

# Leveraging DB2 10 for High Performance of Your Data Warehouse

Comprehensive platform for  
architected data warehouse solutions

Faster time to value for business  
decision making

Standards-based for  
flexibility and change



Whei-Jen Chen  
Scott Andrus  
Bhuvana Balaji  
Enzo Cialini  
Michael Kwok  
Roman B. Melnyk  
Jessica Rockwood





International Technical Support Organization

**Leveraging DB2 10 for High Performance of Your  
Data Warehouse**

January 2014

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

**First Edition (January 2014)**

This edition applies to DB2 for Linux, UNIX, and Windows Version 10.5.

© Copyright International Business Machines Corporation 2014. All rights reserved.

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

# Contents

<b>Notices</b>	ix
Trademarks	x
<b>Preface</b>	xi
Authors	xi
Acknowledgements	xiv
Now you can become a published author, too!	xv
Comments welcome	xv
Stay connected to IBM Redbooks	xv
<b>Part 1. Overview</b>	1
<b>Chapter 1. Gaining business insight with IBM DB2</b>	3
1.1 Current business challenges	4
1.2 Big data and the data warehouse	5
1.2.1 Data warehouse infrastructure	6
1.3 High performance warehouse with DB2 for Linux, UNIX, and Windows	7
<b>Chapter 2. Technical overview of IBM DB2 Warehouse</b>	9
2.1 DB2 Warehouse solutions	10
2.1.1 The component groups	10
2.1.2 Editions	11
2.2 Expert systems	17
2.2.1 PureData for operational analytics	18
2.2.2 DB2 Warehouse components	19
2.2.3 Database management system	22
2.3 DB2 Warehouse topology	25
<b>Chapter 3. Warehouse development lifecycle</b>	29
3.1 Defining business requirements	30
3.2 Building the data model	31
3.2.1 Defining the physical data model	31
3.2.2 Creating the physical data model	32
3.2.3 Working with diagrams	36
3.2.4 Using the Diagram Editor	38
3.2.5 Optimal attributes of the physical data model	39
3.2.6 Model analysis	40
3.2.7 Deploying the data model	42
3.2.8 Maintaining model accuracy	45

3.3 Matching the technology to the requirements . . . . .	48
<b>Part 2. Technologies . . . . .</b>	<b>51</b>
<b>Chapter 4. Column-organized data store with BLU Acceleration . . . . .</b>	<b>53</b>
4.1 Simple and fast with BLU Acceleration . . . . .	54
4.1.1 Dynamic in-memory columnar technology . . . . .	54
4.1.2 Actionable compression . . . . .	55
4.1.3 Parallel vector processing . . . . .	55
4.1.4 Data skipping . . . . .	56
4.2 Getting started with BLU Acceleration . . . . .	56
4.2.1 Capacity planning . . . . .	57
4.2.2 Storage requirements . . . . .	58
4.3 Creating a column-organized data store . . . . .	58
4.3.1 Single configuration setting for analytic workloads . . . . .	59
4.3.2 Fine-tuning a configuration . . . . .	60
4.3.3 Creating column-organized tables . . . . .	61
4.3.4 Converting to column-organized tables . . . . .	63
4.4 Loading a column-organized data store . . . . .	63
4.4.1 Synopsis tables and data skipping . . . . .	64
4.4.2 Compression of column-organized tables . . . . .	65
4.4.3 Reducing load times . . . . .	67
4.5 Managing a column-organized data mart . . . . .	68
4.5.1 Query execution plans and the new CTQ operator . . . . .	68
4.5.2 Database maintenance . . . . .	70
<b>Chapter 5. Row-based data store . . . . .</b>	<b>71</b>
5.1 Scale-out solution: Database Partitioning Feature . . . . .	72
5.1.1 Partition . . . . .	72
5.1.2 Partition group . . . . .	73
5.1.3 Table spaces in a DPF environment . . . . .	74
5.1.4 Partitioning keys . . . . .	74
5.1.5 Partition maps . . . . .	75
5.1.6 Choosing partitioning keys . . . . .	76
5.1.7 Concluding remarks . . . . .	78
5.2 Scale-up solution: Intrapartition parallelism . . . . .	79
5.3 Other features for row-based data warehouse . . . . .	81
5.3.1 Compression . . . . .	81
5.3.2 Multidimensional data clustering . . . . .	83
5.3.3 Range partitioning . . . . .	87
<b>Chapter 6. Data movement and transformation . . . . .</b>	<b>91</b>
6.1 Introduction . . . . .	92
6.1.1 Load . . . . .	92

6.1.2	Ingest	94
6.1.3	Continuous data ingest	96
6.2	DB2 Warehouse SQL Warehousing Tool	98
6.2.1	Architecture	99
6.2.2	Development environment	100
6.2.3	Moving from development to production	101
6.2.4	Runtime environment	102
<b>Chapter 7.</b>	<b>Monitoring</b>	<b>105</b>
7.1	Understanding monitor elements	106
7.1.1	Monitor element collection levels	108
7.1.2	New monitoring elements for column-organized tables	110
7.2	Using DB2 table functions to monitor your database in real time	111
7.2.1	Monitoring requests	111
7.2.2	Monitoring activities	112
7.2.3	Monitoring data objects	112
7.2.4	Monitoring locks	113
7.2.5	Monitoring system memory	113
7.2.6	Monitoring routines	113
7.3	Using event monitors to capture information about database events	114
7.4	Monitoring your DB2 system performance with IBM InfoSphere Optim Performance Manager	116
7.4.1	Information dashboards	117
7.4.2	OPM support for DB2 10.5	118
<b>Chapter 8.</b>	<b>High availability</b>	<b>119</b>
8.1	IBM PureData System for Operational Analytics	120
8.2	Core warehouse availability	123
8.2.1	Roving HA group configuration for the administration hosts	124
8.2.2	Roving HA group configuration for the data hosts	125
8.2.3	Core warehouse HA events monitored by the system console	127
8.3	Management host availability	128
8.3.1	Management host failover events that are monitored by the system console	131
8.4	High availability management	132
8.4.1	High availability toolkit	132
8.4.2	Starting and stopping resources with the HA toolkit	133
8.4.3	Monitoring the status of the core warehouse HA configuration	134
8.4.4	Monitoring the status of the management host HA configuration	137
8.4.5	Moving database partition resources to the standby node as a planned failover	140
8.4.6	Moving resources to the standby management host as a planned failover	141

<b>Chapter 9. Workload management</b>	145
9.1 Introducing DB2 workload management	146
9.1.1 Defining business goals	146
9.1.2 Identifying work that enters the data server	147
9.1.3 Managing work in progress	147
9.1.4 Monitoring to ensure that the data server is being used efficiently	148
9.2 Workload management administrator authority	149
9.3 Workload management concepts	150
9.3.1 Workloads	150
9.3.2 Work classes and work class sets	150
9.3.3 Service classes	151
9.3.4 Thresholds	151
9.3.5 Threshold actions	151
9.3.6 Work actions and work action sets	152
9.3.7 Histograms and histogram templates	152
9.4 Configuring DB2 workload management in stages	153
9.4.1 Default workload management (Stage 0)	153
9.4.2 Untuned workload management (Stage 1)	154
9.4.3 Tuned workload management (Stage 2)	154
9.4.4 Advanced workload management (Stage 3)	155
9.4.5 Moving from Stage 0 to Stage 2	155
9.5 Workload management dispatcher	156
9.6 Default query concurrency management	158
9.6.1 Default workload management objects for concurrency control	159
 <b>Chapter 10. Mining and unstructured text analytics</b>	 163
10.1 Overview	164
10.2 Business goals	164
10.3 Business scenarios	165
10.3.1 In data mining	165
10.3.2 In text analytics and unstructured analysis	166
10.4 Data mining techniques and process	167
10.4.1 Data mining techniques	168
10.4.2 Data mining process: preparing, building, and deploying	169
10.5 Data mining and text analytics in DB2 Warehouse	170
10.5.1 Data mining	172
10.5.2 Unstructured analysis	174
 <b>Chapter 11. Providing the analytics</b>	 177
11.1 Implementing OLAP	178
11.1.1 Dimensional model	178
11.1.2 Providing OLAP data	179
11.1.3 Using OLAP data	181



11.1.4 Pulling it all together . . . . .	181
11.2 Cognos and the cube model . . . . .	182
11.2.1 Cognos architecture . . . . .	182
11.2.2 Cognos metadata and Cognos Cube Designer . . . . .	183
11.2.3 Dimensional metadata and dynamic cubes . . . . .	184
11.3 Dynamic Cubes architecture and its lifecycle . . . . .	184
11.3.1 Design and model phase . . . . .	185
11.3.2 Deploy phase . . . . .	186
11.3.3 Run phase . . . . .	186
11.3.4 Optimization . . . . .	186
11.3.5 Cognos Dynamic Cubes caching . . . . .	188
11.3.6 Cognos Dynamic Cubes and the data warehouse . . . . .	191
11.3.7 Cognos Dynamic Cubes and DB2 with BLU Acceleration . . . . .	192
11.4 Resources . . . . .	193
<b>Related publications . . . . .</b>	<b>195</b>
IBM Redbooks . . . . .	195
Other publications . . . . .	195
Online resources . . . . .	196
Help from IBM . . . . .	196



# 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 grant you any license to these patents. You can send license inquiries, in writing, to:

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

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

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

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

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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. 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 IBM's application programming interfaces.

# 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®	InfoSphere®	Redbooks®
Cognos®	Intelligent Miner®	Redbooks (logo)  ®
DataStage®	Optim™	solidDB®
DB2®	POWER®	SPSS®
DB2 Connect™	PureData™	System z®
DB2 Extenders™	pureQuery®	Tivoli®
GPFS™	pureScale®	WebSphere®
IBM®	PureSystems™	z/OS®
IBM PureData™	pureXML®	
Informix®	Rational®	

The following terms are trademarks of other companies:

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

Building on the business intelligence (BI) framework and capabilities that are outlined in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813, this IBM® Redbooks® publication focuses on the new business insight challenges that have arisen in the last few years and the new technologies in IBM DB2® 10 for Linux, UNIX, and Windows that provide powerful analytic capabilities to meet those challenges.

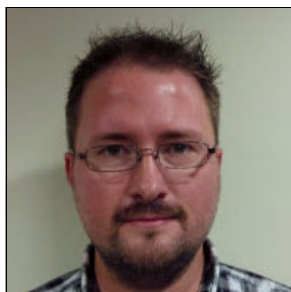
This book is organized in to two parts. The first part provides an overview of data warehouse infrastructure and DB2 Warehouse, and outlines the planning and design process for building your data warehouse. The second part covers the major technologies that are available in DB2 10 for Linux, UNIX, and Windows. We focus on functions that help you get the most value and performance from your data warehouse. These technologies include database partitioning, intrapartition parallelism, compression, multidimensional clustering, range (table) partitioning, data movement utilities, database monitoring interfaces, infrastructures for high availability, DB2 workload management, data mining, and relational OLAP capabilities. A chapter on BLU Acceleration gives you all of the details about this exciting DB2 10.5 innovation that simplifies and speeds up reporting and analytics. Easy to set up and self-optimizing, BLU Acceleration eliminates the need for indexes, aggregates, or time-consuming database tuning to achieve top performance and storage efficiency. No SQL or schema changes are required to take advantage of this breakthrough technology.

This book is primarily intended for use by IBM employees, IBM clients, and IBM Business Partners.

## Authors

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

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



**Scott Andrus** is the development manager for IBM PureData™ for Operational Analytics and IBM Smart Analytics Systems at the IBM Toronto Laboratory in Canada. He focuses on expert integrated systems and ensures that these products deliver the best performance and customer experience possible. Scott works closely with customers and helps them develop and implement key systems to support critical business objectives. He has more than 15 years of experience in software development, architecture, and support.



**Bhuvana Balaji** is a Software Engineering Manager and IBM SWG Information Management Division Technical Quality Champion at IBM Silicon Valley Laboratory. She has more than 15 years experience in software development, technical support, and quality assurance roles. Bhuvana holds a dual Master's degree in Mathematics from the Indian Institute of Technology, Madras, India, and University of Cincinnati, OH, US.



**Enzo Cialini** is a Senior Technical Staff Member in the IBM Information Management Software division at the IBM Toronto Laboratory. He is the Chief Quality Assurance Architect and is responsible for the technical test strategy for DB2 for Linux, UNIX, and Windows, PureData, and Customer Operational Profiling. Enzo has more than 20 years experience in software development, testing, and support. He is actively engaged with customers and the field. He co-authored *The High Availability Guide for DB2* and various papers.



**Michael Kwok** is the senior manager for the DB2 Warehouse and BLU Performance team at the IBM Toronto Lab in Canada. He focuses on database performance in the data warehouse and business analytic space, and ensures that DB2 products continue to deliver the best performance possible. He also works extensively with customers and provides technical consultation. Michael Kwok holds a Ph.D. degree from the University of Waterloo, Canada, with his doctoral dissertation in the area of scalability analysis of distributed systems.



**Roman B. Melnyk** Ph.D., is a senior member of the DB2 Information Development team. Roman co-authored *Hadoop for Dummies* (Wiley, in preparation), *DB2 Version 8: The Official Guide* (Prentice Hall Professional Technical Reference, 2003), *DB2: The Complete Reference* (Osborne/McGraw-Hill, 2001), *DB2 Fundamentals Certification for Dummies* (Hungry Minds, 2001), and *DB2 for Dummies* (IDG Books, 2000). Roman also edited *DB2 10.5 with BLU Acceleration: New Dynamic In-Memory Analytics for the Era of Big Data* (McGraw-Hill, 2013), *Harness the Power of Big Data: The IBM Big Data Platform* (McGraw-Hill, 2013), *Warp Speed, Time Travel, Big Data, and More: DB2 10 for Linux, UNIX, and Windows New Features* (McGraw-Hill, 2012), and *Apache Derby - Off to the Races* (Pearson Education, 2006).



**Jessica Rockwood** is the senior manager for the DB2 Performance Benchmarks team at the IBM Toronto Lab in Canada. Before taking on the benchmark team, her most recent focus was on BLU Acceleration, working to enable other IBM products to leverage the performance enhancements for analytic workloads and building a knowledge base for the technical sales organization. Jessica has also had a wide and varied set of roles in the past, including customer proof of concepts (POCs) and performance benchmarks of warehouse environments, DB2 pureScale® specialist, DB2 HADR specialist, and a long history in autonomies and DB2 administration and monitoring.

## Acknowledgements

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

Chris Eaton, Sam Lightstone, Berni Schiefer, Jason Shayer, Les King  
**IBM Toronto Laboratory, Canada**

Thanks to the authors of the previous editions of this book.

Authors of *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813, published by June 2010, were:

Chuck Ballard  
Nicole Harris  
Andrew Lawrence  
Meridee Lowry  
Andy Perkins  
Sundari Voruganti



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

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

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

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

## Comments welcome

Your comments are important to us!

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

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

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

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- Find us on Facebook:

<http://www.facebook.com/IBMRedbooks>

- Follow us on Twitter:

<http://twitter.com/ibmredbooks>

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



# Part 1

## Overview

The increasing focus on “speed of thought” analytics as a critical success factor for organizations underscores the urgent need for a cost-effective business intelligence framework that is easy to implement, ease to manage, and provides outstanding performance. The IBM DB2 10 for Linux, UNIX, and Windows data warehouse offerings provide such a framework with all of these characteristics and more.

The first phase of developing a data warehouse solution is to understand your requirements and how they can be addressed by the warehouse offering. The chapters in this section provide an overview of data warehouse infrastructure, DB2 Warehouse, and the planning and design process.

Chapter 1, “Gaining business insight with IBM DB2” on page 3 covers the warehouse capabilities that you want when you build a business intelligence (BI) architectural framework to meet the business challenges of today, including the great opportunities around big data.

Chapter 2, “Technical overview of IBM DB2 Warehouse” on page 9 provides an overview of the IBM DB2 for Linux, UNIX, and Windows data warehouse offerings, including solution components, architecture, and product licensing.

Chapter 3, “Warehouse development lifecycle” on page 29 outlines the planning and design process for building your data warehouse, which includes defining business requirements and building a data model.



# Gaining business insight with IBM DB2

With the continued growth and focus on Business Analytics as a critical success factor for any organization, there is a need for a business intelligence (BI) framework that provides ease of adoption, ease of management, and most importantly, optimal performance. This chapter describes the capabilities that are required for building a BI architectural framework to meet the business needs of today.

Building on the BI framework and capabilities that are outlined in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813, this book focuses on the new business insight challenges that have arisen in the last few years and new technologies in DB2 10 for Linux, UNIX, and Windows that provide powerful analytic capabilities to meet those challenges.

There are several aspects to understand that are changing the requirements for BI architectures: the amount of data, the speed with which that data is generated and needs to be analyzed, and the need for more dynamic analysis of that data. Gone are the days when generating an operational report meant taking a coffee break. Today's definition of success is *speed of thought* analytics.

## 1.1 Current business challenges

To succeed in business today, companies are focusing on their bottom line and turning a profit. In general, businesses are willing to spend money if it makes them money. The challenge is in knowing where to invest, what opportunities to pursue, and how to ensure a high return on investment (ROI).

Making an informed decision comes down to having the correct data and analyzing it effectively, that is, business intelligence and analytics. With good business intelligence, organizations can be efficient, agile, and informed risk takers. Organizations can streamline operations, looking for any areas for cost savings. They can identify and respond to business trends quickly, whether it is for understanding customer behavior or understanding key business metrics to realize new opportunities. Just as importantly, organizations can leverage business intelligence to predict future performance and identify new opportunities for a product or the potential cross-sell or up-sell of existing products.

Historically, Business Analytics was “nice-to-have” for organizations as the analysis of data was a lengthy process and therefore tended to be a historical view of what happened. Now, it is a must-have to differentiate an organization from competitors. Studies show that organizations competing on analytics are 2.2 times more likely to substantially outperform their industry peers.<sup>1</sup>

To fully leverage this competitive advantage, there is a renewed focus on combining real-time data with historical data to allow executives and front-line employees to make informed decisions more quickly and with a higher degree of confidence. Instead of basing decisions on what was, today's Business Analytics can provide a picture of what was, what is, and what might be.

In addition to the challenge of analyzing in real time, the amount of data to be analyzed is growing exponentially. The volume and speed with which data is being generated is pushing the bounds of computing capacity and driving business intelligence architectures to handle larger data volume and answer more business questions at a faster rate than ever before. On top of it all, the challenge for business intelligence providers is to meet those requirements all while leveraging the existing investment in the hardware infrastructure that is used to support this data and analysis.

---

<sup>1</sup> *Analytics: The new path to value*, found at:  
[http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=PM&subtype=XB&appname=GBSE\\_GB\\_TI\\_USEN&htmlfid=GBE03371USEN&attachment=GBE03371USEN.PDF](http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=PM&subtype=XB&appname=GBSE_GB_TI_USEN&htmlfid=GBE03371USEN&attachment=GBE03371USEN.PDF)

## 1.2 Big data and the data warehouse

In the last several years, the newest buzz term in the context of Business Analytics is *big data*. Big data is a term that is used to refer to the exponential growth of data created in recent years, which is qualified on the four dimensions of volume, velocity, variety, and veracity. Even more than just the volume of information, big data is also about making your business more agile, answering questions that might not have been previously considered, and finding insights in new and emerging types of data and content.

The future of the data warehouse in the context of big data has been questioned, but when you look at IBM Big Data Platform, it is easy to see the important role data warehouses will continue to play in the future. In Figure 1-1, the data warehouse is shown as a key component of the data access and storage layer.

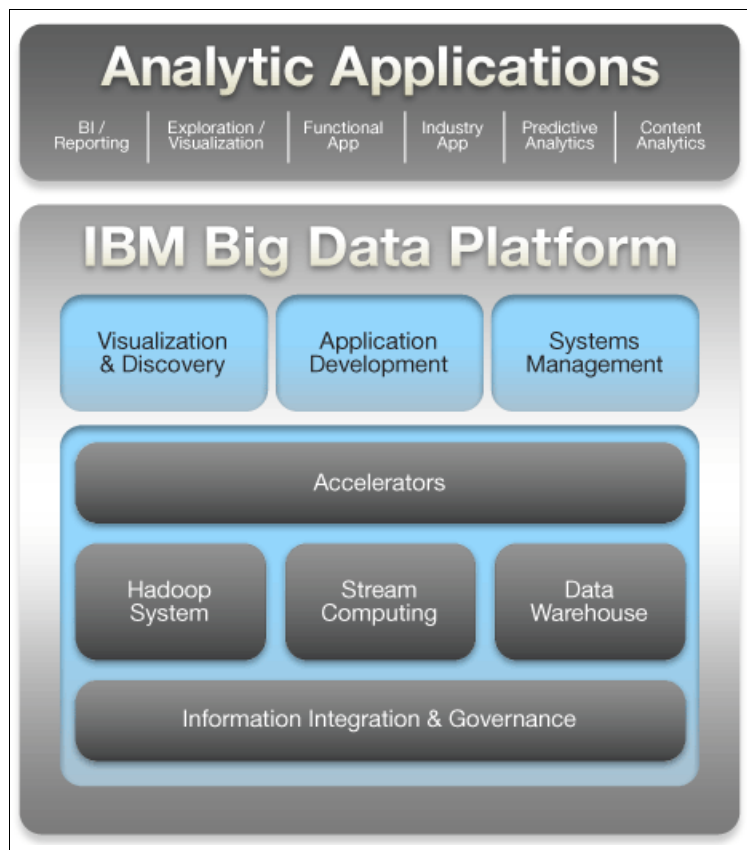


Figure 1-1 IBM Big Data Platform<sup>2</sup>

<sup>2</sup> From <http://www.ibm.com/software/data/bigdata>

A big data platform must support both traditional analytics (on structure data from traditional sources) and a new style of exploratory analytics on unstructured data. Data warehouses provide support for traditional analytics, running deep analytic queries on huge volumes of structured data with massive parallel processing.

### 1.2.1 Data warehouse infrastructure

The critical role of the data warehouse is to provide a central repository of structured data, which is created by integrating data from one or more disparate sources. A single version, or view, of the business is implemented as a single, scalable, and consolidated data warehouse. After a single version of the data is consolidated in the warehouse, it is available for analysis by all levels of the business. Additionally, the warehouse contents can be subdivided into data marts that are specific to particular business units, as shown in Figure 1-2.

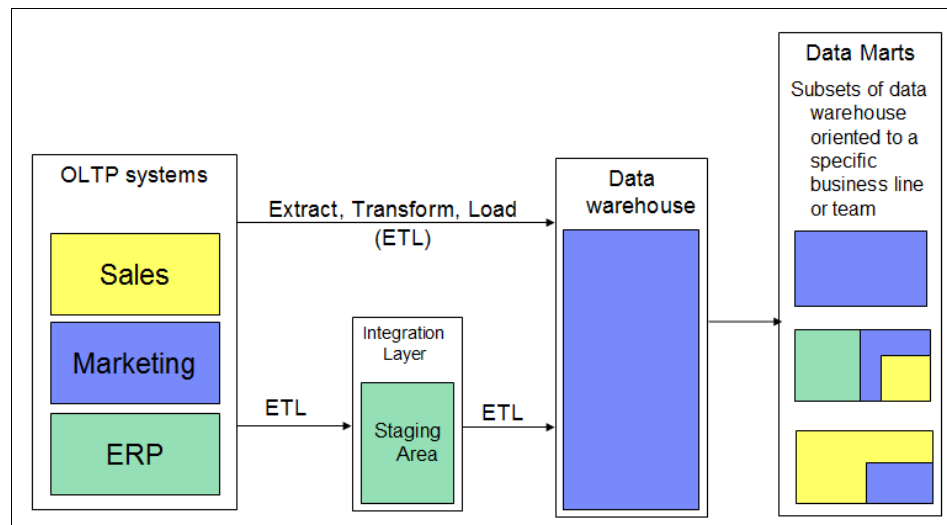


Figure 1-2 Data warehouse architecture

To support the workloads of today, the data warehouse architecture must optimize both traditional deep analytic queries and shorter transactional type queries. There must be capacity for real-time loading and updating of warehouse data and increased storage capacity to cope with the data growth explosion. Furthermore, a shortened time to build and deploy a warehouse leads to a reduced time to value and overall increased ROI.



## 1.3 High performance warehouse with DB2 for Linux, UNIX, and Windows

DB2 for Linux, UNIX, and Windows provides all the capabilities that are required to build a high performance warehouse for whatever business challenges exist. The business requirements and data drive the choice of technologies that are outlined in the following chapters.

The components of the DB2 Business Intelligence Solution Framework, which are shown in Figure 1-3, are little changed from what is outlined in Chapter 1, “Gaining business insight with IBM InfoSphere® Warehouse”, in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813.

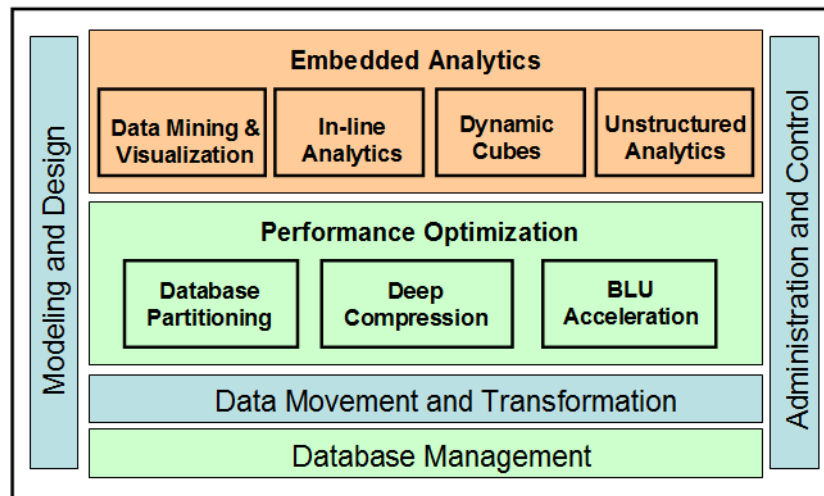


Figure 1-3 DB2 warehouse architecture

The chapters that follow focus on two primary areas: embedded analytics and performance optimization. In embedded analytics, this past year brought the addition of IBM Cognos® Business Intelligence Dynamic Cubes to the architecture. Chapter 11, “Providing the analytics” on page 177 focuses on how to take advantage of the in-memory acceleration that is provided by Dynamic Cubes to provide speed of thought analytics over terabytes of enterprise data.

In performance optimization, there are two main areas of discussion. The first is the introduction of BLU Acceleration to DB2. This new technology, introduced in DB2 10.5 for Linux, UNIX, and Windows, is in-memory optimized columnar processing that optimizes memory, processor, and I/O for analytic queries. The second area of focus is the performance advancements for both scale up and scale out options for standard row-based tables.

Data movement and transformation, modeling and design, and administration and control are reviewed, but primarily from a focus of what is new with DB2 10. For a complete examination of these components, see *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813.



# Technical overview of IBM DB2 Warehouse

IBM DB2 Advance Enterprise Server Edition represents the IBM framework for implementing an expert-integrated system for business intelligence (BI). IBM DB2 Advance Edition integrates a tools and runtime infrastructure for Online Analytical Processing (OLAP) and data mining, and the delivery of embedded Inline analytics on a common platform.

IBM DB2 Advance Edition removes cost and complexity barriers, and enables the delivery of powerful analytics to an enterprise. The integrated tools enable a low total cost of ownership (TCO) and provide improved time-to-value for developing and delivering analytics enterprise-wide.

Furthering the data warehousing proposition, IBM developed and produced the concept of an expert-integrated system. This expert-integrated system transforms your typical static data warehouse from a repository, primarily used for batch reporting, into an active end to end solution, use IBM preferred practices.

This chapter provides an overview of IBM DB2 Data Warehouse offerings and explores solution components and architecture, and reviews product licensing.

**Note:** All the functions, features, and operators for IBM DB2 Advance Enterprise Server Edition on the Linux, UNIX, and Windows platforms are not the same as on the IBM System z® platform. We try to point out the differences where appropriate in each chapter.

## 2.1 DB2 Warehouse solutions

IBM DB2 Advance Edition contains a set of components that combines the DB2 database engine with powerful Business Analytics tools. It provides a comprehensive Business Analytics platform enabling you to design, develop, and deploy analytical applications in your enterprise.

IBM DB2 Advance Edition can be used to build a complete data warehousing solution that includes a highly scalable relational database engine, data access capabilities, Business Analytics, and user analysis tools. It integrates core components for data warehouse administration, data mining, OLAP, inline analytics, reporting, and workload management.

### 2.1.1 The component groups

IBM DB2 Advance Enterprise Server Edition has a component-based architecture, complete with client and server components. Those components are arranged into three logical component groups, as shown in Figure 2-1 on page 11. These component groups are typically installed on three different computers, but can be installed on one or two, because they can operate in a multitier configuration.

The component groups are as follows:

- ▶ Data server components  
This category includes DB2 and Workload Manager.
- ▶ Application server components  
This category includes IBM WebSphere® Application Server, Administration Console, and the Cubing Services cube server.
- ▶ Client components  
The client components are Design Studio, IBM Data Server Client, Intelligent Miner® Visualizer, and the Cubing Services client.

In addition to these component products, the IBM DB2 Advance Enterprise Server Edition documentation and tutorials can also be installed in any of the logical component groups.

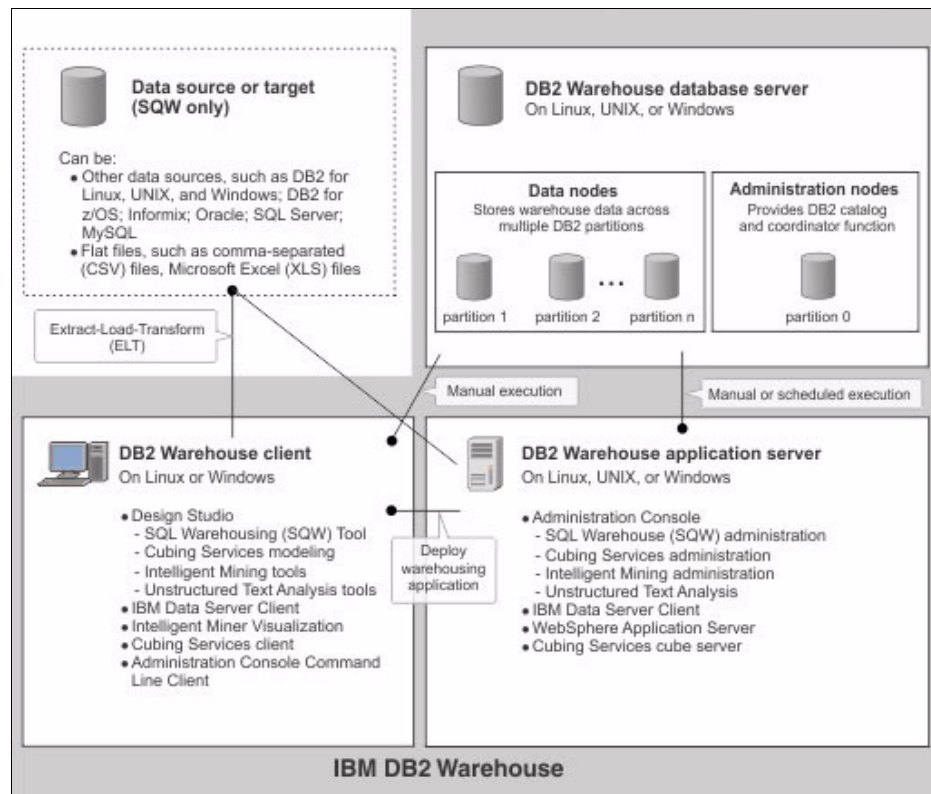


Figure 2-1 Logical component groups in IBM DB2 Advance Enterprise Server Edition

The highly scalable and robust DB2 database servers are the foundation of DB2 Warehouse. All the design and development is performed by using Design Studio, which is an Eclipse-based development environment. The IBM WebSphere Application Server is used as the runtime environment for deployment and management of all DB2 Warehouse based applications.

## 2.1.2 Editions

IBM DB2 for Linux, UNIX, and Windows 10.5 offers various editions and features that provide customers with choices based on business need. For data warehouse usage, you can choose from Advance Enterprise Server Edition, Advance Workgroup Edition, and Developer Edition.

Figure 2-2 shows the DB2 10.5 editions.



Figure 2-2 DB2 10.5 editions

## IBM DB2 Advanced Enterprise Server Edition for Linux, UNIX, and Windows 10.5

DB2 Advanced Enterprise Server Edition includes the functions and tools that are needed for large, complex enterprise environments, and is ideal for both transactional and data warehouse workloads. The edition combines all the functions from the DB2 Enterprise Server Edition with full data compression functionality, including the following features:

- ▶ BLU Acceleration
- ▶ IBM pureScale clustering
- ▶ Workload management
- ▶ Distributed partitioning functionality (DPF)
- ▶ Federation between DB2 for Linux, UNIX, and Windows databases and databases from other vendors, including Oracle database and SQL Server
- ▶ Queue-based replication (Q Replication) between three DB2 for Linux, UNIX, and Windows systems

- ▶ Change Data Capture (CDC) replication between a DB2 for Linux, UNIX, and Windows source and up to two DB2 for Linux, UNIX, and Windows 10.5 targets for HADR
- ▶ IBM solidDB® and solidDB Universal Cache
- ▶ Cognos Business Intelligence
- ▶ Warehouse model packs
- ▶ A rich set of tools:
  - IBM Data Studio
  - IBM InfoSphere Optim™ Performance Manager Extended Edition
  - IBM InfoSphere Optim Query Workload Tuner
  - IBM InfoSphere Optim Configuration Manager
  - IBM InfoSphere Data Architect (ten authorized users)
  - IBM InfoSphere Optim pureQuery® Runtime for Linux, UNIX, and Windows

The six tools that are listed above can be used with any supported version of DB2 Advanced Enterprise Server Edition. Similarly, any of these tools that were included with prior versions of DB2 Advanced Enterprise Server Edition can be used with DB2 Advanced Enterprise Server Edition 10.5, if supported.

DB2 Advanced Enterprise Server Edition can be deployed on Linux, UNIX, or Windows servers of any size, from one processor to hundreds of processors, and on both physical and virtual servers.

DB2 10.5 Advanced Enterprise Server Edition is available on a Processor Value Unit (PVU), per Authorized User Single Install (AUSI), or per Terabyte charge metric. Under the AUSI charge metric, you must acquire a minimum of 25 AUSI licenses per 100 PVUs. Under the per Terabyte charge metric, you must meet the following requirements:

- ▶ Use the database for a data warehouse and run a data warehouse workload. A data warehouse is a subject-oriented, integrated, and time-variant collection of data that integrates data from multiple data sources for historical ad hoc data reporting and querying. A data warehouse workload is a workload that typically scans thousands or millions of rows in a single query to support the analysis of data in a data warehouse.
- ▶ Have user data from a single database spread across two or more active data partitions or at least 75% of the user data in BLU Acceleration column-organized tables, with most of the workload accessing the BLU Acceleration column-organized tables.
- ▶ Not use the DB2 for Linux, UNIX, and Windows built-in high availability disaster recovery (HADR) or pureScale capabilities.

You can upgrade from DB2 Enterprise Server Edition to DB2 Advanced Enterprise Server Edition by using a Processor Value Unit (PVU) based upgrade part number.

## **IBM DB2 Enterprise Server Edition for Linux, UNIX, and Windows 10.5**

DB2 Enterprise Server Edition is designed to meet the data server needs of medium- to large-size businesses and is ideal for transactional and mixed workloads. It can be deployed on Linux, UNIX, or Windows servers of any size, from one processor to hundreds of processors, and on both physical and virtual servers. The edition includes the following features:

- ▶ IBM Data Studio
- ▶ Label Based Access Control
- ▶ Row and Column Access Control
- ▶ Multi-Temperature Data Management
- ▶ Time Travel Query, and Table Partitioning
- ▶ HADR
- ▶ Online Reorganization, Materialized Query Tables (MQTs)
- ▶ Multidimensional Clustering (MDC)
- ▶ Query Parallelism
- ▶ Scan Sharing
- ▶ Connection Concentrator
- ▶ IBM pureXML®
- ▶ Backup compression
- ▶ IBM Tivoli® System Automation for Multiplatforms for use with high availability

The product also includes DB2 Homogeneous Federation and Homogeneous SQL Replication, allowing federated data access and replication between DB2 for Linux, UNIX, and Windows servers, and web services federation.

DB2 10.5 Enterprise Server Edition is available on a PVU or per Authorized User Single Install (AUSI) charge metric. Under the AUSI charge metric, you must acquire a minimum of 25 AUSI licenses per 100 PVUs.

## **IBM DB2 Advanced Workgroup Server Edition for Linux, UNIX, and Windows 10.5**

DB2 Advanced Workgroup Server Edition includes the capabilities that are needed for medium-sized business environments, and is ideal for both transactional and data warehouse workloads. The edition includes the same functions and tools as DB2 Advanced Enterprise Server Edition. The main difference is that DB2 Advanced Workgroup Server Edition has processor core, socket, memory, and terabyte restrictions.



In addition, DB2 Advanced Workgroup Server Edition supports federation only with DB2 for Linux, UNIX, and Windows and IBM Informix® data sources.

DB2 10.5 Advanced Workgroup Server Edition is available on a PVU, per Authorized User Single Install (AUSI), or per Terabyte charge metric. Under the AUSI charge metric, you must acquire a minimum of 25 AUSI licenses per 100 PVUs. Under the PVU and AUSI charge metrics, you are restricted to 16 processor cores and 128 GB of instance memory. Under the per Terabyte charge metric, you are restricted to four processor sockets. These restrictions are per physical or, where partitioned, virtual server, except in a pureScale or DPF cluster where the restrictions apply to the entire cluster. Under all charge metrics, you are restricted to 15 TB of user data per database. Under the Terabyte charge metric, you must also meet the following requirements:

- ▶ Use the database for a data warehouse and run a data warehouse workload. A data warehouse is a subject-oriented, integrated, and time-variant collection of data that integrates data from multiple data sources for historical ad hoc data reporting and querying. A data warehouse workload is a workload that typically scans thousands or millions of rows in a single query to support the analysis of data in a data warehouse.
- ▶ Have user data from a single database spread across two or more active data partitions or at least 75% of the user data in BLU Acceleration column-organized tables with most of the workload accessing the BLU Acceleration column-organized tables.
- ▶ Not use DB2 for Linux, UNIX, and Windows built-in HADR or pureScale capabilities.

## **IBM DB2 Workgroup Server Edition for Linux, UNIX, and Windows 10.5**

DB2 Workgroup Server Edition is suitable for transactional database workloads in a departmental, workgroup, or medium-sized business environment. The edition shares functions with DB2 Enterprise Server Edition, but has the processor core and memory restrictions that are described below. DB2 Workgroup Server Edition can be deployed in Linux, UNIX, and Windows server environments. If using multiple virtual servers on a physical server, there is no limit on the cores or memory that is available to the physical server if the processor and memory restrictions are observed by the virtual servers running DB2. This makes DB2 Workgroup Server Edition ideal for consolidating multiple workloads onto a large physical server running DB2 Workgroup Server Edition in multiple virtual servers.

DB2 10.5 Workgroup Server Edition is available on a per Limited Use Socket, PVU, or Authorized User Single Install (AUSI) charge metric. Under the AUSI metric, you must acquire a minimum of five AUSI licenses per installation. Under all metrics, you are restricted to 16 processor cores and 128 GB of instance memory and 15 TB of user data per database. In addition, under the Socket metric, you are also restricted to no more than four processor sockets. These restrictions are per physical or, where partitioned, virtual server.

## **IBM DB2 Express Server Edition for Linux, UNIX, and Windows 10.5**

DB2 Express Server Edition is a full-function transactional data server, which provides attractive entry-level pricing for the small and medium business (SMB) market. It comes with simplified packaging and is easy to transparently install within an application. With DB2 Express Server Edition, it is easy to then upgrade to the other editions of DB2 10.5 because DB2 Express Server Edition includes most of the same features, including security and HADR, as the more scalable editions. DB2 Express Server Edition can be deployed in x64 server environments and is restricted to eight processor cores and 64 GB of memory per physical or, where partitioned, virtual server. If using multiple virtual servers on a physical server, there is no limit on the cores or memory that are available to the physical server if the processor and memory restrictions are observed by the virtual servers running DB2. This makes DB2 Express Server Edition ideal for consolidating multiple workloads onto a large physical server running DB2 Express Server Edition in multiple virtual servers.

DB2 10.5 Express Server Edition is available on a per Authorized User Single Install (AUSI), PVU, Limited Use Virtual Server, or 12-month Fix Term licensing model. If licensed under the AUSI metric, you must acquire a minimum of five AUSI licenses per installation. DB2 Express Server Edition can also be licensed on a yearly Fixed Term License pricing model. Under all charge metrics, you are restricted to 15 TB of user data per database.

## **IBM DB2 Express-C Edition for Linux and Windows 10.5**

DB2 10.5 Express-C is a no-charge, entry-level edition of the DB2 data server for the developer and partner community. It is designed to be up and running in minutes, and easy to use and embed. It includes self-management features, and embodies all of the core capabilities of DB2 for Linux, UNIX, and Windows, such as Time Travel Query. One notable difference between DB2 Express-C and the production editions of DB2 for Linux, UNIX, and Windows is that you cannot cluster together servers for the purposes of high availability. Solutions that are developed by using DB2 Express-C can be seamlessly deployed by using more scalable DB2 editions without modifications to the application code.

DB2 10.5 Express-C can be used for development and deployment at no charge. It can be installed on x64-based physical or virtual systems and may use up to a maximum of two processor cores and 16 GB of memory with no more than 15 TB of user data per database. DB2 Express-C comes with online community-based assistance. Users requiring more formal support, access to fix packs, or more capabilities such as high availability, Homogeneous Federation, and replication, can purchase an optional yearly subscription for DB2 Express Server Edition (Fixed Term License) or upgrade to other DB2 editions. In addition, IBM Data Studio is also available to facilitate solutions deployment and management.

### **IBM DB2 Developer Edition for Linux, UNIX, and Windows 10.5**

IBM DB2 Developer Edition offers a package for a single application developer to design, build, test, and prototype applications for deployment on any of the IBM DB2 client or server platforms. This comprehensive developer offering includes DB2 Workgroup Server Edition, DB2 Advanced Workgroup Server Edition, DB2 Enterprise Server Edition, DB2 Advanced Enterprise Server Edition, IBM DB2 Connect™ Enterprise Edition, and all of the built-in DB2 10.5 capabilities, allowing you to build solutions that use the latest data server technologies.

The software in this package cannot be used for production systems. You must acquire a separate Authorized User license for each unique person who is given access to the program.

## **2.2 Expert systems**

The time has come for a new breed of systems. Expert-integrated systems are systems with integrated expertise that combine the flexibility of a general-purpose system, the elasticity of cloud, and the simplicity of an appliance that is tuned to the workload. IBM PureData systems can help IBM clients achieve greater simplicity, speed, and lower cost in their IT environments. Expert-integrated systems reduce the time, cost, and risk of custom designed systems by fully integrating, tuning, and providing a management and maintenance framework ready to use. Expert-integrated systems fundamentally change both the experience and economics of IT.

Expert-integrated systems are more than a static stack of self-tuned components: a server here, some database software there, serving a fixed application at the top. Instead, these systems have three truly unique attributes:

- ▶ **Built-in expertise:** Think of expert integrated systems as representing the collective knowledge of thousands of deployments, established preferred practices, innovative thinking, IT industry leadership, and the distilled expertise of business partners and solution providers, captured into the system in a deployable form from the base system infrastructure through the application.
- ▶ **Integrated by design:** All the hardware and software components are deeply integrated and tuned in the lab and packaged in the factory into a single ready-to-go system that is optimized for the workloads it is running. All of the integration is done for you, by experts.
- ▶ **Simpler experience:** The entire experience is much simpler, from the moment you start designing what you need to the time you purchase, to setting up the system, and to operating, maintaining, and upgrading it over time. Management of the entire system of physical and virtual resources are integrated.

IBM offers a PureData system for both transactional and analytics workloads. IBM PureData System for Transactions is a highly reliable and scalable database platform that helps reduce complexity, accelerate time to value, and lower ongoing data management costs. The system enables IT departments to easily deploy, optimize, and manage transactional database workloads. IBM PureData System for Operational Analytics is an expert-integrated data system that is designed and optimized specifically for the demands of an operational analytics workload. The system is a complete, ready for immediate use solution for operational analytics that provides both the simplicity of an appliance and the flexibility of a custom solution. Designed to handle 1000+ concurrent operational queries, it delivers mission-critical reliability, scalability, and outstanding performance. These expert-integrated systems set a new standard in workload-optimized systems.

### **2.2.1 PureData for operational analytics**

IBM is making a strategic investment developing IBM PureSystems™ to deliver innovations that do the following tasks:

- ▶ Simplify system deployment, management, and maintenance.
- ▶ Deliver greater system performance, scalability, and efficiency through deep integration.

- ▶ Accelerate the benefits of cloud computing.
- ▶ Integrate and automate preferred practices through patterns of expertise that are delivered from IBM, solution partners, and captured by clients themselves.

This is why IBM has introduced the PureData System. This system is optimized exclusively for delivering data services for transactional applications and for analytics. It provides an integrated data platform: database and data warehouse management, integrated system management, integrated compute, storage, and networking resources. More than just a bundle of components, the PureData system also builds expertise into models that optimize performance and simplicity for specific types of data workloads.

The PureData system also simplifies the entire lifecycle. It is factory integrated to be data load ready in hours, and offers integrated management and support.

Having an integrated system enables delivery of integrated and automated maintenance, which eliminates manual steps that cost time and increase the risk of human error.

## **2.2.2 DB2 Warehouse components**

The components of DB2 Warehouse provide an integrated platform for warehouse administration and for the development of warehouse-based analytics. The IBM DB2 Warehouse components include the following ones:

- ▶ DB2 Warehouse Design Studio
- ▶ DB2 Warehouse Administration Console
- ▶ Common Configuration
- ▶ SQL Warehousing
- ▶ Cubing Services
- ▶ Mining
- ▶ Cognos Business Intelligence for Reporting
- ▶ IBM Optim Performance Manager Extended Edition
- ▶ Text Analysis

The categories and components of the architecture are shown in Figure 2-3, and described in the remainder of this section.

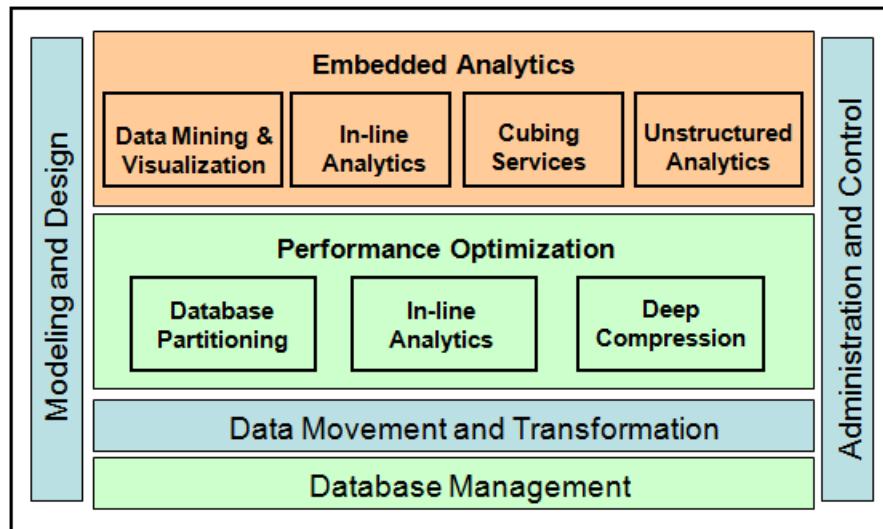


Figure 2-3 IBM DB2 Advance Enterprise Server Edition functional component architecture

## DB2 Warehouse Design Studio

The Design Studio provides a common design environment for creating physical data models, OLAP cubes, SQL data flows, and control flows. The Design Studio is built on the Eclipse Workbench, which is a development environment that you can customize.

## DB2 Warehouse Administration Console

The DB2 Warehouse Administration Console is a web application from which you can deploy and manage applications, control flows, database resources, and system resources. You can do the following types of tasks in the Administration Console:

- ▶ Common configuration
- ▶ SQL warehousing
- ▶ Cubing Services
- ▶ Mining

### ***Common configuration***

This component allows you to create and manage database and system resources, including driver definitions, log files, and notifications.

## ***SQL warehousing***

This component allows you to run and monitor data warehousing applications, view deployment histories, and run statistics.

## ***Cubing Services***

Cubing Services works with business intelligence tools to provide OLAP access to data directly from DB2 Warehouse. Cubing Services includes tools for multidimensional modeling to design OLAP metadata (cubes), an optimization advisor for recommending materialized query tables (MQTs) in DB2, and a cube server for providing multidimensional access to the data. Each of these Cubing Services components is integrated with the DB2 Warehouse user interfaces for design (Design Studio) and administration and maintenance (Administration Console).

The Cubing Services cube server processes multidimensional queries that are expressed in the MDX query language and produces multidimensional results. The cube servers fetch data from DB2 through SQL queries as needed to respond to the MDX queries. The MQTs that are recommended by the optimization advisor are used by the DB2 optimizer, which rewrites incoming SQL queries and routes eligible queries to the appropriate MQT for faster query performance. In addition to these performance enhancements, the cube server includes multiple caching layers for further optimizing the performance of MDX queries.

DB2 Warehouse provides a Cubing Services Client ODBO Provider to allow access to cube data in Microsoft Excel.

## ***Mining***

The mining component allows you to run and monitor data mining applications. DB2 Warehouse includes the following mining features:

- ▶ Intelligent Miner Easy Mining
- ▶ Intelligent Miner Modeling
- ▶ Intelligent Miner Scoring
- ▶ Intelligent Miner Visualization
- ▶ Text Analysis

These features provide rapid enablement of data mining analysis in Data Warehousing, e-commerce, or traditional Online Transaction Processing (OLTP) application programs. Although previously available as separately priced products, they are now available as an integrated part of DB2 Warehouse.

## **Cognos 10 Business Intelligence for Reporting**

IBM Cognos 10 Business Intelligence for Reporting lets organizations implement a rich set of business intelligence capabilities. It is an ideal solution for

companies that want to consolidate data marts, information silos, and Business Analytics to deliver a single version of the truth to all users. It allows easy creation of reports and quick analysis of data from the data warehouse.

### **IBM Optim Performance Manager Extended Edition**

IBM Optim Performance Manager Extended Edition is a follow on to DB2 Performance Expert. Optim Performance Manager Extended Edition helps optimize the performance and availability of mission-critical databases and applications.

### **Text Analysis**

With DB2 Warehouse, you can create business insight from unstructured information. You can extract information from text columns in your data warehouse and then use the extracted information in reports, multidimensional analysis, or as input for data mining.

## **2.2.3 Database management system**

The database management system is the foundation for the DB2 Warehouse. DB2 offers industry-leading performance, scalability, and reliability on your choice of platform from Linux to IBM z/OS®. It is optimized to deliver industry-leading performance across multiple workloads, and lower administration, storage, development, and server costs.

### **Data movement and transformation**

One of the key operating principles of the DB2 Warehouse is to maintain simplicity. More importantly, the principle is to have integrated tools to design, develop, and deploy data warehouse applications. Use the DB2 Warehouse Design Studio to simplify data transformation by designing and managing SQL warehousing applications. With DB2 Warehouse, the data movement and transformation for intra-warehouse data movement is accomplished by using the SQL Warehousing tool (SQW).

The SQW tool is part of the Eclipse-based Design Studio. SQW enables DBAs to define data transformation and data movement tasks by using the simple and intuitive. The SQW tool generates DB2 specific SQL based on visual operator flows that you model in the Warehouse Design Studio.



## **Work flow for building SQL Warehousing applications**

The work flow for developing a data warehousing application involves two main types of users: architects and administrators. The architects use the Design Studio to design data flows and control flows and prepare the application for deployment. The administrators deploy and manage the application in the Administration Console.

## **Designing data warehousing applications**

You design data warehousing applications with the SQW tool in the Design Studio. Data warehousing applications consist of extract, load, and transform (ELT) operations that are run inside a DB2 database.

## **Administering data warehouse applications**

Use the Administration Console to administer the control flows in data warehouse applications.

## **Performance optimization**

The DB2 Warehouse has several features that can be used to enhance the performance of analytical applications by using data from the data warehouse.

You can use the db2batch Benchmark Tool in DB2 for Linux, UNIX, and Windows to compare your query performance results after you create the recommended summary tables against the benchmarked query performance.

In a data warehouse environment, a huge amount of data exists for creating a cube only or materialized query tables (MQTs), and multidimensional clusters (MDCs) can increase query performance.

For more information about these features, see the following website:

- For DB2 for Linux, UNIX, and Windows:  
<http://www-01.ibm.com/software/data/db2/linux-unix-windows/>
- For System z:  
<http://www-01.ibm.com/software/data/db2/zos/family/>

## **Physical data modeling**

You can create a physical data model in the Data Project Explorer so that you can manage OLAP metadata. Application developers and data warehouse architects use this component to work with physical data models for source and target databases and staging tables.

A physical data model is essentially the metadata representing the physical characteristics of a database. The data model can be developed by using the data model graphical editor or can be reverse-engineered from an existing database. From the physical data model, you can perform the following tasks:

- ▶ Create the physical objects in a database.
- ▶ Compare a data model to the database (and generate only the delta changes).
- ▶ Analyze the model for errors, such as naming violations.
- ▶ Perform impact analysis.

The physical model is also used to provide metadata information to the other functional components of your warehouse, such as the SQW tool. Figure 2-4 on page 25 shows a physical data model that is open in the Data Project Explore that is a graphical editor view of the physical data model, and the properties view of a selected table. It is not necessary to read or understand the data model; the figure is for illustration purposes only.

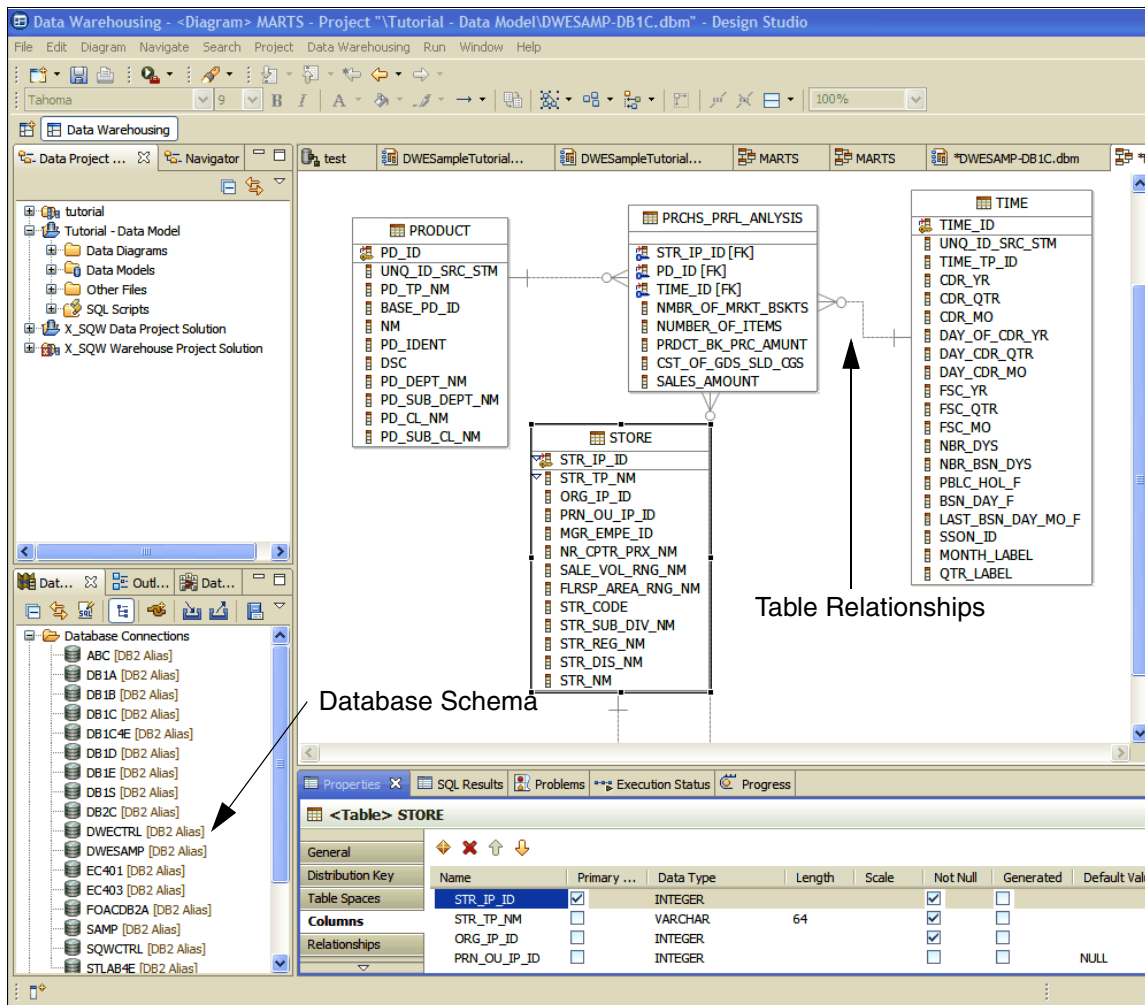


Figure 2-4 Warehouse physical data model

## 2.3 DB2 Warehouse topology

DB2 Warehouse components are grouped into four categories for installation:

- Clients

The client category includes the InfoSphere Warehouse administration client, InfoSphere Warehouse Design Studio, the workload manager, and the data mining visualization programs. The clients are supported on Windows 32-bit and Linux systems.

► Database servers

This category includes the DB2 database server with DPF support plus the database functions to support Intelligent Miner, Cubing Services, and the workload manager. The database server is supported on IBM AIX®, various Linux platforms, Windows Server 2003, and System z.

► Application servers

The application server category includes the WebSphere Application Server, Cognos Business Intelligence server components, and InfoSphere Warehouse Administration Console server components.

► Documentation

This category includes the PDF and online versions of the manuals and can be installed with any of the other categories.

For more information about DB2 Warehouse Installation, see the DB2 information center that is found at the following website:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.qb.server.doc/doc/r0025127.html>

The DB2 Warehouse components can be installed on multiple machines in a number of topologies. Three common topologies are shown in Figure 2-5.

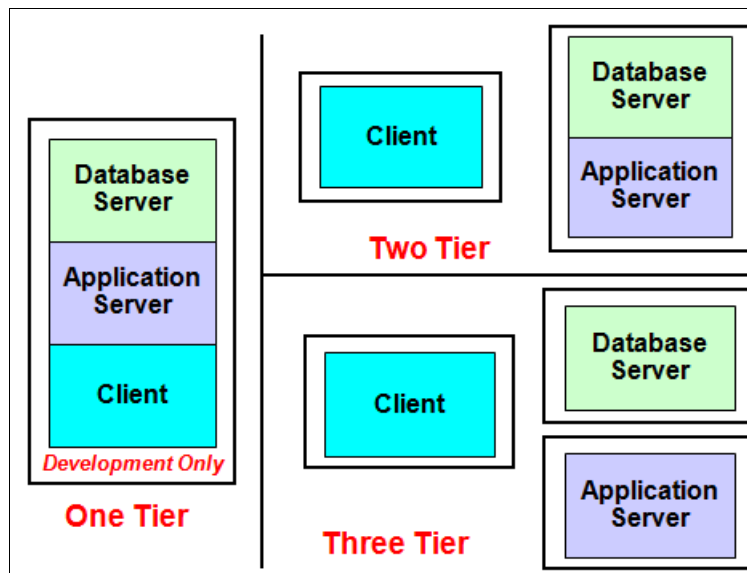


Figure 2-5 Common InfoSphere Warehouse installation topologies

► One tier

In this topology, the InfoSphere Warehouse Client, InfoSphere Warehouse Database Server, and the InfoSphere Warehouse Application Server are all on one system. This topology is used only for development and testing purposes and only on a Windows platform.

► Two tier

In this topology, the InfoSphere Warehouse Database Server and the InfoSphere Warehouse Application Server are on one system with InfoSphere Warehouse Clients on separate systems. This topology can suffice for a test system or for smaller installations. The database and applications can be any of the supported Windows, Linux, or AIX platforms.

► Three tier

In any large installation, the InfoSphere Warehouse Database Server, the InfoSphere Warehouse Application Server, and the InfoSphere Warehouse Clients should all be installed on separate servers. A DB2 client, at a minimum, is required to connect to database servers. It is a preferred practice that a DB2 server be installed for local access to the runtime metadata databases. The application server is supported on AIX, Linux, and Windows Server 2003.





# Warehouse development lifecycle

This chapter describes the planning and design process of building the data warehouse.

The InfoSphere Warehouse Design Studio (Design Studio) provides a platform and a set of integrated tools for developing your data warehouse. You can use these tools to build, populate, and maintain tables and other structures for data mining and Online Analytical Processing (OLAP) analysis.

Design Studio includes the following tools and features:

- ▶ Integrated physical data modeling, which is based on InfoSphere Data Architect
- ▶ SQL Warehousing Tool (SQW) for data flow and control flow design
- ▶ Data mining, exploration, and visualization tools
- ▶ Tools for designing OLAP metadata, MQTs, and cube models
- ▶ Integration points with IBM InfoSphere DataStage® ETL systems

By integrating these tools, Design Studio offers a fast time-to-value and managed cost for warehouse-based analytics. For an introduction to Design Studio, see Chapter 3, “InfoSphere Warehouse Design Studio”, in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813.

## 3.1 Defining business requirements

The first step in the lifecycle of the data warehouse is to define the business requirements. Without understanding the needs of the business, it is impossible to make the correct decisions at the start of the lifecycle to direct the design and implementation of the data warehouse. The data modeler must fully understand the needs of the business user, service level agreements, and data behavior to design an optimal data model and choose the best technology fit for the data warehouse.

To define business requirements, you must answer the following questions:

- ▶ What is the expected query (or report) performance?
- ▶ How many users are going to be using the system at the same time? How many types of activities will those users drive?
- ▶ What is the expected data availability and recoverability?
- ▶ How is the data to be loaded into and pruned from the warehouse, and with what frequency?
- ▶ What initial volume of data is expected? How quickly will data volumes grow? Can the hardware and storage systems support that data volume and growth rate?

Many of these questions might require iterative discussions with all the business stakeholders. Taking the time at the start of the design process ensures that the data modeler and database administrator are equipped with enough knowledge about the expectations for the data warehouse to make effective decisions to optimize the performance and maintenance of the system.



## 3.2 Building the data model

The data infrastructure for any business is built around data models. There are three types of data models, each building on the previous type:

- ▶ Conceptual data model

This is a high-level business view of the data. It identifies the data entities that are used in the business and their relationship to each other, which aids in the understanding and development of the logical and physical data models.

- ▶ Logical data model

This is where the data model is defined and where the data entities are identified in detail along with any existing relationships between them. These are normalized entities that are described by specific attributes and data types, and candidate keys that are used for direct access to data.

It is the logical data model that determines the degree to which the data is normalized and whether, for a given set of related data in the warehouse, the model is implemented as a star or snowflake schema.

- ▶ Physical data model

This is the physical implementation of the logical model in a specific database management system. It identifies the implementation details in terms of such things as the configuration, keys, indexes selected, and constraints, and the means by which it is physically stored in the selected database structures. It is a translation from the logical model to the detailed implementation in the data warehouse.

This is a major step in the lifecycle of the data warehouse. To help with this process, the InfoSphere Warehouse Design Studio includes a subset of the functionality that is provided in the IBM Rational® Data Architect product.

### 3.2.1 Defining the physical data model

Physical data models define the internal schema of the data sources in your warehouse environment. Data models outline data tables, the columns in those tables, and the relationships between tables. Design Studio includes the components that are needed to create a physical model and generate the SQL that is appropriate for your implementation target.

The physical models that you create in Design Studio can be used to create schemas or to update schemas that exist in the target database.

Using Design Studio, in your physical data models, you can define the following data elements for the target database:

- ▶ Databases (one per data model)
- ▶ Tables
- ▶ Views
- ▶ Primary keys and foreign keys
- ▶ Indexes
- ▶ Stored procedures and functions
- ▶ DB2 table spaces and buffer pools

The physical data models that you create and update in Design Studio are implemented as entity relationship diagrams. The models are visually represented by using either Information Engineering (IE) notation or Unified Modeling Language (UML) notation. Before you begin your modeling work, configure Design Studio to use the notation that you prefer. Configure Design Studio by clicking **Data** → **Diagram**, which takes you to the configuration window where you can select the wanted notation.

### 3.2.2 Creating the physical data model

There are several different ways to create physical data models with Design Studio. For example, models can be created as follows:

- ▶ From an empty template
- ▶ From a database or DDL using the Physical Model wizard
- ▶ Dragging and dropping from the Database Explorer view to the project in the Data Project Explorer View

#### Creating the physical data model from a template

You can create a physical model from the empty template that is provided with Design Studio. To create a model this way, complete the following steps:

1. Highlight the Data Models folder in the project, right-click, and select **New** → **Physical Data Model**.

This action starts the New Physical Data Model wizard that is shown in Figure 3-1 on page 33. This wizard can also be started from the main Design Studio menu by clicking **File** → **New** → **Physical Data Model**.

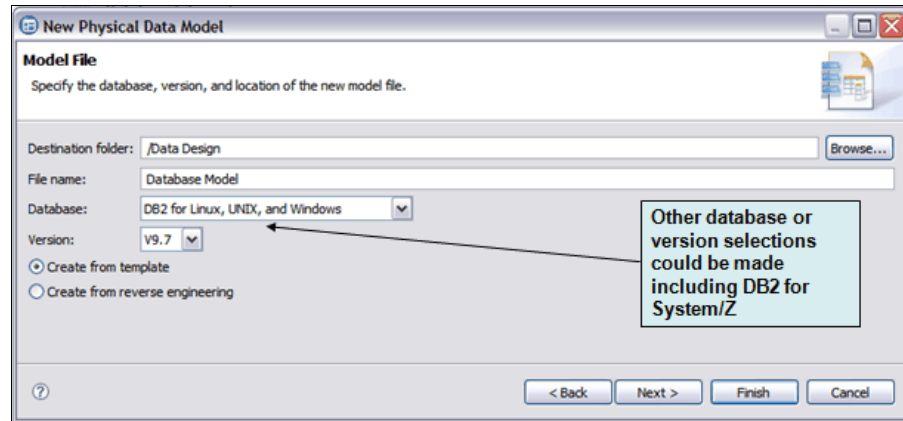


Figure 3-1 Physical Data Model wizard

2. Select the **Create from template** radio button.

Design Studio includes a modeling template that is called Empty Physical Model Template, which is shown on the next page of the wizard.

3. Choose the template and click **Finish**.

From the blank design template, you can use the visual diagram editor with its palette to design the model or you can use the Data Project Explorer to add objects to the model. For more information, see 3.2.4, “Using the Diagram Editor” on page 38.

## Creating the physical data model from an existing database

Another approach to creating physical data models is to reverse engineer one from a database connection or database definition. The steps to reverse engineer start out similar to the template approach:

1. Start the New Physical Model wizard by clicking **File** → **New** → **Physical Data Model** from the main menu or by right-clicking either the project or the Data Models folder in the Data Project Explorer view and selecting **New** → **Physical Data Model**.
2. Provide details, such as a destination folder, model name, and database type and version.
3. Select the **Create from reverse engineering** radio button and click **Next**.
4. The next window provides the choice to reverse engineer from either a database or a DDL script.

5. Enter the requested database connection information or the location of the DDL file (based on the choice you made in step 4 on page 33). You can either use an existing connection or define one. After you enter the information, you are presented with a list of schemas that can be reverse engineered.
6. Select a schema, as shown in Figure 3-2.

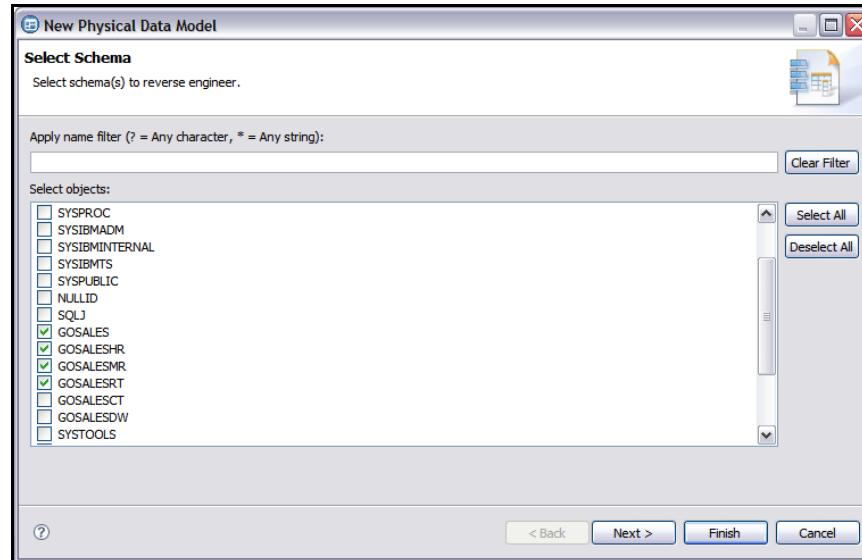


Figure 3-2 Selecting schemas

**Note:** If there is a filter that is defined as part of the database connection, the filter is applied every time the database connection is used. Therefore, this action impacts the list of schemas that you see in the New Physical Data Model wizard. To verify whether filters are in use, review the objects that are associated with the database connection in the Database Explorer view. If the schema folder is labeled Schemas [Filtered], then filters are enabled. To modify or review the filters, select the Schema folder in the Database Explorer, right-click, and select **Filter**. You can then make any changes as necessary.

As illustrated in Figure 3-3 on page 35, the schema object list at this step in the wizard might also be filtered dynamically. This filter specifies the qualifiers to be included in the list.

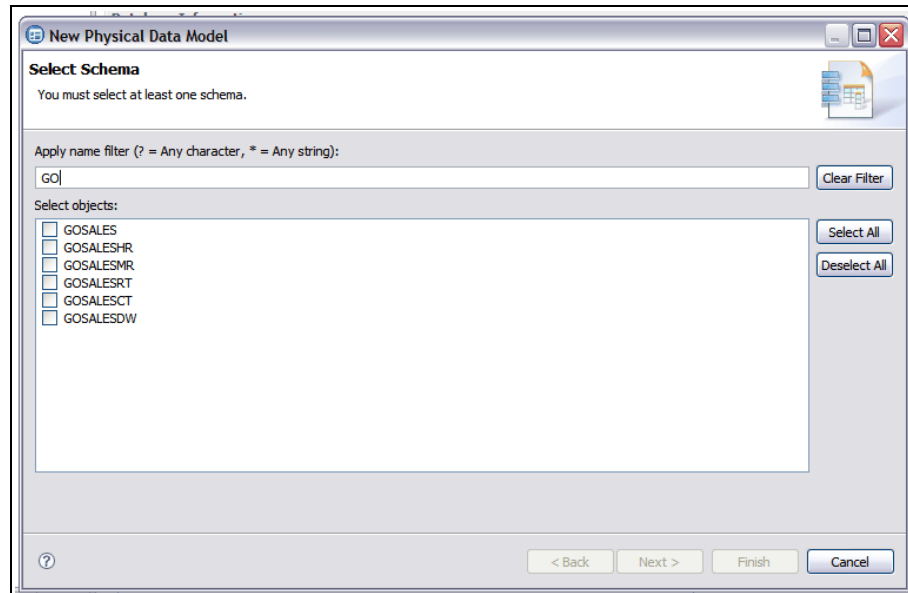


Figure 3-3 Filtering schemas

After you select the schemas that you want to use in the model, click **Next** to be presented with a list of the database elements that you want to include.

7. Select all the database object types of interest and click **Next**.

The final window of the wizard provides options to create an overview diagram and to infer implicit relationships. If you do not select these options, the components may be added to your model later.

## Creating the physical data model from the Database Explorer

Another method for creating a physical model from an existing database is to drag the objects from the Database Explorer to the Data Project Explorer. To use this method, complete the following steps:

1. Verify that there is an active connection to the source database in the Database Explorer view.

2. Expand the database connection and select the objects to be reverse engineered. The level of granularity that is available ranges from database to table. Drag the selected objects to the Data Project Explorer and on to an existing project, as shown in Figure 3-4. The resulting physical model name is the name of the database connection. A number is added to the end of the model name for uniqueness, if necessary.

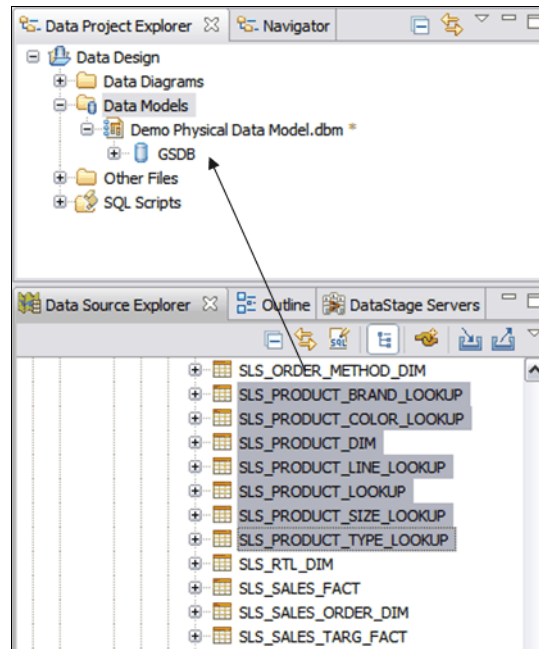


Figure 3-4 Dragging from the Database Explorer to create a physical data model

### 3.2.3 Working with diagrams

Design Studio uses entity relationship diagrams to provide a visual representation of the physical data models in your projects. Entity relationship diagrams are a useful mechanism for understanding the data elements in your projects and communicating that information to others. In Design Studio, you can have multiple diagrams per schema in the projects. It is often helpful to organize the data models into multiple subject areas, especially when these are large and complex.

In addition to the value that the diagrams bring to the projects by simplifying and improving the understanding of complex data environments, the diagrams can also be used to edit and modify the physical data model.

## Creating a diagram

Design Studio provides several ways to add entity relationship diagrams to projects.

If you choose to reverse engineer the data model, you can have the wizard create an overview diagram for you. The wizard provides prompts that allow you to specify what elements you want to include in the diagram.

You can still use diagrams in the data projects, even if you do not create the data models through the reverse engineering approach. New diagrams might be created from any Diagrams folder in the Project Explorer. Select the Diagrams folder, right-click, and select **New Overview Diagram**. You are prompted to select which elements from the current schema you want to include in the diagram.

You can also create a blank diagram, rather than including existing schema elements. To create a blank diagram, right-click the Diagrams folder in the Project Explorer and select **New Blank Diagram**.

### 3.2.4 Using the Diagram Editor

An overall view of the Diagram Editor is shown in Figure 3-5. The two main components to the Diagram Editor are the drawing area, or canvas, and the palette. Elements might be added to the diagram from either the palette or the Data Project Explorer. To use the Data Project Explorer, drag elements from the Data Models folder onto the diagram canvas.

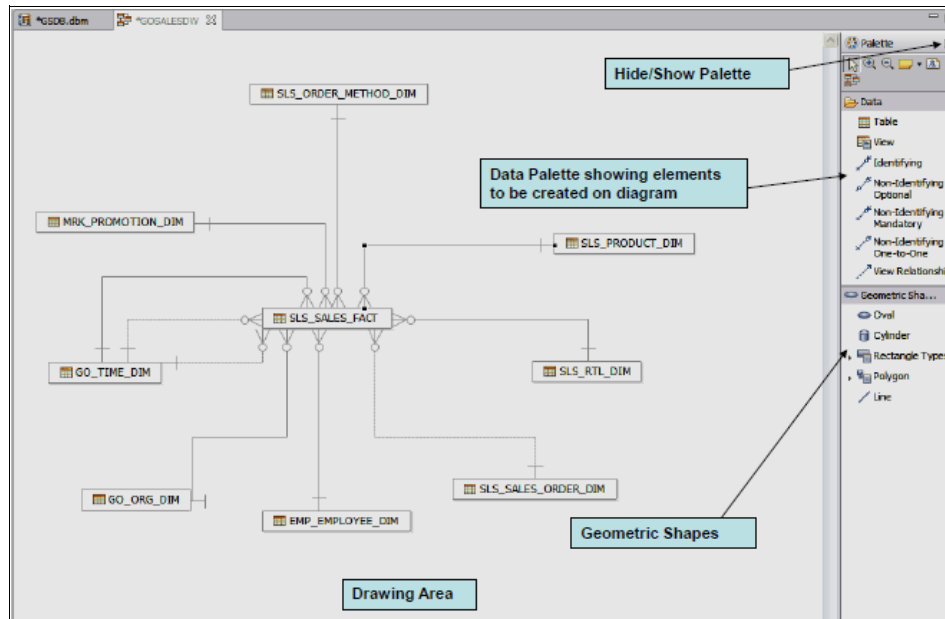


Figure 3-5 Diagram Editor view

The shapes representing tables and elements can be moved to make them more readable by highlighting either a rectangle or a line and dragging it to a new position.

Items from the palette might be placed on the drawing area by clicking an element to select it and moving the mouse to the drawing area and clicking. Elements that are added to the diagram in this manner are provided with a default name, but you can change the name to something more meaningful, as shown in Figure 3-6 on page 39. When elements are added to the canvas, as in Figure 3-6 on page 39, they are also added to the Data Project Explorer.



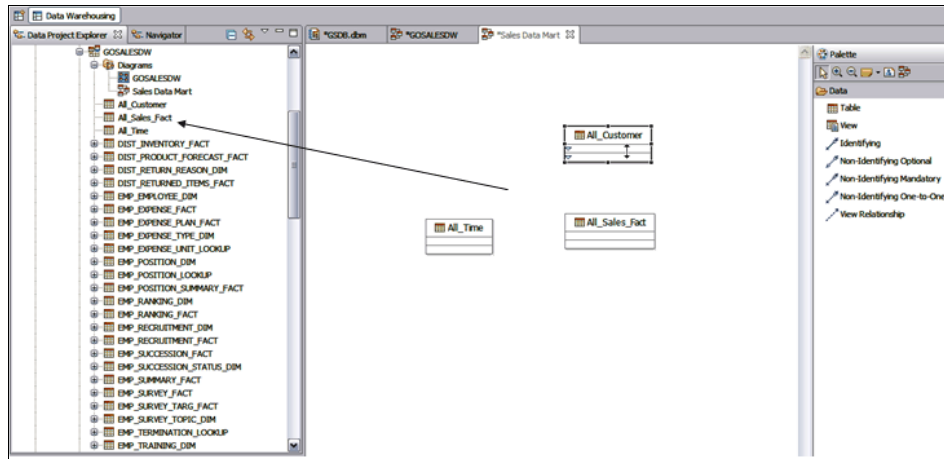


Figure 3-6 Adding palette elements to the diagram

When the diagram contains tables, then columns might be added from the visual diagram. When you select a model element, action bars appear, providing context-aware dialog boxes. The options that are available through the action bar include the ability to add a key, column, index, or trigger. The palette can be used to establish identifying and non-identifying relationships between the various tables that make up each diagram. As you are using this approach, you can also use the Properties view to further define the objects that you are creating.

### 3.2.5 Optimal attributes of the physical data model

As you build your physical data model, there are a number of considerations. There are tools to help confirm that the physical model is valid and to improve the quality of the model, as outlined in 3.2.6, “Model analysis” on page 40. However, the better the model is to start, the faster the model analysis phase is. The remainder of this section outlines some of the optimal attributes of a physical data model.

Ensure that standard data types are used across the data warehouse to facilitate joins between tables and the logical model definition in business intelligence tools. Moreover, use a DATE data type for date columns wherever possible.

For a given table definition, define columns as NOT NULL if they should have data. Use INTEGER data type for primary or foreign key columns, where possible. Define an index on foreign key columns; this index can be part of a unique primary key. For dimension tables, define a primary key to ensure uniqueness (if that cannot be ensured by the ETL processing).

To properly represent the relationship between tables in the data warehouse, use informational referential constraints between a dimension pair. For example, define a foreign key between a column in the fact table and the primary key column of a dimension table.

For more information, see the DB2 for Linux, UNIX, and Windows preferred practices paper *Physical database design for data warehouse environments*, found at:

<https://ibm.biz/Bdx2nr>

### 3.2.6 Model analysis

Design Studio includes a rule-based model analysis tool that allows you to evaluate the adherence of the data model to design and optimization preferred practices. The model analysis utility provides a mechanism to improve the integrity and quality of your physical models, and helps ensure that the physical model is valid. Validate physical data models before they are deployed to be sure that there are not any problems in the design.

Model analysis might be performed against databases or schemas. To perform model analysis, complete the following steps:

1. Start the wizard.
2. Select the database or schema you are interested in analyzing, right-click, and choose **Analyze Model**.

Models might be analyzed against the categories that are shown in Figure 3-7 on page 41. In each of these categories, there are several rules that you can choose to include in the analysis, such as key constraints, object naming standards, and several rules that are related to OLAP preferred practices.

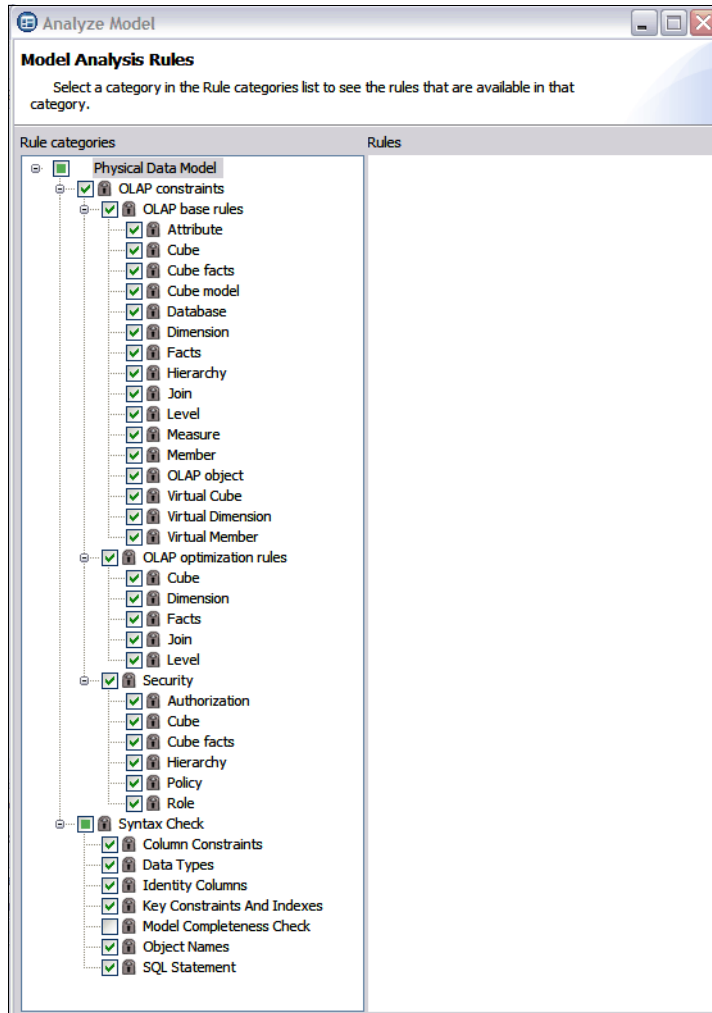
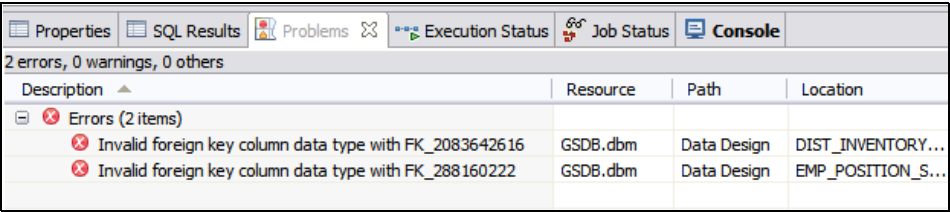


Figure 3-7 Model Analysis

3. Select the set of rules that are of interest and click **Finish** to start the analysis.

The results of the Model Analysis utility are shown in the Problems view. An example of the output is shown in Figure 3-8. Double-click an item in the Problems view to navigate to the relevant model objects in the Project Explorer. Take corrective action and rerun the model analysis to verify that all problems were corrected.



The screenshot shows the 'Problems' tab in a software interface. At the top, there are tabs for 'Properties', 'SQL Results', 'Problems', 'Execution Status', 'Job Status', and 'Console'. Below the tabs, it says '2 errors, 0 warnings, 0 others'. The main area is a table with columns: 'Description', 'Resource', 'Path', and 'Location'. There are two error entries, both marked with a red 'X' icon.

Description	Resource	Path	Location
Errors (2 items)			
Invalid foreign key column data type with FK_2083642616	GSDb.dbm	Data Design	DIST_INVENTORY...
Invalid foreign key column data type with FK_288160222	GSDb.dbm	Data Design	EMP_POSITION_S...

Figure 3-8 Model analysis output

Correct any issues that the model analysis identifies until all errors are cleared. After the model is validated, you can deploy it.

### 3.2.7 Deploying the data model

After the physical model is designed and validated, you can deploy the model into your environment. The deployment process results in the schemas, tables, and other objects that were modeled being created in the target database. After the physical model is deployed, you can begin populating the structures with data.

#### Using Design Studio to deploy a physical data model

Physical data models might be deployed from Design Studio through the Generate DDL wizard, which generates context-driven DDL. When you generate the DDL code, you can choose a database, schema, user, table, buffer pool, or table space as the root for the code generation. To start the DDL wizard, select the relevant data element, right-click, and select **Generate DDL**. Alternatively, to start the wizard from the menu, with the data element selected, click **Data** → **Generate DDL**. Respond to the prompts of the wizard, indicating what model elements and objects you want to include in the DDL script.

For an example of the Generate DDL wizard, see Figure 3-9.

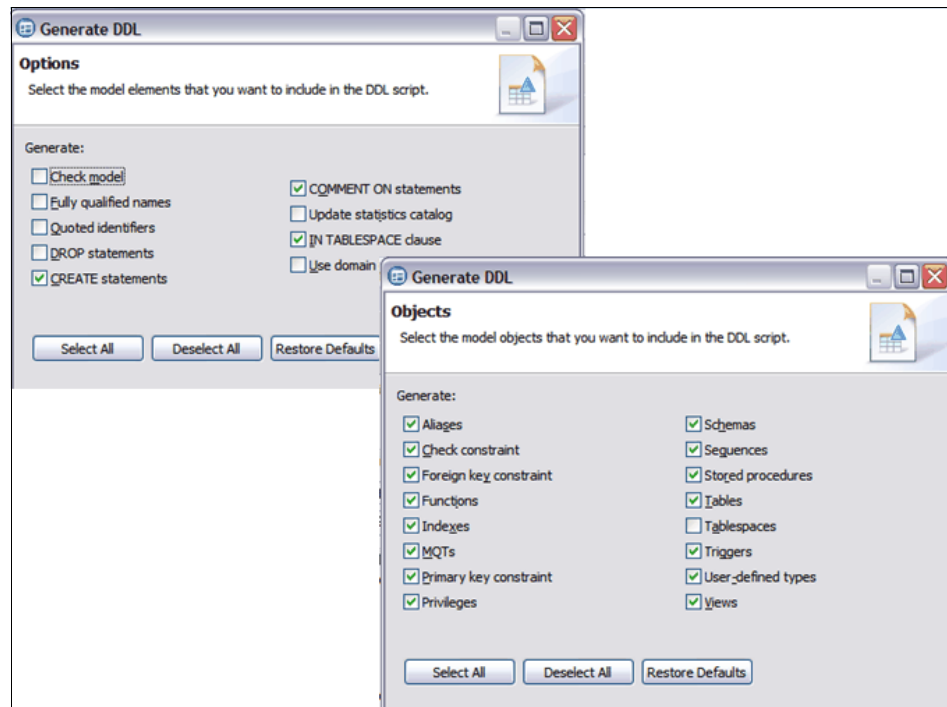


Figure 3-9 Generating a DDL

After you select the objects that you want to deploy, you are presented with a summary of the DDL script, with options to save the DDL file or run the DDL script on the server. These options are shown in Figure 3-10. If you choose to run the DDL on the server, you are prompted to select a database connection or create a database connection. The DDL script is saved in the SQL Scripts logical folder in your data design project.

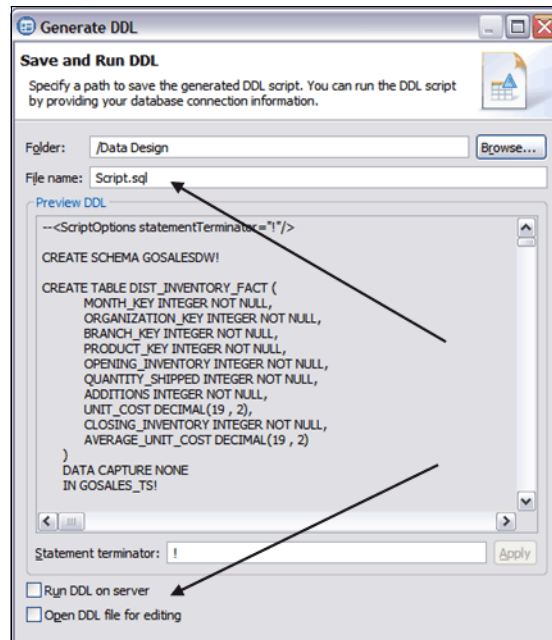


Figure 3-10 Options for saving and running a DDL

The DDL scripts that are in the SQL Scripts folder might be saved for later execution, and they might also be modified before being run. To edit the DDL, select the file in the SQL Scripts folder, right-click, and select **Open With** → **SQL Editor**. This causes the file to be opened within a text editor. When you are ready to run the DDL, select the file, right-click, and select **Run SQL**. You can review the results in the Data Output view, which includes status information, messages, parameters, and results.

### 3.2.8 Maintaining model accuracy

It is important to manage changes to the models that might happen over time. As business requirements change, there is often an adjustment that must be made to the physical data models. Design Studio can help you manage that type of change. It is helpful to compare the differences between two physical data models, and also compare the difference between a physical data model and the database where it was deployed. After changes are identified, it is important to synchronize the two models being compared. It is also important to analyze the impact of a change to the model before the change is implemented.

#### Comparing objects in the physical data model

Design Studio provides object-based compare and synchronization capabilities as a mechanism to assist and simplify the task of maintaining model accuracy. From the Database Explorer, you can select a database object, right-click, and select **Compare With → Another Data Object**. You are prompted to select the object to use for comparison. Another way to achieve the same comparison is to highlight two data objects, right-click, and select **Compare With → Each Other**. In the Project Explorer, there is a third comparison available if the source of the model was created by reverse engineering. If that is the case, select an object, right-click, and select **Compare → Original Source**. This option is only available from the Project Explorer, not the Database Explorer.

These options under the **Compare With** tool are helpful if you must compare physical objects with each other (for example, to compare objects from a test database to objects in a production database). You can compare the baseline database objects from the Database Explorer with changed objects from your project before deploying changes.

#### Visualizing differences between objects

The results of the object comparison are displayed in two views, which are connected. The upper view is called *Structural Compare*. The Structural Compare shows the differences in a tree format. The first column is a tree. Each entry in the tree represents a part of the model where a difference was found. The second column in the Structural Compare output shows the first object as input to the Compare tool, and the third column shows the second object to which it was being compared. A copy of the differences might be saved by clicking **Export** in the upper right portion of the Structural Compare view. The file that results from this option is an XML file.

If two different items that appear on different rows in the comparison output must be compared to each other, you can use the Pair and Separate buttons to facilitate their comparison and align the comparison results. This is helpful when the column names are different, but the objects represent the same data.

As you work down the tree in the Structural Compare, differences are highlighted in the lower view, which is called the *Property Compare*. This shows the Properties view. An example of the output from the Compare Editor is shown in Figure 3-11.

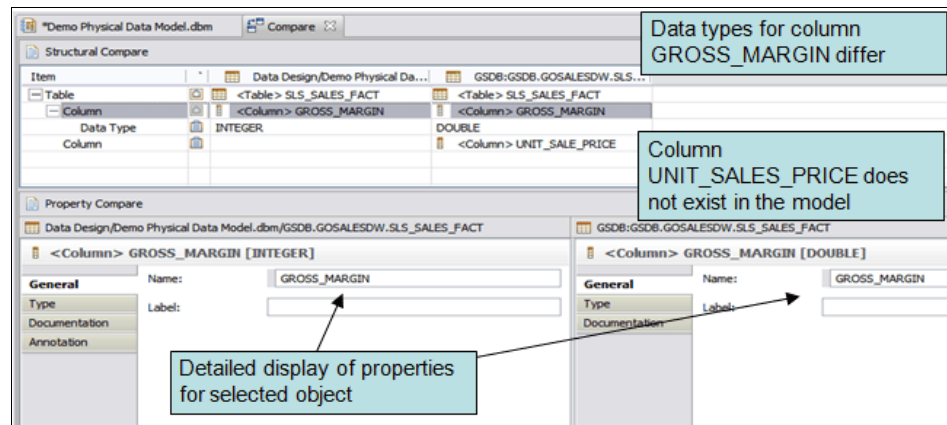


Figure 3-11 The Compare Editor

## Synchronization of differences

As you view the differences between the objects you have compared, you can use the Copy From buttons, which are on the right side of the toolbar that separates the Structural Compare and Property Compare views. These buttons allow you to implement changes from one model or object to another. You can copy changes from left to right, or from right to left. As you use the Copy From buttons, the information that displayed in the Structural Compare is updated to reflect the change.

Another way to implement changes to bring your models in synchronization with each other is to edit the values that are displayed in the Property Compare view. Any changes that are made might be undone or redone by clicking **Edit** → **Undo** and **Edit** → **Redo**.

After you reviewed all of the differences and decided what must be done to bring your models in sync with each other, you are ready to generate delta DDL. This DDL can be saved for later execution or might be run directly from Design Studio.



## Impact analysis

Design Studio also provides a mechanism for performing impact analysis. It is beneficial to understand the implications of changes to models before they are implemented. The impact analysis utility shows all of the dependencies for the selected object. The results are visually displayed, with a dependency diagram and a Model Report view added to the Output pane of Design Studio.

The impact analysis discovery can be run selectively. To start the utility, highlight an object, right-click, and select **Analyze Impact**. Select the appropriate options, as shown in Figure 3-12.

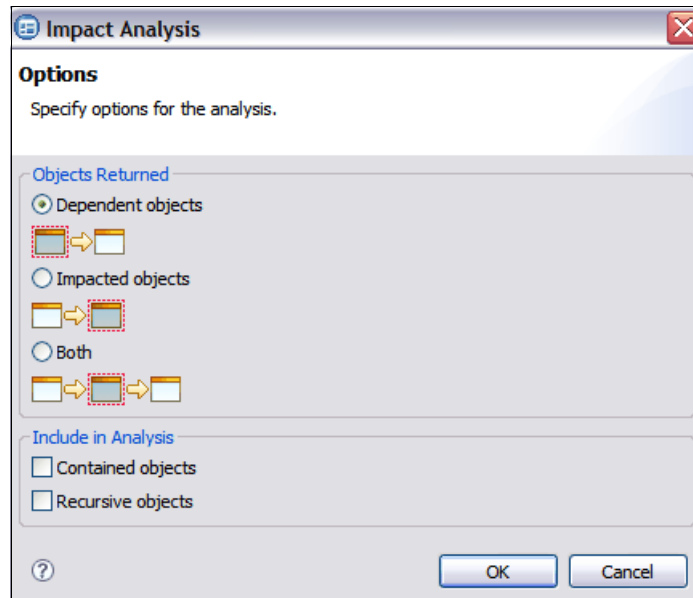


Figure 3-12 Impact Analysis Options

The results of the impact analysis are shown in a dependency diagram, as shown in Figure 3-13.

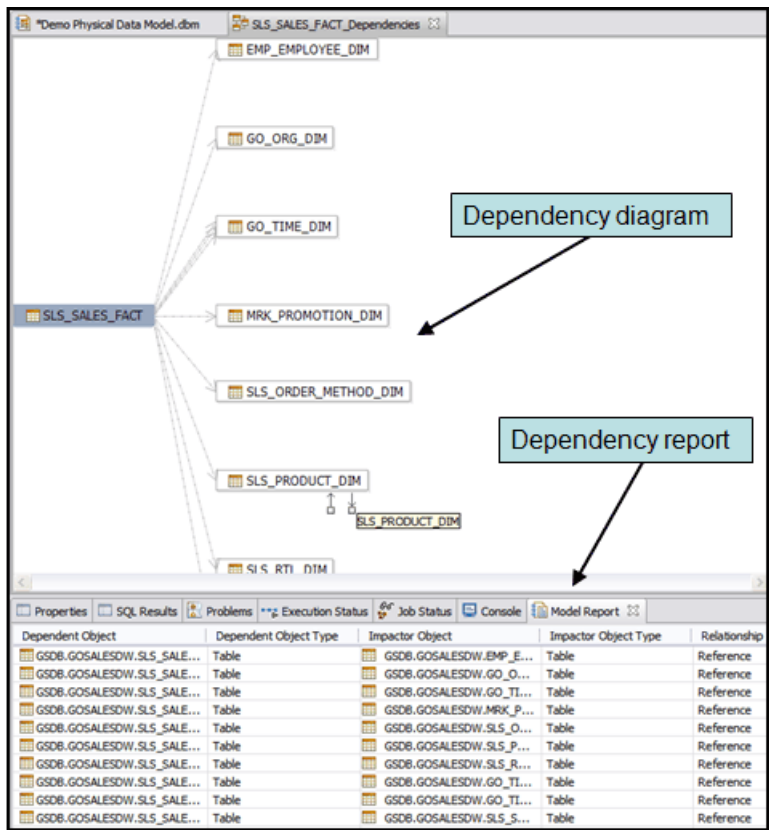


Figure 3-13 Impact Analysis Report

### 3.3 Matching the technology to the requirements

The next phase of developing the data warehouse is choosing the optimal technology for the business requirements and data model. The subsequent chapters of this book describe the strengths of each technology that is available in DB2 10 for Linux, UNIX, and Windows. The database software contains multiple technologies, each optimized for a particular type of workload, that can be used in concert together to match the best technology to a given workload.

It is a *best of both worlds* architecture that can handle both embedded analytics of structured data but also generate and leverage knowledge from unstructured information. It is the business requirements that drive the decision to scale up within a single system (using either BLU Acceleration or standard intrapartition parallelism) versus scale out with a shared-nothing architecture across multiple logical or physical nodes.





## Part 2

# Technologies

The next phase of developing the data warehouse is choosing the optimal technology for the business requirements and data model. The chapters in this section describe the strengths of the major technologies that are available in DB2 10 for Linux, UNIX, and Windows that you can leverage to optimize the value and performance of your data warehouse.

Chapter 4, “Column-organized data store with BLU Acceleration” on page 53 shows how this remarkable innovation provides large order of magnitude improvements in analytic workload performance, significant storage savings, and reduced time to value.

Traditional row-based DB2 warehouse solutions are the subject of Chapter 5, “Row-based data store” on page 71, which covers database partitioning, intrapartition parallelism, and other technologies that you can use to build a high-performance data warehouse environment, including compression, multidimensional clustering, and range (table) partitioning.

Chapter 6, “Data movement and transformation” on page 91 gives you an overview of some of the important DB2 data movement utilities and tools, and describes the data movement and transformation capabilities in the DB2 Warehouse SQL Warehousing Tool (SQW).

Database monitoring, which includes all of the processes and tools that you use to track the operational status of your database and the overall health of your database management system, is the subject of Chapter 7, “Monitoring” on page 105, which provides an overview of the DB2 monitoring interfaces and IBM InfoSphere Optim Performance Manager Version 5.3.

Chapter 8, “High availability” on page 119 describes the high availability characteristics of IBM PureData System for Operational Analytics, which provides redundant hardware components and availability automation to minimize or eliminate unplanned downtime.

DB2 workload management is the focus of Chapter 9, “Workload management” on page 145, which gives you an overview of this important technology whose purpose is to ensure that limited system resources are prioritized according to the needs of your business. In a data warehouse environment, business priorities and workload characteristics are continually changing, and resource management must keep pace.

The data mining process and unstructured text analytics in the DB2 Warehouse are covered in Chapter 10, “Mining and unstructured text analytics” on page 163, which describes embedded data mining for the deployment of mining-based business intelligence (BI) solutions.

Chapter 11, “Providing the analytics” on page 177 focuses on the relational OLAP capabilities of the Dynamic Cubes feature of IBM Cognos Business Intelligence V10.2, which enables high-speed query performance, interactive analysis, and reporting over terabytes of data for many users.



## Column-organized data store with BLU Acceleration

This chapter introduces BLU Acceleration, which provides significant order of magnitude benefits for analytic workload performance, storage savings, and time to value.

BLU Acceleration is a new technology in DB2 for analytic queries. This set of technologies encompasses CPU-, memory-, and I/O-optimization with unique runtime handling and memory management and unique encoding for speed and compression.

# 4.1 Simple and fast with BLU Acceleration

BLU Acceleration is a combination of complementary innovations from IBM that simplifies and speeds up analytic workloads. More importantly, BLU Acceleration is easy to set up and is self-optimizing, which means a lot less work for the DBA to deploy and manage the solution. BLU Acceleration is introduced in DB2 for Linux, UNIX, and Windows 10.5 (DB2 10.5).

BLU Acceleration is a fully integrated feature of DB2 and does not require any SQL or schema changes to implement. In fact, BLU tables can coexist with tradition row tables using the same table spaces and buffer pools. With no need for database tuning, indexes, or aggregates, the DBA can load the data and go.

To understand the benefits of BLU Acceleration, it is important to review its complementary technologies. Dynamic in-memory columnar technologies provide an efficient method to scan and find relevant data. Those technologies are then combined with innovations such as parallel vector processing and actionable compression to make analytic queries far faster.

## 4.1.1 Dynamic in-memory columnar technology

Database tables that use BLU Acceleration are stored differently than the traditional row store table that you know from previous versions of DB2. With BLU Acceleration, a table can now be created as a column organized table. When data is stored in this fashion, only the columns that are necessary to satisfy a query are scanned, which can reduce the time that is required to locate the most relevant data. The logical representation of a row still exists, though, as shown in Figure 4-1.

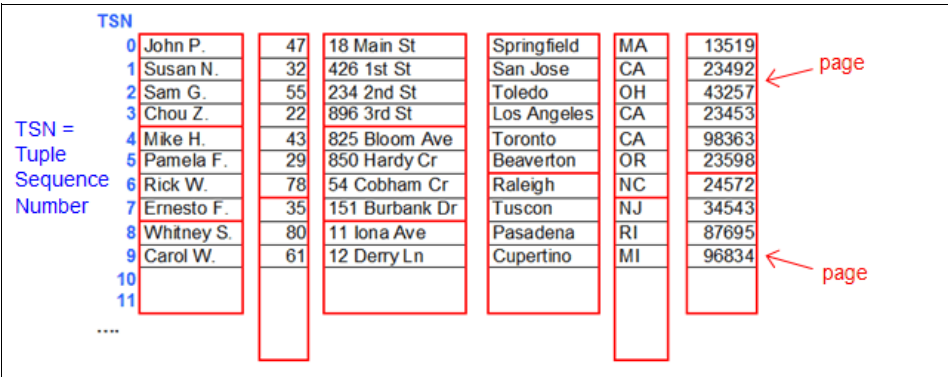


Figure 4-1 The relationship between TSN (logical row representation) and data pages in BLU Acceleration



BLU Acceleration is in-memory optimized, but is not limited by the available memory. Although BLU Acceleration is optimized for accessing data in memory, performance does not suffer as data size grows beyond the memory because of the other technological innovations that are described in the following sections. In addition, BLU Acceleration dynamically moves unused data to storage, ensuring an efficient use of memory resources, as only the most relevant data is kept in-memory.

### **4.1.2 Actionable compression**

There are many benefits to actionable compression.

The first benefit is storage savings. BLU Acceleration uses approximate Huffman encoding, prefix compression, and offset compression. Approximate Huffman encoding is a frequency-based compression that uses the fewest number of bits to represent the most common values so the most common values are compressed the most. As a result, clients report compression rates with BLU Acceleration that are greater than that experienced with static or adaptive compression of row tables. Combine the improved compression of data with no need for indexes or aggregates and the overall storage requirements for the database are dramatically reduced.

The second benefit is that actionable compression is order-preserving on each column, which means that data can be analyzed when compressed. This results in a dramatic I/O, memory, and processor savings. Encoded values (the compressed data) can be packed into a processor register, optimizing each processor cycle to process the largest amount of data. Less I/O and fewer processor cycles lead to dramatic improvements in processing time.

A further benefit of actionable compression is that the data is not decoded (or decompressed) when processing SQL predicates (that is, =, <, >, >=, <=, BETWEEN, and so on), joins, aggregations, and more. Rather, the decompression occurs only when necessary, for example, when the result set is returned to the user.

### **4.1.3 Parallel vector processing**

Building on actionable compression and the ability to pack each processor register with a maximum amount of data to process, parallel vector processing adds yet another level of processor optimization.

Multiple data parallelism is achieved by using special hardware instructions that are known as Single Instruction Multiple Data (SIMD). By using SIMD, it is possible to work on multiple data elements with a single instruction. The processor register is packed with more data (because of actionable compression, which is described in 4.1.2, “Actionable compression” on page 55), and with SIMD, all the data is processed with a single instruction.

Multi-core parallelism is the other aspect of parallel vector processing. BLU Acceleration takes advantage of the available processor cores and drive multi-core parallelism for the queries it processes. Queries on column-organized table (BLU tables) are automatically parallelized to optimize performance.

#### 4.1.4 Data skipping

One final BLU Acceleration technology to be aware of is *data skipping*. Data skipping allows BLU Acceleration to skip unnecessary processing or irrelevant or duplicate data. If BLU Acceleration automatically detects a large section of data that does not qualify for a query, the data is ignored and skipped. This leads to order of magnitude savings for I/O, memory, and processor. There is no DBA action that is required to enable data skipping. Rather, BLU Acceleration automatically maintains a persistent store of minimum and maximum values, called a *synopsis table*, for sections of data values. The automatic maintenance occurs during load, insert, update, and delete of data. When a query accesses a column, BLU Acceleration uses the synopsis table to use data skipping wherever possible.

## 4.2 Getting started with BLU Acceleration

As of the general availability of DB2 for Linux, UNIX, and Windows Version 10.5, BLU Acceleration is supported on IBM POWER®, AIX, x86, and Linux platforms. The operating system minimum requirements for using the BLU Acceleration feature are no different from the standard DB2 10.5 requirements.

BLU Acceleration is available in either DB2 Advanced Enterprise Server Edition (AESE) or DB2 Advanced Workgroup Server Edition (AWSE). As outlined in Chapter 2, “Technical overview of IBM DB2 Warehouse” on page 9, both of these editions are full-featured, but AWSE has capacity limitations (namely, processor core and instance memory limitations depending on license type). For more information about DB2 10.5 system requirements, consult the *System requirements for IBM DB2 for Linux, UNIX, and Windows*, found at:

<http://www.ibm.com/support/docview.wss?uid=swg27038033#105>

## 4.2.1 Capacity planning

To determine the minimum requirements for using BLU Acceleration, it is important to determine the amount of active data in the database.

**Note:** Active data refers to the data that is commonly accessed in queries.

The size of active data, not the total size of the database, is the relevant metric to use when planning the capacity that is required for BLU Acceleration. The total size of the database is required only when considering total storage requirements.

In the context of BLU Acceleration, the process to estimate active data size follows an approach similar to this one:

1. Determine the total amount of raw data.
  - a. If you are starting from scratch, the total amount of raw data is the size of the comma-separated value (CSV) files that are used to load the table.
  - b. If you are sizing based on an existing DB2 database, the total amount of raw data can be calculated based on the size of the data that is loaded into the database, as shown in Example 4-1.

*Example 4-1 Estimating raw data size for an existing DB2 table*

---

```
select sum(a.fpages * (1.0/(1.0 - (cast(a.pctpagesaved as
decimal(5,2))/100)))) * c.pagesize/1024/1024/1024) as
uncompressed_table_data_GB
FROM syscat.tables a, syscat.datapartitions b, syscat.tablespace c
WHERE
a.tabschema not like 'SYS%' AND
a.tabschema = b.tabschema AND
a.tabname = b.tabname AND
b.datapartitionid = 0 AND
b.tbpaceid = c.tbpaceid
```

---

2. Determine the percentage of data that is accessed frequently (that is, by more than 90% of the queries). For example, suppose that you have a table of sales data that contains data that is collected 2000 - 2013, but only the most recent three years are accessed by most queries. In this case, only data 2010 - 2013 is considered active, and all other data in that table is considered inactive and be excluded from sizing calculations.
3. Determine the percentage of columns that are frequently accessed. With BLU Acceleration, only the columns that are accessed in the query are considered active data.

After you determine an estimate of the raw active data size, you can then consider the system resources that are required regarding concurrency and complexity of the workload, the amount of active data to be kept in memory in buffer pools, and more. The minimum system requirements for BLU Acceleration are eight cores and 64 GB of memory. As a preferred practice, maintain a minimum ratio of 8 GB of memory per core.

For more information about sizing guidelines, see *Best Practices: Optimizing analytic workloads using DB2 10.5 with BLU Acceleration*, found at:

[https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/0fc2f498-7b3e-4285-8881-2b6c0490ceb9/page/ecbdd0a5-58d2-4166-8cd5-5f186c82c222/attachment/e66e86bf-4012-4554-9639-e3d406ac1ec9/media/DB2BP\\_BLU\\_Acceleration\\_0913.pdf](https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/0fc2f498-7b3e-4285-8881-2b6c0490ceb9/page/ecbdd0a5-58d2-4166-8cd5-5f186c82c222/attachment/e66e86bf-4012-4554-9639-e3d406ac1ec9/media/DB2BP_BLU_Acceleration_0913.pdf)

## 4.2.2 Storage requirements

BLU Acceleration has more random I/O access than row-organized data marts that typically perform large sequential scans. As a result, it can be beneficial to use I/O subsystems and devices that provide high random I/O access when using BLU Acceleration. Examples of storage that provide better random I/O access are flash-based storage and solid-state drives (SSD). BLU accelerated tables that are accessed frequently benefit from improved I/O subsystem performance, particularly when the table greatly exceeds the memory size that is available for use by BLU Acceleration. In addition, any temporary table space usage can greatly benefit from higher performing I/O subsystems.

In terms of overall storage sizing, the requirements are similar to that of standard row store. You must ensure adequate space for user data and temporary data. With improved compression on column-organized tables, the user data requirements are typically less than that of static or adaptive compressed row-organized tables.

## 4.3 Creating a column-organized data store

A key tenet of BLU Acceleration is simplicity. There is less effort that is required in creating and loading a BLU accelerated data mart. With the usage of a single registry variable, and the review of a few database configuration options, the only DBA action that is required is to load the data into the tables. After the data is loaded, the database is ready for business, and self-optimizes its performance.

### 4.3.1 Single configuration setting for analytic workloads

If you are creating a database that is dedicated to analytic workloads, you can use a simple registry setting to configure DB2 and the database for BLU Acceleration. The single registry variable automatically configures the instance and database configuration for optimal analytic performance. Example 4-2 shows the three simple commands that are required to create a database that is optimized for analytic workloads.

*Example 4-2 Creating a database that is optimized for BLU Acceleration*

---

```
db2set DB2_WORKLOAD=ANALYTICS
db2start
db2 create db mydb autoconfigure using mem_percent 100 apply db and dbm
```

---

**Note:** Do not specify **CREATE DATABASE AUTOCONFIGURE NONE**, which disables the ability of the registry variable to influence the automatic configuration of the database for analytic workloads.

The default setting for the **AUTOCONFIGURE** command are **USING MEM\_PERCENT 25**. Specify the **“USING MEM\_PERCENT”** option to allocate more memory to the database and instance.

When **DB2\_WORKLOAD=ANALYTICS**, the auto-configuration of the database sets the default table organization method (`dft_table_org`) to **COLUMN** (that is, BLU Acceleration) and enables intra-query parallelism. Database memory configuration parameters that cannot use self-tuning memory manager and need an explicit value (that is, sort-related parameters) are set to a value higher than default and optimized for the hardware you are running on. Automatic space reclamation is enabled. Finally, workload management is enabled by default to ensure maximum efficiency and usage of the server. For more information, see the “Column-organized tables” topic in the DB2 10.5 information center at the following website:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp>

**Note:** The registry variable **DB2\_WORKLOAD** is set at the instance level. It influences all databases in that instance. If the instance is going to have multiple database, not all of which are BLU Accelerated, then set the registry variable before creating the databases you want to optimize for analytic workloads and then unset the registry variable before creating the other databases.

To optimize a pre-existing database for analytic workloads, set the registry variable and run the **AUTOCONFIGURE** command. Example 4-3 illustrates the required commands.

*Example 4-3 Optimizing a pre-existing database for analytic workloads*

---

```
db2set DB2_WORKLOAD=ANALYTICS
db2start
db2 connect to mydb
db2 autoconfigure apply db only
```

---

### 4.3.2 Fine-tuning a configuration

The automatic configuration of the database is largely influenced by the existing hardware resources. As a result, you should review the size of the buffer pools, sort heap, and sort heap threshold that are set by auto-configure based on application needs and your expected data volumes.

In general, you want to split 80 - 90% of the memory that is allocated to DB2 between buffer pool and sort memory allocations. The remaining memory is used for all other heaps, including the utility heap, which is critical to LOAD performance. With higher concurrency workloads, you might want to allocate more memory to sort than buffer pool.

When allocating sort memory, there are two configuration parameters to consider: **SHEAPTHRES\_SHR** and **SORTHEAP**. **SHEAPTHRES\_SHR** determines the limit for the allocation of sort memory. When a certain percentage of the value of **SHEAPTHRES\_SHR** is attained, the DB2 database manager begins to throttle the memory that is allocated to sort memory consumers to ensure that over commitment of memory does not occur. This means that consumers might receive only their minimum allocation requirement. As such, it is important to consider the level of concurrency that is expected for your applications and the WLM configuration that controls how many run concurrently.

Table 4-1 provides an example of memory allocation for low and high concurrency workloads where the WLM configuration allows 20 concurrent high-cost queries.

Table 4-1 Example of memory distribution for a BLU-accelerated database based on workload concurrency

Low concurrency (< 20 concurrent workloads)	High concurrency (> 20 concurrent workloads)
40% buffer pool	25% buffer pool
40% SHEAPTHRES_SHR	50% SHEAPTHRES_SHR
SORTHEAP = SHEAPTHRES_SHR/5	SORTHEAP = SHEAPTHRES_SHR/20

### 4.3.3 Creating column-organized tables

When the database is created as optimized for analytic workloads (and the database configuration setting is `dft_table_org=column`), the **CREATE TABLE** statement defaults to creating tables that are column-organized. However, the **ORGANIZE BY** clause can be used to explicitly specify the type of column organization regardless of the default table organization for the database, as shown in Example 4-4.

Example 4-4 Creating a table with an **ORGANIZE BY** clause

```
CREATE TABLE col_table ... ORGANIZE BY COLUMN IN data_tablespace;  
CREATE TABLE row_table ... ORGANIZE BY ROW IN data_tablespace;
```

The same SQL syntax that is used to define columns for row-organized tables is used for column-organized tables. For more information about supported options for the **CREATE TABLE** statement for row- and column-organized tables, see the DB2 10.5 information center at the following website:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp>

**Note:** Create fact tables in their own table space to allow for better manageability and performance by isolating storage groups and buffer pools for a certain fact table.

Ensure that you use the NOT NULL attribute for columns where possible. Every nullable column in a column-organized table is a two-column group of columns: one for the nullability attribute and one for the actual column. These nullable columns cause extra work to be done on each row during query processing.

To determine whether a table is defined as row-organized or column-organized, you can query the SYSCAT.TABLES catalog view, as shown in Example 4-5. A new column, TABLEORG, contains “C” for column-organized and “R” for row-organized.

*Example 4-5 Querying the system catalog to determine table organization*

```
SELECT tabname, tableorg, compression
FROM syscat.tables
WHERE tabname like 'mytab%';
```

TABNAME	TABLEORG	COMPRESSION
-----	-----	-----
SALES_COL	C	
SALES_ROW	R	N

2 record(s) selected.

**Note:** The COMPRESSION column for column-organized tables is blank because the data in such tables is always compressed.

### Constraints

Primary key, informational foreign key, and any informational check constraints should be defined after your initial data is loaded.

DB2 10.5 introduces an option for specifying not enforced (informational) primary key constraints or unique constraints. The **NOT ENFORCED** clause does not ensure uniqueness, so this option should be used only when you are certain that the data is unique. Informational primary key constraints or unique constraints require less storage and less time to create because no internal B-tree structure is involved. However, informational primary key, foreign key, and check constraints can be beneficial to the query optimizer. If your data is cleansed as part of a rigorous ETL process, then consider the use of informational constraints.

**Note:** An *enforced* primary key constraint uses extra space because a unique index is required to enforce the constraint. In addition, like any index, this unique index increases the cost of insert, update, or delete operations. An *unenforced* primary key constraint does not have these additional considerations.



### 4.3.4 Converting to column-organized tables

If you want to convert one or more row-organized tables in your database to column-organized tables, you can use the **db2convert** command to automate the process. With a single invocation of the command, the **db2convert** command uses the `ADMIN_MOVE_TABLE` stored procedure to move data between the original row-organized table definition and the new column-organized table definition. The **db2convert** command also drops any dependent objects (such as secondary indexes or MQTs) as part of the conversion process because they are not required for column-organized tables. Example 4-6 shows an example of calling the **db2convert** command to convert a single table.

*Example 4-6 Sample syntax for converting a row-organized table to column-organized*

---

```
$ db2convert -d mydb -z db2inst1 -t mytable
```

```
Conversion notes for exceptional table(s)
```

```
-----
```

```
Table: DB2INST1.MYTABLE:
```

```
  -Secondary indexes will be dropped from the table.
```

```
Enter 1 to proceed with the conversion.
```

```
Enter 2 to quit.
```

---

The **db2convert** command can convert a single table or multiple tables at one time.

## 4.4 Loading a column-organized data store

The load process for a column-organized table is similar to that of a row-organized table. A column-organized table is loaded from input data sources that are in a row-organized format (for example, a comma-separated value data file). During the load or insert process, the data is compressed and converted into column-organized format and metadata tables are maintained. As part of the initial load operation on a column-organized table, BLU Acceleration builds a column compression dictionary for each column. The following sections provide more information about each of these unique aspects of loading column-organized tables.

A load into a column-organized table automatically collects statistics according to the profile that is defined for the table. If a profile does not exist, default automatic statistics (equivalent to the **WITH DISTRIBUTION AND SAMPLED DEATEILED INDEXES ALL** clause) are collected by the **RUNSTATS** utility. With this automatic statistics collection, it is not required to update statistics after loading a column-organized table unless primary key constraints are created after load.

For more information about data movement with BLU Acceleration, see “Load considerations for column organized tables” on page 98.

### 4.4.1 Synopsis tables and data skipping

As introduced in 4.1.4, “Data skipping” on page 56, BLU Acceleration uses internal metadata tables that are called synopsis tables to determine what data can be skipped if it is of no interest to a query. For example, suppose that a query calculates the sum of last month’s sales. There is no need to look at data that is older than the last month, so BLU Acceleration can automatically skip over that older data.

The synopsis table contains all of the user table’s non-character columns (namely, datetime, Boolean, or numeric columns) and those character columns that are part of a primary key or foreign key definition. The synopsis table stores the minimum and maximum values for each column for a range of data, as shown in Figure 4-2. The range of data is determined by tuple sequence number (TSN), which indicates which column values belong together as a logical row.

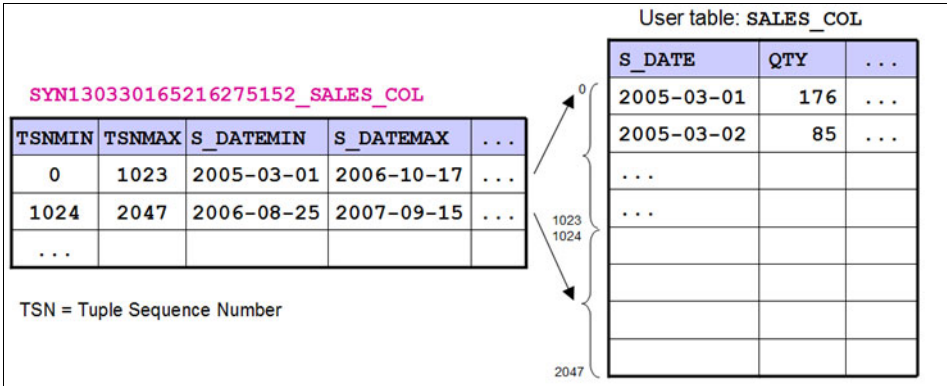


Figure 4-2 Sample of a synopsis table showing the relationship to its user table

A synopsis table is approximately 0.1% of the size of its user table with one row in the synopsis table for every 1024 rows in the user table. To determine the name of the synopsis table for a particular user table, query the **SYSCAT.TABLES** catalog view, as shown in Example 4-7 on page 65.

Example 4-7 Querying the system catalog to identify the synopsis table for a user table

```
SELECT tabschema, tabname, tableorg
  FROM syscat.tables
 WHERE tableorg = 'C' and tabname like '%SALES_COL%';
```

TABSCHEMA	TABNAME	TABLEORG
DB2INST1	SALES_COL	C
SYSIBM	SYN130330165216275152_SALES_COL	C

2 record(s) selected.

It is important to understand how synopsis tables are used to optimize data skipping to ensure an optimal physical data model. Synopsis tables store minimum and maximum values for all numeric, datetime, and primary and foreign key columns. Therefore, the more columns use numeric and datetime data types, the more effective data skipping is. For example, by defining a non-key column that stores a date to use the DATE data type rather than a character data type, the column is included in the synopsis table.

The synopsis tables are maintained during LOAD or INSERT, UPDATE, and DELETE processing to ensure a correct representation of the minimum and maximum values for a data range.

#### 4.4.2 Compression of column-organized tables

A new phase is introduced for the load process of column-organized tables: the ANALYZE phase. During the ANALYZE phase, column compression dictionaries are built if they do not exist (that is, this is the first load of data into the table). If the column compression dictionary exists, the ANALYZE phase is not required.

The ANALYZE phase precedes the LOAD and BUILD phases of load. During the LOAD phase, the data is compressed by using column and page compression dictionaries, the compressed values are written to data pages in column-organized form, and synopsis tables are maintained. During the BUILD phase, any unique indexes for enforced primary key constraints are built. This phase always occurs for column-organized tables.

##### Calculating table size and compression details

There is nothing new to calculating table size; the same method should be used for column-organized and row-organized tables. The ADMIN\_GET\_TAB\_INFO table function is updated to add information specific to column-organized tables.

To calculate the total size of a table, use the SQL statement that is shown in Example 4-8.

*Example 4-8 SQL statement to calculate table size*

---

```
SELECT
    SUBSTR(TABNAME,1,20) AS tabname,
    COL_OBJECT_P_SIZE,
    DATA_OBJECT_P_SIZE,
    INDEX_OBJECT_P_SIZE,
    (COL_OBJECT_P_SIZE + DATA_OBJECT_P_SIZE +
     INDEX_OBJECT_P_SIZE) AS total_size
FROM SYSIBMADM.ADMINTABINFO
WHERE tabname LIKE 'mytable%'
WITH UR
```

---

The COL\_OBJECT\_P\_SIZE element represents the user table size. The DATA\_OBJECT\_P\_SIZE element represents the size of the metadata, including column compression dictionaries and synopsis tables. Both of these values must be combined with the value of the INDEX\_OBJECT\_P\_SIZE element to determine the total size of the table.

To determine how much a table is compressed, look at the PCTPAGESSAVED value in SYSCAT.TABLES. For a column-organized table, the PCTPAGESSAVED value is based on an estimate of the number of data pages that are needed to store the table in decompressed row organization, so the PCTPAGESSAVED value can be used to compare compression ratios between row-organized and column-organized tables.

The unique aspect to understanding the compression factor for column-organized tables is the degree to which a particular column's values were encoded as part of approximate Huffman encoding. The new PCTENCODED column in the SYSCAT.COLUMNS catalog view represents the percentage of values that are encoded as a result of compression for a column in a column-organized table.

If the overall compression ratio for your column-organized table is too low, check this statistic to see whether values in specific columns were left decompressed.

**Note:** If you see many columns with a very low value (or even 0) for PCTENCODED, the utility heap might have been too small when the column compression dictionaries were created.

You might also see very low values for columns that were incrementally loaded with data that was outside of the scope of the column compression dictionaries. Those dictionaries are created during the initial load operation. Additional page-level dictionaries might be created to take advantage of local data clustering at the page level and to further compress the data.

### **Performance considerations for compression**

There are two performance considerations to loading data into column-organized tables: the amount of memory that is available to the utility heap (UTIL\_HEAP\_SZ) and the sorting of input data.

UTIL\_HEAP\_SZ is set to AUTOMATIC by default, which means that the self-tuning memory manager manages the heap and uses the available database memory as required. Ensure that there is sufficient database memory available before loading data.

If possible, presort the data by columns that are frequently referenced by predicates that filter the table data or columns that are used to join highly filtered dimension tables. Presorting the input data on those columns can improve compression ratios and the likelihood of data skipping by ensuring that particular column value is clustered on fewer pages. This can result in improved performance.

## **4.4.3 Reducing load times**

Loading a column-organized table can take longer than loading an equivalent amount of data into a row-organized table because of additional processing. Although the LOAD and BUILD phases cannot be shortened, it is possible to reduce the duration of the ANALYZE phase by creating the column compression dictionary for a table with only a portion of the input data.

The most challenging task to reducing load times is obtaining a representative sample of data for the table. Without a representative sample, the column compression dictionaries are not as effective as though they were based on the full set of input data, which can result in poorer compression.

Example 4-9 illustrates the steps to reduce the load time for a large table. The full set of data is in `all_data.csv`, when a representative sample of data is in `subset_data.csv`. The first command runs the ANALYZE phase only and generates the column compression dictionary. The second command loads the data and skips the ANALYZE phase because the dictionary is created.

*Example 4-9 LOAD commands to generate a dictionary on a subset of data to reduce load time*

---

```
LOAD FROM subset_data.csv OF DEL REPLACE RESETDICTIONARYONLY INTO mytable;  
LOAD FROM all_data.csv OF DEL INSERT INTO mytable;
```

---

The subset of data can be created by either copying a subset of the full file, or by generating a sample of the original source to get a subset. To illustrate a sampling method, Example 4-10 shows the commands to load a column-organized table, building the column compression dictionary by sampling an existing table.

*Example 4-10 Loading a column-organized table with a column-compression dictionary built from a sampling of cursor data*

---

```
DECLARE c1 CURSOR FOR SELECT * from my_row_table TABLESAMPLE BERNOULLI (10);  
DECLARE c2 CURSOR FOR SELECT * from my_row_table;  
LOAD FROM c1 OF CURSOR REPLACE RESETDICTIONARYONLY INTO my_col_table;  
LOAD FROM c2 OF CURSOR INSERT INTO my_col_table;
```

---

## 4.5 Managing a column-organized data mart

With the simplicity of BLU Acceleration, there are far fewer considerations to managing a column-organized data mart. Chapter 7, “Monitoring” on page 105 and Chapter 9, “Workload management” on page 145 provide all the information for monitoring and managing the concurrency of the database. This section focuses on understanding the usage of BLU Acceleration in query plans and database maintenance of a column-organized data mart.

### 4.5.1 Query execution plans and the new CTQ operator

With any new feature, the first question DBAs often ask is, “How do you know whether the feature is working the way that it should?” For BLU Acceleration, the answer should be obvious: the optimal performance of analytic queries. However, additional information is available if you look at the explain plan for queries on column-organized tables.

A new CTQ operator for query execution plans indicates the transition between column-organized data processing (BLU Acceleration) and row-organized processing. All operators that appear below the CTQ operator in an execution plan are optimized for column-organized tables.

A good execution plan for column-organized tables has the majority of operators below the CTQ operator and only a few rows flowing through the CTQ operator. With this type of plan, as shown in Figure 4-3, DB2 is working directly on encoded data and optimizing with BLU Acceleration.

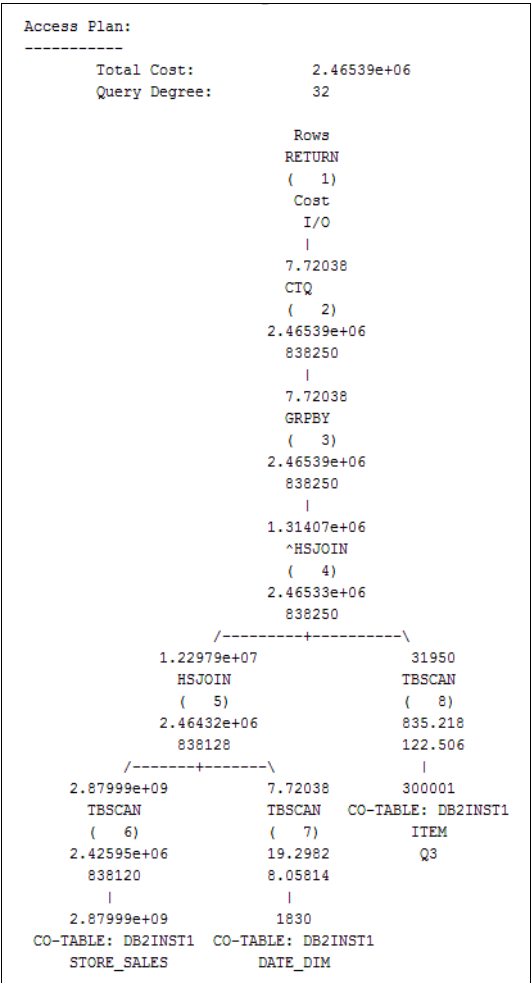


Figure 4-3 An example of a good query execution plan for a column-organized table

Access methods and join enumeration are simplified with BLU Acceleration, but join order is still dictated by cost, so cardinality estimation is still important.

Collecting column group statistics, for example, is helpful for both column-organized and row-organized tables. Information primary key constraints and unique constraints, where applicable, can also provide the optimizer with additional information not otherwise available that results in good query execution plans for BLU accelerated tables.

Query execution plans can be generated by using engine tools (**db2exp1n** and **db2exfmt**). In addition, Optim Performance Manager and Optim Query Workload Tuner can explain statements. For more information, see *IBM Optim Performance Manager for DB2 for Linux, UNIX, and Windows*, SG24-7925.

## 4.5.2 Database maintenance

The automatic statistics collection is enabled by default during a load operation into a column-organized table. In addition, automatic statistics collection is enabled by default for a database containing either row-organized or column-organized tables. As a result, an explicit **RUNSTATS** operation might not be necessary. The only reason to run **RUNSTATS** explicitly is if you cannot wait for automatic statistics collection to run.

Space is automatically reclaimed for column-organized tables when DB2\_WORKLOAD=ANALYTICS or if AUTO\_REORG\_ON is in the database configuration settings. Only if neither of these conditions are true is manual intervention required to reclaim free space, which is done by using the **REORG TABLE... RECLAIM EXTENTS** command. To check the amount of space that is available to reclaim, use the RECLAIMABLE\_SPACE column in the ADMINTABINFO administrative view. Example 4-11 shows a sample SQL statement to check for reclaimable space.

*Example 4-11 SQL statement to check reclaimable space*

---

```
SELECT SUBSTR(tabname,1,20) AS tabname, RECLAIMABLE_SPACE
FROM SYSIBMADM.ADMINTABINFO
WHERE tabname LIKE 'mytable%'
ORDER BY tabname WITH UR
```

---

All other maintenance activities that involve column-organized tables, such as backup and restore, are identical to those that involve row-organized tables.





## Row-based data store

DB2 for Linux, UNIX, and Windows offers a comprehensive platform for building data mart and warehouse solutions that are powerful yet flexible and cost effective. DB2 can create a warehouse database on a single server with uniprocessor or multiple processors, or across multiple servers. Both of these scale-up and scale-out solutions allow you to build a dynamic, scalable, and large database environment.

Chapter 4, “Column-organized data store with BLU Acceleration” on page 53 introduced BLU Acceleration, a new columnar technology in DB2 that provides an order of magnitude of benefits for analytic workload performance, storage saving, and time to value. It is also a scale-up solution where BLU Acceleration scales linearly and seamlessly when more processors and memories are added to the single server.

Traditionally, DB2 also has row-organized tables where data is inserted and stored one row after another. A row-based data store is more suitable for a warehouse environment with a high volume of transactions. DB2 provides a capability to scale up a row-based data store on a single server with the *intrapartition parallelism* feature. As the data volume continues to grow, a single server is not sufficient to meet the increased business demand. DB2 row-based data store offers a scale-out solution where additional servers can be easily added to the warehouse environment. This important feature is known as the *Database Partitioning Feature (DPF)* or *interpartition parallelism*.

This chapter describes the traditional row-based warehouse solution in DB2. It starts with the Database Partitioning Feature, then describes the intrapartition parallelism feature. Finally, it describes some other technologies in DB2 that you can use to build a high-performance data warehouse environment.

## 5.1 Scale-out solution: Database Partitioning Feature

DB2 uses database partitioning to provide a scale-out solution. It allows you to support large database and use large hardware environments efficiently. *Database partitioning* refers to splitting a database into separate physical partitions, which can be distributed across multiple servers, but still appear to the user and database administrator as a single-image database. This feature is known as Database Partition Feature (DPF) or interpartition parallelism.

In a DPF environment, a database is divided into partitions. By searching these database partitions in parallel, elapsed times for queries can be dramatically reduced. Data is distributed across multiple database partitions, enabling database scaling into terabytes.

SQL operations are performed in parallel across all partitions and therefore operate much faster. DPF provides parallel everything such that all transactions (selects, updates, deletes, and inserts), and utility operations, all operate in parallel across all partitions.

The architecture design divides and runs the workload to enhance scalability through enabling partitioned workloads to have their own private data and computing resources. In this way, near-linear scalability is achieved as resources across data partitions are not shared, and there is no central locking of resources.

The following sections give an overview of some of the concepts and preferred practices when applying the Database Partitioning Feature.

### 5.1.1 Partition

A database partition is part of a database that consists of its own data, indexes, configuration files, and transaction logs. A database partition is sometimes called a *node* or a *database node*. It can be either logical or physical. Logical partitions are on the same physical server and can take advantage of the symmetric multiprocessor (SMP) architecture. These partitions use common memory, processors, disk controllers, and disks. Physical partitions consist of two or more physical servers, and the database is partitioned across these servers. Each partition has its own memory, processors, disk controllers, and disks.

User interaction occurs through one database partition, which is known as the coordinator node for that user. The coordinator runs on database partition to which the application is connected. Any database partition can be used as a coordinator node.

### 5.1.2 Partition group

A partition group is a logical layer that allows the grouping of one or more partitions to perform operations on all the partitions in the group. A database partition can belong to more than one partition group. When a database is created, DB2 creates three default partition groups that cannot be dropped:

- ▶ **IBMDEFAULTGROUP**

This is the default partition group for any table you create. It consists of all database partitions as defined in the `db2nodes.cfg` file. This partition group cannot be modified. Table space `USERSPACE1` is created in this partition group.

- ▶ **IBMTEMPGROUP**

This partition group is used by all system temporary tables. It also consists of all database partitions as defined in the `db2nodes.cfg` file. Table space `TEMPSPACE1` is created in this partition.

- ▶ **IBMCATGROUP**

This partition group contains the catalog tables (table space `SYSCATSPACE`), and thus it includes only the database catalog partition. This partition group cannot be modified.

To create database partition groups, you can use the `create database partition group` statement. This statement creates the database partition group in the database, assigns database partitions that you specified to the partition group, and records the partition group definition in the database system catalog tables.

Example 5-1 shows the creation of a `pgall` partition group on all partitions that are specified in the `db2nodes.cfg` file.

*Example 5-1 Creating a partition group across all database partitions*

---

```
create database partition group pgall on all dbpartitionnums
```

---

### 5.1.3 Table spaces in a DPF environment

A table space can be created in specific partitions by associating it with a partition group. The **CREATE TABLESPACE** statement with the **IN DATABASE PARTITION GROUP** clause can be used for this purpose. This allows users to have flexibility as to which partitions store their tables. Example 5-2 shows a table space mytbls, which spans partitions 0, 1, 2, and 3.

*Example 5-2 Creating a table space in the database space pg0123.*

---

```
create large tablespace mytbls in database partition group pg0123
```

---

In a DPF environment, a table can be in one or more database partitions depending on the table space where the table is created and the associated database group. When a table is in a database partition group consisting of multiple partitions, rows are stored in one partition, and other rows are stored in other partitions.

### 5.1.4 Partitioning keys

A partitioning key is a column (or group of columns) that is used to determine the partition in which a particular row of data is stored. A partitioning key is defined on a table by using the **CREATE TABLE** statement, as shown in Example 5-3.

*Example 5-3 Partitioning key*

---

```
create table mytable (col1 int, col2 int, col3 int, col4 char(10))  
    in mytbls1  
    distributed by hash (col1, col2, col3)
```

---

Figure 5-1 on page 75 shows another example of a DPF environment with four partitions (0, 1, 2, and 3). Two tables, customer and store\_sales, are created. Both have a partitioning key named cust\_id. The value in cust\_id is hashed to generate a partition number, and the corresponding row is stored in the relevant partition.

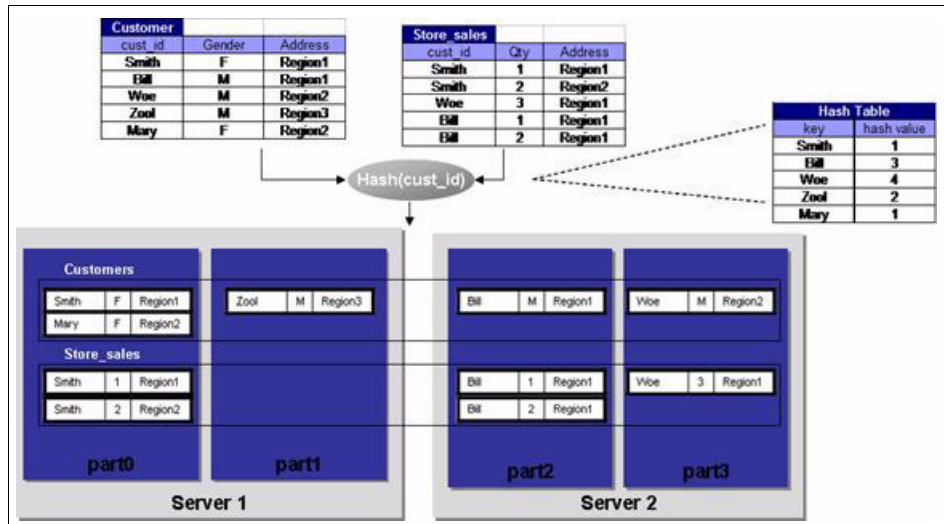


Figure 5-1 Example of a DPF environment

## 5.1.5 Partition maps

When a database partition group is created or modified, a partitioning map is automatically created by DB2. A partitioning map, with a partitioning key and a hashing algorithm, is used by DB2 to determine which database partition in the database partition group stores a row of data.

Where a new row is inserted is determined by hashing the partitioning key value of the rows to an entry in the partitioning map, which contains the partition number to use, as shown in Figure 5-2.

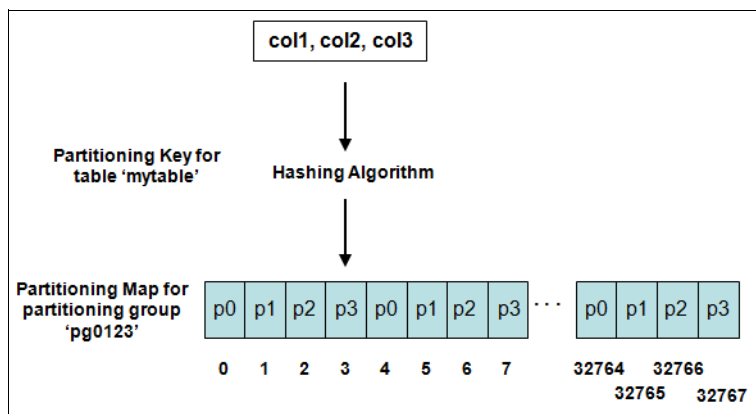


Figure 5-2 Hash partitioning map

Partition group pg0123 is defined on partitions 0, 1, 2, and 3. An associated partitioning map is automatically created. This is an array with 32768 entries containing the values 0, 1, 2, 3, 0, 1, 2, 3.... Partition numbers are stored in round-robin fashion by default, although this can be changed. Table mytable is created with a partitioning key consisting of columns col1, col2, and col3. For each row, the partitioning key column values are passed to the hashing algorithm, which returns an output number 0 - 32767. This number corresponds to one of the entries in the partitioning map array that contains the value of the partition number where the row is to be stored. If the hashing algorithm had returned an output value of 7, the row would have been stored in partition p3.

The partitioning map is an internally generated array containing either 32768 entries for multiple-partition database partition groups, or a single entry for single-partition database partition groups. The partition numbers of the database partition group are specified in a round-robin fashion.

Although it is possible to change the partitioning map and correct data skew or make allowances for clustered servers of different processor performance, in practice, a large data warehouse should be based on balanced configuration principles and have many tables that are partitioned across many identical balanced partition units. It is likely to be impractical to correct for data skew for a subset of tables and likely increase skew for other tables with different partitioning keys in the same partition group.

### 5.1.6 Choosing partitioning keys

Given a particular warehouse workload and configuration in a DPF environment, one of the bigger performance factors is the choice of partitioning keys for the tables. There might not be, however, one absolute and obvious best set of them, as one set might optimize queries and another set optimize other queries, so you must look at the overall workload and know what is most important to optimize. Try various alternatives to get the best results.

The following list details a few preferred practices for partitioning a key selection:

- ▶ Do the following actions:
  - Include frequently used join columns.
  - Spread data evenly across partitions.
  - Have a broad range domain.
  - Use integer, which is more efficient than character and is more efficient than decimal.
  - Use DB2 Design Advisor.

- ▶ Do not do the following actions:
  - Use LOB or LONG fields.
  - Have a unique index or primary key unless it is a superset of the PK.
  - Allow ALTERations of the PK.
  - Update columns.

**Important:** DPF partitioning is based on hashing, with the intent to spread rows evenly over partitions and the available hardware resources to maximize performance. It is not a preferred practice to choose partitioning keys in an attempt to implement range partitioning, which should be accomplished by using table partitioning (also known as range partitioning).

The following list details other considerations that are involved in choosing partitioning keys:

- ▶ After a table is created as partitioned, you cannot directly change its partitioning key. To change the key, try one of these approaches:
  - Unload the table to a file, drop and re-create the table with the new partitioning key, and reload the table.
  - Create a version of the table with a temporary name and the new partitioning key, load the new table from the old one (the fastest way is by running **declare mycursor for select \* from t1**, followed by **load from mycursor of cursor replace into t1\_new**), drop the old table, rename the new table to the real name, and re-create indexes and redo other steps as when the table was originally created.
- ▶ With **ALTER TABLE**, you can add or drop a partitioning key, but only for a non-partitioned table.
- ▶ The columns in any unique or primary key constraint that is defined on the table must be a superset of the partitioning key.
- ▶ Avoid uneven distribution of rows across partitions by choosing partitioning key columns with high cardinality. If this cannot be achieved, try to use columns with uniformly distributed values. After a table is populated, you can check how many of its rows are in each partition, and how many of its rows map to each entry in the partitioning map (for possible redistribution purposes) by running queries to analyze the table contents. Unless you have extremely large tables, differences of a few percent in cardinalities per partition can be ignored.

- ▶ Partitioning keys should not include columns that are updated frequently. Whenever a partitioning key value is updated, DB2 might need to drop the row and reinsert it into a different partition, as determined by hashing the new partitioning key value.
- ▶ Unless a table is not critical or you have no idea what a good partitioning key choice is, you should not let the partitioning key be chosen by default. For the record, the default partitioning key is the first column of the primary key, and if there is none, it is the first column that has an eligible data type.
- ▶ Ensure that you understand collocation and the different join types. For more information, see the *DB2 Partitioning and Clustering Guide*, SC27-2453.

Collocated tables must fulfill the following requirements:

- Be in the same database partition group (one that is not being redistributed). (During redistribution, tables in the database partition group might be using different partitioning maps; they are not collocated.)
- Have partitioning keys with the same number of columns.
- Have the corresponding columns of the partitioning key be partition compatible.
- Be in a single partition database partition group, if not in the same database partition group, that is defined on the same partition.

### 5.1.7 Concluding remarks

DB2 with DPF provides a scale-out solution for building a data mart or warehouse that can grow theoretically unlimited. It supports various hardware configurations such as symmetric multiprocessors (SMP) on a single server, and a cluster of connected servers. As the data grows or the demand of the workload increases, additional resources can easily be deployed. For example, you can add a physical server to the configuration, and consequently one (or more) database partitions on the new physical server. The database can then be spread across these new partitions and the execution of a query can take advantage of this new configuration immediate.



## 5.2 Scale-up solution: Intrapartition parallelism

Within a single server with symmetric multiprocessors (SMP), DB2 also can parallelize query execution by dividing the work among subagents running on different hardware threads or processors. Thus, processor resources in the SMP environment can be better use and query elapsed time can be reduced. It is especially beneficial to long running warehouse queries. More importantly, it does not require any form of data partitioning. This capability is known as intrapartition parallelism or *multi-core parallelism*. It can be enabled by a configuration parameter that is called `intra_parallel`.

Intrapartition parallelism provides a scale-up solution where processor and memory resources can easily be added to the single server. The new columnar technology, BLU Acceleration, belongs to this scale-up solution. In fact, DB2 has been supporting intrapartition parallelism for row-based data for many releases.

Figure 5-3 shows a query that is broken into four pieces that can be run in parallel, with the results returned more quickly than if the query were run in serial fashion. The pieces are copies of each other.

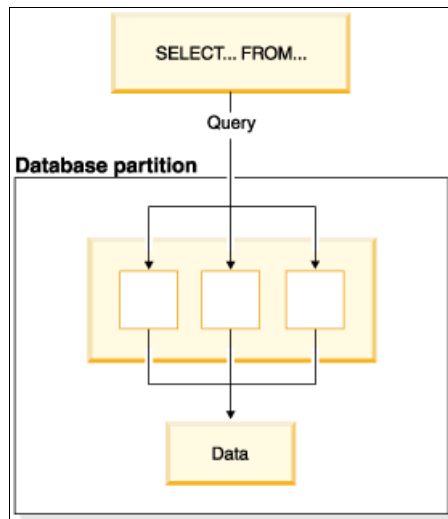


Figure 5-3 Intrapartition parallelism

DB2 utilities can also take advantage of intrapartition parallelism, as in the following examples:

- ▶ The load utility takes advantage of multiple processors for tasks such as parsing and formatting data.
- ▶ During index creation, the scanning and subsequent sorting of the data can occur in parallel with intrapartition parallelism.
- ▶ Backing up and restoring data can use both I/O parallelism and intrapartition parallelism when performing backup and restore operations.

It is worth mentioning that in DB2 10.1 the scalability of intrapartition parallelism has improved through improved load balancing, more efficient parallelization techniques, and reduced latch contention. For example, DB2 10.1 introduces a new operator called REBAL. This operator redistributes rows during the query execution to ensure that all subagents do equal work.

Individual applications or workloads can dynamically throttle the degree of parallelism to optimize performance for the types of queries being run. This is important for warehouse environments with transactional workloads. A transactional workload typically consists of short insert, update, and delete transactions, and does not benefit from parallelization. In some cases, the impact of intrapartition parallelism even impacts the throughput. Conversely, a data warehouse workload benefits greatly from parallelization, as its queries are often processor-intensive and long running. Starting with DB2 10.1, you can now configure different workloads to use the optimal parallelism settings (ON, OFF, and query degree) with WLM workload control to achieve the optimal performance.

Finally, DB2 allows you to combine both scale-up and scale-out solutions. In a DPF environment (scale-out), intrapartition parallelism (scale-up) can be enabled in each database partition. A single database operation (like a query) is first subdivided into subparts to be run on individual database partitions; within each partition, a subpart can be further divided into smaller parts to run on the data in this partition, as illustrated in Figure 5-4 on page 81. Combining both DPF and intrapartition parallelism makes DB2 a powerful and flexible platform to support large data warehouse databases.

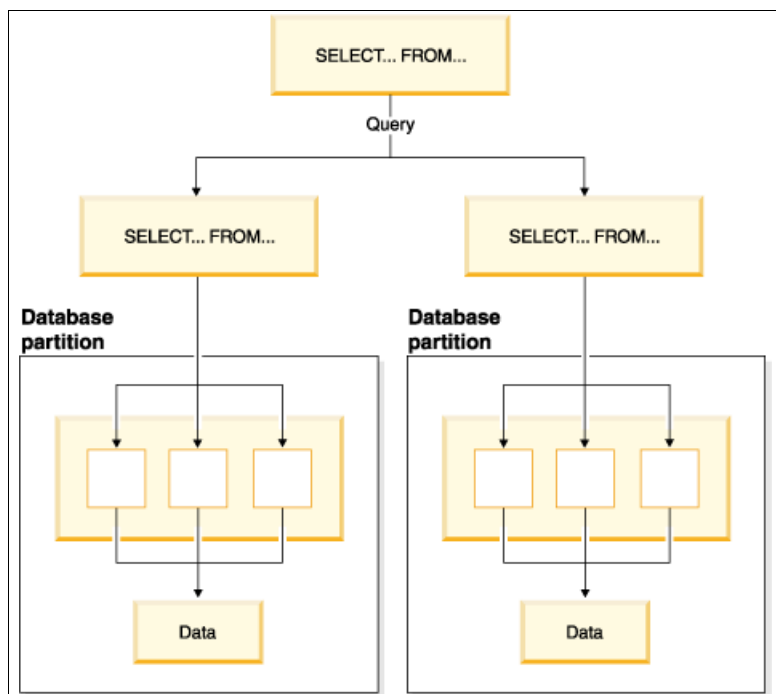


Figure 5-4 DPF and intrapartition parallelism

## 5.3 Other features for row-based data warehouse

The scale-up and scale-out solutions provides unprecedented scalability to a row-based data store. Many other features that can improve performance and reduce the cost of ownership (TCO) are also built into DB2. This section highlights a number of features that are highly recommended for all data warehouse implementations.

### 5.3.1 Compression

The DB2 Storage Optimization Feature enables you to transparently compress data on disk to decrease disk space and storage infrastructure requirements. Because disk storage systems are often the most expensive components of a database solution, even a small reduction in the storage subsystem can result in substantial cost savings for the entire database solution. This is especially important for a data warehouse solution, which typically has a huge volume of data.

## Row compression

DB2 uses a variant of the Lempel-Ziv algorithm to apply compression to each row of a table. Log records are also compressed. Savings are extended to backup disk space, racks, cables, floor space, and other disk subsystem peripherals.

Because compressed rows are smaller, not only do you need fewer disks, but your overall system performance might be improved. By storing compressed data on disk, fewer I/O operations need to be performed to retrieve or store the same amount of data. Therefore, for disk I/O-bound workloads, the query processing time can be noticeably improved.

DB2 stores the compressed data on both disk and memory, reducing the amount of memory that is consumed and freeing it up for other database or system operations.

To enable row compression in DB2, when tables are created, you can specify the **COMPRESS YES STATIC** option. It can also be enabled for an existing table by using the **ALTER TABLE** command. If you alter the existing table to enable compression, a **REORG** must be run to build the dictionary and compress the existing data in the table. For more details, see the DB2 10.5 information center at the following website:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.welcome.doc/doc/welcome.html>

## Adaptive row compression

DB2 further enhances classical row compression by allowing an advanced row compression technique that uses table-level and page-level compression dictionaries. DB2 can reach higher compression ratios than ever before, and maintain them over time without table reorganization. This enables more granular, adaptive, and dynamic updates, enhances query performance, and improves data availability. Adaptive Row Compression is available in DB2 10.1 and later releases. Also, starting from DB2 10.1, it is the default compression method in DB2; if you specify **COMPRESS YES**, Adaptive Row Compression is used by default. (The key word **STATIC** must be used, for example, **COMPRESS YES STATIC**, if only the classic row compression is needed.)

## XML compression

The verbose nature of XML implies that XML fragments and documents typically use much disk space. DB2 stores XML data in a parsed hierarchical format, replacing tag names (for example, employee) with integer shorthand. Repeated occurrences of the same tags are assigned the same shorthand. Storing text-rich tags using integer shorthand reduces space consumption and assists with higher performance when querying data. Moreover, the XML tag parsing, such as data row compression, is transparent to users and applications.

## Index compression

Indexes, including indexes on declared or created temporary tables, can be compressed in DB2 to reduce storage costs. This is especially useful for large data warehouse environments.

By default, index compression is enabled for compressed tables, and disabled for decompressed tables. You can override this default behavior by using the **COMPRESS YES** option of the **CREATE INDEX** statement. When working with existing indexes, use the **ALTER INDEX** statement to enable or disable index compression; you must then perform an index reorganization to rebuild the index.

Processor usage might increase slightly as a result of the processing that is required for index compression or decompression. If this is not acceptable, you can disable index compression for new or existing indexes.

## 5.3.2 Multidimensional data clustering

Multidimensional data clustering (MDC) is a method of organizing and storing data along multiple dimensions. Organizing data in this manner can greatly improve the performance of certain query types and reduce reorg and index maintenance.

MDC enables a table to be physically clustered on more than one key (or dimension) simultaneously. MDC tables are organized physically by associating records (or row) with similar values for the dimension attributes in an extent (or block). DB2 maintains this physical layout efficiently and provides methods of processing database operations that provide significant performance improvements.

Figure 5-5 presents a conceptual diagram of multidimensional clustering along three dimensions: region, year, and color.

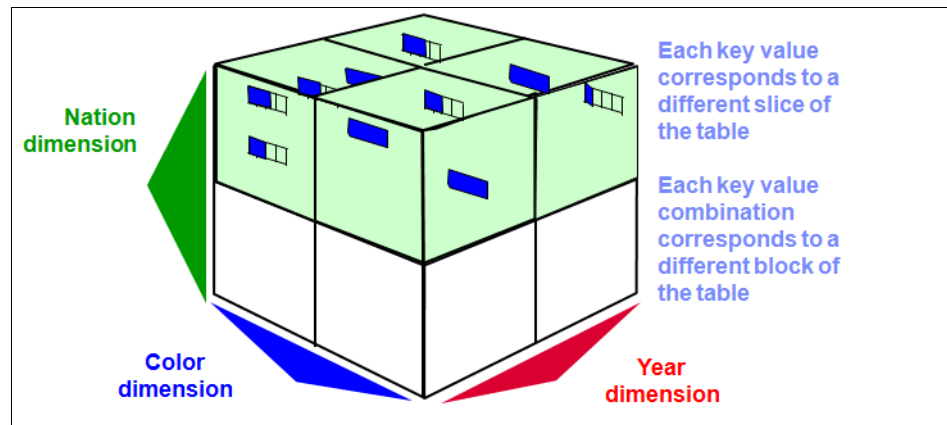


Figure 5-5 MDC conceptual diagram

Each block contains only rows that have the same unique combination of dimension values. The set of blocks that have the same unique combination of dimension values is called a *cell*. A cell might consist of one or more blocks in the MDC table. As shown in Figure 5-5, a cell can be described as a combination of unique values of year, nation, and color, such as 1997, Canada, and Blue. All records that have 1997 as year, Canada as nation, and blue as the color are stored in the same extents of that table.

With MDC tables, clustering is ensured. If an existing block satisfies the unique combination of dimension values, the row is inserted into that block, assuming there is sufficient space. If there is insufficient space in the existing blocks, or if no block exists with the unique combination of dimension values, a new block is created.

Example 5-4 shows the DDL for creating an MDC table. The **ORGANIZE** keyword is used to define an MDC table.

*Example 5-4 MDC example*

---

```
CREATE TABLE sales (
  cust_id INTEGER,
  year    CHAR(4),
  nation  CHAR(15),
  color   CHAR (12),
  amount  INTEGER)
ORGANIZE BY (nation,color,year)
```

---

Starting with DB2 10.5, you must specify **ORGANIZE BY ROW USING DIMENSION** keywords.

MDC introduced a new type of index that is called a *block index*. When you create an MDC table, the following two kinds of block indexes are created automatically:

- ▶ Dimension block index

A dimension block index per dimension contains pointers to each occupied block for that dimension.

- ▶ Composite block index

A composite block index contains all columns that are involved in all dimensions that are specified for the table, as shown in Figure 5-6. The composite block index is used to maintain clustering during insert and update activities. It can also be used for query processing.

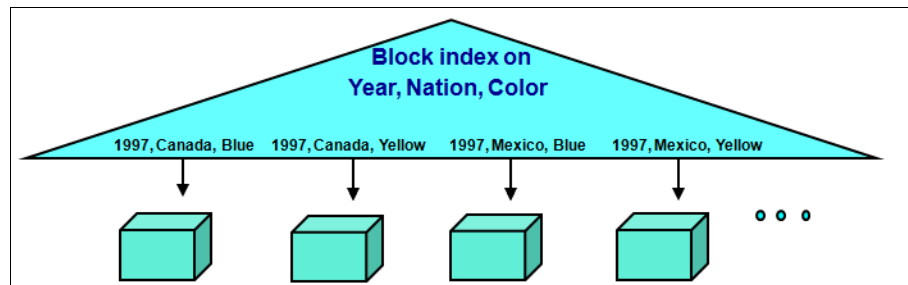


Figure 5-6 Block index

With MDC, data is organized on the disk based on dimensions. Queries can skip parts of the table space that the optimizer has determined do not apply. When data is inserted, it is automatically put in the correct place so that you no longer need to reorganize the data. In addition, because one index entry represents the entire data page (versus having one index entry per row with traditional indexes), MDCs reduce the overall size of the space that is required for the indexes. This reduces disk requirements and produces faster queries because of the reduced amount of I/O needed for a query. MDCs also improve delete performance because DB2 now has to drop only a few data pages. Inserts are also faster because DB2 rarely has to update an index page (only the data page).

## Performance consideration

The performance of an MDC table depends upon the correct choice of dimensions and the block (extent) size of the table space for the data and application workload. A poor choice of dimensions and extent size can result in low disk storage use and poor query access and load utility performance.

In choosing dimensions, you must identify the queries in the existing or planned workloads that can benefit from multidimensional clustering. For existing applications, you can use DB2 Design Advisor to analyze the workload and recommended dimensions for the MDC tables. For a new table or database, you need a good understanding of the expected workload. The following dimension candidates are typical:

- ▶ Columns in range or equality or IN-list predicates
- ▶ Columns with coarse granularity
- ▶ Roll-in and roll-out of data
- ▶ Columns that are referenced in **GROUP BY** and **ORDER BY** clauses
- ▶ Foreign key columns or join clauses in fact table of a star schema database
- ▶ Combinations of the above

Extent size is related to the concept of cell density, which is the percentage of space that is occupied by rows in a cell. Because an extent contains rows only with the same unique combination of dimension values, significant disk space can be wasted if dimension cardinalities are high (for example, when there is a dimension with unique values that would result in an extent per row).

Defining small extent sizes can increase cell density, but increasing the number of extents per cell can result in more I/O operations, and potentially poorer performance when retrieving rows from this cell. However, unless the number of extents per cell is excessive, performance should be acceptable. If every cell occupies more than one extent, it can be considered excessive.

At times, because of data skew, cells occupy many extents when others occupy a small percentage of the extent. Such cases signal a need for a better choice of dimension keys. Currently, the only way to determine the number of extents per cell requires the DBA to run appropriate SQL queries or run **db2dart**.

Performance might be improved if the number of blocks can be reduced by consolidation. Unless the number of extents per cell is excessive, this situation is not considered an issue.

Multidimensional clustering is a unique capability that is targeted for large database environments, providing an elegant method for flexible, continuous, and automatic clustering of data along multiple dimensions. The result is improvement in the performance of queries, and a reduction in the impact of data maintenance operations (such as reorganization) and index maintenance operations during insert, update, and delete operations.



### 5.3.3 Range partitioning

Table partitioning (TP), often known as range partitioning (RP), is a data organization scheme where table data is divided across multiple storage objects, called *data partitions* (not to be confused with database partitions), according to values in one or more table columns. These storage objects can be in different table spaces, in the same table space, or in a combination of both.

DB2 supports data partitions or data ranges based on various attributes. Commonly used partitioning schemes cluster together data partitions by date, such as by year or month, or have numeric attributes for partitioning. For example, records with IDs 1 - 1,000,000 are stored in one data partition, IDs 1,000,000 - 2,000,000 in another data partition, and so on. Or, for example, records for clients with names starting with A - C can be in one data partition, D - M in the second data partition, N - Q in a third data partition, and R - Z in the last data partition.

DB2 provides flexibility for creating partitioned tables. For example, if there is one year of data, it can be partitioned by date so that each quarter is in a separate data partition. The create table syntax in Example 5-5 shows how this can easily be done.

*Example 5-5 Create table syntax for range partitioning*

---

```
CREATE TABLE orders(id INT, shipdate DATE, ...)
PARTITION BY RANGE(shipdate)
(
  STARTING '1/1/2006' ENDING '12/31/2006'
  EVERY 3 MONTHS)
```

---

This results in a table being created with four data partitions, each with three months of data, as shown in Figure 5-7.

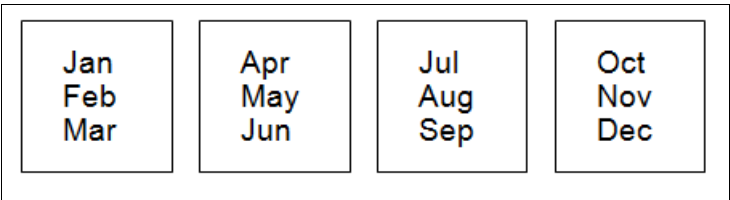


Figure 5-7 Table partitions

Table partitioning provides the following benefits:

- Improved manageability

DB2 allows the various data partitions to be administered independently. For example, you can choose to back up and restore individual data partitions instead of entire tables. This enables time-consuming maintenance operations to be segmented into a series of smaller operations.

- Increased query performance

The DB2 Optimizer is data partition aware. Therefore, during query execution, only the relevant data partitions are scanned. This eliminates the need to scan data partitions that are not impacted by the query, and can result in improved performance, as illustrated in Figure 5-8.

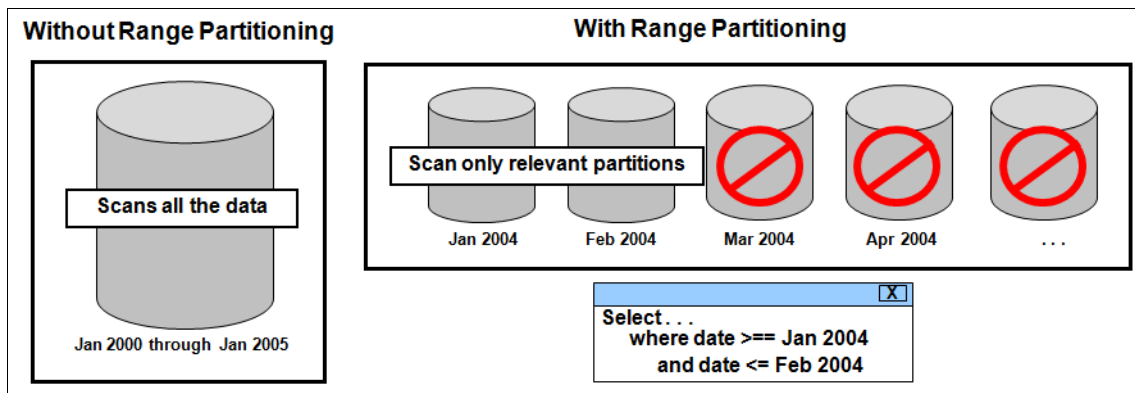


Figure 5-8 Scanning only relevant partitions

- Fast roll-in/roll-out

DB2 allows data partitions to be easily added or removed from the table without having to take the database offline. This ability can be useful in a data warehouse environment where there is the need to load or delete data to run decision-support queries. For example, a typical insurance data warehouse might have three years of claims history. As each month is loaded and rolled in to the data warehouse, the oldest month can be archived and removed (rolled out) from the active table. This method of rolling out data partitions is also more efficient, as it does not need to log delete operations, which is the case when deleting specific data ranges.

- Better optimization of storage costs

Table partitioning enables better integration with hierarchical storage models. By using only the fastest and most expensive storage hardware for only the most active data partitions, DB2 allows optimization of the overall storage costs and improves performance. If most queries run against only the last three months of data, there is an option to assign slower and less expensive storage hardware to the older data.

- Larger table capacity

Without partitioning, there are limits to the maximum amount of data that a storage object, and hence a table, can hold. However, by dividing the contents of the table into multiple storage objects or data partitions, each capable of supporting as much data as in a non-partitioned table, you can effectively create databases that are virtually unlimited in size.

Starting with DB2 9.7, there are indexes that see rows of data across all partitions in a partitioned table (known as *nonpartitioned indexes*), or you can have the index itself partitioned such that each data partition has an associated index partition.

Using partitioned indexes that match the underlying partitioned tables in a large data warehouse environment can reduce the impact that is involved in attaching or detaching partitions.

When a partition is attached to a table, a **SET INTEGRITY** command must be run before the data becomes visible. Among other tasks, the **SET INTEGRITY** command updates any existing non-partitioned indexes that can be a lengthy and use a considerable amount of log space.

When the new partition is attached to a table with partitioned indexes and indexes are already built that match these indexes, the **SET INTEGRITY** statement does not incur the performance and logging impact that is associated with index maintenance.

Table partitioning is similar to MDC in that it enables rows with similar values to be stored together. However, the following TP characteristics distinguish it from MDC:

- TP supports partitioning a table into data partitions along a single dimension. A common design is to create a data partition for each month. MDC supports defining multiple dimensions.
- With TP, the user can manually define each data partition, including the range of values to include in that data partition. MDC automatically defines a cell (and creates blocks to store data for that cell) for each unique combination of MDC dimension values.

- ▶ Each TP partition is a separate database object (unlike other tables, which are a single database object). TP supports attaching and detaching a data partition from the TP table. A detached partition becomes a regular table. Also, each data partition can be placed in its own table space, if wanted.
- ▶ Essentially, the distinct TP benefit is related to adding or removing large numbers of rows from a table, that is, roll in and roll out. If you are familiar with the usage of Union All View (UAV) to partition history tables on date, TP can be considered an analogous but superior solution.



## Data movement and transformation

This chapter provides information about some of the commonly used utilities in DB2 for Linux, UNIX, and Windows tools that are used for data movement. This chapter also describes the data movement and transformation capabilities in DB2 Warehouse SQL Warehousing Tool (SQW) that provides these services.

## 6.1 Introduction

Moving data across different databases is a common task in IT departments. Moving data mainly refers to the task of copying data from a source system to a target system, not necessarily removing it at the source location. You can then process the data (for example, export it to a file) either remotely, which is a valid solution for infrequent processing or reasonable amounts of data. You can also move the data to the most suitable place for the overall processing (for example, move data from one system to another). For example, it is common to copy production data (or at least a subset of it) to a test system.

Typical DB2 data movement tasks involve three steps:

1. Exporting the data from the source database into a temporary data exchange file in binary or text format
2. Moving the generated file between systems
3. Importing or loading the data from the file into the target database.

This section describes some of the most common methods and how they are used in moving data into a table and database.

### 6.1.1 Load

Using the *load* utility to insert data is the most common practice in data warehouse environments. The load utility is used to insert the data into a table of a database after extracting that data from the same or a different database by using other utilities, such as *export*. The export and import utilities use SQL to extract and add data. The load utility is often faster than the import utility because it bypasses the DB2 SQL engine and builds physical database pages and puts them directly into the table space. These utilities can be used to move data between databases on any operating systems and versions. You can also use them to add data in files to a database.

Although the **LOAD** command is often much faster than the import utility and can load much data to a table quickly, there are some aspects of the **LOAD** command that you must consider.

If a table that you load has referential constraints to other tables, then you must run the **SET INTEGRITY** command after loading your tables. This command verifies that the referential and all other constraints are valid. If there are constraints and you do not use the **SET INTEGRITY** command, then the loaded table and its child processes might not be accessible.

You should also understand the **COPY YES/NO** parameters of the **LOAD** command. Depending on which you choose, the table space that contains the table that you load can be placed into “backup pending” status, and no access to any of the tables in that table space is allowed until a backup is done. This utility allows you to copy data between different versions of DB2 and databases on different operating systems.

## Large object and XML considerations

Large objects (LOBs) and XML columns can be imported, exported, and loaded without using the special LOB and XML parameters, but with significant limitations. For LOB data, columns are limited to 32 KB in these utilities if you do not use the LOB parameters. When exporting tables with columns having the XML data types, the XML columns are always exported to separate files from the regular relational data. Before using the **LOAD** command, read and understand how this utility handles these special data types, as described in *Data Movement Utilities Guide and Reference*, SC27-3867.

## Loading from cursor

The **CURSOR** option in the **LOAD** command allows you to connect to another database, or tables in the same database, and have the load utility select data from a table on the source and load it directly into the target. One nice attribute of this process is that you do not need to have space for an intermediate file in which to extract the table data because the data is loaded as it is extracted. This option allows you to copy data between different versions of DB2 and databases on different operating systems.

This option avoids generating the data exchange file by using the export utility, which is often a lengthy process in the case of large amounts of data. Besides, different database code pages and operating systems must be considered when moving data in and out of a database.

**Note:** This situation implies that load from cursor handles database code page consideration, but load through file does not, and must be an additional consideration for that scenario. When you are importing or exporting a DEL file, the code page for the data is assumed to be the same as that of the application running the utility. If it is different, unpredictable results might occur. When loading a DEL file, the code page for the data is assumed to be the same as that of the database.

When you specify the **FROM CURSOR** option, the load utility directly references the result set of a SQL query as the source of a data load operation, thus bypassing the need to produce a temporary data exchange file. This way, **LOAD FROM CURSOR** is a fast and easy way to move data between different table spaces or different databases. **LOAD FROM CURSOR** operations can be run on the command line and from within an application or a stored procedure by using the **ADMIN\_CMD** stored procedure.

For examples of **LOAD FROM CURSOR**, see *Fast and easy data movement using DB2's LOAD FROM CURSOR feature*, found at:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-0901fechner/>

## 6.1.2 Ingest

The *ingest* utility, which was introduced in DB2 10.1, incorporates the best features of both the load and import utilities and adds data import features. The ingest utility meets the demanding requirements of the extract, load, and transform (ELT) operations of data warehouse maintenance. It supports the need to have data that is current and also available 24x7 to make mission-critical decisions in the business environment of today.

Like the load utility, the ingest utility is fast, although not as fast as the load utility. Like the import utility, the ingest utility checks constraints and fires triggers so there is no need to use the **SET INTEGRITY** facility after running ingest utility. In addition, the table being loaded is available during loading. The ingest utility also can do continuous loading of data, so if you have data arriving constantly, you can set up the utility to continuously add data to your table, as described in 6.1.3, “Continuous data ingest” on page 96.

Furthermore, you can use ingest to update rows, merge data, and delete rows that match records in the input file.

With its rich set of functions, the ingest utility can be a key component of your ETL system:

- In near real-time data warehouses

In databases, specifically data marts and data warehouses, where data must be updated in a near real-time basis, the users cannot wait for the next load window to get fresh new data. Additionally, they cannot afford to have offline LOAD jobs running during business hours.



- To populate HADR databases

Using the load utility in HADR databases is complicated. The only way to use **LOAD** in an HADR database is to specify the **COPY YES** clause. Nevertheless, using the ingest utility in this way requires a shared file system between primary and standby servers, or manual DBA intervention to keep the HADR standby in-synch after the **LOAD**.

Ingest provides a better option in this case because it is fully logged, so HADR automatically replicates all the inserted rows into the standby server without any user intervention.

For more about the ingest utility, see *Unleashing DB2 10 for Linux, UNIX, and Windows*, SG24-8032. This section provides some of the salient points here.

The ingest utility is a high-speed and configurable client-side DB2 utility. With its multithreaded architecture, the ingest utility can pump massive amounts of data quickly and continuously into DB2 tables from files, or pipes, with minimal effect on concurrent user workload and data server resources.

The ingest utility differs from other data movement utilities in several key areas:

- With the ingest utility, queries can run concurrently against the DB2 tables that are being loaded with minimal effect on data availability. You no longer need to choose between data concurrency and availability.
- The ingest utility features high fault tolerance and recoverability.
- Unlike other data movement utilities, the ingest utility supports SQL expressions (including predicates and casting), so in addition to doing the load stage of your ETL process, the ingest utility can also participate in the transform stage.
- The ingest utility supports a range of DML statements, including **INSERT**, **UPDATE**, **DELETE**, **MERGE**, and **REPLACE**.
- Not only is the ingest utility compatible with DB2 Warehouse, but it is *database-partition-aware*, which means the utility routes inserted rows directly to the appropriate partition (rather than going through a coordinator). This approach contributes to its speed in a partitioned database environment.
- Support for SQL expressions, including predicates and casting, allows column values to be derived from more than one data field, or from an expression at run time. This feature contributes to decreased costs in the transform step because fewer scripts must be maintained to manipulate data after being exported from the OLTP database.

- ▶ The **INGEST** command can load data continuously when it is called from a scheduled script. Because the Ingest utility issues SQL statements just like any other DB2 application, changes are logged and its statements can be recovered if necessary.
- ▶ Furthermore, the ingest utility commits data frequently, and the commit interval can be configured by time or number of rows. Rejected rows can be discarded or placed into a file or table.

### 6.1.3 Continuous data ingest

Continuous data ingest (CDI) is the process of moving data into DB2 tables with no impact to running applications. The main target of this technology is to feed near-real-time data marts and data warehouses. In such environments, the up-to-date data is so critical that customers cannot wait for the next load windows to get access to newer data.

In these cases, ingest can be used to insert data into the tables without an impact to the running queries or applications. The high concurrency levels that are provided by ingest eliminate the need for an offline window to populate the database. Additionally, the performance that is provided by the tool ensures that large amounts of data are loaded with high throughput.

The CDI function is implemented by the ingest tool, a client-side utility that uses parallel sessions to insert data into the tables. You can deploy your solution on the server or client side. The tool also provides parameters to control the execution, such as defining the number of parallel threads that insert or format data. This new method is supported by single and multi-partitioned databases, pureScale environments, and HADR-enabled databases.

## Ingest architecture

Internally, the ingest architecture is made of a multi-threaded process. There are three types of threads (transporters, formatters, and flushers) that are responsible for each phase of the ingest operation, as shown in Figure 6-1.

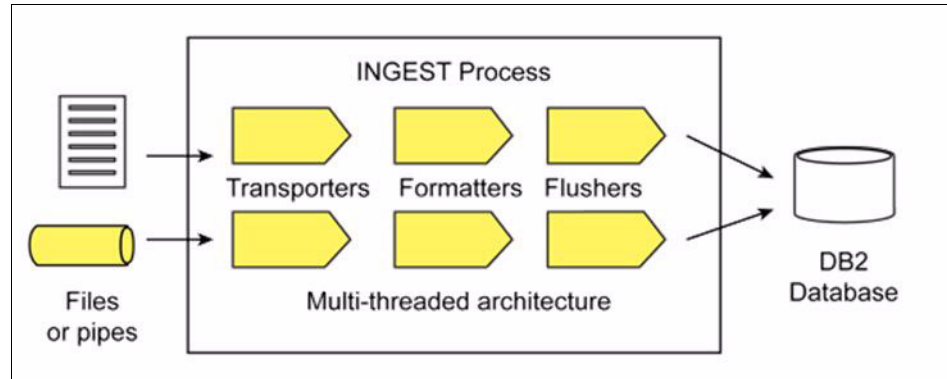


Figure 6-1 Ingest architecture

A single **INGEST** command goes through three major phases:

- Transport

The transporters read from the data source and put records on the formatter queues. For **INSERT** and **MERGE** operations, there is one transporter thread for each input source (for example, one thread for each input file). For **UPDATE** and **DELETE** operations, there is only one transporter thread.

- Format

The formatters parse each record, convert the data into the format that DB2 database systems require, and put each formatted record on one of the flusher queues for that record's partition. The number of formatter threads is specified by the **num\_formatters** configuration parameter. The default is  $(\text{number of logical processors})/2$ .

- Flush

The flushers issue the SQL statements to perform the operations on the DB2 tables. The number of flushers for each partition is specified by the **num\_flushers\_per\_partition** configuration parameter. The default is  $\max(1, ((\text{number of logical processors})/2)/(\text{number of partitions}))$ .

For more information, see *Introduction to the DB2 Continuous Data Ingest feature*, found at:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-1304ingestcmd/>

## Load considerations for column organized tables

Loading data into column-organized tables is similar to loading data into row-organized tables, but you also must consider the following items:

- ▶ The input data source is processed twice; the input source can be reopened or temporarily cached in the load temporary file directory. (specified by the **TEMPFILES PATH** option with the **LOAD** command)
- ▶ Additional storage space (which is equivalent to the raw size of the input data) on the server is required for building the column compression dictionary if the data source (such as a pipe or a socket) cannot be reopened.
- ▶ Memory requirements temporarily increase when the column compression dictionary is created.
- ▶ For optimal load performance, additional cache memory is required to write column-organized data in extent-sized amounts, rather than one page at a time, which reduces I/O costs.
- ▶ By default, when you create a database and **DB2\_WORKLOAD=ANALYTICS**, the **util\_heap\_sz** (utility heap size) database configuration parameter is set to automatic. If the automatic setting is not used, set **util\_heap\_sz** to at least 1,000,000 pages to address the resource needs of the **LOAD** command. If the database server has at least 128 GB of memory, set **util\_heap\_sz** to 4,000,000 pages or more. If concurrent load utilities are running, increase the **util\_heap\_sz** value to accommodate higher memory requirements. If memory is scarce, the **util\_heap\_sz** value can be increased dynamically only when a load operation is running.
- ▶ The **blocknonlogged** database configuration parameter must be set to **N0** before loading data into a column-organized table.
- ▶ Post load, import, and ingest operations, such as **SET INTEGRITY** and **REORG**, are not required for column organized tables.

## 6.2 DB2 Warehouse SQL Warehousing Tool

The basic function of SQW is to manage and move data into and around the data warehouse when transforming it for various purposes. SQW provides these services by using the power of the DB2 relational database engine and the SQL language, which classifies SQW in the ETL category of data movement and transformation tools. SQW also provides sequencing and flow control functions and functions to integrate non-database processing.

## 6.2.1 Architecture

SQW architecture separates the tools that are needed for the development environment and the runtime environment, as shown in Figure 6-2.

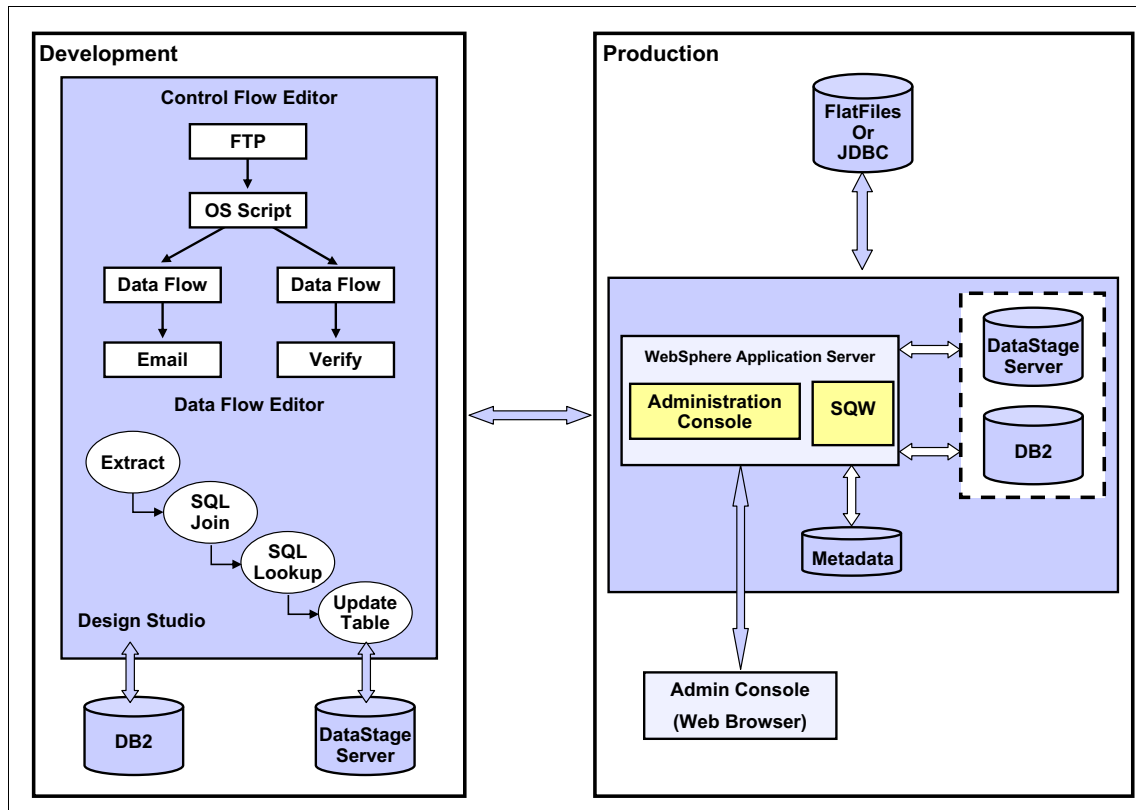


Figure 6-2 SQW architecture

The development tool is the DB2 Warehouse Design Studio, where the data movement and transformation routines are developed and tested.

After testing in the development environment against development databases, the routines are deployed to the runtime environment, where they can be scheduled for execution in the production databases. The web-based administration console is used to manage the runtime environment, including deployment, scheduling, and execution, and monitoring functions.

## 6.2.2 Development environment

The development tool is an Eclipse-based member of the IBM Data Studio family of products that are called the Design Studio, which is a graphical editor.

All the necessary steps to develop the logic to extract, transform, and load target warehouses based on the available source data happens in the development environment.

- ▶ The starting point is a *physical model* of the databases (sources and target), which is essentially an extraction of the schema, tables, and other details that describe the physical implementation of the databases. The physical data model provides metadata about the structure of the source and target tables.
- ▶ The *flow of data* is then designed. A *data flow* is a representation of what happens to the data as it flows into the execution database from source to target. The SQW plug-in in the Design Studio offers many operators and routines to help define source data, transformations, and target tables in a data flow.

A data flow is essentially the set of operations that performs the following tasks:

- Bring data into the data warehouse.
- Move and transform the data in the data warehouse.
- Update target tables.

From this model, Design Studio generates optimized SQL to perform the actions that are specified in the flow model. The SQL is run at the DB2 database that is selected as the execution database. The execution database does the transformation work when the SQL code in a data warehousing application runs.

- ▶ When you determine where data is coming from, what must be extracted, how the data is transformed, and where the data is updated, the sequencing of the activities and how control is passed from one to another is determined based on defined conditions. This activity is the *Control Flow* development phase.

A control flow is a set of operations that accomplish the following tasks:

- Start activities of various technologies.
- Sequence activities into a defined flow.
- Define how control is passed between activities based on defined conditions.

Activities represent work that is accomplished at a server or a flow control decision point. Activities are arranged in sequences that define in what order and under what conditions the activities are run. From this model, Design Studio generates code that manages the execution of activities in the correct sequence, dynamically based on the defined conditions.

- ▶ The next step is to see whether the data and control flows do what they are designed to do. The *testing and debugging* step ensures that the flow of data and sequencing of the activities happen successfully before implementing the process in a production environment. This is an iterative process to ensure that the design is successful.
- ▶ During the process of *validation* and when the flows are ready for deployment into production, code generation happens. The generated code might be optimized DB2 SQL embedded into an internal script-like control language that is called an execution plan graph (EPG), as in the case of data flows. In the case of control flows, it might also contain code units for other tasks, such as sequencing, conditions, and variables, in addition to the optimized SQL.

### 6.2.3 Moving from development to production

After the data flows and control flows are developed and tested in the development environment of Design Studio, they must be installed into the runtime environment for production use.

This *deployment process* consists of two phases:

- ▶ Preparation of the deployment “package”:  
This phase is done in Design Studio and consists of defining a warehouse application that is a collection of data and control flows. Code for the included flows is generated and packaged into a compressed deployment file.
- ▶ Deployment of the package into the runtime environment  
The compressed deployment package file can then be used by the administrator of the runtime environment. The administrator uses the web-based administration console to define all the database and system resources that are required for implementing a specific warehouse application package, install the warehouse application (which is essentially the intelligence that is contained in the data and control flows through the generated code), and configure any needed schedules.

# 6.2.4 Runtime environment

A collection of related control flows that are organized into a data warehouse application is deployed into the runtime environment that is managed by the web-based administration console. The warehouse application can be run on demand or through a schedule, and monitored. Various levels of statistics are also maintained to enable better management of the environment. The SQW runtime server component manages the running of each activity, in the correct sequence and on the appropriate database or system.

In the runtime environment, the unit of execution is the control flow. Control flows can be managed from the control flows page of the SQL Warehousing tab, as shown in Figure 6-3. From this page, you can perform the following tasks:

- ▶ Manually run a control flow.
- ▶ Create and manage execution schedules.
- ▶ Manage execution profiles.
- ▶ View execution statistics.
- ▶ View the application log.

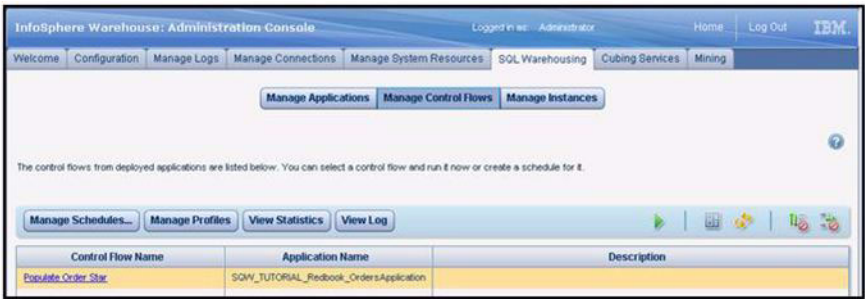


Figure 6-3 SQL Warehouse tab

The SQW component in InfoSphere Warehouse Version 10.1 Fix Pack 2 (10.1.0.2) includes support for the continuous ingest facility (CDI) in DB2. The ingest utility (sometimes referred to as continuous data ingest (CDI)) is a high-speed client-side DB2 utility that streams data from files into DB2 target tables. Because the ingest utility can move large amounts of real-time data without locking the target table, you do not need to choose between the data currency and availability. You can use the ingest utility to continuously pump data into DB2 tables using SQL array inserts, updates, and deletes until sources are exhausted.



Support was added for the Liberty profile for WebSphere Application Server Network Deployment component in InfoSphere Warehouse Version 10.1 Fix Pack 2 for the runtime environment (Administration Console). The Liberty profile is embedded in InfoSphere Warehouse. Users can choose to either use a stand-alone instance of WebSphere Application Server or the embedded version in the InfoSphere Warehouse at the time of installation.

As of SQW for DB2 Warehouse 10.5, SQW has support for BLU Acceleration (load, import, and ingest into column organized tables) and DB2 Oracle Compatible mode.

## **Integration between DataStage and DB2 Warehouse**

DB2 Warehouse provides support for IBM DataStage jobs through the SQW plug-in in Design Studio. Integration of the Design Studio and the DataStage systems allows you to export and import data flows and jobs from one execution environment to the other. XML metadata files serve as the foundation for the movement of jobs and data flows between the two systems.

Not all DataStage stages or even all properties of DataStage stages are supported in the Design Studio data flow environment.

The DataStage parallel job is first exported into an XML metadata format using the Export utility in the DataStage Manager. Then, from within the Design Studio explorer, the Import Parallel Job as Subflow utility is used to create a subflow, which corresponds to the original DataStage job. This feature allows jobs with Design Studio unsupported operators, developed in the Data Stage designer to be used from the Design Studio design and runtime environments.

DataStage also supports column organized tables in its extract and load execution environment.

For more information about creating data flows and how to import a DataStage job into Design Studio, see the information center found at:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp?topic=%2Fcom.ibm.dwe.welcome.doc%2Fdwe91design.html>





# Monitoring

As described in Chapter 9, “Workload management” on page 145, DB2 workload management has four clearly defined stages, the last of which is monitoring to ensure that your data server is being used efficiently. *Database monitoring* includes all of the processes and tools that you use to examine the operational status of your database, which is a key activity to ensure the continued health and excellent performance of your database management system. The DB2 monitoring infrastructure collects information from the database manager, active databases, and connected applications to help you analyze the performance of specific applications, SQL queries, and indexes. You can then use that data to troubleshoot poor system performance or to gauge the efficacy of your tuning efforts.

You can monitor your database in one of two ways:

- **Monitoring with table functions:**

You can monitor your database operations in real time by using monitoring *table functions*. These table functions return data from monitor elements that report on most database operations at a specific point. The monitoring table functions use a high-speed monitoring infrastructure that was introduced in DB2 for Linux, UNIX, and Windows 9.7. Before Version 9.7, the DB2 monitoring infrastructure included snapshot monitoring routines. Although these routines are still available, this technology is no longer being enhanced in the DB2 product, and you are encouraged to use the monitoring table functions wherever possible.

- **Monitoring with event monitors:**

You can configure *event monitors* to capture information about specific database events (such as deadlocks) over time. Event monitors generate output in different formats, but all of them can write event data to regular tables.

This chapter provides an overview of the DB2 monitoring interfaces. It also introduces you to IBM InfoSphere Optim Performance Manager Version 5.3, which provides a convenient web interface to help you identify and analyze database performance problems.

## 7.1 Understanding monitor elements

*Monitor elements* are data structures that are used to store information about the operational status of your database. The DB2 monitor elements can be classified according to the type of data that they track:

- *Counter elements* track the number of times that some specific event occurs. For example, the deadlocks monitor element records the total number of deadlocks that occurred.
- *Gauge elements* track how much of something is happening or used. For example, the total\_sort\_time monitor element tracks the total elapsed time for all sort operations.
- *Watermark elements* track the highest value reached. For example, the sort\_heap\_top monitor element tracks the private sort memory high watermark (in 4 KB pages) across the DB2 instance.

- ▶ *Text elements* track text values. For example, the stmt\_text monitor element contains the text of an SQL statement.
- ▶ *Timestamp elements* track the time at which an event occurred. For example, the conn\_time monitor element tracks the time at which a connection was made to the database.

Monitor elements can be further classified by the type of work that they monitor:

- ▶ *Request monitor elements* measure the amount of work that is needed to process different types of application requests. A *request* is a directive to a database agent to perform some work that consumes database resources. For example, an application request is a directive that is issued directly by an external application.
- ▶ *Activity monitor elements* are a subset of request monitor elements. Activity monitor elements measure the amount of work that is needed to run SQL statement sections, including locking, sorting, and the processing of row-organized or column-organized data.
- ▶ *Data object monitor elements* return information about operations that are performed on specific data objects, including buffer pools, containers, indexes, tables, and table spaces.
  - *Time-spent monitor elements* track how time is spent in the system.
  - *Wait-time monitor elements* track the amount of time during which the database manager waits before it continues processing. For example, the database manager might spend time waiting for locks on objects to be released; this time is tracked by the lock\_wait\_time monitor element.
  - *Component processing time monitor elements* track the amount of time that is spent processing data within a specific logical component of the database. For example, the total\_commit\_proc\_time monitor element tracks the amount of time that is spent committing transactions.
  - *Component elapsed time monitor elements* track the total amount of elapsed time that is spent within a specific logical component of the database. This time includes both processing time and wait time. For example, the total\_commit\_time monitor element tracks the total amount of time that is spent performing commit processing on the database server.

For more information about the DB2 monitor elements, see the DB2 for Linux, UNIX, and Windows 10.5 information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp>

## 7.1.1 Monitor element collection levels

The *collection level* for a monitor element identifies the settings that must be active before data can be collected for that element.

You can use database configuration parameters to set the default collection level for a specific class of monitor elements for the entire database. For example, if you set the **mon\_req\_metrics** database configuration parameter to **BASE**, application request data is collected for all agents running in your database. You can also specify collection levels for specific DB2 workload management objects that are different from the level that is used for the database as a whole.

Data for some monitor elements is always collected. There are no configuration parameters or SQL statement options to control the collection of this data.

- ▶ Monitor elements with collection level **DATA OBJECT METRICS BASE** or **DATA OBJECT METRICS EXTENDED** are collected if you set the **mon\_obj\_metrics** database configuration parameter to **BASE** or **EXTENDED**. A value of **EXTENDED** causes certain monitor elements to collect data at a higher level of granularity. If you set this parameter to **NONE**, data is not collected.
- ▶ Monitor elements with collection level **REQUEST METRICS BASE** or **REQUEST METRICS EXTENDED** are collected if you set the effective collection level to **BASE** or **EXTENDED**. The *effective collection level* for request metrics is determined by the **mon\_req\_metrics** database configuration parameter and the **COLLECT REQUEST METRICS** option on a service superclass. Monitor elements are collected for a request if you set the configuration parameter to a value other than **NONE**, or if the request is submitted by a connection that mapped to a subclass under a superclass that has a **COLLECT REQUEST METRICS** setting other than **NONE**.
- ▶ Monitor elements with collection level **ACTIVITY METRICS BASE** or **ACTIVITY METRICS EXTENDED** are collected if you set the effective collection level to **BASE** or **EXTENDED**. The effective collection level for activity metrics is determined by the **mon\_act\_metrics** database configuration parameter and the **COLLECT ACTIVITY METRICS** option on a workload. Monitor elements are collected for an activity if you set the configuration parameter to a value other than **NONE**, or if the activity is submitted by a connection that is associated with a workload that has a **COLLECT ACTIVITY METRICS** setting other than **NONE**.

- Monitor elements with collection level **SECTION ACTUALS BASE** are collected if you set the effective collection level to **BASE**. The effective collection level for section actuals is determined by the **section\_actuals** database configuration parameter, specification of the **INCLUDE ACTUALS** clause on the **CREATE or ALTER WORKLOAD**, **CREATE or ALTER WORK ACTION SET**, or **CREATE or ALTER SERVICE CLASS** statements, and the **collectsectionactuals** setting on the **WLM\_SET\_CONN\_ENV** procedure.
- Monitor elements with collection level **COLLECT AGGREGATE ACTIVITY DATA** or **COLLECT AGGREGATE REQUEST DATA** are collected if you specify the **COLLECT AGGREGATE ACTIVITY DATA** or **COLLECT AGGREGATE REQUEST DATA** clause on the **CREATE or ALTER** statement for specific types of DB2 workload management objects.

**Important:** The effective collection level for a monitor element is determined by applying the highest collection level that was specified for that element for the scope in which it is collected. For example, if you specify a collection level of **NONE** for request monitor elements at the database level (using the **SAMPLE** database), but want to collect **EXTENDED** metrics for agents that are running in the default service superclass, you can achieve this by running the following command and SQL statement:

```
UPDATE DB CFG FOR SAMPLe USING MON_REQ_METRICS NONE;
ALTER SERVICE CLASS SYSDEFAULTUSERCLASS COLLECT REQUEST METRICS EXTENDED;
```

In this example, request metrics are collected for all agents that run in the **SYSDEFAULTUSERCLASS**, but not for agents that run outside of the **SYSDEFAULTUSERCLASS**.

Now suppose that you specify a collection level of **EXTENDED** for activity monitor elements at the database level, but you do not want to collect activity metrics for the default user workload. You can achieve this by running the following command and SQL statement:

```
UPDATE DB CFG FOR SAMPLe USING MON_ACT_METRICS EXTENDED;
ALTER WORKLOAD SYSDEFAULTUSERWORKLOAD COLLECT ACTIVITY METRICS NONE;
```

In this example, activity metrics are collected for all agents that run in the database, including those that run as part of the **SYSDEFAULTUSERWORKLOAD**. The effective collection level is determined by the broader collection level (**EXTENDED**) that was specified at the database level by setting the **mon\_act\_metrics** configuration parameter.

## 7.1.2 New monitoring elements for column-organized tables

DB2 10.5 introduces column-organized data processing to DB2 databases. The release also introduces new monitor elements to help you monitor workloads that involve queries against column-organized tables.

Here is a list of the new monitor elements:

- New monitor elements to assess buffer pool efficiency

You can monitor data page I/O for column-organized tables separately from that of row-organized tables, and use monitor elements to understand what portion of the I/O is being driven by access to column-organized tables when a workload impacts both row-organized and column-organized tables. You can also use these elements to help you to decide whether to place column-organized tables in separate table spaces, or whether to use a separate buffer pool. Here is a list of these elements:

- Counters for total logical and physical column-organized data page reads and pages found
- A counter for column-organized data page writes
- Counters for asynchronous column-organized data page reads and writes and pages found
- Counters for column-organized data page reads per table and per statement per table

- New monitor elements to monitor prefetch requests for data in column-organized tables

New monitor elements to measure prefetcher efficiency can help you track the volume of requests for data in column-organized tables that are being submitted to prefetchers, and the number of pages that prefetchers skipped reading because the pages are already in memory. Efficient prefetching of data in column-organized tables is important for mitigating the I/O costs of data scans.

- New monitor elements to measure column data size

Column-organized tables are associated with a new table object where the column data is stored. New monitor elements (`col_object_l_size`, `col_object_p_size`, and `col_object_l_pages`) help you estimate the size of the column data.

The `tab_organization` monitor element, which is returned by the `MON_GET_TABLE` table function, reports the organization of data in the table.



- New time-spent monitor elements

Three new monitor elements (`total_col_time`, `total_col_proc_time`, and `total_col_executions`) count the total time that is spent in column-organized data processing across all column-organized processing subagents. The `total_col_time` monitor element represents total elapsed time over all column-organized processing subagents. The `total_col_proc_time` monitor element represents the subset of `total_col_time` in which the column-organized processing subagents were not idle on a measured wait time. The `total_col_executions` monitor element represents the total number of times that data in column-organized tables was accessed during statement execution.

For more information about the new monitoring metrics for column-organized data processing in DB2 10.5, see the information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp>

## 7.2 Using DB2 table functions to monitor your database in real time

DB2 table functions return data from monitor elements that report on the following items at a specific point:

- Application requests
- Activities
- Operations on data objects
- Locks
- System memory use
- Routines

For more information about the DB2 table functions that return data from monitor elements, see the DB2 10.5 information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp>

### 7.2.1 Monitoring requests

You can use request monitor elements when you monitor data server operations that are associated with processing application requests. Request monitor elements are aggregated by unit of work, workload, service class, and connection.

Use the following table functions to access current request monitoring information:

- ▶ MON\_GET\_AGENT
- ▶ MON\_GET\_CONNECTION and MON\_GET\_CONNECTION\_DETAILS
- ▶ MON\_GET\_SERVICE\_SUBCLASS and MON\_GET\_SERVICE\_SUBCLASS\_DETAILS
- ▶ MON\_GET\_UNIT\_OF\_WORK and MON\_GET\_UNIT\_OF\_WORK\_DETAILS
- ▶ MON\_GET\_UTILITY
- ▶ MON\_GET\_WORKLOAD and MON\_GET\_WORKLOAD\_DETAILS

For a database that is created in DB2 10.5, these table functions collect request monitoring information by default.

## 7.2.2 Monitoring activities

You can use activity monitor elements when you monitor the amount of work that is needed to run SQL statement sections. Activity monitor elements are aggregated in the package cache by all executions of each SQL statement section.

Use the following table functions to access current activity monitoring information:

- ▶ MON\_GET\_ACTIVITY\_DETAILS
- ▶ MON\_GET\_PKG\_CACHE\_STMT and MON\_GET\_PKG\_CACHE\_STMT\_DETAILS

For a database that is created in DB2 10.5, these table functions collect activity monitoring information by default.

## 7.2.3 Monitoring data objects

You can use data object monitor elements when you monitor operations that are performed on specific data objects, including buffer pools, containers, indexes, tables, and table spaces.

Use the following table functions to access current data object monitoring information:

- ▶ MON\_GET\_BUFFERPOOL
- ▶ MON\_GET\_CONTAINER
- ▶ MON\_GET\_INDEX

- ▶ MON\_GET\_TABLE
- ▶ MON\_GET\_TABLESPACE

For a database that is created in DB2 10.5, these table functions collect data object monitoring information by default.

## 7.2.4 Monitoring locks

You can use the following table functions to access current lock monitoring information:

- ▶ MON\_GET\_LOCKS
- ▶ MON\_GET\_APPL\_LOCKWAIT

These table functions collect lock monitoring information by default.

## 7.2.5 Monitoring system memory

You can use the following table functions to access system memory information:

- ▶ MON\_GET\_MEMORY\_POOL
- ▶ MON\_GET\_MEMORY\_SET

You can examine memory usage by specific memory pools within a memory set, or at the level of memory sets, which are allocations of memory from the operating system.

## 7.2.6 Monitoring routines

You can use the following table functions to access current information about routines:

- ▶ MON\_GET\_ROUTINE and MON\_GET\_ROUTINE\_DETAILS
- ▶ MON\_GET\_ROUTINE\_EXEC\_LIST
- ▶ MON\_GET\_SECTION\_ROUTINE

## 7.3 Using event monitors to capture information about database events

You can create event monitors to capture information about specific events that take place in your system at the time at which they occur. Monitored events include lock assignments, deadlocks, operations (such as the execution of SQL statements) that create database activity, completion of transactions (units of work), eviction of sections from the package cache, application connections to a database, database deactivations, statistics collection, violations of workload management thresholds, and database or database manager configuration changes.

The process of creating and using an event monitor is similar for all event monitor types. The first step is to create the event monitor by using the **CREATE EVENT MONITOR** statement and specifying the type of event monitor that you want to create. The second step is to activate the event monitor (by using the **SET EVENT MONITOR STATE** statement), and the third step is to access and work with the captured data.

The following example code creates and activates event monitors for activities and statistics in the **SAMPLE** database:

```
CONNECT TO SAMPLE;
```

```
CREATE EVENT MONITOR DB2ACTIVITIES FOR ACTIVITIES WRITE TO TABLE;  
CREATE EVENT MONITOR DB2STATISTICS FOR STATISTICS WRITE TO TABLE;
```

```
SET EVENT MONITOR DB2ACTIVITIES STATE 1;  
SET EVENT MONITOR DB2STATISTICS STATE 1;
```

Event monitors can write their output to relational tables, named pipes, or files. By default, some event monitors activate automatically upon database activation; others require that you activate them manually. To disable an event monitor, run the **SET EVENT MONITOR STATE** statement and specify the **STATE 0** option.

For certain event monitors (activities, locking, package cache, statistics, and unit of work event monitors), you must explicitly enable data collection by setting the appropriate database configuration parameters or to enable the collection of specific kinds of data for specific types of workload management objects.

For example, to configure the collection of basic data for a unit of work event monitor when any transaction finishes, you can set the **mon\_uow\_data** database configuration parameter to **BASE**. To collect unit of work data for a specific workload only, run the **CREATE WORKLOAD** or **ALTER WORKLOAD** statement, specifying the **COLLECT UNIT OF WORK DATA BASE** clause.

**Tip:** After you run the application for which you want to collect data, it is a good idea to deactivate the event monitor to avoid the collection of unneeded data.

If the data is written to a relational table, you can use SQL to access the data, but if the data is written to an unformatted event table, you must run the **db2evmonfmt** command or call the **EVMON\_FORMAT\_UE\_TO\_TABLES** procedure before you can examine the event data.

Monitor elements that complement one another are grouped in useful sets called *logical data groups*. For example, the event\_activity logical data group includes monitor elements like appl\_id (application ID) and uow\_id (unit of work ID), which are often returned together, as, for example, in the following query:

```
SELECT VARCHAR(A.APPL_NAME, 15) AS APPL_NAME,  
       VARCHAR(A.TPMON_CLIENT_APP, 20) AS CLIENT_APP_NAME,  
       VARCHAR(A.APPL_ID, 30) AS APPL_ID,  
       A.ACTIVITY_ID,  
       A.UOW_ID,  
       VARCHAR(S.STMT_TEXT, 300) AS STMT_TEXT  
FROM ACTIVITY_DB2ACTIVITIES AS A,  
     ACTIVITYSTMT_DB2ACTIVITIES AS S  
WHERE A.APPL_ID = S.APPL_ID AND  
      A.ACTIVITY_ID = S.ACTIVITY_ID AND  
      A.UOW_ID = S.UOW_ID;
```

**Important:** DB2ACTIVITIES is the name of the event monitor that you created earlier in this section. The standard table reference ACTIVITY\_DB2ACTIVITIES or ACTIVITYSTMT\_DB2ACTIVITIES consists of the name of the logical data group concatenated with the underscore (\_) character concatenated with the name of the event monitor.

The DB2 data server associates a default set of logical data groups with each event monitor, and the monitor therefore collects a useful set of elements for you automatically.

For a list of the logical data groups and the monitor elements that they can return during event monitoring, see the DB2 10.5 information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp?topic=%2Fcom.ibm.db2.luw.admin.mon.doc%2Fdoc%2Fr0007595.html>

**Tip:** Prune data that you no longer need from event monitor tables for frequently used event monitors. If you must prune event monitor output regularly, consider using an unformatted event table to record event monitor output because unformatted event tables can be pruned automatically after data is transferred to relational tables.

## 7.4 Monitoring your DB2 system performance with IBM InfoSphere Optim Performance Manager

InfoSphere Optim Performance Manager (OPM) is a performance analysis and tuning tool for managing DB2 systems through a web interface. OPM helps you identify and resolve database problems before they impact your business. OPM collects performance metrics and determines whether they exceed defined thresholds.

OPM features and functions can be grouped in to the following categories:

- ▶ Identifying and diagnosing performance problems  
From a single control point, OPM can monitor and analyze multiple DB2 instances that are running various workloads. Predefined and customizable monitoring templates enable you to quickly deploy performance monitoring in online transaction processing and business intelligence environments.
- ▶ Preventing performance problems  
Workload management tools let you easily define workloads, assign business priorities, enable concurrency controls, track resource consumption, system capacity, and response times, identify performance issues, and capture historical data for trend analysis and growth planning.
- ▶ Solving performance problems  
OPM is fully integrated with the IBM Optim family of products to help you quickly resolve problems:
  - Optim Query Workload Tuner analyzes and tunes problematic SQL statements.
  - Optim pureQuery Runtime facilitates collaboration between DBAs and developers to resolve problems and optimize queries without altering your application.
  - Tivoli Composite Application Manager provides a consolidated view of the transactions across your enterprise and enables you to start OPM in context.

## 7.4.1 Information dashboards

When a problem is identified, diagnostic dashboards in the OPM web console focus on the impacted area and display information that helps you to analyze the problem:

- ▶ The *Health Summary* dashboard displays an alerts overview on the Alerts page so that you can identify which of your databases have critical issues that require attention. You can define alert notifications by type, severity, and database.
- ▶ The dashboards that are listed under *Inflight Dashboards* in the main menu provide real-time and historical information about specific databases. The *Overview* dashboard shows the high-level status of your database so that you can identify potential problems. In addition to the Overview dashboard, there are other dashboards that relate to specific performance categories and that you can use to investigate potential problems:
  - *Buffer Pool and I/O*: This dashboard shows database I/O at the buffer pool, table space, and table level. You can find the buffer pools with low hit ratios and high activity and consider increasing their size to improve performance.
  - *Connection*: This dashboard shows active database connections during a specific time frame. The dashboard displays key performance indicators, such as lock wait times, to help you identify applications that are not performing well.
  - *Locking*: This dashboard helps you to identify and analyze deadlocks, timeouts, lock waits, locking conflicts, locked objects, and the SQL statements that are holding and waiting for locks.
  - *Logging*: This dashboard shows configuration and logging activity, and helps you determine whether tuning is needed for the log files or log buffer.
  - *Memory*: This dashboard shows memory consumption by instance, database, and application, and shows the memory that is shared between applications.
  - *SQL Statements*: This dashboard provides performance data for queries.
  - *System*: This dashboard shows the resources for the system on which the database is running.
  - *Utilities*: This dashboard shows the status and progress of utilities. You can use this information to determine whether some utilities should run at different times to avoid high workloads.
  - *Workload*: This dashboard shows information about the workloads on the database, including connected applications and transactions.

- ▶ The Extended Insight dashboard (available with IBM InfoSphere Optim Performance Manager Extended Insight (OPM EI)) displays data about the entire database application system, including clients, application servers, data servers, and the network.

## 7.4.2 OPM support for DB2 10.5

Optim Performance Manager Version 5.3 provides support for DB2 10.5. New performance metrics for column-organized tables are available on the following dashboards:

- ▶ Overview
- ▶ Buffer Pool and I/O
- ▶ Connection
- ▶ SQL Statements
- ▶ Extended Insight

You can use these performance metrics, which include table and column access statistics, I/O statistics, and extended insight statistics, to verify that your workloads with column-organized tables behave as expected.

IBM InfoSphere Optim Query Workload Tuner (OQWT) Version 4.1 can help you tune individual SQL statements or a complete SQL workload. OQWT includes the Workload Table Organization Advisor, which examines the tables that are referenced by the statements in a query workload. The advisor makes recommendations about which tables are good candidates for conversion from row to column organization. The advisor also estimates the performance gain that can be realized if the recommended tables are converted from row to column organization. For more information about the Workload Table Organization Advisor, see the following web page:

<http://pic.dhe.ibm.com/infocenter/dstudio/v4r1/index.jsp?topic=%2Fcom.ibm.datatools.qrytune.workloadtunedb2luw.doc%2Ftopics%2Fgenrecswtoa.html>

For a summary of the new features and enhancements in Version 5.x of IBM InfoSphere Optim Performance Manager for DB2 for Linux, UNIX, and Windows and its modifications and fix packs, see the following web page:

<http://www-01.ibm.com/support/docview.wss?uid=swg27023197>

For more information about IBM InfoSphere Optim Performance Manager, see the Version 5.3 information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/perfmgmt/v5r2/index.jsp>





# High availability

IBM PureData System for Operational Analytics provides redundant hardware components and availability automation to minimize and eliminate impacts from hardware and software failures.

This chapter describes the high availability characteristics present in the IBM PureData System for Operational Analytics.

## 8.1 IBM PureData System for Operational Analytics

IBM PureData System for Operational Analytics provides a highly available database computing environment. The high availability (HA) design consists of both redundant components and automation software for recovery. These capabilities provide many transparent failures for the functioning of the system and help minimize the downtime that is caused by single failure events.

In IBM PureData System for Operational Analytics, the following components are designed for redundancy:

- ▶ RAID arrays for external storage
- ▶ Mirrored volume groups for internal storage
- ▶ Quad-port Fibre Channel adapters
- ▶ Redundant storage area network (SAN) switches
- ▶ Dual active RAID controllers
- ▶ Dual hot-swappable power/cooling units
- ▶ Redundant network switches
- ▶ Highly available network interfaces

In addition to the numerous redundancies that are part of its design, IBM PureData System for Operational Analytics also uses IBM Tivoli System Automation for Multiplatforms (SA MP) cluster management software to provide high availability capabilities at the software and hardware level. Tivoli SA MP integration provides the capability for automated recovery to take specific actions when a detectable resource failure occurs. This action can be as simple as restarting a failed component, for example, software, in-place, or automatically failover components from the active host to the standby host. A resource failure can include:

- ▶ Network failure
- ▶ Host failure
- ▶ DB2 instance or partition failure

The IBM PureData System for Operational Analytics HA design protects the Management Host and the *core warehouse*, which consists of the administration host on the foundation node, and data hosts on the data nodes. The HA configuration consists of two peer domains: one for the management host and one for the core warehouse. This configuration allows the HA resources on the management host to be managed independently of the HA resources for the core warehouse.

The HA solution is implemented in multiple HA groups. Each HA group consists of up to three active nodes and an identical standby node. Having an identical standby node as the failover target ensures that there is no performance degradation after failover occurs. The standby node is available to fail over processing from any one of the active nodes in the same HA group.

Figure 8-1 shows an overview of a high availability configuration consisting of three active nodes and a standby node, also known as a 3+1 HA configuration. This means that for every three active nodes there is one standby node available to fail over processing from any one of the active nodes.

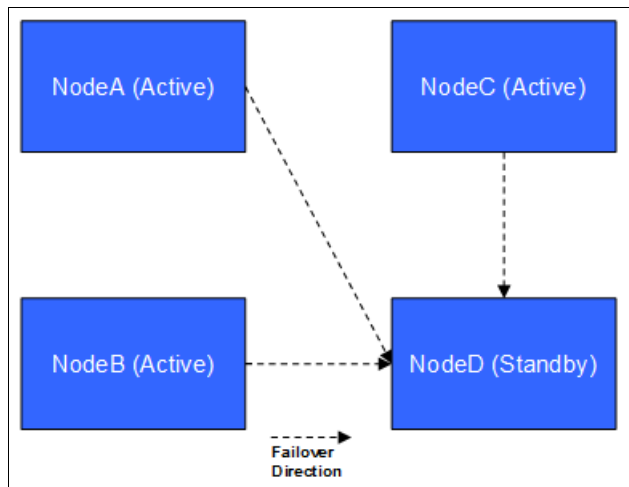
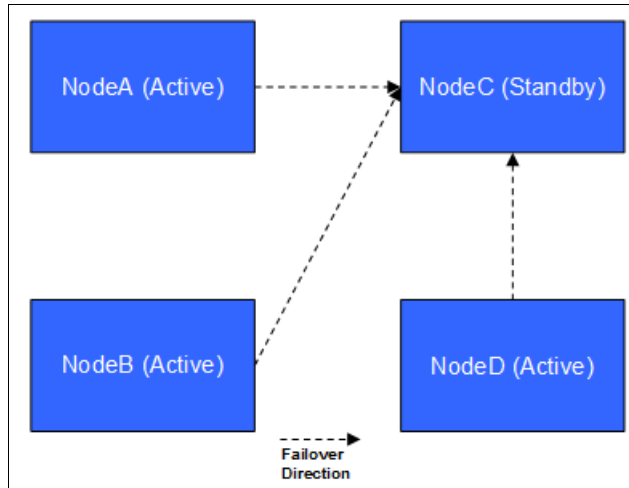


Figure 8-1 3+1 high availability configuration

The HA solution for IBM PureData System for Operational Analytics is implemented with a roving standby configuration for each HA group. This means that the standby role moves among the nodes within an HA group so that any node in an HA group can be an active or standby node as required. When a failure occurs, the standby node assumes the role of the active node. When the failed node is brought back online, it assumes the role of the standby node for the respective HA group, as shown in Figure 8-2.



*Figure 8-2 Roving standby post node failure*

Furthermore, if a subsequent active node fails, its processing is failed over to the standby node in the HA group, as shown in Figure 8-3 on page 123.

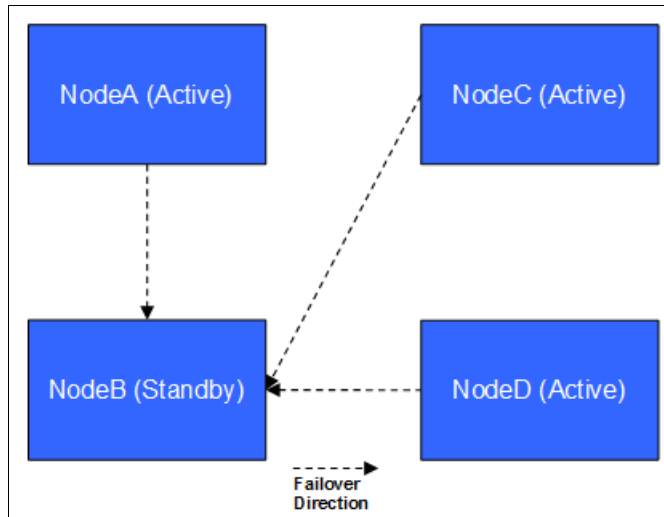


Figure 8-3 Roving standby post subsequent node failure

The roving standby HA configuration that is implemented in IBM PureData System for Operational Analytics increases availability and reduces the administration that is associated with failover activities.

## 8.2 Core warehouse availability

High availability for the core warehouse is implemented in a roving HA group configuration. In the roving HA group configuration, core warehouse hosts are placed in groups that contain up to three active hosts, the nodes where the hosts run, the storage nodes that are attached to those nodes, one standby host, and the standby node where the standby host runs.

The first roving HA group in the core warehouse contains the administration host on the foundation node and the standby administration host on the standby foundation node. This roving HA group contains only one active host and one standby host.

Subsequent roving HA groups in the core warehouse contain up to three active data hosts and one standby data host.

If a detectable failure occurs on an active host, its storage and processing capability fail over to the standby host that is contained in the same roving HA group. Because the standby host is powered on, installed with the full software stack, and connected to both SAN switches, it can access the storage and database partitions from the failed host and take over the processing for new workloads. All hosts in the system communicate through the highly available internal application network, and access the file systems on the external storage nodes through redundant Fibre Channel connections. The roving HA group configuration makes the database partitions on each host in the core warehouse highly available.

## 8.2.1 Roving HA group configuration for the administration hosts

High availability for the administration hosts is implemented as a roving HA group with one active host and one standby host. The administration host on the foundation node is paired with the standby administration host on the standby foundation node. Figure 8-4 shows the roving HA group that contains the active administration host and the standby administration host.

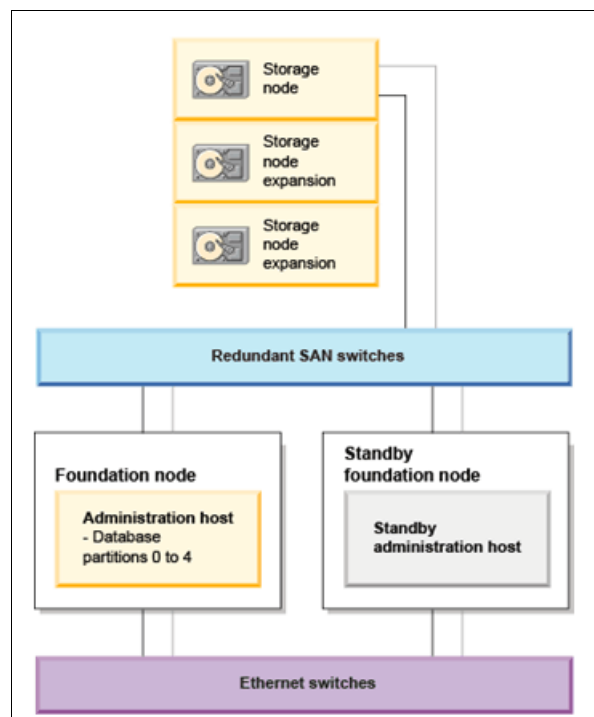


Figure 8-4 Roving HA group for the administration hosts

The HA resources that support the five database partitions run on the active administration host. If a detectable failure occurs, the resources fail over to the standby administration host and the standby administration host continues to process the workload. When the standby administration host takes over the workload processing, it assumes the role of the active administration host. When the failed administration host is brought back online, it assumes the role of the standby administration host.

Figure 8-5 shows the administration hosts after the resources are failed over and the failed administration host is brought back online as the standby administration host.

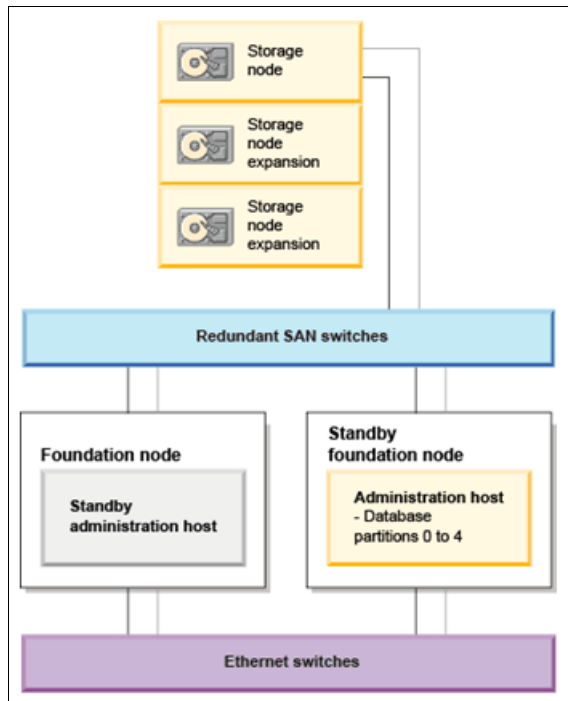


Figure 8-5 Roving HA group for the administration hosts after a failover

## 8.2.2 Roving HA group configuration for the data hosts

High availability for the data hosts is implemented as a roving HA group with up to three active hosts and one standby host. The data host on the data node is paired with the standby data host on the standby node.

Figure 8-6 shows the initial roving HA group configuration that contains three active data host and one standby data host.

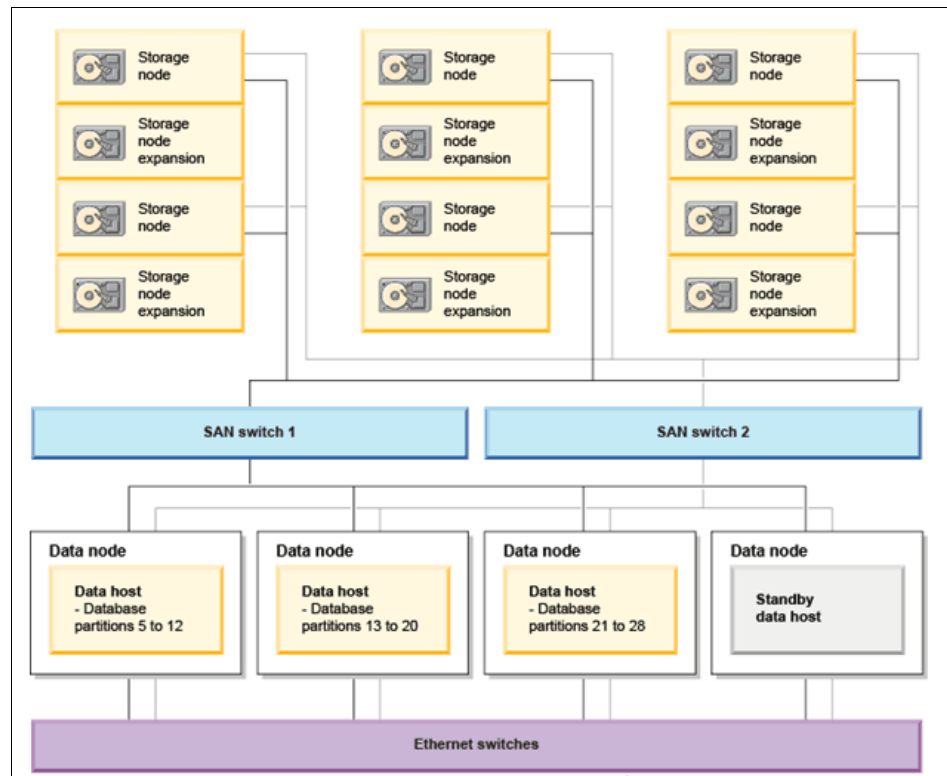


Figure 8-6 Roving HA group for the data hosts

If a detectable failure occurs on any one active data host, the resources of the failed data host fail over to the standby data host, and the standby data host assumes the role of the active data host in the roving HA group and starts processing new workloads. When the failed data host is brought back online, it integrates and assumes the role of the standby data host for the HA group. The resources do not need to be manually failed back to the original data host.

Figure 8-7 on page 127 shows the roving HA group after the resources on a data host have failed over and the failed data host assumes the role of the standby data host.



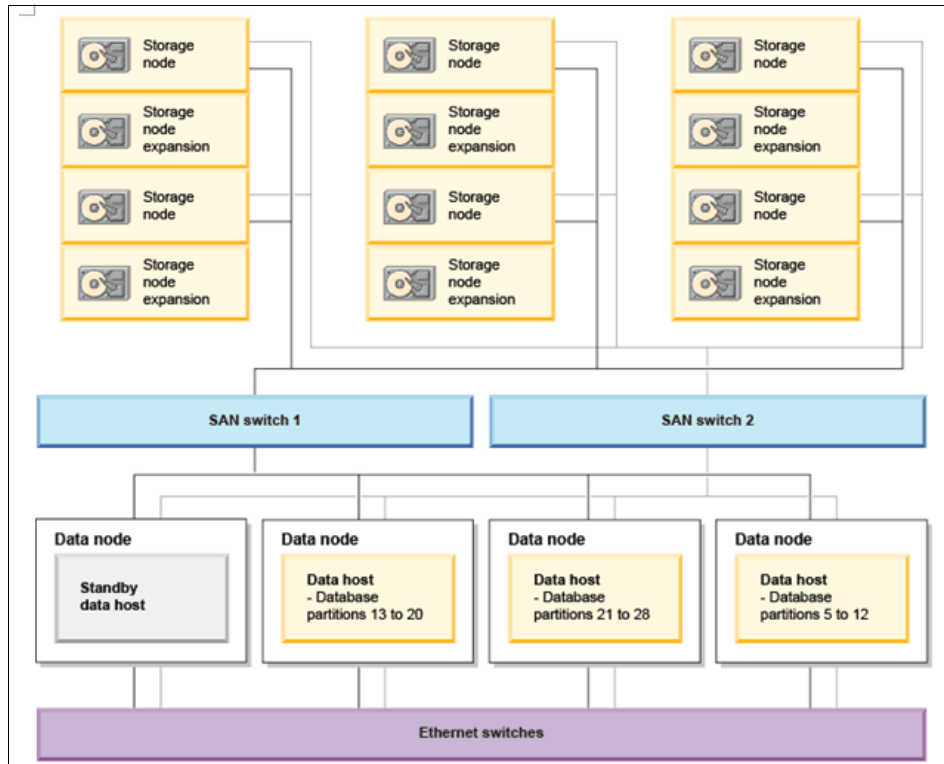


Figure 8-7 Roving HA group for the data hosts after a failover

### 8.2.3 Core warehouse HA events monitored by the system console

The system console displays alerts for the following core warehouse high availability events. The following core warehouse failover and failure scenarios are monitored by the system console:

- ▶ A user-initiated move of the core warehouse resources, called a manual failover or a resource move. Typically, this event indicates that the resources have moved for maintenance purposes when you use the **hafailover** command. The event type is displayed as a Resource Move with a severity of Informational.
- ▶ A failover that was successful. After a detectable failure, the core warehouse resources moved from the active core warehouse host to the standby host. The event type is displayed as a Resource Failover with a severity of Warning.
- ▶ A failure that results in a persisting outage. After a detectable failure, the resources did not move from the active core warehouse host to the standby

host successfully. The event type is displayed as a Resource Failover with a severity of Critical.

## 8.3 Management host availability

The HA solution for the management host is implemented as a HA pair that includes the management host and a standby management host, where the HA resources on the management host can fail over to the standby management host. The HA resources that are included in the configuration are the resources for the database performance monitor and the resources for the warehouse tools. These HA resources on the management hosts are contained in a peer domain (called mgmtdomain).

The HA pair configuration consists of the active management host on the foundation node paired with the passive standby management host on the standby foundation node, as shown in Figure 8-8.

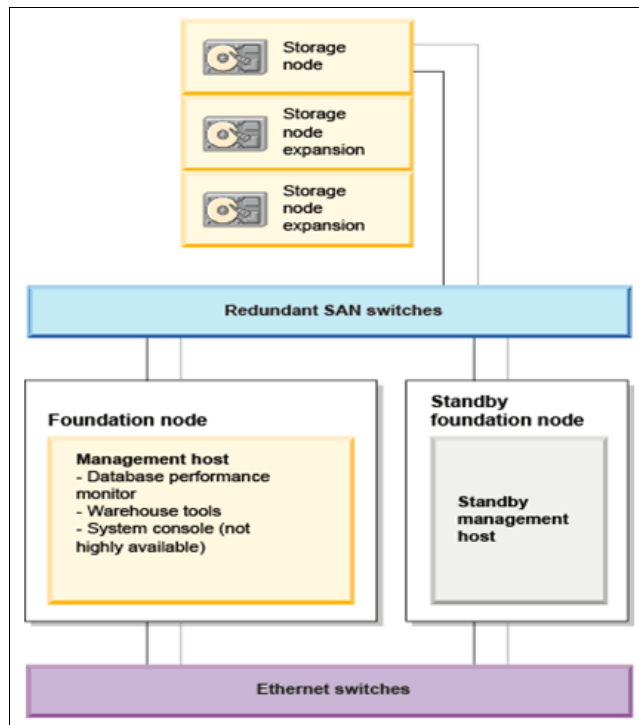


Figure 8-8 Management host HA configuration

The resource model for the management host includes separate resource groups for the database performance monitor and the warehouse tools. This separation of resource groups allows for independent availability management if there is a failure, such that only the affected resources are failed over as required.

If a detectable failure occurs on the management host and affects both the database performance monitor and the warehouse tools, the resources fail over to the standby management host, which allows the database performance monitor to continue to monitor the performance of the core warehouse database and allows the warehouse tools to process new jobs and requests. However, the system console event monitoring capability is not available on the standby management host.

Figure 8-9 shows the HA configuration of management host and the standby management host after a failure of the management host.

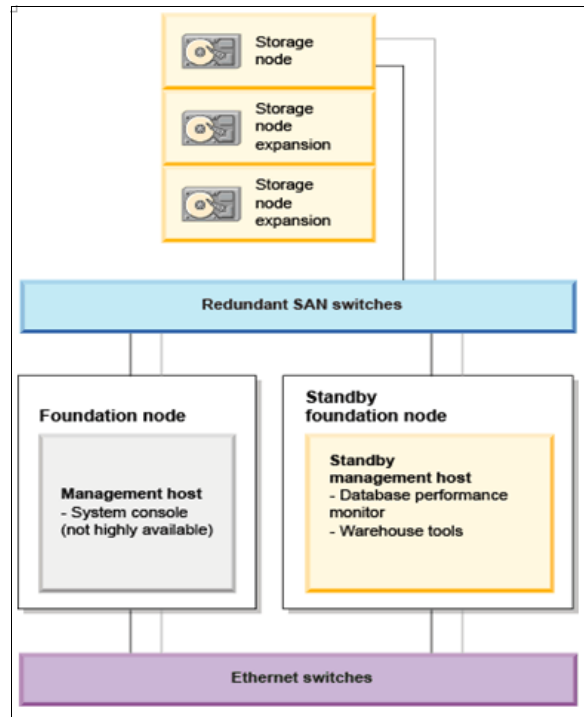


Figure 8-9 Management host HA configuration after management host failure

When the failed management host is repaired and brought back online, the resources must be manually moved back (that is, fall back) from the standby management host to the management host at the next available maintenance window or when a short planned service outage is acceptable. Until the resources are failed back, the system console event monitoring capability is not available and the management host is running unprotected against another failure event.

If a failure of a resource that affects only the warehouse tools occurs, the HA resources for the warehouse tools fail over to the standby management host. The HA resources for the performance monitor continue to run, unaffected, on the management host, as shown in Figure 8-10.

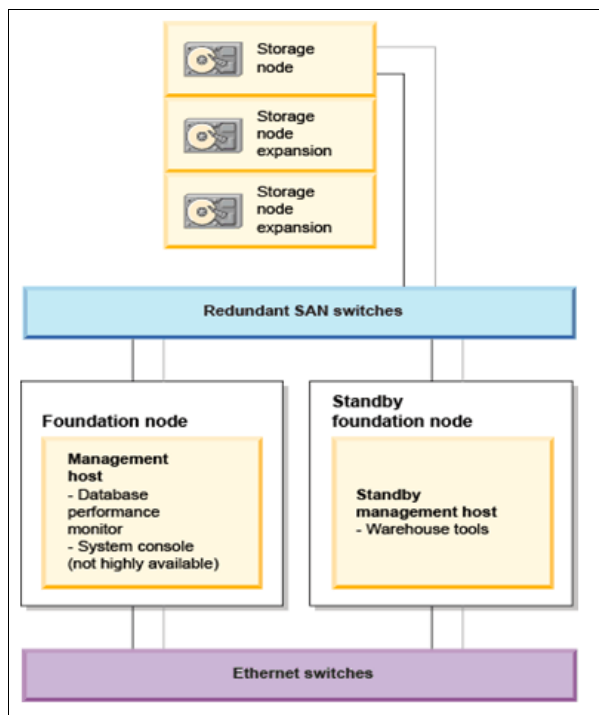


Figure 8-10 Management host HA configuration of warehouse tools resources

Similarly, if a resource fails that affects the database performance monitor resources only, then the database performance monitor resources are failed over to the standby management host and the warehouse tools continue to run, unaffected, on the management host.

After either such event, manually fall back the resources from the standby management host to the management host at the next scheduled maintenance window or when a short planned service outage is acceptable.

### 8.3.1 Management host failover events that are monitored by the system console

The system console sends SNMP notifications for the following failover events on the management host.

#### Database performance monitor failover events

The following database performance monitor failover and failure scenarios on the management host are monitored by the system console:

- ▶ A user-initiated move of the database performance monitor, called a manual failover or a resource move. Typically, this event indicates that the resources were moved for maintenance purposes by using the **hafailover** command. The event type is displayed as a Resource Move with a severity of Informational.
- ▶ A failover of the database performance monitor that was successful. After a detectable failure, the resources moved from the management host to the standby management host. The event type is displayed as a Resource Failover with a severity of Warning.
- ▶ A failure that results in a persistent outage of the database performance monitor. After a detectable failure, the resources did not move from the management host to the standby management host successfully. The event type is displayed as a Resource Failover with a severity of Critical.

#### Warehouse tools failover events

The following warehouse tools failover events on the management host are monitored by the system console:

- ▶ A failover of the warehouse tools that was successful. After a detectable failure, the resources moved from the management host to the standby management host. The system console displays the following critical system events during the failover: Either InfoSphere Warehouse high availability is offline or InfoSphere Warehouse high availability is partially online.

When the warehouse tools resources start on the standby management host, the system console displays an informational alert indicating that InfoSphere Warehouse high availability is online.

- ▶ A failure that results in a persistent outage of the warehouse tools. After a detectable failure, the resources did not move from the management host to the standby management host successfully. The system console displays the following critical system events:
  - InfoSphere Warehouse high availability is offline.
  - WebSphere Application Server is offline.
  - Metadata database is offline.
  - Cube servers are offline.

## 8.4 High availability management

This section describes high availability environment management.

### 8.4.1 High availability toolkit

The high availability (HA) toolkit helps you manage the high availability configuration for the core warehouse (administration host, standby administration host, data host, and standby data hosts). Some commands are also available for the high availability configuration on the management hosts.

The HA toolkit provides the commands, such as **hachkconfig**, for you to manage your high availability environment. For a list of commands, see *High availability toolkit* at the IBM PureData System for Operational Analytics information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/whsesys/topic/com.ibm.7700.r2.user.doc/doc/r00000163.html>

When your system is deployed, the following settings are automatically configured for the HA toolkit:

- ▶ The PATH environment variable on each host is set with the HA toolkit installation directory (`/usr/IBM/analytics/ha_tools`).
- ▶ The configuration settings in the `ha_tools.conf` file on each host are set for your system.

**Note:** If you update a configuration parameter in the `ha_tools.conf` file, you must copy the change to all other hosts in the system.

- ▶ Passwordless SSH is configured for the root user on all hosts.

## 8.4.2 Starting and stopping resources with the HA toolkit

To start and stop core warehouse resources, run the **hastartdb2** and **hastopdb2** commands in the HA toolkit. With these commands, you can either start and stop the resources on all hosts or you can start and stop resources on a specific set of hosts. You must have root authority on the core warehouse hosts.

When you use the HA toolkit to stop resources in the core warehouse, the commands do not check for resource dependencies with components that run outside of the core warehouse. Before you stop the resources in the core warehouse, verify that there are no active workloads running in the core warehouse. If active workloads are running, stopping resources in the core warehouse can result in interrupted transactions on components that run outside of the core warehouse.

If you do not specify any options when you run the **hastartdb2** or **hastopdb2** commands, the commands run synchronously and do not return until the start or stop operation is complete. When the **hastartdb2** command is run synchronously, the core warehouse database is activated as part of the start operation. The database is defined by the DBNAME field in the configuration file for the HA toolkit.

If you specify the **-async** option, the commands run asynchronously and return immediately. To verify that the resources started or stopped correctly when the commands are asynchronous, run the **hal s** command to monitor the status of the resources.

When you run the **hastartdb2** command or the **hastopdb2** command, the following core warehouse resources are started or stopped on each host:

- ▶ An aggregate database partition resource that represents all of the database partitions on the host
- ▶ A service IP resource for the internal application network
- ▶ For the administration host only, the service IP resource for the corporate network

The **hastartdb2** and **hastopdb2** commands enforce the dependencies between the database partition resources and the mount points for the file systems that are managed by IBM General Parallel File System (GPFS™) software. When the core warehouse peer domain is online, the operational state of the GPFS file systems should always be Online, which indicates that they are mounted. When you run the **hastopdb2** command, the database partition resources are stopped but the file systems that are managed by the GPFS software remain mounted.

Perform the following tasks on any core warehouse host:

- ▶ To start the database partition resources, run one of the following commands:
  - **hastartdb2**  
This command starts all database partition resources on all core warehouse hosts (the administration host, the standby administration host, the data hosts, and the standby data hosts).
  - **hastartdb2 *host\_list***  
This command starts the database partition resources on specific hosts in the core warehouse, where ***host\_list*** represents a space-delimited list of host names.  
  
For example, the command **hastartdb2 bcu001 bcu002** starts the resources on the bcu001 and bcu002 hosts only.
- ▶ To stop the database partition resources, run one of the following commands:
  - **hastopdb2**  
This command stops all database partition resources on all core warehouse hosts.
  - **hastopdb2 *host\_list***  
This command stops the database partition resources on specific hosts in the core warehouse, where ***host\_list*** represents a space-delimited list of host names.

**Note:** The file systems that are managed by the GPFS software remain mounted.

### 8.4.3 Monitoring the status of the core warehouse HA configuration

To monitor the status of the highly available configuration for the core warehouse, use the system console and the **ha1s** command of the HA toolkit. You can identify whether the core warehouse is in a good state or that one or more core warehouse resources have failed over.

You must have the system administration role with at least read-only permissions to perform this task in the system console, and you must either have root authority on the core warehouse hosts or be logged in as the core warehouse instance owner.



The sample output that is provided in this section is based on a core warehouse with the following two roving high availability (HA) groups:

- ▶ The first roving HA group includes the administration host (bcu001) and the standby administration host (bcu002).
- ▶ The second roving HA group includes two data hosts (bcu003 and bcu004) and one standby data host (bcu005).

To monitor the status of the core warehouse HA configuration, complete the following steps:

1. Log in to the system console, and determine whether there are any alert events with the type `Resource Failover` and a severity of `Critical`. Events with these values indicate that a failover of the core warehouse resources was attempted and did not complete. You must identify the cause of the failover and correct the problem immediately to restore the operation of the core warehouse. If there are critical resource failover events, complete the following steps:
  - a. As the root user or core warehouse instance owner, run the **lssam** command on any core warehouse host and identify the resources with `Failed Offline` states.
  - b. Determine the cause of the `Failed Offline` state for each resource and correct the problem.
  - c. To clear the `Failed Offline` states, run the **hareset** command as the root user on any core warehouse host.
2. If the system console displays events with the type `Resource Failover` and a severity of `Informational`, this indicates that a failover occurred in the core warehouse. The core warehouse remains operational after the failover, but to restore the HA configuration to a good state, you must address any `Failed Offline` resource states on the failed host so that it can be reintegrated into the system as a standby host. While the failed host remains offline, if there is a subsequent failure in the same roving HA group, there is an outage of the core warehouse. Complete the following steps:
  - a. As the root user or core warehouse instance owner, run the **lssam** command on any core warehouse host and identify the resources with `Failed Offline` states.
  - b. Determine the cause of the `Failed Offline` state for each resource and correct the problem.
  - c. To clear the `Failed Offline` states, run the **hareset** command as the root user on any core warehouse host.

3. If there are no Resource Failover events, you can use the **ha1s** command to determine the status of the HA configuration for the core warehouse. On any core warehouse host, run the **ha1s -core** command.
  - If all resources display a Normal HA status and an Online operational state (OPSTATE) similar to the sample output that is shown in Figure 8-11, it indicates that your core warehouse HA configuration is running in a good state.

PARTITIONS	CURRENT	STANDBY	OPSTATE	HA STATUS	RG REQUESTS
0-4	bcu001	bcu002	Online	Normal	-
5-12	bcu003	bcu005	Online	Normal	-
13-20	bcu004	bcu005	Online	Normal	-

Figure 8-11 HA status - normal and online

- If the resources display an HA status of Manual Mode, it indicates that automation for the core warehouse peer domain is disabled. When automation is disabled, the HA configuration for the core warehouse is not monitored for failures and the operational state of resources that is returned by the **ha1s** command might be incorrect.
- If any resources have an HA status of Pending, this indicates that the resources are in the process of starting or in the process of stopping. The resources might be starting or stopping as the result of you running the **hastartdb2** or **hastopdb2** command, as the result of a move by running the **hafailover** command, or because the resources are failing over.

Check the system console for any alerts with the type of Resource Move or Resource Failover. If the resources are starting or stopping as the result of a manual failover, the Pending state is cleared automatically after several minutes elapse, and the system console displays alerts to indicate that the core warehouse and the core warehouse high availability are running correctly.

After the database partition resources come online, the resources display an Online state in the **ha1s** output. The database crash recovery process might be waiting to roll forward or roll back active work in the core warehouse database. To verify whether there are pending rollback or rollforward operations, where **bcudb** represents the name of the core warehouse database, run the **db2pd -recovery -db bcudb** command.

- If any resources have an HA status of Failed, Stuck, or Offline (Figure 8-12), there might be a hardware or software problem and additional troubleshooting might be required.

PARTITIONS	CURRENT	STANDBY	OPSTATE	HA STATUS	RG REQUESTS
0-4	bcu001	bcu002	Online	Normal	-
5-12	bcu003	bcu005	Failed	Offline	-
13-20	bcu004	bcu005	Online	Normal	-

Figure 8-12 HA status - Failed, Stuck, or Offline

Typically, a resource with a Failed operational state, as shown in Figure 8-12, requires additional troubleshooting. After you resolve the problem that is preventing the resource from starting, run the **hareset** command to reset the HA configuration.

4. If a failover is occurring in your core warehouse HA configuration, run the **hals** command to verify that the failover completes. Similar to the following output, if all resources have an HA status of Normal and an operations state of Online, and the host name of the previous standby host (bcu005) is now listed in the “Current” column, it indicates that the standby host assumed the role of an active data host and can start processing new workloads for the database partitions it hosts. In this scenario, the failover completed and the core warehouse HA configuration are running in a good state, as shown in Figure 8-13.

PARTITIONS	CURRENT	STANDBY	OPSTATE	HA STATUS	RG REQUESTS
0-4	bcu001	bcu002	Online	Normal	-
5-12	bcu005	bcu003	Online	Normal	-
13-20	bcu004	bcu003	Online	Normal	-

Figure 8-13 HA status - Normal and Online

## 8.4.4 Monitoring the status of the management host HA configuration

To monitor the status of the highly available (HA) resources for the database performance monitor and the warehouse tools, use the system console and the **hals** command of the HA toolkit. You must be assigned the system administration role with at least read-only permissions to perform this task in the system console and you must have root authority on the management hosts.

To monitor the status of the management host HA configuration, complete the following steps:

1. Log in to the system console, and determine whether there are any alert events with the type `Resource Failover` and a severity of `Critical`. Events with these values indicate that a failover of the database performance monitor was attempted and did not complete. You must identify the cause of the failover and correct the problem immediately to restore the operation of the database performance monitor. If there are critical resource failover events, complete the following steps:
  - a. As the root user, run the `lssam` command on the management host and identify the resources with `Failed Offline` states.
  - b. Determine the cause of the `Failed Offline` state for each resource and correct the problem.
  - c. To clear the `Failed Offline` states, run the `hareset` command as the root user on the management host.
2. If the system console displays events with the type `Resource Failover` and a severity of `Informational`, this indicates that a failover occurred for the database performance monitor. The database performance monitor remains operational after the failover, but to restore the HA configuration to a good state, you must address any `Failed Offline` resource states on the failed host so that it can be reintegrated into the system. While the failed management host remains offline, if there is a subsequent failure in the management HA pair, there is an outage of all resources on the management host. Complete the following steps:
  - a. As the root user, run the `lssam` command on the management host and identify the resources with `Failed Offline` states.
  - b. Determine the cause of the `Failed Offline` state for each resource and correct the problem.
  - c. To clear the `Failed Offline` states, run the `hareset` command as the root user on any core warehouse host.
3. If there are no `Resource Failover` events, you can run the `hals` command to determine the status of the database performance monitor and the warehouse tools. On the management host, run the `hals` command.

If all resources display a `Normal` HA status and an `Online` operational state (OPSTATE) similar to the sample output that is shown in Figure 8-14 on page 139, it indicates that the HA resources on the management host are running in a good state.

MANAGEMENT DOMAIN						
COMPONENT	PRIMARY	STANDBY	CURRENT	OPSTATE	HA STATUS	RG REQUESTS
WASAPP	bcu01	bcu02	bcu01	Online	Normal	-
DB2APP	bcu01	bcu02	bcu01	Online	Normal	-
DPM	bcu01	bcu02	bcu01	Online	Normal	-
DB2DPM	bcu01	bcu02	bcu01	Online	Normal	-

Figure 8-14 HA status - Normal, Online

If the resources display an HA status of Manual Mode, it indicates that automation for the management peer domain is disabled. When automation is disabled, the HA configuration for the management host is not monitored for failures and the operational state of resources that is returned by the **ha1s** command might be incorrect.

If any resources have an HA status of Pending, this indicates that the resources are in the process of starting or in the process of stopping. The resources might be starting or stopping as the result of a failover because they were started or stopped by the **hafailover** command, or because they were started or stopped by an HA toolkit command (**hastartdpm**, **hastopdpm**, **hastartapp**, or **hastopapp**).

Check the system console for any critical database performance monitor or warehouse tools alerts. If the resources are starting or stopping as the result of a manual failover, the Pending state is cleared automatically in several minutes, and the system console displays alerts to indicate that the database performance monitor or warehouse tools are running correctly.

- If a failover is occurring on your management host, run the **ha1s** command to verify that the failover completes. Similar to the following output, if all resources show a Failover HA status and an Online operational state, and the host name of the standby management host is listed in the "Current" column, it indicates that the failover completed successfully and the HA resources are running on the standby management host, as shown in Figure 8-15.

MANAGEMENT DOMAIN						
COMPONENT	PRIMARY	STANDBY	CURRENT	OPSTATE	HA STATUS	RG REQUESTS
WASAPP	bcu01	bcu02	bcu01	Online	Normal	-
DB2APP	bcu01	bcu02	bcu01	Online	Normal	-
DPM	bcu01	bcu02	bcu02	Online	Failover	-
DB2DPM	bcu01	bcu02	bcu02	Online	Failover	-

Figure 8-15 HA status - failover occurred

## 8.4.5 Moving database partition resources to the standby node as a planned failover

To perform maintenance on a single node, fail over the database partition resources to the standby node. When the resources are failed over, the standby node becomes the active data node and can start processing new workloads. When the node is reintegrated into the system after the node maintenance is complete, it acts as the standby node.

Typically, you update the system by installing fix packs through the system console.

When the database partition resources are moved to the standby node as part of a planned failover, a temporary outage of the core warehouse instance is required. Because this procedure requires a temporary outage of the core warehouse instance, you should perform the procedure during a scheduled maintenance window. If you perform the procedure when the core warehouse instance is active, your system users might experience disconnected sessions or transaction rollbacks.

To perform a planned failover, move the database partition resources from an active host to a standby host by running the **hafailover** command of the HA toolkit. In the example that is shown in Figure 8-16, the resources on the data host (bcu004) are moved to the standby data host (bcu003). Before the resources are moved, the **hal**s output shows that the database partition resources are running on the active hosts in the core warehouse.

PARTITIONS	CURRENT	STANDBY	OPSTATE	HA STATUS	RG REQUESTS
0-4	bcu002	bcu001	Online	Normal	-
5-12	bcu004	bcu003	Online	Normal	-
13-20	bcu005	bcu003	Online	Normal	-
21-28	bcu006	bcu003	Online	Normal	-

Figure 8-16 HA status - resource moved

To move the database partition resources to the standby host for maintenance, complete the following steps:

1. Verify that there are no DB2 jobs running.
2. As the core warehouse instance owner (bcuaix), connect to the core warehouse database and terminate all connections to the database by running the following command:

```
db2 force applications all
```

3. To verify that there are no resources on the target host that are in a Failed or Stuck state, Run the **lssam** command on any host in the core warehouse peer domain.

If a resource on the target host is in one of these states, the failover procedure fails. To reset the Failed or Stuck resources, run the **hreset** command on any host.

4. Fail the active host over to the standby host by running the **hafailover** command.

For example, to fail over the resources from the bcu004 host to the standby host in its HA group, run the following command:

**hafailover bcu004**

Figure 8-17 shows the **hal**s output that shows that the resources for database partitions 5 - 12 that were previously running on the bcu004 host are now running on the bcu003 host.

PARTITIONS	CURRENT	STANDBY	OPSTATE	HA STATUS	RG REQUESTS
0-4	bcu002	bcu001	Online	Normal	-
5-12	bcu003	bcu004	Online	Normal	-
13-20	bcu005	bcu004	Online	Normal	-
21-28	bcu006	bcu004	Online	Normal	-

Figure 8-17 HA status - host is failed over

**Note:** When you complete the maintenance on the node and reintegrate it into the system, you do not have to fail back the database partition resources. The node is reintegrated as the standby node.

## 8.4.6 Moving resources to the standby management host as a planned failover

To perform maintenance on the management host, fail over the database performance monitor resources and the warehouse tools resources to the standby host.

You must have root authority on the management hosts to complete the planned move.

When the resources are moved to the standby management host, the system console and event monitoring are unavailable. The core warehouse remains

online, and the database performance monitor and the warehouse tools continue to run.

To move resources to the standby management host in a planned fashion, complete the following steps:

1. Determine the host name of the host where the database performance monitor (DPM) is running. Run the following command:

**ha1s**

Identify the row in the output for the DPM component. The host name where the database performance monitor is running appears in the CURRENT column. In the sample **ha1s** output that is shown in Figure 8-18, the database performance monitor is running on the bcu01 host.

MANAGEMENT DOMAIN						
COMPONENT	PRIMARY	STANDBY	CURRENT	OPSTATE	HA STATUS	RG REQUESTS
WASAPP	bcu01	bcu02	bcu01	Online	Normal	-
DB2APP	bcu01	bcu02	bcu01	Online	Normal	-
DPM	bcu01	bcu02	bcu01	Online	Normal	-
DB2DPM	bcu01	bcu02	bcu01	Online	Normal	-

Figure 8-18 HA status - database performance monitor running

2. Fail over the resources for the database performance monitor by running the following command:

**hafailover management\_host DPM**

**management\_host** represents the host name of the host where the database performance monitor is running.

3. Determine the host name of the host where the warehouse tools are running. Run the following command:

**ha1s**

Identify the row in the output for the WASAPP component. The host name where the warehouse tools is running appears in the CURRENT column. In the sample **ha1s** output that is shown in Figure 8-19 on page 143, the warehouse tools are running on the bcu01 host.



MANAGEMENT DOMAIN						
COMPONENT	PRIMARY	STANDBY	CURRENT	OPSTATE	HA STATUS	RG REQUESTS
WASAPP	bcu01	bcu02	bcu01	Online	Normal	-
DB2APP	bcu01	bcu02	bcu01	Online	Normal	-
DPM	bcu01	bcu02	bcu01	Online	Normal	-
DB2DPM	bcu01	bcu02	bcu01	Online	Normal	-

Figure 8-19 HA status - warehouse tools running

4. Fail over the resources for the warehouse tools by running the following commands.
  - a. If there are cube servers that are not managed as highly available resources, stop the cube servers.
  - b. Fail over the resources for the warehouse tools by running the following command:
 

**hafailover *management\_host* APP**

*management\_host* represents the host name of the host where the warehouse tools are running.
  - c. If there are cube servers that are not managed as highly available resources, start the cube servers.





# Workload management

The purpose of workload management is to ensure that limited system resources are prioritized according to the needs of the business. Work that is designated as being of greatest importance to the business is given the highest priority access to system resources. In a data warehouse environment, this means that users running online queries can expect predictable and acceptable response times at the possible expense of long running batch jobs.

It is possible to monitor and manage the work at every stage of processing, from the application, the network, the database management system (DBMS), the operating system, and the storage subsystem. This chapter provides an overview of the workload management capabilities of the database.

Workload management is a continuous process of design, deployment, monitoring, and refinement. In a data warehouse environment, business priorities and the volume and mix of work are constantly changing. These changes must also be reflected in the management of resources.

DB2 workload management (WLM) is the primary tool that is available for managing an IBM InfoSphere Warehouse database workload. WLM was introduced with DB2 for Linux, UNIX, and Windows 9.5 and enhanced in every subsequent release of DB2 for Linux, UNIX, and Windows.

InfoSphere Warehouse V9.7 introduces the ability to design and implement a simplified subset of DB2 WLM functions through the administration console. InfoSphere Warehouse Version V10.1 extends those capabilities with enhancements such as integrated multi-temperature data management and the workload management dispatcher. Now, DB2 for Linux, UNIX, and Windows 10.5 includes all of the functions and tools that were offered in the previous generation of InfoSphere Warehouse editions and features, including even more WLM enhancements, such as default query concurrency management.

## 9.1 Introducing DB2 workload management

*DB2 workload management* is an infrastructure and a set of capabilities that facilitate the implementation of successful workload management solutions in various database environments. This infrastructure incorporates a core set of capabilities and a set of capabilities that are available only under license; the licensed set of capabilities is referred to as *DB2 workload manager*.

The core technology is always present and active when a DB2 database is active. All DB2 database work runs within a specific workload management configuration, even if it is only the default configuration that is provided by the DB2 database manager, including default workload and service classes that are installed as part of every DB2 database. You can use all of the workload management monitoring capabilities with this default configuration.

The DB2 workload manager capabilities that are controlled by license enable you to customize the default DB2 workload management configuration, including the ability to create the following objects:

- ▶ A workload
- ▶ A service class
- ▶ A threshold
- ▶ A work action set

DB2 workload management has four clearly defined stages, which are outlined in the following sections.

### 9.1.1 Defining business goals

Understand the work that is running on your data server and how that work relates to your business commitments. If formal service level agreements (SLAs) exist, ensure that the work that is associated with those agreements can be identified.

An example of such a goal is “queries from application A should run in under 30 seconds.” Goals can also be tied to a particular time of day. For example, overnight batch ETL work might have to finish by 8 a.m. so that the daily sales reports can be completed on time. Some goals can be difficult to quantify. Examples of such goals might be “keep users satisfied” and “prevent aberrant database activity from hampering day-to-day work.” Other goals might be less specific, for which it might be appropriate to set priorities in broad categories, such as “low”, “medium”, and “high”.

If clear business requirements do not exist, classify work by its expected impact on the server by using categories such as “short”, “medium”, and “long”. Treat short requests as being most important and long requests as being least important. The rationale is that the more short queries you can push through the system, the better the service that is provided to the enterprise as a whole.

### **9.1.2 Identifying work that enters the data server**

Identify workload types, ideally activities with different performance or monitoring objectives. The following list gives examples of possible activity groupings:

- ▶ Queries from specific user departments
- ▶ Queries from a specific application
- ▶ Batch data extracts
- ▶ ETL processes
- ▶ DDL work by the DBA

### **9.1.3 Managing work in progress**

Determine whether your system can run work without any restrictions. You can submit only as much work as your system can handle. Divide the work based on its importance to the business without exceeding your system’s capabilities.

Work can be managed in the following ways:

- ▶ By queuing queries. Set a maximum concurrency level for each query type and make queries that exceed that limit wait until a running query completes execution.
- ▶ By setting resource priorities. When one segment of the workload is allocated a higher priority, it is more likely to obtain the resources that it needs to continue.
- ▶ By terminating jobs that exceed a preset limit. Set a failsafe to kill long-running queries that might have been submitted in error.

## 9.1.4 Monitoring to ensure that the data server is being used efficiently

Use *real-time operational data* (such as a list of running workload occurrences, the activities running in a service class, or average response times) to determine the health of your system. WLM monitoring involves the usage of table functions to access in-memory data.

Table functions provide access to a set of data (such as workload management statistics) that exists inside a DB2 database as a virtual DB2 table against which you can run **SELECT** statements. You can write applications to query this data and to analyze it as though it were part of a physical table on the data server.

General statistical information is available at the following levels:

- ▶ Service superclass and service subclasses
- ▶ Workloads
- ▶ Work action sets
- ▶ WLM queues

Table functions also provide information about the work that is running on the system. This information is available at the following levels:

- ▶ Service class agents
- ▶ Workload occurrences
- ▶ Workload occurrence activities
- ▶ Activity details

Example 9-1 shows an example query that queries the `WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES` table function and returns a list of the workload occurrences. Figure 9-1 on page 149 shows a sample output.

*Example 9-1 Getting a list of current workload occurrences*

---

```
SELECT SUBSTR(SERVICE_SUPERCLASS_NAME,1,19) AS SUPERCLASS_NAME,
       SUBSTR(SERVICE_SUBCLASS_NAME,1,18) AS SUBCLASS_NAME,
       SUBSTR(CHAR(MEMBER),1,4) AS MEMB,
       SUBSTR(CHAR(COORD_MEMBER),1,4) AS COORDMEMB,
       SUBSTR(CHAR(APPLICATION_HANDLE),1,7) AS APPHNDL,
       SUBSTR(WORKLOAD_NAME,1,22) AS WORKLOAD_NAME,
       SUBSTR(CHAR(WORKLOAD_OCCURRENCE_ID),1,6) AS WLO_ID
FROM TABLE(WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES
(CAST(NULL AS VARCHAR(128)), CAST(NULL AS VARCHAR(128)), -2))
AS SCINFO
ORDER BY SUPERCLASS_NAME, SUBCLASS_NAME, MEMB, APPHNDL,
WORKLOAD_NAME, WLO_ID
```

---

SUPERCLASS_NAME	SUBCLASS_NAME	MEMB	COORDMEMB	APPHNDL	WORKLOAD_NAME	WLO_ID
SYSDEFAULTUSERCLASS	SYSDEFAULTSUBCLASS	0	0	383	SYSDEFAULTUSERWORKLOAD	1

1 record(s) selected.

Figure 9-1 Sample output from Example 9-1

*Event monitors* capture detailed activity information (per workload, work class, or service class, or when a threshold is exceeded) and aggregate activity statistics for historical analysis.

DB2 WLM uses the following event monitors:

- ▶ *Activity event monitors* capture information about individual activities in a workload, work class, or service class that violated a threshold.
- ▶ *Threshold violations event monitors* capture information when a threshold is exceeded. They identify the threshold, the activity that was the source of the exception, and what action was taken in response to the violation.
- ▶ *Statistics event monitors* serve as a low-impact alternative to capturing detailed activity information by collecting aggregate data (for example, the number of activities that are completed, or the average execution time).

**Note:** There is no impact for unused WLM event monitors. Create event monitors so that individual workloads, service classes, and work actions can be altered to capture events when needed.

To store the events in DB2 tables, see the sample DDL in `~/sqllib/misc/wlmevmon.ddl`.

You can find examples of Perl scripts for the analysis of workload management in the following files:

- ▶ `~/sqllib/samples/perl/wlmhist.pl`
- ▶ `wlmhistrep.pl`

## 9.2 Workload management administrator authority

You need workload management administrator authority (WLMADM) authority to manage workload objects for a specific database. This authority enables you to create, alter, drop, comment on, grant access to, or revoke access from DB2 workload management objects, which are system objects similar to other objects, such as buffer pools and table spaces.

The security administrator, who is someone holding SECADM authority, or a user with ACCESSCTRL authority can grant WLMADM authority to a user, group, or role.

WLMADM gives you the authority to perform the following tasks:

- ▶ Issue **CREATE**, *ALTER*, **COMMENT ON**, and **DROP** statements for the following DB2 workload management objects:
  - Histogram templates
  - Service classes
  - Thresholds
  - Work action sets
  - Work class sets
  - Workloads
- ▶ Issue **GRANT** and **REVOKE** statements for workload privileges

If you hold database administrator (DBADM) authority, you hold WLMADM authority implicitly.

## 9.3 Workload management concepts

The following sections explain the basic concepts around DB2 WLM.

### 9.3.1 Workloads

A *workload* is an object that is used to identify incoming work based on its source so that the work can be managed correctly. The workload attributes are assigned when the workload establishes a database connection. Examples of workload identifiers include the application name, system authorization ID, client user ID, and connection address.

### 9.3.2 Work classes and work class sets

The type of work can be identified through *work classes*, which are grouped into *work class sets*. Examples of work types that can be identified by using work classes include READ (such as SQL **SELECT** and XQuery), WRITE (such as SQL insert, update, delete, and merge), CALL, DML, DDL, LOAD, and ALL.



### 9.3.3 Service classes

The purpose of a *service class* is to define an execution environment in which the work can run. This environment can include available resources and various execution thresholds. When you define a workload, you must specify the service class where work that is associated with that workload runs. A default workload ensures that all data server work is running inside a service class. Service classes consist of a two-level hierarchy, in which a *service superclass* contains *service subclasses*, where the work runs.

### 9.3.4 Thresholds

*Thresholds* help you maintain stability in the system. You create threshold objects to trap work that behaves abnormally. Abnormal behavior can be identified *predictively* (before the work begins running, based on the projected impact) or *reactively* (as the work runs and uses resources). An example of work that can be controlled with thresholds is a query that consumes large amounts of processor time at the expense of all other work running on the system. Such a query can be controlled either before it begins running, based on estimated cost, or after it has begun running and while it is using more than the permitted amount of resources.

### 9.3.5 Threshold actions

The types of action that can be taken when a threshold is exceeded depend on the threshold itself and include the following actions:

- ▶ Collect data.
- ▶ Stop execution.
- ▶ Continue execution.
- ▶ Queue activities.

### 9.3.6 Work actions and work action sets

A *work action* defines an action that can be applied to a work class. If a work class is to be active and have activities that are assigned to it, there must be a work action that is defined for that work class. A *work action set* can contain one or more work actions that can be applied to activities in a specific service superclass or to the database as a whole.

- ▶ If you apply a work action set to a service superclass, the work actions can map activities to a service class, prevent execution, collect activity or aggregate data, or count the activities. Typically, a work action set maps an activity to a service subclass and has thresholds that are defined on the subclass to help manage the activity.
- ▶ If you apply a work action set to a database, the work actions can define thresholds, prevent execution, collect data, or count the activities.

### 9.3.7 Histograms and histogram templates

A *histogram* is a collection of *bins*, which are containers for collecting discrete ranges of data. In DB2 workload management, histograms are useful for workload analysis and performance tuning. You can use them to study the distribution of values, identify outliers, or compute statistics.

DB2 workload management histograms have a fixed number of 41 bins. The 40th bin contains the highest defined value for the histogram, and the 41st bin contains values that extend beyond the highest defined value.

In a multimember database environment, histograms are collected on each member. Histogram bins have the same range of values on all database members, with specific counts per bin per member. You can use the bins to analyze information on a per-member basis or globally across all database members.

Histograms are available for service subclasses, workloads, and work classes (through work actions). Histograms are collected for these objects when you specify one of the **COLLECT AGGREGATE ACTIVITY DATA**, **COLLECT AGGREGATE REQUEST DATA**, or **COLLECT AGGREGATE UNIT OF WORK DATA** clauses when creating or altering the objects. For work classes, histograms are also collected if you apply a **COLLECT AGGREGATE ACTIVITY DATA** work action to the work class.

You can optionally define a *histogram template* (by using the **CREATE HISTOGRAM TEMPLATE** statement) and specify a high bin value. All other bin values are automatically defined as exponentially increasing values that approach the high bin value. A measurement unit, which depends on the context in which the histogram template is used, is assigned to the histogram when a service subclass, workload, or work action is created or altered. The new histogram template overrides the default histogram template with a new high bin value.

## 9.4 Configuring DB2 workload management in stages

The *Implementing DB2 Workload Management in a Data Warehouse* preferred practices paper defines four stages in the evolution of a DB2 workload management configuration:

- ▶ Default workload management configuration (Stage 0)
- ▶ Untuned workload management configuration (Stage 1)
- ▶ Tuned workload management configuration (Stage 2)
- ▶ Advanced workload management configuration (Stage 3)

For a complete description of these stages, download a copy of this paper from the following web page:

[https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6\\_4434\\_aece\\_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse](https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6_4434_aece_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse)

If your system includes a data warehouse, implementing a Stage 2 (tuned) workload management configuration at a minimum is recommended. Consider evolving to a Stage 3 implementation when specific performance or strategic requirements must be addressed.

### 9.4.1 Default workload management (Stage 0)

A Stage 0 DB2 workload management configuration is the default workload management configuration that is established when a database is first created. Its objective is to separate user and system work for monitoring purposes. All user connections to a DB2 database are mapped to a default workload, SYSDEFAULTUSERWORKLOAD. All incoming user work for the database runs in the SYSDEFAULTUSERCLASS service class, system work runs in the SYSDEFAULTSYSTEMCLASS service class, and background maintenance work runs in the SYSDEFAULTMAINTENANCECLASS service class.

## 9.4.2 Untuned workload management (Stage 1)

A Stage 1 untuned workload management configuration is the result of transforming the default Stage 0 configuration by creating a service superclass and six subclasses. You can do this transformation by applying the configuration template that is described in *Implementing DB2 Workload Management in a Data Warehouse*. All work is mapped into the appropriate service subclass, based on the estimated cost and type of activity, by using a DB2 work class set.

The Stage 1 configuration has the following objectives:

- ▶ Promote database system stability by ensuring that incoming demand does not exceed capacity.
- ▶ Achieve a better understanding of the work being done and its timing through baseline monitoring. Use this feedback when you tune the settings to better reflect the environment.

## 9.4.3 Tuned workload management (Stage 2)

A Stage 2 tuned workload management configuration is the result of tuning and customizing the Stage 1 untuned configuration. Tuning and customization are accomplished by adding active concurrency thresholds to more closely model the actual working environment. A Stage 2 configuration is a stable configuration that includes a monitoring regime and is subject to periodic review.

The Stage 2 configuration has the following objectives:

- ▶ Achieve the optimal implementation of a stable and predictable system.
- ▶ Identify and address any misclassified or misbehaving work.
- ▶ Identify the primary sources of work to enable more refined monitoring of performance and resource consumption. This objective is a precursor to further Stage 3 customization.

### **Workload management configuration template**

The configuration template provides a structure that functions well for all warehouse deployments. The template consists of several predefined service classes and thresholds, and is intended to help stabilize the performance characteristics of the warehouse and providing a solid foundation for further customization. In practical terms, this means that the template is meant to drive a tuned (Stage 2) workload management configuration with a view to eventually achieving an advanced (Stage 3) configuration.

The template defines the following objects:

- ▶ A single DB2 service superclass that contains a series of service subclasses, each representing a distinct work type that is found in typical DB2 warehouse environments.
- ▶ A DB2 work class set with a unique work class definition representing each of the distinct work types.
- ▶ A DB2 work action set based on these work class definitions on the new service superclass to map incoming work to the appropriate service subclasses; all other work is left to run in the default service subclass of the new service superclass.
- ▶ Activity timeout and concurrency thresholds. The activity timeout thresholds are enabled by the template, but they only collect information about violations; they do not stop any activities that exceed the defined limit. The concurrency thresholds are created in a disabled state and are activated during the implementation process.

### **9.4.4 Advanced workload management (Stage 3)**

A Stage 3 advanced workload management configuration is any configuration that exceeds or differs significantly from a Stage 2 configuration. A Stage 3 configuration is recommended when business requirements exceed basic system stability. Such requirements might include ensuring consistent throughput for one application above all others, or offering different levels of service to users depending on their funding arrangements.

### **9.4.5 Moving from Stage 0 to Stage 2**

The following list summarizes the basic steps that you can take to systematically move your system's workload management configuration from a Stage 0 configuration to a Stage 2 configuration:

1. Apply the workload management configuration template to your system.
2. Collect baseline monitoring information about the estimated cost distribution of SQL statements on your system.
3. Adjust template work class (and possibly service class) definitions to better reflect the types and distribution of work in your environment.
4. Collect baseline monitoring information about resource consumption by service subclasses on your system.
5. Adjust the concurrency threshold definitions on the different service subclasses to better reflect preferred resource allocations.

6. Define activity thresholds to detect misclassified or misbehaving queries and prevent them from threatening the system.
7. Establish a monitoring plan to ensure the ongoing health of your Stage 2 configuration, and repeat steps 1 on page 155 - 6, as needed to keep your configuration tuned.

## 9.5 Workload management dispatcher

The workload management *dispatcher* (available since DB2 for Linux, UNIX, and Windows 10.1) lets you specifically allocate CPU resources to work that is run on a database server; CPU resource entitlements can be controlled by using CPU shares and CPU limit attributes on DB2 service classes.

The workload management dispatcher, which operates at the instance level, limits the number of running agents that are dispatched to the operating system and how long each agent is allowed to run. The term *dispatch concurrency level* refers to the number of running agents that can be dispatched simultaneously. You can set the dispatch concurrency level by using the **wlm\_disp\_concur** database manager configuration parameter.

The dispatcher manages service class CPU resource entitlements by using the following attributes:

- ▶ *Soft CPU shares* (uncapped, almost unrestricted). Use soft CPU shares to give any unused CPU resources to service classes running high priority work.
- ▶ *Hard CPU shares or CPU limits* (capped, limited). Use hard CPU shares or CPU limits to enforce controls on the CPU resource entitlements of service classes running low priority work, limiting their impact on high-priority work.

The dispatcher supports flexible CPU allocation by enabling both permanent allocations that are enforced all the time (hard CPU shares and CPU limits) and dynamic allocations that are enforced only when demand exceeds capacity (soft CPU shares).

The workload management dispatcher is disabled by default. To enable the dispatcher, set the **wlm\_dispatcher** database manager configuration parameter to YES. When the dispatcher is enabled, all work that is running in the user and maintenance service classes is under dispatcher control. Work that is running in the system service class cannot be configured for CPU resource control because critical DB2 subsystems that run in this service class are given maximum priority and are not subject to dispatcher control.

By default, the dispatcher can manage CPU resources only by way of CPU limit settings. To enable the dispatcher to manage CPU resources by using both CPU shares and CPU limits, set the `wlm_disp_cpu_shares` database manager configuration parameter to YES. You can set and adjust CPU shares and CPU limits by using the **CREATE SERVICE CLASS** and **ALTER SERVICE CLASS** statements.

A service class with *hard CPU shares* that are assigned *cannot* exceed its CPU resource entitlement to consume any unused CPU resources that become available on the host or logical partition (LPAR) if work is still running in competing service superclasses or running in competing service subclasses within the same service superclass. If competing workloads are not present, service classes with hard CPU shares are able to claim unused CPU resources.

The hard CPU shares setting is most effective when you want to prevent work running in a service class from interrupting more important work running on the host or LPAR. Assign hard CPU shares to service classes running complex or intensive queries that might otherwise degrade the performance of higher priority work because of contention for limited resources.

A service class with *soft CPU shares* that are assigned *can* exceed its CPU resource entitlement to consume any unused CPU resources that become available on the host or LPAR. If two or more service classes have soft shares, unused CPU resources become available, and there is enough CPU resource demand from each service class to consume the spare capacity, allocation of the CPU resources is done proportionally according to the relative share of each active service class.

The soft CPU shares setting is most effective for high-priority work that should be able to temporarily claim any spare CPU resource that becomes available, or for workloads that are expected to consume little resource beyond their immediate CPU requirements.

Configure a *CPU limit* to enforce a fixed maximum CPU resource entitlement for work in a service class. If a CPU limit is set for all service classes, you can reserve a portion of the CPU resource to perform work regardless of any other work running on the instance. The CPU resource allocation of any service class is computed from the shares of that service class relative to the shares of all other service classes within the instance.

Although CPU limits can be configured at either the service superclass level or the subclass level, by applying CPU limits to your superclasses and CPU shares to your subclasses, you can use the CPU limits to control the absolute CPU resource entitlement of each superclass, and the CPU shares to control the relative CPU resource entitlements of service subclasses running within those superclasses.

The workload management dispatcher always uses the most restrictive CPU limits or CPU shares assignments when allocating CPU resources to service classes. For example, if a service class reaches its CPU limit before it fully uses its shares-based CPU resource entitlement, the dispatcher uses the CPU limit.

Before enabling the workload management dispatcher for the first time, monitor your workloads to determine the relative CPU resources that they consume. This information can help you to make decisions about service class creation, CPU shares assignment, and whether to use CPU limits.

Table functions and monitor elements are provided to help you monitor the performance of the workload management dispatcher. After analyzing the collected data, you can adjust the dispatcher concurrency level or redistribute CPU entitlements by adjusting service class CPU shares and CPU limits to tune the dispatcher performance.

For a complete description of the DB2 workload management dispatcher, see the DB2 for Linux, UNIX, and Windows 10.1 information center at the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/index.jsp>

## 9.6 Default query concurrency management

DB2 for Linux, UNIX, and Windows 10.5 introduces columnar capabilities to DB2 databases, which include data that is stored with column organization and vector processing of column data. (A *column-organized table* is a table where the data pages contain column data instead of row data.) This enhancement is ideal for analytic or OLAP workloads. If most of the tables in your database are going to be column-organized tables, set the **DB2\_WORKLOAD** registry variable to **ANALYTICS** before you create the database. Doing so helps you configure memory, table organization, page size, and extent size, and enables *automated workload management*, which can improve immediately the performance of workloads with several queries running.

To ensure that heavier workloads on column-organized data do not overload the system when many queries are submitted simultaneously, there is a limit on the number of “heavyweight” queries that can run on the database concurrently. This limit can be implemented by using the default workload management concurrency threshold that is automatically enabled on new databases when the value of the **DB2\_WORKLOAD** registry variable is set to **ANALYTICS**, or that can be manually enabled on existing databases.



The processing of queries against column-organized tables is designed to run fast by using the highly parallelized in-memory processing of data. The trade-off for this high performance is a relatively large resource footprint. The running of these types of queries is optimal when relatively few of them are admitted to the system concurrently. When the limit on the number of heavyweight queries is reached, the remaining queries are queued and must wait until other queries leave the system before beginning their execution. This can help ensure system stability when many complex ad hoc queries are running on systems that have not implemented a specific workload management strategy.

### 9.6.1 Default workload management objects for concurrency control

Default query concurrency management uses the following new default workload management objects within the existing DB2 workload management infrastructure:

- ▶ A `SYSDEFAULTMANAGEDSUBCLASS` service subclass under the existing `SYSDEFAULTUSERCLASS` superclass, where heavyweight queries run and can be controlled and monitored as a group.
- ▶ A `CONCURRENTDBCOORDACTIVITIES` threshold, `SYSDEFAULTCONCURRENT`, which is applied to the `SYSDEFAULTMANAGEDSUBCLASS` subclass to control the number of concurrently running queries that are running in that subclass.
- ▶ A `SYSDEFAULTUSERWCS` work class set and a new `SYSMANAGEDQUERIES` work class, which identify the class of heavyweight queries that are to be controlled. The `SYSMANAGEDQUERIES` work class encompasses queries that are classified as `READ DML` (an existing work type for work classes) falling above a timeron threshold that reflects heavier queries.
- ▶ A `SYSDEFAULTUSERWAS` work action set and `SYSMAPMANAGEDQUERIES` work action, which map all queries that fall into the `SYSMANAGEDQUERIES` work class to the `SYSDEFAULTMANAGEDSUBCLASS` service subclass.

When a database is created or an existing database is upgraded, the following behavior applies to the new default objects:

- ▶ The work action set is enabled by default so that queries meeting the criteria specified in the `SYSMANAGEDQUERIES` work class run in the `SYSDEFAULTMANAGEDSUBCLASS` service subclass.

- ▶ If the value of the DB2\_WORKLOAD registry variable was set to ANALYTICS, the threshold on the SYSDEFAULTMANAGEDSUBCLASS service subclass is enabled by default for newly created databases only; otherwise, it is left disabled. If you want to disable the threshold later, it must be disabled manually.
- ▶ Running the DB2 Configuration Advisor causes the work action set and the work action to be enabled if either were disabled, and the threshold to be enabled or disabled, depending on whether **DB2\_WORKLOAD** was set to ANALYTICS.

## The default concurrency limit

A limit on the number of queries that can run concurrently should protect the system from overload, but also avoid over-throttling the workload to a point where performance might suffer. The default concurrency limit is calculated based on the current host configuration when the default workload management objects for concurrency control are created during database creation or upgrade. The limit is recalculated when the DB2 Configuration Advisor runs against the database. Rerun the DB2 Configuration Advisor after significant changes are made to the underlying host, such as enabling or removing CPU capacity.

For more information about how to manually adjust the default query concurrency configuration, see the DB2 for Linux, UNIX, and Windows 10.1 information center on the following web page:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/index.jsp>

## Tuning the settings

The recommended method for tuning the default object settings is to examine the overall system resource usage with a full workload running, and to perform incremental adjustments based on whether the system resources appear underutilized or overutilized. A system with a run queue depth on the order of (10 \* number of processor cores) and physical memory usage under 100% is considered to be in a fully used and healthy state. This section provides guidance about how to tune default concurrency control when the threshold is enabled and the system is running in a non-customized workload management environment.

For recommendations about tuning the default concurrency threshold value and the default work class timeron range when the system appears to be underutilized or overutilized, see the DB2 for Linux, UNIX, and Windows 10.1 information center. For a comprehensive set of recommendations that apply to monitoring both system utilization and workload characteristics, download a copy of *Implementing DB2 Workload Management in a Data Warehouse* from the following web page:

[https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6\\_4434\\_aece\\_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse](https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6_4434_aece_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse)





## Mining and unstructured text analytics

Data mining used to be the exclusive province of highly trained statisticians and mathematicians. Today, most businesses need many business analysts to pose business questions and answer them by using methodologies that are based in analytics. Embedded data mining is an approach to making sophisticated data mining methodology and technology available to many business users and enabling them to solve business problems or take better advantage of business opportunities.

This chapter describes data mining and text analytics and its implementation in the DB2 Warehouse. This includes business goals and scenarios, a description of the data mining process and unstructured (text) analytics, and the concepts of embedded data mining for deploying mining-based business intelligence (BI) solutions to many business users.

## 10.1 Overview

Essentially, *data mining* discovers patterns and relationships that are hidden in your data. It is part of a larger process that is called *knowledge discovery*, specifically, the step in which advanced statistical analysis and modeling techniques are applied to the data to discover useful patterns and relationships. The knowledge discovery process as a whole is essential for successful data mining because it describes the steps you must take to ensure meaningful results.

A pattern that is interesting (according to a user-imposed interest measure) and certain enough (again according to the user's criteria) is called *knowledge*. The output of a program that monitors the set of facts in a database and produces patterns in this sense is discovered knowledge.

*Unstructured analysis* is the technique of extracting or transforming unstructured information into a structure of interest that can be analyzed together with existing structured information in the warehouse to create business insights.

This information of interest lies in various sources, such as call center notes, problem reports, emails, product reviews. The presence or value of the information is hidden or not clear. After it is extracted, this structured data can be used to perform other statistical analysis or used as input in data mining.

DB2 Warehouse provides a robust and tightly integrated environment and tools for data mining and text analytics functions as extensions to core DB2 functions.

## 10.2 Business goals

The goal of data mining and text analytics is to produce low-volume results (only a few rules and relationships) and high-value results (discover new rules and relationships that nobody thought about looking for). It is purely a data-driven discovery approach in contrast to other BI techniques that mostly give precise and detailed answers to precisely formulated questions. The approach in intelligent mining does not assume that the user already knows what he is looking for. Begin with the premise that you do not know what patterns or relationships exist in the data.

The other BI methods (for example, OLAP) are based on hypotheses. The user in this case knows the data well enough to formulate precise queries and use known relationships in the data.

Thus, the goal of data mining and analytics is to solve business problems by using the technology (mining algorithms) to support problem solving. Use the mining algorithms to detect and model potentially valuable but hidden patterns in your data. These techniques characterize relationships that are not expressed explicitly.

## 10.3 Business scenarios

This section describes some of the common scenarios where data mining and text analytics are applicable and useful.

### 10.3.1 In data mining

The data mining process begins with a problem that you want to solve. Because the purpose of this process is to generate results (discoveries, insights, information, and models) that help solve the business problem, you must begin by clearly stating the business problem, translating it into one or more questions that data mining can address, and understand how the data mining results are used in a BI solution to improve the business.

Data mining has many high-value uses, but it is not always the correct tool for a particular purpose. For example, if the business need is to make the sales force more productive by reducing the time that is required to gather certain information, then the correct solution might be a new reporting system that is based on real-time, multidimensional queries. Conversely, if your need is to understand client behaviors and preferences to improve targeting or promotional campaigns, the correct solution might be based on client segmentation and link analysis (associations and sequences). Understanding the business problem and how the data mining results are deployed into the business enable you to determine the best approach to solving the problem.

Stating the business problem clearly and concisely forces you to focus on the root issue of what must change to enable you to accomplish a specific goal. After you identify the root issue and determine that data mining is appropriate, you can formulate one or more specific questions to address that issue.

For example, to design a new targeted promotion, you can ask the following questions:

- ▶ What do my customers look like? What are their behavioral and demographic profiles?
- ▶ Which customers should I target in a promotion? Who is the best target audience?

- ▶ Which products should I use for the promotion?
- ▶ Which products should I replenish in anticipation of a promotion?
- ▶ How should I lay out my stores based on customers preference?
- ▶ What item is a customer most likely to purchase next?

Data mining is used successfully in many different business and scientific fields, including retail, banking, insurance, telecommunications, manufacturing, healthcare, and pharmaceuticals. For example, in the retail, financial, and telecom industries, well-known uses of data mining include client segmentation for targeted marketing and fraud detection, store profiling for category and inventory management, associations for cross-selling and upselling, and predictive techniques for client retention and risk management. In manufacturing, data mining applications include profiling and predictive methods for quality assurance, warranty claims mitigation, and risk assessment. In healthcare and pharmaceutical research, high-value uses include associations, sequences, and predictive techniques for disease management, associations and prediction for cost management, and patient segmentation for clinical trials. In every case, the data mining results must be conveyed in such a way that someone can make better decisions or formulate an action to solve the business problem.

### **10.3.2 In text analytics and unstructured analysis**

Likewise, for text analysis, you solve the problem of identifying insights that are lost or buried in the unstructured data, and make them available in a structured form to allow them to be consumed by data mining or OLAP reports.

For example, a huge amount of unstructured information (text) such as call center notes, problem reports, repair reports, insurance claims, email with customers, and product reviews exist in enterprises. This unstructured information cannot be used with existing BI tools to create insight.

Although there are great insights in the structured data, so many insights are lost because of the difficulty in identifying them in the vast amount of unstructured data that the organization uses to run their business. Instead, the data warehouse contains too much backward-oriented (historic) financial data that does not answer important business questions, which is a problem. This affects business analysts in medium to large enterprises, which leads to wrong decisions because of missing information.



You can try to solve this problem using Text Analytics by taking the following actions:

- ▶ Find the top issues in (unstructured) in the call center log and perform causal analysis by relating these issues to structured data that is associated with the log entry.
- ▶ Identify and address the top types of problems that are encountered by the most profitable customers to reduce loyal customer churn.
- ▶ Analyze customer ratings and customer surveys about your and your competitors' products
- ▶ Analyze call center notes to find interest in other (cross-sell) or new products (growth).
- ▶ Detect early warning signals in problem reports to avoid costly recalls and lawsuits.
- ▶ Analyze claims data and patient records to identify insurance fraud (health insurance).

So, you might end up taking some of the following actions:

- ▶ Create simple reports: To identify the top 10 customer satisfaction problems that are recorded in a text field of a customer survey.
- ▶ Create multidimensional reports: In retail, you might want to derive the new Return reasons OLAP dimension from text analysis and combine it with existing dimensions, such as time, geography, or product.
- ▶ Generate additional input fields for data mining to improve the predictive power of data mining models: You might want to use symptoms that are included in patient records to improve the predictive power of a data mining model. That model can predict the patients who need treatment based only on available structured data, such as patient age or blood pressure and related conditions.

## 10.4 Data mining techniques and process

This section describes the *discovery* and *predictive* methods that are used in data mining and the techniques of each method. It also gives an overview of the various steps in the process of implementing a data mining solution.

## 10.4.1 Data mining techniques

Data mining techniques can be divided into two broad groups:

- ▶ Discovery
- ▶ Predictive

*Discovery methods* are data mining techniques that find patterns that exist in the data, but without any prior knowledge of what those patterns might be. The objective is to discover the relationships that are inherent in the data.

There are three discovery methods:

- ▶ *Clustering* involves grouping data records into segments by how similar they are based on the attributes of interest. Clustering can be used, for example, to find distinct profiles of clients with similar behavioral and demographic attributes to create a client segmentation model.
- ▶ *Associations* are a type of link analysis that finds links or associations among the data records of single transactions. A common usage of the Associations method is for market basket analysis, which finds what items tend to be purchased together in a single market basket, such as chips and soda.
- ▶ *Sequences* are a type of link analysis, sequential patterns, that find associations among data records across sequential transactions. A store can use sequential patterns to analyze purchases over time and, at checkout time, use that model to print customized discount coupons for the customers to use on their next visit.

*Predictive methods* are data mining techniques that can help predict categorical or numeric values.

There are three predictive methods:

- ▶ *Classification* is used to predict values that fall into predefined buckets or categories. For example, they can predict whether a particular treatment cures, harms, or has no effect on a particular patient.
- ▶ *Regression* is used to predict a numerical value on a continuous scale, for example, predicting how much each customer spends in a year. If the range of values is 0 - 1, this becomes a probability of an event happening, such as the likelihood of a customer leaving.
- ▶ *Time series forecasting* predicts the values of a numerical data field for a future period. You can use a time series model to predict future events based on known past events. For example, forecasting can be used to create stock level forecasts to reduce warehouse costs.

For more information about the discovery and predictive mining methods that are available in DB2 Warehouse, see *Dynamic Warehousing: Data Mining Made Easy*, SG24-7418.

## 10.4.2 Data mining process: preparing, building, and deploying

The approach to data mining can be summarized in a series of steps, as shown in Figure 10-1, after you understand that mining is the analytical approach that best fits your goals and requirements, that is, your business problem.

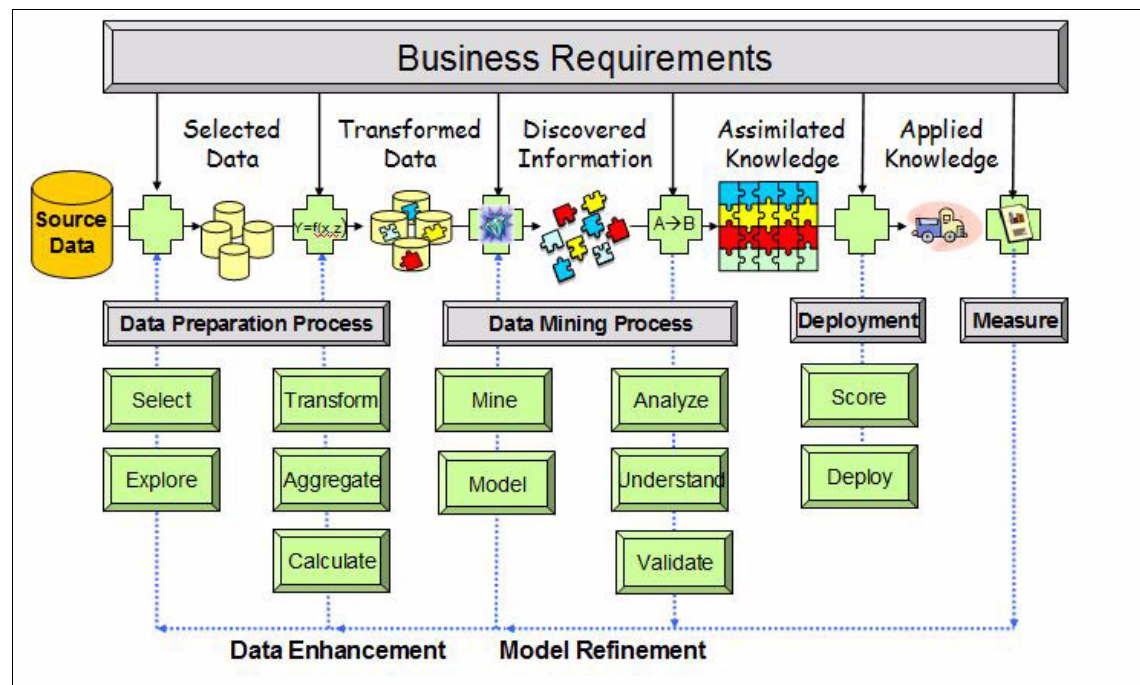


Figure 10-1 Data mining process

The data mining process includes the following steps:

1. Define the data mining approach: Break down the business problem into components and determine whether each component can be solved by using data mining or text analytics and which technique is suitable for that component.
2. Perform data preparation for data mining: Consider the data that is available, and understand what data types and formats (and the ETL process that is required to extract, transform, derive, and format the data) are needed to enable you to achieve your goal.

3. Perform the data mining process: Build models to run the wanted technique with the appropriate parameters. Modeling is an interactive and iterative process as the initial results are reviewed, model parameters are adjusted to produce a better model, and any additional data preparation is performed.
4. Interpret and evaluate results: Visualize the model to help interpret the result, and assess the model quality and determine whether the model fulfills its business purpose. Improvements to the input data, model parameters, and modeling technique can be made to obtain a model that meets the objective.
5. Deploy the solution: The final and most important step in the data mining process because how and where the results are deployed are crucial to realizing the maximum value from the data mining. This step leans heavily on the usage of scoring techniques and scoring results (apply a data mining model to generate a prediction for each record, depending on the type of model).

## 10.5 Data mining and text analytics in DB2 Warehouse

The Mining and Analytic capability in DB2 is a suite of statistical, preprocessing, and mining functions that you can use to analyze large databases. It also provides visualization tools for viewing and interpreting mining results.

Mining functions use elaborate mathematical techniques to discover hidden patterns in your data. After you interpret the results of your data mining process, you can modify your selection of data, data processing and statistical functions, or mining parameters to improve and extend your results.

Data mining is an iterative process that typically involves selecting input data, transforming it, running a mining function, and interpreting the results. The DB2 Intelligent Miner for Data assists you with all the steps in this process. You can apply the functions of the DB2 Intelligent Miner for Data independently, iteratively, or in combination.

Figure 10-2 shows the shared and integrated environment that greatly enhances ease-of-use and accessibility to data mining by many users with various needs and skill sets, not just a small group of experts, which eliminates the need to move data to a separate analytic environment.

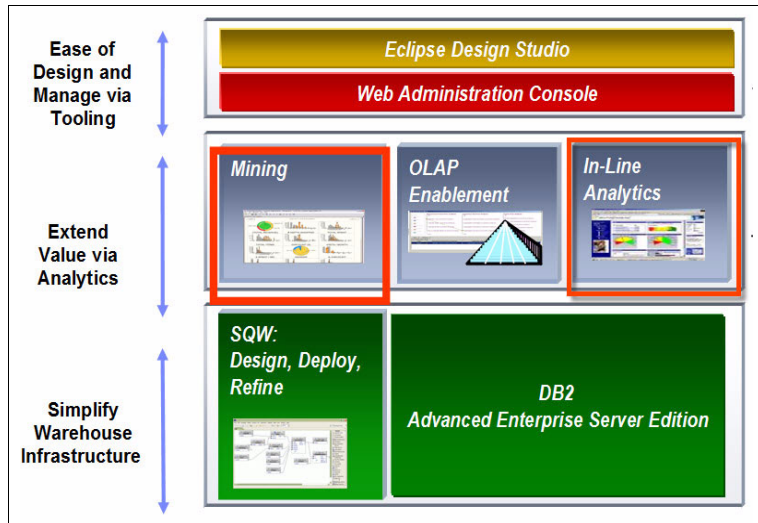


Figure 10-2 Data mining and text analytics in a DB2 Warehouse environment

## 10.5.1 Data mining

DB2 Warehouse has three components (Figure 10-3) for DB2 based data mining to support each of the steps of the mining process:

- ▶ Modeling component
- ▶ Visualization component
- ▶ Scoring component

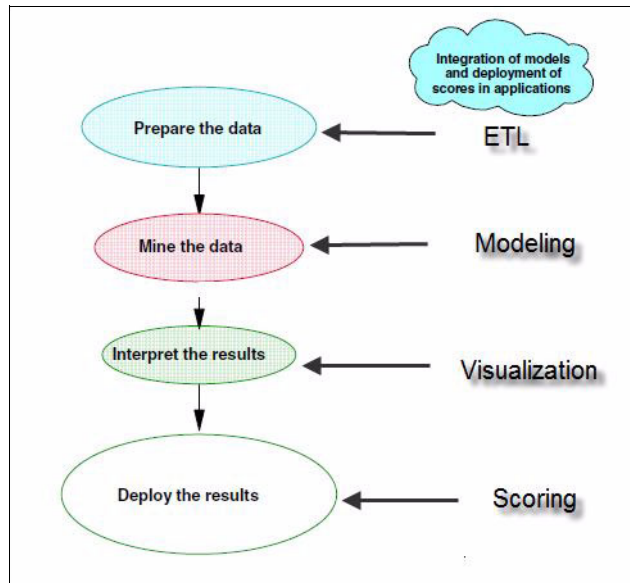


Figure 10-3 Data Warehouse components

### Modeling

The modeling component is a DB2 SQL application programming interface (API) that is implemented as a DB2 extender. Modeling is accessed graphically through the Design Studio to build data mining models from information in DB2 databases.

All six discovery and predictive methods of data mining describe in 10.4.1, “Data mining techniques” on page 168 have multiple algorithms to provide more flexibility in model development.

These IBM DB2 Extenders™ provide a set of SQL stored procedures and user-defined functions to build a model and store it in a DB2 table. These procedures and functions are collectively referred to as *easy mining procedures*. As the model is set up through graphical wizards in Design Studio, the easy mining procedures use the wizard inputs to automatically create mining tasks that specify the type of model to be built, the parameter settings, the data location, and data settings (for example, which columns to use in the model), and to call the appropriate mining kernel in DB2.

A DB2 table containing columns representing the behaviors and other attributes of the records (such as clients, stores, accounts, and machines), including the response or outcome, if any, is used as the data source for building (training) the model and, for predictive mining, validating (testing) the model.

The new model is stored in Predictive Model Markup Language (PMML) format in a DB2 table where it is accessible for deployment.

## Visualization

The visualization component in DB2 Warehouse is a Java application that uses SQL to call and graphically display PMML models, enabling the analyst to assess a model's quality, decide how to improve the model by adjusting model content or parameters, and interpret the final model results for business value.

Visualization has visualizers for all six mining methods in the modeling. The visualizers are tailored to each mining method and provide various graphical and tabular information for model quality assessment and interpretation in light of the business problem.

Visualization can also display PMML models that are generated by other tools if the models contain appropriate visualization extensions, such as quality information or distribution statistics that are produced by modeling. These model types do not contain much extended information and do not present well in visualization.

## Scoring

Like modeling, the scoring component is implemented as a DB2 extender. It enables application programs by using the SQL API to apply PMML models to large databases, subsets of databases, or single records. Because the focus of the PMML standard is interoperability for scoring, scoring supports all the model types that are created by modeling, and selected model types that are generated by other applications (for example, SAS and IBM SPSS®) that support PMML models.

Scoring also supports the radial basis function (RBF) prediction technique, which is not yet part of PMML. RBF models can be expressed in XML format, enabling them to be used with scoring.

Scoring includes scoring JavaBeans, which enables the scoring of a single data record in a Java application. This capability can be used to integrate scoring into client-facing or e-business applications, for example.

A PMML model is either created and automatically stored by modeling or created by another application and imported into DB2 by using the scoring component's SQL import function. Accessed through Design Studio, scoring applies the PMML model to new data. The score that is assigned to each record depends on the type of mining model. For example, with a clustering model, the score is the best-fit cluster for a given record. The results are written to a new DB2 table or view, where they are available to other applications for reporting or further analysis, such as OLAP.

## **10.5.2 Unstructured analysis**

DB2 Warehouse uses the Unstructured Information Management Architecture (UIMA) information extraction implementation to support analysis of unstructured data. UIMA is an open, scalable, and extensible platform for creating, integrating, and deploying text-analysis solutions.

Figure 10-4 on page 175 shows the architecture of the UIMA integration.



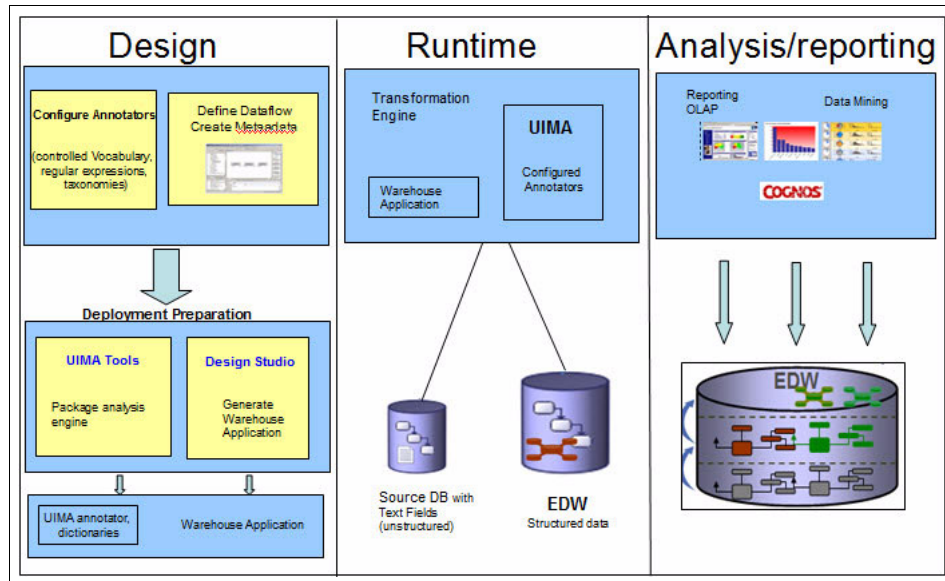


Figure 10-4 The architecture of the UIMA integration

The middle part shows the Text Analytics transformation run time, which is embedded in the database server and runs SQL-based transformations in the DB2 database.

Before you can run the transformation flows, you must design them. The design time components on the left part of Figure 10-4 consist of two parts.

- A workbench to configure the text analysis engines (or in UIMA terminology, annotators). For example, if you have a rule-based annotator, you must specify the rules, depending on your business problem and text corpus. If you have a list- or dictionary-based annotator, you must be able to specify the list of words to be used. UIMA Annotators or Analysis Engines are UIMA-based components that are used to extract entities such as names, sentiments, or relationships.
- The second part, after configuring the analysis engine, is used to define the transformation flows themselves. Specify the input table to be analyzed and the configured analysis engine to be used, and map the analysis results to columns and tables in the database.

After you convert the text into structure, you can use existing and well-known reporting and analysis tools from IBM (for example, Cognos).

UIMA text analytics can be considered an “ELT for text” and extract structured information from the text. The extracted information is stored in a relational database (DB2 Data Warehouse) and can be used to describe OLAP metadata (Cubing Services). Reporting or OLAP tools, such as Cognos, can use that metadata to create reports on the combination of the pre-existing structured data and the structured data that is obtained from the text analysis.

Combining the results of text analysis on unstructured textual information with structured data in data mining allows you to combine the analysis of structured and unstructured information to create new insight. Data mining algorithms can also, by using the text analysis results, improve the predictive power of the data mining models.

Step-by-step details to create data mining and text analysis flows using Design Studio in DB2 Warehouse are described in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813.



## Providing the analytics

Online Analytical Processing (OLAP) is a powerful data analytical method that is a core component of data processing. It enables users to interrogate data by intuitively navigating data from a summary level to a detail level.

The tools that are available in the DB2 Advanced Enterprise Server Edition bundle, such as IBM Cognos Business Intelligence and IBM InfoSphere Warehouse Cubing Services, provide OLAP capability to navigate data that is contained in enterprise DB2 data warehouses.

This chapter focuses on the Relational OLAP (ROLAP) capabilities of Dynamic Cubes feature of IBM Cognos Business Intelligence V10.2. This feature enables high-speed interactive analysis and reporting of terabytes of data. Dynamic Cubes uses in-memory member and data caches, and its awareness of in-memory and in-database aggregates, to provide fast query performance for many users.

This chapter provides an overview of the Dynamic Cubes architecture, and modeling and deploying OLAP functions by using IBM Cognos Business Intelligence. It also describes the Dynamic Cubes lifecycle.

Semantically, Cognos Dynamic Cubes is a generational evolution of the Cubing Services technology. Integration between Dynamic Cubes and IBM InfoSphere Warehouse Cubing Services is also described.

## 11.1 Implementing OLAP

OLAP is a key component of many data warehousing and business intelligence projects and often is a core requirement of users. OLAP enables users (business analysts, managers, and executives) to gain insight into data through a fast, consistent, and interactive interface, enabling increased productivity and faster decision making.

### 11.1.1 Dimensional model

In a dimensional data warehouse, you model relational database tables by using a star or snowflake schema. This type of data warehouse stores information about the data in fact and dimension tables, describing the relationships within the data by using joins between the dimension and fact tables, the collection of dimension keys in a fact table, and the different attribute columns in a dimension table.

The OLAP style of reporting is performed by using a specific type of data organization called *multidimensional modeling*. Multidimensional modeling organizes the data into facts (also called measures) and dimensions. Facts are the data values that you want to analyze. Dimensions contain the values of how you want to view the data. For example, a “sales amount” is a fact that you want to measure, when “store”, “region”, and “time” are the dimensions (or how you want to see the data presented). This is called viewing sales by store, by geography, and by time, and is shown in Figure 11-1.

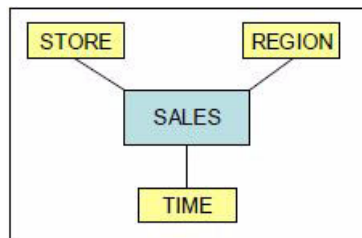


Figure 11-1 A simple star-schema

In addition to the relationship between fact and dimensions, there are data relationships that are present in a dimension. Attributes in a dimension represent how data is summarized or aggregated. These are organized in a hierarchy.

Figure 11-2 shows a typical time hierarchy. Sales Amount in a specific store, in a specific region, can be aggregated at Year, Month, and Day levels.

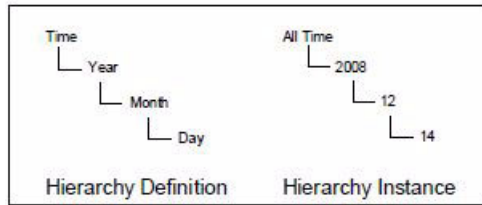


Figure 11-2 A simple time hierarchy

**Note:** It is important to ensure that the snowflake or star schema dimensions and facts that are used in the data warehouses do not have cases where the fact records have unknown or invalid dimension key values and also invalid snowflake dimensions (where an outer table does not have values for the inner table).

To ensure this situation, enforce referential integrity. Typically, referential integrity is turned off or is made declarative and instead is enforced during extract, transform, and load (ETL) processing. Erroneous modifications that are made to the data during or outside of the ETL process can create cases where a fact table has no matching dimension records.

### 11.1.2 Providing OLAP data

IBM Cognos Business Intelligence provides a proven enterprise BI platform with an open-data access strategy. It enables customers to pull data from various data sources, package the data in to a business model, and make the data available to consumers in various interfaces that are suited to the task, using business language that is relevant to consumers.

Figure 11-3 shows how Cognos Dynamic Cubes is tightly integrated in to the Cognos BI stack, and its data can be displayed through any of the Cognos interfaces. This method allows existing customers to integrate this technology into their application environment without affecting existing users, who are already familiar with interfaces such as Report Studio, Business Workspace, and Business Workspace Advanced.

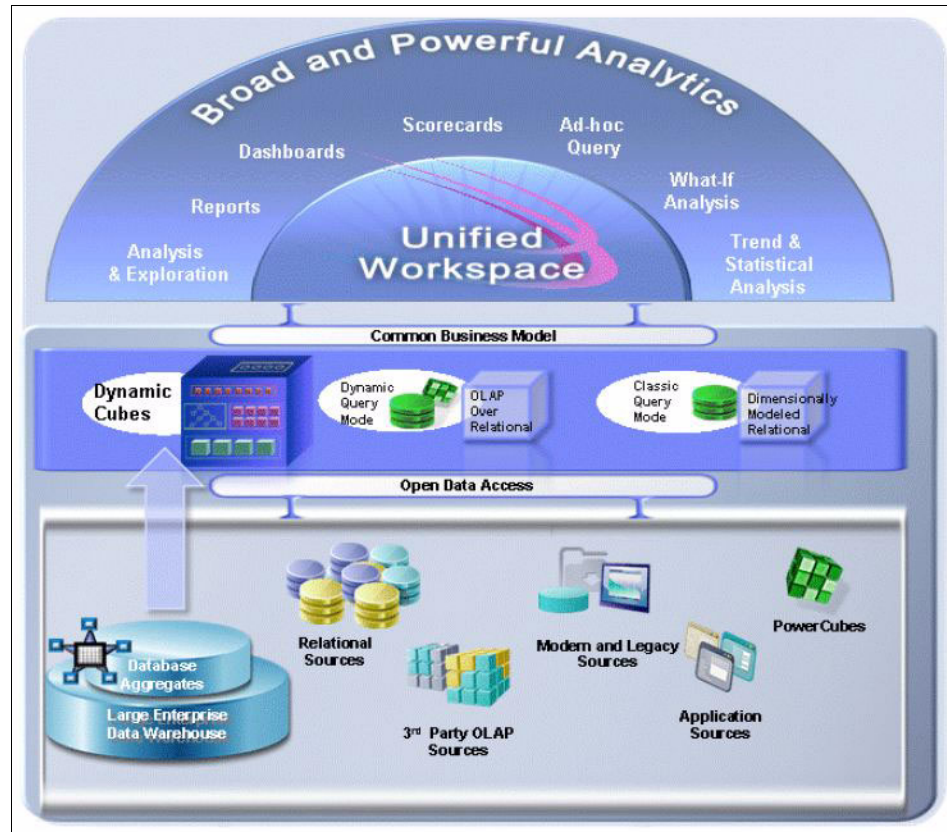


Figure 11-3 Dynamic Cubes in the integrated Cognos BI stack

Cognos Dynamic Cubes uses the database and data cache for scalability, and also uses a combination of caching, optimized aggregates (in-memory and in-database), and optimized SQL to achieve performance. The Cognos Dynamic Cubes solution uses multi-pass SQL that is optimized for the relational database, minimizing the movement of data between the relational database and the Cognos Dynamic Cubes engine. It is aggregate-aware, and able to identify and use both in-memory and in-database aggregates to achieve optimal performance. It optimizes aggregates (in-memory and in-database) by using workload-specific analysis.

This solution can achieve low latency over large data volumes, such as billions of rows or more of fact data and millions of members in a dimension.

### **11.1.3 Using OLAP data**

Data warehouses are the recognized foundation for enterprise analytics. They enable an organization to bring together cleansed data from separate sources of input, both internal and external, such as from partners or suppliers. Data warehouses enable higher-quality information and enable high-performance data access for analytic style applications.

Cognos Dynamic Cubes technology helps leverage the core strengths of an Enterprise Data warehouse (EDW) and takes the EDW to the next level of performance for analytics by making the deploying and tuning easier and faster.

### **11.1.4 Pulling it all together**

With the size of data volumes across businesses exploding in to the petabyte range these days, growth is accelerating at an unprecedented pace. Understanding your own business and the data holdings that you own are more important than ever. Analytics is key to understanding and optimizing the data and providing concrete business results. With data volumes exploding in size, and with the arrival of a growing urgency to make sense of this sea of data, using a data warehouse technology and maintaining a connection between your data warehouse and your analysis tools helps rationalize your investment in this technology.

Easy visibility into enterprise data is a critical step on the journey to gain insight from various sources, whether those sources are traditional or emerging. Semi-structured or unstructured data makes sense only within the context of your business. Your data warehouse is this context is.

Cognos Dynamic Cubes gives customers an additional tool in the query stack to handle the growing business problem of high-performance analytics over large data warehouses.

Because it can provide a correctly designed data warehouse (in addition to using database server performance features such as BLU Acceleration) on a server with adequate memory to handle medium to large data volumes (25 million or more fact table rows), Cognos Dynamic Cubes is the ideal choice for your needs.

## 11.2 Cognos and the cube model

Cognos Dynamic Cubes introduces a performance layer in the Cognos Query stack to allow low-latency and high-performance OLAP analytics over large relational data warehouses. By using the power and scale of a relational database, Cognos Dynamic Cubes can provide OLAP analytics of terabytes of warehouse data.

This section provides an overview of IBM Cognos Dynamic Cubes and describes how the solution extends the scalability of the IBM Cognos Business Intelligence platform and uses the core strengths of an enterprise data warehouse. This section also describes the Cognos architecture, the IBM Cognos Cube Designer, the Cognos Dynamic Cubes Server, and the Aggregate Advisor (a part of IBM Cognos Dynamic Query Analyzer).

### 11.2.1 Cognos architecture

IBM Cognos Business Intelligence (BI) has a multitiered architecture. For description purposes, it can be separated into three tiers: web server, applications, and data. The tiers are based on business function, and are typically separated by network firewalls. IBM Cognos BI user interfaces sit above the tiers (Figure 11-4).

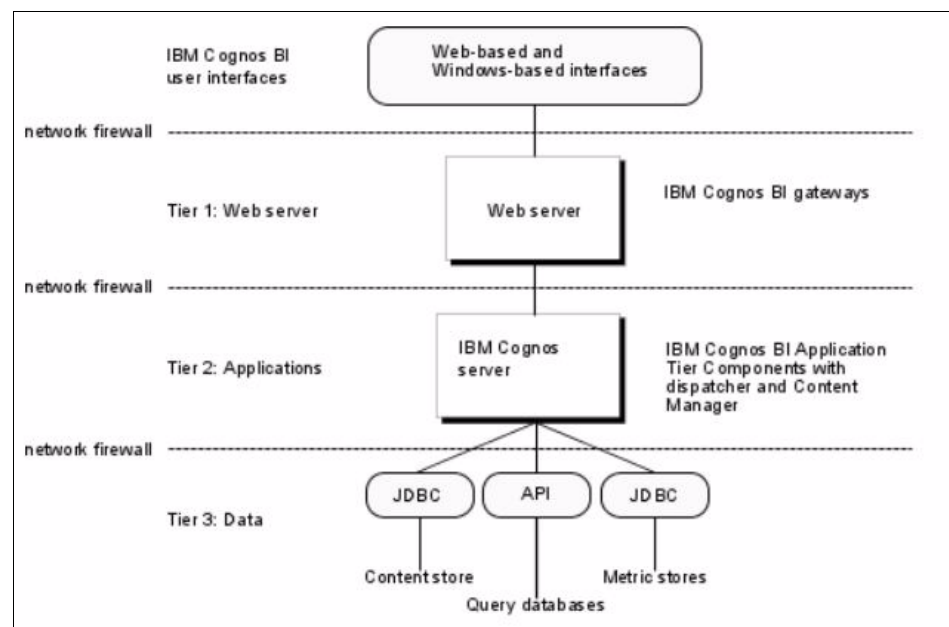


Figure 11-4 IBM Cognos Business Intelligence architecture



IBM Cognos BI is configured by using IBM Cognos Configuration, which is also used to start and stop IBM Cognos services.

In addition to IBM Cognos Configuration, IBM Cognos BI has web-based and Windows-based user interfaces, such as Framework Manager, the various Studios (for querying and analyzing data), and Cognos Administration.

The IBM Cognos BI *web server tier* contains one or more IBM Cognos BI gateways. Web communication in IBM Cognos Business Intelligence is typically through gateways, which are on one or more web servers.

The IBM Cognos BI *applications tier* contains one or more IBM Cognos BI servers. An IBM Cognos BI server runs requests, such as reports, analyses, and queries, which are forwarded by a gateway. It also renders the IBM Cognos Connection, which is the portal to all web-based interfaces.

The IBM Cognos Business Intelligence *data tier* includes the content store. The content store is a relational database that contains data that your IBM Cognos BI product needs to operate, such as report specifications, published models, and the packages that contain them, connection information for data sources, information about the external namespace and the Cognos namespace itself, and information about scheduling. The query databases that contain the data that the reports extract are also part of the data tier.

More details about these various tiers and their components can be found in the IBM Cognos Business Intelligence 10.2.0 information center (see 11.4, “Resources” on page 193).

## 11.2.2 Cognos metadata and Cognos Cube Designer

IBM Cognos Cube Designer<sup>1</sup> is the application that you use to model dimensional metadata and dynamic cubes. You use it to build dynamic cubes and publish them for use in the IBM Cognos Studios. To get started, you import metadata from a relational database. Using the metadata, you model dynamic cubes and save the cube definitions in a project. After you publish the cubes, they are listed as data sources in Content Manager and their related packages are available to report authors.

If you want to model dimensional metadata and dynamic cubes based on a relational database, you import the metadata from a Content Manager data source.

---

<sup>1</sup> There are two modeling tools that are available in IBM Cognos BI V10.2: Framework Manager is a metadata modeling tool that is used for designing dimensionally modeled relational (DMR) metadata. Dynamic Cubes can be modeled by using the Cube Designer only.

You can import cube metadata from an IBM InfoSphere Warehouse Cubing Services model. IBM Cognos Cube Designer creates a project with a separate dynamic cube for each cube that is contained in the imported model.

### 11.2.3 Dimensional metadata and dynamic cubes

Understanding concepts that relate to dimensional metadata and dynamic cubes helps you plan and create effective dynamic cubes.

In Cognos Dynamic Cubes, dimensional metadata refers to dimensions and hierarchies. You can create commonly used dimensional metadata that is independent of any dynamic cubes in a project. The appropriate dimensional metadata can then be shared by one or more cubes in a project.

A dynamic cube represents a multidimensional view of a star or snowflake schema. It is based on a single fact table and defines the relationships between dimensions and measures.

A basic dynamic cube contains the following items:

- ▶ A measure dimension that contains at least one measure
- ▶ At least one dimension
- ▶ At least one hierarchy and that associated levels that are defined for each dimension
- ▶ Mappings between the measures and dimensions
- ▶ Attributes that reference table columns either directly, by expressions, or by an expression that is a constant value

## 11.3 Dynamic Cubes architecture and its lifecycle

Cognos Dynamic Cubes adds relational online analytic processing (ROLAP) capabilities to the IBM Cognos Dynamic Query Mode (DQM) server in Version 10.2 of IBM Cognos, enabling reporting and ad hoc analysis.

This section provides an overview of the various components of the Cognos Dynamic Cubes architecture in relationship to the main activities that are performed during a lifecycle by using the corresponding tools.

Figure 11-5 on page 185 shows the lifecycle of Dynamic Cubes.

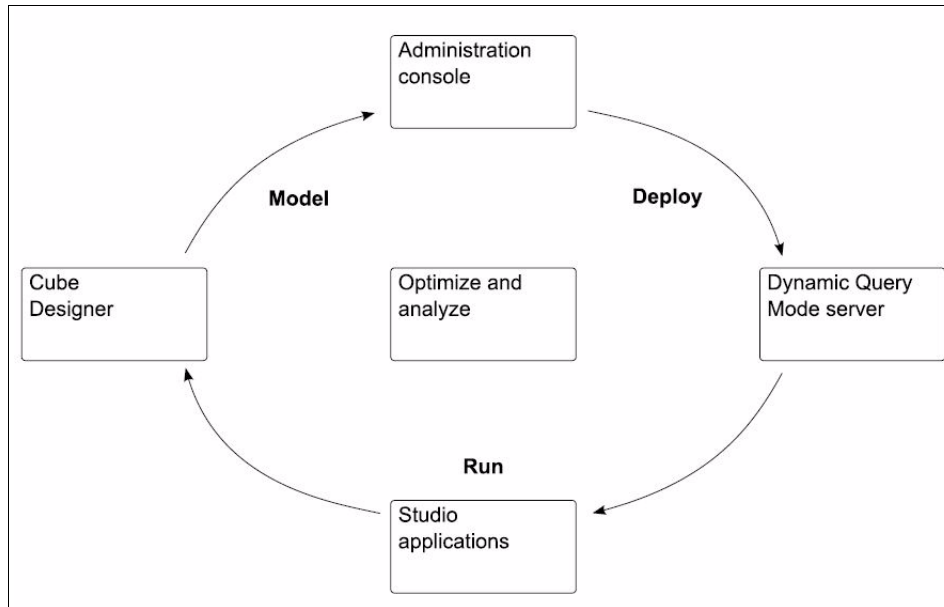


Figure 11-5 Lifecycle of Dynamic Cubes

IBM Cognos Cube Designer provides dynamic cube design and modeling capability. The Administration Console is used to deploy and manage the cube data. The IBM Cognos Dynamic Query Mode (DQM) server maintains the cube data. Studio applications use the data in reporting environments. In addition, various tools, such as Dynamic Query Analyzer, are used to analyze and optimize the data as necessary, such as the Aggregate Advisor.

### 11.3.1 Design and model phase

Cognos Cube Designer is a modeling tool that brings together the best modeling principles from past successful modeling technology, with a modern and extensible architecture. The first step to deploying Cognos Dynamic Cubes is to provide a model by using the Cognos Cube Designer. During the modeling phase of the lifecycle, IBM Cognos Cube Designer is used to perform the following tasks:

- ▶ Importing relational metadata to use as the basis for dynamic cube design
- ▶ Designing dynamic, aggregate, and virtual cubes
- ▶ Setting cube-level security for hierarchies and measures
- ▶ Publishing the dynamic cube

### 11.3.2 Deploy phase

During the deploy phase of the lifecycle, an administrator uses a published dynamic cube as a data source and configures the cube in IBM Cognos Administration and makes it available for use. The administrator might perform the following tasks:

- ▶ Adjust the memory allocation for the query service and in-memory aggregates.
- ▶ Assign users, groups, and roles to security views.
- ▶ Monitor cube metrics.
- ▶ Start and stop cubes.
- ▶ Refresh the caches.
- ▶ Enable logging.
- ▶ Schedule administrative tasks.

The Cognos Dynamic Cubes server manages all aspects of data retrieval, and leverages memory to maximize responsiveness while giving you the full flexibility to manage what is contained in memory and when you want to refresh in-memory data. You manage dynamic cubes in the Cognos Administration environment.

### 11.3.3 Run phase

After the dynamic cube is configured and started, the report author can create reports by using the various reporting applications, such as Business Insight Advanced, Report Studio, or Analysis Studio. In the run phase of the lifecycle, users can do the following tasks:

- ▶ Validate the design of the cube.
- ▶ Author reports.
- ▶ Run reports.
- ▶ Evaluate the performance to determine whether any optimization is required.

### 11.3.4 Optimization

If the performance of the reports does not meet your expectations, optimization is performed to achieve the wanted results. By using Cognos Administration, you can adjust various performance parameters.

Cognos Dynamic Cubes supports two types of pre-computed aggregate values: those stored in database tables and those stored in its in-memory aggregate cache. For warehouses that do not yet have aggregates, or want to supplement existing database aggregates with in-memory and other in-database aggregates, the Aggregate Advisor in IBM Cognos Dynamic Query Analyzer can be used to generate various recommendations.

## **Aggregate Advisor**

Aggregate Advisor is a tool that is available with IBM Cognos Dynamic Query Analyzer that can analyze the underlying model in a dynamic cube data source and recommend which aggregates to create. These aggregates can be created both in-database and in-memory.

Aggregate Advisor can also reference a workload log file that can help Aggregate Advisor suggest aggregate tables (in-database or in-memory) that correspond directly to the reports that are contained in the log file. Before running the Aggregate Advisor, the expectation is that the dynamic cube is published in the Content Store, can be started successfully, and that reports and analysis run and return correct results.

The usage of aggregates can improve the performance of queries by providing data that is aggregated at levels higher than the grain of the fact.

## **Database aggregates**

Database aggregates are tables of pre-computed data that can improve query performance by reducing the number of database rows that are processed for a query.

Cognos Dynamic Cubes brings up the aggregate routing logic into its query engine, where the multidimensional OLAP context is preserved and Cognos Dynamic Cubes can better determine whether to route to aggregates.

Cognos Dynamic Cubes can also use aggregate tables that the database optimizer does not know about (for example, MQTs that are not enabled for queries, or regular tables that have aggregated data) or the database optimizer might not route to because of complex OLAP-style SQL. After a database aggregate table is modeled as an aggregate cube, Cognos Dynamic Cubes can select data directly from that database aggregate table.

## **In-memory aggregates**

In addition to ensured routing to aggregates in the database, a major performance feature of Cognos Dynamic Cubes is its support of aggregates that are in memory. In-memory aggregates provide precomputed data and can improve query performance because the aggregated data is stored in memory in the aggregate cache of the dynamic cube. This situation avoids the impact of transferring data from the data warehouse to the BI server.

In-memory aggregates are aggregate tables that can be created in memory by the IBM Cognos Business Intelligence server every time the cube is started or the data cache is refreshed. The definition of these aggregates is stored in the content store.

In-database aggregates are built during a predefined ETL processing “window”. In-memory aggregates are built during cube-start. The Aggregate Advisor is used to recommend additional in-database aggregate tables and a set of in-memory aggregates.

### **11.3.5 Cognos Dynamic Cubes caching**

The basis for the performance of Cognos Dynamic Cubes is its various in-memory caches.

The aggregate and data caches in Cognos Dynamic Cubes provide fast query response and help alleviate the processing load on the underlying relational database because Cognos Dynamic Cubes does not have to re-retrieve data that is held in its various caches.

Figure 11-6 on page 189 shows the extensive caching structures that are used in a Cognos Dynamic Cubes solution.

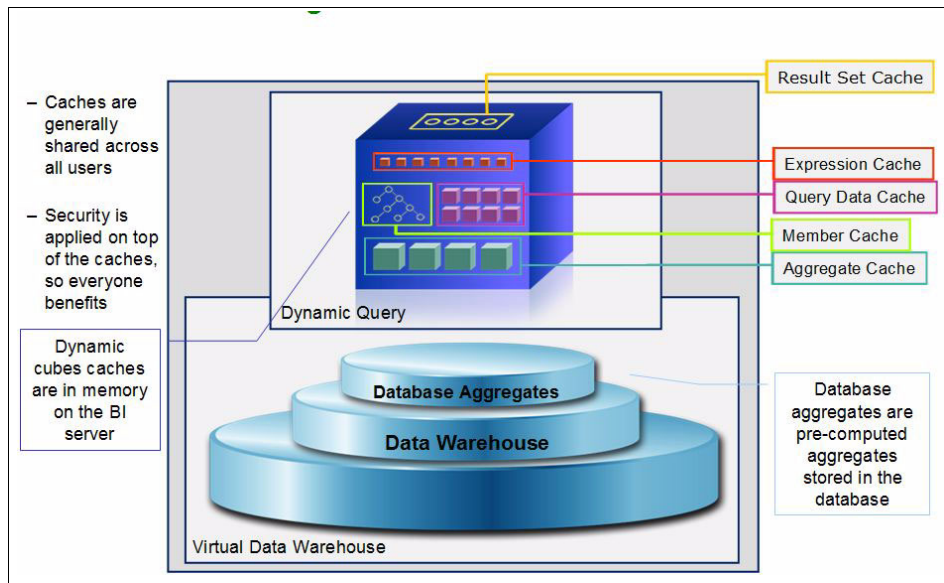


Figure 11-6 Cognos Dynamic Cubes caching model

Cognos Dynamic Cubes uses five caches, each with a separate purpose. When users run the system and run queries, individual data items are loaded in memory. Because security is applied on top of the dynamic cube, the reuse of data is maximized, allowing as many users and reports as possible to benefit from previous queries.

## Data cache

The data cache contains the result of queries that are posed by the MDX engine to a dynamic cube for data. When users run the system and run queries, individual data items are loaded in memory. Because security is applied on top of the dynamic cube, the reuse of data is maximized, allowing as many users and reports as possible to benefit from previous queries.

## Member cache

Members of each hierarchy are retrieved by running a SQL statement to retrieve all of the attributes for all levels within a hierarchy, including multilingual property values, and are stored in memory. The parent-child and sibling relationships that are inherited in the data are used to construct the hierarchical structure in memory. When you start a cube, all members are loaded in to memory. This approach ensures a fast experience as users explore the metadata tree. This approach also helps the server maximize its ability to run the most efficient queries possible.

The member cache must be flushed when there is a change to the dimension/hierarchy (add or delete members and so on). Any flush to the member cache also flushes the data cache.

### **Result set cache**

When a report is run, it is saved in memory for later reuse. For commonly used reports, this way allows the engine to benefit from previous runs, giving users the fastest response time possible.

This cache stores the result set of each multidimensional expression (MDX) query that is run through the report, and it is stored within the on-disk result set cache. Each MDX query that is planned for execution is first searched for in the result set cache. So, if the query was run, then the result set of the query is obtained from the result set cache after applying security view rules for that user. The result set cache, besides sharing report execution results between users, also speeds navigation operations, because the navigation involves the rerunning of a query.

### **Expression cache**

To accelerate query planning, the engine saves expressions that are generated when queries are run. This way, query planning can be accelerated for future queries when expressions can be reused.

As with the result set cache, the intermediate results that are stored in the expression cache are security-aware and are flushed when the data cache is refreshed.

Because the expression cache is stored in memory and is so closely associated with the data cache, the expression cache is stored within the space that is allotted to the data cache.

### **Aggregate cache**

In-memory aggregates contain measure values that are aggregated by the members at the level of one or more hierarchies within the cube. These values can be used to provide values at the same level of aggregation.

The aggregates that are recommended by the Aggregate Advisor and its size are estimated at the time of the advisor run. The value of this estimate grows as the size of the data warehouse grows at a scale relative to the size of the member cache. The amount of memory that is allocated to this cache is determined by the Maximum amount of memory to use for aggregate cache cube property (set from IBM Cognos Administration). This property determines the maximum size that is allocated to this cache. An aggregate that cannot fit in to the cache is discarded.



Any action to refresh member cache, data cache, or cube start /restart initiates the loading of the aggregate cache. Cube metrics that are available in Cognos Administration can be used to monitor and indicate when aggregates complete loading and to monitor the aggregate cache hit rate (along with the hit rates of the result set cache, data cache, and database aggregate tables).

A request for data from the MDX engine can be satisfied by data that exists in the data or aggregate cache. However, if data is not present in either cache to satisfy the request for data, or if only part of it can be retrieved from the cache, dynamic cubes obtain the data either by aggregating data in the aggregate cache or by retrieving data from the underlying relational database. In either case, the data obtained is stored in the data cache as a cubelet, which is a multidimensional container of data. Thus, the data cache is a collection of such cubelets that are continuously ordered in such a manner as to reduce the time that is required to search for data in the cache.

All the caches are built at the time of cube start and are flushed out when the cube stops/restarts.

### 11.3.6 Cognos Dynamic Cubes and the data warehouse

There are two key areas in which interaction occurs between the Cognos Dynamic Cubes layer and the data warehouse:

- During the cube start when the member cache and aggregate cache are initialized. For the cube to become active (or be started), it must retrieve all the member information from the raw dimension tables in the data warehouse. After that data is retrieved and stored in memory, the cube is “started” and reports can be accessed. However, performance is not optimized until the aggregate cache is loaded. The loading of the aggregate cache is parallelized based on the configured maximum number of aggregate load threads. When an aggregate is loaded into the in-memory cache, it is available for use in a report.

The number of queries that are posed concurrently to populate the in-memory aggregate cache can be controlled by an advanced property of Cognos Dynamic Cubes (reduce the value for the `qsMaxAggregateLoadThreads` property by using Cognos Administration) to ensure that the underlying relational database is not saturated with concurrent requests computing summary values. Fewer threads more require more overall time to load the aggregates.

- When queries are pushed to the data warehouse is in report processing. To answer a report request, Cognos Dynamic Cubes uses a waterfall lookup method. If all the required data is not present in the in-memory cache, then queries are pushed to the database.

In summary, at dynamic cube start:

- ▶ A member cache is filled with queries to data warehouse dimension tables.
- ▶ An aggregate cache is filled with queries to database aggregates (if defined) and data warehouse.

During report processing, waterfall lookup for data in descending order on the following caches is performed until all data is provided for the query:

- ▶ Result set cache
- ▶ Query data cache
- ▶ Aggregate cache
- ▶ Database aggregate
- ▶ Data warehouse

### 11.3.7 Cognos Dynamic Cubes and DB2 with BLU Acceleration

The performance of cube start and querying can be further accelerated by taking advantage of BLU Acceleration in DB2 10.5.

The “load and go” simplicity of BLU Acceleration helps implement a far more efficient DB2 data warehouse and analytical environment. BLU Acceleration is integrated with DB2. It does not require SQL or schema changes to implement; therefore, setup is easy. You can easily convert tables to the column-organized format by using the new **db2convert** command-line utility or by using similar tools in IBM Data Studio V4.1, which can convert any number of tables from row to column organization

In addition to an easier setup, there is less ongoing maintenance. There are no indexes or materialized query tables (MQTs) to define or tune, which reduces storage costs and issues.

On the analytics side, the metadata model for Dynamic Cubes is transferable, that is, you can use the same model for the same row-based and column organized tables. There is no need to revalidate reports for the column organized data warehouse (the SQL that is generated by the Cognos reports and queries remains unchanged). BLU Acceleration reuses the same SQL compiler and optimizer as the DB2 row-based engine.

The various caches that are used by Cognos still must be primed and are used to further accelerate performance. However, storage and optimization of the data warehouse and a significant amount of effort towards its maintenance is drastically reduced in this scenario.

Thus, DB2 with BLU Acceleration with IBM Cognos Business Intelligence work together to deliver blazing fast Business Analytics. You can analyze key facts and freely explore information from multiple angles and perspectives to make more informed decisions about enterprise volume levels of data at breakthrough speeds.

## 11.4 Resources

For more information about many of the concepts that are described in this chapter, see the following resources:

- ▶ IBM Cognos dynamics FAQ:  
<http://www-01.ibm.com/support/docview.wss?uid=swg27036155>
- ▶ *IBM Cognos Dynamic Cubes Version 10.2.1.1 User Guide*:  
[http://public.dhe.ibm.com/software/data/cognos/documentation/docs/en/10.2.1/ug\\_cog\\_rlp.pdf](http://public.dhe.ibm.com/software/data/cognos/documentation/docs/en/10.2.1/ug_cog_rlp.pdf)
- ▶ IBM Cognos Business Intelligence 10.2.0 information center:  
<http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp>
- ▶ *IBM Cognos Dynamic Cubes*, SG24-8064:  
<http://www.redbooks.ibm.com/abstracts/sg248064.html>



# Related publications

The publications that are listed in this section are considered suitable for a more detailed discussion of the topics that are covered in this book.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Some publications referenced in this list might be available in softcopy only.

- ▶ *IBM Cognos Dynamic Cubes*, SG24-8064
- ▶ *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

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

## Other publications

These publications are also relevant as further information sources:

- ▶ *Analytics: The new path to value*, found at:  
[http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=PM&subtype=XB&appname=GBSE\\_GB\\_TI\\_USEN&htmlfid=GBE03371USEN&attachment=GBE03371USEN.PDF](http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=PM&subtype=XB&appname=GBSE_GB_TI_USEN&htmlfid=GBE03371USEN&attachment=GBE03371USEN.PDF)
- ▶ *Best Practices: Optimizing analytic workloads using DB2 10.5 with BLU Acceleration*, found at:  
[https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/0fc2f498-7b3e-4285-8881-2b6c0490ceb9/page/ecbdd0a5-58d2-4166-8cd5-5f186c82c222/attachment/e66e86bf-4012-4554-9639-e3d406ac1ec9/media/DB2BP\\_BLU\\_Acceleration\\_0913.pdf](https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/0fc2f498-7b3e-4285-8881-2b6c0490ceb9/page/ecbdd0a5-58d2-4166-8cd5-5f186c82c222/attachment/e66e86bf-4012-4554-9639-e3d406ac1ec9/media/DB2BP_BLU_Acceleration_0913.pdf)
- ▶ *DB2 Partitioning and Clustering Guide*, SC27-2453

- ▶ Eaton and Cialini, *High Availability Guide for DB2*, IBM Press, 2004, ISBN 0131448307
- ▶ *IBM Cognos Dynamic Cubes Version 10.2.1.1 User Guide*, found at:  
[http://public.dhe.ibm.com/software/data/cognos/documentation/docs/en/10.2.1/ug\\_cog\\_rlp.pdf](http://public.dhe.ibm.com/software/data/cognos/documentation/docs/en/10.2.1/ug_cog_rlp.pdf)
- ▶ *Implementing DB2 Workload Management in a Data Warehouse*, found at:  
[https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6\\_4434\\_aece\\_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse](https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc9a068d7f6a6_4434_aece_0d297ea80ab1/page/Implementing%20DB2%20workload%20management%20in%20a%20data%20warehouse)
- ▶ *Physical database design for data warehouse environments*, found at:  
<https://ibm.biz/Bdx2nr>

## Online resources

These websites are also relevant as further information sources:

- ▶ IBM Cognos Business Intelligence 10.2.0 information center:  
<http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp>
- ▶ IBM Cognos dynamics FAQ:  
<http://www-01.ibm.com/support/docview.wss?uid=swg27036155>
- ▶ IBM DB2 Version 10.5 information center:  
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.welcome.doc/doc/welcome.html>
- ▶ IBM Big Data:  
<http://www-01.ibm.com/software/data/bigdata>

## Help from IBM

IBM Support and downloads

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

IBM Global Services

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



**Leveraging DB2 10 for High Performance of Your Data Warehouse**

(0.2" spine)  
0.17" <-> 0.473"  
90 <-> 249 pages









# Leveraging DB2 10 for High Performance of Your Data Warehouse

**Comprehensive platform for architected data warehouse solutions**

**Faster time to value for business decision making**

**Standards-based for flexibility and change**

Building on the business intelligence (BI) framework and capabilities that are outlined in *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813, this IBM Redbooks publication focuses on the new business insight challenges that have arisen in the last few years and the new technologies in DB2 10 for Linux, UNIX, and Windows that provide powerful analytic capabilities to meet those challenges.

This book is organized in to two parts. The first part provides an overview of data warehouse infrastructure and DB2 Warehouse, and outlines the planning and design process for building your data warehouse. The second part covers the major technologies that are available in DB2 10 for Linux, UNIX, and Windows, function that helps you get the most value and performance from your data warehouse. These technologies include database partitioning, intrapartition parallelism, compression, multidimensional clustering, range (table) partitioning, data movement utilities, database monitoring interfaces, infrastructures for high availability, DB2 workload management, data mining, and relational OLAP capabilities. A chapter on BLU Acceleration gives you all of the details about this exciting DB2 10.5 innovation that simplifies and speeds up reporting and analytics. Easy to set up and self-optimizing, BLU Acceleration eliminates the need for indexes, aggregates, or time-consuming database tuning to achieve top performance and storage efficiency. No SQL or schema changes are required to take advantage of this breakthrough technology.

This book is primarily intended for use by IBM employees, IBM clients, and IBM Business Partners.

**INTERNATIONAL  
TECHNICAL  
SUPPORT  
ORGANIZATION**

**BUILDING TECHNICAL  
INFORMATION BASED ON  
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)