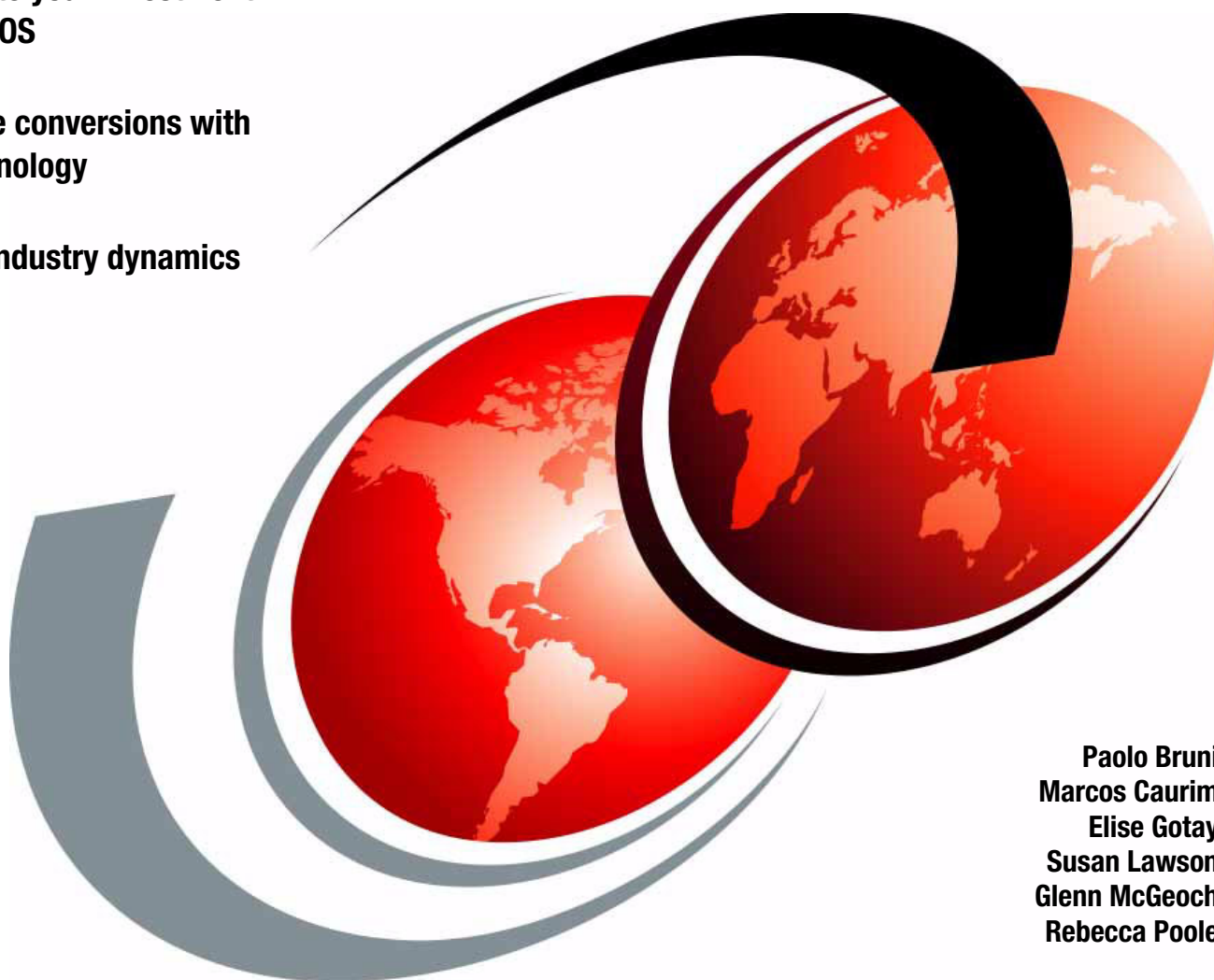


Streamline Business with Consolidation and Conversion to DB2 for z/OS

Consolidate your investment in DB2 for z/OS

Accelerate conversions with tools technology

Adapt to industry dynamics



Paolo Bruni
Marcos Caurim
Elise Gotay
Susan Lawson
Glenn McGeoch
Rebecca Poole

Redbooks



International Technical Support Organization

**Streamline Business with Consolidation and
Conversion to DB2 for z/OS**

September 2012

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

First Edition (September 2012)

This edition applies to IBM DB2 Version 10.1 for z/OS (program number 5605-DB2).

© Copyright International Business Machines Corporation 2012. 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

Figures	ix
Tables	xi
Examples	xiii
Notices	xv
Trademarks	xvi
Preface	xvii
The team who wrote this book	xvii
Now you can become a published author, too!	xix
Comments welcome	xix
Stay connected to IBM Redbooks	xix
Part 1. Introduction	1
Chapter 1. Business considerations	3
1.1 Historically speaking	4
1.2 Industry dynamics driving change	4
1.3 The reality of software conversions	5
Chapter 2. Converting to DB2 for z/OS	7
2.1 Converting to a relational DBMS	8
2.2 Additional benefits of a relational DBMS	8
2.3 Benefits of SQL	10
2.4 Data definition language	11
2.5 DB2 for z/OS is the platform of choice	11
2.5.1 Scalability, availability, and usability	12
2.5.2 Security	13
2.5.3 Database integrity	13
2.5.4 Continuous operation support	14
2.5.5 Centralized catalog	14
2.5.6 Recovery	14
2.5.7 Usability and productivity	15
2.5.8 Transaction manager support	16
2.5.9 Robust Utility Suite and unlimited tools	17
2.6 zEnterprise	20
2.7 Additional benefits of DB2 10 for z/OS	21
Part 2. Migration methodology	23
Chapter 3. Application and database conversion methodology	25
3.1 Mature application challenges on the z platform	26
3.2 Mature application options, examples, and terminology	27
3.2.1 Application and database conversion (manual or automated)	28
3.2.2 Application modernization	29
3.2.3 Application migration	29
3.2.4 Application replacement	29
3.2.5 Application rewrite	29

3.2.6 Application re-engineering	29
3.2.7 Migration	30
3.2.8 Do nothing or status quo	30
3.3 Conversion as a feasible solution	30
3.4 Manual conversions	32
3.4.1 Sample methods for manual conversions	33
Chapter 4. Automated conversion methodology	37
4.1 Conversion tool data access options	38
4.2 Automated conversion tool projects	39
4.2.1 Stage one: Assessment	40
4.2.2 Stage two: Pilot	42
4.2.3 Stage three: Conversion	42
4.3 Data conversion guidelines	43
4.3.1 Preparing for conversion testing	44
4.3.2 Addressing bugs and performance issues	45
4.3.3 Scheduling a dry run	45
4.3.4 Planning for a backout	45
4.4 Tool conversion strategies for 4GL programs	46
4.4.1 The 4GL dilemma	46
Chapter 5. Application and database conversion best practices	47
5.1 Summary of best practices	48
5.2 Stage one conversion planning	50
5.2.1 Application survey and initial project sizing	51
5.2.2 Fee-based assessment	53
5.2.3 Options risk analysis	54
5.3 Conversion skills and roles	54
Chapter 6. Choosing a migration strategy	57
6.1 Conversion strategies	59
6.1.1 Big Bang strategy	60
6.1.2 Loose coexistence strategy	60
6.1.3 Piece-by-piece strategy	61
6.1.4 Using transparency as a bridge strategy	61
6.1.5 Combining migration strategies	62
6.1.6 Comparing strategies	63
6.2 Automated conversion tool case study	64
6.2.1 Case study background	64
6.2.2 Identifying and conducting a pilot project	65
6.2.3 Automated assessment	66
6.2.4 Collecting, converting, and customizing the application and database	66
6.2.5 Application and database testing	66
6.2.6 Performance testing	67
6.2.7 Implementation tests	67
6.2.8 Production implementation	68
6.2.9 Post-conversion tuning	68
6.2.10 Relational redesign	68
6.3 IBM consultancy contact information	69
Part 3. Tool-assisted conversions	71
Chapter 7. Conversion solutions	73
7.1 Automation	74

7.1.1	Application conversion challenges	75
7.1.2	Data conversion	76
7.2	Engaging the conversion tool vendors	76
7.2.1	Qualifying vendor tools	77
7.3	Post project lessons learned	78
7.4	IBM InfoSphere solutions assist conversion process	79
7.4.1	Data replication	80
7.4.2	Data federation	81
7.4.3	Data transformation	81
 Chapter 8. Converting CA IDMS to DB2 for z/OS		
8.1	Performing a CA IDMS conversion	84
8.1.1	Re-engineering and rewriting the application	84
8.1.2	Purchasing a software package	84
8.1.3	Converting existing systems	84
8.2	CA IDMS requirements	85
8.2.1	CA IDMS key conversion factors	86
8.2.2	Mapping CA IDMS data to DB2	87
8.3	DB-Shuttle by ATERAS conversion solution	90
8.3.1	DB-Shuttle solution details	90
8.3.2	Additional tasks outside of the standard DB-Shuttle conversion	92
8.3.3	Assessment process	93
8.4	DB-Shuttle conversion process	95
8.4.1	Challenges of the customer	95
8.4.2	Environment of the customer	96
8.4.3	Project phases	98
8.4.4	Database and data conversion	101
8.4.5	Application conversion	106
8.4.6	Results	108
8.4.7	Lessons learned	110
 Chapter 9. Converting Adabas to DB2 for z/OS		
9.1	Natural and Adabas conversions	112
9.2	Adabas requirements	112
9.2.1	Conversion challenges	113
9.3	ConsistADS automated conversion solution	115
9.3.1	ConsistADS solution overview	116
9.3.2	ConsistADS solution	117
9.3.3	DB2 for z/OS tools	118
9.3.4	The ConsistADS products	118
9.4	How ConsistADS maps ADABAS data files to DB2	123
9.5	ConsistADS conversion process	127
9.5.1	Challenge of the customer	127
9.5.2	Solution architecture	128
9.5.3	Project phases	129
9.5.4	Data conversion	131
9.5.5	Converting the application	138
9.5.6	ConsistADS client results	141
9.6	Other solutions in the market	142
9.7	OnTarget	142
9.7.1	Adabas to DB2 for z/OS design	143
9.7.2	Features of the OnTarget solution	143

9.7.3 Customer conversion scenario	144
9.7.4 OnTarget conversion process	145
9.7.5 Results	147
9.7.6 Lessons learned	147
9.8 eavRPM solution	147
Part 4. Post-conversion tasks	153
Chapter 10. z/OS and DB2 setup	155
10.1 DB2 system parameters	156
10.1.1 Thread management parameters	156
10.1.2 Locking parameters	157
10.1.3 Checkpoint control parameters	158
10.1.4 Distributed Data Facility parameters	159
10.1.5 DB2 log parameters	160
10.1.6 Resource management parameters	161
10.1.7 EDM pool parameters	161
10.2 Configuring the buffer pool	162
10.2.1 Buffer pool assignment	162
10.2.2 Buffer pool sizing	164
10.2.3 Buffer pool thresholds	164
10.2.4 Buffer pool tuning tools	165
10.2.5 Automating buffer pool management	168
10.2.6 Additional buffer pool considerations	168
10.3 CICS configuration	169
10.3.1 CICS system initialization parameters	169
10.3.2 DB2 threads in CICS	170
10.3.3 CICS DB2 resource definitions	170
10.3.4 Threadsafe programs	173
10.4 z/OS operating system	174
10.4.1 Supported z/OS release levels	174
10.4.2 Workload Manager	174
Chapter 11. Application and SQL tuning	175
11.1 Application tuning	176
11.1.1 Reducing number of SQL statements	176
11.1.2 Eliminating programmatic joins	176
11.1.3 Remove cursor in cursor processing	176
11.1.4 Filter in DB2, not in program	177
11.1.5 Selective column retrieval	178
11.1.6 Organize transactions to avoid contention and avoid deadlocks	178
11.1.7 Using optimistic locking to improve concurrency	179
11.1.8 Reducing sort cost	180
11.1.9 Ensuring appropriate use of external stored procedures	180
11.1.10 Using basic SQL procedures	181
11.1.11 Promoting the use of dynamic statement cache	181
11.1.12 Converting single row processing to multi-row fetch or insert	182
11.1.13 Combine statements by using SELECT FROM INSERT	183
11.1.14 Combining statements with SELECT FROM UPDATE or DELETE	183
11.1.15 Using the SQL scalar fullselect	184
11.1.16 Using MERGE for synchronization	185
11.1.17 Combining statements with SELECT FROM MERGE	186
11.1.18 Close cursors	186
11.2 SQL tuning and optimization	187

11.2.1	Improving sequential access	187
11.2.2	Improving index usage	187
11.2.3	Ensuring appropriate predicate filtering	187
11.2.4	Using common table expressions	188
11.2.5	Using recursive SQL for retrieving ordered sets	189
11.2.6	Using uncommitted read	192
11.2.7	Monitoring and maintaining applicable organization and statistics	193
11.2.8	Using EXPLAIN to verify access path	195
11.3	IBM tools for application and SQL tuning	195
11.3.1	InfoSphere Optim Query Workload Tuner for DB2 for z/OS	195
11.3.2	IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS	196
11.3.3	IBM DB2 Path Checker for z/OS	196
11.3.4	IBM DB2 SQL Performance Analyzer for z/OS	196
11.3.5	IBM DB2 Query Monitor for z/OS	197
Chapter 12.	Maximizing DB2 for z/OS features	199
12.1	Using DB2 for z/OS features	201
12.2	Using identity columns and sequence objects	201
12.2.1	Identity columns	201
12.2.2	Sequence objects	201
12.3	Referential integrity	202
12.4	Triggers	204
12.5	Temporal data	205
12.6	In-memory tables	206
12.7	Table check constraints	206
12.8	DSNULI	207
12.9	RELEASE(DEALLOCATE)	208
12.10	INCLUDE columns	208
12.11	Scrollable cursors	208
12.12	DB2 built-in functions	209
12.13	Non-padded indexes	210
12.14	Clustering index	210
12.14.1	Natural keys versus surrogate keys	211
12.15	Materialized query tables	211
12.16	Compression	213
12.17	Append processing	213
12.17.1	Member Cluster	213
12.17.2	APPEND option of CREATE and ALTER table	214
12.18	Temporary tables	214
12.18.1	Created temporary tables	214
12.18.2	Declared temporary tables	215
12.19	Clone tables	217
12.20	Large data support	217
12.20.1	Partitioned table spaces	217
12.21	Storing LOB data	218
12.22	Storing XML data	218
Part 5.	Appendixes	219
Appendix A.	IBM Information Management Services Migration Practice	221
	Strategy for a successful migration	222
	Customer partnering is a key element to success	222
	Migration strategy	224
	Phase zero	226

Calibration, design, and planning phase	227
Conversion phase	227
Testing phase	228
Implementation phase	229
IBM Information Management Software Services	229
Appendix B. Additional material	231
Locating the web material	231
Using the web material	231
System requirements for downloading the web material	232
Downloading and extracting the Web material	232
Related publications	233
IBM Redbooks	233
Other publications	233
Online resources	233
Help from IBM	234
Glossary	235
Index	237

Figures

3-1 Methodology comparisons	27
4-1 The four stages of conversion.	40
4-2 The 10 steps of the assessment.	41
5-1 Options risks analysis	54
6-1 Conversion strategies	59
7-1 Time of each project phase.	74
7-2 Overview of coexistence solutions by using IBM InfoSphere® solutions	79
7-3 Dat replication architecture	80
8-1 Overview of ATERAS solution with DB-Shuttle.	91
8-2 Assessment process.	94
8-3 Global definitions	99
8-4 Conversion to BMS map.	100
8-5 Database conversion process steps	103
8-6 BACHMAN diagram	104
8-7 Converted data model.	105
8-8 Application conversion process	106
8-9 IDMS COBOL to COBOL DB2	107
8-10 Dialog conversion	108
9-1 ConsistADS solution architecture	119
9-2 ConsistADS-IDE environment view	120
9-3 The ConsistADS toolbar	120
9-4 ConsistADS-IDE Add Library	120
9-5 ConsistADS IDE New Source	121
9-6 ConsistADS IDE SAVE	121
9-7 ConsistADS IDE SAVE AS	121
9-8 ConsistADS IDE: COMPILE	121
9-9 ConsistADS IDE RUN	121
9-10 Adabas file design.	124
9-11 Normalized model	125
9-12 The denormalized model	125
9-13 Fully denormalized model	126
9-14 Architecture overview	129
9-15 DMS tasks.	132
9-16 List of imported files	133
9-17 Details of the file attributes	133
9-18 DDM imported.	134
9-19 DDM and corresponding data model	134
9-20 Logical file	135
9-21 Generating DDL	135
9-22 DB2 for z/OS DDL	136
9-23 Generating mapping	136
9-24 Generating JCL to conversion	136
9-25 Creating unload process	137
9-26 Generated JCL	138
9-27 Phases of OnTarget	144
9-28 eavRPM solution overview	148
9-29 New Java interface	151
9-30 Maps flow	152

Tables

5-1	Project planning estimate content	52
6-1	Pros and cons of the migration solutions	63
7-1	Advantages of automated conversion over manual	75
7-2	Considerations to qualify vendor tools	77
8-1	CA IDMS languages versus COBOL functions	88
8-2	CA IDMS DML statements versus SQL statements	88
8-3	Assessed programs versus converted programs	95
8-4	Converted components of the customer	96
8-5	Sample of activities duration	98
9-1	Conversion versus transparency	113
9-2	Natural DML compared to DB2 SQL	127
9-3	Natural applications	128
9-4	Adabas files	128
9-5	Project timing	129
9-6	Results	141
10-1	Default buffer pool system parameters	163
10-2	Sample buffer pool assignment	163
A-1	Summary of activities and results by phase	224

Examples

9-1 Adabas fields names	114
9-2 GET ISN conversion	126
9-3 HISTOGRAM conversion	126
9-4 Natural source program	138
9-5 Program converted to dynamic Advance language	139
9-6 Program converted to Static Assembler	140
9-7 Mapping for the source programs	141
10-1 DB2 checkpoint data in an OMEGAMON PE Statistics Report	158
10-2 Log Activity section of OMEGAMON PE Statistics Report	160
10-3 Buffer Pool Read Operations Section of an OMEGAMON PE Statistics Report	166
10-4 Sample automation of buffer pool configuration changes	168
11-1 DB2 join	176
11-2 A cursor within a cursor	177
11-3 Example of SELECT that uses optimistic locking	179
11-4 Example UPDATE using optimistic locking	179
11-5 Example of a multi-row FETCH cursor	182
11-6 Example of a multi-row INSERT statement	182
11-7 Sample INSERT within SELECT statement	183
11-8 Sample SELECT FROM UPDATE statement	184
11-9 Sample SELECT FROM DELETE statement	184
11-10 Scalar fullselect in a WHERE clause	184
11-11 Scalar fullselect in SELECT list	185
11-12 Sample MERGE statement	186
11-13 CLOSE cursor syntax	186
11-14 Sample of filtering in program code	188
11-15 Sample of filtering in SQL statement	188
11-16 Sample common table expression	189
11-17 Sample ordered set	189
11-18 Sample converted ordered set with ORDER BY	190
11-19 Sample converted ordered set with ORDER BY and next and prior pointers	190
11-20 Sample converted ordered set with ORDER BY on prior pointer	191
11-21 Recursive SQL to retrieve data in appropriate order for an ordered set	191
11-22 Results of recursive SQL that show the ordered set in correct order	192
11-23 Sample use of uncommitted read in SELECT statement	192
12-1 Sample CREATE SEQUENCE statement	202
12-2 Example of referential integrity	203
12-3 Definition of temporal structure by using business time	205
12-4 Definition of temporal structure by using system time	205
12-5 Example of check constraint	207
12-6 Use of built-in function MAX	209
12-7 Use of CLUSTER option on CREATE INDEX	210
12-8 Query to return summarized data when no MQT exists	212
12-9 DDL to create a materialized query table	212
12-10 Sample rewritten query to access materialized query table	212
12-11 Join of base table and temporary table	215
12-12 DDL to create a global temporary table	215
12-13 DECLARE of global temporary table and INSERT to load table	216
12-14 DDL for a table with LOB columns	218

12-15 DDL to create a table with an XML column.	218
---	-----

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 or changes in the products or the programs 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®	Guardium®	Redbooks (logo)  ®
BladeCenter®	IBM®	System Storage®
CICS®	IMS™	System x®
DataStage®	InfoSphere®	System z®
DB2 Connect™	MVS™	Tivoli®
DB2®	OMEGAMON®	WebSphere®
Distributed Relational Database Architecture™	Optim™	z/Architecture®
DRDA®	POWER7®	z/OS®
DS8000®	pureXML®	z/VM®
FICON®	QMF™	z/VSE®
FlashCopy®	RACF®	z10™
	Redbooks®	zEnterprise®

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.

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

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

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.

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

Preface

Time to market, flexibility, and cost reduction are among the top concerns common to all IT executives. If significant resource investments are placed in mature systems, IT organizations need to balance old and new technology. Older technology, such as non-IBM pre-relational databases, is costly, inflexible, and non-standard. Users store their information on the mainframe and thus preserve the skills and qualities of service their business needs. But users also benefit from standards-based modernization by migrating to IBM® DB2® for z/OS®. With this migration, users deliver new application features quickly and respond to changing business requirements more effectively.

When migrating, the main decision is choosing between conversion and re-engineering. Although the rewards associated with rebuilding mature applications are high, so are the risks and customers that are embarking on a migration need that migration done quickly.

In this IBM Redbooks® publication, we examine how to best approach the migration process by evaluating the environment, assessing the application as a conversion candidate, and identifying suitable tools.

This publication is intended for IT decision makers and database administrators who are considering migrating their information to a modern database management system.

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.

Paolo Bruni is a DB2 Information Management Project Leader at the International Technical Support Organization based in the Silicon Valley Lab since 1998. He has authored several IBM Redbooks publications about DB2 for z/OS and related tools and has conducted workshops and seminars worldwide. During his many years with IBM, in development and in the field, Paolo has worked mostly on database systems.

Marcos Caurim is a Technical Sales and DB2 z/OS specialist in Software Group at IBM Brazil, São Paulo. He has 14 years of experience in IT. During this time, he worked on several projects with Brazilian and foreign customers (USA, Canada, German) and 12 of these years he worked with DB2 for z/OS and pre-relational databases. He holds a degree in Science of Computing from the Universidade Paulista and an MBA from Fundacao Getulio Vargas. He is a certified IBM DBA and System Administrator for DB2 z/OS Version 10. Marcos co-authored *Powering SOA with IBM Data Servers*, SG24-7259-00.

Elise Gotay is an IBM Certified DB2 Migration Specialist working in the United States with over 25 years of database, application, and network experience on the distributed and z/OS platforms. She has experience as a DBA using DB2, CA Datacom, and IMS™, and has expertise migrating to DB2 from the Oracle, CA Datacom, CA IDMS, and Software AG Adabas platforms. She has an MS in Information Systems (MSIS) from George Mason University and writes extensively on database technology.

Susan Lawson is an IBM Gold Consultant and IBM Champion with YL&A who has worked with DB2 since 1988. Over the past 14 years, she has worked on several large projects that converted mature data into DB2 for z/OS. She authored and co-authored several books on DB2, including the IBM DB2 10 for z/OS DBA Certification Study Guide. She specializes in DB2 performance consulting and education worldwide.

Glenn McGeoch is a Senior DB2 Consultant for the IBM DB2 for z/OS Lab Services organization based San Francisco, CA, US. He has 34 years of experience in the software industry, with 26 years of experience in working with DB2 for z/OS. He holds a degree in Business Administration from the University of Massachusetts and an MBA from Rensselaer Polytechnic Institute. Glenn worked for 19 years as an IBM customer with a focus on IBM CICS® and DB2 application development, and spent the last 16 years with IBM assisting DB2 customers. His areas of expertise include application design and performance, stored procedures, and DB2 migration planning. He presented to regional DB2 User Groups and to customers on various DB2 topics. Glenn co-authored *DB2 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7083; *DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7604, and *DB2 10 for z/OS Performance Topics*, SG24-7942.

Rebecca Poole is the IBM DB2 for z/OS Business Development Executive. She works across IBM and Business Partners to identify new opportunities and enhance existing strategies. In her IBM career, she worked in business and technical leadership positions that span core systems modernization, composite application management, integrated application development, web portal technologies, enterprise application software s.a. SAP, object-oriented modeling, and relational databases. Her first position at IBM was as a DB2 DBA and Systems Programmer. Rebecca was named as an IBM Master Inventor and has served as the IBM Software Group recruiting manager for Stanford University since 2001.

Special thanks to the following people for providing information on the conversion tools:

- ▶ Steve Bryant
- ▶ Cindy Howard
ATERAS
- ▶ Clauder Balzano
Consist Software Solutions
- ▶ Luis Pereira
IBM Software Group
- ▶ Max Moss
MOST Technologies

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

- ▶ Barry Faust
IBM Sales & Distribution, Somers, NY
- ▶ Carroll Hummer
IBM Software Group, Herndon, VA

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:

<http://www.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:

<http://www.ibm.com/redbooks>

- Send your comments in an email to:

redbooks@us.ibm.com

- Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. 1WLB 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>



Part 1

Introduction

This book describes the business value propositions and feasibility of pre-relational conversions and consolidations to DB2 for z/OS and IBM System z®. The following topics are included in this book:

- ▶ Industry dynamics driving pre-relational conversions
- ▶ Conversions and consolidations onto DB2 for z/OS and System z is a reality today
- ▶ Why DB2 for z/OS and System z is the ideal conversion target
- ▶ How tools provide further business and technological values
- ▶ End-to-end methodologies and best practices
- ▶ Some representative automated conversion solutions
- ▶ Post automated conversions and best practices
- ▶ Additional business benefits to be gained through adoption of more modern technologies and solutions
- ▶ Conversion consultancy and services

This part includes the following chapters:

- ▶ Chapter 1, “Business considerations” on page 3
- ▶ Chapter 2, “Converting to DB2 for z/OS” on page 7



Business considerations

Migrating to a relational database management system (DBMS) is a trend that is gaining momentum among clients of some pre-relational DBMSs. Popular since the 1970s, industry analysts advocated these pre-relational DBMSs to their clients to migrate off to a modern platform. A number of clients made this migration in response to increasing costs, diminishing skills, time to market, and other business challenges.

The remaining clients must determine whether migration is appropriate for them. This decision is especially important if the pre-relational DBMS provider of the client did not make the investments to provide new capabilities, skills, and vitality initiatives to help the client meet the needs of their business.

IBM DB2 for z/OS and System z is an ideal and logical target for pre-relational migrations. New and existing clients rely on IBM DB2 for z/OS and System z for its unsurpassed total system reliability, availability, serviceability, and security. Demonstrated benefits include exceptional availability and scalability, a superior level of protection for business critical data and applications, ease of integration and management across platforms, and reduced infrastructure complexity.

IBM continues to actively invest to enhance the DB2 for z/OS platform to improve performance, reduce resource consumption, and introduce new features, such as bitemporal support. DB2 for z/OS clients have a vibrant platform that positions them to readily adopt more solutions to address their current and future business needs.

This chapter describes the value proposition for pre-relational to DB2 for z/OS and System z conversions in the following topics:

- ▶ Historically speaking
- ▶ Industry dynamics driving change
- ▶ The reality of software conversions

1.1 Historically speaking

In the 1970s and 1980s, before the relational database management systems (RDMS) was established, clients opted to use pre-relational database management systems such as CA Technologies IDMS/DB, CA Technologies Datacom/DB, Software AG Adabas, IBM IMS DB, and Computer Corporation of America Model 204.

Subsequent database and DBMS evolutions led to the establishment of the relational database as the mainstream DBMS in the marketplace. Although pre-relational DBMSs are no longer the pre-eminent option for database systems, some pre-relational DBMS providers continued to innovate and modernize their database with new capabilities to meet the evolving needs of their clients. Providers that failed to modernize their infrastructure placed their clients in a difficult position because the clients cannot keep pace with dynamically changing business climates.

Industry analysts are advising companies that use pre-relational DBMS technologies to migrate to a modern, standards-based, and flexible environment. Clients are migrating to a modern environment and many more clients must migrate in the near future to achieve business agility.

1.2 Industry dynamics driving change

Business agility is essential to client success in the fast-paced, competitive, and highly regulated global business climate of today. To grow and thrive, clients must effectively adopt and apply other modern techniques (including analytics) to gain and sustain a competitive edge.

An inflexible technology that is not based on standards hinders the ability of the client to meet their business needs. These needs include rapid time-to-market for new offerings and timely compliance with required regulations that are essential to stay in business. But the ability of a client to meet their business needs also presents the following challenges:

- ▶ Diminishing skill base
- ▶ Increasing application maintenance costs because of complex data structures
- ▶ Growing data integration costs and data integrity issues

These challenges, combined with efforts to reduce internal costs, simplify operational infrastructure, and increase productivity, are compelling more clients to move away from their pre-relational investments to a modern, standards-based, and flexible platform.

The industry did not ignore the challenges of these clients. A number of enterprising companies emerged that offer automated pre-relational to relational transformation solutions to many mainstream relational DBMSs. Some of these companies work with other companies that offer the mainstream relational DBMSs that are used today.

1.3 The reality of software conversions

In the past, the amount of manual rewrite work needed to migrate a pre-relational database and application far exceeded the work that was done by automated tools. However, with experience, innovations, and time, this dynamic changed. The advent of highly automated conversion solutions and methodologies made pre-relational conversions viable and repeatable. This change reduced the amount of time needed, the risks involved, and the costs of the migration process.

Clients use automated conversion solutions to successfully migrate to modern platforms, such as DB2 for z/OS and System z. A number of significant pre-relational database-to-DB2 for z/OS migrations were completed and many more are in progress. This trend is expected to continue as maturing conversion technologies become more popular and are widely accepted because of successful migrations.

Additionally, the converted database and applications are more flexible and based on ubiquitous transferable skills. Clients readily adopt other business agility and productivity solutions that enable them to grow and thrive in the fast-moving and competitive global business climates of today.

The availability of skilled individuals who administer a pre-relational system also is a key issue often raised by clients. As employees who manage and maintain these pre-relational systems retire, it is challenging to find skilled individuals to take their place. IBM proactively addressed the need for skilled System z individuals through the establishment and support of and active leadership in the following skills vitality initiatives:

- ▶ IBM Academic Initiative helps schools to provide, develop, and nurture System z hardware and software skills as part of their curriculum. As of this writing, over 800 schools around the world participate in this program. This initiative is customized to meet the unique requirements and capabilities of each school. For more information about this program, see this website:

<http://ibm.com/university/academicinitiative>

- ▶ System z Job Board is the follow-on program to the IBM Academic Initiative. This program connects IBM clients, partners, and businesses with students and professionals who seek System z job opportunities. These connections further reinforce the commitment of IBM to ensure the vitality and resource pool of highly skilled System z users. For more information about this program, see this website:

<http://www.systemzjobs.com>

IBM also continues to invest in and grow System z hardware and software business offerings, including the regular release of the following innovate solutions:

- ▶ IBM zEnterprise®: Extends the strengths and capabilities of System z to other systems and workloads to improve data center management
- ▶ DB2 10 for z/OS: Delivers client-focused business benefits that include an industry-first standard bitemporal support
- ▶ IBM DB2 Analytics Accelerator: Delivers unparalleled extreme performance for complex analytics

These solutions are powerful on their own. When they are combined, these solutions deliver record-setting results, such as those achieved in the September 2011 SAP on IBM zEnterprise z196 and DB2 10 for z/OS benchmark for a banking system. In this case, 150 million banking accounts on day and night processing scenarios were completed. For more information, see this website:

http://www.ibm.com/solutions/sap/us/en/news/zenterprise_system_scales_best.html

Through continued innovations and investments in System z, IBM positioned DB2 for z/OS and System z as the ideal platform target for pre-relational consolidations and conversions. This commitment by IBM also ensures a steady availability of skilled professionals to support System z hardware and software.



Converting to DB2 for z/OS

In this chapter, we describe the strengths of DB2 for z/OS in the market and its synergy with System z.

This chapter includes the following topics:

- ▶ Converting to a relational DBMS
- ▶ Additional benefits of a relational DBMS
- ▶ Benefits of SQL
- ▶ Data definition language
- ▶ DB2 for z/OS is the platform of choice
- ▶ zEnterprise
- ▶ Additional benefits of DB2 10 for z/OS

2.1 Converting to a relational DBMS

Relational databases are the superior data stores choice for over 25 years. Relational technology is the method of choice for storing large amounts of data with flexibility and easy access to data with a common language, Structured Query Language (SQL). Data stored in tables is accessible and is manipulated by SQL. This data accessibility, when combined with a language suitable for users, results in reduced maintenance and greater productivity.

SQL and tables give structural independence to data. That is, the data is retrieved in any order of key and column and is not dependent on the way the data is stored on a disk. SQL is the International Standards Organization (ISO) and American National Standards Institute (ANSI) standard language for access to a DBMS. Worldwide acceptance of this standard means that applications developed on one platform are ported to run on another platform. Applications issuing SQL on one platform use the same SQL to manipulate relational data on another platform.

The productivity of an application programmer increased when many programming functions were incorporated into the relational database management system (RDBMS). In the past, these functions were created and maintained separately by the applications programmer. For example, with referential integrity, the programmer does not create code in the application to check the validity and integrity of data. Features inherent in the database, such as triggers, stored procedures, and row set processing reduce the level of programming required. SQL provides a powerful language that also reduces programming.

RDBMSs also include productivity features for the database administrator, thus providing the programmer with the ability to respond faster to requests from users. For example, if changes to data records (tables) are needed, the database administrator (DBA) changes the table with a simple ALTER TABLE statement. Utilities provide fast and efficient processing for data loading and unloading, data reorganization and many other tasks commonly associated with the maintenance of a database environment.

For IT organizations, a relational database management system improves the productivity and efficiency of the user and programmer staff. With query tools, the users access data and create their own reports, rather than wait for the programming staff to respond to user requests. The database administrator makes the changes without affecting the program source code. As a result of these efficiency and productivity improvements, the backlog of application requests is reduced.

An RDBMS provides efficient access to data. Because the success of an enterprise depends on timely access to its data, RDBMS performance becomes a paramount issue.

2.2 Additional benefits of a relational DBMS

The tabular relational model is a simple concept and easily understood by the user, the programmer, and the DBA. If tables are used to store normalized data, powerful data access requests are formulated. The tabular concept reduces the training cost and application maintenance workload for the personnel who are working with the DBMS. The tabular concept also means that the information systems staff communicate readily with application users about database and data issues, such as content and relationships, because the data is easily understood. This capability extends to direct user support. RDBMS is taught in most universities and is understood by many new DBAs and programmers. Older, uncommon files systems and the languages to use them are difficult to support.

Unlike older DBMSs and file systems, the data relationships are not predefined. A requirement to obtain data from adjacent columns together is unnecessary. Only the required rows and columns are selected, and they are obtained in any order. Tables are not linked by pointers, only by data values. This feature allows tables to be linked as required.

Normalization provides a mathematically proven technique for preventing data redundancy. Preventing this redundancy reduces the requirements for data maintenance, ensures data consistency, and eases application extension. Although normalization is part of the relational theory, design features such as repeating groups, periodic fields, and redefines are not part of this design.

Data in an RDBMS is structurally independent. As a result, an RDBMS features the following benefits:

- ▶ New columns and data are added to a table without affecting existing data or programs. Columns and data also are added as the system runs normally.
- ▶ New tables are added to the DBMS through online commands without affecting daily operations or existing programs.
- ▶ New relationships between tables are created dynamically. Tables are linked by the data values in the columns alone.
- ▶ New ways in which to examine the data are set up by using views. These views include a relationship between tables. Through this view, the joined tables are shown as a single table to the user.

When a combination of data access requests is formulated by the user or the programmer and run by the DBMS, the resulting data structure is a table. This structure is important because no matter what the SQL request is, the results are used with subsequent SQL requests that have full capabilities to conduct simple or complex requests. This principle allows a user to construct views on base tables and views on views.

The principle of expressing relationships between various tables by using value comparisons rather than links or pointers allows the expression of any reasonable relationship between the database elements. This expression is true even if the relationship is not explicitly declared in the database definitions. This expression allows for the construction of access paths to major areas of data, and for the use of views to provide more restricted access to parts of the data. This structure is important for user access to data. The structure also has substantial benefits for the programmers because of flexibility, ease of expression, and the independence of the underlying physical implementation of the database. Additional data access is provided with less administration and less maintenance.

All of the characteristics described thus far are responsible for the success of relational technology. These characteristics also fostered the spread of the technology in many supporting products, across all platforms. The popularity of relational DBMSs and relational technology is evident in the fact that they are the automatic choice for most new developments. This popularity has led to the quick development of other DBMS products and widespread attention to related product domains by tool vendors. These tools include data access tools, application development tools, and system and database administration tools.

2.3 Benefits of SQL

SQL is a uniform language that features the following statements:

- ▶ Data manipulation (SELECT, INSERT, UPDATE, and DELETE)
- ▶ Data definition (CREATE, ALTER, and DROP)
- ▶ Data control (SECURITY - GRANT and REVOKE)

As a result, the application development, database administration, and information center functions share a common set of skills in developing and supporting all types of database usage.

The SQL language also features the following capabilities:

- ▶ Built-in functions for arithmetic (for example, SUM, AVERAGE, COUNT), and scalar functions that operate on single operands and date and time manipulation.
- ▶ Built-in support for NULL data values.
- ▶ The construct for simplifying, subsetting, and securing data structures
- ▶ The ability to join data on multiple tables in a single statement. Data is correlated among the tables, based on any data values in the tables for either a static or dynamic request.
- ▶ The ability to treat multiple tables of similar format as a single large table to support a single SQL request (by using a UNION function).
- ▶ The ability to work with whole groups of data with set operations to add, change, or delete a selected set of rows with a single statement.
- ▶ The ability to sequence selected sets of rows and columns as the application requires, without regard to the stored sequence of the data.

The full functionality of SQL for both programmer and user enhances the capability of the language to expand applications and reduce the cost of maintaining those applications. Programmer productivity also is greatly enhanced.

SQL is descriptive in nature, not navigational. The user or the programmer expresses what must be done, not how it is done. The role of the DBMS is to determine the best way to handle the data access request.

The SQL language is based on first-order predicate logic. A data access request specifies the operation performed (SELECT, INSERT, UPDATE, DELETE), the data items and tables involved in the request, and then the predicates that describe the conditions that must be satisfied.

SQL supports the selection of data items across many tables. The selection of rows and various specifications allows the construction of new (virtual) tables based on existing tables (JOIN, UNION, INTERSECT, DIFFERENCE). SQL also supports set operations in which many items of data are changed with a single SQL call.

2.4 Data definition language

In addition to supporting data access requests, the SQL language provides data definition capabilities with which definitions of tables, data items, and all other relational DBMS objects are handled. SQL also provides extensive support for granting and revoking authorization to specific users or application programs for performing privileged operations on parts of the database. The data definition language (DDL) is almost completely online. From small changes, such as adding a column, to larger changes, such as preparing a new application in the DBMS, modifications are implemented without disrupting data access.

2.5 DB2 for z/OS is the platform of choice

DB2 is the relational database of IBM and is used for applications that require high performance because the product architecture is designed to deliver performance. The components of DB2 that affect performance, such as the optimizer and buffer manager, are developed and enhanced to provide efficient and fast data access. Technology that facilitates performance, such as parallel processing, is employed throughout the product architecture.

DB2 is integrated into the System z operating system (z/OS) software and hardware. This integration enables DB2 to use fully the performance features of the underlying hardware and software, thus delivering a level of performance that is unmatched by other mainframe-based RDBMS products. Because mainframe-based RDBMS products must support work that originates from transaction processing (TP) monitors and distributed systems (such as client server), the RDBMS must provide well-defined, efficient interfaces to those systems to facilitate application performance. DB2 provides these interfaces that support both mainframe TP monitors, such as Customer Information Control System (CICS) with the DB2 Adapter from IBM, and client server systems, with support for stored procedures. These interfaces are well-designed and efficient and distinguish DB2 as the enterprise server.

DB2 for z/OS offers the following benefits:

- ▶ Is technically advanced and is a key component of the IBM strategy for the development and support of large systems through the full use of hardware and software
- ▶ Plays a key role in distributing data and providing a secure warehouse architecture for customer data, depending on their requirements
- ▶ Is the leading enterprise data server, designed and tightly integrated with IBM System z mainframe to use the strengths of IBM System z
- ▶ Reduces total cost of ownership through process enhancements and productivity improvements for database administrators and application developers
- ▶ Delivers unparalleled security and streamlined compliance, and enables high-volume transaction processing for new applications
- ▶ Provides the core infrastructure for the next wave of web business applications and SOA initiatives
- ▶ Manages operational applications that feature heavy transaction loads and critical response time requirements
- ▶ Is equally strong in function and capacity for mass activity against tables that hold billions of data rows

DB2 uses fully multi-engine processors and achieves parallelism through the use of multitasking. DB2 also runs in multiple address spaces, with CICS, Information Management System/Transaction Manager (IMS/TM), Time Sharing Option (TSO), and batch applications in their separate address spaces that run in cross-memory mode with DB2. DB2 processing occurs in parallel under these other address spaces.

DB2 processing is not limited by the power of a single processor engine and its performance is highly scalable across multiprocessor machines. DB2 is an authorized subsystem of z/OS and uses cross memory services for efficient communication between address spaces.

DB2 also uses fully large virtual storage when backed by corresponding quantities of real storage. In particular, the DB2 database manager is designed to work efficiently with large buffer pools. This design enables data to become resident in electronic storage to minimize physical I/O to disk. Throughput and response times are improved and processor utilization is reduced.

With every release of DB2, there are more opportunities to take advantage of System z. Version 10 DB2 further uses the latest improvements in the platform. DB2 10 increases the synergy with System z hardware and software to provide better performance, more resilience, and better function for an overall improved value.

DB2 benefits from large real-memory support, faster processors, and better hardware compression. DB2 uses the enhanced features of the storage servers, such as the IBM System Storage® DS8000®. FlashCopy® is used by DB2 utilities, allowing higher levels of availability and ease of operations. DB2 makes unique use of the IBM z/Architecture® instruction set for improvements in reliability, performance, and availability. DB2 continues to deliver synergy with hardware data compression, IBM Fibre Channel connection (FICON®), channels, disk storage, advanced networking function, and Workload Manager (WLM).

2.5.1 Scalability, availability, and usability

DB2 for z/OS remains a leader in continuous availability with unique improvements, such as online schema evolution. This evolution includes the ability to add partitions and change data definitions with ALTER statements, online unload and replace, and improved backup and recovery utilities.

DB2 also provides extensive partitioning, clustering, and index options. Along with important optimization improvements, such as improved INSERT, UPDATE, and DELETE processing, and buffer scalability enhancements, DB2 for z/OS is the right choice for large and growing databases.

Faster response and reduced processing times come from improvements to optimization and better information for the optimizer, which benefits a host of query types. Combined with new database design, options for indexes, clustering, materialized query tables, and the ability to rotate partitions provides support for complex support warehouses.

2.5.2 Security

Legendary mainframe security and enhanced data encryption means that DB2 for z/OS is ready to protect your data and comply with growing compliance and audit requirements.

Through the GRANT and REVOKE constructs in the SQL language, and in cooperation with the IBM Resource Access Control Facility (RACF®) external security manager (or others), DB2 provides a comprehensive and selective mechanism for controlling access to DB2 data. RACF passes individual identification to DB2 or a group identification. Access is granted to individual programs, users, or groups of users. Access authority is specified to the field value level, if required, and distinguished between the retrieval and update functions. Specific authorities are delegated as restrictive or free according to the policy of the installation.

For productivity, authority is granted to functional IDs and individual accountability is provided by recording activity at the user ID level. DB2 provides an audit trail for recording the scope of all activity against any installation-specified subset of the data tables.

2.5.3 Database integrity

DB2 for z/OS includes a comprehensive set of functions to ensure the integrity of the database by automatically backing out incomplete or canceled transactions. These functions also rapidly recover the database when physical damage occurs. The information required to initiate and control these functions is captured and maintained by DB2, thus avoiding human error in selecting the correct log volumes or image copy version required for each recovery. DB2 includes an automatic deadlock detection and timeout facility which is used to detect and automatically back out part of a deadlock so that other tasks continue. This facility prevents indefinite waits for unavailable resources.

In the CICS or IMS/TM environments, the two-phase commit protocol provides the coordination of recovery actions between the transaction processing monitor subsystems (CICS and IMS/TM) and the DB2, DL/I, and virtual storage access method (VSAM) database subsystems. (VSAM is available only for CICS.) This protocol enables a single application transaction to update (for example, both DB2 and VSAM under CICS within the same unit of work), with overall integrity ensured by the system. The use of the same two-phase commit protocol is fundamental in the DB2 distributed unit of work protocol. In this configuration, data that is distributed among multiple DB2 systems is updated with global data integrity that is assured by the system software rather than by the application.

The referential integrity functions within DB2, specified with the definitions of tables rather than in programs, ensures that groups of tables are consistent without relying on external standards and controls. The check constraints on columns are used to validate column values. Database triggers also support rules to govern the data within the database. These features minimize the database administration workload to maintain the integrity of the database.

2.5.4 Continuous operation support

DB2 is equipped with the following functions to ensure continuous operation:

- ▶ New or revised definitions of tables, indexes, and views take effect immediately on completion of the definition, without stopping DB2.
- ▶ Housekeeping utilities for backup, reorganization, and recovery are run on a table space while DB2 continues normal operation on other table spaces.
- ▶ A table space is partitioned by data value so that some utilities are run in parallel on a partition basis. This configuration minimizes the time that data is unavailable for processing.
- ▶ The use of data sharing in a parallel sysplex environment provides support during planned outages as a release is upgraded.

2.5.5 Centralized catalog

The DB2 catalog contains all of the object definitions and statistics about the data in the database. The DB2 catalog is a set of relational data tables that are queried through the SQL language. The fact that the catalog is interrogated with standard SQL means that the information contained in the catalog is used by many classes of authorized users.

The following categories are interrogated:

- ▶ Statistics are read by a database administrator to determine whether a table space must be reorganized.
- ▶ Data definitions are used to find the correct names for tables or to determine where certain data is kept.
- ▶ Application plans are examined by programmers to find which tables are used by a particular program.

The catalog tables contain the full, current set of definitions of SQL objects. Thus, the dictionary and the database management system are not separated. New or changed definitions in the DB2 catalog are immediately active for all purposes without the need for any external action.

DB2 interfaces and utility functions make it easy to employ other data dictionaries external to DB2. These dictionaries are used for purposes such as maintaining sets of definitions in past, current, and future versions of the definitions.

2.5.6 Recovery

DB2 is the best platform with which to entrust precious company data. DB2 for z/OS includes comprehensive backup and recovery facilities. The design of DB2 ensures that the database is simple to operate, control, and manage. Restarting DB2 and recovering a physically damaged database are each done by a single command. Log archiving is automatic, as is allocating the correct data sets for recovery.

The following examples demonstrate how built-in functions in DB2 enforce integrity and recoverability:

- ▶ Delayed update of a newly loaded table until a backup copy is taken
- ▶ Forced logging of all updates
- ▶ Dual logging option, used for most operational systems

- ▶ Automatic switch to the next log when the online log data set is full
- ▶ Automatic archiving of the full log data set
- ▶ Delayed overwriting of the log data set until a successful archive has occurred
- ▶ Ability to take up to four copies of a table space simultaneously. This ability allows two copies to be sent off site for contingency recovery.
- ▶ Ability to take up to four copies of archive log simultaneously
- ▶ If recovery is needed, a single command runs the whole process
- ▶ If presented with the wrong tape or data set, DB2 rejects the tape or data and does not copy the wrong data. DB2 knows which data sets are needed from the DB2 catalog.

2.5.7 Usability and productivity

DB2 developers find the relational database is easy to use, train on, and work with when writing applications for the database. Developers report that they are more productive working with the database because the system is easier to use than traditional systems.

Developers cite the following benefits of DB2:

- ▶ The powerful SQL language is easy to use.
- ▶ Data definitions are created online, directly into the DB2 catalog.
- ▶ Referential integrity rules are defined with the data rather than designed into application programs.
- ▶ The Interactive System Productivity Facility (ISPF)-based DB2 Interactive (DB2I) interface for developers uses conventional languages.
- ▶ Many tools are available that allow easy, graphical development, and access from the workstation environment.
- ▶ The need to browse around the DBMS is eliminated. Decision support and ad hoc query functions often cannot be done by users because they do not know how to browse the data structures. This inability to browse the data structures is especially true if those structures contain redefined fields and repeating groups.
- ▶ By using the single-image SQL interface, application programs and users access data that is stored at multiple locations. DB2 coordinates or participates in the coordination of integrity updates across the DB2 family of databases. This ability simplifies the design, building, and operation of distributed applications.
- ▶ The optimum access path to the physical data is selected by using the DB2 optimizer, an expert system within DB2.

The optimizer is started by using the following methods:

- *Dynamically*: The optimizer is started dynamically when SQL statements are presented to DB2 by user tools with ad hoc, user queries, and application developers who use the tools. This advantage is important because programmers test SQL statements before installing the statements in a program.
- *Statically*: The optimizer is started statically to compile SQL statements embedded in application programs. Static SQL prepares for repetitive execution with maximum efficiency. In static SQL, the fastest access path to the data is determined only once by the optimizer and then stored for reuse. This method avoids the need to select the access path for every transaction (the access path is determined for each transaction when dynamic SQL is used) or activate forms of data caching.

- ▶ DB2 data is accessible for queries and reporting by users. For example, tools such as IBM QMF™ provide the following functions:
 - The SQL language for technically able users and data processing professionals
 - Prompted query for novices and casual users
 - Procedures for running predefined queries and reports
 - Report formatting
 - Business graphics
 - Personal data storage
 - Data importing and exporting in exchange with other systems
 - Data downloading to personal computers

2.5.8 Transaction manager support

DB2 for z/OS supports concurrent access from the following program execution environments:

- ▶ z/OS batch regions
- ▶ Online TSO
- ▶ CICS/ESA
- ▶ IMS/OTMA
- ▶ IBM WebSphere® Application Server

All of these environments concurrently run SQL applications or tools. To provide this support, DB2 supports five types of attachments. An attachment represents a connection between the DB2 subsystem and the program execution environment. Each of the attachment types has the following specific characteristics:

- ▶ The IMS and CICS attachments support transactions that run in their individual environments. These environments support multiple users through the same or multiple transactions and the two-phase commit. These execution environments run thousands of transactions concurrently.
- ▶ The TSO attachment supports users and programs that use TSO. This attachment is a single-user attachment and provides one-phase commit support.
- ▶ The Call Attachment Facility (CAF) provides explicit attachment support and is used by data access tools and system-related tools. CAF is a single-user attachment and provides support for one-phase commit.
- ▶ The Resource Recovery Services Attachment Facility (RRSAF) is used for stored procedures that run in a WLM-established address space or as an alternative to the CAF. RRSAF provides support for z/OS RRS as the recovery coordinator and supports other capabilities not present in CAF.

The attachments to a particular DB2 are controlled with system-managed priorities so that the workload on a particular DB2 subsystem is balanced.

In addition to the access from within the z/OS system, distributed data access facilities implemented in DB2 for z/OS provide access to DB2 data from other relational systems attached in the network. The Distributed Data Facility (DDF) handles all IBM Distributed Relational Database Architecture™ (DRDA®) and non DRDA requests to DB2. DRDA is the for transparent access to distributed data and increased manageability of distributed data across heterogeneous DBMS systems.

2.5.9 Robust Utility Suite and unlimited tools

IBM offers the following set of utilities to manage your DB2 for z/OS data. A host of tools also is available to conduct activities such as database administration and performance monitoring and tuning:

- **DB2 for z/OS Utilities Suite**

IBM DB2 Utilities Suite for z/OS is a comprehensive set of tools for managing all DB2 data maintenance tasks. These tools offer features and functions that increase availability and performance of any DB2 application while integrating with other tools to address specific issues with DB2 performance, administration, and recovery. The utilities suite also helps you to minimize downtime associated with routine DB2 data maintenance and ensures data integrity. There are two types of utilities available: *online* and *stand-alone* utilities. The stand-alone utilities run as batch jobs by using IBM MVS™ Job Control Language (JCL) independently of DB2.

- **CHECK DATA, CHECK INDEX, and CHECK LOB**

These utilities play an important function in the overall integrity of data and indexes, including large object (LOB) data. The CHECK DATA utility checks table spaces for violations of referential and table check constraints, and reports information about detected violations. The CHECK INDEX online utility tests whether indexes are consistent with the data that they index. The utility also issues warning messages when an inconsistency is found. CHECK DATA checks for consistency between a base table space and the corresponding LOB or XML table spaces. COPY, COPYTOCOPY, and MERGECOPY

The COPY utility creates up to four image copies of any table space, table space partition, data set of a linear table space, index space, and index space partition. COPYTOCOPY creates image copies from an image copy that is taken by the COPY utility, including in-line copies produced by the REORG or LOAD utilities. The MERGECOPY utility merges image copies produced by the COPY utility or in-line Copies produced by the LOAD or REORG utilities.

- **LISTDEF and TEMPLATE**

You group database objects into reusable lists by using the LISTDEF utility. You then specify these lists in other utility control statements to indicate that the utility must process all of the items in the list. This level of specification increases user productivity and reduces the chance for manual errors. By using the TEMPLATE utility, you allocate data sets without the need for JCL DD statements as a LISTDEF list is processed. By using templates, you standardize data set names across the DB2 subsystem and identify the data set type when you use variables in the data set name.

- **LOAD and UNLOAD**

The LOAD utility loads data into one or more tables of a table space. This utility also loads records into the tables and builds or extends any indexes that are defined on them. If the table space already contains data, you choose whether you want to add the new data to the existing data or replace the existing data. The UNLOAD utility unloads data from one or more source objects to one or more VSAM sequential data sets in external formats. The source is DB2 table spaces or DB2 image copy data sets.

- **REORG INDEX and REORG TABLESPACE**

The REORG INDEX utility reorganizes an index space to improve access performance and reclaim fragmented space. The REORG TABLESPACE online utility reorganizes a table space to improve access performance and reclaim fragmented space. The REORG TABLESPACE utility also reorganizes a single partition or range of partitions of a partitioned table space. You specify the degree of access to your data during reorganization and collect in-line statistics by using these utilities.

- **RUNSTATS and MODIFY STATISTICS**

The RUNSTATS utility gathers summary information about the characteristics of data in table spaces, indexes, and partitions. DB2 records this information in the DB2 catalog and uses the information to select access paths to data during the bind process. This information is invaluable as you determine when table spaces or indexes must be reorganized. The MODIFY STATISTICS utility deletes unwanted statistic history records from the corresponding catalog tables.

- **RECOVER and MODIFY RECOVERY**

The RECOVER utility recovers data to the current state, or to its previous state. This recovery is done by restoring a copy of the data then applying log records. The MODIFY utility with the RECOVERY option deletes records from the SYSIBM.SYSCOPY catalog table, related log records from the SYSIBM.SYSLGRNX directory table, and entries from the database descriptor (DBD).

DB2 for z/OS tools

Throughout this publication, you see that some tools from the IBM Optim™ set of solutions are used for different functions in DB2 for z/OS to help in areas from administration to performance. You also see some of these tools referenced again as we demonstrate where the tools are beneficial before, during, and after your migration to DB2 for z/OS.

The following DB2 for z/OS tools are designed to help you reduce operating costs, simplify your environment, and provide greater day-one value to fully maximize the benefits of every new release of DB2 for z/OS:

- **DB2 Administration Tool for z/OS**

This tool helps simplify a full range of database management functions so administrators at any skill and experience level conduct key tasks. For example, administrators quickly browse the DB2 catalog and build and run dynamic SQL statements without knowing the exact SQL syntax.

- **DB2 Object Comparison Tool for z/OS**

This tool helps accelerate the process of changing DB2 objects throughout the application lifecycle. The tool compares objects from different sources, automatically produces reports that describe those differences, and generates the information that is needed to apply changes. This tool also helps synchronize files by staging and propagating changes between environments.

- **DB2 High Performance Unload for z/OS**

This tool provides fast and efficient ways to unload and extract data for movement across enterprise systems or for in-place reorganizations. Unloading data during an object rebuild phase often slows the change process and negatively affects DB2 application performance. Integrating this tool into the change process helps reduce the maintenance time for critical DB2 applications and therefore increase database availability.

► DB2 Cloning Tool for z/OS

This tool accelerates the cloning process, and you quickly create DB2 table and index spaces or entire DB2 subsystems for testing, development, or maintenance tasks without affecting availability. By using this tool, you focus on other maintenance tasks and allow users to query the database on the clone when the online system is down.

► IBM Tivoli® OMEGAMON® XE for DB2 Performance Expert on z/OS

This tool comprehensive assessment tool helps you to monitor, analyze, and tune the performance of DB2 and DB2 applications on z/OS. The tool combines real-time monitoring, historical tracking, and batch reporting with buffer pool analysis and expert database analysis to help you identify potential problems and maximize database performance. The tool also supports experienced and novice DB2 for z/OS performance analysts, manages performance reporting scheduling, and populates a performance warehouse for analyzing performance data over time.

► DB2 Query Monitor for z/OS

This tool helps you to efficiently customize and tune SQL workloads and DB2 objects to improve the effectiveness of DB2 subsystems and enhance overall performance. DB2 Query Monitor provides access to current and historical views of query activity throughout DB2 subsystems from a single console, so you identify problems quickly and respond immediately.

► DB2 Recovery Expert for z/OS

This tool is a self-managing recovery solution that helps you recover DB2 data with minimal disruption. By automating the process of rebuilding assets to a specified point in time, the tool helps simplify key tasks without interrupting database and business operations. The tool also delivers a comprehensive analysis of altered, incorrect, or missing database objects for superior manageability.

► DB2 Change Accumulation Tool for z/OS

This tool helps accelerate the recovery of key business functions in the event of a disaster or large-scale outage. The tool actively collects and stores backup information about important objects and keeps current and consistent data images readily available.

► DB2 Object Restore for z/OS

This tool automatically restores dropped objects such as databases, table spaces, tables, indexes, data, and table authorizations. Although recovering entire databases is crucial for business continuity, the ability to rapidly restore single objects is critical for maintaining user productivity. With this tool, you recover valuable assets even if those assets no longer exist in the DB2 catalog.

► DB2 Log Analysis Tool for z/OS

This tool helps minimize the need for DB2 recoveries and maintains data integrity. With detailed reporting, the tool facilitates the identification, isolation, and restoration of unwanted changes to DB2 objects. The tool also helps your organization meet data change analysis requirements for regulatory compliance.

► DB2 SQL Performance Analyzer

This tool helps improve DB2 application design by taking SQL statements from any input source and providing estimated query costs. With this tool, application developers and database administrators quickly and cost-effectively explore various “what-if” scenarios to evaluate the performance of different database design options and production volumes.

- DB2 Path Checker and DB2 Bind Manager

DB2 Path Checker provides information about potential access path changes before the changes occur and about changes that occur after a rebind. DB2 Bind Manager automatically processes only the necessary binds, thus helping to minimize bind operations and conserve resources.

2.6 zEnterprise

The demands of a fast-moving market and the demands of customers, partners, and employees are stretching the limits of data centers. In addition to these demands, the management and integration challenges data centers face as they invest in the next generation of smart applications makes it clear that smarter computing systems are needed that increase efficiency, performance, and cost savings as management complexity is simplified.

The IBM zEnterprise System (zEnterprise) offers a revolutionary system design that addresses the complexity and inefficiency used in multi-architecture data centers. The zEnterprise extends the strengths and capabilities of the mainframe (for example, security, fault tolerance, efficiency, virtualization, and dynamic resource allocation) to other systems and workloads that run on IBM AIX® on IBM POWER7®, Linux on IBM System x®, and Microsoft Windows. By extending the strengths and capabilities of the mainframe, zEnterprise fundamentally changes the way data centers are managed.

The zEnterprise System is a workload-optimized, multi-architecture compute system capable of hosting many workloads integrated together but managed as one entity. The system is designed to deploy and intelligently manage workloads across mainframe and distributed technologies with the same tools, techniques, and management interface.

The zEnterprise System includes the following components:

- A central processing complex (CPC): The zEnterprise EC12 or the zEnterprise 196 (z196) or the zEnterprise 114 (z114)
- IBM zEnterprise BladeCenter® Extension (zBX) with its integrated optimizers or select IBM blades
- zEnterprise Unified Resource Manager

Until recently, the z196 was the fastest and most scalable enterprise system in the industry. To support a workload optimized system, the z196 scale up (over 52,000 MIPS in a single footprint), scale out (80 configurable cores) and scale within (specialty engines, cryptographic processors, and hypervisors), thus running in an environmentally friendly footprint. The z196 is designed to work together with system software, middleware, and storage to be the most robust and cost effective data server.

The zEnterprise EC12 is the most recent addition to the members of the System z family. With operational analytics and near real-time workload monitoring and analysis, clients use the new zEC12 for a variety of workloads and take advantage of the System's 25% more performance per core and 50% greater total system capacity than the z196 and the world's fastest chip running at 5.5 GHz.

The z/Enterprise System includes the following features:

- ▶ A *System of Systems* design that embraces the integration and management of multiple technology platforms (mainframe, UNIX, and x86) to improve the productivity of multi-architecture data centers.
- ▶ Supports z/OS, Linux on System z, IBM z/VSE®, IBM z/VM®, z/TPF, AIX, Linux on IBM System x, and Microsoft Windows operating environments.
- ▶ Unique hybrid computing capabilities powered by the premier enterprise server of the industry provide breakthrough innovation, virtualization, scalability, reliability, and security.
- ▶ Rapidly deploys services by using prepackaged solutions and pre-integrated technologies designed to meet the needs of specific workloads.

2.7 Additional benefits of DB2 10 for z/OS

DB2 10 for z/OS is enhanced to provide improved resiliency, availability, flexibility, and reliability. DB2 also realizes greater cost savings because of reduced CPU usage, simplified database management, and the use of proven technology. Featuring an elegant and simple design, DB2 10 for z/OS is one of the most powerful data servers available.

DB2 10 features the following benefits:

- ▶ Immediate CPU savings
- ▶ Proven resiliency for business-critical information
- ▶ Simplified application and warehouse deployment
- ▶ Enhanced query and reporting facilities

DB2 10 features the following performance and scalability benefits:

- ▶ Immediate CPU reductions
- ▶ Hash access to data and index include columns
- ▶ Ten times more threads than possible with the previous version of DB2, Version 9

DB2 10 features the following availability, security, and productivity benefits:

- ▶ Additional online schema changes
- ▶ Improved catalog, data, and utilities concurrency
- ▶ Row and column access control and masking
- ▶ Administrator privileges with finer granularity

DB2 10 features the following administration productivity enhancements:

- ▶ Utilities and autonomies
- ▶ LOBs management
- ▶ Profiles for distributed environments

DB2 10 features the following application advancements:

- ▶ Versioned data or temporal queries
- ▶ IBM pureXML® enhancements
- ▶ SQL improvements that simplify porting



Part 2

Migration methodology

We describe the migration methodology and the available alternatives to that migration in the following chapters:

- ▶ Chapter 3, “Application and database conversion methodology” on page 25
- ▶ Chapter 4, “Automated conversion methodology” on page 37
- ▶ Chapter 5, “Application and database conversion best practices” on page 47
- ▶ Chapter 6, “Choosing a migration strategy” on page 57



Application and database conversion methodology

Although most data centers understand and are familiar with application development, a database management system (DBMS) and pre-relational conversion often is new and unfamiliar territory. In this chapter, we describe the best practices for an end-to-end conversion methodology. We also describe the planning steps that are necessary and the decisions that must be made throughout the process to achieve predictable and manageable cost results.

“Good fortune is what happens when opportunity meets with planning.”

Thomas Alva Edison, 1847-1931

This chapter includes the following topics:

- ▶ Mature application challenges on the z platform
- ▶ Mature application options, examples, and terminology
- ▶ Conversion as a feasible solution
- ▶ Manual conversions

3.1 Mature application challenges on the z platform

As companies benefited from automation, they developed many applications that use customized software to run their business. Most of these applications use 3270 screen interfaces, large overnight batch jobs, and DBMS or flat files that are pre-relational in terms of how data is organized and accessed. With the advent of browser interfaces, relational database management systems (RDBMS), and service-oriented architectures (SOA) that support more flexible access to information, there are many new development options available.

Many users are requesting better data access and, when denied, users feel that their productivity is compromised. Those users who are granted access often find that they cannot understand the data because of the way the data is coded and accessed by older application systems.

These mature application systems present the following challenges:

- ▶ Companies have a huge investment in their existing application systems and cannot afford to discard or rewrite them. These application systems are essential to running their business.
- ▶ Business rules are found in the application programs with complex data structures so that efforts to give users real-time data access fail. The data often is extracted and then transformed to an RDMS platform for down stream processing that is difficult to support with a pre-relational data model.
- ▶ Maintenance accounts for 80% of the development budget, which results in pressure to cut costs.
- ▶ Various application systems often are integrated by data extracts and replicating data from one system to another. The cost to manage, integrate, and correct the data is expensive.
- ▶ Application downtime often results in a loss of user productivity and revenues. Unavailable real-time data also negatively affect revenues, productivity, and customer satisfaction and retention.
- ▶ Diminishing technical skills also make it difficult to find, train, and retain employees to support outdated technologies.

Under pressure from users to improve data access and from senior management to lower costs, data processing managers need to satisfy users and management as they facilitate designing and developing modern application systems. But, the thought of rewriting mature applications is daunting and managers often are left wondering what must be done.

3.2 Mature application options, examples, and terminology

The Migration Project Options Diagram in Figure 3-1 positions the project migration strategies in terms of the trade-off for technology benefits versus the project cost, risk, and time. The options that provide RDBMS technology benefits for the lowest project cost, risk, and time are the transparency and automated conversion tools (re-engineering is the most expensive). These strategies are used in many successful DB2 for z/OS conversion projects. For more information about worldwide solutions from IBM business partners, see this website:

<http://www.ibm.com/partnerworld/gsd/homepage.do>

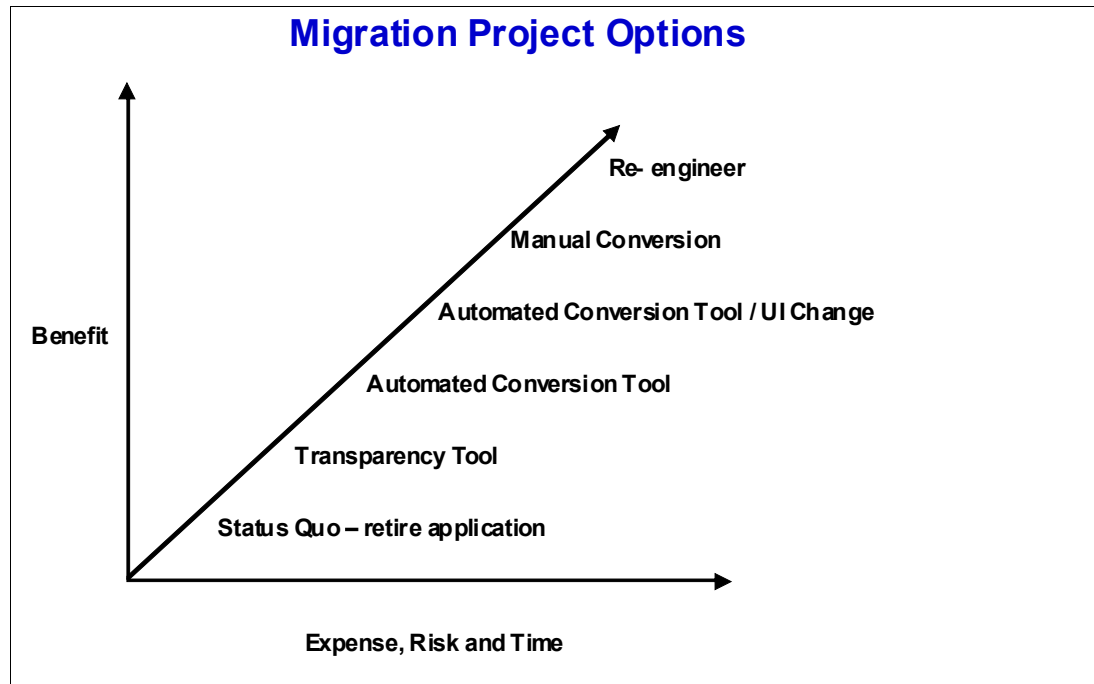


Figure 3-1 Methodology comparisons

Important: IBM makes no representations, warranties, or other commitments whatsoever about any non-IBM websites or third-party resources that might be referenced by IBM herein. A link or pointer to a non-IBM website does not mean that IBM endorses the content or use of such website or its owner. In addition, IBM is not a party to or responsible for any transactions you might enter into with third parties, even if you learn of such parties (or use a link to such parties) from an IBM publication, such as this publication. IBM is not responsible for the availability of such external sites or resources, and is not responsible or liable for any content, services, products, or other materials on or available from those sites or resources.

When you access a non-IBM website, even one that might contain the IBM logo, you must understand that the website is independent from IBM, and that IBM does not control the content on that website. You must take precautions to protect yourself from viruses, worms, Trojan horses, and other potentially destructive programs, and to protect your information as you deem appropriate.

The information in this book is not promised or guaranteed to be correct, current, or complete and might contain technical inaccuracies and typographical errors. You must confirm the accuracy and completeness of all information and how it might apply to your unique situation before making decisions related to any services, products, or other matters.

3.2.1 Application and database conversion (manual or automated)

A *conversion* is a change only in the application or database technology. Changes are not made to the user interface (UI) or other application inputs or outputs. The following examples show how the technology changes in a conversion, but not the functionality:

- ▶ Convert Software AG Natural to COBOL and Adabas to DB2 for z/OS
- ▶ Convert CA ADS/Online to COBOL and CA IDMS to DB2 for z/OS
- ▶ Convert Easytrieve to COBOL

In these conversions, the business processes remains the same and the user community does not need to be retained. Conversion is suitable for an organization with a large user base and a limited budget. This approach uses automated conversion tools, manual conversion tools, or a combination of both. Although conversion alters data and programs to suit DB2, conversion results in minimal changes so that it is consistent with DB2.

The data model is changed in a conversion to remove dependencies such as periodic fields, repeating groups, and last-record indicators. Primary and foreign keys are added, if necessary. The data is adapted to a new relational data model and the program data calls are adapted to fit the new target data model. As a result, changes to database calls and program logic are made to support the following functions:

- ▶ Redefine data as programs and repeating groups are split into tables.
- ▶ DB2 data types, such as date and times.
- ▶ DML calls are replaced with DB2 static SQL calls.
- ▶ Data from one file, record, or table is now included in two or more DB2 tables.
- ▶ Data from two or more files is now included in one DB2 table.
- ▶ Additional tables or relational constructs are required to define a pre-relational database model (such as a network) with pointers to a relational format.

3.2.2 Application modernization

Modernization retains the basic functionality of your existing applications and databases, but changes the look, flow, or general feel for the user community. Modernization changes the underlying technology for your support teams. A modernization of Natural to Java results in web-based applications that are maintained with another standard integrated development environment of your choice. This configuration is used rather than perpetuating 3270 routines that decide what key was pressed or count how many lines of data fits on a screen. Modernization tools use the capabilities of the J2EE Framework or .NET, thus generating methods, classes, functions, and objects that instead provide buttons and scroll bars.

Modernization provides a new set of web-based Java or .NET applications. The developer uses the existing application code and flow as the basis for the new application, guiding the toolset to generate a fully native web application for Java or .NET.

3.2.3 Application migration

Application migration does not include a change in the language or the type of database management system that is used. Migration might be an upgrade to a new release or the movement of a relational COBOL application from a mainframe environment to an off-mainframe environment. Migration is an exact mechanical reimplementations of the same application in a new environment or on a new release. In an automated conversion, the tools support the migration of the teleprocessing monitor to CICS.

3.2.4 Application replacement

In *application replacement*, all or some of the application functionality might be replaced with a software package in which the data from the mature application is converted to the software package.

3.2.5 Application rewrite

The *application rewrite* approach rewrites the program logic, such as the data access calls. The rewrite is begun from the beginning of the writing process when none of the existing application source code is used. The rewrite might be done to improve performance, convert from one program language to another language, or change a utility such as a TP monitor without changing the program functionality.

3.2.6 Application re-engineering

Application re-engineering is the modification of an application before or after conversion to add new functionality or to correct errors. The business processes also might be changed to make them more effective, efficient, and responsive.

3.2.7 Migration

Migration refers to an upgrade or change release for an application or software. In this publication, migration and conversion have the same meaning, which is to change the application database platform without changing the application system inputs or outputs. Exceptions to this definition include the online 4GL programs, which are converted to a browser interface. Any strategy that changes the UI also must have strong organizational support. The conversion strategy is to limit re-engineering and thus minimize the risk to the project.

3.2.8 Do nothing or status quo

In this instance, no change is made because the application is scheduled to be retired. This strategy might result in higher long-term operational costs if the hardware or software costs cannot be economically managed.

3.3 Conversion as a feasible solution

In the next sections of this chapter, we describe how to determine which optimal mature application candidates to convert and the best practices for planning and managing a conversion project. There are several migration project planning strategies available, but the objective of a conversion strategy is to use a solution that significantly lowers the cost, time, and risk of conversion project. The conversion solution is defined by using rules and tools that are used in multiple conversions. These tools are referred to as *automated conversion tools*.

Before you plan any conversion project, you must evaluate and document the current application architecture and components. You then analyze each strategy in terms of rewrite, re-engineering, or modernization. The purpose of this review is to qualify an application or part of an application subsystem as a conversion candidate. Ideal candidates are applications that have high business value and are strategic to the business. In terms of planning, any business process improvements or enhancements must be made before or after the conversion to minimize project risk.

There are some instances in which other conversion rules are developed by using automated conversion tools; for example, replacing the teleprocessing monitor (TP). Most automated conversion tools facilitate replacing the data communication (DC) TP monitor with CICS because CICS is a common TP monitor.

Any project in which the TP monitor was built by the customer or that had a small user base must be evaluated for replacement. The cost of the conversion is higher, as the rules to replace the TP monitor are not defined to the automated conversion tool. The project takes more time to complete, but a pilot tests the TP monitor replacement.

There are some application characteristics, however, that are better suited to a re-engineering project. A conversion has a higher risk of failure if the application and database are not suitable as the project requires re-engineering and conversion. Any planning for this type of project is not the same as the planning done for a typical conversion project because the application functionality is not replaced with reused code. Therefore, application conversion might not be the best solution in the following scenarios:

- ▶ Complex exits are needed to facilitate printing, security, in-built JCL streams, or other functionality that must be analyzed and replaced.
- ▶ A work station developer tool, such as Install/1, is used to generate program business logic that runs in an environment in which application recovery, data access, and other functionality is handled by a set of input/output (I/O) modules. These tools often are unsupported or require special services to modify the manner in which the programs are generated.
- ▶ A complex data model that requires significant modifications to convert to a relational data model. In this scenario, data often is randomly stored. The programs request data but do not directly access the flat files or DBMS in which the data is stored. These applications are often application packages that were purchased by customers by using a complex set of I/O modules. The I/O modules are written in Assembler language and include the following functionality:
 - Job scheduling
 - Data compression
 - Security
 - Auditing
 - Program error handling and recovery
- ▶ In some instances, the application is a candidate for conversion, but the application might run in multiple data centers or environments. Adding to the complexity is that the data centers or environments are incompatible in terms of production software levels, change controls, and the operating system that are used in each.

A rollout strategy also is a concern because there is no common platform in which to deploy the converted software. In this instance, some of the data center consolidation and upgrades must occur so that the environments are in synch with similar change control systems before you proceed with the conversion. Common change control procedures and target environments that have the same level of system software reduce the risk of unpredictable results that are difficult to diagnose. Controlled change management processes mitigate risk in situations in which data center consolidation is not an option.

A better approach for application architectures that require significant re-engineering is to identify and reuse the business rules from the programs for reverse re-engineering. However, application modernization to change the UI from a 4GL to Java, EGL, or other browser interfaces is supported by using automated conversion tools. As of this writing, the 4GL languages compatible with conversion tools include Software AG Natural and CA ADS. Application conversion projects that require significant re-engineering must have a detailed assessment or study that is conducted to determine whether an automated conversion is feasible.

3.4 Manual conversions

Companies often need to convert their data to a modernized architecture. However, a tools-based solution might not be feasible or available so an automated solution must be used. Companies often face the following challenges when they are searching for an automated solution:

- ▶ Complexity of data:
 - Value interpretation and preservation of complex business rules
 - Proprietary compression usage
 - Security and compliance
- ▶ Complexity of programs:
 - Languages used, such as Basic Assembly Language (BAL)
 - Knowledge of and interpretation of code
- ▶ Tool availability:
 - Exact match for conversion type
 - Meeting service level agreements and expectations
- ▶ Tool cost:
 - Budget constraints
 - Potential resources needed

Some mature data stores and applications are many years old and include embedded complexities that are not easily converted automatically by a tool. These complexities range from the sensitivity of the data to the fact that some older flat files might be stored in a proprietary compressed format. Even if portions are automatically converted by a tool, more manual effort and resources might be needed.

Manual conversions take more time (in some instances, many years and many user hours) to complete than an automated solution. Even so, the greatest benefit of a manual conversion is that users acquire an understanding of the current data and the application processes. This knowledge helps users to build a solution best-suited for their needs because they tailor their database to meet the complexities of the data and the application.

Whether the process is manual or automated, users must performance tune the application or database so the application or database takes full advantage of the DB2 for z/OS environment. This tuning helps minimize potential performance problems. For more information about tuning, see Chapter 11, “Application and SQL tuning” on page 175.

3.4.1 Sample methods for manual conversions

Most manual conversion methods are proprietary and therefore specific to the environment of an organization. However, we describe a few high-level approaches and their challenges. In any approach, the biggest challenges lie with whether the application is changing or if the conversion must be transparent to the user. The following manual conversion methods are available:

- No impact on the user

In large organizations with many remote users, a directive is issued that the conversion must not impact the application which leads to changes in the daily activities of the users. This directive often means that little or no changes to the code are made. However, maintaining performance during the conversion is a challenge because the no impact method often uses DB2 in a non-optimal manner. As a result, tuning DB2 after the conversion is difficult.

- Converting flat file data to DB2 large objects (LOBs) or VARCHARs

This conversion method is used because data is loaded into DB2 with few code changes. Users retrieve the data from the single field as if the data were a flat file expected by the application. Although this approach appears to be straightforward and fairly non-disruptive, this approach often results in an unusable data store.

Including the flat file data into an LOB or variable character field (VARCHAR) has the following advantages for users:

- Eliminates need to normalize data or provide for mapping
- Do not have to determine correct DB2 data types
- Better handling of garbage data, including the flat file in a DBMS supported LOB or VARCHAR eliminates redundancies
- Joins of tables are not needed to reconstruct file for application
- Easy comparisons and validation are made against mature data

Although there are advantages to this configuration, the following disadvantages outweigh those advantages if your goal is to convert to a relational DB2 database that takes advantage of all of the benefits the database offers:

- The new database does not perform as well as the old system.
- DB2 for z/OS LOBs and VARCHARs come with their own processing cost.
- LOBs and VARCHARs have size limits.
- Updates to the database are expensive because entire records must be updated.
- The purpose and the benefits of using a relational database (such as greater flexibility, ability to use SQL, and increased compatibility) are eliminated.
- The new database configuration limits future development.

- Using I/O modules

In this type of migration, the application and the user are not affected. Non-intelligent black-box I/O layers often are the chosen method to reduce application changes. However, I/O layers present the greatest challenge to performance because the layers are written to read each DB2 table separately. The calls that were expecting data from some type of flat file, such as virtual storage access method (VSAM), are now replaced with several calls to the I/O modules, which receive the data from the normalized DB2 tables. The data is then programmatically joined back together so to the application it appears to be the old VSAM file.

The result is that many SQL statements are issued which increases CPU costs. Although this type of manual conversion does work, the conversion rarely performs up to expectations unless some performance tuning efforts are conducted after the conversion. (for more information, see Part 4, “Post-conversion tasks” on page 153). The performance problems are compounded if the individual statements are placed in procedures that are externally. This configuration results in a call to workload manager (WLM) every time a statement is issued against a table.

- Denormalization issue

The denormalization issue is the result of a conversion project gone wrong. Problems often occur when an attempt is made to manually convert a database into a normalized design. However, the application is still expecting a flat file or some other format. These problems become expensive to support because the data is selected from several tables and changed to another format. Many users then denormalize the design so that the design resembles the established data store (as is expected by the application). Although this design increases performance, that performance is not optimal. The benefits of a relational environment also are not realized in this design. There are better ways to avoid the issue of denormalization, including the addition of an interface that determines which data is needed by the process. For more information, see “Intelligent application programming interfaces”.

- Intelligent application programming interfaces

The I/O module conversion approach is fast way to convert a database, but this type of conversion requires much work to complete.

When the application and user interfaces cannot change in a conversion, a smart interface that is built between the application and user interfaces is an effective workaround. These interfaces exist in an established system and are modified to meet the needs of a DB2 database. By using this approach, generic I/O modules are not developed to interact with the data. Instead, the modules determine the data that the application needs. The modules then selectively gather this data from the normalized tables and return the data to the application by using the power of SQL.

Migrating established systems in which applications cannot change and code expects a different type of data must be done in phases. In the first phase, the established system is accommodated as planning is made for the future. In a later phase, code is changed and the established access is abandoned so that the advantages of the relational database design are realized.

Planning the initial database conversion and anticipating performance issues that might occur during the conversion sets reasonable expectations of the users.

- Complete database and application conversion

Although complete database and application conversions take the most time and resources to complete, these conversions produce a better result because the database is customized to the data of the organization. Application performance also increases because the code is written with the knowledge that a relational database is used in the configuration. This code takes advantage of the powerful features of DB2 for z/OS.

If data is not converted (some organizations do not allow data to be automatically changed), a manual approach to the database conversion is preferred. In a manual database conversion, data is analyzed to determine the most efficient manner in which the different types of data are handled.

A manual conversion approach also allows for time to perfect the conversion process. The process changes when more is known about the data and how the data is stored and used in the DB2 database. This improvement process is done through the development of and testing of proof-of-concept databases. When the conversion process is improved, the time it takes to complete the conversion is reduced because the process is streamlined.

- Proof of concept

Manual conversions must have a proof of concept phase to ensure that the data is fully understood. This phase also helps prepare programs for conversion when a database is designed. A proof of concept is critical when you want to balance the performance of established and future access with the compatibility of the established processes.

Therefore, an application that reflects the established process must be built. These test programs are written in COBOL or REXX and feature simple query simulations. You also must develop a process that generates data (often by using query scripts and REXX) that is used to populate a test database.

The proof of concept phase simulates current and future access needs and identifies and addresses issues before development begins. Although these proofs of concept might take months to develop, this phase increases confidence in the conversion and helps develop and streamline the conversion process.

- Database for validation

As a proof of concept is developed, a validation database is created by using a quick conversion of the established data. By using data analysis queries, we review and understand the data that is contained in this validation database. This information is used to demonstrate how the established processes are accommodated and to confirm how the data is used in future processes. This information also is used to populate our final database design, thus creating an optimal database.



Automated conversion methodology

Automated conversion tools are an established technology that are used to convert databases from DBMS platforms to DB2 for z/OS. These tools simplify the conversion strategy by providing a repeatable migration methodology that reduces project cost, time, and risk. The use of automated conversion tools offers the following benefits:

- ▶ Application source code maintenance is easier as only the call structure is changed.
- ▶ Employees are familiar with converted programs.
- ▶ The user interface often remains unchanged.
- ▶ The conversion process is completed quickly and with less risk.
- ▶ The results of the conversion are easier to test and verify as application functionality is the same.
- ▶ The skills and expertise of client personnel are used.
- ▶ Changes to data and program structures are standardized.
- ▶ Program and database changes during the conversion are accommodated.
- ▶ The conversion process is documented in a before log and after log.
- ▶ Most automated conversion tools scan the program source code and database structures to generate reports. The reports are used to understand the application relationships for conversion decisions, identify missing and duplicate programs, and identify program interfaces for testing.

This chapter includes the following topics:

- ▶ Conversion tool data access options
- ▶ Automated conversion tool projects
- ▶ Data conversion guidelines
- ▶ Tool conversion strategies for 4GL programs

4.1 Conversion tool data access options

Most automated conversion tools use an input/output (I/O) layer approach, in which the programs must be recompiled. Additional tool options include program inline code to facilitate tuning, or cursor functionality that is inserted in the calling program or in the I/O layer to improve performance. An essential prerequisite for any migration project is to understand the application components and the degree of independence among the application components. In some instances, data is duplicated several times over different applications and systems. A single source of integrated data serves several applications.

Many conversion tools conduct a high-level analysis to determine whether the application is split into program and database-related pieces for the conversion. If a piece-at-a-time migration cannot be done, the conversion might require a *Big Bang approach* (a sudden turnover from the old system to the new system). There are strategies to avoid this approach, such as using transparency tools, dual updates, bridges, and replication to synchronize the data. The term *application bridge* is used as a generic term to indicate where one group of applications uses programs from another group of applications. The term *data bridge* is used as a generic term to indicate any other type of data movement not covered by batch and online processing when data is moved between systems. These terms also are called *application interfaces*.

The objective of the transparency tool conversion strategy is to convert the data without affecting the programs. This approach does not require that the programs are recompiled. Instead, this approach supports a table-by-table conversion, in which related tables are migrated together until the migration is complete. The data is migrated to DB2, and a special program is written to intercept calls to the old DBMS. This program translates the calls into SQL and transparently routes the calls to DB2 from the calling program. The programs remain unchanged and are started with the data in either database system. The transparency runs in an address space on the operating system and is designed to manage the calls of the old database.

The transparency generates I/O modules that become part of the application architecture. These modules handle the following functions:

- ▶ Intercepts database calls.
- ▶ Translates the source database call into one or more target database calls.
- ▶ Retrieves the data from the target DB2 database.
- ▶ Restructures the data into the source data, or old DBMS format.
- ▶ Returns data in such a way that the programs function as before.
- ▶ Translates SQL error codes from DB2 into codes the programs understand, or the transparency handles the error code.

Database failure with I/O modules occurs less often because there are fewer changes made to any one step and it is easier to roll back to an older version of the database.

The DB2 data model is optimized and tuned by using tools in this approach, but the more the data layout is changed to a new model, the more difficult the I/O modules are to tune. The transparency tool uses dynamic SQL, and all of the old code is used. The path lengths are longer in the transparency code. A transparency tool enables all of the data to be reformatted and migrated to DB2 for z/OS by using DB2 utilities. The program database calls feature the option to use the transparency or rewrite the calls by using SQL. Because of this option, the DB2 data model is a low-risk approach. At cutover, applications are unchanged and are rewritten and modified as required. This approach also is more suitable when the long-term

strategy is to rewrite the application, which eventually eliminates the use of the transparency tool.

4.2 Automated conversion tool projects

An automated conversion tool project has three main stages, which are separated with GO or NO GO decisions that are based on the outcome of the preceding stage. A conversion project has optional project stages, but the focus of this chapter is on the first three project stages and the use of tools to automate the conversion. A requirement to rewrite and tune programs during and after a conversion always is present. In Part 4, “Post-conversion tasks” on page 153, we describe several performance and tuning details that must be considered as part of stage 4 after the conversion activities.

The following main project stages are shown in Figure 4-1 on page 40:

- ▶ *Stage one* is the assessment, which sets the project scope, strategy, and objectives to meet the business and technical requirements for the conversion. An inventory of the applications is documented during a portfolio analysis study, and the options are evaluated relative to the cost. The application inventory is not detailed, but sufficient for planning purposes. A tools analysis of tools also is made to assess suitability and estimated cost of the tools. At this point in the assessment, one or more conversion strategies might be required because the applications might not share the architecture and conversion requirements. In this instance, a conversion priority must be set based on the individual projects, as a full conversion might not be feasible for all of the application systems. The initial budget and planning for the project must be started.
- ▶ *Stage two* tests the feasibility of the output from stage one and validates the activities, schedule, and cost of the project. A formal pilot might be conducted and the results documented. A detailed inventory of the applications is required to complete a firm fixed price for the project, which is estimated during the assessment. Any potential conflicts with other scheduled projects must be documented and the steps to review and fund the project must be completed.
- ▶ *Stage three* is the main conversion, which begins with a planning meeting. Most of the conversion and testing takes place as the application is migrated to production.
- ▶ *Stage four* covers the project post-conversion activities. The post-conversion activities must include the lessons learned and any recommendations for more application enhancements and process improvements. We do not address this phase of the project in this section of the methodology. For more information about post-conversion tasks, see Part 4, “Post-conversion tasks” on page 153.

If the DBMS platform includes automated conversion tools, the first three project stages that are described previously might be combined. The program inventory is assessed and a detailed project plan for the conversion is identified when the code is scanned and analyzed. The conversion of a few programs and database tables is used as a pilot to validate the conversion. This pilot conversion must be completed before the tools convert the full program inventory. The technical drivers that define a separate pilot in stage two of the project are required for the following reasons:

- ▶ The conversion tool requires further rule development to support the conversion. This development might be necessary because the tool is not automated for a particular DBMS platform. The tool also might require manual intervention to complete the conversion.
- ▶ The tool requires more rules to support some component of the application. For example, a change of the TP monitor from DC to CICS is supported, but the online programs might use a TP monitor that is not managed by the tools.

- The application has specific function or performance test requirements. In this instance, a test of the converted application is required in the target environment to validate functionality and performance.

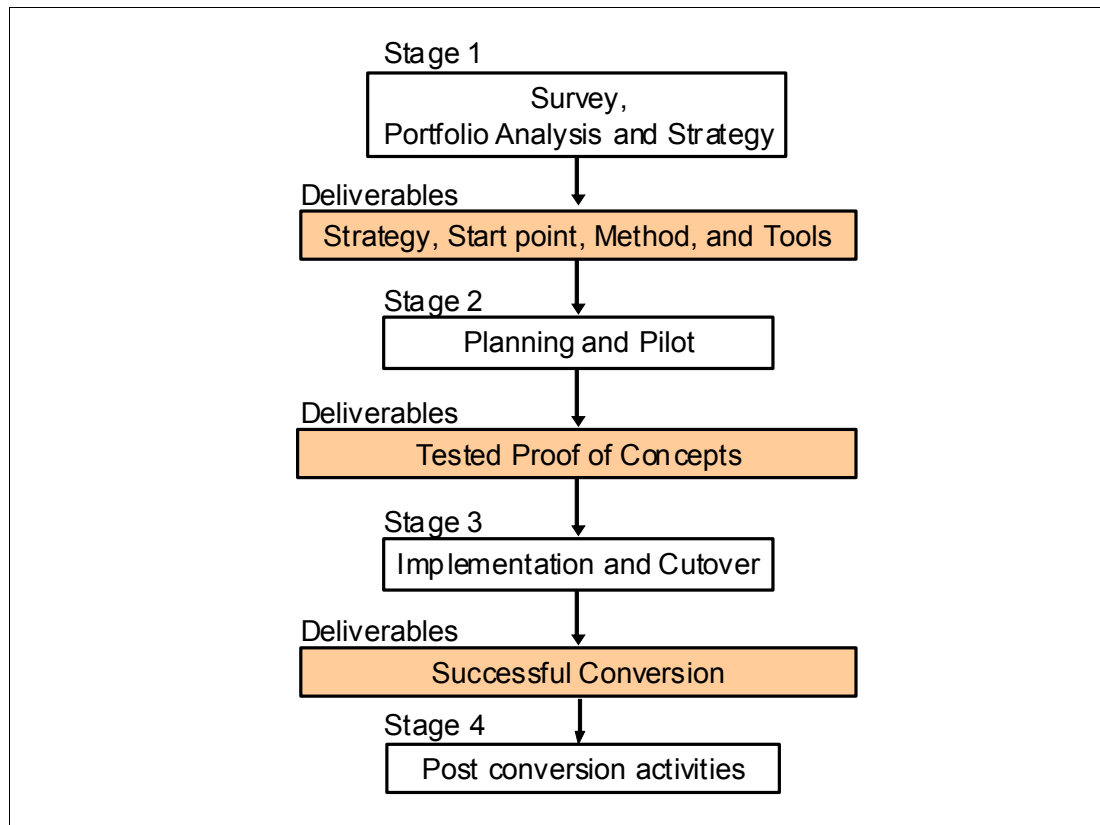


Figure 4-1 The four stages of conversion

4.2.1 Stage one: Assessment

The 10 steps that are shown in Figure 4-2 on page 41 must be completed before proceeding to the second stage of the project. The portfolio analysis and conversion estimate help to determine the initial project scope and budget. Any automated conversion or transparency tool that is selected for the conversion assists with reformatting the data for DB2 data loads after the source and target data model are mapped by using the tools. The data load programs are generated by the automated conversion tools after the data mapping is complete. The DB2 load utilities are used to load the data.

Any known data issues must be documented. A data clean up project often requires more tools and time over what is allocated in an automated conversion project. Additional project planning also is required if data cleaning is part of the project scope. The automated conversion tools provide the option to map to DB2 data types. In some instances, a DB2 data type is not included if special characters are used as part of the program business rule logic.

Future application business requirements that benefit from DB2 for z/OS database features, such as temporal data, also must be noted. These features are added as enhancements after the conversion and improve program productivity. Any specific DB2 for z/OS database requirements for the logical or physical model also must be documented for the conversion.

The stage one assessment includes the following deliverables:

- Identify the project business and technical benefits.
- Identify and assess the tools for the project. For more information, see Chapter 5, “Application and database conversion best practices” on page 47, Chapter 6, “Choosing a migration strategy” on page 576, and Part 3, “Tool-assisted conversions” on page 71.
- Identify the project scope, requirements, and strategy.
- Identify the project roles and responsibilities. For more information, see Chapter 5, “Application and database conversion best practices” on page 47.
- Identify more services for the project. For more information, see Chapter 5, “Application and database conversion best practices” on page 47, Chapter 6, “Choosing a migration strategy” on page 57, and Appendix A, “IBM Information Management Services Migration Practice” on page 221.
- List any outstanding issues.

The first project estimate is described in Chapter 5, “Application and database conversion best practices” on page 47.

The assessment also identifies the core team of the conversion project, and the skills and resources that are required to support the project.

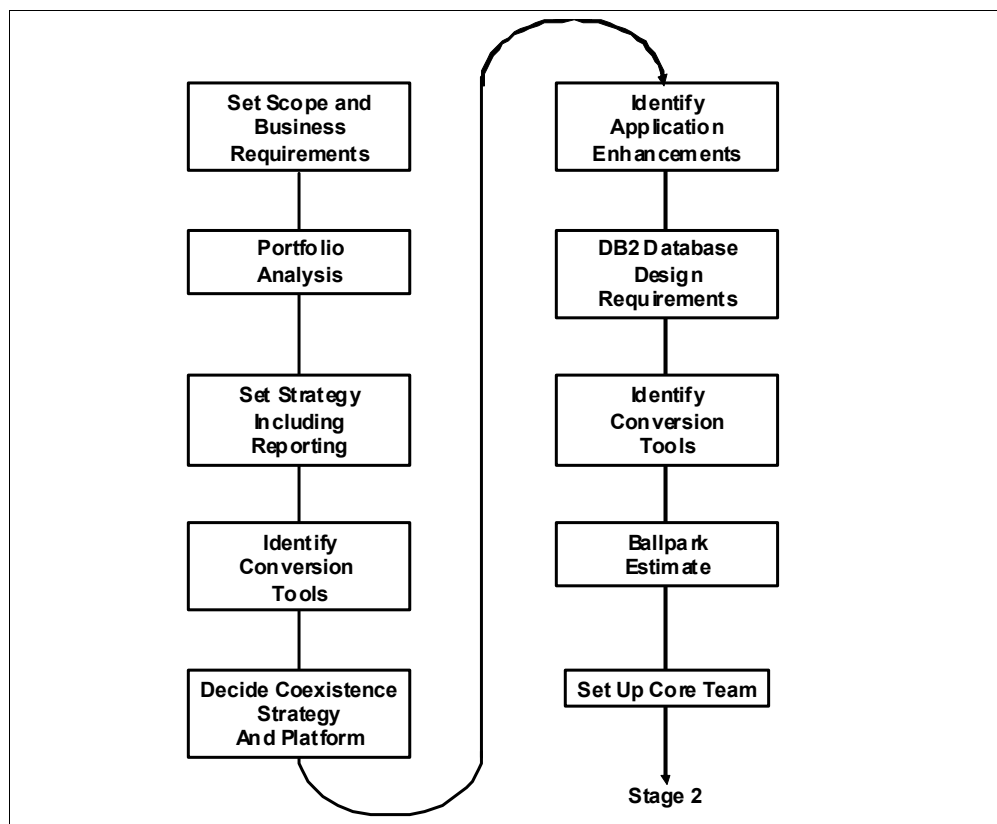


Figure 4-2 The 10 steps of the assessment

4.2.2 Stage two: Pilot

In the second stage, the feasibility of the stage one output is tested and a pilot is used to validate the chosen conversion method. The pilot requirements must be identified in the assessment and confirmed before stage two is started. The pilot must consist of a small group of transactions, programs, and data, including batch data. The objective of the pilot is to test application functionality so that unexpected results during the conversion are eliminated. Testing the functionality of the application validates the following items:

- ▶ Any selected tools to support the target environment:
 - Procedures
 - Standards
 - Productivity of programmers
 - Productivity of tools
 - Usability of documentation and reports
- ▶ Change control procedures
- ▶ Test scripts for the pilot
- ▶ Performance of the converted programs and data
- ▶ Equivalent function of the converted programs

The pilot must include the following aspects of the application:

- ▶ Programs that represent many transactions for load testing
- ▶ Some programs that are complex, moderate, and simple
- ▶ Some batch and online programs

Unless there are special conversion requirements for application modernization, the scope of the pilot must be small in size so that it is completed and tested within a few weeks. The following restrictions help to keep a pilot small:

- ▶ Approximately 50 - 100 programs
- ▶ Appropriate mix of languages
- ▶ Approximately 10 - 20 files or tables

The best pilot often is a small, stand-alone system that completes the conversion work and goes live. However, it is not necessary that the programs converted in the pilot go live.

A report must document the results, concerns, and recommendations of the pilot. A detailed estimate of the scope of the project, the resources that are needed, and the time that is required to convert the remaining applications is based on the pilot.

4.2.3 Stage three: Conversion

This conversion is the culmination of the work that was completed in stage 1 and stage 2. Project planning, coexistence, testing, change control, database design, code conversion, and manual rework are done to complete the conversion to the new database.

The first objective of stage three is to map and convert the data from the source DBMS to the target DB2 data model. The mapping is completed at the customer site where rules are defined on how to generate the target data model.

The automated tools generate the programs that reformat the data, but data quality issues might require multiple data loads. The following database loads must be conducted at the start of the project to identify potential data quality issues:

- ▶ Convert programs

The first step is to change the code (manually or by using a tool) so that the code compiles cleanly with SQL statements. If transparency is used, the tool must be installed and used in test mode.

- ▶ Review program code

The SQL used in this step might be tested with IBM Visual Explain. For a new DB2 program development, the SQL is checked by the database administrator before it is system-tested. For the main conversion, this check is useful. However, because function and performance are tested automatically so that delays and resources are reduced, only a few samples are required. Visual Explain must be used during the pilot to assess the quality of the SQL that is produced and to improve the performance of the SQL statements.

4.3 Data conversion guidelines

The following acceptance criteria for the deliverables in stage 3 of the conversion process often are defined during the contract negotiation process with IBM or the conversion vendor:

- ▶ Database design requirements and standards
- ▶ Conversion program standards
- ▶ Converted data standards
- ▶ Generated data definition language (DDL)
- ▶ Standards for developed DB2 backup and recovery, load and unload scripts, or jobs
- ▶ I/O module standards
- ▶ Converted program standards

Any DB2 standards that customers use must be reviewed for the conversion. The standards might be altered to allow exceptions or to add special migration requirements. Many of the user-friendly automated conversion tools are used to define the following conversion rules and considerations:

- ▶ The process that is used to convert the date and time data must be determined. As this determination is made, the following questions must be considered:
 - Is the date and time format compatible with DB2 date and time data types?
 - If the data is not compatible, is the data easily converted to a DB2 date/time format?
 - If the data is not in a valid DB2 date and time format, is the data defined as numeric (INTEGER) or as character (CHAR)?
 - If the data is numeric, is the data guaranteed to be numeric? (Some databases allow character data in fields defined as numeric.) This determination helps define data as CHAR or INTEGER in DB2.
- ▶ Are there any conversion rules that must be developed to manage instances in which data is stored in one format but is displayed in another format?
- ▶ Is numeric data defined as DECIMAL or as one of the INTEGER data types in DB2?
- ▶ Based on the value of the code fields, are the code fields defined as CHAR or as numeric?
- ▶ Are large character data defined as CHAR or VARCHAR?

- ▶ The following general conversion rules for creating database objects also must be defined:
 - The buffer pools used for tables and indexes
 - The amount of space that is needed for each object (pre-relational space information is used as input)
 - The locking parameters to use
 - The tables that are partitioned and the rules to determine the partition
- ▶ If the automated conversion tool supports generating job control language (JCL), the rules for generating JCL must be defined.

After the conversion standards are set, an initial test of the data extract process is run to validate that the counts results or values are as expected. After the information about the counts is compiled, you compare the results with any other reports that are run. This comparison produces statistics on the pre-relational database and helps you determine whether multiple passes of the data are required to generate keys for related sets.

Next, you run an initial test of the data load process to validate that counts are as expected. Address any issues in which data is rejected because of conversions problems. Alternatives for managing referential integrity issues are included in the following considerations:

- ▶ Are parents loaded before children?
- ▶ How are concurrent loads scheduled so that the elapsed time window of the load process is minimized but referential integrity (RI) issues are still addressed?

Report back to the conversion vendor any issues that are found during the test so that problems are corrected and subsequent tests yield improved results.

4.3.1 Preparing for conversion testing

Complete the following steps to plan for conversion testing:

1. Complete the program preparation steps for all of the converted source code and generated I/O modules.
2. Verify that the programs compile, link, and bind cleanly.
3. Run Visual Explain on all the generated SQL to ensure that the created access paths are as expected.
4. Test the converted applications, database, and JCL.
5. Document any issues in a problem tracking system and meet with the conversion vendor regularly to review problems and the status of how those problems corrected.

Repeat these steps for all unit, system, and integration testing.

As tests are conducted, the following suggestions help produce better test results:

- ▶ Keep a copy of the pre-relational database (including a backup) that is rebuilt for repeatable testing.
- ▶ Establish a copy of the converted DB2 database (including a backup) that is rebuilt for repeatable testing.
- ▶ Decide on a series of online transactions and batch jobs that are used for performance testing.
- ▶ Ensure that the selected workload mix includes short-running and long-running work, and applications that are known potential issues under the pre-relational version of the database.

4.3.2 Addressing bugs and performance issues

Complete the following steps to address any bugs and performance issues that are found during the testing process:

1. Run a benchmark or load test of the pre-relational database and applications.
2. Agree with the client on a service level agreement (SLA) for the converted benchmark as the benchmark relates to the pre-relational tests.
3. Run a benchmark of the converted database and applications.
4. Produce a report that shows the performance results.
5. If there are any instances in which the converted applications do not meet the agreed upon SLA, collect a list of those applications and address those issues with the conversion vendor and the conversion team.

Repeat this process until all of the performance issues are addressed.

4.3.3 Scheduling a dry run

A dry run of the conversion helps to identify any problems or issues in the process. A dry run also gives developers the opportunity to fix these problems before the real conversion is conducted.

The following factors must be considered when you are planning a dry run:

- ▶ Schedule the dry run on a weekend when you test the entire conversion process on a production-sized volume of data with a minimal effect on users.
- ▶ Include in the dry run the data extract, convert, and load processes. These programs must be prepared by using the production-sized DB2 after the data is loaded and the RUNSTATS function is run.
- ▶ Include enough time in the dry run to back up all of the DB2 tables after the data load.
- ▶ Validate that the applications work properly and that the data is converted correctly.
- ▶ Track the duration of the dry run to ensure that the entire process is completed in the time that is allotted for the conversion.

4.3.4 Planning for a backout

You must conduct a database backout if serious problems occur during the conversion process. The backout recompiles the programs by using the pre-migration options or by switching the JCL to point back to the load libraries of the old database.

The application data recovery strategy depends on the tools and migration strategies that are used for the conversion. For more information about the strategies for data co-existence and application migration, see Chapter 6, “Choosing a migration strategy” on page 57.

4.4 Tool conversion strategies for 4GL programs

There are many successful automated conversions in which the online 4GL programs were converted into pseudo-conversational COBOL and CICS. COBOL/CICS is a common conversion target but requires COBOL skills and resources to maintain the programs. For that reason, there is interest in alternative conversion targets in which the online application programs run on alternative platforms and modernize the UI. The Java target is supported for Software AG Natural, CA Ideal, and CA ADS.

The following conversion targets are supported for the Software AG Natural 4GL:

- ▶ Java or .NET conversion
- ▶ Java runtime environment in which the source code is maintained in a business developer language (such as IBM EGL) to simplify program maintenance and then generate the Java or COBOL code dictated by the runtime environment
- ▶ Use of an application runtime server and Consist Advanced Development Solution (ConsistDS) to simplify program development and maintenance
- ▶ A Natural-to-COBOL conversion approach for the z/OS platform that supports a conversational COBOL program structure that runs under CICS with exits and a run time that manages the pseudo-conversational aspects for performance
- ▶ Workbench tools that facilitate re-engineering programs to Java environments

There also are transparency tool options available for Software AG Adabas conversions that support the execution of the Natural and COBOL programs after the database is converted. The end state of the conversion, however, is to not use the transparency access. The end state of the conversion is to rewrite or convert the 4GL to another target. The transparency option is a bridge until the Natural programs are rewritten or converted.

4.4.1 The 4GL dilemma

The application programmers that use 4GL programs are business developers. The business developers often require a flexible programming environment that facilitates application productivity. The 4GL program environment requires fewer instructions to code and maintain and the program flow is conversational. A 3GL target might not be a programming environment option, unless there is an organizational plan to change the programmer skill set. Another high-level language replaces the 4GL, but the programmer integrated development environment (IDE) is affected. Other IDE considerations include the per seat license cost, the web application server for the target environment, and the changes to the user interface (UI) if a browser technology is used.

An organization with a large-scale application that runs on the z system platform might want to consider the approach of a conversational COBOL that runs in a pseudo-conversational CICS environment. This option offers both flexibility and performance. For organizations in which the programmer skill sets are distributed, a Java browser-based approach might be more suitable. This approach often is a mix of choices if there are multiple departments with different skills. A strategy to manage the 4GL options, however, must be communicated within the organization to determine the appropriate conversion requirements.



Application and database conversion best practices

There are many successful DB2 for z/OS conversions, and there are conversions that did not yield the expected results. In the past, some businesses that were considering a conversion overstated the difficulties of a conversion and did not migrate properly or underestimated the task. As a result, conversion projects were over budget and take longer to complete. The best practice conversion guidelines that are described in this chapter are a summary of the lessons that were learned from many projects.

This chapter includes the following topics:

- ▶ Summary of best practices
- ▶ Stage one conversion planning
- ▶ Conversion skills and roles

5.1 Summary of best practices

The following best practices apply to most conversion projects:

- ▶ Establish a project charter, allocate resources, and assign project roles and responsibilities. Project planning, allocation of staff resources, and a commitment to the conversion at all levels of the organization are essential to the successful of a project.
- ▶ Estimate the project metrics (program counts, lines of code, number of files or tables, and so on) to define the initial project scope, and the time that is required to test the application by using test scripts. These metrics also are used to develop an estimate for the project budget and time. This process is called a portfolio analysis, which is part of the assessment planning. Select applications that have business value for this evaluation.
- ▶ Plan on a three-stage approach for the conversion project that includes an assessment, a pilot, and a set of pre-planned methodologies that is repeatable for the main conversion.
- ▶ Identify the conversion requirements and (if feasible) the automated conversion tools that support your requirements. By using automated conversion tools, you significantly reduce the cost, time, and risk of the conversion project.

The following criteria must be included in an automated conversion tool evaluation:

- Automated conversion tool approach with multiple project reference
- Code refresh capability to support database and application changes during the conversion process so that no freeze is required
- Ability to support a functional proof of concept for the conversion of some of the programs and data at an IBM or conversion vendor facility
- Capability to support more than one target for a 4GL language, including COBOL, EGL, or Java for conversion flexibility
- Capability to generate data load programs that reformat and transform the data
- Does not use proprietary code that requires license or maintenance fees

The following business and technical migration requirements might be included:

- ▶ Additional contract services for:
 - Developing test plans, test scripts, and providing test assistance
 - Setting up and customizing a DB2 environment
 - DB2 and application performance and tuning
 - Education and skills transfer
- ▶ Conversion consulting services to handle co-existence strategies, database design trade-off decisions, conversion of large databases, and other conversion challenges.
- ▶ Services for process re-engineering after the conversion of data replication, business intelligence, data warehouse and reporting strategies, portals, web enablement, and security enhancements.

The objective of the conversion is to develop a target DB2 data model that preserves the current program logic. This approach might incur more CPU usage but has the same response times. A small percent of manual conversion is often required to improve the DB2 performance during and after the conversion.

- ▶ Analyzing the data usage in the programs helps develop the following conversion rules:
 - If the program uses all of the retrieved data, over-normalizing is not the best alternative to converting to DB2.
 - Plan on modifying a subset of the programs to filter the data that is required by the program to improve DB2 performance.
 - If possible, verify the need to define additional indexes.
- ▶ Plan on general DB2 performance and tuning throughout the conversion, especially for indexes. For more information about post-conversion details, see Part 4, “Post-conversion tasks” on page 153.
- ▶ Confirm that in-house subject matter experts who know the application are available during the migration project.
- ▶ Ensure that sufficient test tools, resources, and plans are developed to test the conversion. An assessment of the hardware and software resources that are required for the migration test environment must be identified during the assessment planning. There must be sufficient capacity resources if load and performance testing is required for the migration. Although a full unit test often is not required, integrated testing must be included.
- ▶ Start the batch testing as soon as the test data migration is complete if there are many batch processes to test.
- ▶ Identify special production requirements if a transparency tool solution is selected. For example, the IBM CICS VSAM Transparency for z/OS tool runs in the same logical partition (LPAR) as DB2.
- ▶ Define the target environment for the conversion. If there is a 4GL language to convert, identify a target language for this conversion. Confirm whether DB2 for z/OS is installed in the target environment. If DB2 is not installed in the target environment, include tasks in the project plan for the products that require installation and employee skills acquisition.
- ▶ Identify performance and tuning tools to support the conversion.
- ▶ Plan on a piece-at-a-time conversion strategy to avoid a Big Bang conversion. If this strategy is not feasible, consider the use of a transparency tool or other tools that support co-existence strategies to avoid a Big Bang conversion. These tools might also facilitate developing backout strategies for the conversion.
- ▶ Collect system usage information and transaction counts for the current application that are used for an informal capacity sizing assessment of the target DB2 for z/OS application. A proof of concept with a completed load test that validates performance for critical workloads with service level agreements.

5.2 Stage one conversion planning

When you are thinking about a conversion project, there are more questions than answers. Determining the size and budget of a project is difficult if basic information is unknown. An estimate for the conversion is developed by using the initial application and database metrics to understand the project scope and available tools. The business and technical requirements also must be documented to justify the project.

The following sample stage one planning issues must be considered:

- ▶ **Project overview**

Identify the goals for the assessment, the overall business-related factors that affect conversion decisions, and the overall business and technical direction for the next 3 - 5 years relative to the migration.

- ▶ **Conversion tool scanning**

Source code is scanned through the conversion tool to determine the number of database calls, verb usage, and any unique instances of code encountered. Code and data structures are reviewed to determine the use of occurs clauses, repeating fields, and code translation tables.

- ▶ **Database design considerations**

During the assessment, it is determined that the current data structures in the source system require a redesign. There might be a need to simplify database maintenance by merging data into partitioned tables, eliminating redundant data, or modifying the use of audit trails. A special analysis is required to create a new logical and physical database design.

- ▶ **Change control procedures**

If any code enhancements or changes are planned during the conversion project, change control procedures must be defined and managed.

- ▶ **Co-existence alternatives**

Based on the data conversion strategy that is considered, co-existence options must be reviewed and understood. The options are reviewed during the assessment phase if both the source and the new target database must be updated to remain in synch until cutover. This assessment is an evaluation of the technical solutions, cost, and resources to support parallel environments.

- ▶ **Language selection**

Review language options for the new target environment. For example, you must determine whether the target environment uses CICS/COBOL, Java, .NET, EGL, or another target for any 4GL programs.

- ▶ **Test preparation analysis**

Determine the current test capability of the source application systems to understand if the current test beds are viable as test beds for the target system. You must also determine whether the current test tools, scripts, and plans are sufficient to cover all of the applications. If they are insufficient, you must ensure that there is enough documentation to develop the test plans and scripts.

- ▶ **Capacity studies**

A capacity study is an optional fee-based study that is conducted to predict CPU disk space requirements for the conversion.

- Phase one assessment output

Based on these planning issues, a written report is provided, which includes findings and recommendations. This report outlines the project scope, which includes the estimates of time, expense, and resources that are required to conduct a successful conversion. The report also might include recommendations and requirements for a pilot.

5.2.1 Application survey and initial project sizing

Sizing the application is the start of an informal application assessment that is used for conversion project planning. The assessment provides an estimate of the cost of and the time necessary to complete the project.

The estimate is based on information about the current application environment, target environment, and the resources that are required for the conversion project. This estimate is provided with some assumptions, and must include contract resources to assist with performance and tuning. The project cost is dependent on the following questions:

- How many programs are there? How many lines of code? How complex is the code?
- Are all of the programs the same language?
- Do all of the programs make DBMS calls?
- Does the application architecture use a transparency (for example, for VSAM)? Is there a mix of calls to convert?
- Is the current documentation updated?
- What are the current problems with the application?
- What is the mix of batch and online programs?
- Are any of the 4GL program languages slated for replacement?
- If you move one application and its data to DB2, are there other systems that use the same data?
- Is a single application system split so that the system is moved incrementally?
- How many database tables or files exist in the system? How accurate is the data in the DBMS?
- How accurate is the data model of the existing system?
- Do valid test scripts exist to validate the results of the conversion?
- What skills are needed to support the conversion?
- Is there a need for more services to assist with the conversion?

The conversion vendor uses this information to work with the client and IBM to develop an initial budget and planning estimate for the project. The DB2 Migration Project Office uses a set of questionnaires that customers are requested to fill out to assist with the migration planning, estimate, and migration requirements specific to a customer environment. The DB2 Migration Project Office also conducts conference calls with the IBM Sales Team and the customer to discuss conversion options.

An estimate often is drafted with IBM sales support when a target roadmap is developed for the flat file, VSAM, or DBMS conversion. The estimate also is drafted when the feasibility of using automated conversion or transparency tools is assessed.

Table 5-1 on page 52 shows the project planning information that is required to initiate a project.

Table 5-1 Project planning estimate content

Content	Comment
Project objectives	Develop budget and planning estimate
Project assumptions	This budget and planning estimate is developed only for planning purposes
Strategic direction	Database consolidation, eliminate architecture dependencies
Business needs and environment	Use system z investment
Scope	Budget and planning
Requirements	Reduce infrastructure maintenance
Deliverable	Budget and planning estimate
Organization sponsor	IBM sales team
Delivery organization	To be determined

An estimate often includes the following assumptions:

- ▶ The estimate is not a substitute for a formal assessment or proposal.
- ▶ Detailed planning is required to finalize this estimate.
- ▶ The estimate is validated after the application programs are reviewed manually and with tools for sizing information.
- ▶ Program conversion is done offsite by using automated conversion tools.
- ▶ All database objects are converted to DB2 for z/OS objects.
- ▶ The estimate is a migration sizing, not a workload sizing.

The inventory sizing for the estimate might require more assumptions if accurate inventory metrics are not available for each of the applications. This task is complex if the program libraries contain multiple program versions, do not use change management processes, or have programs that are in many repositories that are difficult to locate. In this instance, a fee-based assessment that is conducted with the automated conversion tools might be the best alternative because there is insufficient information for an estimate.

Assessments might be customized only for an inventory sizing or other tasks to address other requirements for the migration. A formal assessment to size the application inventory and analyze the source programs for complexity and conversion is used for projects that are large and complex in scope. Additional analysis might include a CPU or disk capacity sizing, a co-existence conversion strategy, a target environment setup, and a customization and migration readiness review.

The objective of a formal fee-based assessment is to use the selected automated conversion tool that is identified at the beginning of the assessment to scan the program inventory. This reason is why an estimate for planning purposes is the initial step in the process. This information helps developers decide which tools to use for the conversion. In some cases, there is a transparency and an automated conversion tool solution. In this instance, the money that is spent on a fee-based assessment might be spent for a pilot of the transparency tool. An inventory sizing is not required for a transparency tool solution.

The following factors influence the selection of a transparency tool:

- ▶ The transparency tool license might be less costly and less maintenance might be required than the cost and maintenance for an automated conversion.
- ▶ The speed of the conversion might dictate if a transparency tool is used.
- ▶ The transparency tool requires one license and does not run in multiple data centers.
- ▶ There are minimal components or application components (for example, 4GL, and TP monitor) that must be converted.

5.2.2 Fee-based assessment

The fee-based assessment is customized to a customer environment, so the duration of the assessment varies. A smaller assessment takes 6 - 8 weeks to complete. The first 1 - 2 weeks are spent at the customer site in workshops. Interviews are conducted to demonstrate, document, and understand each of the applications that are used by the customer. The assessment often includes the following activities:

- ▶ The portfolio of programs and applications of the customer is analyzed to help determine what must be migrated.
- ▶ Detailed information is gathered about the applications and environment of the customer.
- ▶ Customer management and staff are educated about the central issues that are addressed during a migration.
- ▶ The need for re-engineering during the migration is discussed.
- ▶ Migration experiences and best practices are shared.
- ▶ The requirement for other complementary strategies to manage co-existence (such as a data warehouse) is discussed.
- ▶ Project planning, capacity planning, and high-level budget and planning numbers for the migration are discussed after a detailed application inventory sizing is completed.
- ▶ The next steps for the conversion are defined, and often include planning for a pilot.

A fee-based assessment results in the following deliverables:

- ▶ A report that contains the program inventory reports, project plans, and a migration analysis with recommendations. An executive presentation also is included in a round table discussion for customer feedback. The input from the round table is included in the final version of the report.
- ▶ A final executive presentation that includes a fixed price for the conversion or recommended next step with assumptions. A pilot is the next feasible step. In this instance, the fixed price is set for the pilot and an estimate for the conversion is included.
- ▶ Any other analysis or deliverable that is identified as part of the assessment.

5.2.3 Options risk analysis

The options risks analysis is used to determine the most applicable conversion pattern for the budget and planning estimate. As shown in Figure 5-1, a green arrow that points down represents the least risk, a brown arrow that points to the right represents moderate risk, and a red arrow that points up represents the highest risk.
















Conversion Strategy	Technology Risk	Execution Risk	Cost	Time	Old DBMS Dependency
Big bang, keep old 4GL So risk depends on 4GL removal					
Phased migration with 4GL change					
Big bang with 4GL change, risk is skills retraining					

Figure 5-1 Options risks analysis

The target of the 4GL conversion risk must be assessed. Retaining the 4GL language without changing the language might have hidden costs in terms of license fees and finding skills. The technology risk of the 4GL also must be assessed in terms of the strategic value in continuing to use a niche language. Modernizing the 4GL to a 3GL includes hidden costs because of higher maintenance needs. Conversion to another higher-level language (such as EGL, which is open source) has the benefit of lower maintenance costs because it is easier to use.

The dependence on the database platform risk changes depending on the data and converted program migration strategy. A Big Bang approach increases the risk, but a phased approach lowers the risk. A Big Bang approach might be faster to implement and lower in cost, but such an approach is higher risk. Other options also might be assessed in the risk analysis. This analysis is a way to evaluate the trade-off of many different conversion decisions.

5.3 Conversion skills and roles

A conversion project requires the right mix of skills of everyone that is involved in the project. To be successful, the project also must have the support of the organization at every level.

The staff who are the subject matter experts for the old DBMS platform and application must understand that they are not losing skills. Instead, they have an opportunity to gain new skills in relational technology in general, and in particular in the market leader, DB2. Such skills are important to their future careers. Conversion to DB2 must be seen as a prestige project that requires the highest expertise.

The following project roles and skills often are essential for a conversion project:

- ▶ **Sponsor**
A committed executive sponsor who wants and needs the project to succeed.
- ▶ **Project manager**
A full-time person who is responsible for project planning, scheduling resources, and communication.
- ▶ **Technical team leader**
This full-time technical architect ensures that all technical issues are resolved, that the staff is producing the correct output and that performance is satisfactory. This team leader also facilitates any necessary skill transfers, and makes available their expertise as needed. This role requires an experienced, senior-level professional.
- ▶ **DB2 database administrator**
The database administrator (DBA) is responsible for the installation, configuration, and tuning of the database in the organization. At the outset of the project, the database administrator role is a full-time position. When the prevalent DB2 issues are resolved, this role might become part-time if the technical team leader is familiar enough with DB2. The DBA might be full-time and the technical team leader might be part-time when the third stage of the project begins. However, this paradigm is possible only when the DBA is a senior-level staff person who understands the overall project picture.
- ▶ **Business analyst**
The business analyst is a full-time role until stage three is in the advanced phase. A business analyst manages system and integration testing, and serves as a consultant during the conversion process. The analyst also assesses the effect of any issues on the business that are the responsibilities of the users or their representatives.
- ▶ **Conversion consultant**
This consultant is external to the business and is staffed by someone who understands the conversion process. The conversion consultant asks relevant questions and assists with project management.
- ▶ **Source database administrator**
This DBA is thoroughly knowledgeable about the source DBMS, provides the business data model in use, and understands the process to unload data and how the DBMS is used. This person also has a strong relationship with the DB2 DBA.
- ▶ **Source application programmer**
Although the code is changed in some areas, programmers who understand the old system and have access to the documentation of the old system are invaluable to the target application programmers.
- ▶ **Target application programmer**
These programmers might work within or outside of the business or be brought in as they are needed. The programmers must have a thorough understanding of the language of the target database and how to write quality SQL code.
- ▶ **Systems programmer**
The system programmer installs DB2 and ensures that the correct links to z/OS are in place and that the adopted naming standards conform to company standards.
- ▶ **Test system specialist**
This specialist is responsible for setting up and running the adopted testing systems. The person in this role ensures that testing is comprehensive. The specialist also advises the

testing team in how to use the technology and informs the users of any limitations of the technology.

- Operations analyst

This person is concerned with the conversion of the job streams and utility jobs, and reviews the design of the backup system that is in place. This analyst supports the job control language (JCL) and the environment that is used to run the conversion.

- Change control specialist

This person controls the many small design changes that are requested and ensures that users are aware of those changes. This specialist validates the changes with the technical team leader.

- Performance specialist

This specialist understands all of the performance tools that are used in the conversion and assists the team in understanding any performance-related issues that arise.

- Transaction manager specialist

If the transaction manager is changed, expertise in both the old monitor and the new monitor is needed. Expertise in this area is needed during the conversion if this change is not made.

- Delivery manager

When code is sent offsite, managing the conversion process includes other tasks outside of the standard application development process. The delivery manager must track which programs are in progress, which programs are under test, which programs are returned, and the reason for the status of those programs. The manager also must be aware of repeating error patterns that occur during the conversion so that the process and quality are improved.

How a migration is planned and conducted depends on the customer. The project might be led by the customer who is then responsible for coordinating the project phases and resources. Customer-lead conversion projects often are easier to manage when a transparency tool is used. IBM provides DB2 tuning and performance and testing expertise as needed during the conversion. These types of conversions feature customers who have significant skills and expertise in using DB2 for z/OS.

Many automated tool conversions, however, require consultants with more skills and expertise in IBM-based technology to accelerate the conversion process. IT executives must use a vendor that understands the conversion approaches under consideration and has the experience in migrating DBMS platforms by using various tools and technologies.



Choosing a migration strategy

Organizations spend considerable resources and assume more risk than necessary when migrating applications and data to a relational platform. There are many migration approaches, but the best way to reduce risk is to plan a conversion that limits changes. The best migration option, therefore, is to convert from a DBMS or file system platform to DB2 for z/OS. The use of migration tools further reduces the risk and cost of the migration.

A modern relational database also prevents technological obsolescence of the application, and enables further application modernization after the conversion project is completed. The typical conversion takes a minimum of six months to several years to complete. The migration project must be broken into several smaller projects if the anticipated project scope for the conversion requires more than two years.

In selecting a migration strategy, each of the strategy options must be evaluated with the availability of tools from IBM and a third party that facilitate the conversion. In some instances, the lack of tools for specific pre-relational platforms leads to a manual conversion or a co-existence solution. The choice of conversion tools is much larger if there are a number of installations in which customers invested in a specific pre-relational technology. Although the availability of options changes, the overall trend is an increase in the number of tools to help migrate to relational database platforms.

A conversion strategy and the tools that support the business and technical migration requirements must be identified during the stage one assessment of the process as described in the following chapters:

- ▶ Chapter 3, “Application and database conversion methodology” on page 25
- ▶ Chapter 4, “Automated conversion methodology” on page 37
- ▶ Chapter 5, “Application and database conversion best practices” on page 47

Within North America, the IBM DB2 Migration Project Office advises customers on the latest tools and options to convert from pre-relational databases to DB2 for z/OS. Our role is to enable customers to cost-effectively migrate to the DB2 family of products from any competitive or non-relational database product. We help assess the level of effort to convert based on the available migration strategy options, and coordinate with other IBM service teams or with the DB2 Labs. This support also is available on a worldwide level. For more information, see this website:

<http://www.ibm.com/software/solutions/softwaremigration/dbmigteam.html>

This chapter includes the following topics:

- ▶ Conversion strategies
- ▶ Automated conversion tool case study
- ▶ IBM consultancy contact information

6.1 Conversion strategies

There are several conversion strategies available and each option affects the programs and data of the system, as shown in Figure 6-1. As the strategies are reviewed, consider the following starting points that are common to planning this type of project:

1. Identify the business and technical requirements.

Identifying these requirements is the general approach to answering the following questions about moving to DB2:

- Is the time taken to conduct the migration the most important parameter?
- Is the least amount of risk paramount?
- Are reporting programs available?

2. Choose a starting point.

You must choose a starting point for the migration. The starting point might be decided by answering the following questions:

- What is the first application to be moved to DB2?
- Is the entire application moved or only part of the application?
- How is coexistence managed?

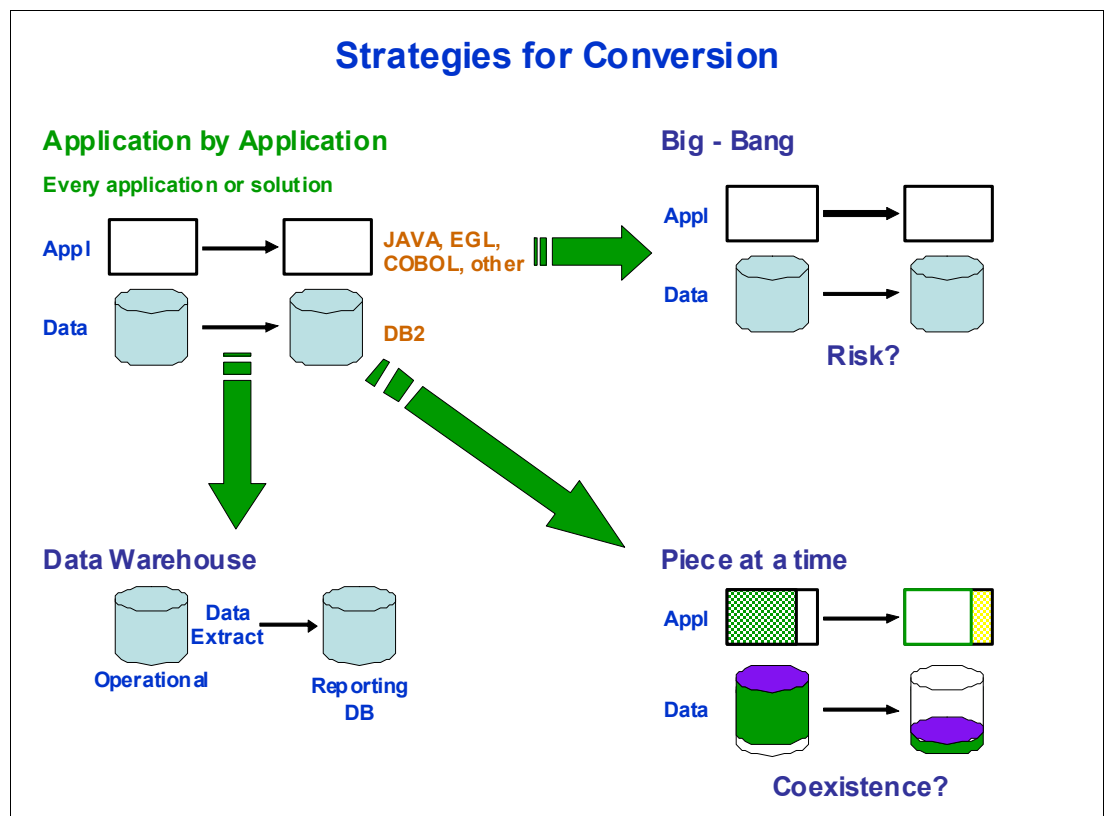


Figure 6-1 Conversion strategies

6.1.1 Big Bang strategy

In this strategy, all applications and data are moved to DB2 and are pushed live simultaneously. This strategy is useful when the client requires the benefits of DB2 and a single DBMS quickly across all applications. This strategy is most appropriate when speed is the overriding consideration.

Although this strategy is a high risk, the strategy features integrated data, which has the following advantages:

- ▶ Data serves more than one application.
- ▶ No coexistence problems with copies in more than one DBMS.
- ▶ DBMS procedures are not duplicated.
- ▶ All users concentrate on one DBMS DB2.

Although this strategy has its advantages, this strategy takes longer to benefit from DB2 because all conversions must be completed before cutover. This strategy also requires complex change management and it is unlikely that all systems are frozen while the details of the conversion are sorted out.

6.1.2 Loose coexistence strategy

Coexistence is the need for different database systems to communicate with each other, an issue that almost all organizations face at some point. Although the need for coexistence is temporary because the old database is eventually discontinued, coexistence becomes permanent because of a business requirement to keep the old system in place. This coexistence issue takes one or more of the following forms:

- ▶ The need to update two different databases from a single program.
- ▶ The need to synchronously update two databases in a single unit of work (usually for transactions).
- ▶ The need to pass data between programs asynchronously (usually for batch).
- ▶ The need to generate information warehouse-type data summaries that are used within another database.

This loose co-existence strategy usually involves a data warehouse or reporting system that consolidates information from different platforms for reporting purposes. This strategy is most effective when the management of the reporting process must improve or there are many reporting programs for the operational systems.

A copy of selected data is retained separately, which provides historical records. Instead of report programs, an extract, translate, and load process replicates and transforms data to a data warehouse. The report programs are then replaced with query or BI tools.

In this strategy, some data is kept multiple times, but operations continue as normal. Copies of the data are used as part of the conversion and are kept up-to-date by asynchronous update or periodic copy. The advantage of this configuration is that fewer coexistence problems occur. A flexible data warehouse has beneficial effects on many businesses, and is implemented by using a web portal. This strategy must be considered as an option that reduces risk during a migration project.

6.1.3 Piece-by-piece strategy

In this strategy, definable applications or parts of the applications and the data that the applications use are migrated to DB2 individually. The following assumptions are made when this strategy is considered:

- ▶ All systems or all pieces of a large application are converted to use DB2.
- ▶ Enhancements are not made to an application when it is converted.
- ▶ Future application enhancements are made in DB2.

This strategy is useful when the primary considerations of a conversion are to reduce risk, track projects, and gain experience as the work proceeds.

The primary complication of this strategy is that rarely is application data dedicated to one application and the effect on non-converted applications must be carefully considered.

The advantage of this strategy is that each application is treated separately and users gain expertise in the application as a result. Because of this advantage, this strategy realizes the benefits of DB2 faster for the first areas that are selected for conversion than the Big Bang strategy. This strategy also has less risk overall and work force of the business gains expertise over time. The costs of any tools necessary for the conversion are spread over the entire project.

The biggest disadvantage is this strategy is managing coexistence. If a migrated system must access data that is used by other systems, there are two possibilities to manage this coexistence:

- ▶ Two copies of the data must be kept with synchronous (or possibly asynchronous) updates.
- ▶ One copy of the data is kept and when that data is moved, all programs in all affected applications must be changed to access that data in the new environment. This change requires that some programs have access to old and new DBMSs together. When a series of small conversions are processed, many programs must be examined and changed several times. This strategy also requires that at least two sets of operational procedures must be maintained with the work force of the business split between the sets of procedures. Each conversion involves a frozen zone where, although migration takes place, users do not see any improvements.

6.1.4 Using transparency as a bridge strategy

The transparency strategy is the easiest to implement and presents the least risk. The programs are not affected, and transparency ensures functional equivalency. The disadvantages of the transparency tool strategy are the effect on performance and the reliance of a product as part of the application architecture. A transparency tool might be used with an application rewrite that eliminates any dependency on the transparency tool.

Customers with high-volume workloads use transparency tools on the z/OS platform to conduct table-by-table conversions and thus avoid a Big Bang result. In each instance, a pilot that used the transparency tool is conducted to ensure equivalent functionality and acceptable performance. This approach results in an increase in CPU usage for the online transactions and batch workloads that run under DB2 for z/OS. Ongoing performance and tuning of the application workloads resulted in an improvement, but increases of approximately 30% or higher in MIPS must be tolerated. The MIPS increase did not affect the response time or batch workload run times, which were equal or better than pre-migration.

Customers who use programs that access 20 - 30 database tables in one transaction cannot use the transparency tool. A rewrite of the programs with careful tuning of the SQL was required before the converted tables and programs were migrated to production. In these scenarios, more filtering of the data that is retrieved by the programs is required to improve performance.

Some of the performance challenges that result from using transparency tools were inherent in the data mapping approach and quality of the SQL. The transparency tool strategy often adds another column that is used as an index to each DB2 table (this column is not required in a normal DB2 data model). Another performance challenge is found in the fact that not all of the SQL generated by the transparency tool was static, and the tool generated dynamic SQL. The sequence of SQL calls also was not optimal. For example, a SELECT statement was issued and then a delete or update SQL statement was issued. In an optimal sequence, the operation is conducted with a delete or update statement without the SELECT statement.

For these reasons, customers often used multiple conversion strategies to eliminate the use of the transparency tool. IBM worked with a customer of a large logistics application that was running in multiple data centers to complete the DB2 conversion and eliminated all tool dependencies. The following methodology was used for a complete application rewrite:

- ▶ A customized in-house tool was developed to rewrite every DBMS call to a static SQL call. This configuration eliminated the use of the I/O layer that was generated by the transparency tool.
- ▶ The in-house tool was used with a manual rewrite approach for each program to improve the quality of the SQL usage. The tool that was written in-house increased the speed of the rewrite so that one programmer rewrites several programs in a day.
- ▶ Every database table was redesigned to remove the column and index that were imposed by the transparency tool. The DB2 tables were analyzed to keep indexes to a minimum, and this analysis frequently was conducted as data and programs were rewritten and migrated to the production environment. The transparency tool was used in the first pass. In the second pass, a rewrite of the application to improve performance and eliminate any tool dependency was completed.

The rewrite project was completed in approximately 18 months and resulted in an application that uses the same or less MIPS than the original DBMS application. The in-house tools were easier to design and build after the data was migrated to a DB2 for z/OS environment. The logistics customer also had staff available with DB2 for z/OS database and programming skills. The project began with insufficient DB2 performance tools. This deficit was discovered during the pilot phase and IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS (OMEGAMON PE) was installed to facilitate tuning the buffer pool other performance analyses.

6.1.5 Combining migration strategies

A substantial aspect of many applications is the reporting programs of the applications. It is important to make these reporting programs available soon. One option is to set up a data warehouse system before the conversion project to manage reporting. This configuration is implemented by converting a few existing report programs to use the data warehouse and provide extra reporting through a reporting tool. As a result of this change, each operational system is converted to DB2 with less effort because the number of converted programs decreased by 20% - 40%.

When the Big Bang strategy is not the best migration solution, some applications might be migrated by using the piece-by-piece strategy. A series of conversions are conducted after this migration in which several applications are migrated together with their common data. If

any of the applications need to be rewritten, different rewriting methods are used for different applications but basic migration strategies remain unchanged.

These migration strategies apply as well to a single application as to whole installations. Successful conversions use a single strategy or a combination of strategies to convert a database. For example, a transparency tool was used with other conversion strategies to support a piece-by-piece conversion when a Big Bang conversion was thought to be the only option. In this scenario, the risk of the initial conversion was reduced, and coexistence was resolved without a Big Bang or a replication solution to synchronize data.

Other customers followed a similar strategy, but choose to use an automated conversion tool that generates I/O modules to support the initial conversion to DB2 for z/OS. An assessment of the programs with the highest number of executions is used to determine which programs to manually rewrite. The rewrite strategy is for a smaller percentage of the application, for example, an 80/20 rule (80% conversion by using tools and 20% manual rewrite) to improve performance. Every application is different, so the percentages used in the 80/20 rule might be too high or too low depending on the customer environment and the workload characteristics.

6.1.6 Comparing strategies

Application migration projects are complex undertakings that require organizations to develop a comprehensive plan encompassing people, processes, and technology. There are many migration strategies that use different tools. The strategies also have different levels of cost and risk with varying advantages and disadvantages. Identifying which methods are optimal for your environment is important.

The advantages and disadvantages of the Big Bang, Piece-by-piece, Loose coexistence or data warehouse and transparency strategies are compared in Table 6-1.

Table 6-1 Pros and cons of the migration solutions

Strategy	Advantages	Disadvantages
Big bang	Quickest conversion	High risk and backout plan needed
	No coexistence issues	DB2 benefits are realized only after full test or production implementation
	Strategic direction	Update test and production while testing
Piece-by-piece	Low risk	Dual platforms require coexistence solutions
	Immediate DB2 benefits	
Loose coexistence or data warehouse	Low risk	High-cost application development
	Near real-time data with IBM replication strategies	Resources and cost to maintain two database platforms

Strategy	Advantages	Disadvantages
Transparency combined with other strategy	Quickest solution	Not available for all platforms
	Easiest – Convert by table No affect on the program	Performance and product license fees
		High cost if target is to convert off the transparency

6.2 Automated conversion tool case study

Customers chose to convert to DB2 for z/OS most often for the following reasons:

- ▶ Users or external customers demand access to relational data
- ▶ Customer must simplify the number of supported database platforms
- ▶ Lack of available customer staff skilled in DBMS
- ▶ Application agility is needed to meet changing business and technical demands requires frequent programming and data model enhancements

Next we examine a real customer case story of an automated conversion from IDMS to DB2 for z/OS. In this case study, we focus on the following aspects of the conversion:

- ▶ The reason why the customer chose to convert.
- ▶ The process that the customer used for the conversion.
- ▶ The issues that the customer had to manage during the conversion.
- ▶ The results that the customer achieved throughout the process.
- ▶ The lessons that the customer learned from the conversion.

6.2.1 Case study background

The customer is a large government institution. The Integrated Database Management System (IDMS) application of the customer was converted from VSAM in the 1980s. As customer base increased and data access requirements expanded and became more complex, changing the application and supporting database to meet regulatory compliance became more challenging. Changes often included the use of the same field for multiple purposes or the use of FILLER fields to hold data. In addition, support for the installed version of the IDMS was scheduled to end within a few years, and the customer did not want to pay for DB2 and IDMS license and support charges.

In late 2007, the customer initiated a project to convert from IDMS to DB2 for z/OS and to evaluate automation tool vendors and conversion support services. The following objectives were identified for the conversion project:

- ▶ Improve responsiveness to business needs:
 - Regulatory compliance
 - Changing business requirements
- ▶ Improve quality:
 - Reduce technical complexity and risk
 - Reduce rework and associated costs
 - Improve data integrity

- ▶ Increase IT organization capacity:
 - Improve testing processes
 - Retrain staff on newer technology
 - Increase available talent pool

The sheer size of the conversion effort dictated that an automation conversion approach for the project must be found. The following statistics of the project reinforced the need for an automated approach:

- ▶ Approximately 7 million lines of code
- ▶ Over 1,400 batch programs
- ▶ Over 400 CICS programs
- ▶ Almost 4 terabytes of data

The customer issued a request for information (RFI) with conversion questions to IBM and other customers that converted from IDMS to DB2 for z/OS. The DB2 Migration Project Office received the RFI and responded to the conversion questions related to IDMS conversion experiences. Additional information about the use and benefits of automated conversion tools also was sent with the RFI to assist with the initial study.

After the customer selected their conversion vendor and IBM services were assigned to the project, the conversion project started. The following sections provide highlights conversion process steps. These sections are not meant to be a complete documentation of all of the steps that are required for all automated conversions. Instead, the following sections document the major steps in the process only for one customer.

6.2.2 Identifying and conducting a pilot project

A self-contained application was identified by the customer to be part of the pilot and a contract to conduct a pilot project was agreed to between the customer and the conversion vendor. The following goals of the pilot project were identified:

- ▶ Demonstrate the capabilities of the conversion tool.
- ▶ Verify the compatibility of the tool with the environment of the customer.
- ▶ Identify problem areas in preparation for the larger conversion effort.

The application was converted successfully. When the results of the conversion were reviewed, the users, developers, and programmers gained experience in and learned lessons from the following areas:

- ▶ Familiarization with the conversion process and the methodology that was used by the vendor
- ▶ Required customization of the current change management process for vendor deliverables
- ▶ Required understanding of the CICS error handling differences between IDMS and DB2
- ▶ Maintained separate conversion and test environments for ongoing business changes
- ▶ Vendor system access simplified problem analysis and resolution
- ▶ Customer-driven issue analysis and tracking is needed

This experience and knowledge was used throughout the subsequent steps of the larger conversion effort. The remaining steps are for the conversion of the production application.

6.2.3 Automated assessment

The assessment process involved a full analysis of all application and database components for the application that was converted. The intent of this step was to estimate the scope of the conversion. This process was conducted iteratively because some components were identified as missing whenever the assessment was run. The components were missing because some source code or jobs that were found in non-standard libraries were not provided to the conversion vendor.

After the assessment was completed, an estimate of the scope of the conversion was provided. Reports identified the components that must be converted. A project plan was developed that detailed the steps of the project and the process to collect the components targeted for conversion began.

6.2.4 Collecting, converting, and customizing the application and database

The conversion vendor ran the following jobs to collect the IDMS components necessary for the conversion:

- ▶ Source code
- ▶ Copybooks
- ▶ JCL
- ▶ CICS maps
- ▶ Database definitions
- ▶ Data

All IDMS components were processed by the automated conversion tool to produce the equivalent DB2 components. The new components were then sent to the customer.

At the beginning of this production step, certain business rules were predefined as input to the conversion process. These rules allowed for renaming fields to conform to DB2 standards and for defining the manner in which differences in IDMS and DB2 data types were managed. For example, DB2 allows for the storage of date data as a DATE data type, with requirements that data stored in columns that are defined with that data type must adhere to valid DATE data. However, not all of the fields that are defined as dates in IDMS adhered to the strict DB2 DATE rules, so decisions were made to manage the conversion of those fields. The goal was to minimize the number of changes to the applications. As a result, the following types of date columns were defined:

- ▶ DATE data type for data that is enforced as a valid date
- ▶ INTEGER data type for data that is not enforced as a valid date, but always was numeric
- ▶ CHAR data type for all other dates

Some customization was conducted to determine which buffer pools to use and to partition certain tables. After the rules were defined and the converted applications and database were returned, the DB2 objects were created. The applications were prepared to run under DB2 and testing began.

6.2.5 Application and database testing

In the previous step, the converted applications and the converted data were provided to the customer. In this step, application and data conversion testing were conducted simultaneously. For these tests, small amounts of data were loaded into unit test environments to allow application testers to begin their analysis. At the same time, tests of DB2 LOAD utilities were run on the completed set of data to identify any data conversion errors or data definition language (DDL) errors.

During the testing phases, all defects were logged in to a defect tracking system, with each defect identified by a defect number. All subsequent communications with the vendor that concerned any defect referenced the defect number.

The testing continued through multiple iterations. This testing phase was the longest phase of the project. Testing progressed through unit test, system test, and performance test phases. Regular weekly meetings were held with the conversion vendor to review the testing results and any defects that were open.

6.2.6 Performance testing

Throughout the testing phases, applications that were potential performance bottlenecks were identified. Individual teams were organized to address online performance issues and batch performance issues.

The online performance testing identified a mix of transactions that were deemed critical for online performance. Automated testing software was used to capture the transaction mix and rerun the mix as needed after program or database configuration changes were made.

The batch testing identified the longest-running batch jobs within IDMS. Equivalent batch jobs that were run within DB2 also were identified and the difference between the two sets of batch jobs was measured. Decisions then were made as to which programs must be rewritten and which jobs must be run against multiple partitions in parallel.

The applications that were targeted for rewriting often were applications that accessed many tables. The converted applications conducted programmatic joins because of the nature of the automated conversion. In many instances, the programmatic joins were replaced by SQL joins. In addition, there were some instances in which batch jobs were completed during an area sweep of the IDMS database. The batch jobs then used program logic to determine which rows qualified for the conditions that were specified in the program. These programs are run as table space scans in DB2. Logic added in the WHERE clause to filter out unnecessary rows earlier in the performance testing process diminished the effect on the CPU of many of the performance problem batch jobs.

6.2.7 Implementation tests

A number of implementation dry run tests were conducted to validate the production implementation process. The tests also provided an estimate of the amount of time that is needed to complete the production implementation. The following tasks were included in the tests:

- ▶ Running the entire data extract and load process
- ▶ Running the processes to switch from the IDMS application load libraries to the DB2 application load libraries
- ▶ Testing the new code with the converted data
- ▶ Backing out the load libraries to point the application back to the IDMS database

These implementation tests required considerable planning. The tests included all of the conversion process personnel, including application developers, database administrators, CICS administrators, testers, users, operators, security administrators, change management staff, and project management staff.

The lessons that were learned from each test were documented and changes to the process were applied to subsequent steps. These changes minimized the chance for errors during production implementation.

6.2.8 Production implementation

The production implementation was completed during a holiday weekend. The same steps that were conducted during the implementation tests also were conducted in the production environment. Initial validation testing was conducted to ensure that there were no issues and no need to roll back to the IDMS version of the applications and data. After the validation testing was completed and a GO decision was made, batch processing began.

Teams were identified to monitor both the batch and online workload during the first few weeks of the processing. The conversion vendor also provided on-site support for most of the first week after the implementation was complete. The few problems that were found were addressed quickly. Most problems occurred when JCL was translated between the test environment and the production environment.

6.2.9 Post-conversion tuning

Performance data that was captured for the first few online and batch cycles was reviewed to identify areas of concern. The online transactions and batch jobs that were completed within the accepted Service Level Agreements (SLAs) for elapsed time. However, the CPU cost for most jobs was considerably higher than the cost under IDMS. The conversion was a like-to-like conversion, therefore, an increase in the CPU cost was expected because the IDMS-like access of the DB2 data.

Changes were made to take advantage of SQL functionality by replacing programmatic joins with SQL joins and by including filtering logic in the SQL instead of in the program. For more information about techniques that are used to tune converted applications, see Chapter 11, “Application and SQL tuning” on page 175.

A team that consisted of staff members from different areas in the company met regularly to track CPU costs and performance tuning efforts. Applications that must be changed were identified and prioritized. Changes were made to the applications, the applications were tested, and performance was measured against the performance that was documented at the production cutover time. Decisions were made whether to completely rewrite a program or make minor changes to build a driving cursor in the application. This cursor reduced the amount of row filtering logic that was completed in the program code. The tuning effort continued, with the programs targeted for change shifting in priority as updated performance measurements became available.

6.2.10 Relational redesign

After the conversion was complete and was stable in production for a few months, an effort began to determine how the application and database might take advantage of true relational database capabilities. This analysis step is a long-term project that involves the following tasks:

- ▶ Reduce dependency on generated I/O modules
- ▶ Take advantage of SQL capabilities
- ▶ Remove any IDMS-like constructs from the database design

The final goal of the redesign is to have an application and database that are truly relational. The time and effort to incorporate this change as part of the first phase unnecessarily lengthens the project. The effort to make these changes also increases the dependency on IDMS beyond the end of the support date for the installed version of IDMS. Therefore, a decision was made to use an automated conversion solution. The application and database must be reviewed after the conversion, however, and plans must be made for a true relational implementation.

6.3 IBM consultancy contact information

The IBM DB2 Migration Project Office works for free with many customers to educate customers about the conversion process. The role of the DB2 Migration Project Office is to enable customers to cost-effectively migrate to the DB2 family of products from any competitive or non-relational database product. Migrations are large projects that require a significant customer commitment and investment. Successful projects might require the cooperative efforts of many customer departments and the assistance of many services organizations from outside IBM and the customer. IBM helps customers develop a plan that integrates all of the resources that are required for a successful migration.

The DB2 Migration Project Office works with conversion vendor selected by the customer and other IBM service organizations that are needed. The conversion vendor tools use software technology that automates much of the migration process. The other IBM organizations possess the skills that are required to complete a successful project team (for example, systems integration, project management, DB2 setup, performance and tuning, and testing). Our objective is to build a team that successfully conducts your migration project.

For more information about the DB2 Migration Project Office, and for the contact information of pre-sales support by geographic region, see this website:

<http://www.ibm.com/software/solutions/softwaremigration/dbmigteam.html>

IBM services are available for any migration phase. If you are planning a migration project, contact the following IBM office in your area:

- ▶ North America and Latin America: db2mig@us.ibm.com
- ▶ UK, Europe, Middle East, and Africa: emeadbct@uk.ibm.com
- ▶ Japan, India, and Asia Pacific: dungj@hkl.ibm.com

Tool-assisted conversions

In the following chapters, we describe the conversion solutions used in the market, the challenges and requirements to convert from CA IDMS and Adabas to DB2 for z/OS, and the tools used during the conversion process:

- ▶ Chapter 7, “Conversion solutions” on page 73
- ▶ Chapter 8, “Converting CA IDMS to DB2 for z/OS” on page 83
- ▶ Chapter 9, “Converting Adabas to DB2 for z/OS” on page 111



Conversion solutions

This chapter describes the solutions customers use to realize the best results of their conversion to DB2 for z/OS projects. By using these solutions, customers reduce the time it takes to complete the conversion and the total cost of the project.

This chapter includes the following topics:

- ▶ Automation
- ▶ Engaging the conversion tool vendors
- ▶ Post project lessons learned
- ▶ IBM InfoSphere solutions assist conversion process

7.1 Automation

There are many tools available that help reduce the costs and the amount of time that is needed to complete a conversion project. For many customers, converting millions of lines of code is a huge project in which money is wasted if an automated conversion process is not used.

The recoding phase is the largest phase of a conversion project. Figure 7-1 shows the amount of time that was spent on each phase of a project.

Important: Project timing varies as the timing is dependent on the customer, technology, challenges, and so on.

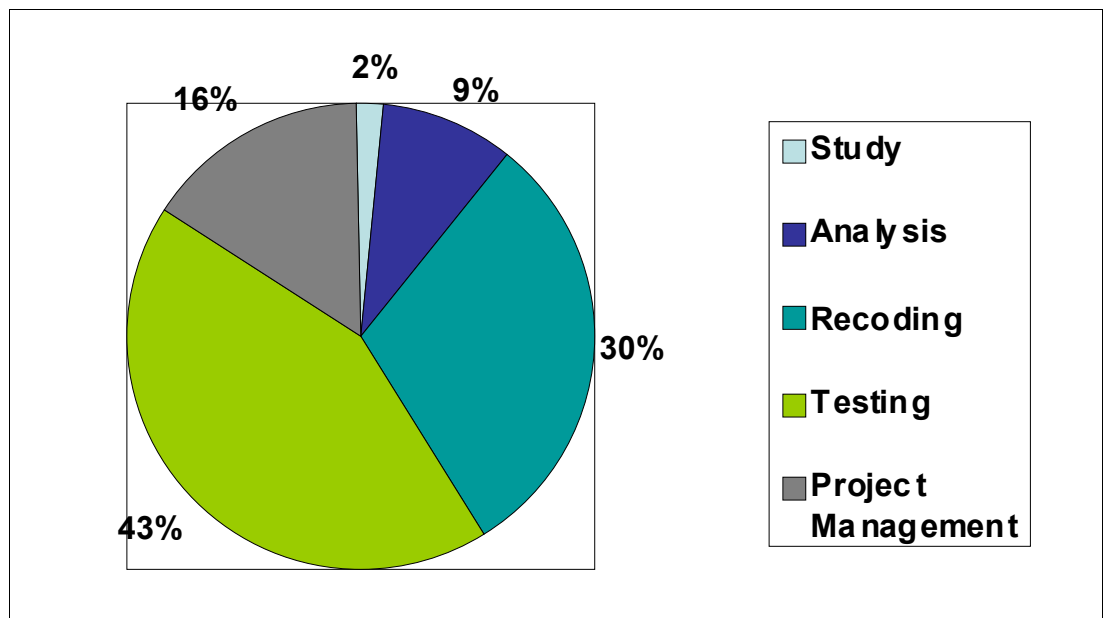


Figure 7-1 Time of each project phase

For this project, even if a code conversion tool was installed, any savings on conversion costs that are realized are proportionate to 30% of the overall costs. An automatic conversion tool might affect the cost of the project because the testing phase includes the cost to fix any issues or problem that are found.

Estimating the amount of time necessary to complete the recoding process helps to evaluate the amount of time and money might be saved by using an automation tool.

The conversion tool solution focuses on translating the application data and must be done with minimal effect on the programs.

Each tool provides different methods of and approaches to the conversion process. The answers to the following questions help evaluate each tool:

- ▶ Which stage of the migration process is supported?
- ▶ Which method of the migration process is supported?
- ▶ What percentage of automation is expected from this tool?
- ▶ Does the tool correct error codes?

- Is possible to install the tool on the environment of the customer?
- What are the possible savings realized by using the tool versus learning the tool and the cost of the tool?
- Is there residual value that might be used later, either to maintain or upgrade the programs?

The time and cost savings that are realized from an automatic program conversion are substantial.

By using the figures from the DBMS Conversion Guide series, you see that a medium-sized program with 1000 lines of COBOL code that use IDMS takes approximately one week to convert. A conversion tool produces an SQL version of the program in a few minutes.

Table 7-1 shows the advantages of an automated conversion over a manual conversion.

Table 7-1 Advantages of automated conversion over manual

Issue	Manual	Automated
Timing	Slow	Fast
Quality	Prone to human errors, numerous debugging cycles needed	No bugs programs are generated, same no bugs programs are used
Consistency	Many developers write different code	Code is uniform because it is kept in same standards
Maintenance	More time is needed	Groups of code are updated during the conversion
Cross-reference	Each program must be analyzed	Performed automatically

7.1.1 Application conversion challenges

Although automated tools are robust, the tools often cannot address the following potential problems that might occur during a conversion project:

- Lost source data
- Program error codes
- Logical errors
- Decrease in performance

There are instances in which source code is lost and the customer must rewrite the code or hire a vendor that generates the source code from the object. In other instances, the company developers must rewrite the code, which adds to the length and cost of the conversion project.

Although the conversion tool converts 100% of the code in most projects, only 80% of the programs worked immediately. Tool vendors want a better outcome, so the vendors determine the percentage of the client code that is converted successfully. However, no vendor claims that their tool provides a 100% conversion rate. When only code re-engineering is necessary, the process is done manually. Such re-engineering often is completed within a single business day as compared to a week when the entire program is converted manually.

7.1.2 Data conversion

Previously, we focused on the importance of the language conversion process, which is often the most expensive and complex part of a project. However, the database conversion process in which the pre-relational database is converted to run the inside DB2 for z/OS also is important.

Customers might choose to convert the database manually, but the following logistical issues must be addressed:

- ▶ Manual conversion is time-consuming
- ▶ Errors during data modeling
- ▶ Exposure of privileged data
- ▶ Loss of data and logical integrity
- ▶ Need to create data extract and load process by using the logical model
- ▶ Must use the correct structure of the DB2 for z/OS
- ▶ Potential performance decrease
- ▶ Lack of user skills with DB2 for z/OS

The automation tool checks for and address these issues automatically. The tool also maps the data structure from the pre-relational model to the relational model and gives the necessary flexibility to the customer to improve the new physical model, DDL, and generated objects.

For more information: For more information about database conversions from pre-relational technology, see Chapter 8, “Converting CA IDMS to DB2 for z/OS” on page 83, and Chapter 9, “Converting Adabas to DB2 for z/OS” on page 111.

7.2 Engaging the conversion tool vendors

Customers choose from a number of vendors that provide automated conversion solutions.

Before or during stage one of the conversion process, information must be obtained from several tool vendors about all of the areas of the conversion process in which tools might help. During stage two, vendors that have a suitable tool might demonstrate their product. The customer tests the tool to assess how the use of the tool might affect the project.

Although not necessary, it is advantageous to work with a source code conversion tool vendor during stage one. Such a vendor is best suited to help determine what percentage of the code conversion process is eligible for automation.

During a week-long stage one phase, the percentage of programs that feature DBMS calls or the average numbers of such calls per module cannot be determined manually. However, many source code vendors use code that assesses thousands of programs quickly and provides statistics on the number of DBMS calls, the complexity of the calls, and how often the calls are used. This information is used to assess the effect of the tool on the project. Knowing the extent of this effect allows developers to accurately assess early in the project lifecycle the cost of using the tool.

An accurate assessment of the effect of using the tool can be reached by working with a DB2 design and conversion consultant with the tool vendor. The tool vendor assesses the effect of the tool on the project and the DB2 conversion consultant ensures that consideration is given to what is really wanted.

Without the assistance of an experienced DB2 consultant, consulting with a tool vendor must be delayed until all tool requirements are determined by using the stage one process that are described in this book. An evaluation of any assistance available from a tool vendor is made, the savings are estimated, and the costs of improving the output from the tool is added to the estimate of the conversion project.

Automated conversion: In the past, tools were not available to convert most of code in a project. Today, tools automatically convert 99.99% of the code.

7.2.1 Qualifying vendor tools

Before a conversion tool is used, the effect the tool might have on the project must be considered. Table 7-2 lists several important factors that help qualify a tool.

Table 7-2 Considerations to qualify vendor tools

Factor	Yes or No
Proprietary solution	
Support new DB2 10 for z/OS features	
Installation on customer environment	
Allows DBA changes to data modeling	
Require new software licenses purchases	
Solution has legal problems about the use of any proprietary code	
Transparency solution	
Provide assessments to estimate price of the project	
Provide supports after the conversion	
References of customers in real production environment	

In addition to considering the factors listed in Table 7-2, the following questions be must asked to determine whether the tool is the best solution for the database conversion:

- ▶ Must the data layout move 1:1?
- ▶ Does the tool support:
 - Only data translation
 - Redefined fields that are defined as extra columns in the same table
 - Redefined fields that are defined as extra columns in another table
 - Periodic fields that are defined as columns in one table
 - Periodic fields that are defined as rows in a table with data as values in a single column and another column
 - Repeating group fields that are defined as extra columns, or as values in a single column
 - A requirement to change date and time fields to DB2 format

- ▶ Does the tool generate an SQL join if a segment or record is split across two DB2 tables?
- ▶ Does the tool update only the columns in which a few fields are changed?
- ▶ If the source code makes multiple calls to obtain all the fields in a redefined record, does the tool substitute those calls with a single call to DB2 to fetch the required columns?
- ▶ Is a primary key added if the data does not have such a key?

7.3 Post project lessons learned

The following “lessons learned” from the conversion project help identify what might be done differently for the next conversion project:

- ▶ Test batch applications earlier in the project, not at the end of the project.
- ▶ Factor in project risk, especially the time that is lost because of unforeseen circumstances (weather, technical issues, and so on).
- ▶ Identify application components (if any) that require re-engineering and ensure that the solutions are tested.
- ▶ Ensure adequate performance monitoring and that test tools are available to support the project.
- ▶ Understand how security works and how security affects performance.
- ▶ Always test in the environment in which you are deploying the conversion.
- ▶ Use project plans based on realistic expectations and staff resource availability.
- ▶ Identify conversion application enhancements that increase the cost of the project.

Although the vendor completed unit testing off-site, testing also must be conducted at the customer site. The customer experienced the following learning curve issues during the automated conversion testing process:

- ▶ The customer expected that the converted code was free of defects. This unrealistic expectation was a misconception.
- ▶ The customer did not understand that global errors might and do occur early in the project. Although errors might be numerous, errors are not a reflection of the quality of the delivered code. Instead, errors are a reflection of the project lifecycle. The errors that were found resulted in a four-week pullback by the subcontractor to test the code at a deeper level.
- ▶ The customers were unaware that subcontractors did not understand the business logic of the application. As a result, the customer experienced the following problems:
 - Because there was not enough information was provided to the subcontractor about how to correct defects, project leaders realized during the late phases of the project that more information must be included in the Test Director descriptions and attachments.
 - Project leaders were unaware that subcontractors did not always understand the data or alternate navigation paths. This knowledge required the assistance of subject matter experts.
 - Developers did not understand how the corrections were made to the code. As a result, manual corrections were not made. However, corrections were made to the conversion tool rules.

7.4 IBM InfoSphere solutions assist conversion process

Regardless of the methodology that is chosen to conduct a conversion, IBM has available a set of solutions that help customers replicate their data from pre-relational databases to DB2 for z/OS during a coexistence project. Solutions also are available to transform the pre-relational data for use by SQL programs from DB2 for z/OS and to conduct data cleaning and transformations to data during the conversion process.

- *Classic data replication*

By using this solution, the customer replicates data from the pre-relational database to DB2 for z/OS. This solution helps in the conversion process during coexistence. Classic data replication also is used to test the applications during coexistence. This replication solution is conducted only in a unidirectional manner.

- *Classic data federation*

With this solution, users choose to develop applications on new languages by using SQL and accessing the pre-relational data. This option is useful during a conversion project in which coexistence is implemented.

- *Data transformation*

Because a data stage provides so much power to data that is cleaned and transformed, some customers might need to provide an extract, transform, load (ETL) process before the data is applied to DB2 for z/OS. In this solution, the data stage receives this data from classic data replication through the message queues (MQ) or by using the classic federation option.

Figure 7-2 shows an overview of how these solutions might be combined during the period of coexistence in which the applications access data from the pre-relational databases and DB2 for z/OS.

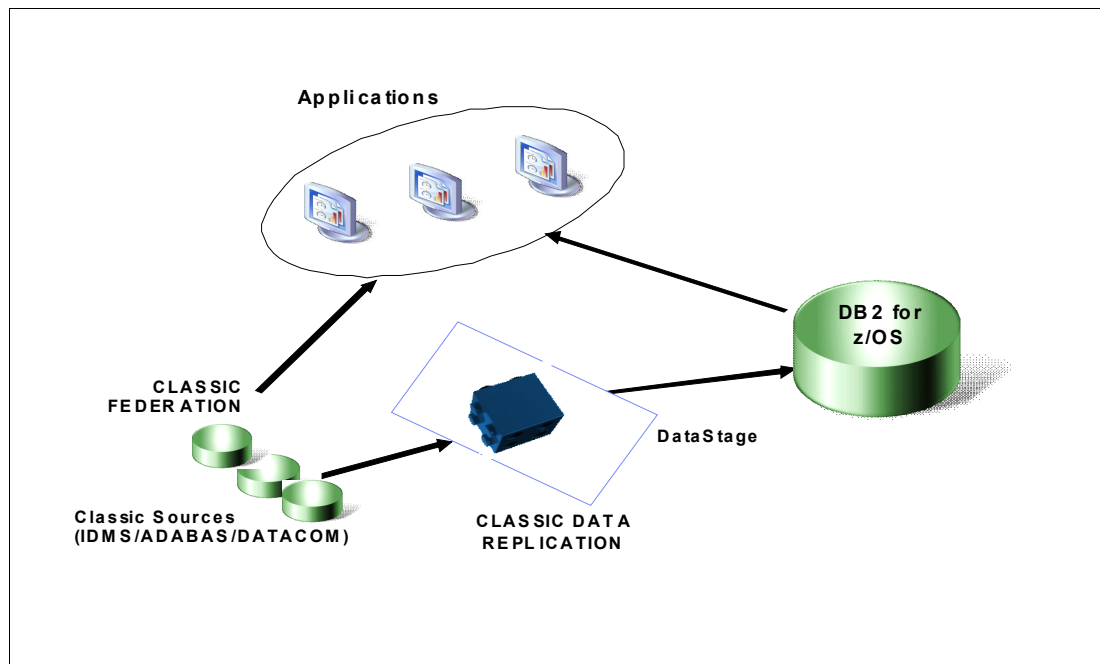


Figure 7-2 Overview of coexistence solutions by using IBM InfoSphere® solutions

Important: CA IDMS and Adabas are classic sources for replication and federation.

The following sections provide an overview of each of these solutions.

7.4.1 Data replication

IBM has powerful replication products available to replicate DB2 for z/OS as the source database. For the pre-relational databases, the replication products call the data sources as classic data.

This replication solution delivers critical mainframe data to local and remote relational replicas as needed. The solution also supports SQL-driven access and near real-time changed data feeds to InfoSphere Replication Server over WebSphere MQ from VSAM, IMS, Computer Associates CA IDMS, and Software AG Adabas data sources. Data is formatted specifically for use by the queue-replication capabilities of InfoSphere Replication Server, which enables the targeting of DB2 for z/OS.

Figure 7-3 shows the basic replication architecture in which the changes made to the classic data sources are captured by using the logs of the sources. The changes are made without affecting the applications and are sent to WebSphere MQ which delivers the data as message queues that are applied on DB2 for z/OS.

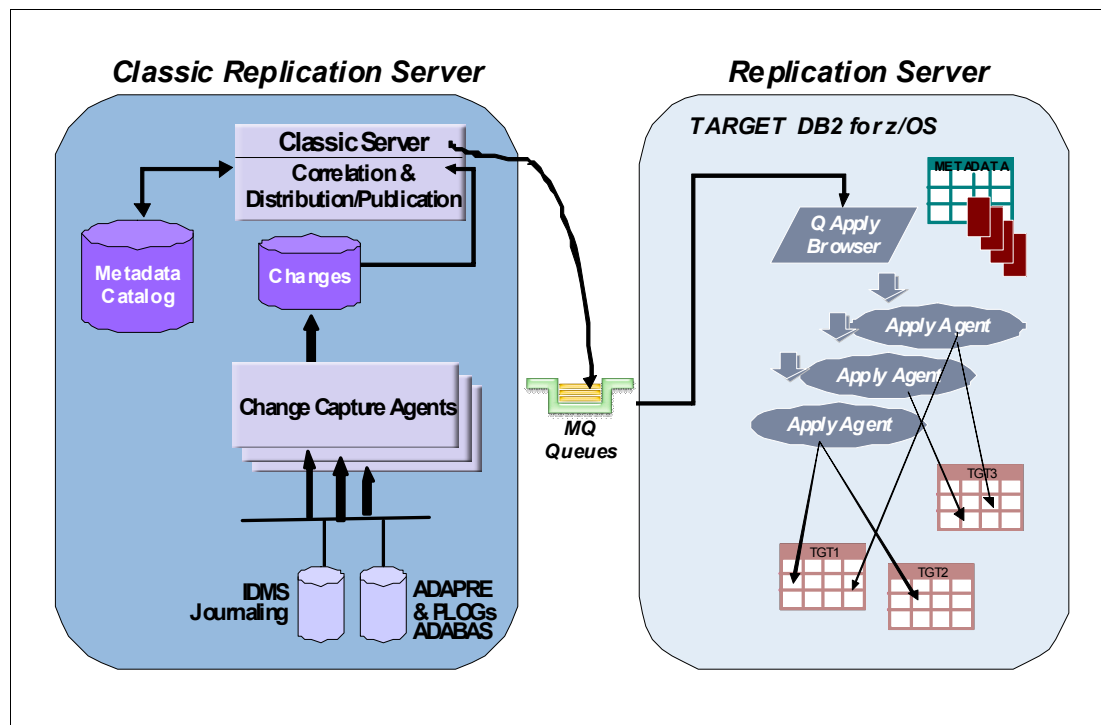


Figure 7-3 Data replication architecture

7.4.2 Data federation

IBM InfoSphere Classic Federation Server for z/OS offers the following features:

- ▶ Direct virtualized SQL access to mainframe data sources.
- ▶ Accesses data that is stored in VSAM, IMS, CA IDMS, CA Datacom, Software AG Adabas, and DB2 for z/OS databases by dynamically translating JDBC and ODBC SQL statements into native read/write application programming interfaces (APIs). The server is driven by metadata that contains a mapping of physical databases and files to logical relational tables. Mainframe data is shown as a single relational database and supports SELECT, INSERT, UPDATE, and DELETE SQL statements and stored procedure calls.
- ▶ Simplifies managing operational platform and performance optimization processes by providing an interactive view of the individual configuration settings of the federation members and a means for temporarily or permanently changing them.
- ▶ Dynamic integration of z/OS data for ETL processing with no need to unload or File Transfer Protocol (FTP) files to the ETL (InfoSphere IBM DataStage®) platform.

7.4.3 Data transformation

IBM InfoSphere DataStage integrates data on demand across many systems through a high-performance parallel framework, extended metadata management, and enterprise connectivity.

DataStage also supports the collection, integration, and transformation of large volumes of data, with data structures that range from simple to highly complex. Through the support of real-time data integration, developers maximize their speed, flexibility, and effectiveness in building, deploying, updating, and managing the data integration infrastructure. DataStage also completes the connectivity between any data source and any application.



Converting CA IDMS to DB2 for z/OS

This chapter describes the process to convert from a Computer Associates (CA) Integrated Database Management System (IDMS) database to DB2 for z/OS. The chapter describes the requirements and challenges of this conversion and uses a real-world scenario that is provided by IBM Business Partner ATERAS who conducted these conversions.

This chapter includes the following sections:

- ▶ Performing a CA IDMS conversion
- ▶ CA IDMS requirements
- ▶ DB-Shuttle by ATERAS conversion solution
- ▶ DB-Shuttle conversion process

8.1 Performing a CA IDMS conversion

IDMS was created in the 1970s. Many businesses worldwide continue to run applications that are powered by this pre-relational database. Companies slowly migrated away from this platform and replaced it with a relational database solution. The remaining IDMS applications used today are mission-critical and impractical to rewrite or are unavailable because they are part of a packaged software solution replacement.

The goals that are most frequently associated with converting from IDMS include the mandate to lower IT costs and increase the ability of IT resources to meet the changing needs of the business. The conversion solution meets each one of these goals and preserves the intellectual property that was built into these applications. The solution also is deployed without any noticeable effect on the users or disruption to the business.

When customers start a conversion process from CA IDMS to DB2 for z/OS, they want the most open database from other applications and to use business analytics, service-oriented architecture (SOA), and other market direction for databases.

CA IDMS customers also are willing to use all of the features and advantages of the DB2 for z/OS relational database, as described in Chapter 2, “Converting to DB2 for z/OS” on page 7.

8.1.1 Re-engineering and rewriting the application

Re-engineering and rewriting the application is costly and takes time, often forcing CA customers into another licensing term. Customers often decide that there is no compelling reason to rewrite these applications because the applications are performing reliably and the users are satisfied with the applications. Although rewriting the application results in a more basic mode implementation of the system in a DB2 environment, the time, costs, and risks that are associated with a rewrite (especially for applications of this size and complexity) are too great to offset the other benefits that might be realized.

8.1.2 Purchasing a software package

Purchasing a software package is costly and because the IDMS applications work adequately, this option is difficult to justify unless the option is part of a larger requirement. Although packages might provide an excellent, cost-effective alternative, certain system areas are prone to require too much customization and, therefore, become more like a rewrite project in terms of cost and risk. Packaged solutions might be implemented more quickly and with slightly lower risk than rewriting a system, when all of the associated costs (customization, testing, training, conversion, and business impact) are considered.

8.1.3 Converting existing systems

Converting IDMS has the advantage of migrating the customer to a modern software environment that features full functionality without reinventing the business logic of the systems. This option is manageable in terms of cost and the time to complete. Although this option does not increase the functionality of the systems, this option does position the system better for the future, particularly for web-enabling the applications and realizing all of the benefits of a relational database that are used for the same reasons that make relational technology the standard used today.

8.2 CA IDMS requirements

When we think about this conversion, we might treat each CA IDMS record as a relational DB2 table, but treating records in this manner is not possible. A network database might be normalized in some instances, but the database is not normalized in most instances. To achieve normalization in some instances, it is necessary to split records into multiple tables or join multiple records into a single table. There also are some instances in which the record is discarded without being converted. A relational system allows users to interrogate any column by name that is based on the value of the column. However, this functionality is not possible in some CA IDMS designs.

There are several basic steps to convert CA IDMS data to DB2 for z/OS:

1. Verify that each CA IDMS record is in third normal form. Each record must be normalized if it is not already.
2. Create a DB2 table for each resulting CA IDMS record.
3. Translate each CA IDMS data element into a DB2 column.
4. Identify a primary key for all tables that represent former CA IDMS-owner records.
5. For each table, select one foreign key per CA IDMS member set.

To convert pre-relation databases to DB2 for z/OS, a customer chooses from different methodologies for CA IDMS to DB2 for z/OS, as described in Part 2, “Migration methodology” on page 23.

Customers also choose the conversion mode. By using this mode, the customer converts all languages and the database. The customer also migrates their COBOL/Application Development System Online (ADSO) to COBOL CICS and changes the COBOL BATCH to manage DB2 for z/OS calls.

Most organizations that used CA IDMS as their primary database management system are finding that some of their CA IDMS application conversions are one-for-one, others require minimal enhancements during conversion, and still others are slated for replacement by purchased packages.

CA IDMS conversion often includes the following primary goals:

- ▶ Full functionality
Ensure that the relational application mirrors the relationships, ordering, and management of CA IDMS data in the new relational database.
- ▶ Full flexibility
Provide a means to modify or create relational table formats, column definitions, and table relationships that are not predefined within the CA IDMS schema.
- ▶ Low risk
Ensure that the existing CA IDMS applications function exactly as they did in the CA IDMS environment. The new features defined by the relational database also must be available to new applications. The goal in converting CA IDMS databases and applications to DB2 for z/OS is not always to create a one-to-one conversion. The ability to guarantee existing functionality and flexibility previously was not available.

8.2.1 CA IDMS key conversion factors

We list here the tasks that are critical for the assessment of the IDMS conversion.

CA IDMS inventory

Estimating the workloads, cost, and schedules, is important so it is necessary to have a thorough inventory of the CA IDMS data and application. The customer often has an Integrated Data Dictionary (IDD) that includes full reports that provide the customer with all of the information they need. The CA IDMS Data Structure Diagram also shows all of the documented physical designs.

Data relationship

Correct and complete data conversion is the basis of any successful relational conversion. The resulting database structure is critical to future processing. Whether you are performing a CA IDMS conversion, your ability to create the data relationships and formats that drive your applications in the future is critical.

Designing the database tables involves mapping CA IDMS network records to DB2 tables. The customer must be concerned about the following CA IDMS issues:

- ▶ Normalization:
 - Duplicates allowed
 - Repeating groups
 - Redefines
 - Redundancy
- ▶ CA IDMS set analysis
- ▶ Referential Integrity:
 - Primary key identification
 - Foreign key identification
- ▶ Ordering fields
- ▶ Multimember sets
- ▶ Access paths
- ▶ Inconsistent data formats

Performance

The most effective way to maximize database performance is to retain the CA IDMS schema as the following benefits of DB2 for z/OS are realized:

- ▶ Convert group fields into a single column
- ▶ Avoid retrieving all columns; DB2 specifies the column
- ▶ Avoid updating all columns; DB2 logs only changed items
- ▶ Close cursors correctly in the program

For more information about performance issues after the migration, see Part 4, “Post-conversion tasks” on page 153.

Applications

The existing CA IDMS applications expect the data to be available in specific formats. They expect to retrieve the data by using the access paths that are in place. They expect to reference the group levels, redefines clauses, and occurs clauses that are built into the CA IDMS database structure. When these applications are converted to the newer languages, the new applications have the same expectation for the data access.

In addition to the changes required to implement DB2 SQL, logic changes might exist that are the result of the DB2 table design phase.

The following application changes result from data changes that are made to resolve the differences between a network database and a relational database:

- ▶ Primary and foreign key replacing links
- ▶ Normalization affect:
 - Record combination or splitting
 - Repeating groups
 - Record to multiple tables
- ▶ Translating group level elements
- ▶ Referential integrity update rules

Complete the following tasks:

- ▶ Convert each CA IDMS program to a corresponding DB2 program wherever possible.
- ▶ Do not include new functions.
- ▶ Determine naming conventions.
- ▶ Confirm which functions cannot be converted.

The following elements of the application dynamic structures must be available for use after the conversion to DB2:

- ▶ Messages and codes
- ▶ Edit and code tables
- ▶ IDMS Integrated Data Dictionary security
- ▶ IDMS online application structures
- ▶ Scratch and queue processing

8.2.2 Mapping CA IDMS data to DB2

When the target database is not a one-for-one image of the CA IDMS database, each CA IDMS data field must be mapped to a new (and potentially different) database column in the relational target. In many instances, the CA IDMS fields from a single CA IDMS record type map to multiple tables in the target database. Some CA IDMS fields are dropped completely. The values for some relational table columns are generated based upon the values in multiple CA IDMS record types.

CA IDMS has its own database design. When this design is converted to DB2 for z/OS, objects are converted to the closest objects of the DB2. The programs language accesses the data by using different commands that must be translated to SQL.

Data design

The CA IDMS records are mapped to DB2 tables and the set members must be handled carefully to ensure the integrity between a member record and an owner record.

The elements and items are mapped as the DB2 columns. For CALC keys, the DB2 primary keys are defined to follow the same CALC keys. This configuration also is used for sets in which foreign keys are created. The NEXT sets are applied in DB2 for z/OS by using table joins.

For the indexes, the ordering definitions for CA IDMS sets suggest the use of a clustering index over the primary key. A normal index is created for those instances in which the indexed set is not based on the primary key.

The CA IDMS subschema view-id is converted directly to a DB2 view. The logical record facility (LRF) access is created by using DB2 views. If the LRF has updates, view with joins are not created and instead-of triggers are used in DB2.

For more information: For more information about views, triggers, and DB2 for z/OS SQL programming, see *DB2 10 for z/OS Application Programming and SQL Guide*, SC19-2969.

Application design

There are many languages that access CA IDMS. Table 8-1 shows the CA IDMS language elements that must be converted to use COBOL functions.

Table 8-1 CA IDMS languages versus COBOL functions

From	To
ADSA	COBOL application definition programs and copybooks
Application Development System (ADS)/Online Dialogs	COBOL CICS presentation and business layer programs and copybooks
ADS/Online Included Code Modules	COBOL copybooks
IDD COBOL Modules	COBOL copybooks
IDD Record Definitions	COBOL copybooks
IDMS Maps	BMS Maps, CICS COBOL map edit layer programs, and related copybooks
IDMS-DC COBOL Programs	CICS COBOL programs that access DB2
IDMS Batch COBOL Programs	Batch COBOL programs that access DB2
IDMS Record Type Definitions	COBOL SQL layer programs
Easytrieve Programs accessing IDMS	Easytrieve programs that access DB2
Culprit Programs accessing IDMS	Culprit programs that access DB2
IDMS Database Definitions	COBOL database loader programs
IDMS Subschema Definitions	COBOL conversion loader programs
Job Control Language (JCL) and PROC Streams referencing IDMS	JCL and PROC streams referencing DB2

Not all data manipulation language (DML) statements translate to SQL. Procedural commands, FIND commands for navigation to a specific object, and commands that relate specifically to currency have no meaning in a relational environment.

Table 8-2 summarizes the CA IDMS DML statements and shows the equivalent SQL statements.

Table 8-2 CA IDMS DML statements versus SQL statements

IDMS DML	DB2 SQL
ACCEPT DBKEY	Can be used by ROWID
BIND	No equivalent, no need to convert
CONNECT rename TO setname	UPDATE tablename with foreign key
CONNECT rename TO index	DB2 handles this statement automatically
DISCONNECT rename FROM setname	UPDATE tablename set foreign key to null

IDMS DML	DB2 SQL
DISCONNCT recname from index	DB2 handles this statement automatically
ERASE	DELETE FROM tablename WHERE
FIND	No direct equivalent, are simulated with SELECT FROM tablename WHERE EXISTS
FIND CURRENT	No equivalent, check scrolling cursors
FINISH	COMMIT
GET	Act as the FIND
MODIFY	UPDATE tablename
OBTAIN CALC	SELECT FROM tablename WHERE primary key = value
OBTAIN FIRST WITHIN setname	SELECT FROM tablename FETCH FIRST 1 ROW ONLY
OBTAIN LAST WITHIN setname	SELECT FROM tablename FETCH FIRST 1 ROW ONLY by using DESCENDING
OBTAIN NEXT WITHIN setname	FETCH ascending cursor
OBTAIN OWNER WITHIN setname	No direct equivalent, are simulated with SELECT from tablename in which the tablename is the parent table
OBTAIN PRIOR WITHIN setname	FETCH descending cursor
OBTAIN recname DBKEY IS dbkey	SELECT from tablename WHERE ROWID = value
READY	Similar to LOCK TABLE
ROLLBACK	ROLLBACK
STORE recname	INSERT INTO tablename

Many SELECTs calls require the use of a cursor to access DB2 data. Cursors must be disabled when only one row is retrieved. Cursors must be open and closed correctly to avoid causing locks after the program closes.

Although ORDER BY is used, the cost of executing the sort can be eliminated by defining indexes on the columns that are referenced by the ORDER BY clause.

Tune your SQL statements by using a tool or manually, and check the DB2 access path, index usage, DB2 resources, and so on.

For more information about tuning SQL, see Chapter 11, "Application and SQL tuning" on page 175.

8.3 DB-Shuttle by ATERAS conversion solution

In Chapter 7, “Conversion solutions” on page 73, we described the benefits of using an automated tools solution during the conversion process. There are many solutions available to help customers who are converting their applications and CA IDMS to DB2 for z/OS.

The most important decision is to select a solution with proven capabilities that were used in previous projects. In this book, we used the DB-Shuttle solution from the company ATERAS for CA IDMS to DB2 for z/OS conversion. This solution was used in many conversions (approximately 90%, according to ATERAS records) from CA IDMS to relational databases in which the most popular migration target was DB2 for z/OS.

This solution uses conversion methodology. The application and database are converted to a new language and to DB2 for z/OS.

8.3.1 DB-Shuttle solution details

DB-Shuttle uses the customer source code as data within its DB2 for z/OS database. DB-Shuttle analyzes and manipulates the data during de-construction of the mature databases and applications and the construction of the replacement relational databases and applications.

This solution has the following benefits:

- ▶ Handles all aspects of CA IDMS
- ▶ Minimal to no affect on existing applications
- ▶ Minimal effect on users because the logic remains the same
- ▶ Repeatable and Iterative
- ▶ Supports late-arriving maintenance updates
- ▶ Consistent (same results each time)
- ▶ Migrate customers to a modern platform – extensible and standards-based
- ▶ Conversion of the databases, data, and applications
- ▶ Rules-based conversions meet technical and business requirements
- ▶ Business rules are retained
- ▶ Shorter conversion times
- ▶ No code freeze, therefore, no disruption to business activities
- ▶ Consistent track record of delivering successful conversions
- ▶ Comprehensive system documentation available
- ▶ Adheres to service level agreements
- ▶ No on-going license fees
- ▶ No proprietary code or languages
- ▶ Complete assessment, planning, and conversion
- ▶ Easily maintainable n-tier architecture

An overview of the solution that was used by ATERAS is shown in Figure 8-1 on page 91.

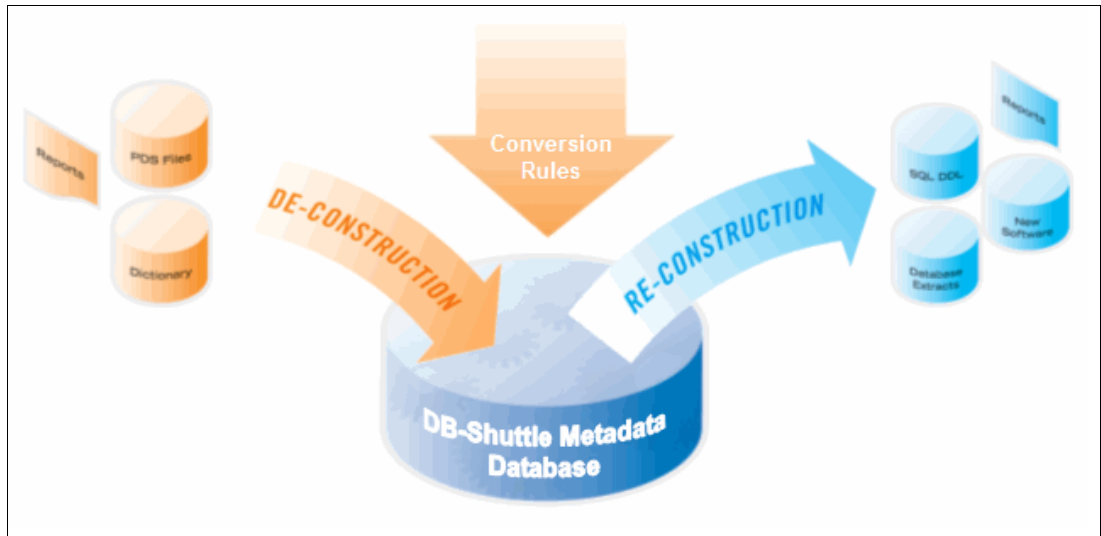


Figure 8-1 Overview of ATERAS solution with DB-Shuttle

Figure 8-1 shows the following aspects of DB-Shuttle:

- ▶ Input data in the form of reports, CA IDMS dictionary, and the .PDS files of the applications are converted to DB2 data definition language (DDL)
- ▶ The data extracted that is loaded
- ▶ The new source code that was developed under the created conversion rules

ATERAS conducted the following phased conversion (these phases are described in greater detail later in this chapter):

- ▶ Phase 1: Automated assessment:
 - Full analysis of all mainframe applications and databases completed
 - All components are classified and listed in detail
 - Missing components are identified, collected, and added to the inventory
 - Targeted project planning enabled
- ▶ Phase 2A: Automated database and data conversion:
 - Relational DDL to mirror existing data structures delivered
 - Data extract programs delivered
 - Workbenches for data conversion created
 - Data cleansing is completed
- ▶ Phase 2B: Automated application conversion:
 - Performed concurrently with Step 2A
 - Converted software delivered ready to install
 - Workbenches for customer-specific rules (renaming, options) created
- ▶ Phase 3: Application testing
- ▶ Phase 4: Deployment
- ▶ Phase 5: Post-implementation support

8.3.2 Additional tasks outside of the standard DB-Shuttle conversion

Although DB-Shuttle provides an automated solution for converting from IDMS to DB2, the following situations and tasks must be managed by the customer:

- ▶ Special uses of the CA IDMS dictionary within the applications
- ▶ Missing code
- ▶ Physical configuration of the CICS regions
- ▶ DB2 for z/OS setup
- ▶ DB2 for z/OS security
- ▶ Physical database design considerations
- ▶ Application tuning specific for DB2 adoption

Special uses of the CA IDMS dictionary within the applications

Some IDMS applications are coded to read the IDMS data dictionary, also known as the IDMS Network Database. The IDMS Network Database is the equivalent of the system catalog in DB2. Applications access the IDMS Network Database to retrieve information such as the IDMS record length, and table and field definitions. These applications must be manually changed.

Missing code

During the code collection phase of the conversion, there might be some programs for which the object code exists, but the source code no longer exists. In these instances, the code cannot be automatically converted by using DB-Shuttle. The customer must find a tool to regenerate the source code or determine whether the equivalent functionality is rewritten from scratch.

Physical configuration of the CICS regions

Many customers use CICS for their OLTP environment. DB-Shuttle converts the IDMS OLTP environment, ADSO, or CICS, to COBOL CICS accessing DB2. This conversion introduces new DB2 workload to the CICS environment. The customer is responsible for programming the necessary definitions in CICS to handle the new DB2 workload. For more information about configuring the CICS regions for DB2, see 10.3, “CICS configuration” on page 169.

DB2 for z/OS setup

The customer is responsible for installing and configuring the DB2 for z/OS environment to handle the new DB2 workload. This responsibility includes such tasks as defining DB2 system parameters and configuring buffer pools. For more information about configuring the DB2 for z/OS environment, see Chapter 10., “z/OS and DB2 setup”.

DB2 for z/OS security

The customer is responsible for ensuring that the data on DB2 for z/OS is secure. DB2 controls access to data within a DB2 subsystem. DB2 also works with outside security systems, such as RACF, that control access to the DB2 subsystem. If the converted data and applications are introduced to an existing DB2 subsystem, it is likely that an existing security scheme is in place. If the converted data and applications are the first DB2 implementation, a security scheme must be implemented. For more information about managing security for your DB2 data and applications, see *DB2 10 for z/OS Managing Security*, SC19-3496.

Physical database design considerations

The automated conversion produces a like-to-like DB2 database design. Each IDMS record is converted to a single DB2 table. In some instances, more columns or tables are created to maintain IDMS set relationships. In addition, indexes are created to support primary and foreign keys, and to support any ordered sets.

Primary and secondary space allocations, and buffer pool assignments are determined during the conversion process by providing the information in the conversion workbench. When you start testing data and transaction activity at high volumes, you might find that the space allocations or the buffer pool assignments must be adjusted.

All of these considerations might result in a physical database design that is different from the design that is generated during the automated conversion process. These design changes are a normal part of any database conversion.

Application tuning specific for DB2 adoption

Some IDMS applications return data to the program and then the program provides filtering logic to process some data and reject other data. You might determine that you want to take advantage of relational database capabilities and provide filtering in the SQL statements rather than in the programs. You might need more indexes to support those SQL changes and provide adequate performance concurrently. For more information about the advantages of modifying the converted programs to provide filtering in the SQL statements, see 11.1.4, “Filter in DB2, not in program” on page 177.

8.3.3 Assessment process

A mainframe IT assessment is a consistent method of understanding the existing program and script components of a mainframe environment. Understanding these components helps you deciding whether to conduct a conversion, modernization, or migration project. An ATERAS IT assessment includes fully defining the goals and areas of concentration before data collection is conducted. ATERAS uses DB-Shuttle automation technology to conduct a detailed, targeted IT assessment of the entire mainframe application environment. This assessment helps define a conversion project plan that includes assignments, responsibilities, time frames, and costs.

Figure 8-2 on page 94 shows how ATERAS and customers interact during the assessment process.

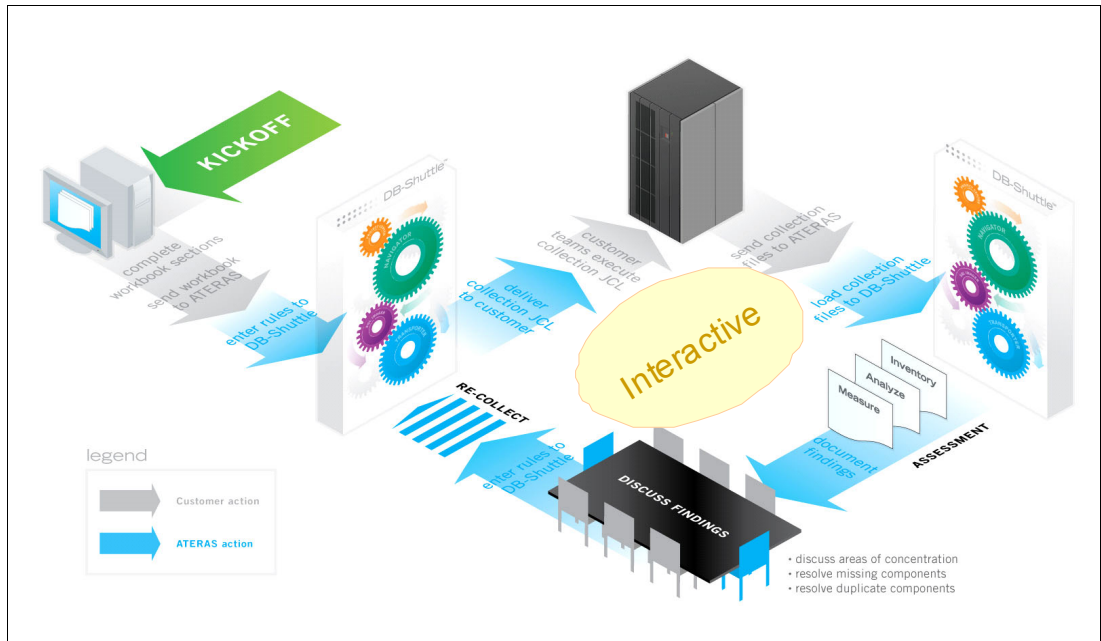


Figure 8-2 Assessment process

Process

The Assessment phase is a well-defined process that includes the automated collection, inventory, analysis, and measurement of all software and database components in the customer environment. The assessment begins with a question and answer session and ends with the presentation of findings and plans for the modernization effort. Iterations of recollection and reassessment processes might be required as more components are identified and brought into the assessment scope. The DB-Shuttle assessment process is fast, simple, and comprehensive because it is automated.

Deliverable

The ATERAS Assessment results in a well-defined plan for the conversion project, including costs, time frames, and a project map that details the concept to deployment process. High-level and detailed assessment reports are presented to appropriate customer team members for discussion.

Tasks

The ATERAS assessment process includes the following tasks that simplify, summarize, and define the project:

- ▶ Presentation of technical and business findings
- ▶ Definition of the overall customer processing
- ▶ Details of missing and duplicate components
- ▶ Summarization of areas that require special attention during conversion
- ▶ Documentation of all areas of concern
- ▶ Organization of primary findings into an executive summary
- ▶ Disclosure of recommended actions

Reports

The ATERAS Assessment produces the following set of reports that is based on the source code and database environment:

- ▶ Component list detail and summary
- ▶ DML usage detail, matrix, and summary
- ▶ Duplicate components
- ▶ File usage by batch program, data set name, and online program
- ▶ Logical application by logical application
- ▶ Logical applications by component: detail, matrix, and summary
- ▶ Logical applications matrix
- ▶ Logical applications message detail
- ▶ Missing component detail and summary
- ▶ Resolved component dependency
- ▶ Subschema logical record detail and matrix
- ▶ Subschema record matrix
- ▶ Subschema view matrix
- ▶ Bill-of-material structures
- ▶ Multi-member sets
- ▶ TOPS (online and batch application entry points)

Table 8-3 shows that the number of converted programs decreased after the assessment was run.

Table 8-3 Assessed programs versus converted programs

Project name	Assessed component count	Assessed line count	Converted component count	Converted line count
PROJ1	43922	8995039	231	109613
PROJ2	25505	5529138	1744	565749
PROJ3	104833	12093107	13667	4688021
PROJ4	36776	4735421	6790	2173267
PROJ5	64727	14777426	36332	11574369

8.4 DB-Shuttle conversion process

In this topic, a conversion process that is based on a real customer scenario is reviewed. In this scenario, ATERAS assisted the customer to convert a database from IDMS to DB2 for z/OS by using DB-Shuttle.

8.4.1 Challenges of the customer

The customer listed the following challenges that the conversion process must solve:

- ▶ Losing skilled CA IDMS DBA resources
- ▶ Cannot easily access CA IDMS data from other platforms
- ▶ Use solutions available to DB2
- ▶ Cannot provide 24x7 uptime as is available with DB2 for z/OS

8.4.2 Environment of the customer

The customer uses one or more of the following IDMS languages or dialects:

- ▶ ADS/Online
- ▶ ADSA
- ▶ IDMS-DC COBOL
- ▶ CICS COBOL with IDMS access
- ▶ Standard batch COBOL with IDMS access
- ▶ JCL and procedures that use IDMS utilities and references

The numbers for each component assessed in this project are shown in Table 8-4.

Table 8-4 Converted components of the customer

Component	Count	Lines of code
IDMS Databases	1	N/A
IDMS Schema Records	795	N/A
IDMS Subschemas	241	57653
TABLE PROCEDURE	331	12106
ADSA Applications	23	571709
ADS/Online Dialogs	1547	3119935
COBOL/IDMS-DC	146	147536
EDIT/CODE TABLE	2142	104767
MAP	1367	2546097
MAP/PAGEABL	27	21801
TASK	42	N/A
COBOL/BATCH	1161	1002908
COBOL/DC-BATCH	426	446814
CULPRIT	2771	445216
CULPRIT-IDMS	1095	115044
CONTROL MEMBER	2579	150465
JCL	4151	521684
JCL/CULPRIT	1313	1091249
JCL/CULPRIT-IDMS	399	142970
JCL/PROC	134	13677
JCL/PROC/INSTREAM	8	1069
Assembler Programs	12	1867
COBOL	212	144591
COBOL/COPYBOOK/RECORD	46	9122
IDD/COBOL MODULE	230	40488

Component	Count	Lines of code
IDD/CULPRIT MODULE	38	10437
IDD/RECORD	4143	296616
IDMS FILE	163	2071

Customers use the following capabilities of the IDMS network database structures. The solution ensures that the same capabilities, relationships, and constraints of IDMS are built into the new DB2 for z/OS relational database:

- ▶ Multi-member sets
- ▶ Optional sets
- ▶ Bill-of-material structures
- ▶ System-owned indexes
- ▶ NEXT sets
- ▶ Sorted sets with duplicates first or last
- ▶ Find/Obtain KEEP EXCLUSIVE
- ▶ Find/Obtain USING
- ▶ Find/Obtain DBKEY without a record type
- ▶ Accept DBKEY
- ▶ Database procedures
- ▶ Conversion to DATE data types
- ▶ Data cleansing
- ▶ Renaming tables and columns
- ▶ Redefines clauses
- ▶ Occurs clauses
- ▶ Group-level elements
- ▶ System-owned optional sorted sets with duplicates allowed
- ▶ Record-owned optional multi-member sets stored NEXT
- ▶ Index structures on redefined data fields

Online application commands

Customers use the following capabilities of the current IDMS-DC and ADS/Online runtime environment. The solution ensures that the same capabilities are replaced in the new CICS COBOL DB2 applications and that those capabilities provide functionality that is identical to the functionality in IDMS:

- ▶ Scratch and queue record management
- ▶ IDD message codes and text
- ▶ Edit and code table replacements
- ▶ Write to log, operator, and printer
- ▶ ADSA flow of control
- ▶ Deferred and immediate responses
- ▶ LINK commands
- ▶ Modify map temp and perm
- ▶ Built-in functions
- ▶ Nested built-in functions
- ▶ Map edits not supported by BMS
- ▶ Pageable maps
- ▶ ADS moves versus COBOL moves
- ▶ Numeric and alpha data moves
- ▶ Error-status checks in WHILE and IF statements
- ▶ Set membership checks in WHILE and IF statements
- ▶ Exit commands within WHILE statements

8.4.3 Project phases

The conversion project was completed in approximately six months. Table 8-5 shows the timing of each phase of the project. The tasks that are conducted in each phase are described later in this section.

Table 8-5 Sample of activities duration

Phase	Timing
Assessment phase	One month
Database delivery	One month
Online application delivery and batch application delivery	Two months
Testing support	Three months
Deployment and post-implementation support	One month

Assessment phase

The assessment phase includes the delivery of reports, lists, counts, and matrices (at the summary and detail levels) that describe the application components and the inter-relationships between the programs and databases. A detailed project plan is the final deliverable. The general time frame for the assessment phase to be completed is 4 - 6 weeks. This phase is described in 8.3.3, "Assessment process" on page 93.

Figure 8-3 on page 99 shows the inputs necessary to start the assessment of the global definitions in the environment of the customer, including job cards, job libraries, and so on.

	A	B	C	D
1		3. Global Definitions		
2				
3				
4		1. Job Card	Enter or paste the default JCL job card to use on all JCL streams created by DB-Shuttle for source code collection, data extracts, etc. Your existing production job streams will continue to have the same job cards as collected, except where other rules a	
5				
6				
7			<pre>//jobcard // /*JOBPARM xxxxxxxx /***** /** your name /** your company/organization /** your phone /** your e-mail address /*****</pre>	
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18		2. Job Library	Enter the default JCL JOBLIB to use on all JCL streams created by DB-Shuttle for source code collection, data extracts, etc. Your existing job streams will continue to have the same JOBLIB as collected, except where other rules are applied. A STEPLIB ma	
19				
20			<pre>//STEPLIB DD DISP=SHR,DSN=steplib1 // DD DISP=SHR,DSN=steplib2</pre>	
21				
22				
23				
24				
25		3. Sysctl/ Dictname	List the names of all of your production dictionaries along with their related SYSCTL ddname in the format "sysctl/dictionary". The components within these dictionaries will be collected for assessment.	
26				
27			<pre>SYSCTL1/DICT1 SYSCTL1/DICT2 SYSCTL2/DICT1 SYSCTL2/DICT2</pre>	
28				
29				
30				
31				
32				
33				

Figure 8-3 Global definitions

Database delivery phase

During the third month of the project, the customer receives the first pass DDL to replace the IDMS databases and delivers all of the COBOL extract programs to unload the IDMS data. By the end of the month, the process to install and populate the new DB2 databases is complete, and the databases are ready for query, diagramming, and other technical inspection by the Database Administration teams.

The database and data conversion phase also includes the construction of new relational database structures that are designed to provide the same traversal and performance capabilities as the former database. All of the requirements for table and column naming, date conversion, element grouping, occurs clause handling, and data cleansing are managed in this phase.

Online application delivery

By the end of the fourth project month, the first pass delivery of the online CICS COBOL applications is complete and any other components that are required for the new applications are delivered. The customer teams receive the new source components, which are installed, compiled, and tested.

The application conversion phase occurs simultaneously with the database and data conversion. All renaming of components and generation of new software takes place during this phase.

Figure 8-4 on page 100 shows the IDMS map conversion to basic mapping support (BMS).

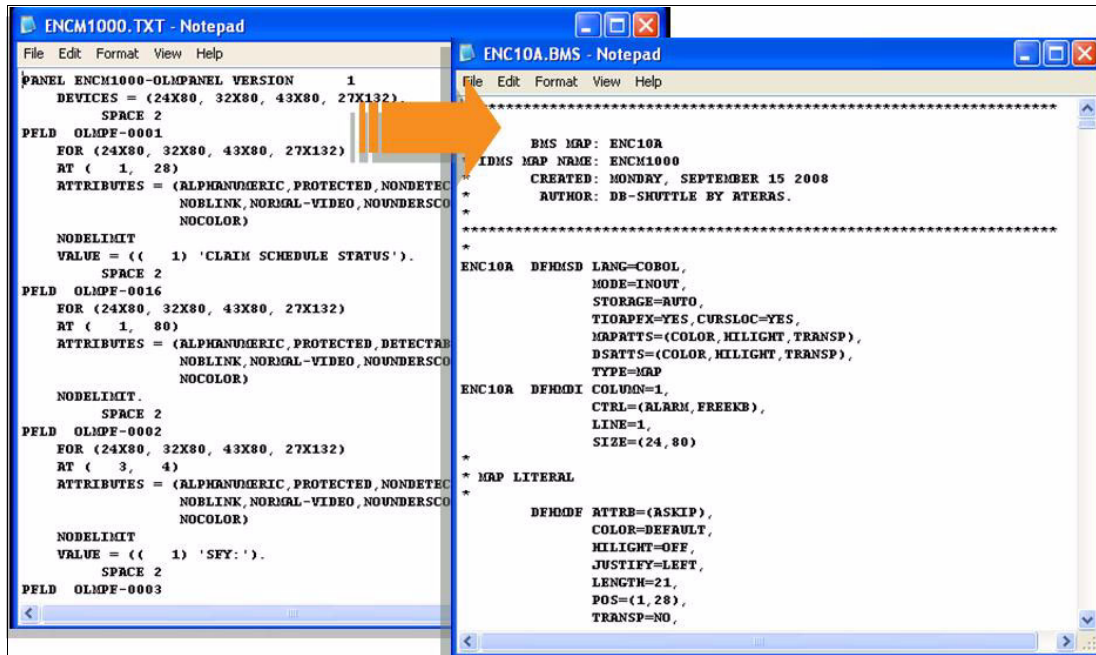


Figure 8-4 Conversion to BMS map

Batch application delivery

The first pass delivery of the batch replacement components, including the COBOL, Culprit, and JCL components, often takes place during month five. As with the online applications, the customer teams receive the new source components and then install, compile and build, and begin the testing process.

Additional collections and deliveries

As the customer continues to test and the production changes to the current applications are recollected for processing, the database and application components are regenerated, delivered, and reintroduced to the testing environment. A regeneration and delivery cycle takes from one hour to three days to complete.

Testing support

The ATERAS teams always are available to assist with any conversion-related issues that are encountered during the customer testing phases of the process. When conversion-related issues are found, the issues are reported and tracked until the issues are resolved by using the web-based Foot Prints system. Additional rules are introduced into DB-Shuttle and any components that are affected by the new rule set are regenerated.

Deployment and post-implementation support

A primary team is on-site with the customer during the system cut-over, and all other ATERAS team members are on-call if the customer needs assistance. The ATERAS team also provides any required assistance during the 90-day warranty period that follows the deployment. The deployment and post-implementation support phases ensure that the implementation of the new software and databases is complete and precise.

8.4.4 Database and data conversion

The database and data conversion phase begins early in the modernization process. The new database designs are the basis for the conversion and regeneration of the related applications.

Overview

ATERAS works with the database administration team of the customer to select the options and define the rules that are used for modernization. This working relationship ensures that the resulting database conforms to the naming standards of the customer. By working together, the teams also define the rules for data cleansing and managing *occurs* clauses, *group-level* elements, and *redefines* clauses.

DB-Shuttle generates the new databases according to the defined naming rules and options.

The first pass of the new SQL DDL is delivered within the first few days of this conversion phase. The teams implement rule adjustments and regenerate the DDL until the DDL meets the customer standards.

Although the database and data conversion phase often is conducted simultaneously with the application software conversion, it is possible to conduct these phases independently.

Process

The database and data conversion phase includes the deconstruction of the mature database definitions into metadata artifacts. ATERAS enters more rules that are based on the following customer workbook entries:

- ▶ Column and table naming date-type formats
- ▶ Overrides for redefines and group level clauses
- ▶ Preferences for each statement and clause in the resulting DDL

The generation of new DDL defines a complete relational database and provides the same data access as the mature database. This well-defined process includes the collection and automated inventory, analysis, and measurement of the software and database components that are collected from the customer environment.

The assessment begins with a question-and-answer workbook section. The assessment ends with the presentation of the final findings and the plans for the database environment and the modernization process. Recollection and reassessment might be required as other components are identified and added to the scope of the assessment. Because the DB-Shuttle assessment process is automated, the process is fast, simple, and comprehensive.

The resulting relational database definitions are installed at ATERAS before the definitions are delivered to the customer, which ensures that the generated code is syntactically correct. For mainframe-to-mainframe conversions, ATERAS delivers the JCL that is required to compile and run the mature database extracts. Workbenches for date-type management allow the new databases and applications to use all of the “date and time” data types in the relational database.

The mature database definitions are recollected and reconverted anytime throughout the project lifecycle. This ability prevents the need to prohibit changes to the mature database structure.

Deliverables

The resulting relational database definitions are packaged, delivered, and ready for installation in the new target environment. The data extract programs also are packaged, delivered, and ready for compilation and execution in the former database processing environment.

Tasks

ATERAS performs the following high-level tasks during the database and data conversion phase:

- ▶ Provide a database definition form for each database that is converted.
- ▶ Provide reports on occurs clauses, redefines clauses, group levels, and potential date fields.
- ▶ Provide guidance in correctly and completely filling out the database definition forms.
- ▶ Import the schema and database definition reports.
- ▶ Generate the initial SQL Imager load summary reports for the schemas.
- ▶ Review the complexity reports for the schemas.
- ▶ Review log messages for any errors during the import and initial parse phases.
- ▶ Load the table naming rules defined in the submitted Database Definition forms.
- ▶ Load the column naming rules as defined in the submitted Database Definition forms.
- ▶ Load the date rules for each date element.
- ▶ Load the adjustment rules for any redefines, occurs clauses, and group levels.
- ▶ Generate the first pass of the converted database.
- ▶ Present questions regarding the first-pass conversion.
- ▶ Generate the DDL, Load syntax, and related JCL for the converted databases.
- ▶ Generate the record relationship diagrams for the old database and file structures.
- ▶ Generate the entity relationship diagrams for the relational databases.
- ▶ Generate the non-relational extract program series and cross-reference reports.
- ▶ Conduct a walkthrough of the deliverable and processing instructions.
- ▶ Apply changes based on input from the customer teams.
- ▶ Apply cleansing rules as provided by the customer teams.

Reports

The ATERAS conversion produces the following reports:

- ▶ Data cleansing report
- ▶ Date conversion detail
- ▶ Element rename detail
- ▶ Extract file cross-reference
- ▶ Extract program matrix
- ▶ Index columns
- ▶ Key columns
- ▶ Key inheritance existing fields
- ▶ Key inheritance inserted fields
- ▶ Key mapping summary
- ▶ Key rename summary
- ▶ Load errors

- Load summary
- Record scoring
- Record table comparison all
- Record/table view by schema
- Date finder report

Relational results

The resulting database is fully relational. Primary keys, foreign keys, and index definitions are created automatically. All constraints are generated into the resulting DDL. Table spaces, indexes, table names, and column names are generated according to the naming standards of the customer.

Data conversion

During the extract process, all IDMS traversal is based on mandatory and optional set relationships and the clustering specifications defined within the IDMS schema. The IDMS data extracts are run simultaneously. ATERAS provides a number of extract variations for sites that have special requirements for a short IDMS conversion window. Data conversion during the cutover weekend or evening is fast and complete.

Figure 8-5 shows the database conversion process steps.

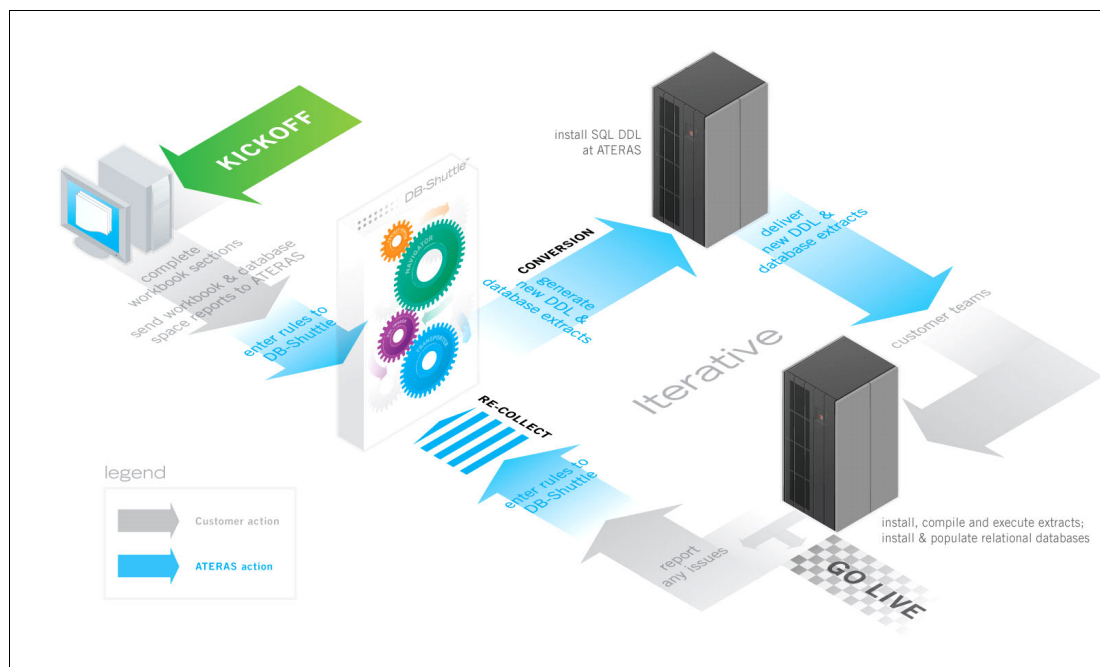


Figure 8-5 Database conversion process steps

Re-engineering and customization workbenches

The following special workbenches within DB-Shuttle provide more capabilities for tailoring your IDMS migration so that the migration better meets your needs and requirements:

- Rename workbenches allow full naming of all the tables, columns, table spaces, and indexes by using a rules basis or a full-name basis.
- Data cleansing workbench provides rule-based data cleansing during the IDMS data extract process.

- Date conversion workbench allows the specification of the IDMS date fields format and the individual IDMS date field minimum and maximum values so that the DATE column specification is used in relational databases (all date formats are supported).
- Element rename workbench allows selection of group level or elementary IDMS fields for use as columns in the relational database.
- Record redefinition workbench allows changes to field types and lengths during the relational conversion.
- Set redefinition workbench allows changes in the set definitions during the IDMS conversion.
- Co-existence record conversion workbench allows the conversion and implementation of large IDMS databases by using a phased approach. In this approach, groups of IDMS record types are specified for each conversion phase.

Data cleansing

Any type of data cleansing rule might be applied to any field. The customer provides specialized code to conduct the checks, including the retrieval and interrogation of other data fields to help determine the manner in which a different data field is cleansed. The customer enables the ATERAS default cleansing rules for numeric fields. With these default rules enabled, every numeric field that contains non-numeric data and the DBKEY for the record occurrence that contained the data are displayed. During the extract program execution, the cleansing rules are completed and the results are displayed in the extract output files.

Figure 8-6 shows the BACHMAN diagram that is mapped to the relational model shown in Figure 8-7 on page 105.

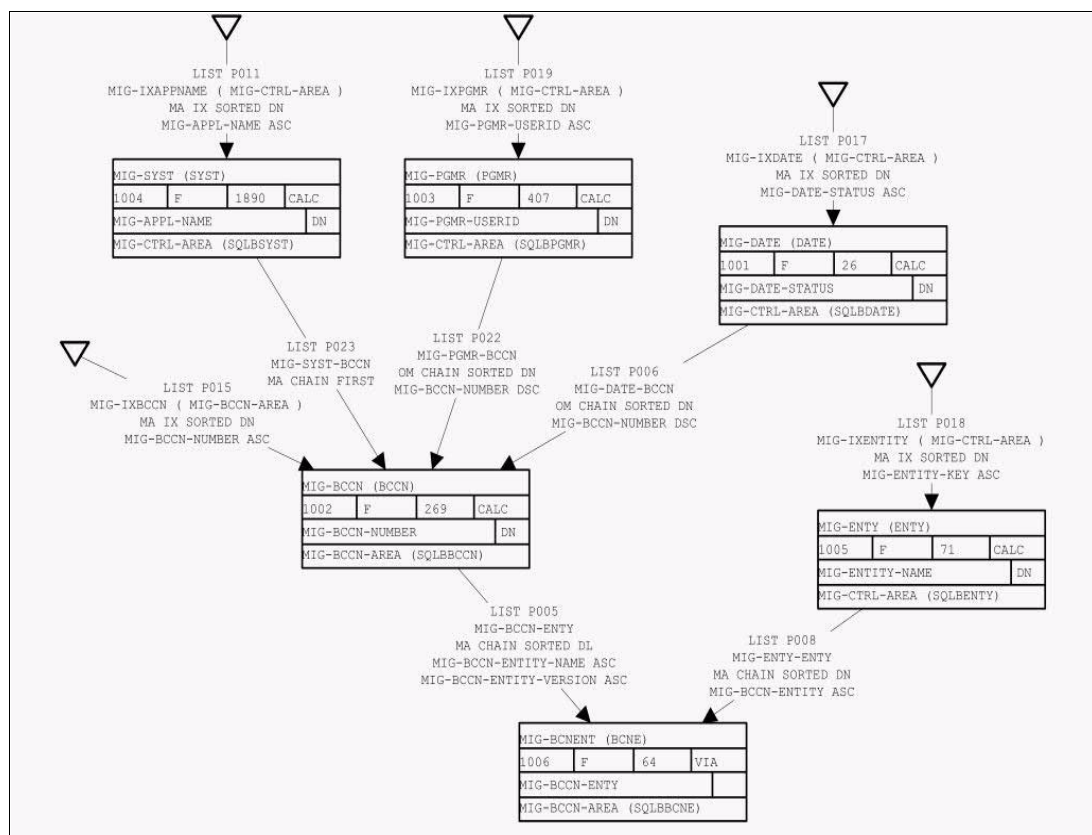


Figure 8-6 BACHMAN diagram

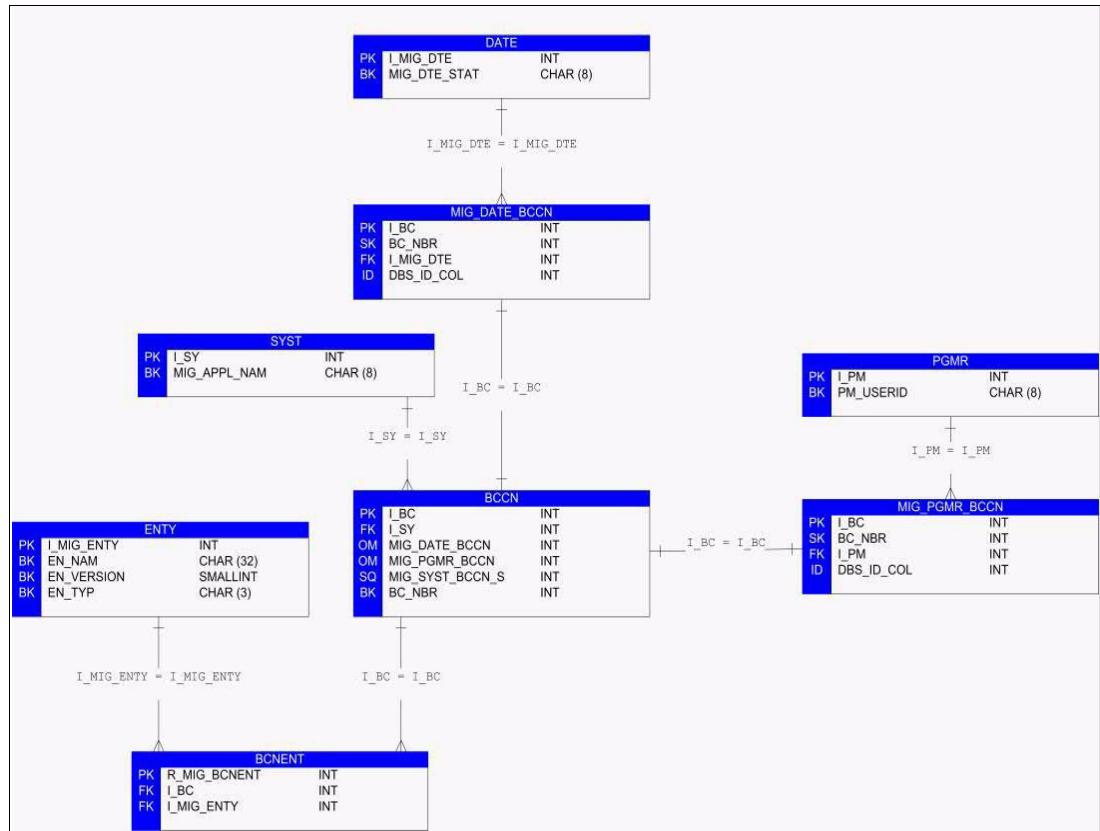


Figure 8-7 Converted data model

Delivered components

As part of the delivery process, ATERAS generates and delivers the following component types for installation in the new relational database processing environment by using DB-Shuttle:

- ▶ DDL syntax for the new database
- ▶ IDMS data extract programs (generated in COBOL and fully self-documented) to unload all IDMS data to the correct format for relational database load utility
- ▶ Optional:
 - Load syntax for use by relational database load utility
 - RI check syntax for use by relational database utility package
 - RUNSTATS syntax for use by relational database utility package
 - IDMS data extract JCL to run the extracts and other key-processing utilities (customized to your environment)
 - DCLGEN syntax to define COBOL layouts of the tables for replacement applications

8.4.5 Application conversion

The application conversion phase begins immediately after the assessment phase. New applications are generated in the target languages and run against the new relational database structures. If necessary, components are renamed for use in the new target environment.

Overview

The application software conversion and the database and data conversion phases take place concurrently. The database administration teams must be involved in the database and data conversion. The application teams must be involved in the application conversion.

The application conversion process steps are shown in Figure 8-8.

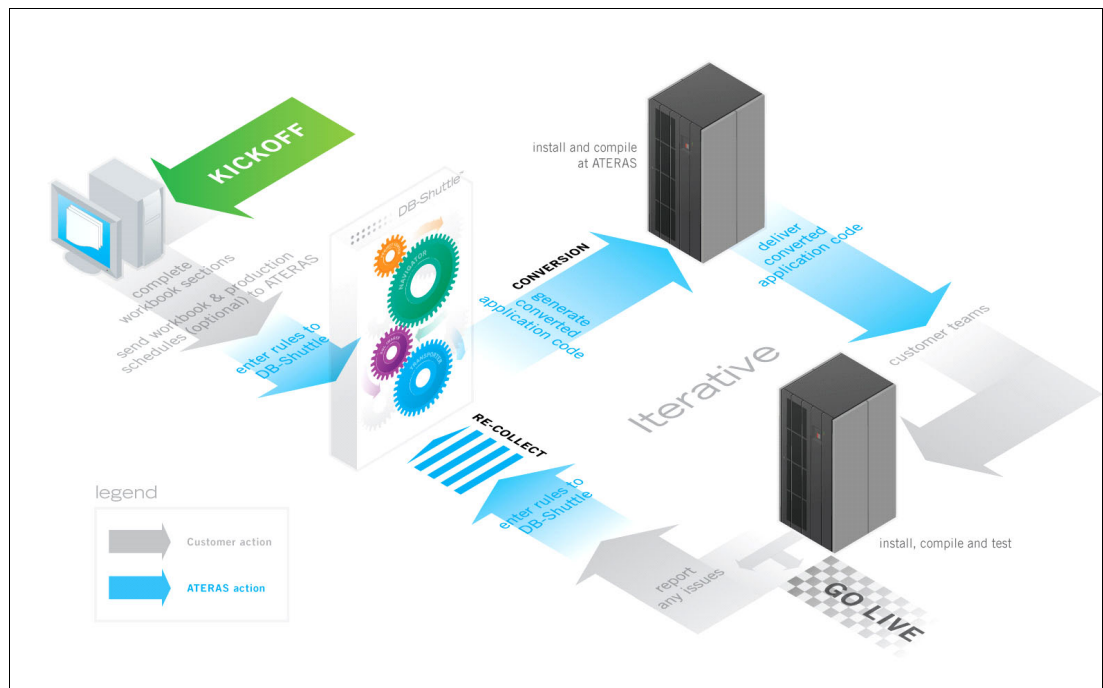


Figure 8-8 Application conversion process

Process

This conversion phase is an iterative process that includes application pre-conversion setup and application conversion. Support for testing is provided at multiple levels.

Deliverable

ATERAS packages and delivers the converted applications, which are ready for installation, compilation, and testing.

Figure 8-9 on page 107 shows the IDMS COBOL as it is converted to COBOL.

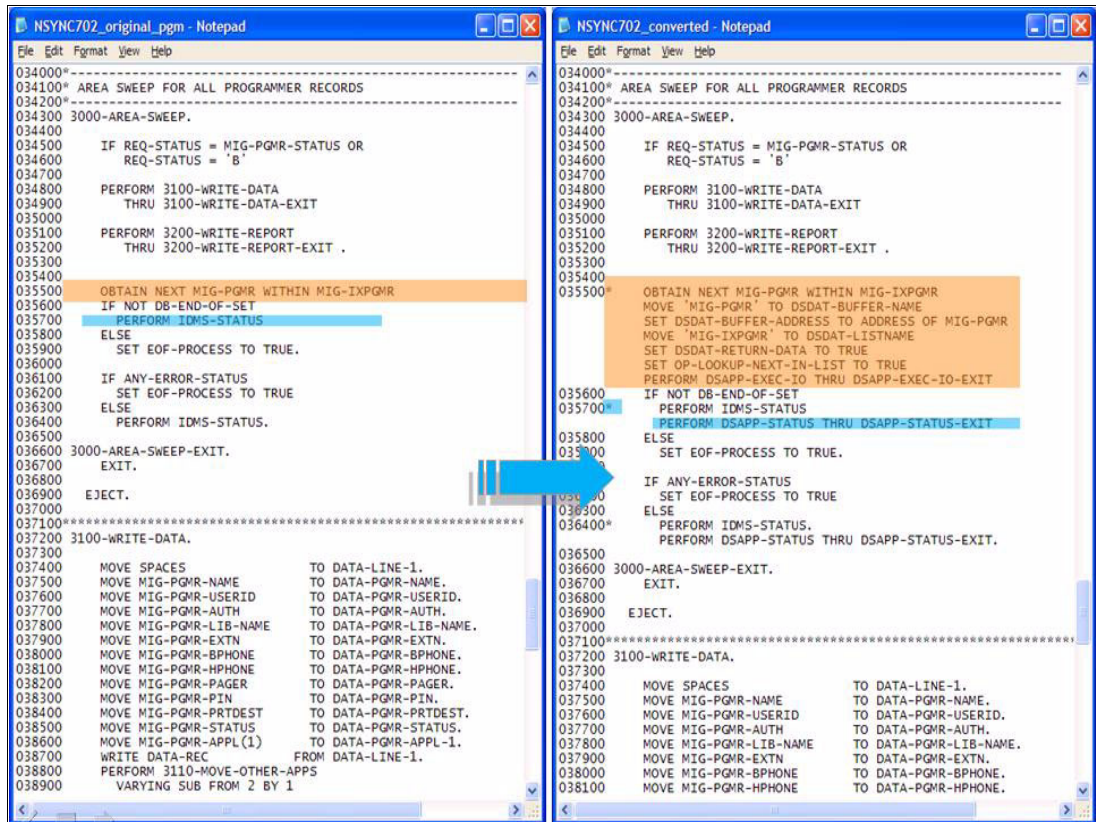


Figure 8-9 IDMS COBOL to COBOL DB2

Tasks

ATERAS performs the following high-level tasks during the application conversion phase:

- ▶ Provide the JCL to collect software from all source code repositories.
- ▶ Define and implement a secure customer database within DB-Shuttle.
- ▶ Import all source code from the collection files into the secure database.
- ▶ Provide reports on components that are referenced and not referenced in logical applications.
- ▶ Provide detailed and summary reports of the collected components.
- ▶ Review and understand renaming and rules definitions that were submitted by the customer teams.
- ▶ Enter the renaming and processing rules that were defined in the conversion workbook into DB-Shuttle.
- ▶ Generate reports of the proposed renaming and conversion rules.
- ▶ Define a secure FTP site for the delivery of converted components.
- ▶ Generate the converted components.
- ▶ Conduct preliminary compiles of the converted code in the ATERAS environment.
- ▶ Deliver the first pass of the supplier deliverable to the FTP site.
- ▶ Conduct a walkthrough of the software installation process and target architecture.
- ▶ Assist customer teams with the installation of the new software into the target environment.

- Provide new and revised components (as required) to resolve any conversion issues.
- Conduct maintenance training that covers the converted applications and databases.

Figure 8-10 shows the dialog to CICS layer presentation conversion.

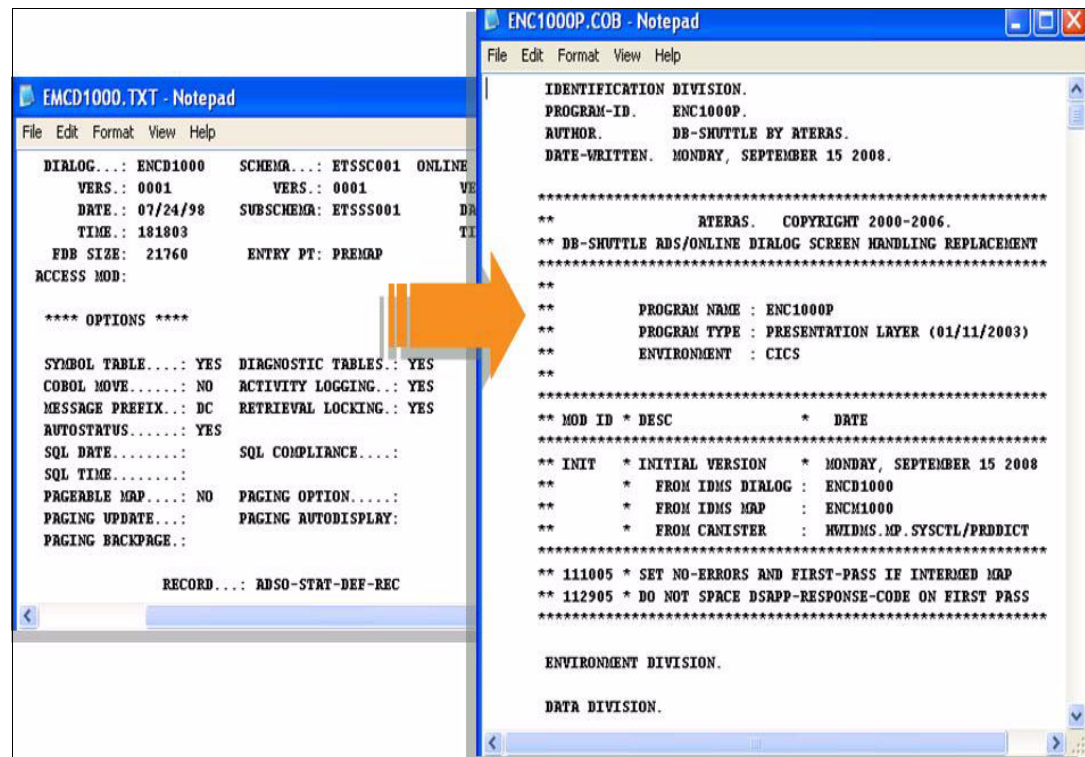


Figure 8-10 Dialog conversion

Reports

The ATERAS conversion produces the following reports:

- Converted online detail
- Converted common code detail
- Converted dictionary detail
- Converted maps detail
- Converted edit components detail
- Software delivery by component
- Software delivery by type
- Software delivery for COBOL
- Software delivery for JCL
- Language-specific detail analysis (as required)

8.4.6 Results

From the perspective of the business, the customer sees the following results after the conversion project is completed:

- No other black boxes, nothing proprietary
- No on-going license fees
- Reduced maintenance costs
- DBA team improves skills to use DB2 for z/OS

- ▶ Multi-tier architecture and SOA-enabled database to improve agility for IT to deliver business requirements

From a technology perspective, the customer sees the following results:

- ▶ Database is fully relational.
- ▶ Processing is high-performance.
- ▶ Full use is made of cursor processing.
- ▶ Data access is isolated to a set of COBOL DB2 programs; Data Access Layer (DAL).
- ▶ Former DML requests from the application code were changed to call the DAL.
- ▶ The DAL maintains currency, key values, and positioning that were managed by CA IDMS.
- ▶ COBOL CICS CA IDMS programs were changed to COBOL CICS DB2.
- ▶ COBOL CA IDMS-DC programs were changed to have CICS services instead of CA IDMS-DC.
- ▶ ADS/Online dialogs were converted to CICS COBOL multi-structure (presentation and business).
- ▶ CA IDMS Maps were converted to BMS maps and a map edit subroutine for each was produced.
- ▶ Project results in numbers:
 - Converted over 9 million lines of code
 - Created over 795 IDMS schema records
 - Created 2504 JCL job streams
 - Converted 1395 IDMS maps
 - Converted 1559 ADS or online dialogs to COBOL/CICS
 - Converted 1414 Batch and 116 DC-COBOL programs
 - Converted 327 IDMS table procedures to DB2 stored procedures
 - Created 881 unique database structures

Visibility and knowledge building

ATERAS ensures that the customer teams (and the ATERAS teams) have a full understanding of existing IDMS databases and the post-conversion relational database. DB-Shuttle generates the following reports and diagrams to assist with this knowledge-building process:

- ▶ Complete Bachman diagrams for the IDMS database structures.
- ▶ Complete entity relationship model (ERM) diagrams for the relational databases.
- ▶ Summary reports of the sizes and relationships within the IDMS and relational structures.
- ▶ Scoring Reports that rate the complexity of each IDMS record type as it relates to the IDMS migration to a relational definition.
- ▶ Matrix reports that relate records to tables, elements to columns, IDMS records to extract programs, and so on.
- ▶ Date finder and key Identifier reports that allow team members to define selected fields and columns to ensure that the migration addresses all requirements for date conversion and key mapping.
- ▶ Hundreds of ad hoc reports that help identify the unique characteristics of the IDMS database and the conversion.

8.4.7 Lessons learned

The following lessons were learned during the conversion process:

- ▶ A project manager for the customer team is important.
- ▶ ATERAS must have access to the customer team.
- ▶ Customer teams must be able to make decisions.
- ▶ Conversational ADS/Online requires more time with ATERAS by using a specialized workbench in DB-Shuttle.
- ▶ Testing requires input from the customer IT and business teams.
- ▶ Testing is the longest phase of the project.
- ▶ The introduction of the new applications into a software configuration management tool must occur after the applications are tested.
- ▶ Changes to any design or functionality of the application must occur pre-conversion or post-conversion.
- ▶ Some areas of concentration that are defined during the assessment process are best managed by a customer team member because that member understands the design creation process.



Converting Adabas to DB2 for z/OS

In this chapter, we review Natural and Adabas conversion solutions to DB2 for z/OS, including prerequisite technical assessment requirements and conversion challenges. We also review a real customer conversion scenario that was provided by the IBM business partners that facilitated these conversions with customers.

We include an implementation that uses Consist Advanced Development Solution (ConsistADS). ConsistADS is an end-to-end conversion solution that combines conversion and transparency methodologies and includes DB2 for z/OS and several DB2 tools as part of a package that was delivered to the customer.

We also describe the solutions OnTarget from Most, and eavRPM from ATERAS. These solutions focus on the process to convert the language to COBOL or Java accessing DB2 for z/OS by using the environment of the customer.

This chapter includes the following topics:

- ▶ Natural and Adabas conversions
- ▶ Adabas requirements
- ▶ ConsistADS automated conversion solution
- ▶ How ConsistADS maps ADABAS data files to DB2
- ▶ ConsistADS conversion process
- ▶ OnTarget
- ▶ eavRPM solution

9.1 Natural and Adabas conversions

Companies used the Natural and Adabas database to write mission-critical applications that are core to their business. However, Natural applications that are written in a pre-relational database such as Adabas are no longer mainstream and do not follow accepted IT industry standards. This shortfall makes Natural mature applications more difficult and expensive to maintain. The applications also cannot offer greater flexibility at delivering new information services as compared to a relational database.

When customers start a conversion process from Natural and Adabas to DB2 for z/OS, they want to use the relational database capabilities and features that are offered by using relational database architecture. These capabilities and features provide a flexible database platform that delivers on the following business requirements:

- ▶ Web-enabled applications
- ▶ Service-oriented Architecture/Business Process Management (SOA/BPM)
- ▶ Data warehouse
- ▶ Business Analytics
- ▶ Easy recoverability
- ▶ Two-phase commit
- ▶ Increased synergy with security
- ▶ Utilities performance
- ▶ SQL flexibility
- ▶ Uses tables larger than 4 billion rows
- ▶ Use of DB2 for z/OS features, such as partitioning and parallelism
- ▶ Data sharing
- ▶ Large objects
- ▶ Stored procedures
- ▶ Triggers
- ▶ Referential constraints

For more information about these advantages, see Chapter 2, “Converting to DB2 for z/OS” on page 7.

9.2 Adabas requirements

When we think about an Adabas conversion, it is possible to treat each Adabas file as a relational DB2 table. However, this simplistic assumption must not be made. Many Adabas database designs create multiple data type records within a single Adabas record, which is counter intuitive to the relational table normalization philosophy. Because of the major differences in the data file system design models, a database administrator often is required to split Adabas files into several tables to reap the full benefit of a relational model design. A relational database system also allows the users of a client to query any table column by the name of the column based on a value, which is a capability that is not always allowed for in an Adabas file design.

The following steps are used to convert Adabas data to DB2 for z/OS:

1. Verify that each Adabas file is in third normal form. The file must be normalized if the file is not already.
2. Create a DB2 table for each resulting file.
3. Translate each Adabas field to a DB2 column.

4. Identify a primary key for all defined tables.
5. If referential integrity is implemented, a foreign key must be defined for each dependent table.

To convert Adabas to DB2 for z/OS, you choose different methodologies as explained in Part 2, “Migration methodology” on page 23. Customers decide between the following conversion and transparency solutions, depending on their specific needs:

► **Conversion**

In this method, a customer chose to convert all languages and the database. The customer manually migrates their Natural online and batch programs to Java or COBOL, COBOL/CICS, and Adabas to DB2. This conversion process is slower because the customer has many objects to manually convert (source code modules, data modeling, tables definitions, and so on) to create these objects in the new environment.

► **Transparency**

In this method, a customer chose to convert only the Adabas database and use conversion programs to intercept the ADABAS calls of the current Natural program and replace the calls with SQL calls to DB2 for z/OS. This conversion occurs when a customer is reluctant to migrate their Natural code, so the customer chooses to keep Natural running with the same source code as a new object run outside the run time of the Natural. This method often has lower risk and speeds up the project. This solution is combined with the conversion solution completed later.

The two methodologies are compared in Table 9-1.

Table 9-1 Conversion versus transparency

	Conversion	Transparency
Risk	Medium	Low
Project timing	Slower	Faster
DB2 usage	Medium	Lower
Proprietary licenses	Not needed	Needed
Maintenance costs	Lower	Higher
Performance	Higher	Lower

9.2.1 Conversion challenges

The following challenges must be considered when a customer chooses an Adabas conversion method:

- Adabas inventory
- Disk storage estimates
- Data relationship
- Performance
- Applications

These challenges are reviewed next.

Adabas inventory

Estimating the workloads, costs, and schedules for the project is important. Therefore, it is necessary to have a thorough inventory of Adabas and application data volumes. Customers that have PREDICT data dictionary generate reports that provide all of the requisite information.

A conversion project is an opportunity to assess what is required and to plan accordingly. Many programs serve only the development environment and are not used.

Disk storage estimates

Helping customers estimate the amount of disk space that is used in the DB2 for z/OS is important. In some instances, DB2 might require more disk space than ADABAS.

Data relationship

The goal of the conversion is to avoid redundant data by normalizing the data model. The chosen conversion tool must provide flexibility to help the customer decide whether to denormalize the multiple occurring field (MU) and the periodic group (PE) field. However, normalization often is used to create new tables.

The tool must conduct the data type conversions and the character conversion according to the UNICODE, ASCII, and EBCDIC formats.

Adabas allows field names to include hyphens, as shown in Example 9-1. The tool must convert these field names to the accepted DB2 for z/OS standards.

Example 9-1 Adabas fields names

VIEW	:	CUSTOMERS	DEF.SEQ:	DBID:1	FNR:	45
COMMAND:						
I T L	DB	NAME	F	LENG	S	D
						REMARK
- - - -		-----bottom-----	- - - -	- - - -	- - - -	- - - -
	1	CU CUSTOMER-NAME	A	8.0		D
G 1	SZ	STATE-ZIP				
	2	ST STATE-NAME	A	2.0		D

The NULL value is treated differently between the databases. Adabas defines blank alphabetic fields and zero numeric field as “null”. In DB2, these two values are neither zero or blank, it is unknown. A solution to address this difference is to define the new columns by using NOT NULL WITH DEFAULT.

With alphanumeric fields, Adabas allows for storing a field that is longer than its definition; therefore, you use the VARCHAR data type to manage the field.

Performance

Before you start the process to convert to DB2 for z/OS, you must consider the following issues:

- ▶ Too much de-normalization of the physical model might cause DB2 for z/OS performance degradation.
- ▶ Apply filter factors to avoid reading unnecessary data.
- ▶ Keep the same indexes used in Adabas and improve the indexes whenever possible.
- ▶ Understand the DB2 sort process.
- ▶ Avoid the use of cursors when only one row is retrieved.

- ▶ Avoid the use of numerous joins because of the creation of Natural data definition modules (DDMs).
- ▶ Use CURSOR for UPDATE only in the necessary columns.

Applications

The conversion of the Natural procedural language is one of the most challenging steps for the migration. This conversion is challenging because of the manner in which Natural compiles all of the objects (MAP, PROGRAMS, SUBPROGRAM, DDM, and so on), commands, and when accessing the Adabas database.

The customer chooses to migrate Natural programs to COBOL, Java, or EGL language, but this scenario leads to manually rewriting many programs. Alternatively, a customer chooses to keep running Natural programs, and access DB2 for z/OS through a transparency layer solution. The conversion is simpler by using COBOL programs that access Adabas because only the ADASQL calls must be changed to standard SQL statements to be pre-compiled.

In addition to the changes required to convert Adabas calls to DB2 SQL, there might be application logic changes required as a result of the DB2 design. The following application changes result from data changes made to resolve differences between an inverted list, non-relational database, and a relational database:

- ▶ Normalization affect
- ▶ Record splitting
- ▶ Repeating groups
- ▶ Record to multiple tables
- ▶ Translating group level fields
- ▶ Perform the READ LOGICAL by using DB2 cursors

In addition, the following tasks must be applied:

- ▶ Convert each Adabas (Natural) program to a corresponding DB2 program wherever possible.
- ▶ Do not include new functions.
- ▶ Determine the naming conventions.
- ▶ The check function cannot be converted.

9.3 ConsistADS automated conversion solution

In Chapter 7, “Conversion solutions” on page 73, we described the benefits of using automated tool solutions during this conversion process. There are many such solutions available to help customers who decide to convert their Natural applications and Adabas to DB2 for z/OS.

Consist Software Solutions Inc. (Consist) developed a solution that combines the benefits of the conversion and transparency methodologies. Consist examined numerous approaches to the conversion and designed a solution based on the following demands of the client:

- ▶ Faster conversion
- ▶ Lower risk
- ▶ Increased cost savings
- ▶ Automatic functionality enhancement of the applications

Conversions that are conducted by Consist are highly successful and the company is positioned as specialists in IBM DB2 for z/OS when a low-cost and short conversion period is required.

The important decision for any customer is to choose a solution that replaces Natural and Adabas from a vendor capable of providing an end-to-end automated conversion solution. A vendor also must have extensive experience in working with Natural and Adabas, converting applications to a DB2 environment, and a working knowledge of application development in the System z environment.

Consist offers to their customers Consist Advanced Development Solution (ConsistADS) to convert Natural and Adabas to DB2 for z/OS. Consist has years of experience in delivering solutions that are written in Natural and Adabas in the System z environment.

ConsistADS is an end-to-end conversion solution that includes DB2 for z/OS and several DB2 Tools as part of a packaged solution that is delivered to the customer.

The ConsistADS solution minimizes the risk factor to and concerns of a customer during the conversion from Natural and Adabas to DB2 for z/OS by using the Consist automated conversion engine.

Combining methodologies: This solution uses a combination of conversion and transparency methodologies. The Adabas database is converted to DB2 for z/OS and the Natural language is converted to the language used by Consist, which is named Advance. The Advanced language runs inside the runtime environment that is provided by the solution.

9.3.1 ConsistADS solution overview

ConsistADS is a solution that delivers complete automatic conversion and execution of business applications that are written in Natural and Adabas. ConsistADS with DB2 for z/OS offers a set of tools that automatically runs the conversion from ADABAS Database to DB2, and converts NATURAL programs, including mapping the original Adabas access calls to SQL commands.

ConsistADS with the ConsistDMS component integrates DB2 for z/OS database and converts the ADABAS hierarchical data file structures into DB2 SQL Data Manipulation Language (DML) objects.

Included in the ConsistADS solution is an Integrated Development Environment (IDE) platform that is accessible through a web browser. By using the IDE, the client maintains existing converted applications and performs future application development to add new functions (such as Business Analytics modules) onto the converted applications.

ConsistADS preserves the accumulated application development expertise of the client because ConsistADS enables a client to maintain converted and new applications in the original source code in the z/OS platform.

ConsistADS IDE combines functionality that is used by any organization. ConsistADS IDE also allows the organization to develop business applications quickly and efficiently by realizing the benefits of the latest current information technology.

Editing, compiling, debugging, installing, and distributing the functions of the ConsistADS objects allows customers to develop and test an application in a z/OS controlled environment.

This flexibility results in IT cost reductions and increased productivity by optimizing System z resources. Organizations often realize the following benefits of ConsistADS:

- ▶ Preserves existing application investments
- ▶ Automatically converts applications from the original sources
- ▶ Converts ADABAS hierarchical data into DB2 relational structures
- ▶ Provides a robust communication layer for efficient integration to optimize resources and maximize the performance of an application
- ▶ Provides thorough security by using ADS modules and IBM RACF software
- ▶ Replaces the use of TP monitors by delivering online web services directly
- ▶ Transparently provides automatic web functionality to applications
- ▶ Offers an IDE development platform with access through a web browser
- ▶ Allows the use of other development environments for editing and maintaining existing programs
- ▶ Controls the versions in distributed and collaborative environments
- ▶ Supports sysplex environment, ensures scalability, and supports large numbers of concurrent users
- ▶ Delivers remote development option which allows for better utilization of resources and increased support for the IBM Mainframe® z/OS platform
- ▶ Offers optional Business Analytics integration in addition to components for implementing SOA and integrating Business Process Management (BPM) tools

9.3.2 ConsistADS solution

With ConsistADS suite, Natural and Adabas-based application systems are fully migrated and modernized. In addition to saving significant licensing and maintenance costs, customers benefit from DB2 for z/OS database and modern web User Interface technology.

ConsistADS is a complete IDE suite that features a compiler, runtime system, and migration toolset. The special feature of ConsistADS is the capability to automatically migrate and run multiple application systems that are coded in Natural in a single conversion step.

ConsistADS provides all of the tools necessary to preserve the original investment in mature applications that are written in Natural code. The source code automatically is converted to Consist Advance, the 4GL language of Consist that is compatible with Natural. The full scope of the functionality of the language is run in ConsistADS with DB2 for z/OS.

ConsistADS is an IDE platform that includes the Advance language, the RDBMS, and the ConsistDMS components for the conversion of the Adabas data to an RDBMS, such as DB2 for z/OS. Ancillary DB2 tools are offered and packaged directly by Consist under a global OEM software partnership with IBM for the integration of IBM products into ConsistADS.

Customers experience the following benefits of the ConsistADS solution with IBM software:

9.3.3 DB2 for z/OS tools

Consist includes a DB2 for z/OS license in their ConsistADS solution and (at the request of the customer) important DB2 for z/OS tools. The DB2 tools are the part of the solution that provides the following system management tasks that are related to DB2 for z/OS:

- ▶ Application management
- ▶ Backup and recovery
- ▶ Business intelligence and dynamic warehousing
- ▶ Data governance
- ▶ Data replication
- ▶ Database administration and change management
- ▶ Performance management
- ▶ Utilities management
- ▶ Version upgrade acceleration

For more information about DB2 for z/OS tools, see this website:

<http://www.ibm.com/software/data/db2imstools/products/db2-zos-tools.html>

The following set of DB2 for z/OS tools is provided by Consist at the request of the customer:

- ▶ DB2 Administration Tool for z/OS
This tool simplifies the complex tasks that are associated with safely managing DB2 objects and schema throughout the application lifecycle with the least possible effect on availability.
- ▶ Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS
This tool evaluates the efficiency and optimizes the performance of the DB2 for z/OS environment.
- ▶ DB2 Utilities Suite for z/OS
The utilities suite minimizes the downtime that is associated with routine DB2 data maintenance and ensures the highest degree of data integrity.
- ▶ DB2 SQL Performance Analyzer for z/OS
This tool helps improve DB2 application design to achieve maximum productivity by testing different “what if” scenarios.
- ▶ DB2 SQL Tuning for z/OS
The DB2 SQL Tuning for z/OS tool enables efficient customization and tuning of SQL workloads and DB2 objects.
- ▶ DB2 Query Monitor for z/OS
The tool enables the customer to efficiently customize and tune the SQL workload and DB2 objects.

ConsistADS is packaged with IBM DB2 for z/OS and related DB2 tool products in the ConsistADS/DB2 for z/OS solution set.

9.3.4 The ConsistADS products

As shown Figure 9-1 on page 119, the following tools are included in the ConsistADS solution architecture:

- ▶ Runtime system
- ▶ Monitoring system
- ▶ Common Web Architecture (CWA)

- Compiler for applications that are written in Natural or Advance
- Editing software for the web
- Data migration system
- Security application
- Integration infrastructure
- Web interface customization tool

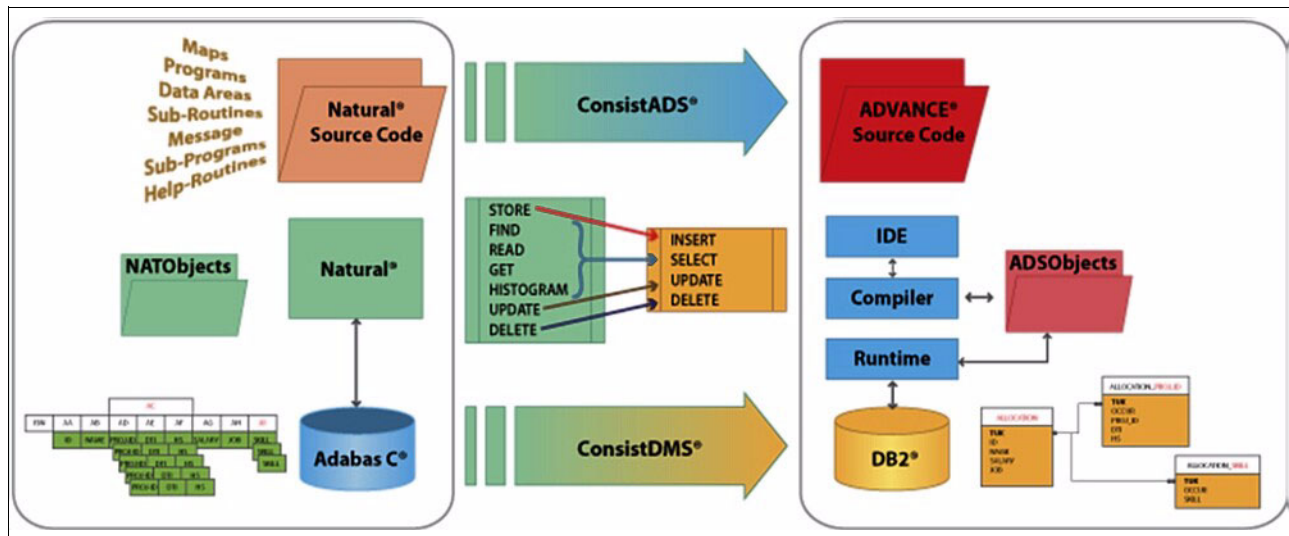


Figure 9-1 ConsistADS solution architecture

The ConsistADS components are described next.

ConsistADS runtime system

The ConsistADS engine system runs the application programs. The system operates in batch or online modes and includes a mode for batch execution and a mode to run under the ConsistADS monitor.

The runtime system performs all of the standard functionality of the languages, such as calculations, file access, device access, database access, and data movement.

The ConsistADS runtime system configuration is provided through configuration files, which are maintained by the ADS utility.

ConsistADS monitor for mainframes

The ConsistADS Monitor is the component that is used to communicate with the web server responsible for the user interface to run online programs. The communication is with a Transmission Control Protocol/Internet Protocol (TCP/IP) set of connections that establishes a link between the web server and the monitor threads.

ConsistADS software operates in a distributed, multiuser, and complex environment and provides a high level of security that is implemented in the solution. Security is provided through a set of layers that support security constraints to avoid any data breach. The security control is provided through the ConsistADS monitor interfacing with RACF.

ConsistADS subprograms are integrated through web services. Each subprogram is converted into a Java class and those Java classes are packaged and accessible through Apache Axis2. This tool manages maps that are customized by HTML Designer (an editor) and changes the properties of HTML elements. The tool also compares maps to detect the

differences between two library versions of the maps and then graphically displays those differences.

ConsistADS-IDE

ConsistADS-IDE is a graphical IDE used for software application development and maintenance of source code through a web interface. As shown in Figure 9-2, the tool provides comprehensive facilities such as, authoring, modifying, compiling debugging source code software, and deployment of compiled software.

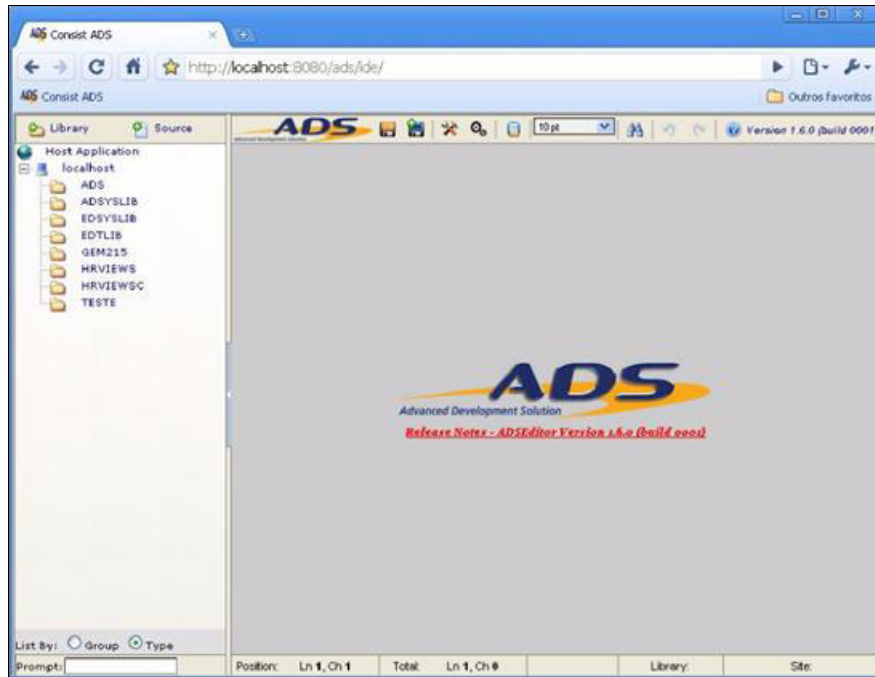


Figure 9-2 ConsistADS-IDE environment view

ConsistADS - IDE Toolbar

As shown in Figure 9-3, the toolbar contains all the following function buttons necessary to develop or maintain source code, including compilation and execution functions:

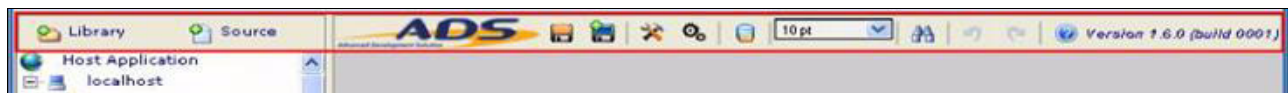


Figure 9-3 The ConsistADS toolbar

- The Add Library button (as shown in Figure 9-4) creates a library in the server.

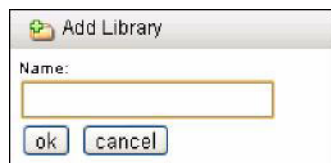


Figure 9-4 ConsistADS-IDE Add Library

- The New Source button (as shown in Figure 9-5) creates a New Source object for editing.



Figure 9-5 ConsistADS IDE New Source

- The SAVE button (as shown in Figure 9-6) saves the edited object. If it is changed by another user, a confirmation message with the option to rename or change the library is displayed.



Figure 9-6 ConsistADS IDE SAVE

- The button SAVE AS (as shown in Figure 9-7) allows a developer to save the new or updated source object with a different name and select a new library, if it is required.

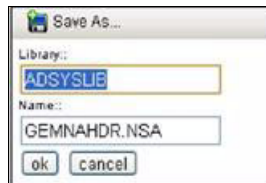


Figure 9-7 ConsistADS IDE SAVE AS

- The COMPILE button (as shown in Figure 9-8) is used to compile a selected source object, and displaying any messages in the editor footer. If compilation errors occur, a cursor is positioned on the error line and location of the source object.



Figure 9-8 ConsistADS IDE: COMPILE

- The RUN button (as shown in Figure 9-9) runs the program in a new browser window. The object type determines the run environment. If the object type is a program, the program runs under the terminal context (for example, 3270 on web UI). If the object type is a web subprogram, the program runs under the web UI context.



Figure 9-9 ConsistADS IDE RUN

ConsistADS-Data Migration System

An overview of the ConsistADS data migration process is provided by using the Data Migration System (DMS) tool. This tool complements the information that is provided in the DMS help system and the DMS manual. The tool guides the user through the steps to complete the migration process.

DMS overview

The DMS is a core component of the ADS solution that is used to migrate data from Adabas to a DB2 database. DMS is implemented by using the Common Web Architecture (CWA) environment. DMS contains its own database in which the migration control information is stored and ensures a complete and accurate migration from the original ADABAS hierarchical files to the DB2 target database. DMS also provides the specific information that is needed to allow the original application to run and access the new DB2 database.

DMS environment

DMS delivers the components that automatically convert ADABAS hierarchical files into a DB2 target database in a single conversion process. The following components are included in DMS:

- **DMS data repository**

This repository is the DMS catalog that contains all of the information that is needed for the conversion process. The repository also contains the information about the source and target environments, such as DDMs and fast data transfers (FDTs). The new data model is stored in the tables of the data repository.

- **Project**

The Project concept allows different developers to work with different sets of data in a segregated manner. A Project represents a logical data partition in the DMS Data Repository that corresponds to a set of objects, such as DDMs, FDTs, modeling, tables, and maps, that are related to a specific application or conversion process.

- **Logical files**

A logical file defines the mapping of tables and columns between an original ADABAS file and the target relational data model. Through logical file abstraction, DMS models ADABAS multi-files into several logical views that conform to a new relational database paradigm.

- **Modeling**

A developer defines the relational target structure and relationship between the table objects.

The DMS provides a wizard that defines a standard model that is based on a set of fixed rules. For example, each periodic group has a correspondent child table that includes all of the periodic fields. The super-descriptors are defined as extra fields in the main table.

In addition, DMS offers a modeling process that is customized. A developer defines a set of conversion rules to allow different target specifications for the same original field or group of fields. For example, one developer might want to keep the contents of a multi-valued field in a single column in the relational table. The developer instead might choose to split the first five occurrences of a particular multi-valued field into five different fields (table columns).

- **Catalog tables**

Catalog tables contain the complete target relational database structures and all of the information about the target DB2 environment, such as, tables and columns, keys, referential integrity, and index specification.

- **Mapping**

This component generates the map file DDMMAPPING that is used by the ADS compiler to translate ADABAS commands to SQL statements.

- **Models and Templates**

Several parts of the conversion process are done through scripts execution by using JCL. DMS generates these JCL scripts that are run during the migration process.

- **Batch subsystem**

The DMS system uses the CWA batch subsystem environment to conduct several conversion tasks, such as, importing and generating. Those batch procedures are started automatically and are analyzed by using the CWA batch processing options.

- **Source environment**

This environment contains the ADABAS/NATURAL environment in which the data is unloaded and converted. The environment also includes the FDT applications and DDM from the source environment that used in the modeling process. The DMS system generates all of the JCL scripts for the unload process and the conversion programs that arrange the data according to the new database relational model.

- **Target environment**

The new DB2 relational database data is loaded into the target environment. All DDL that is required to create the target database, table objects, and JCL scripts that are used in the load process are generated by the DMS system.

- **Server environment**

The ConsistDMS system works under the CWA environment of Consist so that the user runs client server applications through the web.

- **Local environment**

The web browser runs under the local z/Linux or z/OS UNIX System Services (USS) environment.

- **Conversion programs**

These programs are a set of programs that are automatically generated to convert the data from the source model (ADABAS) to the target model (relational) according to the data modeling specified for each file. The conversion programs read the unloaded and decompressed data from the ADABAS environment to generate the sequential files that are used as input for the corresponding DB2 load utility. All of the JCL that is required to run these batch programs are automatically generated by DMS.

- **Validation programs**

The validation programs are a set of programs that is automatically generated for data conversion validation. These programs read both ADABAS source data files and the new DB2 tables to generate sequential files that are used to compare source and target data files.

- **ADS application environment**

This ADS Library structure includes the system libraries necessary to manage the ADS environment.

9.4 How ConsistADS maps ADABAS data files to DB2

ADABAS has its own database design, such as, data structures or objects. When you convert a database from ADABAS to DB2 for z/OS, the database objects are converted into optimized DB2 table/index objects. The Natural language programs access the Adabas database with different command-calls that need to be translated to SQL calls. These conversions are described in this section.

Adabas concepts: The specific concepts of Adabas are not covered in this book. The focus of the book is on the overall conversion process.

Data design

In this section, we examine data constructs mapping from Adabas to DB2.

Adabas file layouts

The Adabas files are converted to DB2 tables according to the FDT definition.

For the DDMs, the conversion tool creates the VIEWS on DB2.

When Adabas data fields are translated as DB2 columns, the columns must adhere to the following rules:

- ▶ Column names are unique
- ▶ Redefines cannot be used
- ▶ Defined as NULL, NOT NULL, or a default value
- ▶ Fixed data fields and user VARCHAR fields must be reviewed to avoid unnecessary space

Group-level elements (MU/PE occurrences)

Pre-relational databases often use groups of repeating data in the same field to save disk space for data and index pointers. Figure 9-10 shows these groups of repeating data inside an Adabas file.

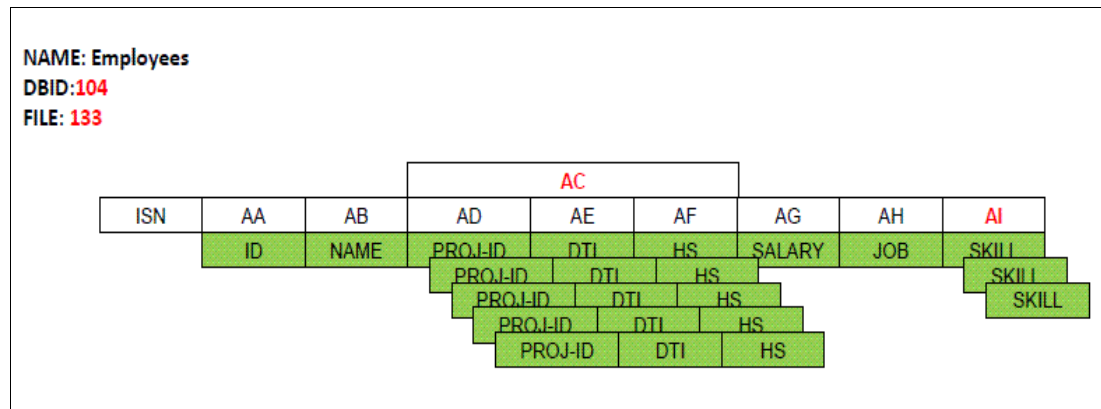


Figure 9-10 Adabas file design

Several approaches are used to resolve these repeating groups by using a relational model. Figure 9-11 on page 125 shows the relational model that was normalized for these groups of repeating data, which results in new DB2 tables. The tables require foreign keys to establish all of the relationships that are declared in DB2 by using the referential integrity rules.

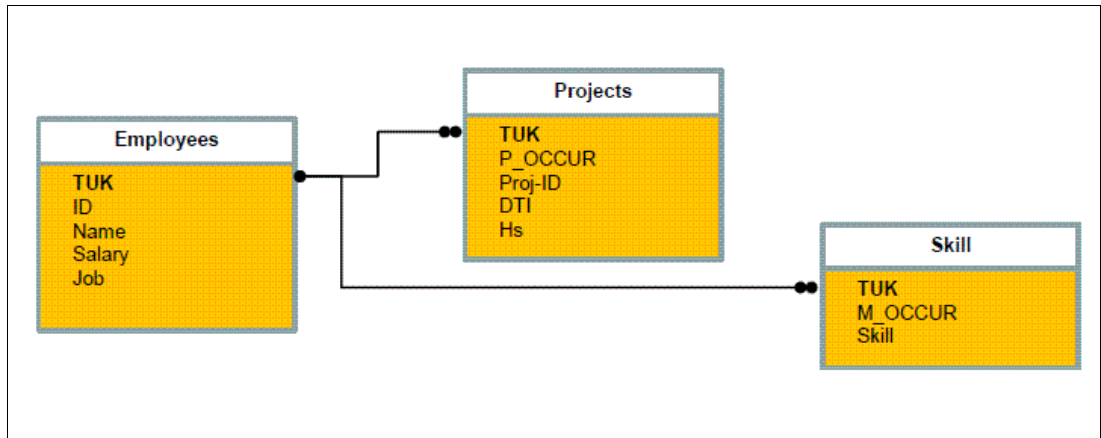


Figure 9-11 Normalized model

The primary and foreign keys are defined in DB2 to maintain those relationships that are created in Adabas. If the keys exist, the keys are left in the code for the conversion.

Referential integrity: For more information about referential integrity in DB2 for z/OS, see *Data Integrity with DB2 for z/OS*, SG24-7111-00.

Other options are available to use according to performance needs and data modelling de-normalization rules.

A denormalized data model is shown in Figure 9-12.

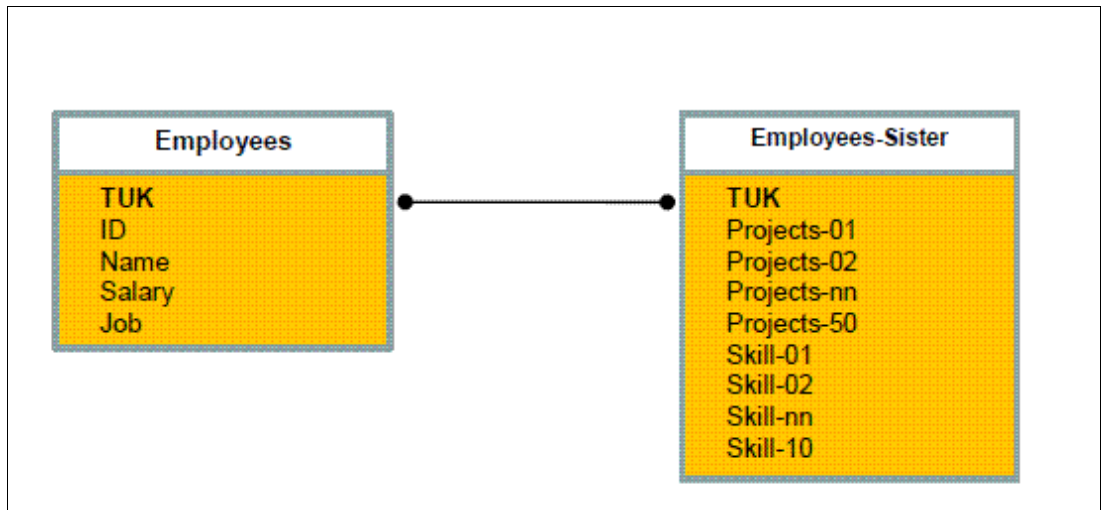


Figure 9-12 The denormalized model

Another option is to use a fully denormalized data model. However, this option causes performance issues, as shown in Figure 9-13.

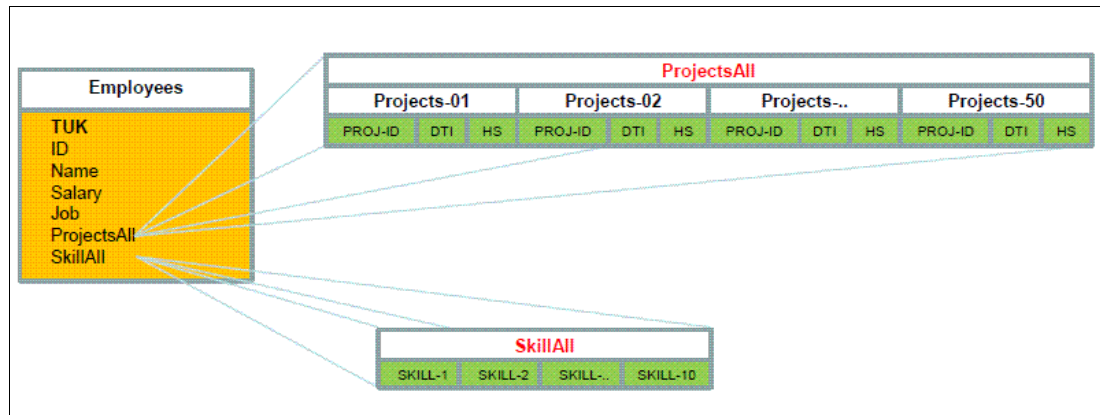


Figure 9-13 Fully denormalized model

Super-Descriptors

Super-descriptors are the equivalent of indexes in DB2 for z/OS. A super-descriptor is a new index that is created and used in programs as search criteria.

Application design

Adabas supports a number of procedural languages, such as COBOL by using ADASQL and direct calls. Natural is the most commonly used language, so we focus more on this language in this section.

Example 9-2 shows the conversion to GET ISN, in which the access is made physically to the record location.

Example 9-2 GET ISN conversion

```
GET *ISN or READ ISN = value
```

```
SELECT FROM ... WHERE ROWID = value
```

Example 9-3 shows the use of the HISTOGRAM command.

Example 9-3 HISTOGRAM conversion

```
HISTOGRAM ...
```

```
SELECT DESCRIPTOR COUNT(DESCRIPTOR)
WHERE DESCRIPTOR => value
GROUP BY DESCRIPTOR
```

The ORDER BY clause might be added as well.

Table 9-2 shows how Natural commands in the application are matched to SQL statements in DB2.

Table 9-2 Natural DML compared to DB2 SQL

Natural DML	SQL
FIND	SELECT
FIND FIRST	SELECT, FETCH FIRST 1 ROW ONLY
FIND UNIQUE	SELECT
FIND SORTED	SELECT... ORDER BY
READ ... LOGICAL	SELECT ... ORDER BY
GET, GET SAME	N/A
READ ... PHYSICAL	SELECT
HISTOGRAM	SELECT ... GROUP BY
STORE	INSERT
UPDATE	UPDATE ... OF CURRENT CURSOR
DELETE	DELETE ... OF CURRENT CURSOR
END TRANSACTION	COMMIT
BACKOUT TRANSACTION	ROLLBACK

Many SELECT calls require a CURSOR on DB2 for z/O. Cursors must be disabled when only one row is retrieved.

The ORDER BY clause requires sort use, unless indexes are used in the columns that are referenced by the ORDER BY clause.

An SQL tuning team must check the DB2 access path, index usage, DB2 resources, and so on.

For more information about creating a list under Tuning considerations, see Chapter 11, “Application and SQL tuning” on page 175.

DB2 for z/OS programming: For more information about DB2 for z/OS programming, see *DB2 10 for z/OS Application Programming and SQL Guide*, SC19-2969.

9.5 ConsistADS conversion process

In this section, a case study example of a ConsistADS conversion is reviewed.

9.5.1 Challenge of the customer

The customer wanted to upgrade their System z servers, modernize their IT environment, and reduce costs.

The customer required the conversion of the Natural applications that are listed in Table 9-3 and the Adabas files that are listed in Table 9-4.

Table 9-3 Natural applications

Application Name	Program	Lines of code
APP1	2574	637885
APP2	520	46281
APP3	489	127884
APP4	75	13100
APP5	1105	266242
APP6	1405	452458
APP7	411	80825
APP8	167	40860
APP9	3682	767663
APP10	357	53771
APP11	225	18091
APP12	304	54861
APP13	812	136671
APP14	202	10474
Total	12328	2707066

Table 9-4 Adabas files

DB ID	Files	Size GB	Data GB	Millions of records
3	176	330	210	1397.0
30	5	123	75	869.0
10	52	6.3	4.1	13.7
15	72	16.3	7.2	30.8
Total	305	475.6	296.3	2310.5

9.5.2 Solution architecture

The customer decided to implement a logical partition (LPAR) that is used in the user acceptance testing environment in an existing z/OS hardware machine. The customer created an LPAR for the production environment in the new z/OS hardware machine, as shown in Figure 9-14 on page 129.

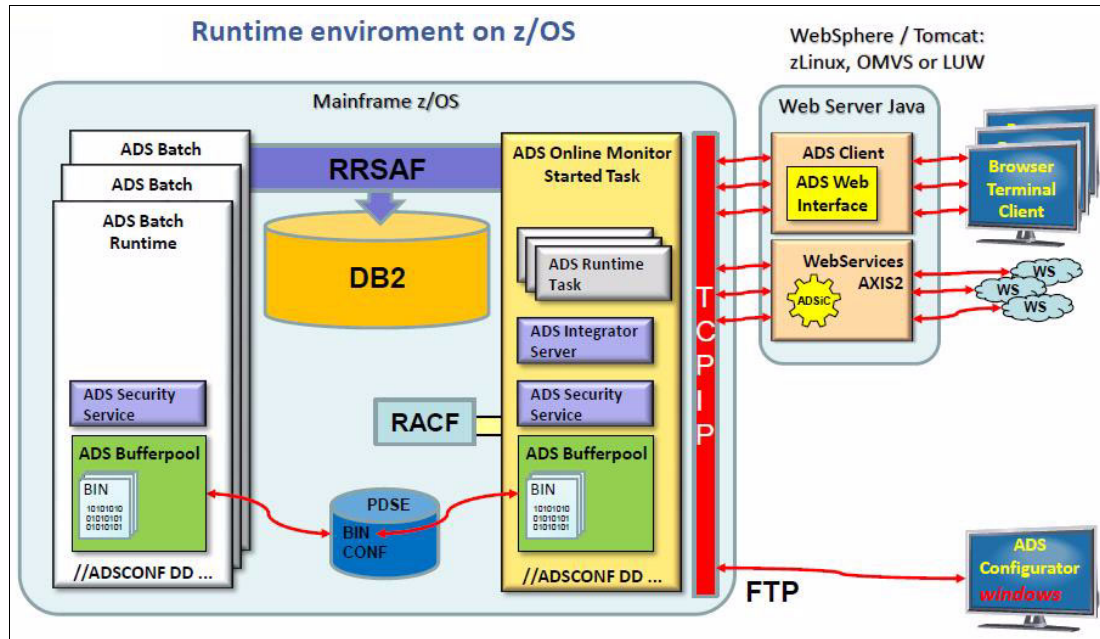


Figure 9-14 Architecture overview

The converted Natural language, named Advance, runs over the runtime environment in batch and online. The interface uses the web server to show web windows instead of showing the Natural maps.

9.5.3 Project phases

The whole project took approximately six months to complete. Table 9-5 shows the amount of time spent on each phase of the project.

Table 9-5 Project timing

Phase	Amount of time
Performing environment analysis	One month
Executing data conversion	One month
User unit testing	Two months
User quality assurance testing	One month
Define JCL streams and batch jobs integration	Two months (conducted in parallel with the two previous phases)
Final user testing and training	One month
Turn over into production	Two days (weekend)

The customer decided to implement each application as described in the methodology for the conversions.

Unique challenges: Each customer environment could pose unique challenges that might affect timing for the project.

The steps of the project included the following details:

- ▶ Project initiating process: Develop the following project charter:
 - Develop stakeholder management strategy
- ▶ Project planning process: Develop Project Management Plan.
- ▶ Project running process: Collection and analysis of the following information and requirements at the client:
 - Answer questionnaire
 - Analysis of online applications
 - Database analysis
 - Premises compliance by client
 - Analysis of batch applications
 - Preparation of maps (linking business services with applications)
 - Preparation of diagrams that describe the network configuration
 - Back up generation of initial application
 - Back up generation of initial DB
 - Back up generation: JCL, PROC, and PARAM
 - Environment of migration at Consist:
 - First cycle of migration - automatic
 - Research and JCL adaptation
 - General tests of first cycle
 - Second cycle of migration - refinement
 - Migrated object packing
 - Quality assurance environment at client:
 - Preparation of the work infrastructure z/OS (LPAR, DB2, RACF, and ADS)
 - Preparation of the work infrastructure z/Linux (Tomcat, Java, and ADS)
 - Test plan development
 - ADS installation
 - ConsistADS mainframe installation
 - ConsistADS Linux installation
 - Installation of application converted
 - DB2 for z/OS installation
 - SMP/E work
 - DB2 subsystem installation
 - DB2 Administration Tool
 - Activate DB2 Administration Tool
 - Data loading
 - Web services integration
 - User access settings
 - ADS training to client (systems analysts)
 - Process tests (systems analysts) and adjustments
 - ADS training to client (users)
 - Process tests (systems analysts) and adjustments
 - Performance tests and adjustments
 - Stress tests (online and batch applications) and adjustments
 - User training
 - Production environment at client, which includes ADS installation, DB2 for z/OS installation, converted application installation, and data load migration
 - Monitoring and adjustments in production:
 - Follow-up and final adjustments.
- ▶ Project controlling process:
 - Project status report
 - Monitor and control the project
- ▶ Project closing process:

- Run project acceptance
- Develop and document lessons learned

9.5.4 Data conversion

The ConsistADS solution uses the DMS to convert Adabas files to DB2 tables. DMS maps the ADABAS calls to SQL commands to access DB2 for z/OS.

DMS contains its own database in which the control information is stored. The contents of DMS ensures a complete and accurate migration from the original Adabas to the target DB2, and allow the original applications to access DB2.

Figure 9-15 on page 132 shows the flow of the steps to conduct the data conversion.

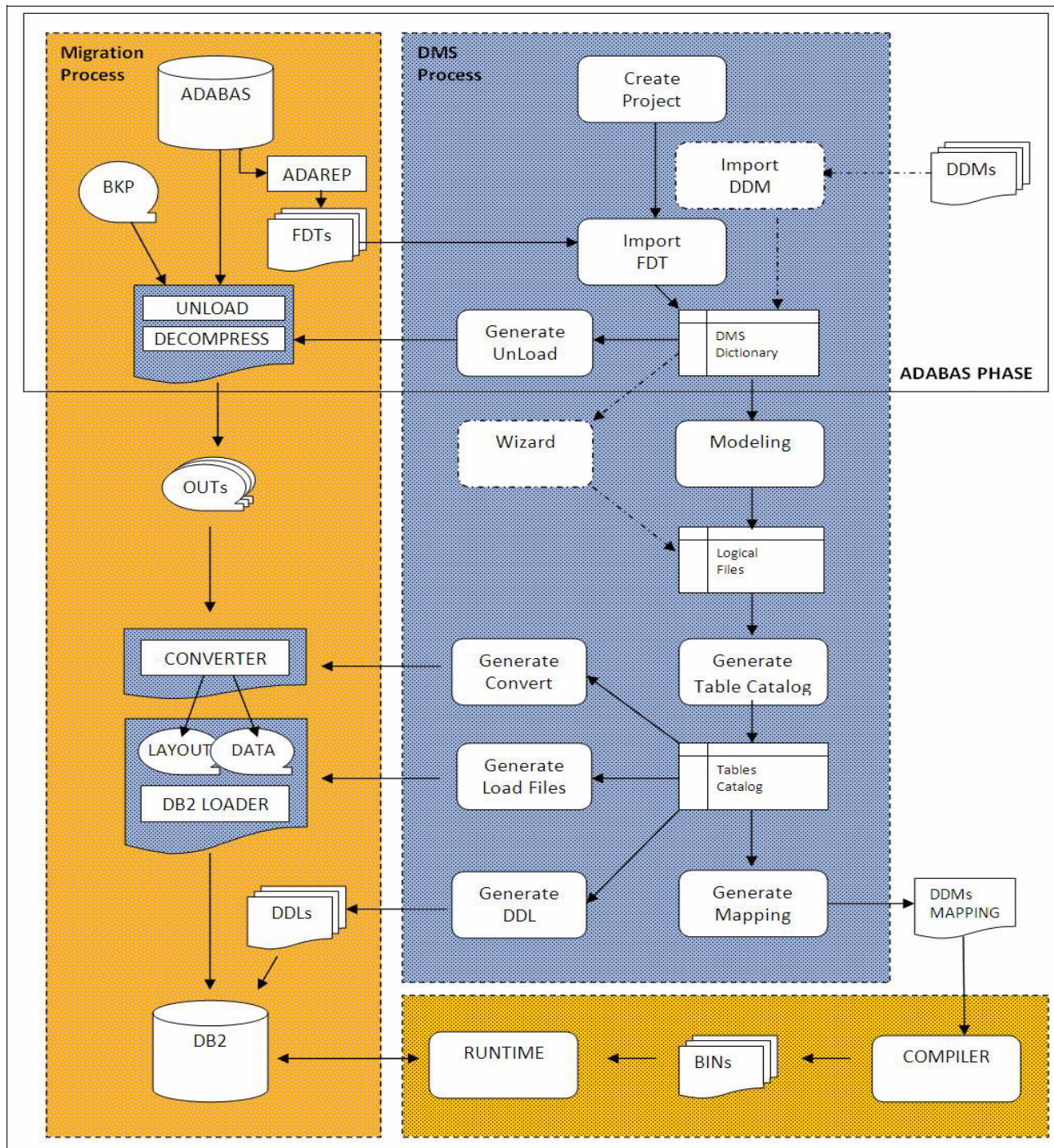


Figure 9-15 DMS tasks

This section describes the steps shown in Figure 9-15:

1. Generate ADAREP from the environment of the customer and gather all Adabas information, such as the FDT and DDMs.
2. Create a project work flow including iterations for the source and target databases.
3. Import FDTs.

The FDTs used in the import process must be generated in the source environment by using the Adabas Report utility. The Report output must be made available either in the local or server environment for use by the import utility.

The imported files are listed after the import process is complete, as shown in Figure 9-16.

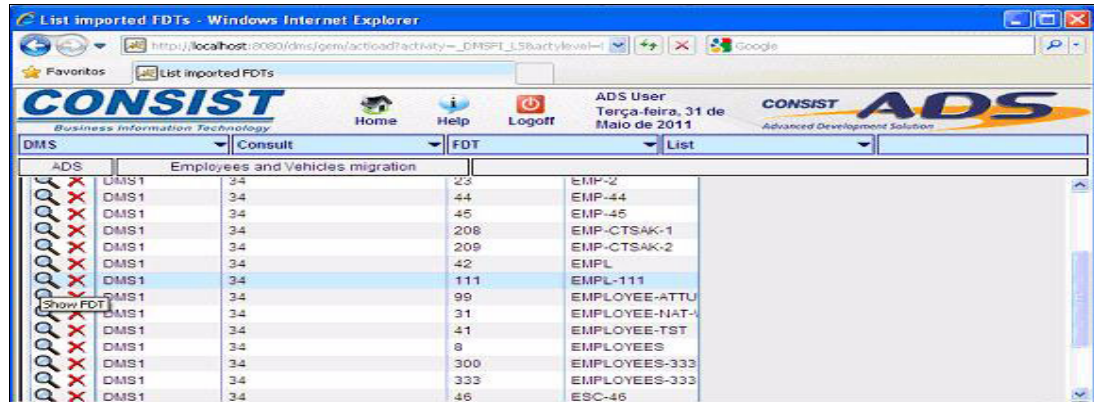


Figure 9-16 List of imported files

The files are opened and inspected, as shown in Figure 9-17.

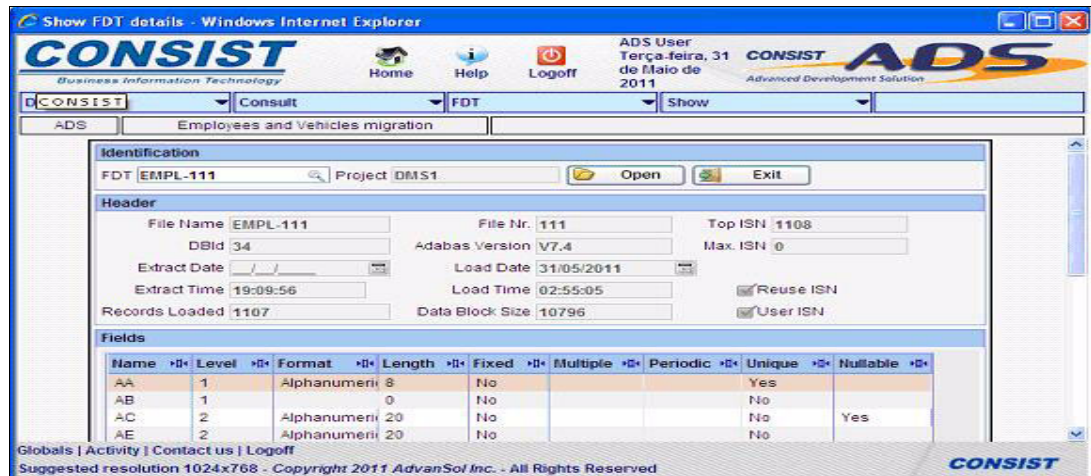


Figure 9-17 Details of the file attributes

Importing DDMs

DDMs are the equivalent of the table views in DB2 for z/OS.

The DDMs must be converted to text format through the DMS Source File Migration function before the DDMs are used by the IMPORT utility. The DDMs must be extracted from the Natural production environment by using the SYSOBJH utility (Transfer Mode). The DDMs are then transferred to the source objects in the development environment by using FTP. Figure 9-18 on page 134 shows the DDM listed in DMS.

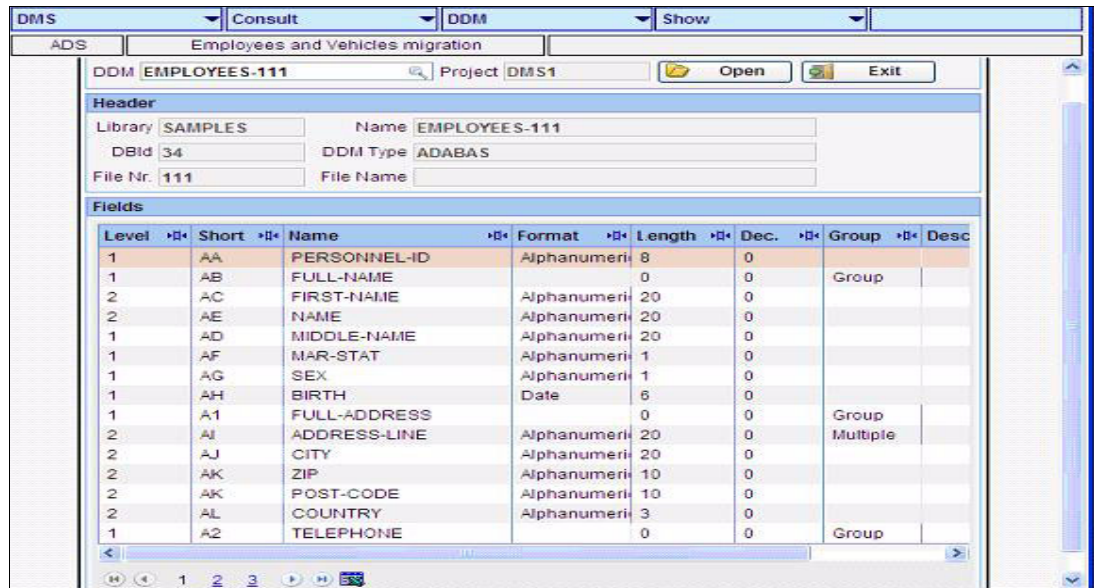


Figure 9-18 DDM imported

Modeling

After the FDTs and DDMs are imported into the DMS system, the Modeling process is started for each imported FDT.

Each modeling process that is run generates a logical file that represents the correspondence between the original Adabas file and the new relational data mode. ADABAS allows for a single FDT to support several DDMs, and a single field in the FDT is reused within several DDMs that serves as the logical record criteria to extract data into DB2.

A logical file defines the relationship between the original Adabas file and the target relational data model in terms of tables and columns. Through the logical file abstraction, we model Adabas multi-files into several logical views in a relational database paradigm.

Figure 9-19 shows the correlation of the DDMs to a data model.

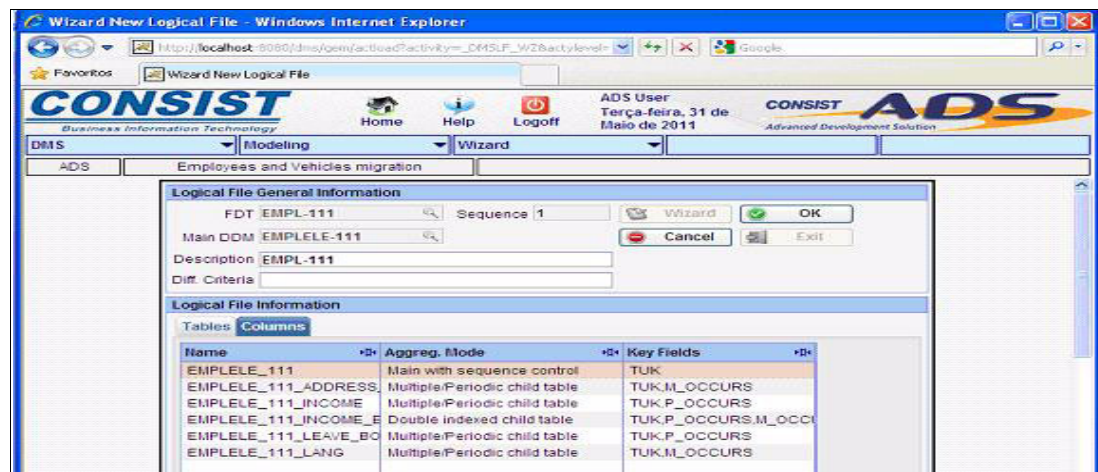


Figure 9-19 DDM and corresponding data model

Figure 9-20 shows the relationship of DDMs fields to each column in the target table.

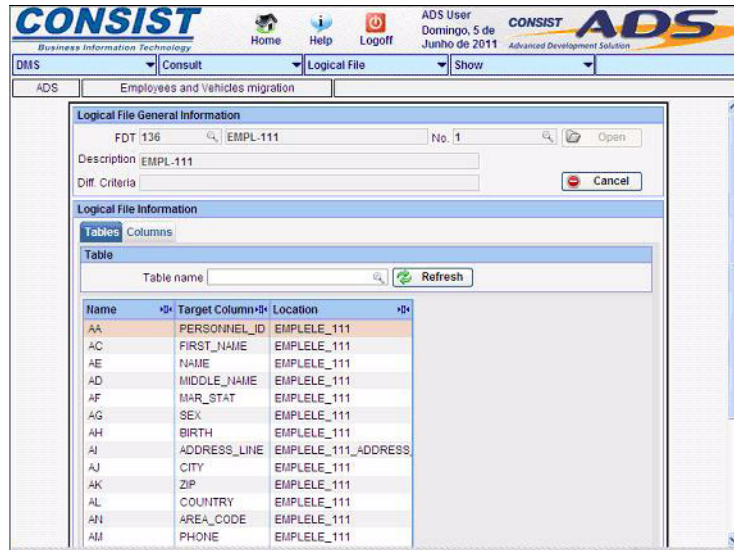


Figure 9-20 Logical file

Generating table catalog

Generating the table catalog is an important function because the metadata tables of the Target Environment in the DMS Data Repository are generated. These tables contain all of the information about the data modeling, which represents the new relational data organization. The DMS Data Repository is extensively used by the data conversion process, and during DDL generation and load process mapping.

To start the process, you select an FDT (the Adabas model) and the corresponding logical file (the new relational model), as shown in Figure 9-21.

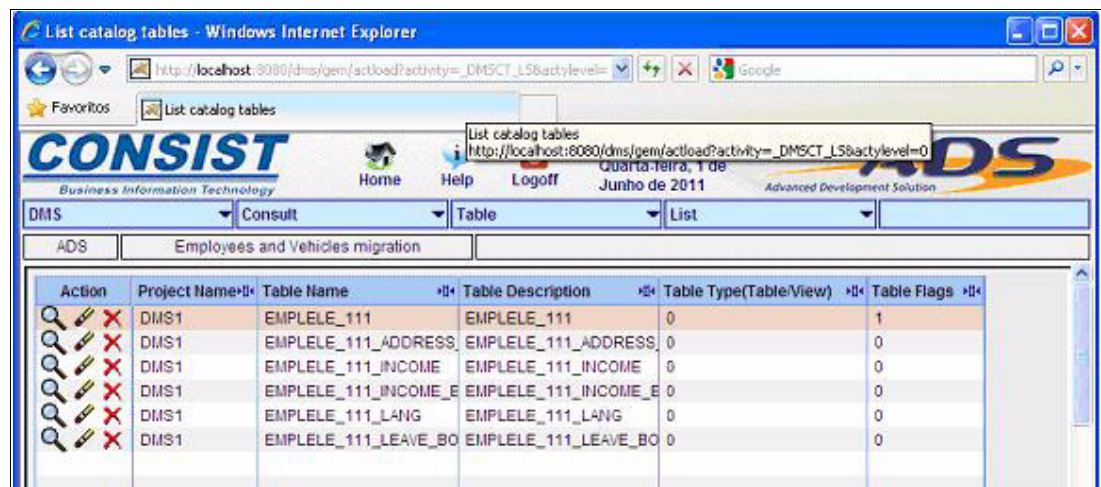


Figure 9-21 Generating DDL

The user generates the DB2 DDL for each logical file, as shown in Figure 9-22 on page 136.

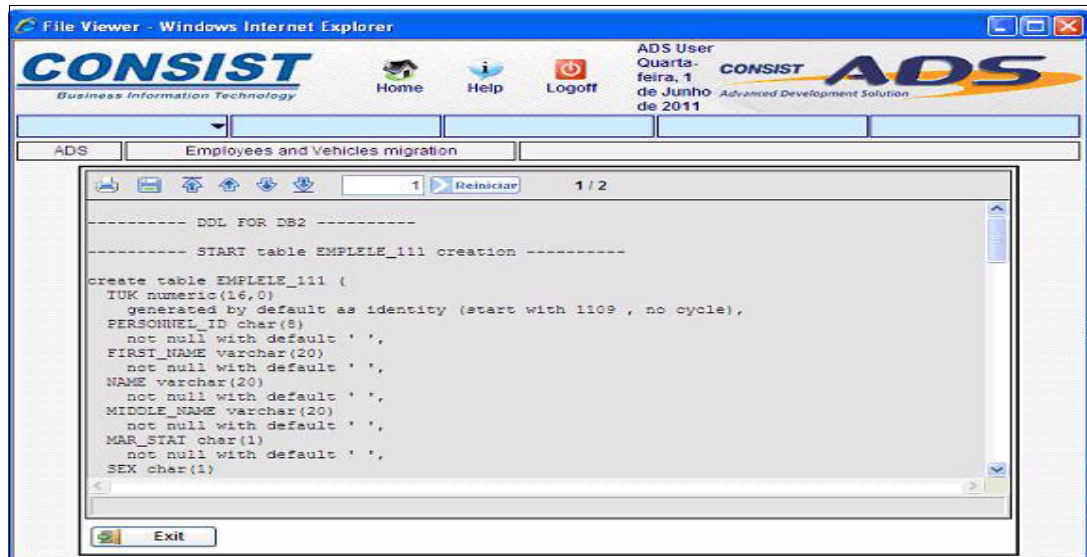


Figure 9-22 DB2 for z/OS DDL

Mapping

The Mapping function generates the DDMAPPING file, which is used by the ADS compiler to replace all of the Adabas call statements with SQL. This replacement is an important component of the application conversion process. Figure 9-23 shows the interface that is used to conduct this process.

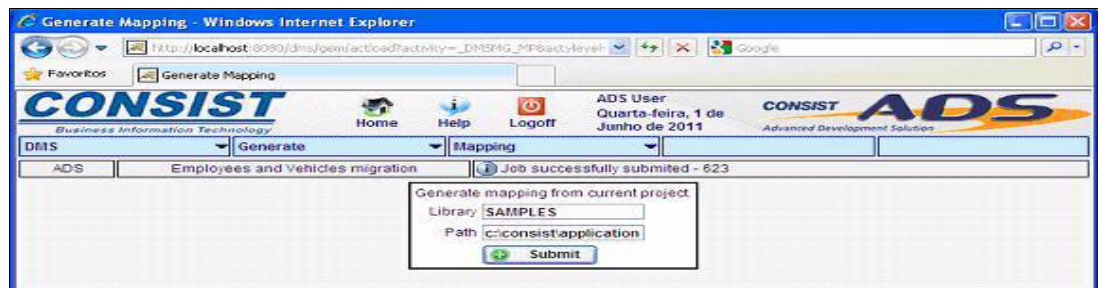


Figure 9-23 Generating mapping

Generating the conversion and validation programs

This function generates the conversion and validation programs. The programs must be cataloged by using the ADS compiler. The corresponding objects are transferred to the target environment for subsequent execution, as illustrated in Figure 9-24.



Figure 9-24 Generating JCL to conversion

This function generates the JCL necessary for the execution of the Conversion and Validation programs as part of the migration process on the source or the target environment.

The following groups of JCL statements must be generated:

- ▶ Conversion: Generates the programs to create the unload data
- ▶ Unload/Decompress: Unloads the Adabas by using the new data modelling
- ▶ Load: Inputs the data into DB2 for z/OS
- ▶ Validation: Validates the process

After the generation process is complete, the JCL statements must be transferred to the source and target environments for execution.

Figure 9-25 shows an example of selecting a file to generate the unload process. The customer provides the number of occurrences and the arguments for space allocation on the appropriate fields.

The screenshot shows the 'Generate JCL for Unload/Decompress' window in the CONSIST ADS application. The window has a navigation bar with 'DMS', 'Specific Activities', 'Mainframe', 'JCL', and 'Unload/Decompress' tabs. The 'JCL' tab is selected, and the 'Unload/Decompress' sub-tab is active. The main content area is divided into two sections: 'Identification' and 'Arguments'.

Identification Section:

- FDT: 136
- EMPID: 111
- DB_Id: 34
- FNR: 111
- Buttons: Submit, Cancel

Arguments Section:

Field Name	Occurrence	Mandatory	Space Type	Primary Alloc.	Secondary Alloc.
AI	5	No	TRACKS	15	0
AQ	5	Yes	TRACKS	15	0
AT	0	Yes		0	0
ANY	0	No		0	0

Below the table, there are input fields for:

- Occurrence: 5
- Space Type: TRACKS (dropdown menu)
- Primary Alloc.: 15
- Secondary Alloc.: 5
- Buttons: Change, Submit, Cancel

Figure 9-25 Creating unload process

The JCL that is run in z/OS environment is shown in Figure 9-26.

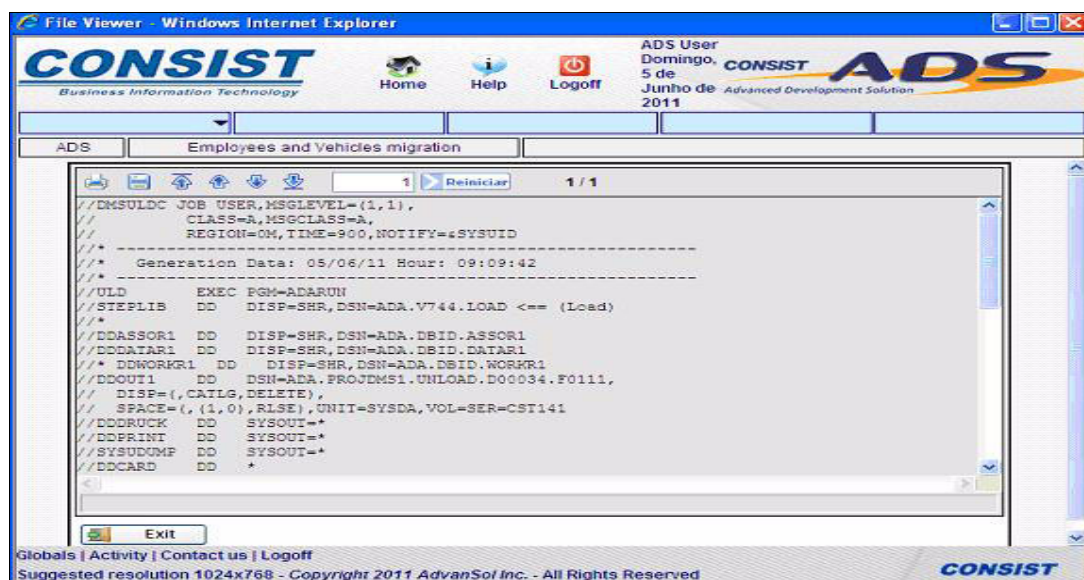


Figure 9-26 Generated JCL

After these steps are completed, it is possible to start the conversion of the Natural language as described in the next section.

9.5.5 Converting the application

To start the code conversion from Natural to Advance, you must send the SYSTRANS to generate all of the Natural source codes. ConsistADS reads the source code and generates the binary object in ADS with the SQL statements.

Example 9-4 shows the source code of a Natural program before the conversion.

Example 9-4 Natural source program

```
define data local
  1 v view of SPE-NOTIFY-CAMREG
  2 CAN-NSS-OFI
  1 counter (i4)
  1 cont2 (i4)
  1 read1 (i4)
end-define
write 'PGM - BEGIN'
reset counter cont2 read1
read v by isn
  add 1 to read1
  if CAN-NSS-OFI >= 0 then
    Update
    add 1 to counter
    add 1 to cont2
    if counter = 1000 then
      commit
    write 'commit record:' cont2 ' read nreg:' read1
    reset counter
```

```

    end-if
  end-if
end-read
write 'FINAL NREG:' cont2 'read nreg'read1
end

```

ConsistADS compiler generates dynamic or static SQL by creating assembler modules.

Example 9-5 shows the generation of dynamic programs.

Example 9-5 Program converted to dynamic Advance language

```

1 v view of SPE-NOTIFY-CAMREG
  2 CAN-NSS-OFI
  1 counter (i4)
  1 cont2   (i4)
  1 read1   (i4)
end-define
write 'PGM - BEGIN'
reset counter cont2 read1
ADS_LB1.
SELECT CAN_NSS_OFI,SPE_EMP_NRP,SPE_TPO_REG,SPE_EMP_NRP,
      INTO V.CAN-NSS-OFI,
           SPE_SUP_NSS_NRP.SPE_SUP_NSS_NRP_SPE_EMP_NRP,
           SPE_SUP_TPO_NSS_NRP.SPE_SUP_TPO_NSS_NRP_SPE_TPO_REG,
           SPE_SUP_TPO_NSS_NRP.SPE_SUP_TPO_NSS_NRP_SPE_EMP_NRP
FROM SPE_NOTIFI_CAMREG ORDER BY SPE_NOTIFY_CAMREG.TUK
      FOR UPDATE OF CAN_NSS_OFI,SPE_SUP_NSS_NRP,SPE_SUP_TPO_NSS_NRP
ADD 1 TO READ1
IF V.CAN-NSS-OFI >= 0
  UPDATE SPE_NOTIFI_CAMREG SET CAN_NSS_OFI = V.cAN-NSS-OFI,
    SPE_SUP_NSS_NRP=SPE_SUPNSS_NRP.SPE_SUP_NSS_NRP_B,
    SPE_SUP_TPO_NSS_NRP.SPE_SUP_TPO_NSS_NRP_SPE_EMP_NRP_B
  WHERE CURRENT OF CURSOR(ADS_LB1)
  ADD 1 TO CONTADOR
  ADD 1 TO CONT2
IF COUNTER = 1000
  COMMIT
  WRITE 'commit nreg:' CONT2 'read nreg:' READ1
  RESET COUNTER
END-IF
END-IF
END-SELECT
WRITE 'FINAL : 'CONT2 'read nreg:'READ1
END

```

Example 9-6 shows the generation of static assembler modules.

Example 9-6 Program converted to Static Assembler

```

*=====
* Object   : PGM
* TimeStamp: NRUNSTPL13514194
*=====
ADSWK01 DSECT
*
*=====
*      Parameters
*=====
P0000001 DS    P'99999999999'
P0000002 DS    CL17
P0000003 DS    CL18
*
*=====
*      Variables
*=====
V0000001 DS    P'99999999999'
V0000002 DS    P'99999999999'
V0000003 DS    CL11
V0000004 DS    P'9'
V0000005 DS    CL11
*
*=====
*      SQL cursors
*=====
*
      EXEC SQL DECLARE CADS0001 SENSITIVE STATIC SCROLL CURSOR WITH X
      HOLD FOR SELECT SPE_NOTIFI_CAMREG.TUK,CAN_NSS_OFI,      X
      SPE_EMP_NRP,SPE_TPO_REG,SPE_EMP_NRP FROM      X
      SPE_NOTIFI_CAMREG ORDER BY SPE_NOTIFI_CAMREG.TUK FOR      X
      UPDATE OF CAN_NSS_OFI,SPE_SUP_NSS_NRP,      X
      SPE_SUP_TPO_NSS_NRP
LBOP0001 DS    OH
      EXEC SQL OPEN CADS0001
LBFT0001 DS    OH
      EXEC SQL FETCH CADS0001 INTO :V0000001,:V0000002,:V0000003,      X
      :V0000004,:V0000005
LBCL0001 DS    OH
      EXEC SQL CLOSE CADS0001
*
      EXEC SQL UPDATE SPE_NOTIFI_CAMREG SET CAN_NSS_OFI=:P0000001,      X
      SPE_SUP_NSS_NRP=:P0000002, SPE_SUP_TPO_NSS_NRP=      X
      :P0000003 WHERE CURRENT OF CADS0001
*
*=====
*      Finalize
*=====
      EXEC SQL INCLUDE SQLCA
*
      END

```

As shown in Figure 9-4 on page 138, Figure 9-5 on page 139, and Figure 9-6 on page 140, you use the programs to access DB2 for z/OS dynamically or in static mode.

The DDMMAPPING function, as described in “Mapping” on page 136, is a critical step to accurately generate the application code. Example 9-7 shows the DDMMAPPING for the code.

Example 9-7 Mapping for the source programs

```
[FILE=TABLES]
SPE-NOTIFI-CAMREG = SPE_NOTIFI_CAMREG
[FIELDS]
CAN-NSS-OFI      = SPE_NOTIFI_CAMREG.CAN_NSS_OFI
SPE-EMP-NRP      = SPE_NOTIFI_CAMREG.SPE_EMP_NRP
SPE-TPO-REG      = SPE_NOTIFI_CAMREG.SPE_TPO_REG
SPE-SUP-NSS-NRP  = SPE_NOTIFI_CAMREG.SPE_SUP_NSS_NRP
SPE-SUP-TPO-NSS-NRP= SPE_NOTIFI_CAMREG.SPE_SUP_TPO_NSS_NRP
```

Once the application is compiled, both batch and online processes are run in the ADS runtime environment, as shown in Example 9-5 on page 139 and Example 9-6 on page 140.

9.5.6 ConsistADS client results

From the business perspective, the customer sees the following results after the conversion project is completed:

- ▶ Maintenance costs are reduced.
- ▶ Multi-tier architecture and SOA is enabled to improve agility for IT to deliver business requirements.

Table 9-6 summarizes the results of the conversion.

Table 9-6 Results

Before conversion	After conversion
Mainframe model 2096/S03	Mainframe model 2817/504
3 CPUs	3 CPUs
12 GB memory	5 GB memory
2 terabytes on disks	5 terabytes on disks
20 MSUs	20 MSUs
z/OS 1.6	z/OS 1.10
Complete 6.3.1	ADS monitor
Natural 3.1.6	ADS runtime
Adabas 7.1.3	DB2 for z/OS version 9
EntireBroker 7.2	ADS integrator
Natural connection	ADS web interface
Adabas Security 7.1.3	ADS access control / DB2 privileges
Natural Security 3.1.6	ADS security

Before conversion	After conversion
305 Adabas files	459 DB2 tables
2310 millions of records	4859 millions of records
14 applications with 2.7M lines of code (13000 online programs and 600 batch programs)	Same
350 Online Users (with an average of 80-125 concurrent users)	Same
1 Main Office site and 120 remote office sites	Same

9.6 Other solutions in the market

In this section, we describe the following solutions that focus on the conversion to DB2 for z/OS of the languages that access Adabas and the database:

- ▶ OnTarget from Most Software Technologies (MOST)
- ▶ eavRPM solution from ATERAS

The approaches of these solutions are different from the ConsistADS solution described earlier in this chapter. In the ConsistADS solutions, the Natural language is converted to Advance language and requires the runtime that is provided by Consist to run the applications.

The focus of the OnTarget from Most and eavRPM from ATERAS solutions is to convert the language to COBOL or Java that access DB2 for z/OS by using the environment of the customer.

9.7 OnTarget

OnTarget from MOST is a set of tools that automates the following processes:

- ▶ Establishes a DB2 data model to replace the ADABAS data model.
- ▶ Automates migrating the data from ADABAS to DB2.
- ▶ Transforms the NATURAL online application to a three-tier application that runs in a pseudo conversational mode under CICS.
- ▶ Transforms the NATURAL batch applications to COBOL batch.
- ▶ Adapts COBOL programs that access ADABAS through ADASQL to access DB2 through an SQL-based Data Access Layer (DAL).
- ▶ Provides a transparency solution that facilitates COBOL programs that access ADABAS through direct commands to access DB2 through an application-specific DAL.

This solution includes the following features:

- ▶ Automated database design

A normalized relational DB2 data model is automatically designed that is easily fine-tuned by the client. This ability provides an accurate, efficient, and high performance DB2 data model.

- ▶ Automated data migration

OnTarget automatically produces the basic DDL necessary to create the DB2 tables. The solution also generates the programs that migrate the Adabas data to the DB2 tables.

- ▶ Automated software conversion

OnTarget Relational automatically converts COBOL and Adabas and Natural and Adabas by embedding the appropriate SQL commands within the program code. Because the conversion is automated, the conversion is accurate and efficient and quality, consistent code is produced.

- ▶ Integrated refresh mechanism

OnTarget Relational enables refreshing the new DB2 system (the data model and the code), by incorporating any changes that are made to the original Adabas system after the start of the migration process.

9.7.1 Adabas to DB2 for z/OS design

The following Adabas to DB2 conversion issues are automatically accommodated for at the programming language level and the database migration level:

- ▶ Data normalization that is a result of converting from the non-relational database to the DB2
- ▶ Conversion of naming conventions, dashes, and underscore
- ▶ Full support for the transformation of null value in Adabas to DB2
- ▶ Substituting special routines for intrinsic Adabas functions, such as ISN usage
- ▶ Adabas allows the existence of multiple fields (MU) and periodic groups (PE); whereas DB2 manages only normalized data
- ▶ Adabas permits storage of data in compressed format; DB2 displays all data

The following inter-dependencies between the Adabas files and Natural application processing results are analyzed to produce a refined and efficient target database design:

- ▶ Fields not in use by application: The field (optionally) is no contained in the target table.
- ▶ Keys not in use by application: If Adabas keys (SUPER, SUB Descriptors, and so on.) are not used in the application, an INDEX is not created in the target table. This configuration improves the performance and the efficiency of the data migration process.
- ▶ Multiple DDMs in the same field: This configuration is for logical views that are used by the Natural application that define the format length of the field. OnTarget identifies inconsistencies in a field to ensure that the correct setting for target columns is used.

9.7.2 Features of the OnTarget solution

Figure 9-27 on page 144 shows the process that was conducted by MOST, from the analysis phase through the database design, application conversion, and all of the testing phases before production implementation.

The solution includes the following tasks:

- ▶ Automatic analysis of the Natural code and Adabas file structures
- ▶ Confirmation of application inventory
- ▶ Identify exceptional inventory issues
- ▶ Identify migration complexities

- ▶ Derive and estimate required tasks and produce project plans
- ▶ Automated COBOL code generation
- ▶ Full preservation of business logic and user interfaces
- ▶ COBOL three-layer architecture implementation:
 - Presentation layer
 - Business logic layer
 - Data access layer
- ▶ Business logic separation serves as a foundation for by using SOA
- ▶ Transition from Natural Conversational mode to COBOL Pseudo-conversational (CICS) for efficient online mainframe processing
- ▶ Automated conversion of JCL, procedures, and parameters
- ▶ Coexistence capabilities to support phased implementation of not-yet-migrated Natural and Adabas and DB2 for z/OS applications during a multi-phase migration project
- ▶ Automated testing of migrated applications to ensure that the application replicates the functionality and user interface of the original Natural and Adabas system

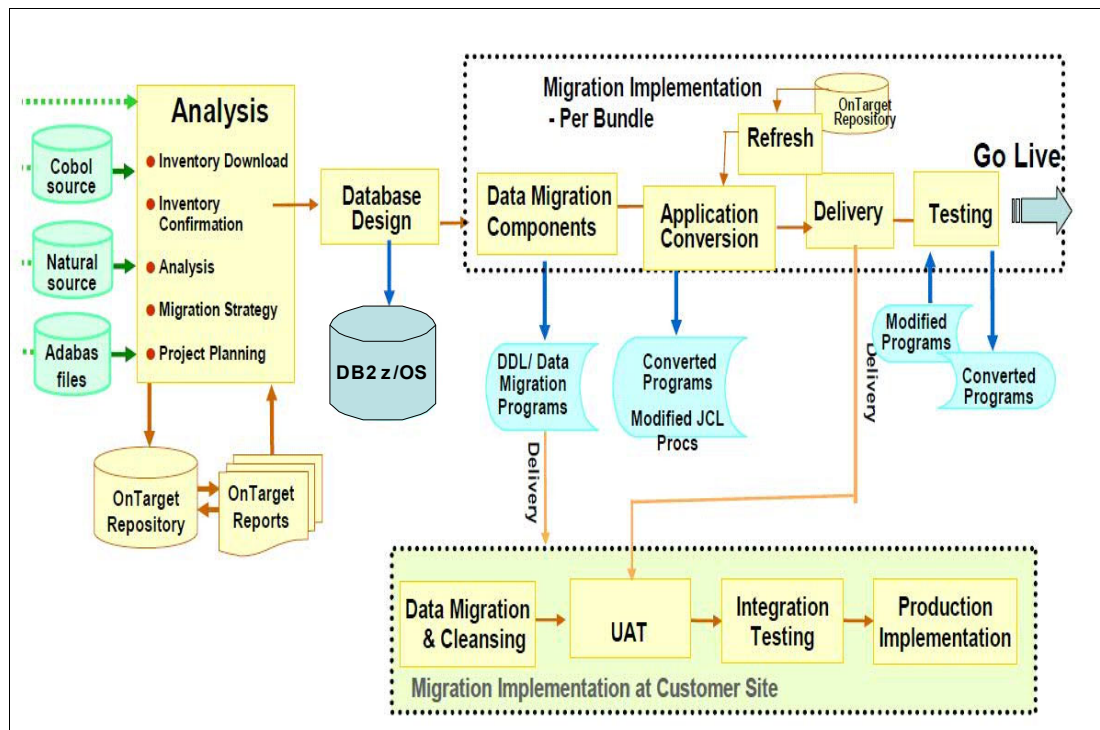


Figure 9-27 Phases of OnTarget

9.7.3 Customer conversion scenario

The next real-world scenario we review is the conversion process that was used by a large US insurance company. The customer wanted to convert the back-office insurance systems of the company from Adabas to DB2 for z/OS.

Challenge of the customer

The customer continued to face increasing costs to maintain the mature SQL access group (SAG) applications. The data was located in a non-relational database, which made the

process of sharing corporate data with other applications in the organization more complex and costly.

Environment of the customer

The database environment of the customer included the following logistics:

- ▶ Industry: Insurance
- ▶ Language source: Natural and Adabas and COBOL and Adabas
- ▶ Language target: COBOL and DB2
- ▶ Database source: Adabas
- ▶ Database target: DB2 for z/OS
- ▶ Number of Adabas files: 272
- ▶ Number of Natural modules: 17,500
- ▶ Number of COBOL programs: 30,000
- ▶ Project timeframe: Three years

9.7.4 OnTarget conversion process

The customer started the conversion process by identifying a small, stand-alone subsystem that was migrated to the new DB2 environment and placed into production. This first step was a learning process that eased the migration process of more systems.

The OnTarget conversion process included the following tasks:

Assessment

The application inventory of the project was assessed to identify any missing modules or application entities that were orphan entities or were not required as part of the production environment.

The code was parsed to identify any specific code constructs that required special handling during the code conversion process. The code also was segmented into bundles to facilitate a phased delivery of the converted code. Any dependencies between migrated and non-migrated code were identified. JCL and relevant procedures were identified and then slated for modification during the conversion process. The assessment process produced the following documentation:

- ▶ Detailed inventory of the application modules and entities that were targeted for conversion or migration
- ▶ List of the division of responsibilities between MOST and the customer
- ▶ Project plan that outlined the processes and resources necessary to migrate off the SAG platform
- ▶ List of customization actions to OnTarget (the automated conversion tool) to optimize the code conversion process

Architecture

By using the COBOL three-tier architecture implementation, the customer renewed the presentation layer, which provided a browser-based front end. The customer accessed the elements of the business layer to ease the integration process with the new application systems that were built within the organization.

Preparing for migration

All open issues that were identified in the assessment and analysis phase were resolved and the data and application inventory for the migration was confirmed. The project management framework also was established.

Designing the database

An automated default design process was applied to the current Adabas database structure to determine the final DB2 design. This design process analyzed the existing database structures and the usage of the database by the application to determine an efficient DB2 data model. As a result of this process, a fully normalized DB2 data model was designed. After all of the changes to the data model from the customer were finalized, the DDL used to build the DB2 database were generated and given to the customer. The customer used the DDL to set up the new DB2 environment.

Migrating data

After the customer took delivery of the components that migrate data from Adabas to DB2, the customer proceeded with the data migration process in parallel with other phases of the project. The customer also received programs to validate the migration process and to conduct a reverse migration. The reverse migration programs are used to restore the mature database if problems occurred during the cutover phase of the process.

Converting Natural code

All Natural programs were converted to a three-layer application architecture that consisted of the Presentation layer, Business Logic layer, and Data Access layer (OnTarget was the tool that generated the new COBOL programs). These new programs were delivered to the customer with MOST base elements that comprised the MOST frameworks for implementing the converted code. The converted code consisted of COBOL source code that completed a clean compile process and vendor testing.

Modifying COBOL code

The COBOL programs that accessed Adabas by using the SAG preprocessor (ADASQL) was modified through OnTarget. A DAL module was generated for each program that accessed Adabas and the programs were modified to access DB2 through the generated DAL.

Testing the MOST vendor

Vendor testing by MOST was conducted by using customer-supplied testing scenarios and test data. MOST used MF-Test, the automated regression testing tool of MOST that runs the original Natural programs and compares the output of these programs to the output that was generated by the converted COBOL programs. The output generated by the business scenario included online screens, reports, work files, and DB2 data. MF-Test automatically captured and compared the source and target application output.

User acceptance testing

User testing of the new migrated applications was conducted in the operating environment of the customer. A Warranty support period was provided after the migrated code was delivered.

Conducting a code refresh

Any changes that were made to the source application code during the project resulted from ongoing application maintenance needs and were passed on to MOST. MOST used OnTarget to regenerate the updated code modules that were affected by the application changes. From the moment the code was received through the go live date, the application code was frozen.

Production readiness

The final phases of the project were conducted by the customer to bring the system to production, including system integration and testing and performance testing.

9.7.5 Results

The insurance company is in a better position to integrate its enterprise systems with its billing database. Many of the business processes of the company, from enrollment to claims settlement, are faster and more efficient. A major goal within the IT organization is to keep pace with database growth and provide access to real-time data.

With DB2 for z/OS, this customer provides real-time information and quickly develops productivity-enhancing applications that enable the company to manage an increase in business.

The following DB2 for z/OS tools were used in support of the new DB2 environment:

- ▶ IBM DB2 Administration Tool for z/OS
- ▶ IBM DB2 Object Comparison Tool for z/OS
- ▶ IBM DB2 Automation Tool for z/OS
- ▶ IBM DB2 Log Analysis Tool for z/OS
- ▶ IBM DB2 High Performance Unload for z/OS

9.7.6 Lessons learned

The company learned the following lessons from this project:

- ▶ The migration of a small subsystem from the mature environment to the new application platform and the implementation of this application into production was a useful learning process that laid the ground work to roll out subsequent converted systems with relative ease.
- ▶ The Assessment phase that preceded the project contributed to a successful project planning process and identified areas of potential risk.
- ▶ The Assessment phase also provided an opportunity to update the applications before migration. During this phase, components that were not required in the production environment were eliminated.

9.8 eavRPM solution

The sEAV Rapid Program Modernization (eavRPM) solution is provided by ATERAS. eavRPM is a desktop toolset that provides *rapid program modernization* capabilities to Natural and COBOL developers. eavRPM uses Natural and COBOL online applications that access Adabas to create new Java applications.

The existing Adabas databases are converted to DB2 for z/OS. The batch Natural and COBOL programs are converted to COBOL batch jobs that runs against the new DB2 for z/OS databases. The DB2 for z/OS conversion and the Natural-to-COBOL batch conversions are provided as an automated service from ATERAS.

eavRPM uses copy-and-paste techniques to quickly and efficiently modernize the existing user interface windows and underlying business rules of the database. By using this solution, developers realize the following results:

- ▶ The newly generated applications are object-oriented, well-documented, and are maintained in any industry-standard IDE (no licensing fees are required).
- ▶ The windows of the database interface are enhanced to use drop-down menus, check boxes, and graphic buttons.
- ▶ Function keys, tabs, and enter key sequences that are familiar to users are retained.
- ▶ Application business code is based on Java.
- ▶ Web pages are generated by using HTML, XML, and JSP.

This conversion solution is implemented, in part, by using DB-Shuttle. For more information about DB-Shuttle, see 8.3.1, “DB-Shuttle solution details” on page 90.

The basic overview of the eavRPM solution is shown in Figure 9-28.

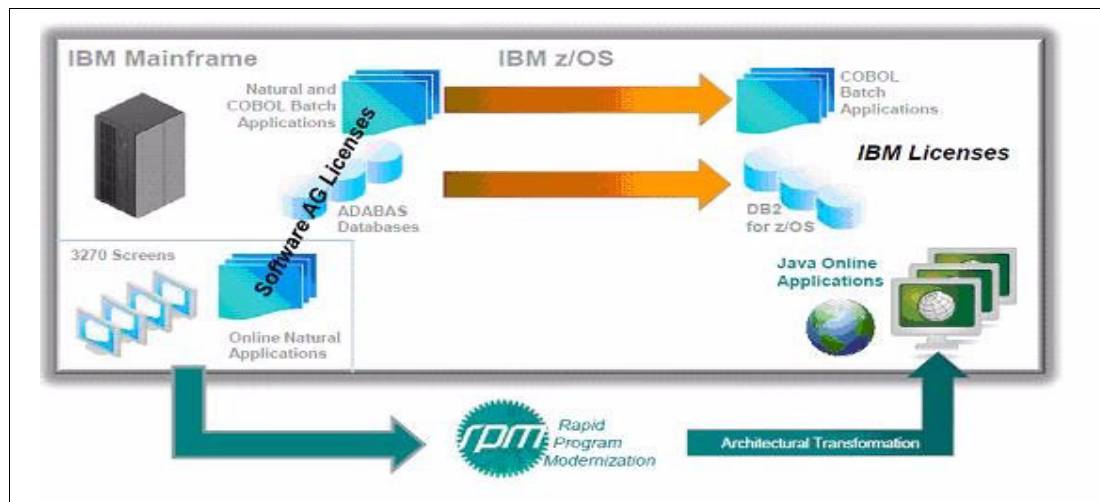


Figure 9-28 eavRPM solution overview

The eavRPM desktop workbench tool is used to understand source code. By using this tool, the following benefits are realized:

- ▶ Instant source documentation, flow diagrams, code thread analysis, and on-demand rules identifications are created.
- ▶ COBOL and Natural online programs are transformed to web-based applications (Java).
- ▶ Intuitive workbench for Natural or COBOL is made available to developers.
- ▶ Developers copy and paste mature maps and business rules.
- ▶ Native Java web pages are quickly generated from mature code.
- ▶ Optional functions for web or SOA initiatives.
- ▶ Workflow and program flow remain unchanged.
- ▶ Natural commands are retained as comments or the commands are deleted.

Part of the ATERAS conversion process includes generating a relational database to replace the functionality and content of the Adabas databases. The target database is DB2 for z/OS.

Database functionality

The ATERAS solution completely replaces the following Adabas file functionality:

- ▶ Adabas file layouts
- ▶ Group-level elements

- ▶ MU occurrences
- ▶ PE occurrences
- ▶ Super-Descriptors
- ▶ Sub-Descriptors

Relational results

The resulting relational database is fully relational. Primary keys, foreign keys, and index definitions are created automatically. All constraints are generated into the resulting DDL. Table spaces, indexes, table names, and column names are generated according to the naming standards of the customer.

Delivered components

As part of the delivery process, ATERAS generates and delivers the following component types that are installed in the new relational database processing environment by using DB-Shuttle:

- ▶ Data Definition Language syntax for the new database.
- ▶ Adabas Data Extract programs (generated in COBOL) to unload all Adabas data to the correct format for the relational database load utility.
- ▶ Adabas Data Extract JCL (customized to your environment) to run the extracts and other key-processing utilities.
- ▶ Load syntax for use by the relational database load utility (optional).
- ▶ RI Check syntax for use by the relational database utility package (optional).
- ▶ RUNSTATS syntax for use by the relational database utility package (optional).
- ▶ DCLGEN syntax to define COBOL layouts for the replacement applications (optional).

Adabas data conversion

During the extract process, all Adabas data is extracted and written to a set of sequential files that are loaded into the new target relational database. The Adabas data extracts are run simultaneously.

The solution provides a number of extract variations for sites that have special requirements for a short Adabas conversion window. The data conversion that is conducted during the cutover weekend or evening is fast and complete.

Customization workbenches

Special workbenches within DB-Shuttle provide the following capabilities for tailoring an Adabas conversion to meet the requirements of the customer:

- ▶ Rename workbenches to allow full naming of all tables, columns, table spaces, and indexes by using a rules basis or a full-name basis
- ▶ Data Cleansing Workbench to provide rule-based data cleansing during the Adabas data extract process
- ▶ Element Rename Workbench that allows the selection of group-level or elementary Adabas fields for use as columns in the relational database
- ▶ Record Redefinition Workbench to allow changes to field types and lengths during the relational conversion

Visibility and knowledge-building

Customer teams and the ATERAS teams need a full understanding of the existing Adabas files and the post-conversion relational database. DB-Shuttle generates the following reports and diagrams to assist with this knowledge-building process:

- ▶ Complete entity relationship model (ERM) diagrams of the new relational databases
- ▶ Summary reports of the Adabas file sizes and usage
- ▶ Matrices that relate files to tables, fields to columns, and Adabas files to extract programs
- ▶ MU and PE *maximum occurrences* reports that are based on automated mining of the Natural code base
- ▶ Ad hoc reports that identify the unique characteristics and conversion of the Adabas database

Modernization results

Modernization of the application by using eavRPM is provided as a service by ATERAS or by the customer team that uses eavRPM. In the modernization process, native web applications are used without the need for middleware, black boxes, or ongoing license fees.

The modernization process also results in the following improvements:

- ▶ Business rules are unchanged
- ▶ New user interface
- ▶ Continued keyboard features with new mouse-driven and scrollable features
- ▶ Lower operating costs for your organization
- ▶ Natural-based maps are converted to web pages that are based on Java, as shown in Figure 9-29 on page 151.

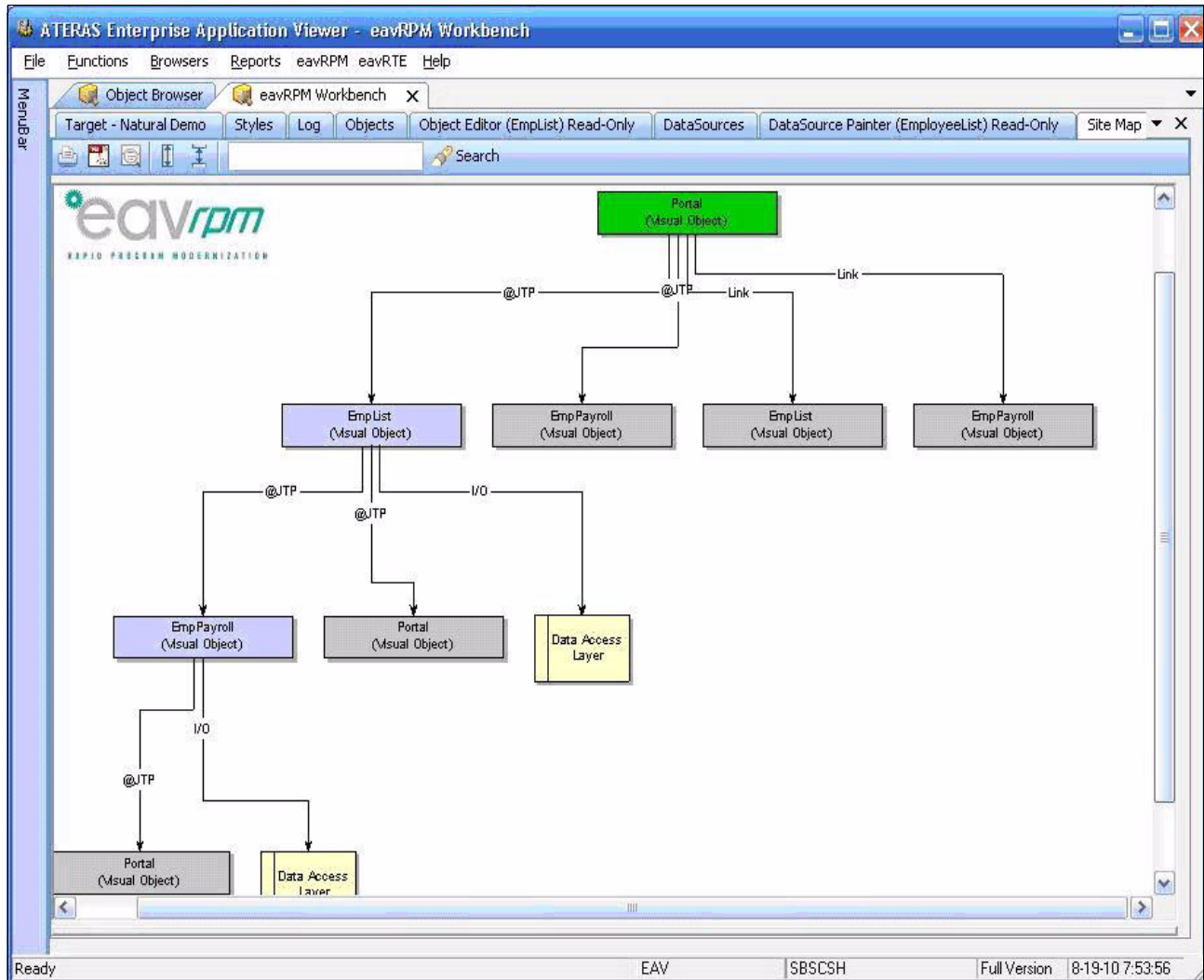


Figure 9-30 Maps flow

To convert to a web application, the Natural or COBOL developer must complete the following tasks:

1. Begins with the existing application code base.
2. Selects map images and business rules to include in the new web application.
3. Adds hyperlinks, buttons, graphics, and other web features.
4. Generates a well-structured native Java solution.
5. Runs the new web application against the new DB2 for z/OS database.

The new web application is implemented directly into Eclipse or the IDE for all future maintenance. Only a few objects are included in the resulting Java ATERAS framework.

The DB2 for z/OS database is accessed through a standard ODBC connection.



Part 4

Post-conversion tasks

The automated database conversion solutions that are described in this book provide a like-to-like conversion from a pre-relational database to DB2 for z/OS. In many instances, this conversion is the end solution for a customer.

However, there are instances in which customers want to initiate a subsequent phase of the conversion process to take advantage of the following opportunities:

- ▶ Database tuning
- ▶ Usage of relational functionality
- ▶ Data model redesign
- ▶ Application re-architecture
- ▶ Usage of database tools

In this part, we describe the post-conversion tasks that are most often conducted by customers. This part includes the following chapters:

- ▶ Chapter 10, “z/OS and DB2 setup” on page 155
- ▶ Chapter 11, “Application and SQL tuning” on page 175
- ▶ Chapter 12, “Maximizing DB2 for z/OS features” on page 199



z/OS and DB2 setup

One phase of the conversion solutions that are described in this book is extracting the physical database definitions from the pre-relational database and building the data definition language (DDL) for the DB2 objects. This process builds the DDL necessary to define the storage groups, databases, buffer pools, table spaces, tables, indexes, and any other relational constructs that are required by the application. The initial definition of object naming conventions and object sizes are provided as input to the database conversion process. The DDL is generated by using this input.

Some changes might be made to the generated DDL during the testing phases of the conversion project. DB2 and z/OS system configuration changes also might be needed.

In this chapter, we describe the changes to the DB2 database and the z/OS configuration that must be reviewed after the conversion phase and before the production implementation phase.

This chapter includes the following topics:

- ▶ DB2 system parameters
- ▶ Configuring the buffer pool
- ▶ CICS configuration
- ▶ z/OS operating system

10.1 DB2 system parameters

If you are converting from a pre-relational database to DB2 for z/OS and you did not have access to a DB2 environment before the conversion, you must complete all of the installation and configuration tasks that are associated with setting up a new DB2 environment. If you have an existing DB2 environment and are converting a pre-relational database to run in that environment, you must review your DB2 system parameters (DSNZPARMs) to determine whether the existing values are sufficient to support the existing workload and the new workload.

In this chapter, not all of the DSNZPARMs required to configure a new DB2 environment is described because the DB2 documentation provides this information. Instead, we focus on the DSNZPARMs that must be reviewed when a new workload is added to an existing environment. The reviewed DSNZPARMs are categorized into the following groups:

- ▶ Thread management parameters
- ▶ Locking parameters
- ▶ Checkpoint control parameters
- ▶ Distributed Data Facility parameters
- ▶ DB2 log parameters
- ▶ Resource management parameters
- ▶ EDM pool parameters

10.1.1 Thread management parameters

A required critical task in planning for your conversion to DB2 is to estimate the amount of workload that is migrated from the pre-relational environment to DB2. You must review reports on the thread activity of the pre-relational environment and the existing DB2 environment, if the reports are available.

If you are using an automation tool that facilitates a like-to-like conversion, often there is an increase in the amount of thread activity in the DB2 environment that is equivalent to the amount of thread activity in the pre-relational environment. If the conversion is a complete application replacement, rewrite, or re-engineering conversion, you must determine the way in which the pre-relational threads translate to DB2 threads.

The following DSNZPARMs must be evaluated to determine the expected number of DB2 threads:

▶ CTHREAD

The CTHREAD parameter is shown as the MAX USERS field in the DB2 installation panels. This parameter controls the maximum number of concurrent allied or local threads for a DB2 subsystem. The value for this parameter must be the sum of all of the following expected activity:

- Each TSO user (whether running a DSN command or a DB2 request from DB2 QMF)
- Each batch job (whether running a DSN command or a DB2 utility)
- Each IMS region that accesses DB2
- Each active CICS transaction that accesses DB2
- Each utility (each utility uses one thread, and another thread is used for each subtask)
- Each connection from users of CAF and RRSAP

The maximum value of CTHREAD is 2,000 in DB2 9 for z/OS and 20,000 in DB2 10 for z/OS.

- **MAXDBAT**

The MAXDBAT DSNZPARM is shown as the MAX REMOTE ACTIVE field in the DB2 installation panels. This DSNZPARM controls the maximum number of allowable concurrently active database access threads (DBATs). A DBAT thread accesses data at the local subsystem on behalf of a remote subsystem; for example, a thread that is associated with a query submitted from QMF for Workstation. Other examples of a DBAT include work that originates from a web interface or through a JDBC or ODBC connection.

The maximum value of MAXDBAT is 1,999 in DB2 9 for z/OS and 19,999 in DB2 10 for z/OS.

10.1.2 Locking parameters

Adding workload to an existing DB2 environment introduces locking concerns. System parameters that are established for an existing DB2 workload might not be adequate for the new applications that are running on DB2.

The following DSNZPARMs must be evaluated to determine whether the appropriate levels of concurrency are established:

- **IRLMRWT**

The IRLMRWT DSNZPARM is shown as the RESOURCE TIMEOUT field in the DB2 installation panels. This DSNZPARM controls the number of seconds that must elapse before a resource timeout is detected (the default value is 30 seconds). Depending on the expected number of concurrent users, the typical commit scope of each user, and the amount of update activity that is conducted, you might need to evaluate whether your existing IRLMRWT value is appropriate.

Often you are unaware if the IRLMRWT value is appropriate based on a unit test or system test environment configuration. When you run high-volume transactions, you often encounter locking issues such as timeouts and deadlocks. Increasing IRLMRWT is not the best solution for every instance of locking issue because IRLMRWT retains locks longer, which causes more threads to wait longer for the locks to release. In most instances, it is best to incur timeouts and deadlocks and then tune the applications to hold fewer locks or commit more frequently.

- **NUMLKTS**

The NUMLKTS DSNZPARM is shown as the LOCKS PER TABLE(SPACE) field in the DB2 installation panels. This DSNZPARM controls the maximum number of page, row, or LOB locks that an application holds simultaneously in a table or table space. Lock escalation occurs when a single application exceeds the maximum number of locks in a single table or table space.

Ideally, lock escalation is minimized. If an application acquires multiple page locks (which is the most common level of locks), and lock escalation occurs because the number of page locks on a single table exceeds the NUMLKTS value, DB2 escalates to the partition or table space level (the next highest level of lock). This escalation severely affects concurrency with other users who access the same table or partition. You must understand the locking behavior of your applications and adjust NUMLKTS appropriately to ensure concurrency.

- **NUMLKUS**

The NUMLKUS DSNZPARM is shown as the LOCKS PER USER field in the DB2 installation panels. This DSNZPARM represents the maximum number of locks that a single application holds concurrently for all table spaces. You must understand your application so that you tune NUMLKUS in the same manner that you tuned NUMLKTS.

10.1.3 Checkpoint control parameters

A checkpoint is a point at which DB2 records status information on the DB2 log. The DB2 recovery process uses this information if DB2 abnormally terminates. Applications must commit frequently to avoid long recovery processes. The following system parameters are used to control checkpoint frequency and to control monitoring applications that do not follow guidelines for committing frequently:

► **CHKTYPE**

The CHKTYPE DSNZPARM is shown as the CHECKPOINT TYPE field in the DB2 installation panels. This DSNZPARM indicates whether the interval between log checkpoints is based on the number of written log records (value LOGRECS), the time between checkpoints (value MINUTES), or both (value BOTH). This parameter is used with the CHKFREQ DSNZPARM to determine the frequency that DB2 checkpoints are taken.

► **CHKFREQ**

The CHKFREQ DSNZPARM is shown as two fields in the DB2 installation panels: RECORDS/CHECKPOINT and MINUTES/CHECKPOINT. The values that are used depend on whether the value of CHKTYPE is set to LOGRECS, MINUTES, or BOTH.

If you specify a checkpoint frequency rate that is based only on the number of log records that were written, you might experience a varying frequency rate. For example, heavy activity in your online transaction processing (OLTP) workload results in frequent checkpoints if those checkpoints are based on the written log records. If most of your batch workload is reporting, there is less update activity so the checkpoints occur further apart. Because of this configuration, you must manage your checkpoints according to minutes rather than log records.

You want to ensure that a checkpoint is taken no more than every 3 - 5 minutes. If your checkpoints occur less frequently, your recovery times are affected. You specify your checkpoints according to log records and minutes by specifying BOTH for DSNZPARM CHKTYPE. When you specify BOTH, DB2 takes a checkpoint when the specified number of log records is written. DB2 also takes a checkpoint after the specified number of minutes if the threshold for the number of written log records is not exceeded.

If you add your pre-relational workload to an existing DB2 workload, the update activity increases such that checkpoints occur more frequently than needed. An OMEGAMON PE Statistics Report shows how frequently DB2 is taking a checkpoint. This frequency rate also is shown in a performance test statistics report taken during the conversion process. Example 10-1 shows that 46 checkpoints were taken over a period of three hours and 45 minutes, which is a rate of one checkpoint taken every 4.89 minutes. This rate is within the ideal range.

Example 10-1 DB2 checkpoint data in an OMEGAMON PE Statistics Report

INTERVAL FROM: 12-03-13 14:14:48.69
 TO: 12-03-13 17:59:48.69

SUBSYSTEM SERVICES	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----
SYSTEM EVENT CHECKPOINT	46.00	0.00	0.00	0.00

► **URCHKTH**

The URCHKTH DSNZPARM is shown as the UR CHECK FREQ field in the DB2 installation panels. This DSNZPARM specifies the number of checkpoint cycles that are complete before DB2 issues a warning message to the console for an uncommitted unit of

recovery (UR). If applications do not commit frequently enough, the applications hold locks for a longer time and might incur longer backout times if the applications stop.

This DSNZPARM provides the capability to report on long-running update jobs that do not commit. If you are unsure of the commit frequency of your converted jobs because you do not control the commit frequency, this DSNZPARM must be set to a non-zero value. If a job exceeds this DSNZPARM value, the job does not abend. A console message is issued instead, so you are able to track long-running update jobs.

The value of URCHKTH must be set high enough so that console messages are not produced for the majority of your jobs. However, the value must be set low enough so that jobs that are reviewed for commit frequency are identifiable.

- URLGWTH

The URLGWTH DSNZPARM is shown as the UR LOG WRITE CHECK field in the DB2 installation panels. This DSNZPARM specifies the number of log records that are written before DB2 issues a warning message. The setting is similar to the URCHKTH DSNZPARM, but URLGWTH DSNZPARM is affected only if the CHKTYPE DSNZPARM is set to LOGRECS or BOTH.

10.1.4 Distributed Data Facility parameters

The Distributed Data Facility (DDF) is used to access a local DB2 subsystem remotely. DB2 10 for z/OS also allows increased granularity for monitoring and controlling connections and threads at the server with the use of profile tables. These profile tables allow you to override certain system parameters and control connections by using the following categories:

- IP Address (IPADDR)
- Product Identifier (PRDID)
- Role and Authorization Identifier (ROLE, AUTHID)
- Collection ID and Package Name (COLLID, PKGNAME)

For more information about profile tables, see *DB2 10 for z/OS Technical Overview*, SG24-7892.

The following DDF DSNZPARMs must be reviewed when workload is added to an existing DB2 subsystem, and when any of the new workload accesses DB2 from a remote location:

- IDTHTOIN

The IDTHTOIN DSNZPARM is shown as the IDLE THREAD TIMEOUT field in the DB2 installation panels. This DSNZPARM is a time-out value that specifies the number of seconds that an active server thread remains idle. The thread is canceled after the time-out value expires, and any locks and cursors that are held by the thread are released.

If you introduce distributed threads into your workload, you must adjust this value so that idle threads do not remain active for a long time. Ideally, this value is set to a range within 5 - 10 minutes.

- POOLINAC

The POOLINAC DSNZPARM is shown as the POOL THREAD TIMEOUT field in the DB2 installation panels. This DSNZPARM specifies the number of seconds (the default value is 120) that a database access thread (DBAT) remains idle in the pool before the DBAT is terminated. A DBAT in the pool is considered an active thread against MAX REMOTE ACTIVE (DSNZPARM MAXDBAT) and holds locks, but does not feature any cursors. This value cleans up unused inactive DBATs.

► TCPALVER

The TCPALVER DSNZPARM is shown as the TCP/IP ALREADY VERIFIED field in the DB2 installation panels. This DSNZPARM specifies whether DB2 accepts TCP/IP connection requests that contain only a user ID (no password, RACF PassTicket, or Kerberos ticket) or if a higher form of security is required. If your pre-relational applications include built-in security to ensure that work that is coming in over a TCP/IP connection is valid, TCPALVER is set to YES.

10.1.5 DB2 log parameters

The DB2 log contains before and after images of data that was modified by applications through INSERT, UPDATE, and DELETE statements. The log also contains images for data that was loaded by utilities that run with the LOG YES option. The amount of update activity that is introduced by the newly converted applications increases the activity on the DB2 log. The following log-related DSNZPARMs must be reviewed to ensure that they are adequately sized for the new workload:

► NUMBER OF LOGS

The NUMBER OF LOGS field specifies the number of data sets that are established for each copy of the active log during installation. The value that you specify for this field depends on how heavy your insert, update, and delete activity is expected to be, and how far back you want to recover active logs. You specify this value during the DB2 installation or migration process in the Interactive System Productivity Facility (ISPF) panels when you are customizing your DB2 environment. You adjust this value if your update activity or recovery needs change. The size of each log data set is 4 GB.

► OUTBUFF

The OUTBUFF DSNZPARM is shown as the OUTPUT BUFFER field in the DB2 installation panels. This DSNZPARM specifies the size of the output buffer that is used for writing active log data sets. You must have enough log output buffers to manage writing to and reading from the log without requiring an I/O to the log data sets for each log access.

The DB2 log activity is shown in the Log Activity section of the OMEGAMON PE Statistics Report, as shown in Example 10-2.

Example 10-2 Log Activity section of OMEGAMON PE Statistics Report

LOG ACTIVITY	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----
READS SATISFIED-OUTPUT BUFF	479.00	0.05	0.19	0.00
READS SATISFIED-OUTP.BUF(%)	100.00			
READS SATISFIED-ACTIVE LOG	0.00	0.00	0.00	0.00
READS SATISFIED-ACTV.LOG(%)	0.00			
READS SATISFIED-ARCHIVE LOG	0.00	0.00	0.00	0.00
READS SATISFIED-ARCH.LOG(%)	0.00			

All log reads in this example are satisfied by the output buffer, which is the wanted outcome. If a value of less than 100% is shown in this field, you must adjust the value of the OUTBUFF DSNZPARM.

10.1.6 Resource management parameters

The DB2 environment includes a number of resources that are managed by system parameters. Resources that are reviewed when new workloads are introduced are the number of data sets, the sort pool, the RID pool, and the EDM pool. We describe the first three resources in this section and the EDM pool parameters in the next section. The following DSNZPARM values must be reviewed to ensure sufficient size for the new workload:

- DSMAX

The DSMAX DSNZPARM is shown as the MAXIMUM OPEN DATA SETS field in the DB2 installation panels. This DSNZPARM specifies the maximum number of data sets that are allowed open at the same time.

When you introduce new workload to an existing DB2 subsystem, you must estimate the number of new DB2 data sets that are potentially opened at the same time. Each table is a separate data set. If a table is partitioned, each partition of the table is a separate data set. Each index is a separate data set. You must take these factors into consideration when a value for DSMAX is set.

The DSMAX value must be set low because a high value reduces the amount of available virtual storage. However, more resources are above the 2 GB bar than in previous versions of DB2. For more information about setting a DSMAX value, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

- SRTPOOL

The SRTPOOL DSNZPARM is shown as the SORT POOL SIZE field in the DB2 installation panels. This DSNZPARM specifies the amount of storage that is needed for the sort pool.

DB2 sorting occurs in memory by using work files. The SRTPOOL parameter controls the amount of memory that is allocated for sort activity. The work files are objects that are created in the work file database.

For more information about a formula to estimate the size of the sort pool, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

- MAXRBLK

The MAXRBLK DSNZPARM is shown as the RID POOL SIZE field in the DB2 installation panels. This DSNZPARM specifies the amount of storage (in KB) that is needed for the RID pool.

The RID pool is an area of local storage that is reserved for record identifier (RID) sort processing, including RID list sort processing.

For more information about a formula to estimate the size of the RID pool, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

10.1.7 EDM pool parameters

The environmental description manager (EDM) consists of the following pools: the prepared-statement cache pool, the skeleton pool for packages and plans, and the database descriptor (DBD) pool. This storage is allocated above the 2 GB bar.

The following DSNZPARM values must be reviewed to determine their sufficient size for the new workload:

- EDMSTMTC

The EDMSTMTC DSNZPARM is shown as the EDM STATEMENT CACHE field in the DB2 installation panels. This DSNZPARM specifies the size (in KB) of the statement cache

that is used by the EDM. The prepared-statement cache pool contains prepared dynamic SQL statements that are cached for reuse.

The number of prepared statements that are stored in the cache depends on the characteristics of the dynamic SQL statements that your application runs. For more information about guidelines for setting a value for this parameter, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

► EDMDBDC

The EDMDBDC DSNZPARM is shown as the EDM DBD CACHE field in the DB2 installation panels. This DSNZPARM specifies the minimum size (in KB) of the DBD cache that is used by the EDM.

The EDM DBD cache is used for caching DBDs for database objects, such as table spaces, tables, indexes, index spaces, relationships, check constraints, and triggers. The cache reduces the need to access object descriptions from the DB2 catalog.

For more information about the formulas that are used to estimate the size of the EDM DBD cache, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

► EDM_SKELETON_POOL

The EDM_SKELETON_POOL DSNZPARM is shown as the EDM SKELETON POOL SIZE field in the DB2 installation panels. This DSNZPARM specifies the minimum size (in KB) of the EDM skeleton pool.

The EDM skeleton pool is used for storing skeleton package tables (SKPTs), in which each SKPT represents a package.

For more information about the formulas that are used to estimate the size of the EDM skeleton pool, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974.

10.2 Configuring the buffer pool

The conversion solutions that are described in this book produce a set of DDL statements that include the buffer pool values for table spaces and indexes. These values are set at the time that conversion rules are defined in the conversion tool. The relationships between DB2 objects might be unknown at the time the conversion rules are defined, so adjustments to the buffer pool assignments are required before the production implementation and during subsequent post-production performance tuning.

Post-conversion buffer pool configuration often falls into the following categories:

- Buffer pool assignment
- Buffer pool sizing
- Buffer pool thresholds
- Buffer pool tuning tools
- Automating buffer pool management
- Additional buffer pool considerations

10.2.1 Buffer pool assignment

DB2 provides a number of system parameters that control the default buffer pools that are used for table spaces, indexes, large object (LOB) data, and XML data. The system parameters that control these defaults are shown in Table 10-1 on page 163.

Table 10-1 Default buffer pool system parameters

System parameter	Description
TBSBPOOL	Default 4-KB buffer pool for user data
TBSBP8K	Default 8-KB buffer pool for user data
TBSBP16K	Default 16-KB buffer pool for user data
TBSBP32K	Default 32-KB buffer pool for user data
TBSBPLOB	Default buffer pool for user LOB data
TBSBPXML	Default buffer pool for user XML data
IDXBPOOL	Default buffer pool for user indexes

If the default buffer pools are used as input rules to the conversion tool, the DB2 objects are placed in a few buffer pools, which often is not an ideal configuration. There are best practices for assigning DB2 objects to buffer pools which also hold true when objects are converted from other databases.

Applications that are converted by a tool often do not take advantage of the relational capabilities of DB2 when the applications access data. Therefore, there are other considerations that factor into the conversion planning process. For example, the requirements for an application that tracks client data and reports on names and addresses for each client might state that there are one or more names and addresses that are associated with each client.

If a batch job reads all of the clients and produces a report with names and addresses of the clients, DB2 performs a table space scan of the client table and uses dynamic prefetch. If the client table is assigned to the same buffer pool as other tables that are accessed randomly, the data for the other tables might be paged out quickly because of the high volume of prefetched data from the client table that is stored in the same buffer pool. Prefetch threshold values are used to prevent buffer abuse from occurring, as described in 10.2.3, “Buffer pool thresholds”. However, consideration must be given to separating tables that are used primarily for sequential access from tables that are used for random access.

A sample buffer pool configuration is shown in Table 10-2. The high buffer pool numbers that are shown in the table are used so that the buffer pools do not interfere with buffer pool BP0, which is used for the DB2 catalog and directory, and with BP7, which customers often use for sort work activity.

Table 10-2 Sample buffer pool assignment

Buffer pool	Description
BP10	General TS buffer pool - Table spaces with no special requirements
BP11	General IX buffer pool - Index spaces with no special requirements
BP12	Sequential TS buffer pool - Table spaces with high prefetch activity
BP13	Sequential IX buffer pool - Index spaces for high prefetch table spaces
BP14	Common TS buffer pool - Code tables / commonly accessed R/O tables
BP15	Common IX buffer pool - Indexes for tables in common TS buffer pool
BP16	Critical TS buffer pool - Table spaces with special requirements
BP17	Critical IX buffer pool - Index spaces with special requirements

The values shown in Table 10-2 on page 163 are suggestions for assigning objects to buffer pools. There are many alternatives and the configuration that works best for each environment is different. Choosing a buffer pool configuration that allows for efficient access of index and table space data and minimizes the amount of synchronous I/O is important.

The chosen buffer pool configuration is affected by the level of normalization of the converted database. If the pre-relational database included highly intermingled data and the converted database is highly normalized, you must consider the best way to access data in multiple tables at the same time.

For more information about buffer pool assignments and suggestions on techniques for buffer pool monitoring and tuning, see *DB2 9 for z/OS: Buffer Pool Monitoring and Tuning*, REDP4604.

10.2.2 Buffer pool sizing

Creating a large enough buffer pool is the same concern for DB2 databases that are converted from other databases as it is for existing databases. DB2 works efficiently when enough memory is provided to minimize the amount of I/O. DB2 buffer pools reside above the 2 GB bar since DB2 for z/OS Version 8; therefore, the size of the buffer pools is no longer limited by the amount of virtual storage that is available to DB2. The limitation exists in the amount of available real storage. Allowing for enough real storage to back up your workload is important because this allowance avoids costly paging operations.

The sizing of each buffer pool often is directly related to the assignment of objects to buffer pools. The buffer pools that are designated for general-purpose table spaces and indexes must be large enough to manage data that is stored for multiple objects. The buffer pools that are designated for critical table spaces and indexes must be sized appropriately to minimize synchronous I/O for those critical objects.

Although code tables and common tables that are accessed by multiple applications often are smaller, they are critical because of how frequently they access the database. These tables and their associated indexes must be assigned to buffer pools that are large enough to keep all or most of the data pinned in memory. However, this issue often is minor because these tables are smaller than the objects that contain transactional and historical data.

The process to determine the correct size of a buffer pool often involves the need to run multiple tests with several alternative buffer pool configurations. Conducting these tests at a high enough level of I/O activity so that the buffer pools are sufficiently sized is crucial. For more information about that tools that are used to assist with buffer pool assignment and sizing, see 10.2.4, “Buffer pool tuning tools” on page 165.

10.2.3 Buffer pool thresholds

There are a number of thresholds that are defined at the buffer pool level to control the behavior of each buffer pool. Configuring the following thresholds to fit your workload provides considerable performance savings (especially for I/O-intensive workloads):

- Virtual Pool Sequential Steal Threshold (VPSEQT)

This threshold often requires the most adjustment that is based on workload. VPSEQT determines the percentage of pages for a buffer pool that are occupied by pages that are accessed sequentially. The default value for this threshold is 80%. The value that is specified varies because of the behavior of the applications that are accessing each buffer pool. During regular business hours, greater amounts of random access are more likely. During overnight hours, a greater amount of sequential access through batch jobs is likely.

If you choose a buffer pool assignment schema that separates objects that are often accessed sequentially from those objects that are accessed randomly, you must choose a higher VPSEQT value for the buffer pool for objects that are accessed sequentially.

- ▶ **Deferred Write Threshold (DWQT)**

This threshold is a percentage of the buffer pool that might be occupied by unavailable pages, including updated pages and in-use pages. When this threshold is reached, DB2 schedules write operations for enough data sets to decrease the number of unavailable buffers to 10% below the threshold. Specifying too low a value causes write operations to occur too frequently, which results in unnecessary overhead. Setting this threshold too high causes write operations to bunch up, which results in sporadic delays when these write operations occur.

- ▶ **Vertical Deferred Write Threshold (VDWQT)**

Although this threshold is similar to DWQT, it applies to the number of updated pages for a single page set (table, partition, or index) in the buffer pool.

- ▶ **Data Management Threshold (DMTH)**

This threshold represents a percentage of the buffers that are not available for other uses. The value of this threshold is fixed at 95% and cannot be changed. If this threshold is exceeded, DB2 accesses the page in the buffer pool once for each row that is retrieved or updated in that page, rather than accessing the pool once for the entire page. This level of access results in a large CPU increase and must be avoided. If a non-zero value is observed in the number of times this threshold is exceeded (as reported in an OMEGAMON PE Statistics Report), one or more of the following actions must be taken:

- Increase buffer pool size
- Reduce deferred write thresholds (VDWQT, DWQT)
- Increase system checkpoint frequency (reduce DSNZPARM CHKFREQ)

For more information about tuning buffer pool thresholds, see *DB2 10 for z/OS Managing Performance*, SC19-2978.

10.2.4 Buffer pool tuning tools

There are many tools available with which to tune DB2 buffer pools. Two tools available from IBM are DB2 Buffer Pool Analyzer for z/OS and Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS.

DB2 Buffer Pool Analyzer for z/OS

DB2 Buffer Pool Analyzer for z/OS helps database administrators manage buffer pools more efficiently by providing information about current buffer pool behavior and by using simulations to anticipate future buffer pool behavior. DB2 Buffer Pool Analyzer for z/OS assists with placing objects, sizing, and determining the correct thresholds. The tool suite is available as a stand-alone product or is integrated into the Buffer Pool Analysis functions of Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS.

Buffer Pool Analyzer provides a suite of tools that support in-depth analysis of the performance of DB2 buffer pools. The suite of tools includes the following functions:

- ▶ Collection of buffer pool related performance data
- ▶ Host-based creation of reports about buffer pool performance and group buffer pool performance
- ▶ Conversion of performance data to formats suitable for client-based functions and for loading into DB2 tables

- ▶ Client-based graphical representation of buffer pool performance data
- ▶ Client-based optimization of major buffer pool attributes, such as the optimum assignment of objects in buffer pools and optimum buffer pool sizes that are based on actual performance data
- ▶ Client-based simulation of the effects of different buffer pool attributes (based on actual performance data)
- ▶ Client-based, long-term analysis of historical and current performance data

These functions help performance analysts and database administrators to effectively monitor, analyze, and optimize DB2 buffer pools on different levels.

Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS is a single, comprehensive tool that is used to assess the efficiency of, and optimize performance from, your DB2 for z/OS environment. The software combines the sophisticated reporting, monitoring, and buffer pool analysis features of the Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS and IBM DB2 Buffer Pool Analyzer products. The software also adds expert database analysis functions to help you maximize database performance and enhance productivity.

You use the reports from the Performance Monitor component to produce DB2 statistics and accounting reports that monitor the performance of your buffer pools. Example 10-3 shows an excerpt of an OMEGAMON XE for DB2 Performance Expert Statistics Report that provides information about the read operations for a specific buffer pool.

Example 10-3 Buffer Pool Read Operations Section of an OMEGAMON PE Statistics Report

BP22	READ OPERATIONS	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----	-----
BP00L	HIT RATIO (%)	76.33			
GETPAGE	REQUEST	5071.3K	1878.25	5.16	3.42
GETPAGE	REQUEST-SEQUENTIAL	21.00	0.01	0.00	0.00
GETPAGE	REQUEST-RANDOM	5071.3K	1878.24	5.16	3.42
SYNCHRONOUS	READS	662.2K	245.27	0.67	0.45
SYNCHRON.	READS-SEQUENTIAL	4.00	0.00	0.00	0.00
SYNCHRON.	READS-RANDOM	662.2K	245.27	0.67	0.45
GETPAGE	PER SYN.READ-RANDOM	7.66			
SEQUENTIAL	PREFETCH REQUEST	8.00	0.00	0.00	0.00
SEQUENTIAL	PREFETCH READS	8.00	0.00	0.00	0.00
PAGES READ	VIA SEQ.PREFETCH	486.00	0.18	0.00	0.00
S.PRF.PAGES	READ/S.PRF.READ	60.75			
LIST	PREFETCH REQUESTS	1976.00	0.73	0.00	0.00
LIST	PREFETCH READS	106.00	0.04	0.00	0.00
PAGES READ	VIA LIST PREFETCH	3291.00	1.22	0.00	0.00
L.PRF.PAGES	READ/L.PRF.READ	31.05			
DYNAMIC	PREFETCH REQUESTED	77344.00	28.65	0.08	0.05
DYNAMIC	PREFETCH READS	19671.00	7.29	0.02	0.01
PAGES READ	VIA DYN.PREFETCH	534.3K	197.89	0.54	0.36
D.PRF.PAGES	READ/D.PRF.READ	27.16			

PREF.DISABLED-NO BUFFER	0.00	0.00	0.00	0.00
PREF.DISABLED-NO READ ENG	0.00	0.00	0.00	0.00
PAGE-INS REQUIRED FOR READ	0.00	0.00	0.00	0.00

A statistics report contains the following pertinent buffer pool counters for review:

► **BPOOL HIT RATIO**

This ratio represents the percentage of time that a requested row already was in the buffer pool. The number varies widely (generally, a higher number is desirable) and is affected by the nature of the workload, especially whether random or sequential access was used. If tests show that hit ratios of less than 50% for all of the buffer pools are common, you must review the sizes of the buffer pools to ensure that the sizes are adequate.

► **SYNCHRONOUS READS**

This number represents the number of synchronous read I/O operations that are conducted by DB2 for applications and utilities. A high number indicates that a large amount of I/O operations is occurring.

► **Prefetch counters**

Statistics concerning the use of the three types of prefetch operations that are conducted by DB2 (sequential prefetch, list prefetch, and dynamic prefetch), are provided in the statistics report. Starting with DB2 9 for z/OS, DB2 uses dynamic prefetch more often than sequential prefetch, so a large amount of dynamic prefetch often is observed when data is accessed sequentially.

► **Prefetch disabled counters**

There are two counters that show the frequency with which prefetch was disabled: PREF.DISABLED - NO BUFFER, and PREF.DISABLED - NO READ ENG. DB2 read operations continue when prefetch is disabled (the read operations are not as efficient when multiple pages must be read):

- PREF.DISABLED - NO BUFFER represents the number of times that prefetch was disabled because no buffer was available. This event occurs when 90% of the buffers cannot be stolen or when there are fewer sequential buffers available that are based on the value that is set for the VPSEQT. If this value is non-zero, one of the following measures must be taken:
 - Increase the size of the buffer pool.
 - Reduce the value of one of the deferred write thresholds (DWQT or VDWQT).
 - Increase the frequency of system checkpoints by reducing DSNZPARM CHKREQ.
 - Increase the value of VPSEQT (this value affects random access).
- PREF.DISABLED - NO READ ENG represents the number of times that prefetch was disabled because there was a read engine was unavailable. Because there are 300 read engines, a maximum of 300 concurrent sequential prefetch operations are processed at one time. When this threshold is reached, sequential prefetching is disabled and the count is incremented. The value for this field must be close to zero. If the value is not close to zero, you must consider rescheduling the workload so that you do not have hundreds of concurrent processes that conducts prefetch operations.

► **PAGE-INS REQUIRED FOR READ**

This value represents the number of times that a paging operation was required to access a data page or index page because the entire buffer pool is not backed by real storage. This value must be zero if you have sufficient real storage to back your buffer pools. If sufficient real storage is unavailable, you must size your buffer pools such that the counter is not more than 5% of the number of getpage requests.

These counters are a few of the counters available in IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS reports. Other sections of the report provide information about buffer pool write activity. Similar buffer pool read and write information is available at the application level in accounting reports. For more information about monitoring the performance of buffer pools, see the Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS documentation.

10.2.5 Automating buffer pool management

After the buffer pool assignments, thresholds, and sizes are determined and buffer pool performance behavior is assessed, you use an automated scheduling tool to adjust buffer pool settings for various workloads.

Buffer pool sizes and thresholds are changed by issuing the ALTER BUFFERPOOL command. By using your system automation tool, you schedule these commands to occur throughout the day to reconfigure your buffer pools according to the expected workloads. Example 10-4 shows an altered VPSEQT value that allows for more sequential processing during the night and less processing during the day.

Example 10-4 Sample automation of buffer pool configuration changes

```
1800 to 0600 plus weekends:  
  ALTER BUFFERPOOL(BP22) VPSIZE(200000) VPSEQT(95)  
0600 to 1800 less weekends:  
  ALTER BUFFERPOOL(BP22) VPSIZE(200000) VPSEQT(10)
```

You set up your automation to alter the buffer pools at the desired times of the day. The changes take effect immediately.

10.2.6 Additional buffer pool considerations

DB2 10 for z/OS introduced the following new buffer pool features that you must consider when you are configuring your buffer pools:

- 1 MB page frames

For buffer pools defined with PGFIX=YES, DB2 requests that buffer pools are allocated by using 1 MB page frames (rather than 4 KB pages frames) if the frames are available. 1 MB page frames are available in IBM z10™ and later processors. You define the number of 1 MB page frames that are available to z/OS in the LFAREA parameter of SYS1.PARMLIB(IEASYSxx). Manipulating storage in 1 MB chunks rather than 4 KB chunks significantly reduces CPU memory management by increasing the hit ratios of the hardware translation lookaside buffer. For more information about 1 MB page frames, see *DB2 10 for z/OS Technical Overview*, SG24-7892.

- In-memory table spaces and indexes

You define some DB2 table spaces and indexes so that the spaces and indexes remain in memory when they are accessed. This configuration is done by defining a buffer pool with the option of PGSTEAL(NONE) and assigning the table space or index to that buffer pool. When the object is first accessed, an asynchronous task is scheduled under the DBM1 address space to prefetch the entire page set into the buffer pool. The pages remain resident if the object remains open.

In-memory objects provide considerable performance benefits by eliminating the need for I/O when the data is loaded into the buffer pool. For more information about in-memory table spaces and indexes, see *DB2 10 for z/OS Technical Overview*, SG24-7892.

10.3 CICS configuration

CICS Transaction Server is the IBM general-purpose transaction processing software for z/OS. This application server meets the transaction-processing needs of large and small enterprises. The server builds on z/OS and System z facilities to provide high availability and scalability at a low cost per transaction and supports large transaction volumes with a fast and consistent response time.

CICS Transaction Server provides the following features and benefits:

- ▶ An environment for running transactions; CICS Transaction Server manages concurrency, resource sharing, data integrity, and work prioritization
- ▶ Support for business applications that are written in COBOL, C, C++, PL/I, Java, and assembly language, by using an application programming interface that provides access to CICS services
- ▶ Access by applications to data stored in DB2 and DL/I databases, and in VSAM and BDAM data sets
- ▶ Interoperation with WebSphere MQ and access to the Message Queue Interface from CICS application programs
- ▶ Distribution of work between multiple CICS regions in a z/OS sysplex
- ▶ Connectivity with non CICS systems in client/server and peer-to-peer configurations
- ▶ Interfaces for configuring and managing your CICS regions
- ▶ Aids for debugging application programs and diagnosing system problems

Applications that are converted from other databases to DB2 for z/OS often include some component of online transaction workload. This workload might already run on CICS or some other OLTP software that is associated with the pre-relational database. In either case, you must configure CICS for new DB2 workload.

The following topics are of interest when DB2 workload is added to a CICS environment. For more information, see *CICS Transaction Server for z/OS Version 4 Release 2 DB2 Guide*, SC34-7164:

- ▶ CICS system initialization parameters
- ▶ DB2 threads in CICS
- ▶ CICS DB2 resource definitions
- ▶ Threadsafe programs

10.3.1 CICS system initialization parameters

You use the CICS system initialization parameters to modify CICS system attributes when you start your CICS regions. The information that is defined by system initialization parameters is grouped into the following categories:

- ▶ Information that is used to initialize and control CICS system functions (for example, the dynamic storage area limits and the region exit time interval)
- ▶ Module suffixes that is used to load your own versions of CICS control tables (for example, DFHMCTxx)
- ▶ Special information that is used to control the initialization process

The following CICS system initialization parameters are most pertinent to setting up your CICS environment for a new DB2 workload:

► **MXT**

This parameter specifies the maximum number (in the range of 1 - 999) of user tasks CICS allows to exist at any time. CICS queues the requests for tasks above this threshold but does not attach the requests until the number of attached tasks drops below the MXT limit.

► **MAXOPENTCBS**

This parameter controls the total number of L8 and L9 mode task control blocks (TCBs) that the CICS region has in operation at any time. Within this limit, there are no restrictions on the number of L8 and L9 TCBs in the pool.

CICS operates with an OPENAPI task-related user exit and uses L8 TCBs when connected to DB2 by using the CICS DB2 Attachment Facility.

The value of MAXOPENTCBS must be greater than the value of the TCBLIMIT parameter on the DB2CONN definitions for any of the CICS regions.

10.3.2 DB2 threads in CICS

There are several types of threads that are used by the CICS DB2 Attachment Facility, including, command threads, entry threads, and pool threads. These threads are defined in the CICS DB2 resource definitions, which are described in the next section.

When CICS is connected to DB2, the thread TCB is the open TCB on which the CICS DB2 attachment facility is running. When a thread must be run, the CICS DB2 attachment facility associates the open TCB with the DB2 connection control block and thread the facility wants to use, and the open TCB runs the thread. When the thread is not needed, the open TCB dissociates from the thread, and the DB2 connection control block and thread are available for reuse by another open TCB.

Entry threads are defined as protected or unprotected threads in the DB2ENTRY definition. Protected threads must be used for high-volume transactions or transactions that feature many commits. Protected threads reduce the cost of creating threads. Unprotected entry threads are used for low-volume transactions that do not require a fast response time. All transactions for unprotected threads are forced to use pool threads.

Protected threads and unprotected entry threads overflow to the thread pool and become pool threads. This overflow occurs when conditions defined in the CICS DB2 resource definitions are met.

10.3.3 CICS DB2 resource definitions

The CICS DB2 attachment facility provides a multithread connection to DB2. Defining the CICS DB2 connection involves several CICS resource definitions: DB2CONN (the DB2 connection definition), DB2ENTRY (the DB2 entry definition), and DB2TRAN (the DB2 transaction definition). The DB2CONN, DB2ENTRY, and DB2TRAN objects describe the global attributes of the CICS DB2 connection, the relationship between CICS transactions and DB2 resources (including application plans and command processors), the attributes of each type of thread, and the type of thread that each transaction uses.

The DB2CONN, DB2ENTRY, and DB2TRAN definitions of the CICS DB2 attachment facility define the authorization and access attributes on a transaction and transaction group basis.

You optimize the performance between CICS and DB2 by adjusting the transaction class limits, MXT system parameters of CICS, and the THREADWAIT, TCBLIMIT, THREADLIMIT, and PRIORITY attributes of DB2CONN and DB2ENTRY. An overview of the DB2CONN, DB2ENTRY, and DB2TRAN definitions is provided next.

For more information about CICS DB2 resource definitions, see *CICS Transaction Server for z/OS Version 4 Release 2 DB2 Guide*, SC34-7164.

DB2CONN

DB2CONN is the primary definition for the CICS DB2 connection. You must install a DB2CONN resource definition before you start the CICS DB2 connection. You use the DB2CONN definition to define the following types of attributes:

- ▶ Overall CICS DB2 connection
- ▶ Command threads, which are the special threads that are used by the DSNB transaction for issuing DB2 commands
- ▶ Pool threads, which are the general-purpose threads that are used when transactions do not need a special command or entry thread, or when there are no special threads available for a transaction to use

The following items are defined in the DB2CONN:

- ▶ DB2 subsystem to which CICS connects
- ▶ Number of TCBs that CICS uses to run threads into DB2 at any one time
- ▶ Duration that protected threads are kept before termination
- ▶ Limit on the number of times a thread is reused

All of these definitions (and others) must be determined based on your expected workload in DB2. Each CICS region has a single DB2CONN definition.

For more information about setting your DB2CONN definition, see *CICS Transaction Server for z/OS Version 4 Release 2 DB2 Guide*, SC34-7164.

DB2ENTRY

A DB2ENTRY definition specifies a CICS transaction, or a group of transactions, by using a wildcard. You set up a DB2ENTRY definition for each protected or unprotected entry thread you define.

You set up many DB2ENTRY definitions to define different types of entry threads. The entry threads are used by the transactions that you specify to gain priority access (or specialized access) to DB2 resources. You reserve some threads that are used only by those transactions. You also protect a number of each type of entry thread, which improves performance for transactions that are used heavily.

In a DB2ENTRY definition, you specify a particular transaction, or (by using a wildcard) a group of transactions, that are associated with the DB2ENTRY. The transactions also use the type of entry thread that the transaction defines. When those transactions request a thread, the CICS DB2 attachment facility provides the transaction with the necessary type of entry thread, if a thread is available. If you want other transactions to use the same type of entry thread, you create a DB2TRAN definition for those transactions. This definition tells CICS that the transactions are associated with a particular DB2ENTRY.

The following pertinent attributes are for the DB2ENTRY definition are:

PROTECTNUM

This attribute specifies the maximum number of protected threads that are allowed for this DB2 entry definition. A thread that is released by a transaction when no other work is queued is protected, meaning that the thread does not end immediately. A protected thread ends after two complete purge cycles if the thread is not reused in the meantime. If you specify a value of zero, the thread is not protected. If you specify a non-zero value, the number of transactions for that entry (up to the specified value) are protected.

THREADLIMIT

This attribute specifies the maximum number of threads for a DB2 entry definition that the CICS DB2 attachment allows active before requests are made to wait, are abended, or diverted to the pool.

THREADWAIT

This attribute specifies whether transactions wait for a DB2ENTRY thread, end abnormally, or overflow to the pool if the number of active DB2ENTRY threads reaches the THREADLIMIT number. THREADWAIT uses one of these values:

- ▶ **POOL:** If all threads are busy, the transaction is diverted to use the pool of threads. If the pool is busy and you specified THREADWAIT(NO) on the DB2 connection (DB2CONN) definition, the transaction ends abnormally with abend code AD3T.
- ▶ **NO:** If all threads are busy, the transaction ends abnormally with abend AD2P.
- ▶ **YES:** If all threads are busy, the transaction waits until a thread becomes available.

The combination of DB2ENTRY attributes you use for your transactions depends on the type of transaction. The following recommendations are for different types of transactions:

- ▶ **Protected threads:** For high-volume transactions or transactions with many commits; define with PROTECTNUM(n) and THREADLIMIT(n)
- ▶ **Unprotected entry threads for critical transactions:** Define with PROTECTNUM(0) and THREADLIMIT(n)
- ▶ **Unprotected entry threads for background transactions:** Define with PROTECTNUM(0), THREADLIMIT(0), and THREADWAIT(PPOOL)

DB2TRAN

A DB2TRAN resource defines a transaction, or a group of transactions, that is associated with a DB2ENTRY that are in addition to the transactions specified in the DB2ENTRY. Only one DB2TRAN definition is installed for a specific transaction. The DB2TRAN definition allows a DB2ENTRY to have an unrestricted number of transactions that are associated with it, including names that use wildcard characters. You define any number of DB2TRANS associated with a single DB2ENTRY.

You do not specify on a DB2TRAN entry whether it is protected or unprotected, nor do you specify a limit on the number of active threads or whether the thread overflows to the pool. You specify the DB2ENTRY name to which a DB2TRAN entry is associated. By using this definition, you have an unlimited number of transactions that are associated with a DB2ENTRY resource definition.

10.3.4 Threadsafe programs

The CICS DB2 attachment facility uses the open transaction environment (OTE) to enable the CICS DB2 task-related user exit to start and return from DB2 without switching TCBs. In the OTE, the CICS DB2 task-related user exit operates as a threadsafe and an open API task-related user exit program. The CICS DB2 task-related user exit is automatically enabled by using the OPENAPI option on the ENABLE PROGRAM command during connection processing. This configuration enables the exit to receive control on an open L8 mode TCB. Requests to DB2 also are issued on the L8 TCB, so the L8 TCB acts as the thread TCB, and no switch to a subtask TCB is needed.

In the OTE, if the user application program that started the task-related user exit conforms to threadsafe coding conventions and is defined to CICS as threadsafe, the program also runs on the L8 TCB. Before its first SQL request, the application program runs on the CICS main TCB the quasi reentrant (QR) task control block (TCB). When the program makes an SQL request and starts the task-related user exit, control passes to the L8 TCB, and DB2 processing is carried out. On return from DB2, if the application program is threadsafe, the program continues to run on the L8 TCB.

Where the correct conditions are met, the use of open TCBs for CICS DB2 applications decreases the usage of the QR TCB and avoids TCB switching. An ideal CICS DB2 application program for the OTE is a threadsafe program, which contains only threadsafe EXEC CICS commands, and uses only threadsafe user exit programs. An application such as this moves to an L8 TCB when it makes its first SQL request, and then continues to run on the L8 TCB through any amount of DB2 requests and application code, requiring no TCB switching. This situation produces a significant performance improvement in which an application program issues multiple SQL calls. The gains also are significant when you are using an enterprise bean, because enterprise beans require more TCB switches to and from the TCB of the enterprise bean when a request is made. If the application program issues few SQL calls, the performance benefits might not be as significant.

CICS switches back to the QR TCB if running the program involves any of the following actions that are not threadsafe:

- ▶ Non-threadsafe CICS requests issued by the program
- ▶ Use of non-threadsafe dynamic plan exits
- ▶ Use of non-threadsafe, task-related user exits
- ▶ Inclusion of non-threadsafe global user exits

Switching back and forth between the open TCB and the QR TCB is detrimental to the performance of the application.

The details of how to code DB2/CICS applications that are threadsafe are beyond the scope of this book. For more information about coding DB2 applications to be threadsafe in CICS, see *CICS Transaction Server for z/OS Version 4 Release 2 DB2 Guide*, SC34-7164. For more information about changing application programs and user exit programs so that they are threadsafe, see *CICS Transaction Server for z/OS Version 4 Release 2 Application Programming Guide*, SC34-7158.

By defining a program to CICS as threadsafe, you are specifying that only the application logic is threadsafe, not that all of the EXEC CICS commands included in the program are threadsafe. CICS ensures that EXEC CICS commands are processed safely by switching to the QR TCB for those commands not yet converted that still rely on quasi-reentrancy. To allow your program to run on an open TCB, CICS needs you to guarantee that your application logic is threadsafe.

If you want to take advantage of the performance benefits of threadsafe programming, it is your responsibility to ensure that your CICS programs, and any programs that are generated by your conversion vendor, are written in accordance with threadsafe programming techniques.

10.4 z/OS operating system

DB2 for z/OS runs on the z/OS operating system. z/OS is designed to offer a stable, secure, continuously available, and scalable environment for applications that are running on System z hardware. As you migrate workloads to DB2 for z/OS, you must be aware of the version of DB2 on which your application runs and the version of the z/OS operating system on which your application runs. Some features of DB2 9 for z/OS and DB2 10 for z/OS require minimum levels of the z/OS operating system. In this section, we describe the following features of z/OS that you must consider when you are converting your workload to DB2 for z/OS:

- ▶ Supported z/OS release levels
- ▶ Workload Manager

10.4.1 Supported z/OS release levels

DB2 10 for z/OS requires a minimum level of z/OS V1.10. Some features of DB2 10 for z/OS require later versions of z/OS or other maintenance beyond the base level of z/OS V1.10. For more information about these prerequisites, see the DB2 10 for z/OS documentation.

10.4.2 Workload Manager

z/OS Workload Manager (WLM) is a component of z/OS that manages incoming work requests for the operating system and allocates available system resources to meet these requests. WLM manages this work by allowing specific goals for different categories of work. You set these goals for all work that runs on z/OS.

With regards to DB2 work, you must ensure that the IMS/Vs Resource Lock Manager (IRLM) address space is assigned to a service class that receives one of the highest MVS dispatching priorities (a service class of SYSSTC often suffices). IRLM needs this priority because IRLM manages resources that might or might not be accessed by work within the DB2 address spaces.

The following DB2 address spaces are assigned to service classes the goals of which are similar, but result in an MVS dispatching priority below the dispatching priority for the service class for the IRLM address space:

- ▶ DB2 system services address space (MSTR)
- ▶ DB2 database services address space (DBM1)
- ▶ Distributed data facility address space (DIST)

For more information about defining the WLM environments for DB2, see *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974. For more information about how z/OS uses WLM to manage workloads, see *DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7604.



Application and SQL tuning

To truly consider a conversion a success, we must ensure that the performance meets the service level agreements and expectations of the users.

After a conversion is conducted, whether by using a tool, a manual process, or a combination of both, there are some tuning efforts that must be completed to fully use DB2 for z/OS and ensure that the access to the new database and application is as efficient as possible.

In this chapter, we describe some of the tuning efforts that help achieve performance goals with the new database.

This chapter includes the following topics:

- ▶ Application tuning
- ▶ SQL tuning and optimization
- ▶ IBM tools for application and SQL tuning

11.1 Application tuning

When code is delivered from a conversion or migration effort, the code might not be optimized for the DB2 for z/OS environment. There are many issues to consider so that the application runs efficiently.

11.1.1 Reducing number of SQL statements

The number of times DB2 is called adds up quickly. For each SQL statement that is run, DB2 must call across the address space to run the statement and return the result. If statements that result from the conversion are simple and frequently run, the performance suffers. Many things are reviewed here to help improve performance: use of multi-row fetch, more advanced statements, combining statements (SELECT FROM INSERT), and so on. There are many options that are described in this section about the use of the features of DB2 to reduce the number of calls.

11.1.2 Eliminating programmatic joins

When a conversion occurs that uses simple I/O methods to the table, you often have an SQL SELECT from one table followed by an SQL SELECT from related data in another table and the two results are joined in the program. These joins are known as programmatic joins and compared to a join inside DB2, these joins are expensive (approximately 30% more CPU to run). If this type of join is occurring often, the joins have a negative affect on performance. If applicable indexing is completed and the statistics are reflective of the data so that DB2 chooses the correct join sequence, it is difficult to outperform a DB2 join versus a join in the program. A DB2 join is shown in Example 11-1.

Example 11-1 DB2 join

```
SELECT COL1, COL2
  FROM TABLEA A LEFT JOIN TABLEB B
    ON A.COL3 = B.COL1
 AND A.COL4 = :hv1
```

11.1.3 Remove cursor in cursor processing

Often our mature code includes opening a cursor to fetch a row of data and then opening another cursor to fetch within that row. This type of operation is easily converted to a subquery or even a join. Some considerations for cursors include not opening a cursor several times, but instead by using an inner or outer join, a subselect with an EXISTS or an IN list. These changes reduce the run time for some processes from hours to minutes, depending on the amount of data processed. Cursors in cursors often indicate a relationship (either mandated or optional), and must be written as a single statement. An example of a cursor within a cursor is shown in Example 11-2 on page 177.

```
DECLARE CUR1 CURSOR FOR
    SELECT DEPT, DEPTNAME, MGRNO
FROM DSN81010.DEPT
DECLARE CUR2 CURSOR FOR
    SELECT EMPNO
    FROM DSN81010.EMP
    WHERE EMPNO = :hvm
    AND EDLEVEL > 17

OPEN CUR1
DO LOOP1
FETCH CUR1 INTO :hv, :hvd, :hvm
IF SQLCODE = 0 THEN
OPEN CUR2
DO LOOP2
    FETCH CUR2 INTO :hv3
    IF SQLCODE = 0 FETCH AGAIN
    ELSE ENDLOOP2
ENDLOOP2
ENDLOOP1
```

This code might be changed to a subquery:

```
SELECT DEPTNO, DEPTNAME
FROM DSN81010.DEPT
WHERE MGRNO IN
(SELECT EMPNO FROM DSN81010.EMP
WHERE EDLEVEL > 17)
```

This code also might be changed to a join:

```
SELECT DEPTNO, DEPTNAME
FROM DSN81010.DEPT, DSN81010.EMP
WHERE MGRNO = EMPNO
AND EDLEVEL > 17
```

11.1.4 Filter in DB2, not in program

Mature applications often are responsible for all of the work in the application code. When a file was retrieved, the application code conducted all of the data selecting and filtering that was required by the user. In DB2, a powerful DBMS engine is used to get the most benefit from filtering. We do not want to treat DB2 as an access method, but rather as an intelligent processing engine. Therefore, when it comes to selecting and retrieving data, often the more independence-friendly (IF) logic you perform when following an SQL statement, the more tests you are missing in your SQL statement and the more expensive DB2 is because you are using DB2 only to move around data. Instead, you must move the program functionality work into the SQL statements. For example, if the following program tests are displayed after fetches from a cursor, every row from the table was returned to the program for the calculations:

```
IF RCODE = 'B'
THEN AMOUNT = AMOUNT + B-AMOUNT
ELSE IF RCODE = 'C'
THEN AMOUNT = AMOUNT + C-AMOUNT
```

```
ELSE IF RCODE = 'D'
THEN AMOUNT = AMOUNT + D-AMOUNT
```

This logic (fetches and calculations) is transferred into the following SELECT clause:

```
SELECT SUM(CASE WHEN RCODE = 'B' THEN AMOUNT END)
      , SUM(CASE WHEN RCODE = 'C' THEN AMOUNT END)
      , SUM(CASE WHEN RCODE = 'D' THEN AMOUNT END)
FROM TABLE
INTO :b-amount, :c-amount, :d-amount
```

When CASE statements are considered Stage 2 predicates, the statements are performed better in DB2 than passing the data on to the program to conduct the filtering.

11.1.5 Selective column retrieval

Because we are managing mature code that expects all or most of the data back from a format other than a normalized table, we often see that more data is returned than the application actually needs. This event also occurs when I/O modules are developed as a result of the conversion. This development is costly in DB2 for z/OS because cost is associated with every returned column. When DB2 brings pages of data into the buffer pool, DB2 pulls one column at a time across memory back to the application. The more columns that are selected, the higher the cost for the return. This situation also is compounded by the number of rows returned. Efforts must be made to select only the columns that are used by the application.

In essence, retrieve only exactly what is needed. Never use SELECT *, or list all of the columns in the table unless the columns are required by the application. Selecting a column value for the sake of including the value in the ORDER BY clause is unnecessary. Also, if you provide the column value in the WHERE clause, it is not necessary to retrieve the value in the SELECT clause:

```
SELECT EMPNO, LASTNAME, SALARY
FROM DSN81010.EMP
WHERE EMPNO = :hv
```

Must be:

```
SELECT LASTNAME, SALARY
FROM DSN81010.EMP
WHERE EMPNO= :hv
```

There was no reason to retrieve the EMPNO column because its value is known. The access path is based on the predicates. Retrieving the extra column moved the column from the page in the buffer pool to a user work area, passed to stage 2, and returned cross-memory to the work area of the program. This configuration is a waste of CPU, especially if the SQL is running thousands of times a day.

Ensuring that you are using all of the host variables that are populated in the program is good practice.

11.1.6 Organize transactions to avoid contention and avoid deadlocks

One reason organizations convert to DB2 for z/OS is to scale an application beyond its boundaries (and thus run other transactions). If there are contention issues in the old systems, the issues must be fixed before an increase in workload is introduced.

Transaction design is dictated by the following primary rules:

- All non-SELECT SQL (INSERTs, UPDATEs, and DELETEs) is at the end of commit scope
- All non-SELECT objects must be processed in a predefined order

These two rules significantly improve performance. If all the data manipulation language (DML) is at the end of the transaction, the DML does not affect the logic, but reduces lock wait time for other processes. This condition reverts to the amount of lock wait time that is accumulated in systems with a high number of concurrent users. We must do everything possible to reduce lock wait time.

The second rule is important but more so in systems that are built with modular concepts. This rule prevents those applications that are designed to deadlock. Deadlocks almost never happen on their own because they must be “designed” into the system. If all processes were to process tables with DML in a predetermined sequence, an application triggered deadlock does not occur.

It might not be possible to change all transactions, but a few transaction might benefit from some reworking if the transactions are the primary transactions. If you want to identify which transactions might benefit from some redesign to minimize locks, you take advantage of DB2 performance tools to measure locking behavior. IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS produces statistics and accounting reports that show locking behavior at the system level and the application level. For more information about IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, see 11.3.2, “IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS” on page 196.

11.1.7 Using optimistic locking to improve concurrency

Use optimistic locking and test whether another transaction changed the underlying data source column since the last read operation. This testing confirms that the resource is not held before the decision is made whether to update the resource. You read the data with an isolation level of uncommitted read (UR) and then check the timestamp that is maintained by DB2. To set up the ability to use optimistic locking, you must define a column in the table with FOR EACH ROW ON UPDATE AS ROW CHANGE TIMESTAMP and use the timestamp for comparison during the update. Example 11-3 shows the SELECT statement that is used to retrieve the ROW CHANGE TIMESTAMP for checking at UPDATE time.

Example 11-3 Example of SELECT that uses optimistic locking

```
SELECT EMPNO, SALARY, ROW CHANGE TIMESTAMP FOR DSN81010.EMP
FROM DSN81010.EMP
WHERE EMPNO = :EMP-ID WITH UR
```

After the data is selected (including the timestamp) and the EMP_ID is known, you issue the update statement and check the row change timestamp to see whether the data changed since the last read. Example 11-4 shows the syntax for UPDATE that uses optimistic locking.

Example 11-4 Example UPDATE using optimistic locking

```
UPDATE DSN81010.EMP
SET SALARY= :SALARY
WHERE EMPNO= :EMP_ID
AND ROW CHANGE TIMESTAMP FOR DSN81010.EMP= :UPD-TSP
```

This coding method takes no locks on the initial SELECT, and then checks to ensure that the row is updated since it was read.

11.1.8 Reducing sort cost

Any code that starts a sort must be reviewed to ensure that correct indexes are in place to help with sort avoidance. Some sorts are avoided if index keys are in the order they are needed by an ORDER BY, GROUP BY, a join operation, or DISTINCT in an aggregate function. In other cases, such as when list prefetch is used, the index does not provide useful ordering, and the selected data is sorted.

When it is necessary to prevent a sort, consider creating an index on the column or columns necessary to provide the wanted ordering. Consider also the use of the clause OPTIMIZE FOR 1 ROW to discourage DB2 from choosing a sort for the access path.

Consider the following query:

```
SELECT C2, SUM(C3)
FROM T1
WHERE C1 = 17
GROUP BY C2;
```

An ascending index on C1 or an index on (C1, C2, C3) eliminates a sort.

Sort cost also is reduced by ensuring selective column retrieval and keeping VARCHAR columns out of the sort record if the columns are not needed because the columns are padded to the full length.

11.1.9 Ensuring appropriate use of external stored procedures

Some tools produce I/O modules to support a black box approach to a migration in which code is not necessarily changing. Often, these modules are implemented by using DB2 external stored procedures. Although the use of external stored procedures is an advantage in a mature conversion to encapsulate business logic, you must be mindful of the performance issues if the stored procedures are used as I/O modules for each table. If the external stored procedure is issuing a single statement to access the table, then the statement is misused and, in high volumes, is detrimental to performance. These statements are better in a program than in an external stored procedure.

External stored procedures that are run in a Workload Manager (WLM) that managed address spaces; therefore, a cross memory call must be made to use the stored procedures. Because there is a cost that is associated with the cross memory call, you do not want to use the calls excessively.

However, because external stored procedures run in stored procedure address spaces, the address spaces access the following resources that any other address space in the system accesses:

- ▶ VSAM files
- ▶ Flat files
- ▶ IMS data

This access might be useful during a conversion process to quickly enable data in mature data stores that are returned by the stored procedures through result sets.

11.1.10 Using basic SQL procedures

Another option during the conversion is to use basic SQL procedures. This option often is more useful when you are converting from other relational databases because the SQL procedure language is common among relational databases and across platforms. However, this option might be used in conversion from non-relational sources as well.

DB2 provides support for basic SQL procedures that run entirely within DB2 and do not require a WLM-managed address space. By running the SQL procedure in the DBM1 address space, the stored procedure invocation overhead and the round-trip cost of switching between WLM and the DBM1 address space for each SQL call is avoided. The SQL procedure workload is eligible to run on a System z Integrated Information Processor (zIIP) engine if the call to the procedure is a DDF request through the DRDA because it runs in the DBM1 address space under a DDF enclave SRB. Workloads that run on a zIIP engine result in reduced software costs that are based on CPU usage.

These types of procedures are ideal for a few short-running statements, rather than an encapsulation of several statements, such as with an external procedure.

11.1.11 Promoting the use of dynamic statement cache

Access to the new data store in DB2 for z/OS might include dynamic SQL. DB2 is able to make dynamic SQL efficient by avoiding unnecessary prepares if the statement is found in the global statement cache.

The dynamic statement cache is an area of memory that is reserved inside DB2 for the storage of the text of dynamic SQL statements along with the compiled runtime structures associated with the prepared statement. The primary purpose of this cache is to avoid the overhead of preparing again a statement for every execution. The statement prepare process, in which DB2 determines the access path, is expensive, especially for complicated statements, and results in high CPU costs. If the user and statement do not change, and no table structures or statistics changed in the database, the statement does not have to be prepared again and the previous executable version is reused.

A potential performance issue is that dynamic statements must be bound to the database during a prepare process much in the same way that static statements are bound. During the bind process, statements are validated, object definitions and statistics are accessed in the DB2 system catalog, the access paths are determined with the optimal path chosen, and the runtime structures are created and stored. This process is expensive, especially for more complicated statements. The dynamic statement cache is used to retain the output from the prepare process across executions so that it is reused, thus avoiding the prepare overhead. If an incoming statement matches the authorization ID and SQL text (byte for byte), and the object structures and catalog statistics remain unchanged, the runtime structures are reused. This configuration dramatically improves the performance of some dynamic statements.

Some applicable coding techniques are needed to take advantage of the dynamic statement cache. The statements that run often and repetitively are the best statements to take advantage of the cache and are often associated with transactions. In order for a statement to match in the statement cache, the statement must match an existing statement, byte for byte, for the particular authorization ID. If any part of the statement does not match, the statement must go through the prepare process. Coding SQL statements in a way that provides a good chance of matching in the cache by using *parameter markers* for variables in the statement is important.

If embedded literals are used, and the literal values change across statement executions, there is little or no matching of the statement in the cache. In DB2 10, there is *literal replacement* which is an option to replace the literals with an '&' to try to get a better match in the cache. Constants must be used for authorization IDs because a statement matches only in the cache for the same authorization ID. Using a three-tier architecture with connection pooling helps to increase the matching in the statement cache. Some cost is associated with the matching, but that cost is small compared to the cost of the prepare process. However, if most of statements that are run are not correctly matching, it might be better not to use the cache.

11.1.12 Converting single row processing to multi-row fetch or insert

This feature greatly improves the performance of distributed applications and local applications that perform several fetches in a cursor. The feature also allows for multiple rows to be INSERTed or FETCHed in a single API call. The feature is supported by static, dynamic, non-scrollable, and scrollable cursors.

Again, we want to reduce the number of calls that are made to the database by the application. By making some simple changes to code after a conversion, we use this feature to improve performance.

Example 11-5 shows a DECLARE CURSOR statement that uses multi-row fetch. Note the WITH ROWSET POSITIONING clause, which indicates that the cursor is enabled for multi-row fetch.

Example 11-5 Example of a multi-row FETCH cursor

```
DECLARE CUR1 CURSOR
WITH ROWSET POSITIONING
FOR SELECT EMPNO, SALARY FROM DSN81010.EMP;
OPEN CUR1;
FETCH FROM CUR1
FOR :hv ROWS INTO :values1, :values2;
```

Example 11-6 shows an INSERT statement that inserts multiple rows. The "FOR :hv ROWS" clause defines the number of rows that are inserted. The host variables within the VALUES clause are arrays that contain the data for the number of inserted rows.

Example 11-6 Example of a multi-row INSERT statement

```
INSERT INTO DSN81010.EMP FOR :hv ROWS
VALUES (:values1, :values2) ATOMIC
```

This feature is easy to implement, but some performance testing must be done. Although it is possible to fetch as many as 32767 rows in a single API call, there is a point of diminishing returns (often around 50 - 100 rows). Not all workloads see the same benefits. For example, a program that reads many rows sequentially might see more CPU savings than a program that reads many rows randomly. The same considerations are true for INSERTs. You must be reasonable concerning the size of your array that is processed and consider that you are still logging and row size matters.

You also must consider whether you implement multi-row fetch within called I/O modules or change your application to return multiple rows in a single call. If you implement multi-row fetch within called I/O modules, and your applications make calls to the I/O module interface to return data a row at a time, the amount of savings provided by multi-row fetching is offset by a high amount of calls to return each row to the application.

In general, multi-row FETCH and INSERT provides considerable performance gains for applications that read and write a group of rows at a time. However, most automated conversions do not use multi-row operations because of the record-at-a-time processing when I/O modules are called to run SQL statements. You might need to make a conscious effort to rewrite those applications that best take advantage of multi-row processing.

11.1.13 Combine statements by using SELECT FROM INSERT

Often conversions result in several calls to DB2 because of a modularized implementation. Calling DB2 repeatedly for simple retrieval is expensive. Opportunities are available to combine statements when we are retrieving values that are generated by INSERT statements.

This SQL feature provides our application powerful functionality by giving us the ability to combine statements. The INSERT within SELECT capability gives us the ability to retrieve the following values that are created or modified during an INSERT:

- ▶ Identity column or sequence value
- ▶ User-defined defaults and expressions
- ▶ Values that are modified by BEFORE triggers
- ▶ ROWIDs
- ▶ CURRENT_TIMESTAMP value

The values are retrieved for an inserted row without specifying column names. This functionality is supported by the keywords FINAL TABLE and INPUT SEQUENCE. Example 11-7 shows that you obtain the value of an identity column after you perform an INSERT (assuming that the table is created with an identity column on EMPNO defined with GENERATED ALWAYS):

Example 11-7 Sample INSERT within SELECT statement

```
SELECT EMPNO FROM FINAL TABLE (INSERT INTO DSN81010.EMP (FIRSTNAME,  
MIDINIT, LASTNAME) VALUES ('JOSEPH', 'D', 'SMITH') )
```

By using this feature, the number of SQL statements and application code that is run to obtain and use these values is reduced. This reduction affects the overall application performance.

11.1.14 Combining statements with SELECT FROM UPDATE or DELETE

There are instances in which statements are consolidated by using SELECT FROM UPDATE or DELETE to obtain values as the result of an UPDATE or a DELETE.

You select values from rows that are updated by specifying the UPDATE statement in the FROM clause of the SELECT statement. When you update one or more rows in a table, you retrieve the following values:

- ▶ Automatically generated column values, such as a ROWID or identity column
- ▶ Any default values for columns
- ▶ All values for an updated row, without specifying individual column names

In most cases, you want to use the FINAL TABLE clause with SELECT FROM UPDATE statements. The FINAL TABLE consists of the table rows or view after the update occurs. For example, suppose that all designers for a company are receiving 5% raises. You use the SELECT FROM UPDATE statement that is shown in Example 11-8 on page 184 to increase the salary of each designer by 5% and to retrieve the total increase in salary for the company.

Example 11-8 Sample SELECT FROM UPDATE statement

```
SELECT SUM(SALARY) INTO :SALARY FROM FINAL TABLE
(UPDATE DSN81010.EMP SET SALARY = SALARY * 1.05
WHERE JOB = 'ENGINEER');
```

You also select values from rows that are deleted by specifying the DELETE statement in the FROM clause of the SELECT statement. When you delete one or more rows in a table, you retrieve the following values:

- ▶ Any default values for columns
- ▶ All values for a deleted row, without specifying individual column names
- ▶ Calculated values that are based on deleted rows

When you use a SELECT FROM DELETE statement, you must use the FROM OLD TABLE clause to retrieve deleted values. The OLD TABLE consists of the table rows or view before the delete occurs. For example, suppose that a company is eliminating all operator positions and the company wants to know the amount of salary money it saves by eliminating these positions. You use the SELECT FROM DELETE statement that is shown in Example 11-9 to delete operators from the EMP table and to retrieve the sum of operator salaries.

Example 11-9 Sample SELECT FROM DELETE statement

```
SELECT SUM(SALARY) INTO :SALARY FROM OLD TABLE
(DELETE FROM DSN81010.EMP
WHERE JOB = 'OPERATOR');
```

Both the SELECT FROM UPDATE and SELECT FROM DELETE statements often require changes to the converted application code because most conversion tools do not build this type of logic for you. You must review situations in which the converted code results in two separate operations and determine whether you convert to one of these constructs.

11.1.15 Using the SQL scalar fullselect

Expressions in SQL are placed almost anywhere in the SQL statement. By using this placement, you build some powerful SQL statements and add logic and processing in the statement. A SELECT statement is coded within a SELECT, a WHERE clause, or CASE expression. Scalar fullselects within a SQL statement are used for performance and also to help reduce the quantity of SQL statements issued.

Example 11-10 shows the use of the scalar fullselect in a WHERE clause. In this clause, we are comparing the value of the SALARY column from the EMP table to the results of two other SELECTs in which the minimum and maximum value for the same column of the table are obtained. Although this example is simple, it shows how you use scalar fullselect in the WHERE clause.

Example 11-10 Scalar fullselect in a WHERE clause

```
SELECT EMPNO
FROM DSN81010.EMP A
WHERE SALARY BETWEEN
(SELECT MIN(SALARY) FROM DSN81010.EMP)
AND
(SELECT MAX(SALARY) FROM DSN81010.EMP)
```

Example 11-11 shows a statement in which a SELECT is used in the SELECT list which finds the maximum salary of all employees for a particular manager of a department.

Example 11-11 Scalar fullselect in SELECT list

```
SELECT MGRNO, DEPTNO,  
      (SELECT MAX(SALARY)  
       FROM DSN81010.EMP B  
       WHERE A.MGRNO = B.EMPNO)  
FROM DSN81010.DEPT A  
WHERE MGRNO = 1234
```

The ability to combine statements by using the scalar fullselect helps produce more intelligent SQL statements and reduce the number of calls that are made to DB2.

11.1.16 Using MERGE for synchronization

During a conversion or migration, we might handle existing synchronization requirements. Some past techniques to conduct the replication involved an attempt to update the data and, if the data did not exist, the data was inserted. In DB2 for z/OS, this process results in several calls to DB2 and much overhead. This type of process is simplified in DB2 for z/OS by using the MERGE statement.

The MERGE statement updates a target (a table or view, or the underlying tables or views of a fullselect) by using the specified input data. Rows in the target that match the input data are updated as specified, and rows that do not exist in the target are inserted. Updating or inserting a row in a view results in an update or insert of the row in the tables on which the view is based. MERGE is embedded in an application program or issued interactively. MERGE also is an executable statement that is dynamically prepared. The update or insert is completed in a single SQL statement.

You update existing data and insert new data in a single operation by using the MERGE statement. This operation is useful when you want to update a table with a set of rows, some of which are changes to existing rows and or new rows. A multi-row merge also is available. A use often has many rows to merge. As with multi-row insert, we use arrays to provide the multiple rows of data to merge into the table.

For example, an application might request a set of rows from a database, allow a user to modify the data through a GUI, and then store the modified data in the database. Some of the modified data results in updates to existing rows, and some of the modified data results in inserts of new rows. You conduct these updates and insert operations in one step.

To update existing data and insert new data, specify a MERGE statement with the WHEN MATCHED and WHEN NOT MATCHED clauses. These clauses specify how DB2 manages matched and unmatched data. If DB2 finds a matching row, that row is updated. If DB2 does not find a matching row, a row is inserted.

Example 11-12 shows an example of a MERGE statement.

Example 11-12 Sample MERGE statement

```
MERGE INTO DSN81010.ACCT AS T
  USING (VALUES(?,?) FOR 5 ROWS) AS S(ID,AMT)
  ON T.ID = S.ID
  WHEN MATCHED THEN
    UPDATE SET BALANCE = T.BALANCE + S.AMT
  WHEN NOT MATCHED THEN
    INSERT (ID,BALANCE) VALUES(S.ID,S.AMT)
  NOT ATOMIC CONTINUE ON SQLEXCEPTION;
```

The MERGE statement is useful for a post-conversion that takes another data feed and must merge the data into the populated database. The MERGE statement allows the program to update data that was loaded as part of the conversion process and insert data if it was not loaded as part of the conversion process.

11.1.17 Combining statements with SELECT FROM MERGE

To further streamline the UPDATE/INSERT process, you also take advantage of the ability to SELECT values generated during a MERGE. You select values from rows that are merged by specifying the MERGE statement in the FROM clause of the SELECT statement. When you merge one or more rows into a table, the following values are retrieved:

- ▶ The value of an automatically generated column, such as a ROWID or identity column
- ▶ Any default values for columns
- ▶ All values for a merged row, without specifying individual column names
- ▶ Calculated values that are based on the changes to the merged rows

You want to use the FINAL TABLE clause with SELECT FROM MERGE statements. The FINAL TABLE consists of the table rows or view after the merge occurs.

11.1.18 Close cursors

Some conversion processes do not insert the CLOSE CURSOR command after a cursor completed its processing. You want to close a row cursor when the cursor completes processing rows if you want to free the resources or reuse the cursor. To free the resources that are held by the cursor, close the cursor explicitly by issuing the CLOSE statement. There must be a CLOSE CURSOR for every OPEN CURSOR. An example of the syntax for CLOSE cursor is shown in Example 11-13.

Example 11-13 CLOSE cursor syntax

```
EXEC SQL
  CLOSE CUR1
END-EXEC
```

Some automated conversion processes result in cursors that are declared in which they are not needed. In some instances, the cursor is declared WITH HOLD. Application developers must be aware of the state of cursors; it is the responsibility of application program to close the cursor upon return from an I/O module.

11.2 SQL tuning and optimization

After the application is tuned, efforts are made to improve the SQL. We must be careful when the SQL statements are reviewed after a conversion because the statements might seem perfect in the sense that the access path looks accurate. But if the SQL statement is simple (such as used for I/O), the statement likely is run often and not useful. This status degrades overall performance and CPU consumption. We must ensure that we have intelligent SQL statements, not another access method.

The topics in this session describe techniques used to tune the SQL statements that were created during a conversion to increase performance.

11.2.1 Improving sequential access

Many mature applications include large processes that process data sequentially. While these processes were optimal when they were run in the mature data store, they are not as efficient when converted to DB2. We must review how the data in each table is clustered and ensure that the data is clustered in a sequence that supports the sequential processes. This process is done through the creation of a clustering index (see 12.14, “Clustering index” on page 210). Ensuring good clustering also applies to tables that are joined frequently. Ensuring that joined tables are clustered in the same sequence so the join does not suffer from excessive random I/O is worth the effort.

We also want to take advantage of efficiencies, such as sequential prefetch and well-organized indexes. Therefore, we want to ensure that `RELEASE(DEALLOCATE)` is used on our binds and that the data and index are organized well.

11.2.2 Improving index usage

Ensuring our predicates (both join and filtering) have applicable indexes helps improve the performance of processes in the relational environment. However, we must be cautious of adding too many indexes as the indexes negatively affect the insert, update, and delete processes (about 25% CPU overhead). To determine the applicable indexes for our workload, we use the InfoSphere Optim Query Workload Tuner for DB2 for z/OS. This tool analyzes the query workloads and recommends applicable indexes. For more information, see 11.3.1, “InfoSphere Optim Query Workload Tuner for DB2 for z/OS”.

11.2.3 Ensuring appropriate predicate filtering

Predicate processing and filtering is an important part of SQL processing. Mature programs often are coded to retrieve a large amount of data and then process and filter the data in the program code. In DB2, we want to filter in the DB2 engine by using SQL. For optimal performance, the filtering must be done as early as possible so predicates must be applied to the data.

Rows that are retrieved for a query go through two stages of processing. Certain predicates are applied during the first stage of processing, whereas other predicates are not applied until the second stage of processing. You improve the performance of your queries by using predicates that are applied during the first stage whenever possible.

Predicates that are applied during the first stage of processing are called *Stage 1 predicates*. These predicates are also sometimes said to be *sargable*. Similarly, predicates that are not

applied until the second stage of processing are called *Stage 2 predicates*, and are sometimes described as *non-sargable* or *residual* predicates.

Whether a predicate is stage 1 or stage 2 depends on the following factors:

- ▶ The syntax of the predicate
- ▶ Data type and length of constants or columns in the predicate
- ▶ Whether DB2 evaluates the predicate before or after a join operation
- ▶ Join sequence

Applicable indexing helps the performance so that we have stage 1 *indexable* predicates (range-delimiting) and then we determine whether it is possible to push down the predicates. We might look to the code and to see whether any processing (filtering) is done by the application that must be done in DB2. For example, there might be code that resembles the code that is shown in Example 11-14.

Example 11-14 Sample of filtering in program code

```
FETCH ..... INTO ..... END-EXEC

IF (BONUS + COMM) < (10% OF SALARY) THEN
  PERFORM FETCH-NEXT-ROW
END-IF
```

This type of filtering that is completed in the application after the data is passed through the various stages of DB2 is an inefficient use of DB2 because DB2 is treated like an access method. We push this filtering into the DB2 engine by moving the code into the SQL statement, as shown in Example 11-15.

Example 11-15 Sample of filtering in SQL statement

```
SELECT EMPNO, LASTNAME
FROM DSN81010.EMP
WHERE BONUS + COMM >= SALARY * 0.1
```

Although this predicate is stage 2, it is better than filtering in the application code.

Applicable predicate filtering is a common issue in converted applications because the pre-relational database often does not have the predicate filtering capabilities that are available in SQL statements.

11.2.4 Using common table expressions

Common table expressions provide performance improvements by computing a value once (not multiple times) during the execution of a query. Common table expressions are used in SELECT, CREATE VIEW, and INSERT. The expressions also might contain references to host variables.

A common table expression allows the programmer to create powerful statements. The common table expression that is created lasts only for the length of a statement, and is nested and reused throughout the statement. The expressions are useful for situations in which data must be built as needed, and then reused multiple times within a query. An example of a common table expression is shown in Example 11-16 on page 189.

Example 11-16 Sample common table expression

```
WITH DEPTSUM (DEPT_NO, MAXSALARY) AS
  (SELECT WORKDEPT, SUM (SALARY+COMM)
   FROM DSN81010.EMP
   GROUP BY WORKDEPT)
SELECT DEPT_NO FROM DEPTSUM
WHERE MAXSALARY =
  (SELECT MAX(MAXSALARY)
   FROM DEPTSUM);
```

DEPTSUM is the common table expression here and is built by calculating the sum of the salary plus commissions for each department. The common table expression is materialized in database DSNDB07 upon first reference in the SQL statement. The materialized result is reused for further references in the same statement.

The use of common table expressions is a good alternative to global temporary tables when the use of the data that is generated within the table is needed only for one operation (within one statement). Multiple common table expressions are included in a single SQL statement, and re-referenced if necessary. This configuration allows a programmer to create complex program loops and other processes within a single SQL statement. Common table expressions also enable recursive SQL which is used to process data with recursive relationships (such as bill of materials).

11.2.5 Using recursive SQL for retrieving ordered sets

Some non-relational databases maintain the concept of ordered sets, in which the order the data is inserted matters but the order is not maintained by some type of ever-increasing key. In an integrated database management system (IDMS), these types of sets are called NEXT sets.

Consider the code that is shown in Example 11-17. This example shows an ordered set with five member rows that belong to the same owner (999). The rows are displayed in the order in which they were inserted.

Example 11-17 Sample ordered set

```
-----+-----+-----+-----+
SELECT
  MEMBER_PRIM_KEY, OWNER_IN_SET
FROM TB_ORDERED_SET
WHERE OWNER_IN_SET = 999
;
-----+-----+-----+-----+
MEMBER_PRIM_KEY  OWNER_IN_SET
-----+-----+-----+-----+
              100          999
              102          999
              101          999
              104          999
              103          999
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
```

To select the rows in DB2 and return the rows in the necessary order, we use an ORDER BY clause on primary key column MEMBER_PRIM_KEY. The result is shown in Example 11-18.

Example 11-18 Sample converted ordered set with ORDER BY

```

-----+-----+-----+-----+
SELECT
  MEMBER_PRIM_KEY, OWNER_IN_SET
FROM TB_ORDERED_SET
WHERE OWNER_IN_SET = 999
ORDER BY MEMBER_PRIM_KEY
;
-----+-----+-----+-----+
MEMBER_PRIM_KEY  OWNER_IN_SET
-----+-----+-----+-----+
              100          999
              101          999
              102          999
              103          999
              104          999
DSNE610I NUMBER OF ROWS DISPLAYED IS 5

```

Notice that the rows are returned in the order of column MEMBER_PRIM_KEY, but the rows are returned in the order that the rows are returned in IDMS, so this result is incorrect.

When an ordered set is implemented in DB2, the design often is altered to address the ordering issue with NEXT sets. One alternative is to add two pointer columns to each row, with the keys of the prior row in the set and the next row in the set. Example 11-19 shows the same SELECT with the ORDER BY as shown in Example 11-18, but with the other columns that point to the next and prior rows for each row.

Example 11-19 Sample converted ordered set with ORDER BY and next and prior pointers

```

-----+-----+-----+-----+-----+-----+-----+-----+
SELECT
  MEMBER_PRIM_KEY, OWNER_IN_SET
  , MEMBER_ROW_NEXT, MEMBER_ROW_PRIOR
FROM TB_ORDERED_SET
WHERE OWNER_IN_SET = 999
ORDER BY MEMBER_PRIM_KEY
;
-----+-----+-----+-----+-----+-----+-----+-----+
MEMBER_PRIM_KEY  OWNER_IN_SET  MEMBER_ROW_NEXT  MEMBER_ROW_PRIOR
-----+-----+-----+-----+-----+-----+-----+-----+
              100          999             102             0
              101          999             104            102
              102          999             101            100
              103          999              0             104
              104          999             103            101
DSNE610I NUMBER OF ROWS DISPLAYED IS 5

```

When you use pointers to handle the order for NEXT sets, the first row in the set includes a zero in the PRIOR pointer, while the last row in the set includes a zero in the NEXT pointer. The use of ORDER BY still returns the rows in the wrong order.

Next, the results are sorted in the order of the PRIOR pointer column, starting at zero and working to the highest value. Example 11-20 shows the result.

Example 11-20 Sample converted ordered set with ORDER BY on prior pointer

```

-----+-----+-----+-----+-----+-----+-----+
SELECT
  MEMBER_PRIM_KEY, OWNER_IN_SET
, MEMBER_ROW_NEXT, MEMBER_ROW_PRIOR
FROM TB_ORDERED_SET
WHERE OWNER_IN_SET = 999
ORDER BY MEMBER_ROW_PRIOR
;
-----+-----+-----+-----+-----+-----+-----+
MEMBER_PRIM_KEY  OWNER_IN_SET  MEMBER_ROW_NEXT  MEMBER_ROW_PRIOR
-----+-----+-----+-----+-----+-----+-----+
                100          999          102              0
                102          999          101             100
                104          999          103             101
                101          999          104             102
                103          999           0              104
DSNE610I NUMBER OF ROWS DISPLAYED IS 5

```

The result still is in the wrong order when compared to our original query. This result demonstrates that recursive SQL is needed to produce the wanted result. The recursive SQL used to return the correct result is shown in Example 11-21.

Example 11-21 Recursive SQL to retrieve data in appropriate order for an ordered set

```

WITH ORDERED_SET
( LEVEL, OWNER_IN_SET, MEMBER_PRIM_KEY
, MEMBER_ROW_PRIOR, MEMBER_ROW_NEXT )
AS
(
  SELECT 1, ROOT.OWNER_IN_SET, ROOT.MEMBER_PRIM_KEY
        , ROOT.MEMBER_ROW_PRIOR, ROOT.MEMBER_ROW_NEXT
  FROM TB_ORDERED_SET ROOT
  WHERE ROOT.OWNER_IN_SET = 999
        AND ROOT.MEMBER_ROW_PRIOR = 0
  UNION ALL
  SELECT OWNER.LEVEL+1, MEMBER.OWNER_IN_SET, MEMBER.MEMBER_PRIM_KEY
        , MEMBER.MEMBER_ROW_PRIOR, MEMBER.MEMBER_ROW_NEXT
  FROM ORDERED_SET OWNER,
        TB_ORDERED_SET MEMBER
  WHERE MEMBER.OWNER_IN_SET = OWNER.OWNER_IN_SET
        AND OWNER.MEMBER_ROW_NEXT = MEMBER.MEMBER_PRIM_KEY
        AND OWNER.LEVEL < 100
)
SELECT
  OWNER_IN_SET, LEVEL, MEMBER_PRIM_KEY
, MEMBER_ROW_PRIOR, MEMBER_ROW_NEXT
FROM ORDERED_SET
ORDER BY
  OWNER_IN_SET, LEVEL, MEMBER_PRIM_KEY
, MEMBER_ROW_PRIOR, MEMBER_ROW_NEXT
;

```

The result is shown in Example 11-22. Notice that the results are now in the correct order. The NEXT column on each row is used to select another row at the next level of recursion.

Example 11-22 Results of recursive SQL that show the ordered set in correct order

OWNER_IN_SET	LEVEL	MEMBER_PRIM_KEY	MEMBER_ROW_PRIOR	MEMBER_ROW_NEXT
999	1	100	0	102
999	2	102	100	101
999	3	101	102	104
999	4	104	101	103
999	5	103	104	0

DSNE610I NUMBER OF ROWS DISPLAYED IS 5

You run similar recursive SQL to produce the set in reverse order by changing two lines in the SQL statement: the first WHERE clause selects the row in which the NEXT pointer is zero; the second WHERE clause selects the rows in which the PRIOR pointer owner is equal to the primary key of the member. The LEVEL column is introduced to provide numbering for each level of the hierarchy, and to provide a fail safe for pointers with incorrect values and that otherwise result in an infinite loop.

Recursive SQL: When you use recursive SQL, be careful to ensure that you have indexes to support the owner and member values for which you are searching. You use recursive SQL to validate a single set by specifying an owner value in the WHERE clause. If you use recursive SQL to build the set lists for all owners, DB2 builds a work file and performs multiple scans of the work file, which results in unacceptable performance.

11.2.6 Using uncommitted read

Many mature systems feature the concept of a dirty read, in which you do not wait for a process to commit or back out a change. Instead, you see data that is changed. Although not all applications tolerate viewing dirty data, those applications benefit from the use of an uncommitted read (UR) because it is less costly than locking data or waiting for released locks. We bind packages with UR, or we change individual SQL statements to run by using UR. The use of UR at the statement level is shown in Example 11-23.

Example 11-23 Sample use of uncommitted read in SELECT statement

```
SELECT MAX(BONUS), MIN(BONUS), AVG(BONUS)
  INTO :MAX, :MIN, :AVG
  FROM DSN81010.EMP
  WITH UR;
```

If there are concerns about using uncommitted read before a decision is made to update a value, optimistic locking is used. Optimistic locking is used to check a timestamp to determine whether the value changed since the last time you read it. By using optimistic locking, you read with UR but check the timestamp before the update occurs to ensure that you have the most current data. For more information about optimistic locking, see 11.1.7, “Using optimistic locking to improve concurrency” on page 179.

11.2.7 Monitoring and maintaining applicable organization and statistics

While the DB2 optimizer is a sophisticated engine, it is only as good as the information it is provided. The data must be kept well-organized and the optimizer needs current statistics about the data to correctly determine the access path.

Complete the following steps to ensure that the statistics in the DB2 catalog accurately reflect the organization and content of your data:

1. Start the REORG utility to reorganize the necessary tables, including the DB2 catalog table spaces and user table spaces. You start the DSNACCOX stored procedure to determine when reorganization is needed.
2. Start the RUNSTATS utility to capture statistics.
3. Rebind the plans or packages that contain affected queries. Specify the PLANMGMT bind option to save previous copies of the packages. You use the APCOMPARE bind option to detect access path changes for your static SQL statements. For dynamic SQL statements, DB2 uses the newly collected statistics at the next prepare.
4. Capture EXPLAIN information to validate access path changes.
5. If an access path regresses, use the REBIND command and specify the SWITCH option to revert to a previous access path. This action depends upon the PLANMGMT bind option that was specified when packages were first rebound.

Implement a strategy for routinely reorganizing your data and collecting statistics. Routine statistics collection is necessary for maintaining good performance, and often is required at times other than after reorganization. You also use RUNSTATS profiles and certain stored procedures to automate statistics maintenance.

To determine when to reorganize your data, use any of the following approaches:

- Monitor statistics for increases in the following values:

- I/O operations
- Get page operations
- Processor consumption

When performance degrades to an unacceptable level, analyze the statistics to develop your own rules for when to reorganize the data in your particular environment.

- Start the DSNACCOX stored procedure to get recommendations based on real-time statistics values.
- Query the catalog. Member DSNTESP of the SDSNSAMP data set contains sample useful queries for determining whether to reorganize.
- Use the REORG utility. The REORG utility embeds the function of catalog queries. If a query returns a certain result (you use the default or supply your own), REORG either reorganizes or does not reorganize. Optionally, REORG runs a report instead of reorganizing. The following REORG options start the catalog queries:
 - The OFFPOSLIMIT and INDREFLIMIT options of REORG TABLESPACE
 - The LEAFDISTLIMIT option of REORG INDEX
- Always reorganize your data after table definitions change. When ALTER TABLE statements are used to make any of the following changes to a table, the table space is placed in advisory REORG-pending (AREO*) status:
 - Add columns
 - Data type changes
 - Changed column lengths

When an existing column is changed, the table space is placed in AREO* status because the conversion to the new definition is not immediate. Reorganizing the table space causes the rows to reload with the data converted to the new definition. Until the table space is reorganized, the changes must be tracked and applied as the data is accessed, which might degrade performance.

For example, depending on the number of changes, you might encounter performance degradations for the following types of operations:

- Dynamic SQL queries
- Updates and deletes
- ALTER statements (especially those statements that run concurrently)
- Concurrent REORG and LOAD utilities
- Unloading a table that includes many changes before reorganization

Determining when to reorganize an index

To investigate whether to reorganize indexes, check for the following conditions in the SYSIBM.SYSINDEXSPACESTATS catalog table:

- ▶ $REORGPSEUDODELETES/TOTALENTRIES > 10\%$ in a non-data sharing environment, or $REORGPSEUDODELETES/TOTALENTRIES > 5\%$ in a data sharing environment.
- ▶ $REORGININSERTS/TOTALENTRIES > 25\%$
- ▶ $REORGDELETES/TOTALENTRIES > 25\%$
- ▶ $REORGAPPENDINSERT/TOTALENTRIES > 20\%$
- ▶ $EXTENTS > 254$
- ▶ High values in LEAFNEAR or LEAFFAR

Check for indexes in the following states:

- ▶ Advisory REORG-pending state (AREO*) as a result of an ALTER statement
- ▶ Advisory REBUILD-pending state (ARBDP) as a result an ALTER statement

Determining when to reorganize a table space

You use values in the SYSIBM.SYSTABLESPACESTATS catalog table to determine when a REORG is necessary. The following conditions are based on values in the SYSIBM.SYSTABLESPACESTATS table:

- ▶ $REORGUNCLUSTINS/TOTALROWS > 10\%$
- ▶ $(REORGNEARINDREF+REORGFARINDREF)/TOTALROWS > 5\%$ in a data sharing environment, or $(REORGNEARINDREF+REORGFARINDREF)/TOTALROWS > 10\%$ in non-data sharing environment
- ▶ $REORGININSERTS/TOTALROWS > 25\%$
- ▶ $REORGDELETES/TOTALROWS > 25\%$
- ▶ $EXTENTS > 254$
- ▶ $REORGDISORGLOB/TOTALROWS > 50\%$
- ▶ $SPACE > 2 * (DATASIZE / 1024)$
- ▶ $REORGMASDELETE > 0$
- ▶ $REORGCLUSTSENS > 0$

Consider running REORG if one of the following conditions are true:

- ▶ The table space is in the advisory REORG-pending state (AREO*) as a result of an ALTER TABLE statement.

- An index on a table in the table space is in the advisory REBUILD-pending state (ARBDP) as result an ALTER TABLE statement.

11.2.8 Using EXPLAIN to verify access path

By using the DB2 EXPLAIN facility, we verify that our access paths are efficient. If the paths are not efficient, we must consider changing the SQL statement or adding or changing an index.

A word of caution: We mentioned several times in this book that the first conversion effort might result in simple SQL statements that are solely retrieving data without much filtering or logic. In this instance, the EXPLAIN of the access path for this query likely works, but you must remember that performance is affected because of the quantity of SQL that is issued, rather than the quality. After efforts are made to better use DB2 and the SQL language, the use of EXPLAIN to tune queries becomes more valuable.

If you are a new user to DB2 for z/OS and are unfamiliar with DB2 access paths and how to interpret them, there are tools to help you become more familiar with these paths (in particular, the InfoSphere Optim Query Workload Tuner for DB2 for z/OS). For more information, see 11.3.1, “InfoSphere Optim Query Workload Tuner for DB2 for z/OS” on page 195.

11.3 IBM tools for application and SQL tuning

IBM offers various tools to help with the efforts of tuning applications and queries after the conversion to DB2. These tools help with tuning queries for overall performance management.

11.3.1 InfoSphere Optim Query Workload Tuner for DB2 for z/OS

This tool provides expert recommendations to DBAs to help improve the performance of a query workload by providing expert recommendations to maximize application performance, reduce specialized skill requirements, and reduce the total cost of ownership for DB2 for z/OS servers.

This tool performs the following tasks:

- Group your SQL statements into workloads to compare and track performance improvements.
- Identify and tune poorly performing query workloads.
- Manage and monitor the performance of SQL queries and workloads to proactively optimize physical database design and improve performance.
- Launch InfoSphere Optim Query Workload Tuner from IBM Data Studio V3.1 and gain immediate insight into performance enhancements during development.
- Provide tighter integration with other SQL tuning and monitoring solutions, including:
 - DB2 Query Monitor
 - DB2 Path Check for z/OS
 - DB2 SQL Performance Analyzer for z/OS
 - Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

- ▶ Work with Workload Access Plan Comparison, which helps compare cost and plans structure and Workload Access Plan Explorer that helps identify performance problems faster.
- ▶ Reduce downtime and associated costs that occur when responding to emergent problems.

11.3.2 IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

This tool is a single, comprehensive assessment tool that is used to evaluate the efficiency of and optimize performance from your DB2 for z/OS environment. The tool provides the following functions:

- ▶ Provides extended insight into application response time.
- ▶ Helps you monitor, analyze, and tune the performance of IBM DB2 for z/OS and IBM DB2 applications.
- ▶ Provides robust views of performance to help improve productivity.
- ▶ Features predefined rules so that you quickly and easily identify performance bottlenecks.
- ▶ Combines batch-reporting capabilities with real-time monitoring and historical tracking functions.
- ▶ Supports an enterprise-wide integrated systems management strategy that is activated by the IBM Tivoli OMEGAMON XE.
- ▶ Stores performance data and analysis tools in a performance warehouse.

11.3.3 IBM DB2 Path Checker for z/OS

This tool provides information about potential access path changes before they occur and about changes that occur after a rebind. The DB2 Path Checker helps you avoid costly performance issues that relate to access path changes before they occur. The tool provides the following functions:

- ▶ Identifies, analyzes, and documents potential access path changes in advance without initiating bind processing.
- ▶ Quickly determines whether a bind of a Database Request Module (DBRM) results in an access path change.
- ▶ Provides information about potential access path changes before they occur and after a rebind.
- ▶ Tests that are proposed or actual changes in plans and creates a report of changed and unchanged paths.
- ▶ Optionally provides cost estimates and DB2 catalog statistics for path changes.

11.3.4 IBM DB2 SQL Performance Analyzer for z/OS

This tool helps you to improve DB2 application design to achieve maximum productivity. The tool tests different “what if” scenarios quickly and economically to determine performance of various database design and production volumes. The tool includes the following features:

- ▶ Easy customization by using the new IBM Tools Customizer for z/OS feature.
- ▶ Access any report for any input source (DBRM, plan, package, and so on).
- ▶ View summary reports that with which you access a specific report directly from the summary report.

- ▶ Efficient navigation enables quicker access to the information you need most.
- ▶ Smooth workflow with a single point of entry into the tool.
- ▶ Access embedded SQL from source programs (PL/I, COBOL, C, Assembler, and so on).

11.3.5 IBM DB2 Query Monitor for z/OS

By using this tool, you efficiently customize and tune your SQL workload and DB2 objects. It also arms your staff with the ability to spot trouble before it causes a significant waste in DB2 resources. The tool provides for the following functions:

- ▶ View and configure monitoring across your enterprise from a single console.
- ▶ Identify users and locations that are issuing problem SQL statements. You have extensive choices for determining what monitoring information you gather and when to collect it.
- ▶ Set user-defined settings that trigger warnings when query activity crosses defined thresholds.
- ▶ Manage your data assets more cost effectively with GUI-based reporting, viewing, and configuration capabilities with which you access consolidated data and events for DB2 subsystems, whether within a single z/OS image or across multiple z/OS images.
- ▶ Research DB2 commands, display host variables, display SQL Communications Area, and find SQL error patterns.
- ▶ Shares an SQL collector with IBM Guardium® for System z. Provides autonomic functionality that enables DB2 Query Monitor to run user-configurable responses to a wide variety of events. Among the available responses are email, write-to-operator (WTO) and curative actions, such as console commands and batch job submissions.



Maximizing DB2 for z/OS features

This chapter describes several features of DB2 that must be considered before a conversion and some features that are taken advantage of after the conversion is complete. Whether the conversion was conducted by a tool or by a manual process, often the first conversion is fairly basic and not necessarily optimal or using the full potential of DB2.

We often find challenges with the performance of the new database or application. In this chapter, we describe the features of DB2 for z/OS that we take advantage of to further enhance our new database and achieve higher levels of performance with DB2 for z/OS. See Chapter 11, “Application and SQL tuning” on page 175 for more application and SQL features for performance.

The chapter includes the following topics:

- ▶ Using DB2 for z/OS features
- ▶ Using identity columns and sequence objects
- ▶ Referential integrity
- ▶ Triggers
- ▶ Temporal data
- ▶ In-memory tables
- ▶ Table check constraints
- ▶ DSNULI
- ▶ RELEASE(DEALLOCATE)
- ▶ INCLUDE columns
- ▶ Scrollable cursors
- ▶ DB2 built-in functions
- ▶ Non-padded indexes
- ▶ Clustering index
- ▶ Materialized query tables
- ▶ Compression
- ▶ Append processing
- ▶ Temporary tables
- ▶ Clone tables
- ▶ Large data support
- ▶ Storing LOB data
- ▶ Storing XML data

12.1 Using DB2 for z/OS features

With every release of DB2 for z/OS, there are new features to take advantage of with any application or database. However, often we do not go back and take advantage of these features with existing applications because of the complexities of changing existing applications without cause. With a migration or conversion effort, we change the application or database as we conduct the conversion. During this effort, we must consider various features of DB2 for z/OS to achieve the best possible performance for our application after the conversion.

We describe several features of DB2 for z/OS that help with various conversion efforts in achieving a higher level of performance during or after the conversion.

12.2 Using identity columns and sequence objects

Many of our mature applications need the ability to generate sequence numbers (such as get next available account number). The problem was the most common way to enable this ability by using a single control table to store these numbers, then use a mechanism to get the next available number while other users were prevented from obtaining the same number. This table often is needed by multiple transactions. This condition causes a single point of contention in our applications because only one transaction at a time retrieves the next value. Otherwise, the transaction locks the table to increment the number.

When an application is converted to DB2 for z/OS, this type of number generation must not be carried over. There are better options for number generation in DB2: identity columns and sequence objects.

12.2.1 Identity columns

An identity column is assigned to a table (part of the table or column definition). When a row is inserted into the table, the identity column is populated according to how it was defined (START WITH and INCREMENT BY values). While the performance of an identity column far outperforms the performance of a single control table, identity columns are limited in use and feature a few administrative challenges (such as the use of the generated value to populate RI-related tables).

12.2.2 Sequence objects

A sequence object is a user-defined object that generates a sequence of numeric values according to the specifications in which it was created. The objects provide an incremental counter that is generated by DB2 and is available to several users.

Sequence object values are used by applications for various reasons and feature the following benefits:

- ▶ No waits for incrementing values
- ▶ Stand-alone sequential number generating object (not tied to a table)
- ▶ Ascending or descending number generation
- ▶ Useful for application porting from other databases or platforms

- Can help to generate keys that are used to coordinate keys across multiple tables (RI or application-related)

Sequence objects are created by using the CREATE SEQUENCE statement and all attributes are defined by the user (or defaults). The values in the sequence object are any exact numeric data type. The starting value is defined with a START WITH value and advances with INCREMENT BY (ascending or descending). The values are cached and generated in the order of request.

Example 12-1 shows the syntax to create a sequence object.

Example 12-1 Sample CREATE SEQUENCE statement

```
CREATE SEQUENCE ACCOUNT_SEQ
AS INTEGER
START WITH 1
INCREMENT BY 10
CYCLE
CACHE 20
```

Usage of sequence objects is flexible. The syntax to support the value generation and usage is available, such as the ability to do NEXT VALUE FOR and PREVIOUS VALUE FOR. NEXTVAL FOR generates and returns the next value for the sequence object. PREVVVAL FOR generates and returns the previous value for the sequence object. These statements are used with the following statements:

- SELECT and SELECT INTO
- An INSERT statement within SELECT clause of fullselect
- UPDATE statement within the SET clause (Searched or positioned)
- SET host-variable
- VALUES or VALUES INTO
- CREATE PROCEDURE, FUNCTION, TRIGGER

12.3 Referential integrity

Although it seems obvious to describe referential integrity (RI) when it comes to a relational database such as DB2, if you are converting from a non-relational environment, there are some benefits to DB2 declarative referential integrity that are worth noting.

First, referential integrity (RI) is the ability to define parent-child relationships between tables. This definition ensures that the relationship is in place so that you do not incur orphaned data (a child without a parent). It also allows you to define what occurs in the event the parent is deleted: Is the child removed? Does the child prevent the removal of the parent?

Example 12-2 on page 203 shows a declarative RI definition in which the DEPT table is the parent and the EMP table is the child.

Example 12-2 Example of referential integrity

```
CREATE TABLE DSN81010.DEPT
  (DEPTNO   CHAR(3)           NOT NULL,
   DEPTNAME VARCHAR(36)       NOT NULL,
   MGRNO    CHAR(6)           ,
   ADMRDEPT CHAR(3)           NOT NULL,
   LOCATION CHAR(16)          ,
   PRIMARY KEY(DEPTNO))
IN DSN8D10A.DSN8S10D
CCSID EBCDIC;

--INDEX DEFINITIONS FOR DEPT
CREATE UNIQUE INDEX DSN81010.XDEPT1
  ON DSN81010.DEPT
    (DEPTNO   ASC)
  USING STOGROUP DSN8G10
    PRIQTY 12
    ERASE NO
  BUFFERPOOL BP0
  CLOSE NO;

CREATE TABLE DSN81010.EMP
  (EMPNO     CHAR(6)           NOT NULL,
   FIRSTNME  VARCHAR(12)       NOT NULL,
   MIDINIT   CHAR(1)           NOT NULL,
   LASTNAME  VARCHAR(15)       NOT NULL,
   WORKDEPT  CHAR(3)           ,
   PHONENO   CHAR(4) CONSTRAINT NUMBER CHECK
    (PHONENO >= '0000' AND PHONENO <= '9999'),
   HIREDATE  DATE NOT NULL WITH DEFAULT ,
   JOB       CHAR(8)           ,
   EDLEVEL   SMALLINT          ,
   SEX       CHAR(1)           ,
   BIRTHDATE DATE              ,
   SALARY    DECIMAL(9, 2)     ,
   BONUS     DECIMAL(9, 2)     ,
   COMM      DECIMAL(9, 2)     ,
   PRIMARY KEY(EMPNO),
   FOREIGN KEY RED (WORKDEPT) REFERENCES DSN81010.DEPT
    ON DELETE CASACADE)
EDITPROC DSN8EAE1
IN DSN8D10A.DSN8S10E
CCSID EBCDIC;
```

In Example 12-2, the delete rule is CASCADE, so if a DEPTNO is deleted from the parent table DEPT, all employee records in the EMP table with a WORKDEPT value equal to that DEPTNO value also are deleted. These types of delete processes that are supported in DB2 though RI help reduce the amount of application code and processing that was necessary in the past to support such operations.

RI does incur some cost so we must use it only where it is needed and defines a relationship between two tables. We do not want to use it for code checking. For more information about an alternative to code checking, see 12.7, “Table check constraints”.

RI also is useful for utilities and the optimizer to know what tables are related. However, in some instances we cannot use DB2 declared RI but still need the ability to define a relationship between two tables. DB2 also features informational referential integrity. The relationship between the tables is documented and recognized by DB2 utilities (such as QUIESCE) and the optimizer. Although the relationship is not enforced by DB2, but it is enforced with declarative RI.

You also must consider whether you want to define declarative RI for tables that was defined as optional sets in the pre-relational database. Optional sets, when implemented in a pre-relational database such as IDMS, allow for child rows that do not have a parent (or, in IDMS terms, member rows that do not have an owner). If you need to define such relationships in your converted DB2 database, you must implement one of the following design changes:

- ▶ Allow for NULL values in the foreign key on the child (member) table.
- ▶ Introduce an association table between the two tables and insert a row in the association table for each IDMS CONNECT process and then delete the row from the association table for each IDMS DISCONNECT process.

The option that you choose depends upon whether you want to allow for NULL values in your foreign key columns. If your applications are coded to handle NULL values, you choose to allow them and avoid creating associative tables.

12.4 Triggers

Triggers are a useful option for supporting data intensive application processes, such as the enforcement of business rules, auditing changes, minor replication, and data generation. Triggers are enforced at the table level and cannot be bypassed by normal application processing (except for a LOAD Utility). Triggers are a useful way to encapsulate business rules without the need to write or convert more code.

BEFORE triggers are used to validate, set, or modify input values, and prevent invalid updates from occurring. AFTER triggers are evaluated after the triggering event and they take action outside of the database by calling a stored procedure or user-defined function. A BEFORE trigger is used to condition data before the data moves into the final database.

Important: Triggers are used as an alternative for converting large amounts of redundant code by simplifying the process to put the work into the database. Be careful not to use triggers in the wrong place (such as edits) because there is some overhead associated with triggers. If triggers are used inappropriately, they are expensive.

Triggers also are used to support referential integrity-type functions when declared RI cannot be used. This situation occurs when non-relational environments are converted because it is possible not all data fits well into a normalized design that supports true primary key or foreign key relationships, even though a 1-to-1 relationship exists.

A common use of triggers for data that is converted from a non-relational database is in cases in which the data was in multiple databases and was converted to a single DB2 database. For example, data that is found in 10 different databases is delineated by the last digit of the account number. Some customers do not want to maintain 10 separate DB2 databases. Therefore, at conversion time, the customers combine the data from the 10 databases into a single DB2 database. All of the tables in the database are partitioned by the last digit of the

account number. A BEFORE INSERT trigger is used to generate a column from the last digit of the account number and the generated column is used as the partitioning column.

12.5 Temporal data

DB2 for z/OS supports temporal data in its table definitions. Many mature applications must support temporal data because the applications must see data changes over time and provide for future validity periods of data. As of DB2 10, this ability is supported within DB2 for z/OS.

DB2 for z/OS supports two types of periods: the application period (BUSINESS_TIME) and the system period (SYSTEM_TIME).

An application period consists of a pair of columns with application-maintained values that indicate the time when a row is valid. A table with only an application period is called an *application-period temporal table*.

The definition an application-period temporal table is shown in Example 12-3.

Example 12-3 Definition of temporal structure by using business time

```
CREATE TABLE policy_info
(policy_id CHAR(4) NOT NULL,
coverage INT NOT NULL,
bus_start DATE NOT NULL,
bus_end DATE NOT NULL,
PERIOD BUSINESS_TIME(bus_start, bus_end));
```

The system period consists of a pair of columns with system-maintained values that indicate the time when a row is valid. The system period is meaningful because you define system-period data versioning on a table that includes this period. System-period data versioning specifies that old rows are archived into another table. The table that contains the current active rows of a table is called the *system-period temporal table*. The table that contains the archived rows is called the *history table*. If you have the correct authorization, you delete the rows from the history table when those rows are no longer needed.

When you define a base table to use system-period data versioning, or when you define system-period data versioning on an existing table, you must create a history table, specify a name for the history table, and create a table space to hold that table. You define system-period data versioning by issuing the ALTER TABLE ADD VERSIONING statement with the USE HISTORY TABLE clause. Example 12-4 lists the definitions that are needed to create a system-period temporal data structure.

Example 12-4 Definition of temporal structure by using system time

```
CREATE TABLE POLICY_INFO
(POLICY_ID CHAR(10) NOT NULL,
COVERAGE INT NOT NULL,
SYS_START TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
SYS_END TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
CREATE_ID TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
PERIOD SYSTEM_TIME(SYS_START,SYS_END));

CREATE TABLE HIST_POLICY_INFO
(POLICY_ID CHAR(10) NOT NULL,
```

```

COVERAGE INT NOT NULL,
SYS_START TIMESTAMP(12) NOT NULL,
SYS_END TIMESTAMP(12) NOT NULL,
CREATE_ID TIMESTAMP(12));

ALTER TABLE POLICY_INFO
ADD VERSIONING USE HISTORY TABLE HIST_POLICY_INFO;

```

A *bitemporal table* is a table that is a system-period temporal table and an application-period temporal table. You use a bitemporal table to retain application period information and system-based historical information. Therefore, there is flexibility in how you query data based on periods of time.

Temporal tables are useful when you convert data from a pre-relational database to DB2 and you do not have the time or the processes in place to archive the data before the conversion. You convert all of the data and archive data after the conversion by using the temporal data feature.

12.6 In-memory tables

During conversion from non-relational environments, a normalized model of the data is produced. These models often produce a large amount of code and reference tables. Ideally, the number of code or references tables must be kept to a minimum and if the codes are placed in memory (possibly in CICS or WebSphere), performance is greatly improved. If the codes must remain in DB2 tables and are checked by SQL, we must make this process as efficient as possible.

Before DB2 10, to have fast code access in the buffer pool, a buffer pool large enough to hold the pages of the table space and the indexes was created. The VPSEQT threshold was set to 0 to turn off prefetch overhead and VPSTEAL was set to FIFO to turn off latch management of the queue. The challenge is to stage the data initially, but after the data is in the buffer pool, there is no I/Os needed to conduct code checking. In DB2 10, you use the new PGSTEAL = NONE option on the buffer pool to achieve the same result:

```
-ALTER BUFFERPOOL (BP1) PGSTEAL (NONE)
```

Pre-staging the data in this instance is easier because DB2 reads all of the pages after the objects are opened. You also pre-open the objects by using the -ACCESS DATABASE command:

```
-ACCESS DATABASE(DSN10001) SPACENAM(DSN10002) PART(1,3) MODE(OPEN)
```

12.7 Table check constraints

After a conversion, you might have with tables that contain codes that are needed by the application. You must evaluate these tables carefully because if the tables contain static codes, they might be better placed in a table check constraint. Checking or verifying that code in a table is expensive (even if the tables are in memory). Verifying that values in a table check constraint is more efficient. Check constraints also are used to verify value ranges and to compare two column values together.

Example 12-5 shows a check constraint to validate the phone number (PHONENO) in the EMP table.

Example 12-5 Example of check constraint

```
CREATE TABLE DSN81010.EMP
  (EMPNO      CHAR(6)          NOT NULL,
   FIRSTNME   VARCHAR(12)     NOT NULL,
   MIDINIT    CHAR(1)         NOT NULL,
   LASTNAME   VARCHAR(15)     NOT NULL,
   WORKDEPT   CHAR(3)         ,
   PHONENO    CHAR(4) CONSTRAINT NUMBER CHECK
     (PHONENO >= '0000' AND PHONENO <= '9999'),
   HIREDATE   DATE NOT NULL WITH DEFAULT,
   JOB        CHAR(8)         ,
   EDLEVEL    SMALLINT        ,
   SEX        CHAR(1)         ,
   BIRTHDATE  DATE            ,
   SALARY     DECIMAL(9, 2)    ,
   BONUS      DECIMAL(9, 2)    ,
   COMM       DECIMAL(9, 2)    ,
   PRIMARY KEY(EMPNO),
   FOREIGN KEY RED (WORKDEPT) REFERENCES DSN81010.DEPT
     ON DELETE SET NULL)
EDITPROC DSN8EAE1
IN DSN8D10A.DSN8S10E
CCSID EBCDIC;
```

You also add a check constraint with an ALTER statement. The following example adds a check constraint to ensure that the WORKDEPT value was 10:

```
ALTER TABLE DSN81010.EMP ADD CHECK (WORK DEPT = 10)
```

If the table was populated, it is set to CHKP status and the CHECK DATA utility must be run.

12.8 DSNULI

The use of the DB2 attach facility Universal Language Interface module (DSNULI), which is available in DB2 10, supports reusable Application Programming Interfaces (APIs) that are used for sharing code between converted mature applications. The DSNULI attach allows the use of single-purpose programs in different environments without the need to compile and link different programs and applications that are equivalent in DB2.

The DSNULI is used to call an application that has a DB2 connection. DSNULI also is used to run applications under the following attachment facilities:

- ▶ CAF (call attachment facility)
- ▶ CICS (Customer Information Control System)
- ▶ TSO (Time Sharing Option)
- ▶ RRS (Resource Recovery Services)

Use this option when you need to run programs by using an attachment facility and when flexibility is a higher priority than optimal application performance (a benefit in that only one user program load module and library is required for multiple execution environments).

12.9 RELEASE(DEALLOCATE)

Many mature applications feature efficient sequential processing that we also must provide for in the DB2 for z/OS environment. We design our tables for this feature (such as clustering) and we also take advantage of efficiencies in DB2, such as sequential detection and index lookaside. The RELEASE(DEALLOCATE) package bind parameter helps us with this effort because it retains the caches that are used for those processes across commits.

Before DB2 10, RELEASE(DEALLOCATE) was not accepted for remote DDF threads; instead, threads automatically were converted to RELEASE(COMMIT). In DB2 10, RELEASE(DEALLOCATE) is used with remote DDF threads (High Performance DBATs) by using the MODIFY command. The following example of the MODIFY command accepts the bind option on the package (this case, if the bind option was DEALLOCATE, the option is used):

```
-MODIFY DDF PKGREL (BNDOPT)
```

12.10 INCLUDE columns

INCLUDE columns are useful for applications that are converted to DB2. Now that the data is in a relational database, objects such as unique constraints and primary or foreign keys are implemented and enforced. To support these constraints, we use indexes. With a unique constraint, the unique index is on the columns that are needed to enforce the constraint. However, when it comes to coding the most efficient SQL, more columns might be needed to achieve index-only access.

In DB2 10, a column is added to an index that does not violate the unique constraint, but does allow for index-only access on all columns. With this configuration, another index is not needed for the query. The following example of an ALTER statement allows an index to include another column. Assume that the index XEMP1 was constructed initially to support a unique constraint on COL1 and COL2 and then COL3 is needed to allow for index only access but not participate in the unique constraint:

```
ALTER INDEX DSN81010.XEMP1 ADD INCLUDE(COL3)
```

An INCLUDE column also is part of the index during the initial creation of the index.

12.11 Scrollable cursors

Mature applications tend to be efficient when it comes to scrolling and paging through data. When applications are converted to DB2 for z/OS, we want those types of processes to be as efficient as possible and we have a few options that are considered in lieu of older scrolling techniques. DB2 scrollable cursors are used in both static and dynamic z/OS compiled programs, compiled stored procedures (including SQL stored procedures), ODBC applications, and client programs that use IBM DB2 Connect™. The biggest drawback to the use of scrollable cursors is that the cursors are used with pseudo-conversational CICS applications.

There are two types of scrollable cursors in DB2: static and dynamic. Dynamic scrollable cursors use a declared temporary table which was filled with a fixed number of rows. The use of these cursors allows DB2 to scroll back and forth because the database already had the result set in a table. The static scrollable cursors have some challenges, so often we prefer to look at the dynamic scrollable cursors.

Dynamic scrollable cursors allow us to implement these types of cursors in our application with better performance and more usability. Dynamic scrollable cursors allow for base table access instead of using a declared temporary table. Dynamic cursors also do not materialize at any time. Each FETCH shows the most current activity and is sensitive to all INSERTs, UPDATEs, and DELETEs with no delete holes and missing updates.

The cursors default to single row fetch, but do support multi-row fetch and positioned updates or deletes with multi-row fetch. Order is always maintained with these cursors.

The following syntax example defines a dynamic scrollable cursor:

```
DECLARE CUR1 SENSITIVE DYNAMIC CURSOR
FOR SELECT EMPNO, LASTNAME FROM DSN81010.EMP
```

Some SQL statements require that result sets be placed in work files for joins. These results sets are considered read only, and cannot support dynamic scrolling. The dynamic scrollable cursors take advantage of backwards index scans and backwards sequential detection and support index scan access and table scan access paths.

Remember that dynamic scrollable cursors are not always appropriate for every situation. If you want to scroll back and populate a screen the way it was, these cursors do not work because the data changes. Also, parallelism is not supported by this type of cursor. SELECT list scalar functions and arithmetic expressions are reevaluated for every FETCH, but column functions are calculated once at cursor open. These cursors might not be useful because the result table changes. If a work file is needed, the cursor cannot be SENSITIVE DYNAMIC. If changes are made to a table that are referenced in subqueries, those changes are not seen. Built-in functions or user-defined functions (UDFs) in the WHERE clause have misleading results because the results of functions vary from one FETCH to the next in the same row.

In summary, scrollable cursors are useful. There are multiple implementations, so you must review all of the options to determine the method that is best for your application.

12.12 DB2 built-in functions

Mature applications might be managing more work than is necessary. By using a few built-in functions in DB2, you start to take advantage of more of the power of the DB2 engine. Built-in functions are DB2 operations that operate on a column or a table and return some value or values that are based on the operation defined by the function. For example, you return the maximum or minimum value of a column in a table. Some of the more common built-in functions are SUM, AVG, MIN, MAX, and YEAR, though there are many other factors. User-defined functions also are used if a more robust function is needed than the finite set of built-in functions that are provided by DB2.

Example 12-6 shows the use of the MAX built-in function to obtain the maximum salary for each department in our employee table.

Example 12-6 Use of built-in function MAX

```
SELECT DEPT, MAX(SALARY)
FROM DSN81010.EMP
GROUP BY DEPT
```

The use of a built-in function allows efficient and easy processing in DB2. Data is not passed back to the application unnecessarily to conduct the operation or calculation in the application source code.

12.13 Non-padded indexes

If some of the data from the conversion results in VARCHAR fields, those fields are part of an index. For best performance and use of this index, use CREATE or ALTER so that the index is NOT PADDED. An index on a VARCHAR column pads to the full length of the column, which increases the index more than necessary and does not allow for index-only access for queries by using this index. Creating or altering the index for NOT PADDED allows for a smaller size and a better access path, as shown in the following code:

```
ALTER INDEX DSN81010.XEMP5 NOT PADDED
```

There might be some CPU overhead to manage a non-padded index; however, the overhead often is minimal and acceptable. This overhead depends on the size and number of VARCHAR columns in the index.

12.14 Clustering index

When data is converted from a non-relational format, often it comes from an environment in which there was no construct or need to support clustering. In DB2 for z/OS, data clustering is important for performance, especially for sequential processing items such as batch jobs.

The clustering index is designed to keep table rows in a specified sequence to minimize page access for a set of rows. The clustering index is defined during the creation of the index by using the CLUSTER keyword. Example 12-7 shows the syntax to create the XEMP2 index as the clustering index.

Example 12-7 Use of CLUSTER option on CREATE INDEX

```
CREATE INDEX DSN81010.XEMP2
  ON DSN81010.EMP
  (EMPNO ASC)
  USING STOGROUP DSN8G100
  PRIQTY 36
  ERASE NO
  CLUSTER
```

After the data is defined, the data is clustered according to this index. When data is inserted into the table, the data is in the clustering sequence, provided there is ample free space. Only one clustering index per table and the index must be defined to support the sequential processing of your data (such as batch jobs or range retrieval). This index must be explicitly defined, or DB2 implicitly clusters by the index of oldest definition, which might not be what you want.

Also, if two tables are constantly joined, it might be helpful to have the tables that are clustered in the same sequence (if possible) to avoid random I/O during the join.

Do not be concerned if a clustering index was not defined when the conversion occurred. DB2 implicitly clusters by the first index that was created (the index of oldest definition). When you determine a better candidate, you perform an ALTER to define a new index by which to cluster, as shown in the following code:

```
ALTER INDEX DSN81010.XEMP1 NOT CLUSTER
ALTER INDEX DSN81010.XEMP2 CLUSTER
```

After the ALTERs are issued DB2, immediately begins to insert according to the new clustering index. A REORG must be conducted immediately after this change to incorporate the data into the new clustering sequence.

12.14.1 Natural keys versus surrogate keys

Conversions often produce a surrogate key construct instead of a natural key. A surrogate key often is used when a natural key is unknown. A surrogate key is a random number without any meaning or true relation to the data. This lack of meaning or relation is a problem if this key becomes the primary key of the data and is chosen as the clustering index. In these instances, the data becomes random and this leads to many performance problems if the correct clustering for joins and sequential processing is not maintained. If possible, we want to use a natural business key that relates to our data and is used to correctly store and access our data in the manner in which the process expects it.

If the conversion results in your tables that contain surrogate keys, and you have child tables with foreign keys that refer to the surrogate keys of the parent, you might want to consider clustering the child tables on the foreign keys. This clustering scheme allows access to the parent and child data in roughly the same page order. This configuration works well for a two-level parent-child relationship, but breaks down at the third level because the foreign keys of the third level table relate to the primary key of the middle level table, yet the middle level table is clustered on its foreign key.

12.15 Materialized query tables

Some mature applications or databases might have a mechanism for keeping aggregated or summarized data for use by the application. Although the relational model does not incorporate the idea of storing aggregated or summarized data, the model often is needed to help applications quickly retrieve data that is pre-processed for them.

Materialized query tables (MQTs) simplify query processing, greatly improve the performance of dynamic SQL queries, and are effective in data warehousing applications in which you use MQTs to avoid costly aggregations and joins against large fact tables.

The MQTs contain information that is derived and summarized from other tables. Materialized query tables pre-calculate and store the results of queries with expensive join and aggregation operations. By providing this summary information, materialized query tables simplify query processing and greatly improve the performance of dynamic SQL queries.

For dynamic queries, automatic query rewrite is the process DB2 uses to access data in a materialized query table. If you enable automatic query rewrite, DB2 determines whether it is possible to resolve a dynamic query or part of the query by using a materialized query table.

Suppose that you have a large table named TRANS that contains one row for each transaction that a certain company processes. You want to tally the total amount of transactions by a specific time. Although the table contains many columns, you are most interested in the following columns:

- ▶ YEAR, MONTH, DAY, which contain the date of a transaction
- ▶ AMOUNT, which contains the amount of the transaction

To total the amount of all of the transactions by year between 2001- 2006, you run the query that is shown in Example 12-8 on page 212.

Example 12-8 Query to return summarized data when no MQT exists

```
SELECT YEAR, SUM(AMOUNT)
  FROM TRANS
  WHERE YEAR >= '2001' AND YEAR <= '2006'
 GROUP BY YEAR
 ORDER BY YEAR;
```

This query might be expensive to run, particularly if the TRANS table is a large table with millions of rows and many columns. Now suppose that you define a materialized query table named STRANS by using the CREATE TABLE statement that is shown in Example 12-9.

Example 12-9 DDL to create a materialized query table

```
CREATE TABLE STRANS AS
  (SELECT YEAR AS SYEAR,
   MONTH AS SMONTH,
   DAY AS SDAY,
   SUM(AMOUNT) AS SSUM
   FROM TRANS
   GROUP BY YEAR, MONTH, DAY)
 DATA INITIALLY DEFERRED REFRESH DEFERRED;
```

After you populate the MQT named STRANS with a REFRESH TABLE statement, the table contains one row for each day of each month and year in the TRANS table.

By using the automatic query rewrite process, DB2 rewrites the original query into a new query. As shown in Example 12-10, the new query uses the materialized query table STRANS instead of the original base table TRANS

Example 12-10 Sample rewritten query to access materialized query table

```
SELECT SYEAR, SUM(SSUM)
  FROM STRANS
  WHERE SYEAR >= '2001' AND SYEAR <= '2006'
 GROUP BY SYEAR
 ORDER BY SYEAR
```

If you maintain data currency in the materialized query table STRANS, the rewritten query provides the same results as the original query. The rewritten query offers better response time and requires less CPU time.

Also, if you are not using dynamic queries that take advantage of query rewrite, you use an MQT with a static SQL statement. You select from the MQT and easily manage and populate these tables by DB2 by using the REFRESH statement.

If you have reporting requirements in your application, and those reports are run against data for a specific time and do not need to show absolute current data, you might rewrite some of those converted programs to use MQTs for better performance.

12.16 Compression

A conversion from a mature environment, especially a highly optimized flat file environment such as BDAM or QSAM, might result in a large amount of data that is stored in relational structures. You must consider DB2 compression for your data when you are estimating storage needs for the conversion.

When you compress data, bit strings that occur frequently are replaced by shorter strings. Information about the mapping of bit strings to their replacements is stored in a compression dictionary. Computer processing is required to compress data when it is stored and to extract the data when it is retrieved from storage. By using DB2 compression, in many instances you significantly reduce the amount of disk space that is needed to store data, but the compression ratio that you achieve depends on the characteristics of your data.

With compressed data, you might see some of the following performance benefits, depending on the SQL workload and the amount of compression:

- ▶ Higher buffer pool hit ratios
- ▶ Fewer I/Os
- ▶ Fewer getpage operations

Compression is defined on the CREATE of the TABLESPACE or through the ALTER by using the COMPRESS YES option. Compression takes place at LOAD or REORG time, or dynamically when rows are inserted.

Indexes also are compressed; however, this compression must be evaluated carefully as it is a different type of compression and provides savings only on disk. Index compression incurs some CPU overhead for indexes not resident in the buffer pool with a high I/O rate and with infrequently reused pages.

12.17 Append processing

Another challenge that often faces a database under conversion is the ability to insert data quickly into a DB2 table. DB2 inserts are expensive and take time. There are some features in DB2 that allow us to speed up our insert times by using APPEND processing.

12.17.1 Member Cluster

This table option specifies that data that is inserted by an INSERT operation is not clustered by the implicit clustering index (the first index) or the explicit clustering index. DB2 places the data in the implicitly created table space that is based on available space. This option allows for inserts to occur quickly because the inserts occur where space is available. To best use this option, the table space and clustering index must be defined with PCTFREE 0 and FREEPAGE 0.

With MEMBER CLUSTER, each space map covers 199 data pages. Because there are more space map pages and some might be partially used, table spaces that are defined with MEMBER CLUSTER use more disk. In data sharing, MEMBER CLUSTER reduce the overhead of reacquiring page P-locks.

The MEMBER CLUSTER option is ALTERed with the deferred alter capability and it is applicable.

This option and the next option, APPEND, have the same caveat that the inserts potentially are not clustered and might need more space. If any process (such as sequential batch) relies on them as clustered for performance, frequent reorganizations might be necessary.

12.17.2 APPEND option of CREATE and ALTER table

The APPEND option on the table inserts or loads data into the table without clustering. With this option, you do not specify the PCTFREE 0 and FREEPAGE 0 as with member cluster. Rather than attempting to insert rows in cluster-preserving order, the rows are appended at the end of the table or appropriate partition. For range-partitioned table spaces, the appropriate partition is dictated by the value in the partitioning column of the row. For partition-by-growth table spaces, the appropriate partition is any partition with space at the end. This option is specified on the CREATE TABLE statement or ALTERed, as shown in the following code sample:

```
ALTER TABLE DSN81010.EMP APPEND YES
```

Either of these options helps with insert performance. One difference between the options is that member cluster allows for the reuse of deleted space, whereas the APPEND option does not. Also, member cluster is needed to help with clustered inserts in a data sharing environment so that the space map is managed more efficiently to avoid any p-lock overhead.

12.18 Temporary tables

With the new large systems and more complex transactions, temporary tables are useful for the quick process in a unit of work, and for the large scratch pad for a process. There are instances in which when a table does not need to exist for a long. For example, a table often acts as a staging area for data for use by a program. Temporary tables are used to meet this need. DB2 has two types of temporary tables: created or declared.

12.18.1 Created temporary tables

Created temporary tables (CTTs) help improve performance in many ways. When a repetitive SQL statement is used to return a result set and produces the same result each time, a CTT might provide benefit. A subquery that is run more than once is a good example. It might be better to issue the subquery once, store the result set rows in the CTT, and use the CTT in subsequent subqueries.

The biggest advantage of CTTs is that logging is not done because recovery is not possible. However, indexing also is not done, so a table space scan always is used as the access path. No modifications to the data in the CTT are allowed, only INSERTs. The CTT exists during the unit of work, and are automatically deleted when a commit is issued unless the table is used in a cursor definition by using CURSOR WITH HOLD.

CTTs also are useful for stored procedures. CTTs are used as a holding area for non-relational or non DB2 data such as data extracted from VSAM files. The data is held during the unit of work and is referenced in SQL statements. This feature is valuable when a left or full outer join is required by using one DB2 table and one non DB2 table (such as VSAM data). An INSERT statement loads the temporary table with the VSAM data, and a SELECT statement such as the statement in Example 12-11 on page 215 is used to conduct the outer join.

Example 12-11 Join of base table and temporary table

```
EXEC SQL
SELECT * FROM T1 LEFT JOIN GBLTEMP
ON EMPNO = DEPTEMP
END-EXEC.
```

This function is conducted in a stored procedure and other processes that need the result and run it. One of the benefits of this function is that the joins are completed in DB2, instead of in the program.

Another benefit of CTTs is the ability to use them when a materialized set is present for a result set, a view, or a table expression, and the materialized set must be used more than once.

The only access path available against a CTT is a table space scan, the size of the scan must be considered when designing CTTs. When a CTT is used in a join, the access path often is a merge scan join that might require a sort on the CTT.

The CTT is created in the same manner as a normal table (through DDL), except that the CTT is not created in a table space. These tables are not created with default values, and they do not have unique, referential, or check constraints defined for them. Example 12-12 shows the DDL to create a created temporary table that holds rows that contain an amount and a date.

Example 12-12 DDL to create a global temporary table

```
CREATE GLOBAL TEMPORARY TABLE EMP
(SALARY DECIMAL(5,2) NOT NULL,
HIRE_DATE DATE NOT NULL)
```

An empty instance of the table is created when the first implicit or explicit reference is made in a SQL statement. An insert is the first statement that is issued to populate the table. The temporary table exists only until the originating application commits, performs a rollback, or terminates (unless the table is used in a cursor by using WITH HOLD option).

12.18.2 Declared temporary tables

Declared temporary tables (DTTs) are another option for storing data temporarily. DTTs include the following characteristics:

- ▶ Not recorded in the DB2 catalog
- ▶ Cannot be shared
- ▶ Declared in program code
- ▶ Not predefined
- ▶ Are updated
- ▶ Indexes are created on them

DTTs declare a temporary table for use in a program. The DECLARE GLOBAL TEMPORARY TABLE statement defines a temporary table for the current session, not only for a unit of work. The table description does not show in the DB2 catalog. A DTT is not persistent and cannot be shared (unlike a CTT).

This statement is embedded in an application program or issued by using dynamic SQL statements. The statement is an executable statement that is dynamically prepared. Each session that defines a declared global temporary table of the same name has its own unique

instantiation of the temporary table. When the session terminates, the temporary table is dropped. With DTTs, some of the locking, DB2 catalog updates, and DB2 restart forward and backward log recoveries that are associated with persistent tables are avoided.

DTTs are useful for applications that must extract data from various sources and use them in SQL joins, or for data that must be used repetitively or kept separate from other online transaction processing (OLTP) processes. DTTs also are used as staging areas for data that comes from various sources so that the data is manipulated before it is stored permanently in regular tables.

The DECLARE GLOBAL TEMPORARY TABLE statement defines a temporary table for the current application process and instantiates an empty instance of the table for the process. The statement is an executable statement that is embedded in an application program or issued interactively and is dynamically prepared.

Example 12-13 shows the SQL syntax to declare the temporary table and to run an INSERT into the temporary table by using rows that are selected from another table.

Example 12-13 DECLARE of global temporary table and INSERT to load table

```
EXEC SQL
DECLARE GLOBAL TEMPORARY TABLE SESSION.EMPT
    LIKE DSN81010.EMP
END-EXEC
EXEC SQL
INSERT INTO SESSION.EMPT
    SELECT * FROM DSN81010.EMP
END-EXEC.
```

These DTTs are used as a way to temporarily hold or sort data within a program. DTTs are useful for relational online analytical processing (ROLAP) and multidimensional online analytical processing (MOLAP) queries for warehouse tools and as a staging area for IMS or VSAM data so it is SQL- and ODBC-accessible.

Only Undo records are logged and the full range of Data Manipulation Language (DML) statements (INSERT, UPDATE, SELECT, DELETE) are run on the records. DTTs are supported by rollback to savepoint or last commit point. The table exists until thread termination, or if thread reuse is used, the table exists until it is implicitly dropped, which is not desirable, depending on the application.

No locks are taken (PAGE, ROW, or TABLE) on DTTs; however, there are locks that are taken on the table space and DBD in share mode. Locks also do not require that the cursor is defined WITH HOLD to hold rows across commits.

Static SQL referencing a DTT is incrementally bound at run time. The number of incremental binds are seen on OMEGAMON Performance Expert reports.

The cost that is associated with a DTT is equivalent to the cost of running a dynamic SQL statement. High-volume transaction applications need careful evaluation when you are planning to use DTTs.

12.19 Clone tables

Data availability is a challenge in any environment. The ability to keep data current and available is one of the reasons we chose to convert to a relational DB2 environment because this DBMS has many features to help with continuous availability. Clone tables help with managing tight batch windows in which applications want to conduct updates.

One useful feature in DB2 is the ability to use clone tables. A clone table is a replica of the base table. Each table contains different data so it is easier to switch between the two tables.

A clone table is created by specifying the `ADD CLONE` keywords on an `ALTER TABLE` statement. The clone table receives a distinct name from the base table on which it is defined, but it is defined within the same schema and has the same structure as the base table. The clone table exists within the same universal table space as the base table. The clone table inherits the column names, data types, null attributes, check constraints, indexes, and before triggers of the base table.

After a clone table is created, the table is manipulated independently of the base table, and you use an `SQL EXCHANGE` statement to exchange data between the base table and the clone table. You have query activity that is running against one version of the table, while updates occur on the other version of the table.

12.20 Large data support

Almost every mature conversion requires the ability to store more data, make more data readily available (not archived), and efficiently process large amounts of data. DB2 for z/OS stores vast amounts of data. There are many features in DB2 that allow us to efficiently store and manage terabytes of data.

12.20.1 Partitioned table spaces

DB2 stores up to 128 terabytes of data in a single table, with the correct settings. DB2 stores data into individual VSAM linear data sets (LDSs) called *partitions* and has up to 4096 partitions. The partitions are broken up by a key range (range-partitioned) or by simple physical boundaries (partition-by-growth). Either partition type allows data to be stored in DB2 and indexed for the application process. This data also is compressed. For more information about DB2 compression, see 12.16, “Compression” on page 213.

The partitioning of the data also is helpful for availability. In some instances, DB2 utilities process against several partitions in parallel.

The structure of the partitioned table space also is flexible. After the table space is created, it is easy to add partitions with an `ALTER` statement. We also rotate the partitions if necessary, to roll off old data and provide room for insertion of new data.

12.21 Storing LOB data

DB2 has a unique capability over other non-relational databases because it stores normalized relational data and large object data, such as LOBs. LOBs support data types of binary large object (BLOB), character large object (CLOB), and double byte character large object (DBCLOB). In Example 12-14, a picture (BLOB) and a text document (CLOB) are stored.

Example 12-14 DDL for a table with LOB columns

```
CREATE TABLE DSN81010.EMP_PHOTO_RESUME
  (EMPNO          CHAR( 06 ) NOT NULL,
   EMP_ROWID      ROWID NOT NULL GENERATED ALWAYS,
   PSEG_PHOTO     BLOB( 500K ),
   RESUME         CLOB(  5K ),
   PRIMARY KEY ( EMPNO ) )
IN DSN8D10L.DSN8S10B
CCSID EBCDIC;
```

The LOB data is stored in an auxiliary table that is defined by DB2, but you access the LOB data by using SQL against the LOB column on the base table.

12.22 Storing XML data

Now that the mature data is converted into a relational table, this type of data is not the only data that we are able to store. DB2 stores XML data in a table. We use various DB2 built-in functions to construct XML documents and store them. We also retrieve XML data from DB2 and return it to the application as a DB2 table by using the XMLTABLE function.

In Example 12-15, a DB2 table is created that includes a column defined with an XML data type.

Example 12-15 DDL to create a table with an XML column

```
CREATE TABLE DSN81010.PRODUCT
  ( PID          VARCHAR(10) NOT NULL PRIMARY KEY
  ,NAME          VARCHAR(128)
  ,PRICE         DECIMAL(30, 2)
  ,PROMOPRICE    DECIMAL(30, 2)
  ,PROMOSTART    DATE
  ,PROMOEND      DATE
  ,DESCRIPTION   XML )
IN DSN8D10X.DSN8S10X
CCSID EBCDIC;
```

For more information about setting up and storing LOB or XML data, see *IBM DB2 10 for z/OS Administration Guide*, SC19-2968.

Part 5

Appendixes



IBM Information Management Services Migration Practice

One of the many benefits of a migration is that an organization is positioned to take advantage of newer or more powerful technology. Many migration projects are technologically simple, while others are more robust in scope.

Information Management Services Migration Practice of IBM developed a methodology for understanding the changes that must be made during a migration and minimizing the risk and cost of such projects. This methodology is described in this appendix.

This appendix contains the following topics:

- ▶ Strategy for a successful migration
- ▶ Customer partnering is a key element to success
- ▶ Migration strategy
- ▶ IBM Information Management Software Services

Strategy for a successful migration

A successful migration strategy includes the following key components:

- ▶ Well-understood process

The Migration Practice of IBM conducted many successful migrations, and we understand the necessary steps to ensure a successful project.

- ▶ Utilization of *best practices* techniques

Our methodology takes advantage of the experience of running projects in many different environments. We observed the techniques and practices that lead to satisfied customers, for users and IT professionals. Some of these practices are incorporated into our methodology, while others practices are recommendations we make for implementation.

- ▶ World-class conversion vendors

The Migration Practice of IBM nurtured relationships with skilled business partners over many years. These partners are not new to IBM. Instead, they are conversion vendors with whom we developed long-term relationships and worked with successfully.

- ▶ Use of automated tools

Our goal is to run successful migration projects with minimum cost and risk. IBM and our partners are invested in the development of migration tools that help accomplish that objective. These tools help us in the following ways:

- Gather information about an environment to be migrated
- Automate the migration of a large percentage of objects in that environment
- Ensure the quality of all migrated objects
- Give us the flexibility to impose the standards of a customer on those migrated objects

- ▶ Availability of skills within IBM

Migration projects take full advantage of the IBM Toronto and Silicon Valley laboratories, and other sources of skills within IBM. Migration projects are not passed to partners for execution. Instead, experienced internal IBM experts are assigned to ensure that best practices and lessons learned are used and the implementation of the workload of the customer on DB2 is effective and efficient.

The IBM Migration Practice draws upon these skills for project management, performance management, and capacity analysis.

The goal of the IBM Migration Practice is to run successful migration projects that are on time and within budget. We are responsible for blending all the resources into an effective team, and for ensuring that the team works together effectively throughout the project. Our responsibility is to provide customers with price estimates and contracts. We want your migration project to be a positive experience with IBM, thus paving the way for more successful interactions with IBM in the future.

Customer partnering is a key element to success

IBM found that a key element of every successful project is to establish early in the project a clear plan for how the IBM team works together with your staff. During a project, there are tasks for which the IBM team takes primary responsibility, and tasks for which your staff takes primary responsibility. Whenever possible, your staff shadows our team as we conduct many of our tasks, and we make recommendations to your staff as they conduct their tasks. In this way, your staff benefits from the expertise of our team, and your staff gains a deeper understanding of the migrated objects they ultimately are responsible for maintaining.

The structure and content of the migration project that is described here is a suggested baseline approach for maximizing the technical and migration process value of the IBM Migration Practice. At the same time, the approach takes advantage of customer functional Subject Matter Expert skills and familiarity with the application implementation used in the area of testing. The goals are to reduce project costs to the customer, use existing materials and skills to utmost advantage, provide customer insight to and ownership of migration testing acceptance, and reduce project timeline while maintaining the quality of the project. This general approach is the best balance of project cost, timeline, and quality results. However, the roles and specific task ownership might be modified based on unique customer requirements.

Highlighting the roles and responsibilities involved in a migration project is important. Your staff plays an important role in the following areas:

- Project management

Although IBM bears the overall responsibility for the success of the project, a project manager must be assigned from your staff to manage all of the tasks that are assigned to you. This manager also is important for confirming the accomplishment of milestones, and communicating the status of the project back to your management.

- Management of the physical and machine environment

The IBM team relies on the client staff to maintain things such as the physical machine resources, ongoing DBA maintenance tasks, operating system software, and so on.

If the client does not have the staff to conduct these tasks, this need is identified in the Assessment phase of the project, and more IBM effort is planned to support those requirements.

- Testing

IBM and the client jointly develop the test strategy for the converted objects. The client is responsible for providing the best approach for testing each application, thus ensuring the best functional coverage that is based on their existing functional and test process expertise with the applications.

IBM uses your existing test process and materials to conduct integrity-level testing of the migrated applications. The customer performs system testing of the migrated applications with IBM personnel available to support the system testing in a defect fix role. The client is responsible for providing suitable test plans, test cases, data population, expected results, and augmenting these test materials as required to meet testing coverage requirements. If the customer needs assistance with augmenting test materials, the scope is defined during the assessment phase.

During system testing, IBM reviews the performance of the database workload and tunes access paths and DB2 resources to optimize performance.

The client is responsible for acceptance testing the applications before the applications are implemented into production. IBM personnel are available to provide defect support as required during this customer testing.

Testing is structured in this manner because only the staff of the customer and users understand all of the functional and operational intricacies of your code and therefore are in a position to specify the necessary testing.

IBM provides a warranty for all migrated objects that are defined by the Statement of Work, and resolves any issues that are related to the migrated objects through the end of that period.

Migration strategy

A high-level view of the project is shown in Table A-1.

The phased conversion and implementation of the client source DBMS environment requires analysis and study to determine precisely the order and grouping of applications that are migrated. The summary and detail representations of the project do not encompass the application or data object phasing of the project. The sequence and grouping of applications that are migrated are determined during the phase zero assessment. As a deliverable of phase zero, the plans that are shown in Table A-1 are expanded to accommodate the sequence, grouping, and constraints of a phased migration of the applications.

Table A-1 Summary of activities and results by phase

Phase	Activities	Results
Phase zero	Collate inventory	Inventory
	Complexity analysis	Complexities defined
	Platform configuration	Platform definition
	Project planning	Project plan
	Configuration management	Freeze policy
		Problem/fix management
	Communication policy	Weekly status meetings
		Monthly status meetings
		Escalation procedure

Phase	Activities	Results
Calibration, design, and planning	Conversion strategy definition	Review client source DBMS, DB2, and JCL development standards
		Review error handling and security standards
		Calibrate code translation to specific client requirements
	Develop phasing and coexistence strategy	Analyze interdependencies between applications
		Develop strategy for co-existence
	Convert calibration sample of code	Calibrated sample of converted components
		Validated conversion strategy
	Platform design	Environment definition (DDL, scheduling, query access)
	Test strategy	Review content and suitability of existing test materials (test plans, cases, data population, expected results, and coverage) and augmentation requirements.
		Integrity testing strategy
		System testing strategy
		Acceptance testing strategy
		Test Script augmentation development process
		Batch schedule analysis
		Test data capture process
Conversion	Database conversion	DDL
	Data migration	Data migration programs
	Code conversion	Converted programs, JCL
	Scheduler modification	Schedule
	Unit-level integrity testing	Execution of customer test materials and verification of results
System testing (customer-owned with IBM defect support)	Test environment creation	Populated source DBMS and DB2 test environments
	Test each online transaction by using test scripts	Tested DB2 online programs
	Test each batch job step	Test DB2 batch programs
	Performance tuning review	Performance tuning report
Acceptance testing (customer-owned with IBM defect support)	Client acceptance testing	Accepted DB2 applications ready for implementation

Phase	Activities	Results
Implementation (customer-owned with IBM support)	Set up production environment	Support documentation
	Modify production schedule	
	Migrate production data	Verification of migrated data
	Acceptance	Applications live

The migration strategy detail is provided next.

Phase zero

Phase zero is used to confirm the scope of the project and plan for the strategy and deployment of the remainder of the effort. The following primary tasks and associated deliverables result from Phase zero:

- Inventory confirmation

In this task, the IBM Migration Team and the client work together to confirm the inventory of applications, databases, and their associated components. We identify the location of all source DBMS components, the count of components by type, and the complexity of this inventory. The inventory of objects to convert, and other migration-related tasks become the agreed upon scope of the project. Before the remainder of the project begins, IBM and the client agree on the complete scope of the migration project.

- Conversion strategy

Most migration projects require that the applications are migrated in a phased approach. A significant amount of analysis is conducted to determine the sequence of the migrations, and which applications are grouped.

In the initial weeks of the migration project, the IBM team works with the client team to determine the best strategy to migrate all of the objects in scope. This analysis examines things such as code and data object dependencies, naming standards, security schemes that are employed, business processing cycles, and coexistence requirements.

The benefits and drawbacks of available migration strategies are presented to the client team. The client and IBM decide on the strategy to use for the remainder of the project.

The outcome of this process results in a document that is delivered to you, which explains all of the decisions that are made thus far. Decisions also are made during this phase about the method of code delivery to IBM, and change management procedures, which are followed during the project.

- Development of a project plan and project management processes

The inventory of components to migrate and the associated complexities are used to drive the resource planning for the IBM migration team.

The client resources to be dedicated to the testing tasks are determined. A schedule of external constraints, such as maintenance projects, hardware or software upgrades, and periods of high business activity, is built.

All of the information that is gathered and the decisions that are made are incorporated into a Detailed Project Plan, which is delivered to you. This plan identifies all of the tasks that are assigned to the IBM team and to the client staff, and the milestones for the project. The plan serves as the definitive document to synchronize the work of the project between the IBM and the client teams.

The processes and controls that are used to control a project are key to its success. IBM works with the client to implement suitable processes that are based on existing client practices. When required, IBM provides standard procedures that are based on best practices. The key processes are communication, configuration management, freeze policy and action item, and problem and risk management.

Calibration, design, and planning phase

The calibration, design, and planning phase of the migration project includes the following tasks:

- ▶ IBM reviews the DB2 development standards for coding, security, error handling, and so on, and develops a set of standards for the converted code.
- ▶ IBM factors into the conversion process the specific needs of the client. The resulting conversion process is documented for the client in the Conversion Strategy document. This Conversion Strategy document contains the technical roadmap of the migration, including the physical database design.
- ▶ Calibration is designed to validate the Conversion Strategy. In this phase, we factor into our tooling the specific conversion strategy for the client. A few representative components of each type are chosen and converted to ensure that the conversion is working correctly, and proceeds efficiently.
- ▶ This phase is used to validate that the code produced is easy to maintain.
- ▶ Detailed resource and schedule planning is finalized.
- ▶ The data transfer requirements from the source DBMS to DB2 are reviewed and the data transfer strategy is optimized.
- ▶ The test strategy for system and acceptance is finalized.

Conversion phase

The conversion phase of the migration project includes the following tasks:

- ▶ Insights gained during the Calibration phase are incorporated into the migration strategy.
- ▶ The sequence in which to convert and group components is determined.
- ▶ The conversion of the entire portfolio of objects begins.
- ▶ The conversion is conducted by using IBM or partner migration tools to ensure quality and consistency.
- ▶ This work often is completed off-site at an IBM location or at the location of an IBM partner. The decision about where this work is done is agreed to by the client and IBM during the Assessment phase.

Milestones are identified in the detailed project plan to measure the progress of the migration throughout this phase.

Testing phase

The testing phase consists of the following steps:

- Conversion refresh

Because most customers are unable to freeze their applications during a conversion, it is important to plan for the reconversion of some objects that are found at inappropriate levels as the project proceeds. Because the calibration programs are converted early in the project, often these programs require reconversion. Planning for this phase initially is done in the Assessment phase, and monitored and updated throughout the project as appropriate.

- Data migration

Data migration includes several tasks: unload, restructure, cleanse, and reload. The IBM team uses a combination of automated tools and manual processes to ensure that the data is accurately deployed to the new environment.

- Testing

- Augmenting and running test scenarios
- Input that is needed for testing:
 - Database content
 - Input transactions for batch jobs
 - Scripts for interactive online programs
 - Test plans and cases, including the flow of your programs and processes
 - Expected results
- Unit integrity testing of individual modules
- System testing of subsystems and applications, including performance testing by using actual volumes of data
- Acceptance testing (running in parallel mode is assumed):
 - Create the parallel system test environment.
 - Take a data baseline of the application.
 - Convert the data to DB2 by using the data migration routines that are created by IBM.
 - Each group of components is converted as the group is installed in the parallel test environment.
 - Run the two systems in parallel.
 - After each component is run, the modified database tables, reports, and so on, are compared daily to ensure functionally equivalent results.
 - Validate performance of user functionality.
- Responsibility for execution of testing:
 - Development and execution of test scenarios often is owned by the client who reuses testing materials. Any required augmentation by IBM is identified during the assessment phase.
 - All final testing must be run in the client environment to measure performance and the accuracy of the migrated objects in a meaningful way.
 - Program output and performance results must be captured within the client source environment in the early stages of the project for later comparison with the output and performance results that are achieved by the converted code in the target environment.

Implementation phase

This phase consists of implementation of and cutover to the new database.

Implementation often is conducted during two weekends. During the first weekend, a dry run is conducted to validate the implementation procedures and to determine what, if any, changes must be made to the procedures. The implemented system is then backed out and the unconverted system restored. Any issues or problems that are discovered during the dry run are corrected and the implementation proceeds on a subsequent weekend.

IBM Information Management Software Services

IBM Information Management Software Services maintains a website that lists all of the services offerings for DB2 for z/OS and other Information Management products. If you are interested in engaging IBM to assist with your conversion to DB2 for z/OS, more information is available at this website:

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



Additional material

This book refers to material that is downloaded from the Internet as described in the following sections.

Locating the web material

The web material that is associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server:

<ftp://www.redbooks.ibm.com/redbooks/SG248044>

Material also is available at the IBM Redbooks website:

<http://www.ibm.com/redbooks>

Select **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG248044.

Using the web material

The web materials that accompany this book include the following files:

- ▶ SG24-8044-Application and SQL Tuning Samples.zip

This .zip file contains the following application and SQL Tuning samples shown in Chapter 11, “Application and SQL tuning” on page 175:

- Common Table Expression.txt
- Filter In SQL.txt
- Recursive SQL DDL.txt
- Recursive SQL example.txt
- Recursive SQL INSERTS.txt
- Sample ordered set.txt
- Sample ordered set2.txt
- Sample ordered set3.txt

- Sample ordered set4.txt
 - Sample Scalar Fullselect in SELECT clause.txt
 - Sample Scalar Fullselect in WHERE clause.txt
 - SELECT From DELETE.txt
 - SELECT From UPDATE.txt
 - Select Row Change Timestamp.txt
 - Update Using Row Change Timestamp.txt
- SG24-8044-DB2 for z/OS feature usage samples.zip
- This .zip file contains the following DB2 for z/OS feature usage samples shown in Chapter 12, “Maximizing DB2 for z/OS features” on page 199:
- Create Sequence Object.txt
 - DDL for a Table with an XML Column.txt
 - DDL for a Table with LOB Columns.txt
 - DDL for Temporal Example of Business Time.txt
 - DDL for Temporal Example of System Time.txt
 - DDL to Create a Global Temp Table.txt
 - Declare and Insert Into Global Temp Table.txt

System requirements for downloading the web material

The Web material requires the following system configuration:

Hard disk space:	2 MB minimum
Operating System:	Windows
Processor:	Intel 386 or higher
Memory:	16 MB

Downloading and extracting the Web material

Create a subdirectory (folder) on your workstation, and extract the contents of the web material .zip file into this folder.

Related publications

The publications listed in this section are considered suitable for a more detailed description of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide more information about the topic in this book. Some of the following publications might be available only in softcopy:

- ▶ *DB2 10 for z/OS Technical Overview*, SG24-7892
- ▶ *Data Integrity with DB2 for z/OS*, SG24-7111
- ▶ *DB2 9 for z/OS Stored Procedures: Through the CALL and Beyond*, SG24-7604
- ▶ *DB2 for z/OS: Data Sharing in a Nutshell*, SG24-7322

Search for, view, download, or order these publications and other Redbooks, Redpapers, Web Docs, draft, and other materials, at this website:

<http://www.ibm.com/redbooks>

Other publications

The following publications also are available as information sources:

- ▶ *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974
- ▶ *CICS Transaction Server for z/OS Version 4 Release 2 DB2 Guide*, SC34-7164
- ▶ *CICS Transaction Server for z/OS Version 4 Release 2 Application Programming Guide*, SC34-7158
- ▶ *DB2 10 for z/OS Installation and Migration Guide*, GC19-2974
- ▶ *DB2 10 for z/OS Managing Security*, SC19-3496
- ▶ *DB2 10 for z/OS Application Programming and SQL Guide*, SC19-2969
- ▶ *DB2 10 for z/OS Administration Guide*, SC19-2968

Online resources

The following websites also are relevant as information sources:

- ▶ IBM Academic Initiative
<http://ibm.com/university/academicinitiative>
- ▶ IBM System z Job Board
<http://www.systemzjobs.com>
- ▶ DB2 for z/OS and IMS tools
<http://www.ibm.com/software/data/db2imstools/products/db2-zos-tools.html>
- ▶ Software AG Adabas

- http://www.softwareag.com/corporate/products/adabas_2010/default.asp
- ▶ CA Datacom
<http://www.ca.com/us/products/detail/ca-datacom.aspx>
- ▶ CA IDMS
<http://www.ca.com/us/products/detail/CA-IDMS.aspx>
- ▶ ATERAS
<http://www.ateras.com/Home.aspx>
- ▶ Consist Software Solutions
<http://www.consist.com/>
- ▶ MOST
<http://www.mosttechnologies.com/>
- ▶ The Free Library
<http://www.thefreelibrary.com/ATERAS+Announces+Successful+Conversion+of+University+of+Miami%27s+IDMS...-a0159260844>

Help from IBM

IBM DB2 Migration Project Office

<http://www.ibm.com/software/solutions/softwaremigration/dbmigteam.html>

IBM Support and downloads

<http://www.ibm.com/support>

IBM Global Services

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

Glossary

Application conversion A conversion is a change of your application or database technology without changing the user interface and other application inputs or outputs. For example, convert Software AG Natural to COBOL, and Adabas to DB2 for z/OS; CA ADS/Online to COBOL and CA IDMS, and Easytrieve to COBOL. These examples show how the technology is changed, but not the functionality. You do not change business processes, or need to retrain the user community. Conversion is suitable for an organization with a large user base and a limited budget.

Application migration Migration often does not include a language change or a change in the type of database management system. Migration might be an upgrade to a new release, or the movement of a relational COBOL application from a mainframe environment to an off-mainframe environment. Migration is a mechanical reimplementation of the same application in a new environment or on a new release.

Application modernization Modernization retains the basic functionality of your existing applications and databases, but changes the look, flow, or general feel for the user community, and changes the underlying technology for your support teams. A modernization of Natural to Java results in web-based applications that you maintain with another standard IDE of choice, rather than perpetuating 3270 routines that decide what key was pressed or to count how many lines of data fit on a screen. Modernization tools use the capabilities of the J2EE frameworks or .NET, generating methods, classes, functions, and objects that provide buttons and scroll bars. Modernization provides a new set of web-based Java or .NET applications. The developer uses the existing application code and flow as the basis for the new application, guiding the toolset to generate a fully native web application for Java or .NET.

Application re-engineering The application of technology and management science to the modification of existing systems, organizations, processes, and products to make them more effective, efficient, and responsive.

Application rewrite This approach implements a large portion of the existing application functionality without code re-use. The rewrite is done from scratch when none of the existing application source code is used.

Application transformation Transformation is what happens to an application or database when it is converted, modernized, or migrated. The core of the application or database is unchanged, but the form and technology are different.

Business agility Ability for a business to adapt rapidly and in a cost-efficient manner to changes in the business environment. Concepts that are addressed include flexibility, adaptability, efficiency, and cost effectiveness.

Index

Numerics

1 MB page frames 168
4GL programs 46

A

access 8, 26, 61, 79, 86, 115, 156, 175, 206, 223
access control 21, 141
access data 8, 164, 211
access path 15, 89, 127, 178, 210
Adabas 4, 71, 111
Advance 129
ALTER BUFFERPOOL 168, 206
ALTER statement 194, 207
ALTER TABLE 194, 205
 statement 195, 217
American National Standards Institute 8
APIs 81, 207
application xvii–xviii, 8, 25, 37, 48, 57, 74, 84, 114, 155,
175, 199, 223, 231, 246
Application migration 29
application period 205
Application re-engineering 29
Application replacement 29
Application re-write 29
Assessment 40, 145
ATERAS 111
attribute 172
Automated conversion 37, 144
Automated conversion tool
 lessons learned 78
auxiliary table 218

B

base table 17, 205
base tables 9
Big bang 60
bind option 193, 208
bind parameter 208
bitemporal support 5
BLOB 218
BPOOL HIT RATIO 167
buffer pool 19, 62, 93, 162, 178, 206
 activity 93, 163
 BP0 163
 CPU overhead 213
 directory 163
 scan 163
 size 164
 space 93, 163, 206
 storage 164
buffer pool management 168
buffer pools 12, 66, 92, 155
built-in function 209

C

CA IDMS 71
CA-IDMS 83
CALC keys 87
catalog table 18, 193
CCSID 203
CHECK DATA 17, 207
CHECK INDEX 17
CHECK LOB 17
CHKFREQ 158
CHKTYPE 158
CICS xviii, 11, 16, 29, 39, 49, 65, 85, 113, 156, 206
CICS Transaction Server 169
class 119, 171, 222
Classic data federation 79
Classic data replication 79
CLOB 218
CLOSE CURSOR 186
CLUSTER 210
COLLID 159
Column
 XML 232
column value 178
common table expressions 188
complex data model 31
components 11, 30, 38, 53, 66, 78, 91, 222
compression 12, 31, 213
compression dictionary 213
condition 204
CONNECT 88, 204
ConsistADS 111
ConsistADS- IDE 120
ConsistDMS 117
Conversion 42, 113
conversion best practice 47
conversion guidelines 43
conversion methodology 25
conversion mode 85
conversion tool case study 64
COPY 17
COPYTOCOPY 17
COUNT 10, 126
CPU overhead 168, 187, 210
CPU time 212
CTHREAD 156

D

data xviii, 8, 25, 48, 57, 74, 85, 112, 157, 176, 202, 223
data access options 38
Data control 10
Data definition 10
Data definition language 11
Data Management Threshold 165
Data manipulation 10

- data set 15, 160, 193
- data sharing 14, 194, 213
- Data transformation 79
- Data type 188
- database administrator 8, 43, 55
- database system 38, 112
- Datacom/DB 4
- DATASIZE 194
- DATE 66, 97, 203
- DB2 10 xviii, 5, 12, 77, 88, 127, 156, 182, 205
 - base 174
 - data 21, 92, 161, 206
 - environment 174
 - migration planning xviii
 - running 174
 - SQL 162
 - table spaces 168
 - use 159, 206
- DB2 9 xviii, 156
 - DB2 10 157
 - dynamic prefetch 167
 - system 174
- DB2 Administration Tool for z/OS 18, 118
- DB2 Buffer Pool Analyzer for z/OS 165
- DB2 catalog 14
- DB2 Change Accumulation Tool for z/OS 19
- DB2 Cloning Tool for z/OS 19
- DB2 family 15, 58
- DB2 for z/OS 71, 111
- DB2 for z/OS Tools 18
- DB2 for z/OS Utilities Suite 17
- DB2 High Performance Unload for z/OS 18, 147
- DB2 Log Analysis Tool for z/OS 19, 147
- DB2 Object Comparison Tool for z/OS 18, 147
- DB2 Object Restore for z/OS 19
- DB2 optimizer 15
- DB2 Path Checker and DB2 Bind Manager 20
- DB2 Query Monitor for z/OS 19, 118
- DB2 Recovery Expert for z/OS 19
- DB2 SQL Performance Analyzer 19, 118
- DB2 subsystem 16, 92, 130, 156
- DB2 system 92, 143, 156, 181
- DB2 utility 156
- DB2CONN 171
- DB2ENTRY 171
- DB2TRAN 172
- DBAT 157
- DBCLOB 218
- DBM1 address space 168, 181
- DBMS 8, 25, 37, 51, 57, 75, 177, 217, 224
- DBRM 196
- DDF 16, 159, 181, 208
- DDL 11, 43, 66, 76, 91, 123, 155, 215, 225, 231
- DDMMAPPING file 136
- default value 124, 157
- deferred write 165
- Deferred Write Threshold 165
- DELETE 10, 81, 89, 127, 160, 183, 203, 232
- delete 10, 62, 148, 160, 184, 203
- Denormalization 34

- dirty read 192
- DISTINCT 180
- Distributed 16, 159
- Distributed Data Facility 159
- DMTH 165
- DRDA 16, 181
- DS8000 12
- DSMAX 161
- DSNZPARM 157
 - CHKFREQ 158
 - CHKTYPE 158
 - TCPALVER 160
- DWQT 165
- dynamic SQL 15, 38, 62, 162, 181, 211
- dynamic SQL statement 216
- dynamic statement cache 181

E

- Easytrieve 28
- eav 148
- eavRPM 111
- EDM_SKELETON_POOL 162
- EDMDBDC 162
- EDMSTMTC 161
- efficiency 8, 73, 118, 166, 196
- element 85, 222
- entity relationship model 109, 150
- environment xvii, 4, 8, 29, 40, 48, 61, 75, 84, 113, 156, 176, 202, 222, 246
- EXEC SQL 140, 215
- EXPLAIN 193, 195
- Explain 43
- expression 9, 184, 215
- external stored procedures 180

F

- FETCH 89, 127, 177, 209
- fetch 176, 209
- FICON 12
- FINAL TABLE 183
- FlashCopy 12
- flat file 33
- flexibility xvii, 8, 33, 46, 48, 76, 85, 112, 206, 222, 246
- FROM OLD TABLE 184
- function 10, 38, 85, 115, 180, 204

G

- GB bar 161
- GENERATED ALWAYS 183, 205
- getpage 167, 213
- GRANT 13
- GROUP BY 180

H

- handle 10, 30, 38, 48, 59, 92, 114, 160, 185, 204
- history table 205
- host variables 178

I

I/O 12, 31, 38, 62, 160, 176, 210
I/O module 33
IBM Academic Initiative 5
IBM DB2 Migration Project Office 69
IBM DB2 Path Checker 196
IBM DB2 Query Monitor 197
IBM DB2 SQL Performance Analyzer 196
IBM Tivoli OMEGAMON XE for DB2 Performance Expert
on z/OS 19
IDE 117
IDMS/DB 4
IDHTOIN 159
IDXBPOOL 163
image copy 13
IMS xvii, 4, 12, 16, 80, 156, 180, 216
index xix, 12, 62, 87, 122, 161, 180, 208
index usage 187
in-memory table space 168
INPUT SEQUENCE 183
INSERT 10, 81, 89, 127, 160, 176, 202
insert 160, 182, 204
installation 13, 49, 99, 130, 156
integrated development environment 117
Integrity 13, 125
International Standards Organization 8
IRLM 174
IRLMRWT 157
IS 8, 89, 189
IX 163

J

Java 29, 113, 169
JDBC 81, 157

K

KB 161
keyword 210

L

LFAREA 168
LIKE 216
list 17, 45, 115, 161, 178, 209
list prefetch 167, 180
LISTDEF 17
literal replacement 182
LOAD 17, 66, 194, 204
LOB 33, 157, 218, 232
LOB column 218
LOBs 21, 33, 218
LOCK 89
locking 44, 157, 179, 216
locks 89, 157, 179, 213
LOG 159
logging 14, 182, 214
logical record facility 87
Loose co-existence 60
LPAR 49, 128

LRF 87

M

M xvii
maintenance 8, 37, 48, 90, 117, 174, 193, 223
manual conversion 33, 75
materialized query tables 12, 211
MAXDBAT 157
maximum number 156
MAXOPENTCBS 170
MAXRBLK 161
MEMBER CLUSTER 213
member cluster 214
MERGE 185
MERGECOPY 17
Migration xvii, 27, 58, 119, 161, 221
migration strategies 27
migration strategy 57
Modernization 29, 147
MODIFY 18, 89, 208
MODIFY RECOVERY 18
multi-row fetch 182
MXT 170

N

native SQL procedures 181
Natural 29, 111
NULL 10, 114, 203
NUMBER OF LOGS 160
NUMLKTS 157
NUMLKUS 157

O

Object 140, 232
ODBC 81, 152, 157, 208
OnTarget 111
open transaction environment 173
optimistic locking 179
optimization 12, 81, 166, 187
OPTIMIZE FOR 1 ROW 180
options 12, 26, 49, 57, 123, 176, 201
ORDER 89, 126, 178, 212
ORDER BY 89, 127, 180, 212
ORDER BY clause 127, 190
ordering 85, 180
OTE 173
OUTBUFF 160

P

page access 210
page set 165
PAGE-INS REQUIRED FOR READ 167
parameter markers 181
PART 206
partition-by-growth 214
partitioned table space 18, 217
partitioning 12, 66, 112, 205
partitions 12, 67, 217

- Performance xviii, 42, 56, 67, 86, 113, 158, 179, 208, 225
- performance xviii, 5, 8, 29, 38, 48, 61, 75, 86, 112, 158, 175, 199, 222
- performance improvement 173
- PGFIX=YES 168
- Piece at a time 61
- Pilot 42
- POOLINAC 159
- Post-conversion tuning 68
- predicate 10, 187
- Prefetch counters 167
- Prefetch disabled counters 167
- productivity 15, 117
- Proof of concept 35
- PROTECTNUM 172
- pureXML 21

Q

- query 8, 35, 60, 99, 112, 157, 180, 206, 225

R

- RACF 13, 92, 117, 160
- RANDOM 166
- range-partitioned table spaces 214
- READS 160
- real storage 12, 164
- real-time statistics 193
- RECOVER 18
- Recovery 14
- recursive SQL 189
- Redbooks website 233
 - Contact us xix
- referential integrity 8, 44, 113, 202
- Relational redesign 68
- remote location 159
- REORG 17, 193, 211
- REORG INDEX 18
- REORG TABLESPACE 18, 193
 - utility 18
- REORG utility 193
- reorganize a table space 194
- reorganize an index 194
- requirements xvii, 5, 39, 48, 57, 71, 77, 83, 111, 163, 185, 212, 223, 232, 246
- residual predicates 188
- return 34, 93, 173, 176, 209
- REVOKE 13
- RID 161
- risk analysis 54
- ROLLBACK 89, 127
- ROWID 88, 126, 183, 218
- RRSAF 16, 156
- RUNSTATS 18, 45, 105, 149, 193

S

- same page 211
- same way 181
- Scalability 12

- scalar fullselect 184
- scalar functions 10, 209
- Schema 96
- schema 12, 85, 118, 165, 217
- Security 13, 119
- SELECT from MERGE 186
- SELECT FROM UPDATE 183
- SEQ 114, 166
- sequential access 187
- Server 16, 80, 123, 169
- SET 139, 179, 202
- side 109
- skills and roles 54
- sort cost 180
- sort record 180
- space map 213
 - page 213
- SQL 8, 28, 38, 55, 62, 75, 86, 112, 162, 176, 199, 231
- SQL interface 15
- SQL procedure 181
- SQL procedures 181
- SQL scalar 184
- SQL statement 62, 176, 212
- SQL stored procedures 208
- SQL tuning 195
- SQLCODE 177
- SRTPOOL 161
- statement 10, 34, 62, 101, 161, 176, 202
- static SQL 15, 62, 139, 193
- statistics 14, 44, 65, 76, 158, 176
- Strategy comparison 63
- Structured Query Language 10
- synchronous I/O 164
- SYNCHRONOUS READS 167
- SYSCOPY 18
- SYSIBM.SYSINDEXSPACESTATS 194
- SYSIBM.SYSTABLESPACESTATS 194
- SYSLGRNX 18
- system period 205
- System z 1, 7, 116, 169, 181
- system-period data 205
- system-period data versioning 205
- system-period temporal table 205
- SYSTRANS 138

T

- table expression 188
- table space
 - buffer pool 164
 - data 17, 164, 193, 205
 - data set 17
 - definition 194, 214
 - level 157
 - lock 157
 - name 205
 - option 168, 213
 - page 157
 - page set 168
 - partition 17, 157
 - REORG TABLESPACE 18

- scan 214
- structure 205
- table space scans 67
- tables 8, 28, 38, 48, 62, 78, 85, 112, 155, 179, 201, 228
- TBSBP16K 163
- TBSBP32K 163
- TBSBP8K 163
- TBSBPLOB 163
- TBSBPOOL 163
- TBSBPXML 163
- TCP/IP 119, 160
- TCPALVER 160
- TEMPLATE 17
- temporal 21, 40, 205
- THREADLIMIT 172
- THREADWAIT 172
- times DB2 176
- TIMESTAMP 179, 205
- Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS 166
- TOTALROWS 194
- Transaction manager support 16
- transparency 61, 113
- triggers 8, 88, 162, 183, 204
- TS 163
- TYPE 158

U

- UDF 209
- UDFs 209
- uncommitted read 192
- UNIQUE 127, 203
- unique index 208
- universal table space 217
- UNLOAD 17
- UPDATE 10, 81, 88, 115, 160, 179, 202, 232
- URCHKTH 158
- URLGWTH 159
- user defined function 209

V

- VALUE 202
- VALUES 182, 202
- VARCHAR 33, 43, 114, 180, 203
- variable 202
- Version xvii, 12, 164
- versions 14, 52, 117, 161
- Vertical Deferred Write Threshold 165
- Virtual Pool Sequential Steal Threshold 164
- VPSEQT 164
- VPSIZE 168

W

- WHEN MATCHED 185
- WITH 114, 179, 201
- WLM 12, 34, 174, 180
- work file 161, 192, 209

X

- XML 17, 148, 162, 218
- XML column 218
- XML data 218
- XML data type 218
- XML documents 218

Z

- z/Architecture 12
- z/OS xvii, 1, 153, 155, 175, 246
- z/OS Installation 130, 161
 - DB2 10 161
- z/OS Workload Manager 174
- zEnterprise 114 20
- zEnterprise 196 20
- zIIP 181



Streamline Business with Consolidation and Conversion to DB2 for z/OS

(0.5" spine)
0.475" <-> 0.873"
250 <-> 459 pages



Streamline Business with Consolidation and Conversion to DB2 for z/OS



Consolidate your investment in DB2 for z/OS

Accelerate conversions with tools technology

Adapt to industry dynamics

Time to market, flexibility, and cost reduction are among the top concerns common to all IT executives. If significant resource investments are placed in mature systems, IT organizations need to balance old and new technology. Older technology, such as non-IBM pre-relational databases, is costly, inflexible, and non-standard. Users keep their information on the mainframe and thus preserve the skills and qualities of service their business needs. But users also benefit from standards-based modernization by migrating to DB2 for z/OS. With this migration, users deliver new application features quickly and respond to changing business requirements more effectively.

When migrating, the main decision is choosing between conversion and re-engineering. Although the rewards associated with rebuilding applications are high, so are the risks and customers that are embarking on a migration need that migration done quickly.

In this IBM Redbooks publication, we examine how to best approach the migration process by evaluating the environment, assessing the application as a conversion candidate, and identifying suitable migration tools.

This publication is intended for IT executives who are considering migrating their information to a modern database management system.

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