable creates a trace file. The trace file is designed to tell support, quality assurance, and development about the programs, build, system information, zaps, trace settings, and the progress for the execution of a program.

This information is current as of Optim Release 7.3.1 Build 3620 for Linux, UNIX, and Microsoft Windows when new traces were added. However, the majority of the trace entries are available on the z/OS platform and in previous versions of Optim for Linux, UNIX, and Microsoft Windows.

#### **When to use tracing**

Tracing is always on to provide diagnostic information. However, you can switch additional detail on at the request of IBM Support, which is what is meant when we refer to turning tracing “on”. Only use this additional level of tracing for problem diagnosis and do not leave this level of tracing on during standard operations.

Table 12-2 on page 476 lists the trace variables.

Table 12-2 Trace variables

Service	Category	Description
Relational database services	RDB	Relational database RDBMS product access Common services Code page translation
	DSQ	Dynamic SQL RDBMS connections Statement management Cursor management RDBMS specific interfaces DB2 handles DB2 MVS
	DSS	Directory service Partition specification table (PST) directory PST definition management RDBMS specific interfaces No Open Database Connectivity (ODBC) or DB2 MVS
	CSS	Catalog services RDBMS catalog access Look-aside cache RDBMS specific interfaces
	COL	Column services Column maps Data transformation Temporary database
General services interface	ENV	Environment services Platform management Operating system interfaces
	TRC	Trace services Trace record generation Trace file management
	ERR	Error services Formatted error recording Exception manager
	MEM	Memory services Storage allocation Heaps, cells, hash, and so on String sink

Service	Category	Description
Archive services	APS	Application plan services
	SEL	Select query compiler
	IXM	Index manager
	RSM	Removable storage manager
	SEC	Security
	DSX	Directory services extensions
	DDL	Data definition generation
Extract services	DML	Data manipulation statements
	DPP	Data process insert-update-delete
	DXX	Data extract
	XFM	Extract file manager
	XFB	Extract File Builder
	DAL	Dynamic allocation

### How to enable tracing

We have identified the available traces and now we look at how to enable tracing on the various platforms. If you have contacted the IBM Support Center, they will provide advice about which traces to turn on and further instructions, if required.

#### *Mainframe*

When adding tracing to a batch process, include =FD in the jobcard along with the trace request:

```
//RDB2EXTR EXEC PGM=PDPMAIN,
//  PARM='CON BATCH DSNCR PSRDS PK PSTRDS PSTRDS =TRACE=A =FD',
```

The trace file is in PSTRACE DD of SYSOUT.

When running processes online, include the parameter when you invoke the product, for example:

```
pst parm(=fd)
```

After invoking the product using this command, enter TRACE ON from the command line to enable the tracing.

## Microsoft Windows

The location of the trace files on Microsoft Windows is specified in Personal Options for the Temporary Working Directory. See Figure 12-7.

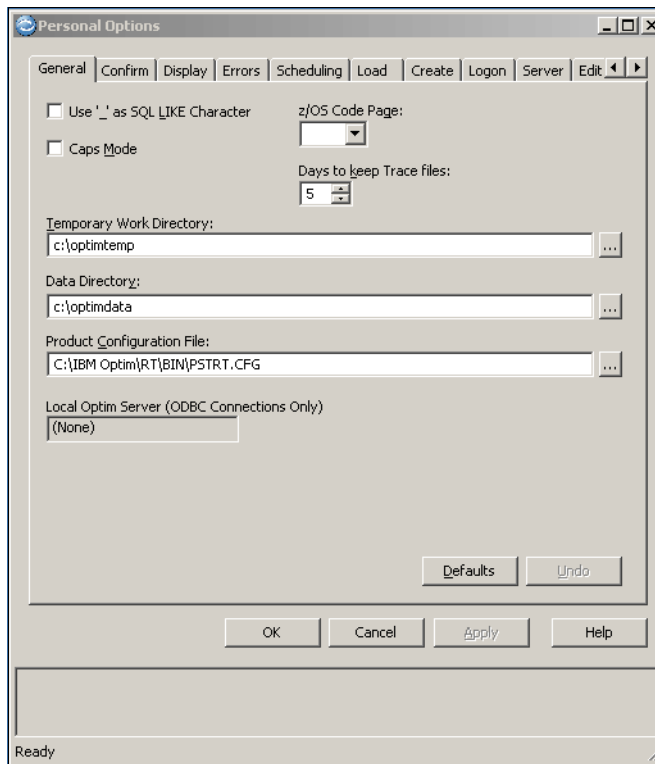


Figure 12-7 Trace file location on Microsoft Windows

Because the trace files get reused, be sure to look at the date and timestamp of the trace file and get the most current trace file, as shown in Figure 12-8.

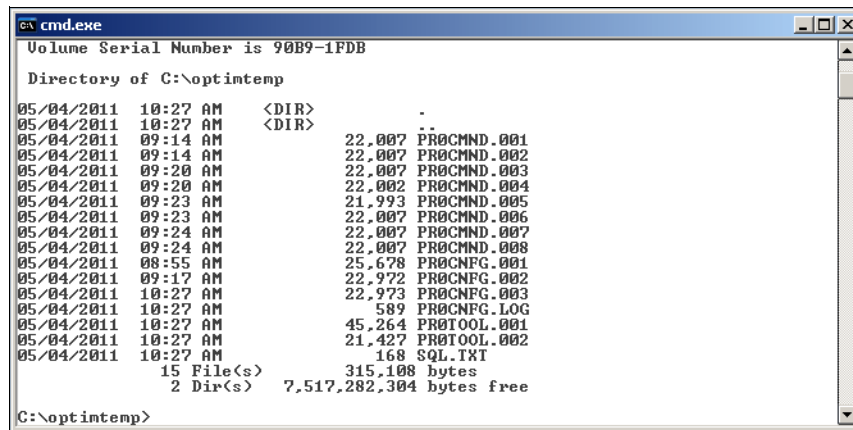


Figure 12-8 Date and timestamp of the trace files

Next, we show the common trace requirements and how to invoke them. The following trace invoking steps are examples only. You might need to invoke alternate or additional traces, as directed by IBM Support:

1. Click **Help** → **About Optim option**.
2. Double-click the logo in the upper-left corner, which opens the Trace Settings panel (Figure 12-9).

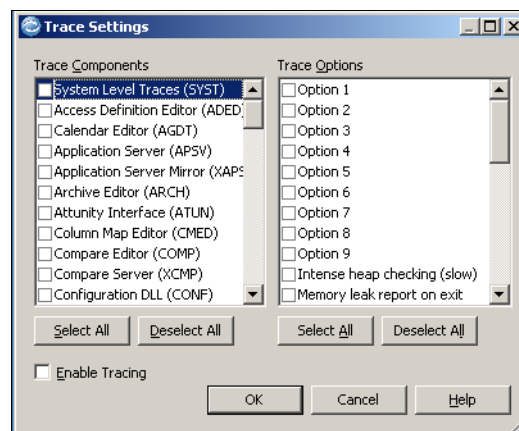


Figure 12-9 Trace Settings

3. Select **Trace Component for NEC Data Processes (DPP)**.

4. Select trace options **Option 30** and **Option 31**.
5. Select **Trace Component for NEC Dynamic SQL (DSQ)**.
6. Select trace option **Option 25** and **Trace SQL statements from DSQL\_ExecStmt and DSQL\_OpenCursor**.
7. Select **Enable Tracing** in the lower-left corner.
8. Click **OK**.
9. Run the Archive and Delete processes.
10. Close the product.

Depending on the origin of the trace information, the following output trace files are created:

- ▶ `PROT00L.nnn`: Contains detail from the Microsoft Windows GUI.
- ▶ `PROCMND.nnn`: Contains detail from the command-line-invoked processes.
- ▶ `PROSVCE.nnn`: Contains detail from the Optim server processes.
- ▶ `PROCNFG.nnn`: Contains detail from the Optim configuration.

The extension is sequential and is reused so you need to check the date and timestamp for when the product was closed in Step 10. To locate the files on the server, look in the `PROT00L.nnn` file that was produced for this run and search for the word “chain”. You have to send each of the files that are recorded in each of the “chain” references in the server to IBM support.

*Remember to turn tracing off when it is no longer required.*

## UNIX

On a UNIX system, perform these steps to run tracing:

1. Stop the Optim server by using the **rtserver stop** command.
2. Open the `PSTSERV.CFG` file in the `/rt/etc/` directory.
3. In the bottom of the file, there is a trace section. Enter the trace that you want to include, for example, trace DSQL 25. On the next line, enter trace enable.

Example 12-1 shows an example of the file contents.

*Example 12-1 Trace section in the configuration file*

---

```
# trace
#
#   Traces should be entered only on advice from Princeton Softech Technical Support.
#
#   format is:
#
#       trace component ALL
#       trace component (numbers 1 to 32 separated by blanks)
#       trace enable | disable
```

```
#
# Windows counterpart: Control Panel Applet's About - Trace Dialog
trace DPP 30 31
trace DSQL 25
trace enable
```

---

Save the file.

4. Start the server daemon using the **rtserver start** command.

To find the location of the trace files, look in the PSTSERV.CFG file in the /rt/etc/ directory and see what directory is listed for the **tempdir** parameter. You need to send the files, which were generated during the time that the archive request ran, to IBM Support. There is a PROCMND.*nnn* file (if you used the command line) or PROSVCE.*nnn* file, as well as a PROSVCE.*nnn* file.

It is important to turn tracing off when it is no longer required. You can disable additional tracing by clearing the **Enable Tracing** option on the Trace Setting Panel on the workstation. For the server side, you can stop the Optim server and comment out the two lines that were added, save the file, and then start the Server daemon again.

## Open data manager traces

Tracing is also available within the open data manager (ODM). To enable traces in ODM, perform the following steps:

1. Edit the NAV binding.
2. Turn timeTrace to TRUE.
3. Turn generalTrace to TRUE.

The following example enables tracing in ODM:

```
<environment name='NAV'>
  <misc codepage='UTF-8' language='ENUS' nlsString=''/>
  <debug generalTrace='true' timeTrace='true'
    traceDir=''/>
  <queryProcessor/>
  <optimizer/>
  <transactions/>
  <odbc/>
  <oledb/>
  <tuning/>
</environment>
```

Within the Optim Connect Tmp folder on Microsoft Windows or the navroot/tmp folder on UNIX, you can see relevant traces, such as Nav.log and irpcd.log, which contain communication traces. You also see error logs, such as Nav\_0.log in the temporary folder, as shown in Figure 12-10 on page 482.

Name	Size	Type	Date Modified	Attributes
admin_svc0.log	1 KB	Text Document	5/9/2011 5:16 AM	A
admin_svc1.log	1 KB	Text Document	5/9/2011 5:16 AM	A
ivp.log	1 KB	Text Document	3/26/2011 3:15 PM	A
nav.log	1 KB	Text Document	3/26/2011 3:16 PM	A
Nav_0.log	1 KB	Text Document	5/9/2011 5:16 AM	A
Nav_1.log	1 KB	Text Document	5/9/2011 5:16 AM	A
status.flg	1 KB	FLG File	3/26/2011 3:15 PM	A

Figure 12-10 Trace files

## 12.3 Issues getting data out of sources

Now that you know what your Optim environment is like and the aids that are available, you can consider what type of problems you might face. This section describes issues getting information out of the data sources and general tips on managing the Optim server.

### 12.3.1 Troubleshooting your Optim server

If you have trouble with connections between workstations and servers or after changes have been performed, you might need to review the installation and configuration actions to validate that your server is available and works correctly. You received an installation and configuration document with your product. Our aim in this section is not to replicate the *Install and Configuration Guide*, but to highlight areas that might need validating to ensure that your archiving solution works correctly.

If you fail to connect to your Optim Server from the Optim Workstation, as shown in Figure 12-11 on page 483, you need to validate that your server is available and works correctly.

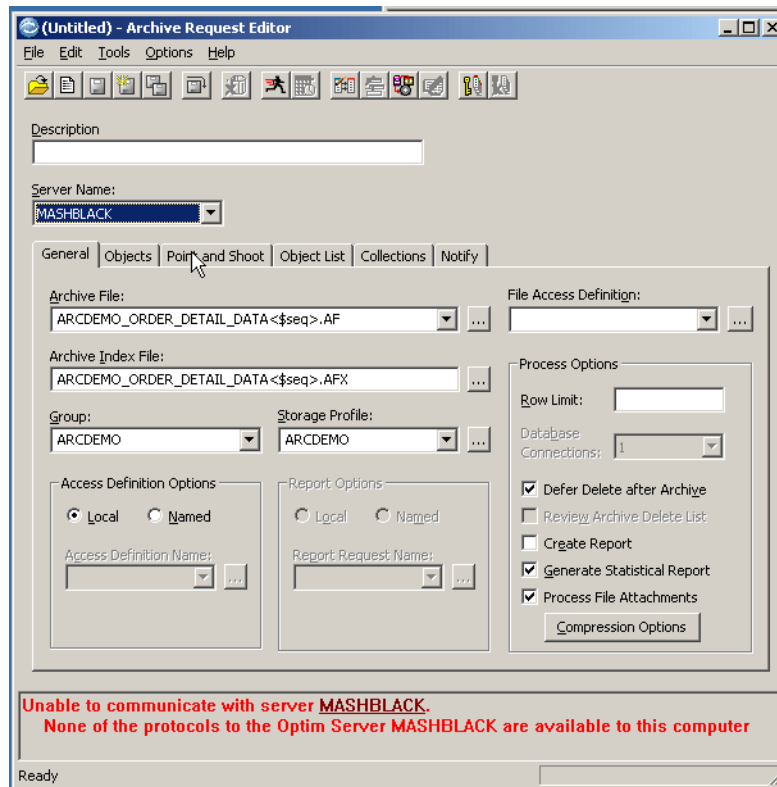


Figure 12-11 Example error message when the Optim server is unavailable

For Microsoft Windows installations, you can perform most of the tasks to validate your server availability through the available GUI interfaces. The configuration of your Microsoft Windows environment as a server adds Optim into the Microsoft Windows Control Panel. Go to **Start** → **Control Panel** → **Optim**. The Optim Server Settings window opens. In this window, you see tabs for a variety of elements, including connections, access, and server status. You can monitor, start, and stop Optim.

Figure 12-12 on page 484 shows where you can check that your Microsoft Windows Optim Server is available and ready for work.

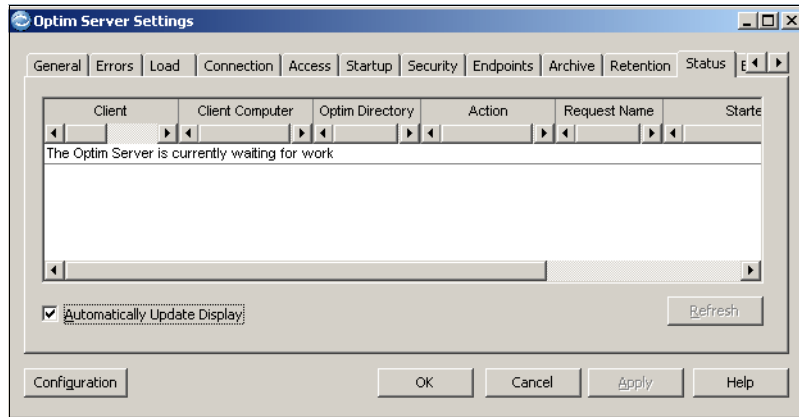


Figure 12-12 Status tab in Optim Server Settings window

If your server is not waiting for work, you might have to restart it. You can restart it from the **Startup** tab that is shown in Figure 12-13.

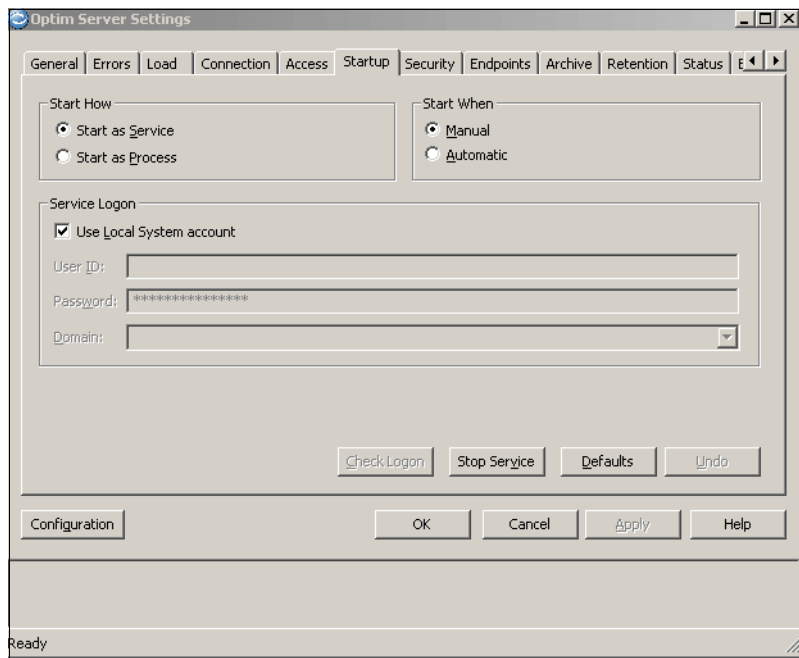


Figure 12-13 Startup tab in Optim Server Settings window

If you work with a Linux or UNIX server, check that your server is available. Check if the server daemon is running by using the following command:

```
pr0svce -d
```

Figure 12-14 shows the command output if the server is stopped.

```
[loptim@optimsrv4 ~]$ pr0svce -d
There are no active pr0svce daemons
```

Figure 12-14 Example command output if the server is stopped

You also can use the **rtserver list** command, as shown in Figure 12-15.

```
[loptim@optimsrv4 ~]$ rtserver list
Active client processes for pr0svce daemon in pid 19837
  PID Client      Computer      Pst Dir      Action      Request
  Started          Duration
(No clients are currently active)
```

Figure 12-15 Example command output if the server is started

If you have to start your Optim server because there are no active pr0svce daemons running, run the **rtserver start** command, as shown in Figure 12-16.

```
[loptim@optimsrv4 ~]$ rtserver start
Killing running Mainwin applications...
No display defined - don't clean xprops
Seems that there are no applications running.
No display defined - don't clean xprops
MainWin System Core Services stopped.
MainWin System Core Services version 5.2 Started.
```

Figure 12-16 Example command output to start the Optim server

The **rtserver start** command is a superset of commands that can include cleanup activities, as well as commands to start the relevant services, including the pr0svce daemons and the Mainwin services.

If your server fails to start, check the **PROT00L.xxx** file for error messages.

Because your configuration files contain information that is loaded when the server starts, you might need to validate them outside of the server start-up. Validate these configuration files by using the **pr0svce -v** command, which provides the output that is shown in Figure 12-17 on page 486. Error messages contain information about the type of error and on which line within the configuration file the error occurs. Rectify the mistake, save the file, and rerun the verification until it is successful.

```
[optim@optimsrv4 ~]# pr0svce -v
Reading Configuration File '/opt/IBM/Optim/rt/etc/pstserv.cfg'
Configuration file verified, no errors detected
[optim@optimsrv4 ~]# _
```

*Figure 12-17 Example of command output to verify configuration file*

On occasion, the Optim server might not stop after using the **rtserver stop** command initially, as shown in Figure 12-18.

```
rtserver stop
```

```
[optim@optimsrv4 ~]# rtserver stop
pr0svce daemon in pid 19837 will shut down after last client disconnects
```

*Figure 12-18 Example command output to stop the Optim server*

Check if there are any processes still running with this command:

```
pr0svce -d
```

If there are outstanding processes, you can use either the **rtserver kill** command or kill the outstanding process IDs (PIDs) individually. After you have gotten rid of all the PIDs, you must clean up the underlying Mainwin processes by using the following commands:

```
mwadm stop
mwcleanup -s
```

Figure 12-19 shows an example.

```
[optim@optimsrv4 ~]# mwadm stop
Failed to stop Core Services: There are still processes connected.
[optim@optimsrv4 ~]# mwcleanup -s
Killing running Mainwin applications...
```

*Figure 12-19 Example command output to clean up Mainwin processes*

Table 12-3 on page 487 lists common, useful commands for managing your Optim server. `<PSTHOME>` and `$PSTHOME` are substitutions for the home directory where Optim is installed.

Table 12-3 Common commands

Description of action	Commands
Update the .bashrc and .bash_profile of the Optim process owner	<PSTHOME>/rt/rtsetenv export PATH=\$PATH:<PSTHOME>/rt/:<PSTHOME>/rt/bin: <PSTHOME>/etc
Validate the configuration files	# pr0svce -v # pr0cmd -v
To sign the default exit	# pr0sign -d <company-id> "<company name>" company-password For example: # pr0sign -d 388713 "OptimDemo" 9863ZU  Be aware that company name and company password are both case sensitive.
Start up or shut down the Optim server	# \$PSTHOME/rtserver start # \$PSTHOME/rtserver stop
Check the status of the Optim daemon process	# pr0svce -d
To start the ODM server	# \$PSTHOME/navroot/bin/irpcd start
To stop the ODM server	# \$PSTHOME/navroot/bin/irpcd shutdown
To check the status of the Optim daemon process during ODM shutdown	# \$PSTHOME/navroot/bin/irpcd status
To encrypt the passwords that are entered in the configuration files, use the following command.	pr0pass -a <type> <name> <user ID> <password> where: <i>type</i> = pstdir, dbalias, or server <i>name</i> = name entered for dbalias, server name, or pst directory <i>User ID</i> and <i>password</i> are the login credentials for the database and server.
To delete the encryption	pr0pass -d <type> <name> <user ID>

**Important:** You must start the Optim server before starting the ODM server. Conversely, you must stop the ODM server before you shut down the Optim server.

You can obtain more details about all these commands in the manuals.

## 12.3.2 Cannot get to the source data

Source data is generally the data that is accessed by one of the following two methods:

- ▶ Optim's native DBMS support, such as Oracle, Sybase, SQLServer, and DB2
- ▶ Data accessed through the Optim federated data source middleware approach

If you have a problem getting to the natively supported DBMS access:

- ▶ Check the DB alias definition. Is the user, password, and connection string combination correct?
- ▶ Check DBMS access authorities for the user that is referenced in the DB alias.
- ▶ Check that the DB alias definitions are consistent between the Microsoft Windows workstation setup and the Linux, UNIX, or Microsoft Windows server configurations.

For sources which are not supported natively, check your connectivity both outside of Optim and through the same channels that Optim uses:

- ▶ To check the network access, use the **ping** command between the Optim client, Optim server, and the database sources.
- ▶ Check access through Optim Connect using the **nav\_util** command with the default connection information:

```
nav_util execute <datasource_name>
```

You also can use this command:

```
nav_util -b bindurl=localhost:2551 execute <datasource_name>
```

where `localhost:2551` must be replaced with the appropriate connectivity information for the server.

If this command works, you can move off of the server to a fat client and try to access the data source.

After you have verified that the connectivity to your source is not a problem, check if anything has changed recently, for example, code upgrade or the introduction of security.

Within Optim, the complexity of the data model or the volume of data can, in turn, cause or mask problems. It is important to try to isolate the problem by reducing any complexity and volume to ascertain if the problem is a core issue or a problem that is caused by a large amount of data. To isolate the problem, try to re-create the issue with fewer tables and row limits in place.

**Important:** Due to Optim's 32-bit nature, you must ensure that the 32-bit libraries are also available if your source data is on a 64-bit instance.

### 12.3.3 Poor archive process performance

Read this section in conjunction with the performance information in Chapter 10, "Optim performance considerations" on page 405.

Optim is similar to any other application. Perceived poor performance can be attributable to any of the components that form the solution. Understanding where the bottleneck lies is key to resolving poor performance. You can waste a lot of time if you do not eliminate the components that perform well.

Due to the high volumes of data that can be in transit during archiving processes, experience has shown that the network and storage system throughput can be problematic. You have to validate the performance of these systems with the appropriate tooling and support teams.

After you have discounted network and storage systems, the next element to consider is if the delay is in the data source, any connecting middleware, or Optim. Use standard DBMS and operating system performance tooling to establish where the bulk of time is being spent. This exercise might help you to discount issues, such as competing processes or parallel Optim processes holding locks on tables. The Statistics Report that is available as part of your archive process can also provide useful intelligence about where process time was spent during execution.

Remember that Optim accesses the source tables in a similar manner to any other application accessing a database. The good performance of the database under the Optim workload is extremely important to archive processing. Therefore, check the simple things first:

- ▶ Have you checked your access plan?
- ▶ Are RUNSTATS current?
- ▶ Does the Optim buffer match the row length?

The default buffer length of 512K is usually a good match. However, where rows exist that are longer, Optim uses two buffers and consequently more network traffic will ensue.

- ▶ Do the tables that are involved in the archive definition have indexes defined?

Too many indexes can cause problems, as well as too few. Use the "Key Lookup Limit" in the archive definition to limit the number of keys to look up in the database at any one time.

To implement for client/server, select **Advanced Options** from the Option menu in the archive definition. You then see the “Key Lookup Limit” section that is enabled in the Relationships panel.

The performance tips that we have covered so far are generally applicable. The next two sections might or might not apply, depending on your implementation.

## Archive process issues

If you think the archive process is hanging, check these things:

- ▶ There might be an archive action defined for the process that hangs? For instance, if the process hangs during a delete, is there a “Before Delete of A Row” action (or something similar)? If the action is removed, does the process still appear to hang? You need to know if there is possibly something in the action that causes the archive process to hang.
- ▶ Mail notifications can also cause hangs if Optim is unable to complete the action.
- ▶ It is possible that you have cyclical relationships in your archive definition. Show Steps and good archive definition design, as covered in Chapter 10, “Optim performance considerations” on page 405, can help identify and resolve this situation.

## Federation

Two types of attributes can affect performance on a federated data source:

- ▶ The server attributes: These attributes are the way in which the local DB2 and the remote federated database interact with each other.
- ▶ The nickname attributes: These attributes are the objects that are associated with the local nickname for a remote table.

See the chapters that describe federation and the Federation Server documentation for further details. Here, we concern ourselves with describing the changes that can affect performance.

The following commands are specific to the instance of DB2. You are only required to issue these commands one time from a DB2 Command Window:

```
DB2 UPDATE DBM CFG USING KEEPFENCED NO
DB2 UPDATE DBM CFG USING ASLHEAPSZ 256
DB2SET DB2_FMP_COMM_HEAPSZ=2000
```

KEEPFENCED prevents orphan fenced procedures from retaining memory and keeping connections open.

ASLHEAPSZ increases agent memory for passing back SQL result sets.

You must issue a DB2 ACTIVATE command after each **db2start** to ensure that the database is always up and running. DB2 ACTIVATE allocates memory and stacks in advance to improve performance, for example:

```
DB2 ACTIVATE DB OPTIMDB2
```

Within Federation Server, you can affect the way that the DB2 optimizer determines how the SQL will be processed, what to ship to the data source, and what to process on the local server. Consider altering the server definition in the following ways:

- ▶ Change CPU\_RATIO to better match processing power differences. A value of 0.25 means that the data source server has four times more processing power than the Federation Server.
- ▶ Change IO\_RATIO to better match I/O rate differences. A value of 0.25 means that the data source I/O subsystem is four times faster than the Federation Server I/O subsystem.
- ▶ Change the network bandwidth COMM\_RATE if the network is fast. A value of 2 means that the network has a two megabit per second throughput.
- ▶ Set DB2\_MAX\_ASYNC\_REQUESTS\_PER\_QUERY to -1 to allow for parallelism, if possible.
- ▶ Set PUSHDOWN to Y.
- ▶ Set DB2\_MAXIMAL\_PUSHDOWN to Y.
- ▶ Set Collating Sequence N to I.

With regard to potential performance improvements on nicknames, it is possible to add indexes on nicknames. After the index has been added, it is important to get the statistics for the table, which are obtained from the remote source.

To validate your index changes, run EXPLAIN to ensure that the access paths have improved.

## **z/OS**

When questions arise surrounding the access strategy that has been chosen while performing an archive process on z/OS, we have a utility, **\$\$STRGTY**, that displays the chosen strategy when performing the archive.

To invoke this utility, you must create an access definition where **\$\$STRGTY** is the high-level qualifier of the archive definition name, as shown in Figure 12-20 on page 492.

```

Type of Access Definition to Use for Archive ==> P (P-Perm T-Temp)

If Permanent Specify New or Existing Access Definition Name
Group ==> $$$TRTGY
User ==> ACCESS
Name ==> STRAT

```

Figure 12-20 Example of using \$\$\$TRTGY utility within an access definition

After the archive definition is created, run the archive process in batch. After the completion of the job, in the output log, you can see the access strategy information. Figure 12-21 is an example.

```

06:17:05 Extracting Primary Key and Foreign Key Definitions
06:17:05 Extracting Index Definitions
06:17:06 Extracting View Definitions
06:17:06 Extracting Alias Definitions
06:17:07 CPU Time for Initialization = 0.572 seconds
Total CPU Time = 0.572 seconds
06:17:07 PSTTJN.CUSTOMERS Tbl Rows: 0 Total Rows: 0
06:17:07 CPU Time for CUSTOMERS = 0.113 seconds
Total CPU Time = 0.685 seconds
06:17:07 PSTTJN.CUSTOMERS Tbl Rows: 704 Total Rows: 704
06:17:07 Determine Access Strategy for PSTTJN.ORDERS
Row Count = 1709
FK Lookup for Rel RCO, NumFK=704
FKSelectCost=1.711017E+01, ThisFKCost=1.204556E+04
Total NumPK=0, Total NumFK=704
Total PKLookupCost=0.000000E+00, Total FKLookupCost=1.204556E+04
Scan Cost w WHERE = 5.235286E+02
Strategy = FK_SCAN
06:17:07 CPU Time for ORDERS = 0.280 seconds
Total CPU Time = 0.965 seconds
06:17:07 PSTTJN.ORDERS Tbl Rows: 1709 Total Rows: 2413

```

Figure 12-21 Access strategy in an archive job's output

This information is useful for determining on a table-by-table basis whether a SCAN or KEY\_LOOKUP was used to access the data. We only use this output when absolutely necessary, because there is overhead, and depending on the number of tables, it can produce significant output.

## 12.3.4 Deleting data from sources

Here, we look at a few of the issues that can affect the deletion of data.

### Validation to delete data from sources

If you delete data from the source tables using Optim, you can use one of these methods:

- ▶ Compare row ID.
- ▶ Compare row content.

Comparing the row content can provide you absolute confidence that you are only deleting exactly the same information that is held in the archive file. However, you have to balance the performance of this approach with the need for validation. Consider that to delete a row, Optim identifies each row in the source data and the archive file and then compares the contents of each row to determine if it needs to be deleted.

## Database-driven delete

If you drop database partitions to remove large amounts of transactional data, you can use statistically valid comparisons or the row count saved in an audit table to validate that you have archived the same data you are about to delete. Both of these methods for validation occur outside of Optim and need to form part of your operational procedures.

## Delete performance

If you use Optim to delete data after it has been successfully archived, indexing is key to delete performance and it is a matter of balance. Follow these suggestions for better delete performance:

- ▶ Define a primary key:
  - Either a DBMS primary key or an Optim primary key over a unique index.
  - Define your primary key as small as possible. If you are building an Optim primary key and there are multiple unique indexes, choose the smallest for your primary key.
  - Any index is generally better than no index. If only a non-unique index is available, use it.

- ▶ Are there multiple indexes defined on the tables in the delete process?

Updating a large number of indexes when a delete occurs in the database can negatively affect the performance of the delete process.

Consider the following options:

- ▶ Bypass the “Compare Row Contents” option if you are confident that the archived data is unchanged since the archive was taken. You can clear this option on your delete request.
- ▶ Clear the “Include LOB columns in row comparison” option on your delete request if one or more of your tables contain large object (LOB)-type columns, and you are confident that the data has not changed since the archive was taken.

- ▶ On large tables (many rows in the archive file to delete), consider using the “Force Key Lookup” access method and setting the key lookup limit to a value greater than 1. You can only adjust the key limit value if the following conditions are true:
  - The table has a unique primary key defined.
  - You are not using the “Compare Row Contents” option.
  - The delete process has no row-level delete actions defined.

**Key limit value:** Adjusting the key limit value is only available if the table resides in a Sybase, Informix, or SQL Server database. Optim always uses multiple key-style delete operations for DB2 on Linux, UNIX, or Windows and Oracle databases, when the other required conditions are met.

- ▶ Check that you are using multiple database connections if possible to allow concurrent processing of multiple leaf tables within your delete process.  
You must have enabled parallel database connections in the Optim Product options.
- ▶ Use the “Lock Tables” option in your delete request:
  - This option is appropriate if concurrent access to the tables is not required.
  - The commit occurs at the end of processing each table, so consider the effect on rollback or redo areas in your DBMS.

Remember that Optim deletes child rows before parent rows where referential integrity is enforced by the database. Where relationships are defined by Optim, tables are processed as independent tables and can be processed concurrently.

## 12.4 Issues accessing data after it has been archived

You have archived your data successfully. Now, we discuss issues that might arise when you try to access your data from the archive files.

### 12.4.1 Open data manager

ODM provides access to data in the archive files that you have created. This section looks at several of the problems that you might encounter in ODM while trying to get to your data.

#### Native application access

When using native application access (NAA), you might receive a message in the application that indicates a permission, object not found, or other unexpected

error. One possible cause is a failure to replicate certain objects from the production schema to the shadow schema. To overcome the error, analyze the query that generated the error for references to objects that are missing from the shadow schema. If any references to missing objects are identified, the referenced objects must be replicated to the shadow schema.

## Open data manager bindings and code pages

When you access data that has been archived out of multiple databases, be aware of the implications of differing code pages on accessing the data.

If all the databases have the same code page, it is possible to use a single NAV binding to access all the archives and collections within ODM.

When the sum total of all archive file code pages cannot be covered by a single setting that handles all code points, a new binding is needed. The default binding, NAV, is only a default. You can create numerous additional bindings.

**Important:** Each binding requires its own workspace.

Example 12-2 shows an example of a NAV binding with a UTF-8 code page.

### *Example 12-2 NAV binding with a UTF-8 code page*

---

```
<environment name='NAV'>
  <misc codepage='UTF-8' language='ENUS' nlsString=''/>
  <debug generalTrace='true' timeTrace='true'
    traceDir=''/>
  <queryProcessor/>
  <optimizer/>
  <transactions/>
  <odbc/>
  <oledb/>
  <tuning/>
</environment>
```

---

Finally, you must consider the DB2 server database in which the nickname is defined. In the case of UTF-8 as an example, it is possible that your DB2 database is already defined as UTF-8. In that case, everything matches in this example and nothing further is required for this wrap or server definition (underneath which you define your nicknames to map the ODM tables). However, if your DB2 server is on Microsoft Windows, it is possible that your database was defined as single-byte ISO Latin using Microsoft Windows code page 1252. It can be checked quickly by connecting to the database for which you need to check the code page and running the GET DATABASE CONFIGURATION

command at a DB2 command line. Within the early part of the output, you see lines similar to these lines:

Database territory	= US
Database code page	= 1208
Database code set	= UTF-8

If this code page does not match the ODM binding code page, the CODEPAGE attribute of your wrapper or server must be set to match the code page of the ODM data source. The following command is an example of how the CODEPAGE attribute can be changed:

```
ALTER SERVER <your server> OPTIONS (ADD CODEPAGE '1208')
```

You can obtain details about these commands in the DB2 manuals.

## 12.4.2 Security

If you do not get any data returned when you expected data, but no errors have been reported either, you might, if Optim security is in use, have setup issues with the file access definition (FAD). Because Optim is working as designed, this situation can be difficult to identify, because no error messages are returned.

Check that the archive files that you are trying to access have the appropriate security in place, that is, that they are covered by an appropriate FAD profile within Optim. Check that your user ID is included in the FAD to allow access to the data that is protected by the FAD.

## 12.4.3 Poor query performance on archived data

ODM provides you the flexibility to access the contents, which have been written into an Optim archive file through any SQL ODBC or Java Database Connectivity (JDBC)-complaint tool, application, or utility by using the Optim driver, without the need to restore the data in the files into a live database. Because the ODM is not truly a database, its performance, tuning, and operational considerations differ.

Figure 12-22 on page 497 shows a typical access to archive files.

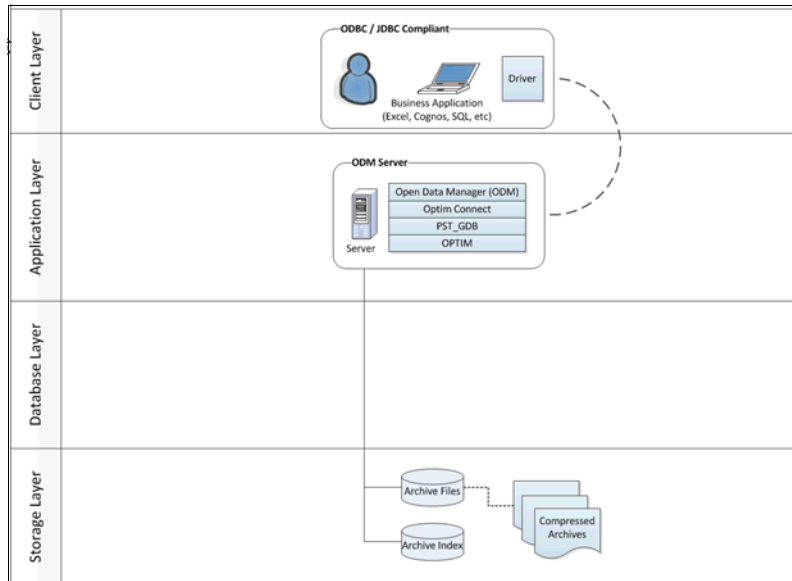


Figure 12-22 Archive file access architecture

Both 10.2.8, “Optim archive processing steps” on page 410 and 10.5, “Analyzing performance with Optim Connect” on page 422 provide extensive details about how to use the **Explain** facility to aid with the development of indexes on archive files.

Here are few of the performance considerations for archive access that you can review and incorporate during problem resolution.

## General points

This section helps you to pin down the area that might contribute to your poor performance:

- ▶ Capture and review the query showing performance degradation.
- ▶ Use the **Explain Plan** command in Optim Connect to review the execution plan and the need for an appropriate archive index.
- ▶ Consider if a query rewrite might help improve execution.
- ▶ Consider if putting an archive index on one or more columns of the table, which is being queried in the archive file, can improve search performance.
- ▶ After putting on an index, ensure that the query picks up the index. The **Explain** utility shows a change in execution strategy from a scan to used index, which is an indication that the new index is being used.

- ▶ Note the change in the cost of the plan. If the cost improves, evaluate if the gain is beneficial enough in terms of the improvement in performance versus the amount of storage space that it consumes.
- ▶ On the host where ODM is running, check the CPU usage and disk I/O using a suitable utility to provide additional information about unavailable system resources where there might be an improvement.

## **General considerations for adding indexes**

Adding indexes does not add unlimited improvement, and indexes are not always good. Here are several limitations to consider when deciding on your course of action for problem resolution:

- ▶ Put only as many indexes on the archive file as needed. A cost is associated with adding each index in terms of the required storage, because archive indexes are created in a non-compressed state.
- ▶ Evaluate the additional time and resources that are required to build indexing on each archive file. The added time and resource will affect your archive create schedule as more indexes on the file are needed. If you work under a strict operating window, consider the additional time that is involved.
- ▶ There can be no more than 16 archive indexes for each table.
- ▶ You cannot create an index on a long data-type column or a column that has more than 255 characters.
- ▶ The Optim Connect query engine can make use of only one index at a time.

## **Criteria for index creation**

This section identifies areas of specific interest after you have decided that you need additional indexing:

- ▶ Create an appropriate archive index that is based on selection criteria.
- ▶ Define the index on a column with high cardinality.
- ▶ If any functions, for instance, AVG(), SUM(), or UPPER(), are applied on the indexed column in the WHERE clause of the query, the index will not be used.
- ▶ Predicate order does not matter in the SQL statement.
- ▶ No index intersection or union is allowed. Only one index can be used at one time.

## Optim connect studio general suggestions

Here are certain Optim connect studio options to consider when investigating poor performance:

- ▶ For each binding configuration, an environment is set up using environment properties. Although the default values are preferable, review the *IBM Optim Connect User Guide and Reference Manual* for a complete list of properties and values to decide which option is applicable to your situation for optimal query performance. The following values are the best values to use to tune the query optimizer and processor:
  - DSM Max Buffer Size (dsmMaxBufferSize) – 100000000
  - DSM Mid Buffer Size (dsmMidBufferSize) – 10000000
  - DSM Max Hash File Size (dsmMaxHashFileSize) – 2000000000
  - DSM Max Sort Buffer Size (dsmMaxSortBufferSize) – 100000000
  - Hash Buffer Size (hashBufferSize) – 100000000
  - Hash Max Open Files (hashMaxOpenFiles) – 500
- ▶ The segments section of a table definition describes the segments of the key. The first segment must be in the WHERE clause of the query, because it determines index eligibility. Lower segments are used if the prior segments have been used.
- ▶ RANGE causes the lower segments to not be used, so put RANGE at the end of a compound index.
- ▶ Use the **EXPLAIN** utility in `nav_util`, for example:

```
explain select count(*) from optim_details where item_quantity = 10;
```
- ▶ Use hints in SQL Queries:

```
select count(*) from optim_details <access(without scan)> where  
item_quantity = 10 or item_quantity = 20;
```

## 12.5 Recovery

The most important aspect of recovery is to plan ahead. Recovery is not possible if you have not taken backups and thought about potential failure scenarios. Chapter 9, “Deployment and operational considerations” on page 375 has introduced considerations for recovery within an operational environment and the product manuals cover how to perform the required tasks.

### 12.5.1 Optim objects

If your Optim directory becomes unusable due to storage problems or issues with the database where the directory resides, you can import the directory from the

exports that you have taken as part of your operational backup procedures to guard against failures.

Refer to the product manuals for the details about performing exports and imports.

## **12.5.2 Accessing a duplicate copy of archive files**

When setting up the storage profile for your archive files, it is possible to take a *duplicate copy* of your archive file through Optim, in addition to any third-party storage management activities. Typically, you use this file for disaster recovery purposes. Although the paths of the two copies of the archive file differ, the actual file names are the same; therefore, it is not possible to register the backup file in the live Optim directory.

We suggest that if you use this approach to provide disaster recovery capabilities, you create an alternate Optim directory and register the duplicate archive files so that you have a record of the files and their retention is managed as per established business rules. Ensure that you back up this disaster recovery Optim directory. If you find yourself in a disaster situation, you can recover your disaster recovery Optim directory for access to the duplicate archive files.



# Data model

In this book, we reference a data model to exemplify various scenarios and events that might trigger the requirement to archive data from the underlying tables. Here, we describe the data model that is used.

This data store serves two major functions: creation and maintenance of reference, master, and transactional data and the recording of business event data. We have deferred from showing a physical database design and limited ourselves to a more logical representation of the model. The model is normalized and might require additional thought when referenced to an actual system. The model does not include any derived attributes, because they do not add anything to the information requirements for our examples.

The logical data model represents possible data requirements for enterprises. The data requirements do not include many of the business processing rules that might accompany the data model. The data model generally provides all the information needed to enforce business rules; however in many cases, you might need to develop additional business rules to supplement the data models. We provide examples of the need for business rules here for reference.

# A.1 Business event processing

Business processes in the form of events include purchasing, inventory management, preparing purchase orders, receiving, shipping products, accounting, and so on. When these events occur, business event data is captured to essentially describe *who*, *what*, *where*, and *when* about the event, and this business event data is stored in a relational database.

## A.1.1 Reference data

*Reference data* is the data that is used to classify or categorize other data, and it is used within a data-centric process. This reference data is dictated by certain business rules so the data conforms to several allowable values. This set of allowable values is called a *value domain*.

For instance, the industry standards (master reference data), such as postal code or state code, are used more globally. Other data might be internal to an organization, such as order status or product code, as shown in Table 12-4.

Table 12-4 Application\_Status

Status code	Description
N	New
C	Closed
P	In-progress
Z	Cancelled
I	Invalid

Reference data can change over time through transactions. For example, a logistical transaction can change the location of an object. A financial transaction, such as adding tax, can change the price of an object. And, a series of work transactions can change a virtual object, such as a planned building into a physical object.

## A.1.2 Master data

*Master data* is the data about the business entities that provides context for business transactions. Unlike reference data, master data values are usually not limited to predefined domain values. However, business rules typically dictate the format and allowable ranges of master data values.

Common organizational master data includes data about these subjects:

- ▶ Parties that include individuals, organizations, and their roles, such as customers, citizens, patients, vendors, suppliers, business partners, competitors, employees, students, and so on
- ▶ Products, both internal and external
- ▶ Financial structures, such as general ledger accounts, cost centers, profit centers, and so on
- ▶ Locations, such as addresses

Master data is the authoritative, most accurate data that is available about key business entities, and it is used to establish the context for transactional data. Master data values are considered “golden.” Historically, master data has moved into the foreground through the needs of back-end systems, such as data warehouses. However, all transactions and dimensions of information architecture, front end, middle, and back end, invoke and make use of master data, whether operational or analytic.

### **A.1.3 Transaction data**

Transaction data is the data describing an event, such as the change that happens as a result of an interaction. Data can be new or data that has been changed or deleted as a result of an associated event. This data must be left in a consistent state and be complete. A transaction cannot be partial. A transaction denotes an atomic unit of work that is treated as coherent, reliable, and independent of other transactions. Transaction data always has a time dimension, a numerical value, and refers to one or more objects, such as reference data.

The following examples are typical transactions:

- ▶ Financial: orders, invoices, and payments
- ▶ Work: Plans and activity records
- ▶ Logistics: Deliveries, storage records, or travel records

## **A.2 Order management system**

An order management system consists of several critical business processes and triggers, including order, shipment, receipts, customer, invoice processing, and electronic data exchange. These processes spawn important business metrics, such as sales volume and invoice revenue, that are key performance indicators for any organization that sells products or services to others.

There are several order management transactions, including the common characteristics and complications that are encountered during the life cycle of a business event. It is crucial to any archival project to understand these verticals in the data structures to logically carve a subset (or whole) of data to form a sound data archival strategy. Figure A-1 shows the typical components of an order processing subsystem.

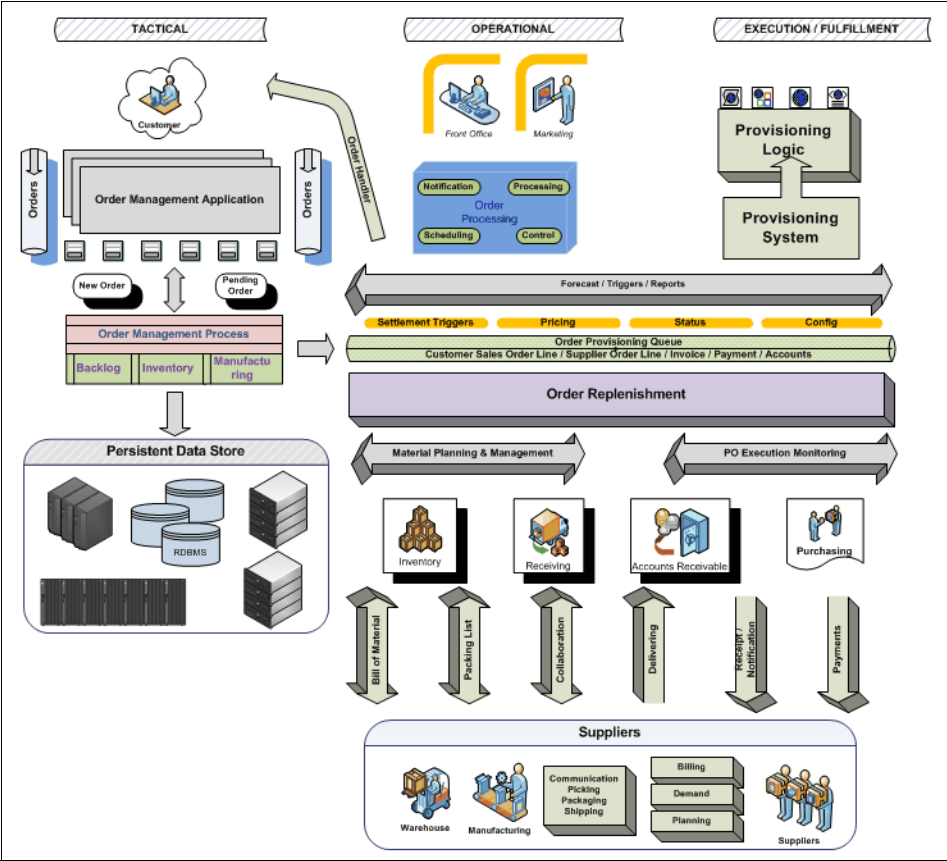


Figure A-1 Order management system

### A.2.1 Ordering products

In a standard data model, a supplier is related to one or more purchase orders, which have purchase order line items that are related to product. A customer can be related to one or more sales orders, which have sales order line items that relate to specific products. Figure A-2 on page 505 shows the relationships among these entities.

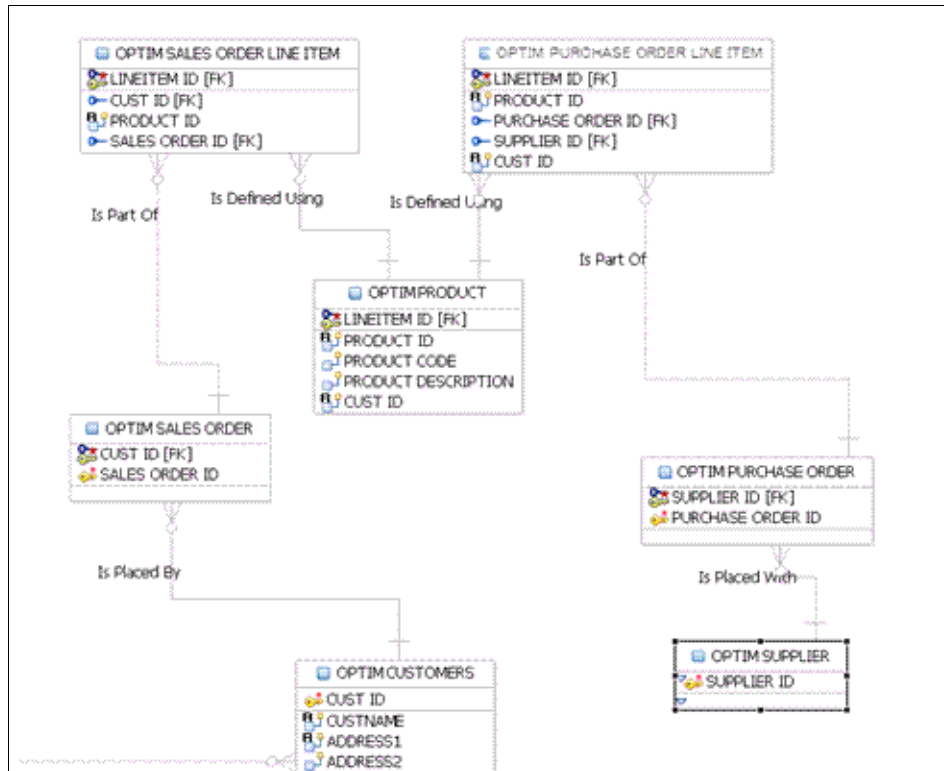


Figure A-2 Ordering products

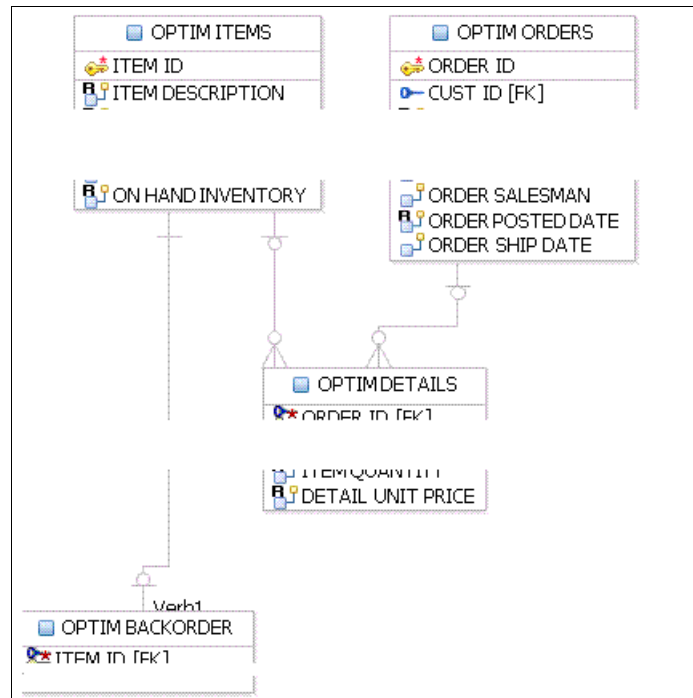
## A.2.2 Order transactions

The order details and order line items make up order transactions. The natural granularity of an order transaction is one row for each line item on an order. The following information is available in order transactions:

- ▶ Order quantity
- ▶ Amount associated
- ▶ Dates
- ▶ Back orders
- ▶ Returns
- ▶ Shipping information
- ▶ Order lines
- ▶ Items (or products)

In an order of magnitude, order transactions form one of the largest sets of table data.

You can archive selected closed orders if no open demand exists. You can select orders to archive based on a combination of criteria: an order number range, an order type and order category, an order date range and creation date range, and a customer. Figure A-3 shows a typical sales order's line items and its relationships to customers.



*Figure A-3 Order transactions*

### A.2.3 Invoice transactions

A sale or invoice typically occurs when a product is sold and shipped to the customer. The invoice governs the current shipment of products for a particular customer, for a particular order, and to a particular address. The invoice has multiple line items. Various prices and rates, discounts, and allowances are associated with each line item. The invoice has information related to the following areas:

- ▶ Customer
- ▶ Contracts and deals
- ▶ Off-invoice discounts and allowances
- ▶ Revenue that is generated by customers purchasing products
- ▶ Variable and fixed costs that are associated with manufacturing and delivering products

- ▶ Money left over after the delivery of the product
- ▶ Customer satisfaction metric
- ▶ Dates that are associated with perhaps many of these areas

### A.2.4 Items master data table

Items describe the complete portfolio of products sold by a company. This data can be large for organizations that sell a lot of products. The verbose description of the items sold is maintained in this table. As new products are added and old ones retired, this table maintains the history and usage of the products. Figure A-4 shows an Items table that lists products.

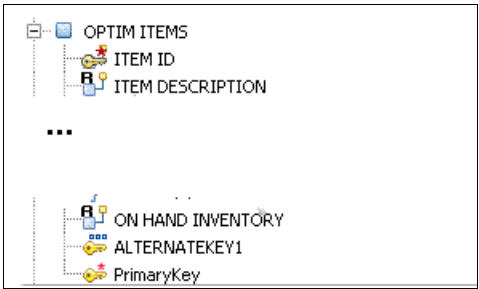


Figure A-4 Items master data table

### A.2.5 Customer

This entity contains information about any person, business, group, or association that is of a business interest or is involved with the organization. Each customer has one or more roles that designate why that customer is of business interest. You can capture additional information about a customer, such as the programs in which the customer participates, demographic profiling, analytical scoring, segmentation, channels, and privacy preferences.

For archival, you can select customers and orders that are based on a combination of criteria: customer types, the segment based on geography, demographics, status, score, and ratings, as well as date ranges. Figure A-5 on page 508 shows a typical Customer table.

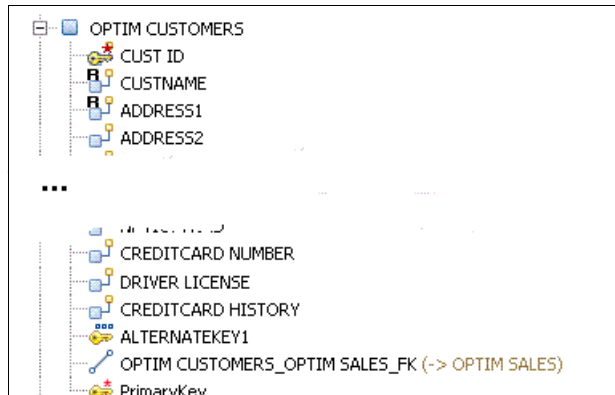


Figure A-5 Customer table

## A.2.6 Shipments

The shipment entity relates to the customer and the shipping instructions. The master reference data to track postal code is used to complete shipping tracking from where the shipment began and where it was delivered. Other critical pieces of information, such as customer contact, shipment dates, shipping cost, handling instructions, and so forth are also available. These tables contain information for each discrete location to which the product is shipped.

The shipping details provide information regarding what items are shipped and their status. Each shipment might be detailed by many shipment items, therefore providing a mechanism to track what was shipped or received and how many items were shipped.

The Ship\_To table contains information regarding shipping address, receiving party's name, sales representative assigned, shipping instructions, and any other pertinent shipping details. A customer ship-to table can range from a moderate size (thousands of rows) to an extremely large size (millions of rows), depending on the nature of the business. Figure A-6 on page 509 shows the relationships among Customers, Ship\_To, and Ship\_Instr tables.

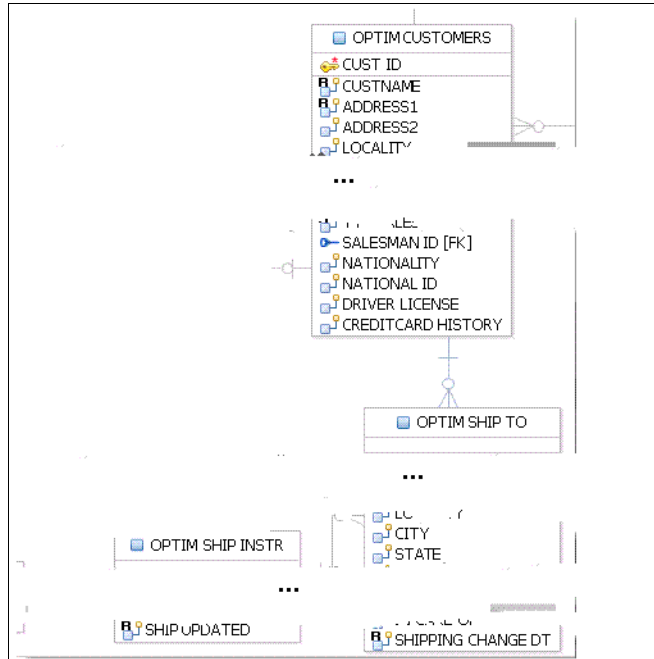


Figure A-6 Shipments

## A.3 Optim data model explained

The model that we describe next is a logical representation of the data structures, entities, and data attributes. At this time, we only describe it logically so that it is easier to understand conceptually the organization and its relationships. Refer to the *Optim Installation and Configuration Guide* for more details about the tables, columns, and relationships.

### A.3.1 Logical data model

Figure A-7 on page 510 shows how each entity relates to the other entities.

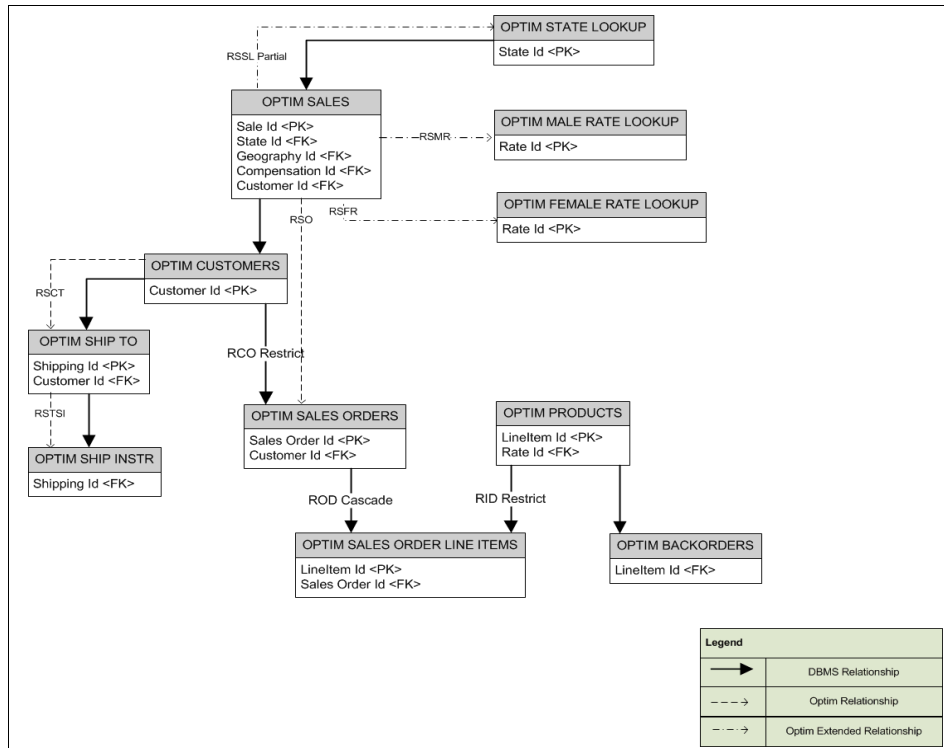


Figure A-7 Logical data model

This diagram shows the relationships among the tables:

- ▶ Arrows indicate the flow from parent to child.
- ▶ Solid lines represent relationships that are defined to the DBMS.
- ▶ Dashed lines indicate relationships that are defined to the Optim directory during training and demonstrations.
- ▶ Dotted lines indicate Optim extended relationships. The extended relationships can emulate implicit, or application-managed, relationships in your database. The extended relationships allow you to manipulate sets of relational data in the same manner as in your production environment.

### A.3.2 Optim sales table

The Optim\_Sales table identifies each salesperson by name, ID number, and manager.

### **Primary key**

The SALESMAN ID column is the primary key column.

### **Relationships to other tables**

The Optim\_Sales table is a parent of the following tables:

- ▶ The Optim\_Customers table, through a foreign key on the SALESMAN ID column
- ▶ The Optim\_Male\_Rates table, through an Optim data-driven relationship on the column AGE when SEX = 'M'
- ▶ The Optim\_Female\_Rates table, through an Optim data-driven relationship on column AGE when SEX = 'F'
- ▶ The Optim\_State\_Lookup table, through an Optim substring relationship using SALESMAN ID

## **A.3.3 Optim customers table**

The Optim\_Customers table contains customer names, ID numbers, and addresses.

### **Primary key**

The CUST ID column is the primary key column.

### **Relationships to other tables**

The Optim\_Customers table is a parent of these tables:

- ▶ The Optim\_Orders table, through a foreign key on the column CUST ID
- ▶ The Optim\_Ship\_To table, through an Optim relationship on column CUST ID

The Optim\_Customers table is a child of the Optim\_Sales table, through its foreign key on column SALESMAN ID.

## **A.3.4 Optim orders table**

The Optim\_Orders table contains information for orders, including the order number, customer ID, and salesperson.

### **Primary key**

The ORDER ID column is the primary key column.

### **Relationships to other tables**

The Optim\_Orders table is a parent of the Optim\_Details table, through a foreign key on the column ORDER ID.

The Optim\_Orders table is a child of the Optim\_Customers table, through its foreign key on the column CUST ID.

## **A.3.5 Optim details table**

The Optim\_Details table contains additional information for each order in the Optim\_Orders table.

### **Primary keys**

The ORDER ID and ITEM ID columns form the primary key.

### **Relationships to other tables**

The Optim\_Details table is a child of these tables:

- ▶ The Optim\_Orders table, through its foreign key on column ORDER ID
- ▶ The Optim\_Items table, through its foreign key on column ITEM ID

## **A.3.6 Optim items table**

The Optim\_Items table contains information about each item for an order, including the description, price, and quantity in inventory.

### **Primary key**

The ITEM ID column is the primary key column.

### **Relationship to other tables**

The Optim\_Items table is a parent of the Optim\_Details table, through a foreign key on the column ITEM ID.

## **A.3.7 Optim\_Ship\_To table**

The Optim\_Ship\_To table contains order shipping information.

### **Primary key**

The SHIP ID column is the primary key column.

### **Relationships to other tables**

The Optim\_Ship\_To table is a parent of the Optim\_Ship\_Instr table, through an Optim relationship on the column SHIP ID.

The Optim\_Ship\_To table is a child of the Optim\_Customers table, through its Optim relationship on the column CUST ID.

## **A.3.8 Optim\_Ship\_Instr table**

The Optim\_Ship\_Instr table contains detailed information for order shipping.

### **Primary key**

The SHIP INSTR ID column is the primary key column.

### **Relationship to other tables**

The Optim\_Ship\_Instr table is a child of the Optim\_Ship\_To table, through its Optim relationship on the column SHIP ID.

## **A.3.9 Optim\_Male\_Rates table**

The Optim\_Male\_Rates table contains insurance rates, which are based on age.

### **Primary key**

The RATE PER 1000 column is the primary key column.

### **Relationship to other tables**

The Optim\_Male\_Rates table is a child of the Optim\_Sales table, through its Optim data-driven relationship on the column AGE.

## **A.3.10 Optim\_Female\_Rates table**

The Optim\_Female\_Rates table contains insurance rates that are based on age.

### **Primary key**

The RATE PER 1000 column is the primary key column.

### **Relationship to other tables**

The Optim\_Female\_Rates table is a child of the Optim\_Sales table, through its Optim data-driven relationship on column AGE.

### **A.3.11 Optim\_State\_Lookup table**

The Optim\_State\_Lookup table contains state codes and corresponding abbreviations.

#### **Primary key**

The Optim\_State\_Lookup table does not have a primary key.

#### **Relationship to other tables**

The Optim\_State\_Lookup table is a child of the Optim\_Sales table through a substring relationship on the column DISTRICT using a SUBSTR of SALESMAN\_ID.

# Functions and actions access permissions by roles

Table B-1 breaks down the roles. It suggests IBM InfoSphere Optim Data Growth Solution features and the actions under the functions to which they must have access. We use the following notation in the tables:

- ▶ a: Allow
- ▶ d: Deny

*Table B-1 Suggested access of functions and the actions for each function*

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
<b>Create new actions</b>	N/A	N/A	N/A	a	N/A	N/A
Archive request	N/A	N/A	N/A	a	N/A	N/A
Compare request	N/A	N/A	N/A	a	N/A	N/A
Convert request	N/A	N/A	N/A	a	N/A	N/A
Convert request (local)	N/A	N/A	N/A	a	N/A	N/A
Delete request	N/A	N/A	N/A	a	N/A	N/A
Extract request	N/A	N/A	N/A	a	N/A	N/A

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Insert request	N/A	N/A	N/A	a	N/A	N/A
Insert request (local)	N/A	N/A	N/A	a	N/A	N/A
Load request	N/A	N/A	N/A	a	N/A	N/A
Load request (local)	N/A	N/A	N/A	a	N/A	N/A
Report request	N/A	N/A	N/A	a	a	a
Report request (local)	N/A	N/A	N/A	a	a	a
Restore request	N/A	N/A	N/A	a	N/A	N/A
Table editor	N/A	N/A	N/A	a	N/A	N/A
<b>Create new definitions</b>	N/A	N/A	N/A	a	N/A	N/A
Access definition	N/A	N/A	N/A	a	N/A	N/A
Access definition (local)	N/A	N/A	N/A	a	N/A	N/A
Column map	N/A	N/A	N/A	a	N/A	N/A
Column map (local)	N/A	N/A	N/A	a	N/A	N/A
Column map procedure	N/A	N/A	N/A	a	N/A	N/A
Column map proc (local)	N/A	N/A	N/A	a	N/A	N/A
Optim primary key	N/A	N/A	N/A	a	N/A	N/A
Optim relationship	N/A	N/A	N/A	a	N/A	N/A
Table map	N/A	N/A	N/A	a	N/A	N/A
Table map (local)	N/A	N/A	N/A	a	N/A	N/A
<b>Create security definitions</b>	d	a	d	d	d	d
Access control domain	d	a	d	d	d	d
File access definition	d	a	d	d	d	d
<b>Create utility definitions</b>	a	N/A	N/A	N/A	N/A	N/A

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Archive file collection	a	N/A	a	a	N/A	N/A
Calendar	a	N/A	N/A	N/A	N/A	N/A
Currency	a	N/A	N/A	N/A	N/A	N/A
Storage profile	a	N/A	a	N/A	N/A	N/A
<b>Editor options</b>	N/A	N/A	a	N/A	N/A	N/A
Create indexes during primary key index analysis	N/A	N/A	a	a	N/A	N/A
Create indexes during relationship index analysis	N/A	N/A	a	a	N/A	N/A
Create objects during create	N/A	N/A	a	a	N/A	N/A
Drop objects during create	N/A	N/A	a	a	N/A	N/A
Modify SQL during create	N/A	N/A	a	a	N/A	N/A
Modify SQL during primary key index analysis	N/A	N/A	a	a	N/A	N/A
Modify SQL during relationship index analysis	N/A	N/A	a	a	N/A	N/A
<b>File maintenance</b>	N/A	N/A	a		N/A	N/A
File deletion	N/A	N/A	a	N/A	N/A	N/A
File renaming	N/A	N/A	a	N/A	N/A	N/A
<b>Invoke action editors</b>	N/A	N/A	N/A	a	N/A	N/A
Archive request	N/A	N/A	a	a	a	N/A

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Compare request	N/A	N/A	a	a	a	N/A
Convert request	N/A	N/A	a	a	a	N/A
Delete request	N/A	N/A	a	a	a	N/A
Extract request	N/A	N/A	a	a	a	N/A
Insert request	N/A	N/A	a	a	a	N/A
Load request	N/A	N/A	a	a	a	N/A
Report request	N/A	N/A	a	a	a	a
Restore request	N/A	N/A	a	a	a	N/A
Table editor	N/A	N/A	a	a	N/A	N/A
<b>Invoke command line actions</b>						
Archive directory maintenance	a	N/A	a	a	N/A	N/A
Browse	N/A	N/A	N/A	a	a	N/A
Export	a	N/A	a	a	N/A	N/A
Import	N/A	N/A	a	a	N/A	N/A
Migrate and FMF	a	N/A	N/A	N/A	N/A	N/A
Restart and retry	a	N/A	a	a	a	N/A
Run	a	N/A	a	a	a	N/A
Table editor	N/A	N/A	N/A	a	N/A	N/A
<b>Invoke configuration actions</b>	a					
Apply Maintenance for DB Alias	a	N/A	N/A	N/A	N/A	N/A
Create/Update DB Alias	N/A	N/A	a	N/A	N/A	N/A

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Drop Optim Directory/DB Alias	N/A	N/A	a	N/A	N/A	N/A
Rename Optim Directory	a	N/A	a	N/A	N/A	N/A
Update DBMS Version for DB Alias	a	N/A	N/A	N/A	N/A	N/A
<b>Invoke definition editors</b>	N/A	N/A	N/A	a	N/A	N/A
Access definition	N/A	N/A	a	a	N/A	N/A
Column map	N/A	N/A	a	a	N/A	N/A
Column map procedure	N/A	N/A	a	a	N/A	N/A
DBAlias	N/A	N/A	a	a	N/A	N/A
Point and shoot	N/A	N/A	a	a	N/A	N/A
Primary key	N/A	N/A	a	a	N/A	N/A
Relationship	N/A	N/A	a	a	N/A	N/A
Table map	N/A	N/A	a	a	N/A	N/A
<b>Invoke options</b>						
Access control domain	d	a	d	d	N/A	N/A
Export security definitions	d	a	d	d	N/A	N/A
File access definition	d	a	d	N/A	N/A	N/A
Import security definitions	d	a	d	d	N/A	N/A
Product options	a	N/A	a	N/A	N/A	N/A
<b>Invoke utilities</b>						
Archive directory maintenance	a	N/A	a	a	N/A	N/A

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
Archive file collection	a	N/A	a	a	N/A	N/A
Archive index maintenance	a	N/A	a	a	N/A	N/A
Browse	N/A	N/A	N/A	a	a	N/A
Calendar	a	N/A	N/A	N/A	N/A	N/A
Create	N/A	N/A	a	N/A	N/A	N/A
Currency	a	N/A	N/A	N/A	N/A	N/A
Export	a	N/A	a	N/A	N/A	N/A
Import	a	N/A	a	N/A	N/A	N/A
Register archive file	a	N/A	a	N/A	N/A	N/A
Remove archive file directory entry	a	N/A	a	N/A	N/A	N/A
Restart and retry	a	N/A	a	a	N/A	N/A
Scheduling editor	N/A	N/A	a	a	a	N/A
Storage profile	a	N/A	a	N/A	N/A	N/A
<b>Run untitled actions</b>						
Archive request						
Compare request						
Convert request						
Delete request						
Extract request						
Insert request						
Load request						
Report request						a
Restore request						

	System manager	Security manager	Optim manager	Designer	Operator	Auditor
<b>Security tasks</b>	N/A	a	N/A	N/A	N/A	N/A
Export secured archive file	a	a	a	N/A	N/A	N/A
Import secured archive file	a	a	a	N/A	N/A	N/A
Modify file security with migrate	N/A	a	N/A	N/A	N/A	N/A
Report security privileges	N/A	a	a	N/A	N/A	a



# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that several publications referenced in this list might be available in softcopy only.

- ▶ *IBM WebSphere Information Analyzer and Data Quality Assessment*, SG24-7508
- ▶ *IBM InfoSphere DataStage Data Flow and Job Design*, SG24-7576

You can search for, view, or download IBM Redbooks, Redpapers, web docs, draft publications and additional materials, as well as order hardcopy IBM Redbooks publications, at this website:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Other publications

These publications are also relevant as further information sources:

- ▶ *IBM InfoSphere Classic Federation Server for z/OS, Version 9.5*
- ▶ *IBM Optim Installation and Configuration Guide*
- ▶ *IBM Optim Connect User Guide and Reference*
- ▶ *IBM Optim Archive User Manual*
- ▶ *IBM Optim Common Elements Manual*
- ▶ *Installing and Configuring your Optim Solution for z/OS with Classic Federation*

## Online resources

These websites are also relevant as further information sources:

- Manage Data Growth

<http://www-01.ibm.com/software/data/optim/manage-data-growth/>

- IBM Data Governance

<http://www.ibm.com/ibm/servicemanagement/us/en/data-governance.html>

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)



**Redbooks**

# Implementing an InfoSphere Optim Data Growth Solution

(1.0" spine)  
0.875" <-> 1.498"  
460 <-> 788 pages







# Implementing an InfoSphere Optim Data Growth Solution

**Understanding the InfoSphere Optim Data Growth solution architectures**

**Implementing InfoSphere Optim Data Growth solutions**

**Managing data growth with flexibility**

The Optim Data Growth solutions are consistent, scalable solutions that include comprehensive capabilities for managing enterprise application data across applications, databases, operating systems, and hardware platforms. You can align the management of your enterprise application data with your business objectives to improve application service levels, lower costs, and mitigate risk.

In this IBM Redbooks publication, we describe the IBM InfoSphere Optim Data Growth solutions and a methodology that provides implementation guidance from requirements analysis through deployment and administration planning. We also discuss various implementation topics including system architecture design, sizing, scalability, security, performance, and automation.

This book is intended to provide various data solution development professionals, a broad understanding about IBM Optim Data Growth solutions, and to provide guidance for implementing an optimal solution for managing their enterprise application data.

## INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

### BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)