IBM® Information Management Software

# Architecting and Deploying DB2 with BLU Acceleration

**DB2 with BLU Acceleration deployment details**

**Integration with SAP Business Warehouse information**

**Examples with Cognos Business Intelligence**


Whei-Jen Chen
Brigitte Bläser
Marco Bonezzi
Polly Lau
Jean Cristie Pacanaro
Martin Schlegel
Ayesha Zaka
Alexander Zietlow


Redbooks

ibm.com/redbooks

International Technical Support Organization

# Architecting and Deploying DB2 with BLU Acceleration

October 2014

SG24-8212-01

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

**Second Edition (October 2014)**

This edition applies to IBM DB2 for Linux, UNIX, and Windows Version Cancun Release 10.5.04, SAP NetWeaver 7.40 with SAP BASIS SP5, SAP Business Warehouse 7.40 SP5, and Cognos Business Intelligence 10.2.

# Contents



# 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®
BigInsights™
Cognos®
DB2 Connect™
DB2®
DS8000®
FlashSystem™
GPFS™
IBM FlashSystem™
IBM PureData™
IBM®
Information on Demand®
InfoSphere®
Insight™
Optim™
PartnerWorld®
Passport Advantage®
Power Systems™
POWER7+™
POWER7®
PowerVM®
POWER®
PureApplication®
PureData™
pureScale®
Redbooks®
Redbooks (logo) ®
System x®
Tivoli®
WebSphere®
XIV®

The following terms are trademarks of other companies:

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

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

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

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

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

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

# Preface

IBM® DB2® with BLU Acceleration is a revolutionary technology that is delivered in DB2 for Linux, UNIX, and Windows Release 10.5. BLU Acceleration delivers breakthrough performance improvements for analytic queries by using *dynamic in-memory* columnar technologies. Different from other vendor solutions, BLU Acceleration allows the unified computing of online transaction processing (OLTP) and analytics data inside a single database, therefore, removing barriers and accelerating results for users. With observed hundredfold improvement in query response time, BLU Acceleration provides a simple, fast, and easy-to-use solution for the needs of today's organizations; quick access to business answers can be used to gain a competitive edge, lower costs, and more.

This IBM Redbooks® publication introduces the concepts of DB2 with BLU Acceleration. It discusses the steps to move from a relational database to using BLU Acceleration, optimizing BLU usage, and deploying BLU into existing analytic solutions today, with an example of IBM Cognos®. This book also describes integration of DB2 with BLU Acceleration into SAP Business Warehouse (SAP BW) and SAP's near-line storage solution on DB2.

## 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.

**Brigitte Bläser** is a Senior Software Engineer at the IBM Böblingen Lab. She is a member of the joint IBM and SAP development team that enables SAP applications for DB2 for Linux, UNIX, and Windows. She leads the SAP NetWeaver Business Warehouse (BW) development team that is responsible for integrating DB2 for Linux, UNIX, and Windows solutions into SAP Business Warehouse (SAP BW) applications. Brigitte joined IBM in 1984 and worked as a Software Engineer in many database-related projects. She has more than 20 years of experience with DB2 and 15 years of experience with SAP basis and SAP BW technology.

**Marco Bonezzi** is a Technical Enablement Specialist for DB2 in the Information Management Data Ecosystem team, based in the IBM Dublin Lab in Ireland. He has a Masters degree in Computer Engineering and Embedded Systems. His career started with an internship at IBM Spain in the IBM System x® Technical PreSales team. Before his current role, Marco worked with databases from different vendors including DB2, high availability architectures, and replication environments. His experience is based both on technical pre-sales roles and as project consultant, participating in customer projects mainly involving databases and high availability architectures.

Since he joined IBM in Ireland, Marco has concentrated on DB2 and high availability solutions such as HADR, IBM pureScale®, and IBM PureData™ for Transactions; and currently his main focus is DB2 with BLU Acceleration.

**Polly Lau** is a Solution Architect in the Information Management Data Ecosystem team, working at the IBM Toronto Lab. She is currently focusing on DB2 with BLU Acceleration and big data technologies. Before this role, Polly was a manager of a worldwide team focused on Business Partner technical enablement and consulting. She and her team enabled Business Partners around the world about adopting InfoSphere information governance solutions for clients. As part of her job, she regularly delivered technical bootcamps for Business Partners; and she has been a frequent speaker at the annual IBM Information on Demand® conference (now known as IBM Insight™).

In her early years at IBM and previous roles in other enterprise organizations, Polly had worked as a developer, system and database administrator, and technical consultant in enterprise heterogeneous infrastructure environments.

**Jean Cristie Pacanaro** is a Senior Software Engineer in the Information Data Management Ecosystem team at IBM Software Lab in Brazil. He graduated in Computer Engineering and has over 18 years of experience in IT area. Jean has been with IBM since 2006 and has had various roles in the company, working in many divisions such as services, IT outsourcing, and currently in Software Group. Jean is a certified IT specialist. His specialty is information and data management. He is a certified DB2 specialist and also holds in-depth knowledge of other database systems such as Oracle and Microsoft SQL Server. Jean provides support for DB2 enablement projects, training, proofs of concept, and proofs of technology. Currently, he focuses on DB2 BLU engagements, enabling clients and business partners in the BLU technology.

**Martin Schlegel** is a Senior Software Engineer on the Information Management Data Ecosystem team with over 10 years of DB2 experience working at the IBM Toronto Lab in Canada.

He started his career as a Broadband Network Engineer before joining IBM. At IBM, Martin has worked in DB2 quality assurance, in DB2 development, and as an enablement consultant with IBM's largest customers. Martin has focused on DB2 performance tuning, DB2 High Availability and Disaster Recovery, including HADR, IBM pureScale, IBM PureData™ System for Transactions and their integration with deployment automation, cluster managers, and cluster file systems.

Martin is a regular speaker at the annual IBM Information on Demand conference (now known as IBM Insight), and has held many deep technical training sessions in over 17 countries worldwide for customers, Business Partners, and IBMers.

**Ayesha Zaka** is a Software Engineer in the Information Management (IM) Data Ecosystem team, working at the IBM Toronto Lab. She is currently working on DB2 with BLU Acceleration and has over four years of experience conducting technical enablement and consulting for various IM products. Ayesha started her career with DB2 9.7 and has been enabling and training business partners and clients since. At IBM, her various roles have allowed her to work with DB2, IBM InfoSphere® BigInsights™, InfoSphere Streams, and PureData for Hadoop. She has enabled business partners and clients around the world in adopting IM solutions within their organizations. As part of her role, she has delivered technical training sessions, worldwide, and has been a frequent speaker at the annual IBM Information on Demand conference (now known as IBM Insight),

**Alexander Zietlow** works as Software Engineer for the SAP/IBM DB2 Platform Development team since 2013. He works on performance evaluations of BLU Acceleration in the SAP environment. While studying for his Bachelors degree in Information Technology at Baden-Württemberg Cooperative State University Karlsruhe, he worked at SAP. He holds a Master of Computer Science degree from the Eberhard Karls University in Tübingen, Germany. His master thesis is based on an internship at IBM Research and Development Germany in Böblingen.

# Acknowledgment

The authors express their deep gratitude for the guidance they received from **Boris Bialek**, **Felix Lee**, **Tadhg Murphy**, and **Melody Ng**.

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

Amine Abbour
Jonathan Baxter
Marco Bonezzi
Madelyx Brianto
Petrus Chan
Yvonne Chan
Philip Cho
Marc Desbiens
Adolph Prashant Dsouza
Joe Duffy
Karl Fleckenstein
Douglas Gibbs
Philip Jordan
David Kalmuk
Jungsun Kim
Anton Korshunov
Al Krishnapillai
Michael Kwok
Steve La
Wladimir Lanin
Robert A Matchett
Derek McNamara
Matthias Nicola

Amyris Rada
Kendrick Ren
Jessica Rockwood
Aruna De Silva
Mike Springgay
Rainer Staib
Lip Gee Tan
Veronica Tudor
Ralf Wagner
Shu Ji Wan
Yasir Warraich
**Software group, IBM**

Karen Kuck
Sabine Reich
Torsten Ziegler
**SAP, Germany**

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

- Authors of the first edition, Architecting and Deploying DB2 with BLU Acceleration, published in June 19, 2013, were:

  Brigitte Blaser
  Aidan Craddock
  Polly Lau
  Cong Lin
  Kushal Munir
  Martin Schlegel
  Alexander Zietlow

## 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

## 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

- Send your comments in an email to:

  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

# Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-8212-01
for Architecting and Deploying DB2 with BLU Acceleration
as created or updated on May 11, 2015.

## October 2014, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

### New information

New features and functions that the BLU Acceleration supported in DB2 for Linux, UNIX, and Windows Cancun Release 10.5.04:

- Shadow tables
- High Availability and Disaster Recovery feature
- SAP support

1

# Introducing DB2 BLU Acceleration

This chapter introduces DB2 with BLU Acceleration. It describes the *Seven Big Ideas* behind this significant technology innovation from IBM and an example of how the technology works in action and under the covers. This chapter also describes technologies that work together with BLU Acceleration, and how users can get started with BLU Acceleration.

The following topics are covered:

- DB2 with BLU Acceleration
- BLU Acceleration: Seven Big Ideas
- Next generation analytics: Cognos BI and DB2 with BLU Acceleration
- IBM DB2 with BLU Acceleration offerings
- Obtaining DB2 with BLU Acceleration

## 1.1 DB2 with BLU Acceleration

BLU Acceleration is one of the most significant pieces of technology that has ever been delivered in DB2 and arguably in the database market in general. Available with the DB2 10.5 release, BLU Acceleration delivers unparalleled performance improvements for analytic applications and reporting using dynamic in-memory optimized columnar technologies.

A brainchild of the IBM Research and Development labs, BLU Acceleration is available with DB2 10.5 Advanced Enterprise Server Edition, Advanced Workgroup Server Edition, and Developer Edition. While the industry is abuzz with discussion of in-memory columnar data processing, BLU Acceleration offers so much more. It delivers significant improvements in database compression but does not require you to have all your data in memory.

Although BLU Acceleration is important new technology in DB2, it is built directly into the DB2 kernel as shown in Figure 1-1. It is not only an additional feature; it is a part of DB2, and every part of DB2 is aware of it. BLU Acceleration still uses the same storage unit of pages, the same buffer pools, and the same backup and recovery mechanisms. For that reason, BLU Acceleration tables can coexist with traditional row tables in the same schema, storage, and memory. You can query a mix of traditional row and BLU Acceleration tables at the same time. Converting a row-based table to a BLU Acceleration table can be easy.

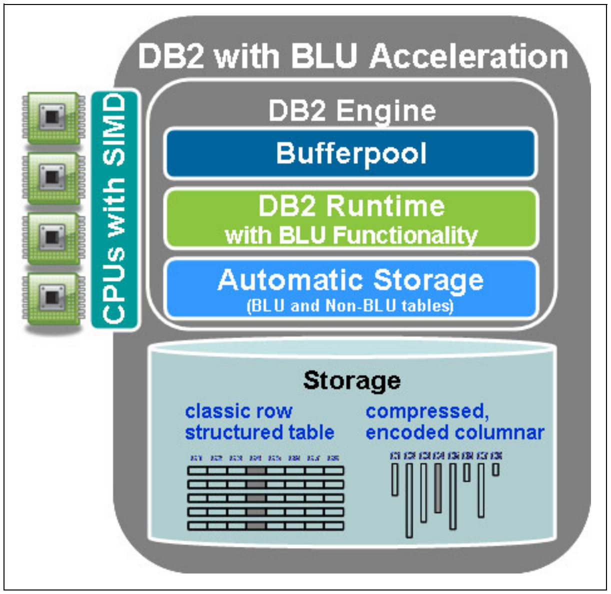



*Figure 1-1 DB2 10.5 engine with BLU Acceleration*

## 1.2 BLU Acceleration: Seven Big Ideas

DB2 with BLU Acceleration includes several features that work together to make it a significant advancement in technology. We refer to these as the *BLU Acceleration Seven Big Ideas*.

### 1.2.1 Big Idea 1: Simplicity and ease of use

Simplicity and ease of use is one important design principle of BLU Acceleration. As a result, much autonomics and intelligence is built into the DB2 engine to optimize the current hardware for analytic workloads. Users do not necessarily have to purchase new hardware for a BLU Acceleration deployment. In addition, the goal is to minimize the complexity for deployment and further database maintenance tasks for DBAs.

From a deployment perspective, many manual tuning efforts are minimized. It is replaced with a single registry variable `DB2_WORKLOAD=ANALYTICS` that automates initial tuning of an analytics database environment. With this registry variable set, database manager and database parameters are automatically tuned by DB2 engine to optimize analytic-type workloads. For example, database memory parameters and intraquery parallelism are set to optimize the current system resources available. Workload management is also enabled automatically to handle different types of concurrent running queries more efficiently.

With the database environment set and optimized, BLU Acceleration is simple to implement and easy to use. BLU Acceleration tables coexist with traditional row-organized tables in the same database, the same table spaces, and the same buffer pools. All you have to do is create your BLU Accelerated tables, load the data into tables, and run your queries. Because of the nature of BLU Acceleration processing designed to speed analytic workloads, there is no need to create user-defined indexes, partition tables, or aggregates, define multidimensional clustering tables, build materialized tables or views, create statistical views, provide optimizer hints, and so on. Database administrators (DBAs) no longer need to spend effort in creating these objects to boost query performance. BLU Acceleration does it naturally for us.

From an ongoing maintenance perspective, without the auxiliary database objects, manual administrative and maintenance efforts from DBAs are significantly reduced. Aside from that, many administrative tasks are also automatic. For example, REORG is done automatically to reclaim space without intervention by DBAs. Of course, BLU Acceleration is built into DB2 overall, all of the database tools that DB2 DBAs are familiar with remain the same. DBAs do not have to spend time learning new skills to use BLU Acceleration.

### 1.2.2 Big Idea 2: Column store

BLU Acceleration is a dynamic in-memory columnar technology. Each column is physically stored in a separate set of data pages.

A column-organized approach offers many benefits to analytic workloads. Most analytic workloads tend to do more table joins, grouping, aggregates, and so on. They often access only a subset of columns in a query. By storing data in column-organized tables, DB2 pulls only the column data you want into memory rather than the entire row. As a result, more column data values can be packed into processor cache and memory, significantly reducing I/O from disks. Therefore, it achieves better query performance.

Another benefit is that column-organized tables typically compress much more effectively. BLU Acceleration compresses data per column. The probability of finding repeating patterns on a page is high when the data on the page is from the same column.

By being able to store both row and column-organized tables in the same database, users can use BLU Acceleration even in database environments where mixed online transaction processing (OLTP) and online analytical processing (OLAP) workloads are required. Again, BLU Acceleration is built into the DB2 engine; the SQL, optimizer, utilities, and other components are fully aware of both row- and column-organized tables at the same time.

### 1.2.3 Big Idea 3: Adaptive compression

One of the key aspects of DB2 with BLU Acceleration is how the data is encoded on disk. This is important not only because it results in significant compression, but also because DB2 is using a different encoding scheme that allows data to be scanned or compared while it remains encoded. The ability to scan with compressed data directly allows DB2 to delay materialization as long as possible and do as much work without decoding the data. The result is savings in processing effort and I/O throughput. The result is that DB2 does as much as it can without decompression.

We first explain how compression works in BLU Acceleration. First, because BLU Acceleration stores data using column-organization, each column is compressed with its own compression dictionaries. When BLU Acceleration compresses data, it uses a form of Huffman encoding, with the traditional adaptive mechanisms. The key principle is that data is compressed based on frequency of values in a column. That is, something that appears many times should be compressed more than other values that do not appear as often. Because data values are likely similar in pattern, DB2 achieves a favorable compression rate by compressing values in column-level rather than in table-wide-level.

For instance, taking a US state column as an example, because of the population of the states, the *California* state might appear more frequently than the *Vermont* state. So DB2 can encode *California* with a shorter, single bit (1). The result is good compression for this state that appears often in a column. At the same time, for *Vermont* state that might not appear as often in the same column, DB2 can encode it with a longer eight bits (11011001). The result is that those values, appearing more often in a column, obtain higher level of compression. Aside from approximate Huffman encoding, DB2 also combines this with prefix encoding and offset coding to give a more complete compression in different scenarios.

In addition to column-level compression, BLU Acceleration also uses page-level compression when appropriate. This helps further compress data based on the local clustering of values on individual data pages.

Because BLU Acceleration is able to handle query predicates without decoding the values, more data can be packed in processor cache and buffer pools. This results in less disk I/O, better use of memory, and more effective use of the processor. Therefore, query performance is better. Storage use is also reduced.

### 1.2.4 Big Idea 4: Parallel vector processing

Single-instruction, multiple-data (SIMD) is a common technology used in modern day processors. DB2 with BLU Acceleration has special algorithms to take advantage of the built-in parallelism in the processors automatically, if SIMD-enabled hardware is available. This is another big idea in BLU Acceleration; it includes being able to use special hardware instructions that work on multiple data elements with a single instruction.

For instance, assume that a query requires a predicate comparison sales order shipped in December 2013, as illustrated in Figure 1-2 on page 6. Without SIMD, the processor must process data elements one at a time. Data elements can be loaded into its own processor register one at a time, for example from September, October, November, and so on. As a result, multiple iterations are required to process a single instruction. With BLU Acceleration using SIMD parallelism on a processor and the ability to process predicates encoded from efficient columnar compression, multiple-encoded data values can be packed into the processor register at the same time. It is possible to take all compressed September, October, November, and December data values for a single comparison concurrently if they are in the same processor register. Therefore, the predicate processing is much faster.

> **Note:** DB2 uses the power of the SIMD-enabled processors to do arithmetic operations, and can also do scans, joins, and grouping operations.

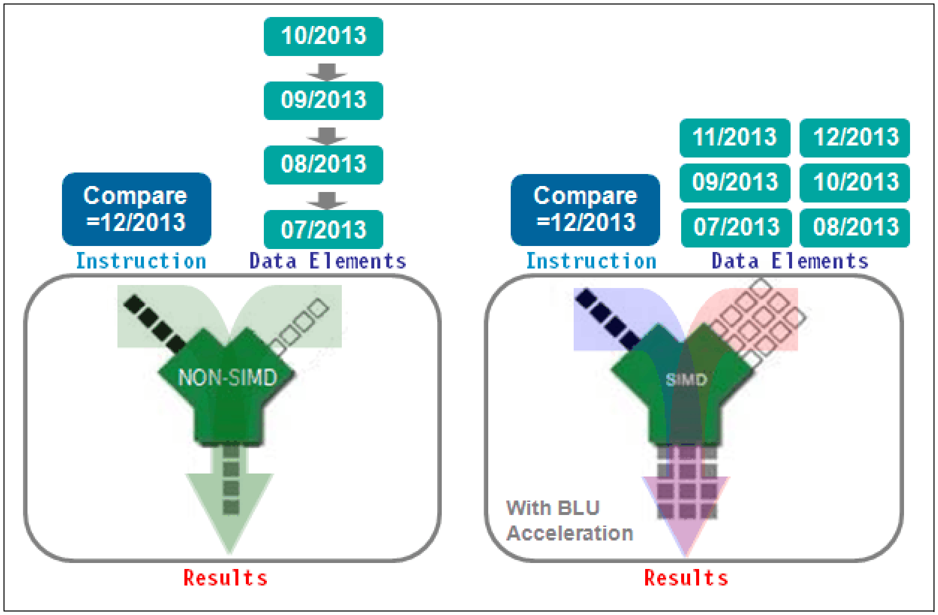



*Figure 1-2 Efficient use of SIMD processor packing in BLU Acceleration*

One of the many benefits of BLU Acceleration is cost-efficiency. BLU Acceleration does not require users to acquire new hardware. If a SIMD-enabled processor is not available in current hardware, BLU Acceleration still functions and speeds query processing using other big ideas. In some cases, BLU Acceleration even emulates SIMD parallelism with its advanced algorithms to deliver similar SIMD results.

## 1.2.5 Big Idea 5: Core-friendly parallelism

BLU Acceleration is a *dynamic in-memory* technology. It makes efficient use of the number of processor cores in the current system, allowing queries to be processed using multi-core parallelism and scale across processor sockets. The idea is to maximize the processing from processor caches, and minimize the latencies having to read from memory and, last, from disk.

Much effort was invested in comprehensive algorithms that do what is called *core-friendly* parallelism. The algorithms are designed so that they delicately place and align data that are likely to be revisited into the processor cache lines. This maximizes the hit rate to the processor cache, increasing effectiveness of cache lines.

An example is shown in Figure 1-3. When a query is run, first, BLU Acceleration uses a separate agent per available processor core to fetch individual column data values. Each agent can then work on different query functions. In the illustration, each processor core 0, 1, 2, 3 works on fetching data for different columns being queried. This is only one level of parallelism.

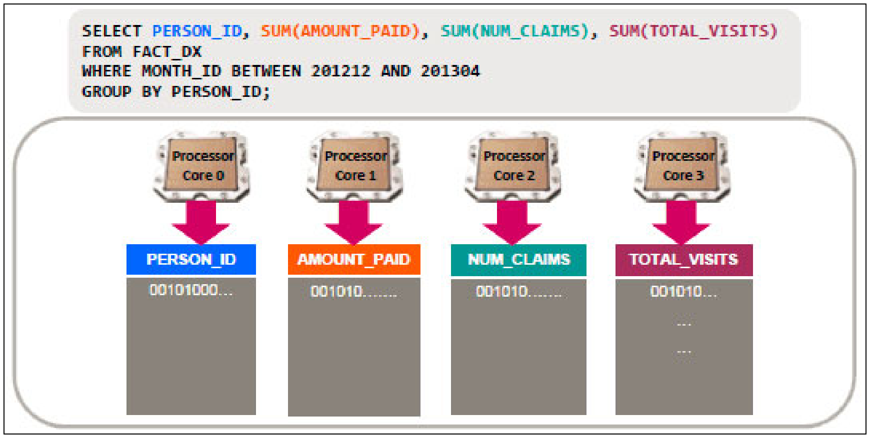



*Figure 1-3 Multiple cores are used in parallel on multiple columns with BLU Acceleration*

Figure 1-4 illustrates another level of parallelism. BLU Acceleration has intelligent algorithms that effectively place and align data onto processor caches. This helps reduce the number of cache misses and maximize processor cache hit rates. With data being available for reading from the processor cache without reading from the main memory or disks, query performance is improved significantly.

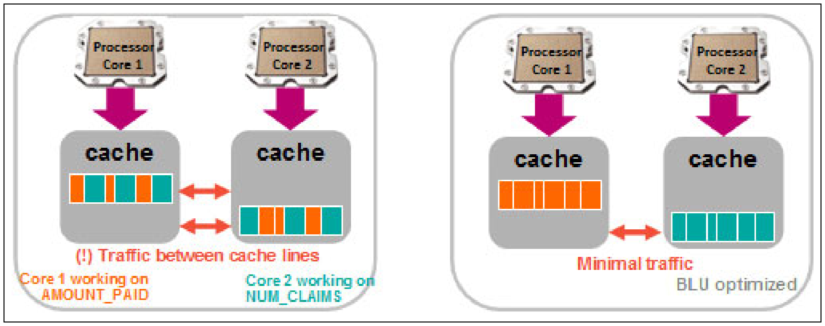



*Figure 1-4 BLU Acceleration optimizes data alignment in cache lines*

With the advanced technologies that IBM research and development groups have put together in BLU Acceleration, BLU Acceleration delivers a whole new level of query performance. Also worth mentioning is that BLU Acceleration does all the parallelism without DBAs intervention or maintenance.

## 1.2.6 Big Idea 6: Scan-friendly memory caching

BLU Acceleration is designed for the demands of a big data world where it maybe less likely that all of the data that queries need can fit into memory. As we mentioned before, the product is cost-efficient. Users are not required to purchase the latest or fastest hardware to run BLU Acceleration. Although DB2 would make good use of having as much hot (active) data in memory as possible, it is not a requirement to fit everything in memory.

BLU Acceleration includes a set of big data-aware algorithms, for cleaning out memory pools, that is more advanced than the typical least recently used (LRU) algorithms that are associated with traditional row-organized database technologies. These BLU Acceleration algorithms are designed from the ground up to detect “interested” data patterns that are likely to be revisited, and to hold those pages in the buffer pool as long as possible. These algorithms work with the DB2 traditional row-based algorithms. Along with the effective columnar compression, even more hot data can fit into the buffer pool. Clients typically have a memory-to-disk ratio of 15% to 50%. What this means is that they cannot possibly fit their entire table, or even the data that is required to run a complex query, entirely into memory. With BLU Acceleration, at least 70% - 80% of active data can likely fit into memory. The net result is that you can get significant performance boosts from data that is already in memory, although memory on your database server might not be as big as your tables.

## 1.2.7 Big Idea 7: Data skipping

To speed query processing, BLU Acceleration can skip ranges of data that are not relevant to the current query, reducing unnecessary I/O. For instance, if a report will be looking for the sales orders from December 2013, scanning through sales orders that were made in October, November, and so on, is not necessary. BLU Acceleration goes straight to the ranges of data that contain sales orders from December 2013, and computes the rest of the query.

This is how BLU Acceleration does *data skipping*. As data is loaded into column-organized tables, BLU Acceleration tracks the minimum and maximum values on ranges of rows in metadata objects that are called the *synopsis tables*. These metadata objects (or synopsis tables) are dynamically managed and updated by the DB2 engine without intervention from the DBA. When a query is run, BLU Acceleration looks up the synopsis tables for ranges of data that contain the value that matches the query. It effectively avoids the blocks of data values that do not satisfy the query, and skips straight to the portions of data that matches the query. The net effect is that only necessary data is read or loaded into system memory, which in turn provides dramatic increase in speed of query execution because much of the unnecessary scanning is avoided.

### 1.2.8 The seven big ideas in action

You might have noticed that from all of the big ideas, BLU Acceleration is designed to be simple, and easy to use. First, the effort required to deploy and maintain a BLU Acceleration environment is minimal. Second, advanced technologies such as columnar compression, parallel vector processing, core-friendly parallelism, scan-friendly memory caching, and data skipping are all used by DB2 automatically without DBAs having to explicitly deploy extra auxiliary structures for it to work. It is in the DB2 engine's nature to process queries using these technologies.

With our client base who are using BLU Acceleration, we found that clients have experienced an average, conservatively, of approximately 10 times compression on their analytics databases, without any complex tuning. In terms of performance, we observed that their queries ran 35 - 73 times faster on average (and some are even faster).

To help you understand how DB2 clients are observed to have significant performance and storage improvements using BLU Acceleration, we go through an example there. This example can help you understand how BLU Acceleration works under the layers when running a query on a large amount of data.

Assume that we have an analytics environment with the following attributes:

- Raw data is of 12 TB data in size.
- Database server has SIMD-enabled processor, 30 cores, and 240 GB of memory.
- Database stores two years of data (2012 - 2013).
- Sales order table has 100 columns.

When a query is run to look for sales orders that are shipped in December 2013, the following actions occur:

- *Query*:

  ```
  SELECT SUM(ORDER_TOTAL) FROM SALES_FACT WHERE MONTH='201312'
  ```
- *Columnar compression*: From raw data of 12 TB data in size, conservatively we assume five times the compression rate. (This assumption is conservative because BLU Acceleration is often observed to have average of about 10x.) This lowers data size to 2.4 TB.
- *Column store*: Query is accessing the `ORDER_TOTAL` and `MONTH` columns of a total of 100 columns. Access to column-organized tables decreases the size of interested data to 2/100 columns = 1/50 of compressed data (which means 48 GB).

- *Data skipping*: With use of synopsis tables, BLU Acceleration skips to the data that matches the predicate without decoding or any evaluation processing. Consider query is accessing `ORDER_TOTAL` that is in December 2013. This is one month out of total of two years (1/24) of records in the entire database. This brings down the interested data to 48 GB / 24 = 2 GB.
- *Core-friendly parallelism*: Data is processed in parallel across 30 available processor cores. That is, 2 GB / 30 = 68 MB.
- *Parallel vector processing*: Data can be scanned using a SIMD-enabled processor to achieve faster performance. The improvement depends on various processors. With a conservative assumption that this SIMD-enabled processor obtains 4 times faster per byte, DB2 only has to scan approximately 68 MB / 4 =~ 17 MB of data.

To summarize this example, DB2 must process only about 17 MB of data at the end. This is roughly the size of a few high-quality digital pictures or cell phone apps. Consider a modern processor that processes 17 MB of data; results can be returned almost instantly. This example demonstrates how BLU Acceleration can easily achieve significant performance and storage improvements for analytic workloads.

## 1.3 Next generation analytics: Cognos BI and DB2 with BLU Acceleration

In the first few chapter of this book, we use Cognos Business Intelligence (BI) as an example to demonstrate deployment, testing, and verification of BLU Acceleration in analytic workloads from Cognos BI. First, we give a brief introduction of how Cognos BI and DB2 with BLU Acceleration work together.

All DB2 editions that support BLU Acceleration include five authorized user license entitlements to Cognos BI 10.2. The following BLU Acceleration editions are supported in DB2 10.5 FP3:

- DB2 Advanced Workgroup Server Edition (AWSE)
- Advanced Enterprise Server Edition (AESE)
- Developer Edition (DE)

Cognos Business Intelligence is a proven scalable enterprise BI platform that provides customers with the power of analytics and helps drive better business decisions. Cognos Business Intelligence 10.2.1 includes Dynamic Cubes technology that helps boost query performance by making efficient use of memory. With Cognos Dynamic Cubes, aggregates, expressions, results, and data can all be stored in memory. Ad-hoc queries or detailed drill-down reports can have instant responses across terabytes of data. Besides Dynamic Cubes, Cognos BI 10.2 also achieves faster load time in Active Report through improved use of compression and engine optimization. There are also other new extensible, customizable visualization features that help enhance the BI user reports experience.

### Enhanced performance from database layer

Although Cognos BI provides various techniques such as Dynamic Cubes to optimize analytics on a large amount of data, completely bypassing access to the underlying database is almost unavoidable. For instance, sometimes data might not be found in the Dynamic Cubes so the system then needs to access the required data from the relational database. DB2 with BLU Acceleration complements the optimizations in Cognos with dynamic in-memory column store technologies on the database layer. It is optimized for analytic workloads, providing an efficient data platform for accessing large amount of data for specific columns when needed.

Different from other fully in-memory database technologies in the market, DB2 with BLU Acceleration does not require all data to be stored in memory. You can continue to use your existing hardware with BLU Acceleration, making this a flexible, cost-effective, and yet efficient data platform solution for Cognos users.

In performance tests with Cognos BI and DB2 with BLU Acceleration running together, Cognos cube loading is tested to achieve faster performance through the improved data compression, hardware instructions, and column-data processing in BLU Acceleration technologies. Cognos BI modelers can now use less time to load and refresh data cubes; users can run more conditional queries in much less time. Furthermore, Cognos report workload performance is also tested to achieve faster performance. Users can now experience click-through of their reports much faster.

### Simplicity of the combined solutions

The combination of Cognos BI and DB2 with BLU Acceleration provides enhanced performance improvement for analytic workloads and simplicity in usage and maintenance.

First, Cognos engine recognizes BLU Acceleration column-organized tables as regular DB2 tables. For Cognos BI developers or users, existing Cognos reports or definitions do not require being changed when converting existing row-organized data into BLU Acceleration column-organized tables.

Second, BLU Acceleration is deployed with even more autonomics. For DBAs, the needs for creating and managing indexes or materialized query tables (MQTs), aggregates, partitioning, tuning, SQL changes and schema changes are gone and replaced with the DB2 engine doing all the hard work. In sum, the Cognos BI and DB2 with BLU Acceleration combined solution provides speed-of-thought performance for reports, and reduces storage costs with enhanced data compression from both solutions and fewer database objects required from BLU Acceleration. The combined solution is thoroughly tested, optimized together to achieve most optimal analytics for existing or new Cognos BI users.

## 1.4 IBM DB2 with BLU Acceleration offerings

IBM DB2 with BLU Acceleration includes several offerings to address various client needs. As with the key design principle of DB2 with BLU Acceleration, these offerings further simplify deployment by providing an integrated hardware and software platform for an enterprise business intelligence environment. The following offerings are described in this section:

- IBM Business Intelligence with BLU Acceleration Pattern
- IBM BLU Acceleration for Cloud
- IBM BLU Acceleration Solution – IBM Power Systems™ Edition

### 1.4.1 Simplified IBM Business Intelligence with BLU Acceleration Pattern deployment on IBM PureApplication Systems

As data volumes grow exponentially and demands on analytics increase, having a ready-to-use optimized BI analytics environment built in a short amount of time is beneficial. *IBM Business Intelligence with DB2 BLU Acceleration pattern* is for businesses that require fast access to an available enterprise analytics solution. It is an automation deployment package that speeds deployment of DB2 with BLU Acceleration and IBM Cognos Business Intelligence with Dynamic Cubes. The pattern comes together with expertise deployment preferred practices. It can be deployed on pattern-enabled hardware, such as the IBM PureApplication® system, with a complete functioning BI environment in as short amount of time as 1 hour. After the system is deployed, users can start loading data for analytics, modeling, and start deploying their projects immediately.

The IBM PureApplication System on which the Business Intelligence with BLU Acceleration Pattern can be deployed, builds an open and scalable cloud application platform for web and database applications. A PureApplication System is preassembled and includes predefined patterns to simplify and automate IT lifecycle tasks of applications. Patterns can deploy application architectures, including platform services, and manage infrastructures to help customers and partners reduce time and cost, adjust to rapid changes, and identify problems faster. The PureApplication System is available in scalable configurations ranging from small development and testing to large enterprise environments that can be managed and updated through a single console.

With the Business Intelligence Pattern with BLU Acceleration, users do not have to build hardware and software systems from scratch. The result is a short deployment, which provides a virtual application instance on the PureApplication System that includes the following components:

- IBM Cognos Business Intelligence: A web-based business intelligence solution with integrated reporting, analysis, and event management features.
- IBM WebSphere® Application Server: A web application server for hosting Java-based web applications.
- IBM Tivoli® Directory Integrator: Synchronizes data across multiple repositories.
- Cognos content store database on IBM DB2: Stores Cognos metadata such as application data, including security, configuration data, models, metrics, report specifications, and report output in a DB2 database.
- IBM DB2 with BLU Acceleration: A database used for the BI Pattern.

For more information about IBM Cognos Business Intelligence, consult the following website that includes the *Business Intelligence Pattern with BLU Acceleration Installation and Administration Guide*:

http://www.ibm.com/support/docview.wss?uid=swg27039326#cbiimav1r0m0en

Also, you can find an entry point into the IBM PureApplication System at this location:

http://www.ibm.com/ibm/puresystems/us/en/pf_pureapplication.html

## 1.4.2 IBM BLU Acceleration for Cloud

With cloud computing being a popular trend, DB2 with BLU Acceleration is also available in a software service on-the-cloud offering. This allows companies to launch their new analytics projects or prototypes on the cloud platform without having to add new IT infrastructure requirements and capitals. The software stack includes DB2 with BLU Acceleration and Cognos Business Intelligence, a combined optimized data warehousing and analytics solution for data analytics. Companies can choose a preferred cloud provider (Softlayer or AWS), or have IBM managed services to help you move to a cloud environment.

For more information, visit the following website:

http://bluforcloud.com

### 1.4.3 IBM BLU Acceleration Solution – Power Systems Edition

*IBM BLU Acceleration Solution – Power Systems Edition* includes IBM AIX®, IBM PowerVM®, and DB2 with BLU Acceleration on a Power ESE server. This combined solution is designed to take advantage of the multiple cores and simultaneous multi-threading capability in IBM POWER7+™ processors and provides fast performance data warehouse infrastructure for analytic workloads.

For more information, see the following web page:

http://www.ibm.com/systems/power/solutions/analytics/editions/db2-blu.html

## 1.5 Obtaining DB2 with BLU Acceleration

Several methods are available for trying DB2 with BLU Acceleration:

- *IBM DB2 BLU Acceleration Kit for Trial*: Install a fresh operating system and DB2 software stack together with minimal system setup hassle.
- *IBM DB2 with BLU Acceleration for Cloud Beta:* Access DB2 with BLU Acceleration on the cloud with your Internet browser for up to 5 hours.
- *Download DB2 with BLU Acceleration Trial*: Download 90-day DB2 trial software and install it on your own systems.
- *Attend an IBM DB2 BLU Bootcamp or IBM Education:* Attend bootcamps and education classes with presentations and hands-on lab exercises to learn and try everything about BLU Acceleration.

### 1.5.1 IBM DB2 BLU Acceleration Kit for Trial

The IBM DB2 BLU Acceleration Kit for Trial is ideal for users who want to try DB2 with BLU Acceleration on their own existing hardware with minimal system setup hassle. Essentially, it is an installable `.iso` file or raw USB image that installs a full DB2 and operating system software stack on any x86_64 physical and virtual machine. The kit includes automated scripts and simple installation wizards to quickly get a running DB2 trial system including the following components:

- SUSE Linux Enterprise Server operating system
- IBM DB2 10.5 with BLU Acceleration (90 days trial)
- IBM Data Studio
- IBM Database Conversion Workbench

> **Attention:** This installation kit wipes out any existing operating system and data on an existing system.

For more information and to download the kit, visit Information Management Solution Portal at the following website:

http://ibm.co/15zDz3C

### 1.5.2 IBM BLU Acceleration for Cloud trial option

For users who want to experience DB2 with BLU Acceleration without any setup, the BLU Acceleration for Cloud trial plan provides a fully functional analytics environment for up to 5 hours at no charge. For more information, see the IBM BLU Acceleration for Cloud website:

http://www.ibmbluhub.com/get-blu/blu-for-cloud/

### 1.5.3 DB2 with BLU Acceleration trial software

To install DB2 with BLU Acceleration trial software on your own platforms, download the 90-day no-charge trial software from the Get DB2 with BLU Acceleration website:

http://www.ibmbluhub.com/get-blu/

### 1.5.4 IBM DB2 BLU bootcamps and education

Extensive training materials are available for DB2 with BLU Acceleration, including bootcamps and IBM education offerings. For more information, contact your IBM Software Group representative for details.

# 2

# Planning and deployment of BLU Acceleration

New technology sometimes includes the burden of complex deployments. DB2 with BLU Acceleration, however, can be easily adopted and is practically “load and go.” This chapter shows how fast and easy it is to get DB2 activated with BLU Acceleration.

Although the examples in this chapter are based on the Cognos “Sample Outdoors Warehouse model,” the commands shown apply to any BLU Accelerated environment.

The following topics are covered:

- BLU Acceleration deployment made easy
- Data environments targeted for analytic workloads
- Data environments with mixed workloads
- Prerequisites
- Deployment
- Configuration preferred practices for BLU Acceleration deployment
- Configuration preferred practices for HADR

## 2.1 BLU Acceleration deployment made easy

The term BLU Acceleration represents several IBM technologies that are deeply integrated with DB2 for Linux, UNIX, and Windows Version 10.5. IBM did not just bolt on a new component. Instead, everything is already included in the DB2 engine so that DBAs or developers do not have to learn a new set of skills. For this reason, setting up, configuring, and using can be easy.

Simply stated, you only need to install DB2 10.5 on the currently supported platforms, either AIX on IBM POWER® or Linux on x86 64-bit, and enable a single configuration parameter to tell DB2 that you will optimize defaults for analytical workload characteristics. Next, you create your database and tables, and load data using common DB2 tools. Such a workload can then immediately benefit from the significant performance boost without the need for SQL syntax or schema changes.

At the center of BLU Acceleration is the column-organized table store. It is combined with *actionable compression* that operates on a column and page level to save on storage space. The column organization eliminates the need for creating and maintaining secondary indexes and aggregates. In DB2 10.5, both column-organized and traditional row-organized tables can coexist in the same database. For optimal performance, run analytical queries against tables that are all column-organized.

Other new IBM technologies such as data skipping, parallel vector processing, core-friendly parallelism, and scan-friendly memory caching work in sync to make efficient use of hardware resources and to deliver these compelling advances.

## 2.2 Data environments targeted for analytic workloads

Users with typical data warehouses, data marts, or databases that are dedicated for analytic workloads can benefit most from loading and converting of the entire database schemas that support analytic workloads into BLU Accelerated column-organized tables. These workloads can involve regular analytic reporting queries and business intelligence ad hoc queries. For example, workloads that carry the following characteristics benefit most from BLU Acceleration:

- Analytical, data mart workloads
- Queries that involve grouping, aggregation, range scans, and joins
- Queries that access only subset of columns in a table
- Star or dimensional schemas

- SAP Business Warehouse application workloads (for more information, see Chapter 8, “DB2 with BLU Acceleration and SAP integration” on page 245)

In past client experiences, for databases with purely analytic workloads, immediate query performance and storage improvements are seen after converting the entire database schemas to use BLU Acceleration.

To provide great flexibility, DB2 with BLU Acceleration supports both row-organized and column-organized tables in the same database, schema table spaces, buffer pools, and queries. Some users might choose to load or convert only certain tables to column-organized tables for their own reasons. When tables are selected to convert, we suggest that all tables accessed in join queries be converted to column-organized tables for optimal performance. Otherwise, additional internal casting operations are needed to join row-organized tables and column-organized tables together and can consume unnecessary resources.

> **Tip:** Put as many tables as possible into column-organized format for environment with a purely analytics or OLAP workload. This includes typical data marts or data warehouse workloads with extensive range scans, grouping, joins, or aggregation.

## 2.3 Data environments with mixed workloads

DB2 with BLU Acceleration introduces a new, innovative, way to handle mixed workloads. The new shadow table feature takes advantage of BLU Acceleration technologies to eliminate the overhead involved in the traditional handling of mixed workloads, making processing faster, simpler and easier. Instead of maintaining separate systems for transactional and analytic processing, BLU Acceleration Cancun release allows efficient processing for both OLTP and OLAP workloads on the same data. Chapter 3, “Planning and deployment of BLU Acceleration shadow tables for mixed workload environments” on page 53 discusses the deployment and use of shadow tables in greater detail.

For users intending to convert existing tables to facilitate their analytic processing needs in a mixed workload environment, it is suggested to choose the use of BLU Acceleration only on those tables that are purely used for analytics. The **db2convert** utility converts row-organized tables to column-organized tables, while source tables remain accessible online.

Another database tool, IBM InfoSphere Optim™ Query Workload Tuner for DB2 for Linux, UNIX, and Windows (version 4.1 or later), is extended with an advisor called the Workload Table Organization Advisor (WTOA). With WTOA, users can run analysis of their existing workloads, and get recommendations for tables that are expected to benefit from BLU Accelerated column-organized table storage. The tool virtually tests the conversion of existing row-organized tables to column-organized tables, and provides estimated performance improvement statistics of your workloads to help with easier BLU Acceleration deployment decisions.

We describe more about using Optim Query Workload Tuner and WTOA with DB2 in Chapter 4, "Optim Query Workload Tuner and BLU Acceleration" on page 137.

## 2.4 Prerequisites

Review the requirements and recommendations when you get ready to deploy DB2 with BLU Acceleration.

### 2.4.1 DB2 system requirements

The BLU Acceleration feature in DB2 10.5 is supported on the AIX and Linux x86 (Intel and AMD) platform using 64-bit hardware. It shares the same DB2 10.5 minimum operating system requirements.

For the most recent information about *DB2 system requirements,* consult the general documentation at the following location:

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

#### Hardware for DB2 with BLU Acceleration

As of this writing, we suggest the hardware and operating systems for DB2 with BLU Acceleration as shown in Table 2-1. The processors supported in DB2 10.5 are also supported for BLU Acceleration. The processors listed in the table include specific hardware optimizations that BLU Acceleration can benefit from.

*Table 2-1 Hardware and platforms suggested for DB2 with BLU Acceleration*

| Operating system | Minimum version requirement | Version to use | Hardware suggested |
|---|---|---|---|
| AIX | ► AIX 6.1 TL7 SP6<br>► AIX 7.1TL1 SP6 | ► AIX 7.1 TL2 SP1 or later | IBM POWER7® or later |

| Operating system | Minimum version requirement | Version to use | Hardware suggested |
|---|---|---|---|
| Linux x86 64-bit | ► Red Hat Enterprise Linux (RHEL) 6<br>► SUSE Linux Enterprise Server (SLES) 10 SP4<br>► SLES 11 SP2 | ► RHEL 6.3 or later<br>► SLES 11 SP2 or later | Intel Nehalem (or equivalent) or later |

## 2.4.2 DB2 license requirements and functionality

DB2 for Linux, UNIX, and Windows Version 10.5 is available in multiple product editions, each including a different number of features and providing functionality that we describe in this section.

In terms of required license entitlements, the BLU Acceleration feature includes the following DB2 10.5 editions for production environments:

- Advanced Enterprise Server Edition (AESE)
- Advanced Workgroup Server Edition (AWSE)

Non-production environments can use the following DB2 10.5 edition that also entitles the use of BLU Acceleration:

- Developer Edition (DE)

### DB2 Advanced Enterprise Server Edition (AESE)

This is the premier DB2 product edition with the largest set of features. It can be characterized as follows:

- Suitable for transactional, warehouse, and mixed workloads
- No processor, memory, or database size limits
- Provides all features from DB2 *Enterprise Server Edition* plus column-organized tables, in-memory database, data compression, workload management, replication, and distributed partitioning capability
- Available on either a Processor Value Unit or per Authorized User Single Install pricing model

For more information, see the following web page:

http://www-03.ibm.com/software/products/en/db2advaenteservedit

### DB2 Advanced Workgroup Server Edition (AWSE)

This DB2 edition is similar to the *DB2 Advanced Enterprise Server Edition*, but applies resource limits:

- Data server of choice for deployment in a departmental, workgroup, or medium-sized business environment
- Limits on processor (16 cores), memory (64 GB), and database size (15 TB)
- Available on either a Processor Value Unit or per Authorized User Single Install pricing model

For more information, see the following web page:

http://www-03.ibm.com/software/products/en/db2-advanced-workgroup-server-edition

### DB2 Developer Edition (DE)

This edition is suitable for a single application developer for the purpose of designing and implementing applications for deployment on any of the IBM Information Management client or server platforms. It is characterized as follows:

- For non-production development use only
- Includes all the DB2 server editions, IBM DB2 Connect™ Enterprise Edition
- Requires a user license for each Authorized User of this product

For more information, see the following web page:

http://www-03.ibm.com/software/products/en/db2-developer-edition/

### Other DB2 product editions

For other DB2 product editions with descriptions of functionality and features, see the following web page:

http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.licensing.doc/doc/c0058536.html

## 2.4.3 Capacity planning

DB2 with BLU Acceleration is optimized for parallel vector processing using single-instruction, multiple-data (SIMD) architecture while exploiting physical attributes of multi-core processors, and to efficiently use large memory configurations.

For production environments, the general suggestion is to use a minimum of eight processor cores with 8 GB of main memory per core. For high concurrency environments, increase the main memory to16 GB per core. We suggest maintaining the main memory-to-processor core ratio when adding more processors as demanded by a growing workload with a higher query complexity and with increasing data volumes.

An essential measurement in sizing the memory that BLU Acceleration uses for a workload is the amount of raw active data. This value does not describe the total database size, but does describe the amount of required data frequently needed for processing active queries.

Although DB2 with BLU Acceleration does not require all active data to fit in memory, it can process queries faster with more data to keep in memory while not having to wait for synchronous, physical reads from the storage system. Based on the estimated raw active data size combined with the levels of concurrency and query complexity, a sizing decision can be made regarding how much memory and how many processor cores to accommodate for.

You can approach the estimation of the raw active data size by two steps:

1. Calculate the uncompressed table sizes of frequently accessed tables.
2. Narrow the number from step 1 to the percentage of rows and columns actually accessed.

For example, if you have a total of four years of sales data in the analytics database, and your reports usually query the last year of data, then frequently accessed table data is 1/4 (25%) of the total database size. At the same time, not all columns might be accessed by the queries. By using column-organized tables, BLU Acceleration does not have to read or look at columns that are not in the queries. Both the accessed column and row data gives an approximate percentage of the active table data, the raw active data size.

A first rough estimate for this indicator can be taken from the raw size of your comma-separated value (CSV) load files. If the data is already loaded, you can use the sizes of the used table spaces as a starting point. Alternatively, you can list the table sizes using the SQL query as shown in Example 2-1. Note that the query returns most accurate results with updated table statistics.

*Example 2-1 Raw data size calculation of existing DB2 tables*

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

```
a.tabname = b.tabname AND
b.datapartitionid = 0 AND
b.tbspaceid = c.tbspaceid
```

More information about sizing system resources is in *Best practices: Optimizing analytic workloads using DB2 10.5 with BLU Acceleration*, at this location:

http://bit.ly/1s5HCX5

### 2.4.4 Storage requirements

There is no special storage requirement for DB2 with BLU Acceleration. A general suggestion is to store column-organized tables on storage systems with good random read and write I/O performance. This notably helps in situations where active table data exceeds the main memory available to DB2 or when temporary tables are populated during query processing.

In these situations, the following storage types can be of benefit:

- Solid-state drives (SSDs)
- Enterprise SAN storage systems
- Flash-based storage with write cache

The following IBM storage systems exhibit good random I/O characteristics:

- IBM FlashSystem™ 810, 820, and 840
- IBM Storewize V7000 (loaded with SSDs)
- IBM XIV® storage system
- IBM DS8000® Series

## 2.5 Deployment

This section describes and demonstrates the activation of DB2 with BLU Acceleration in new or existing DB2 databases. We assume that an installation of a standard DB2 product with required prerequisites exists and an instance in a current or previous version already exists.

### 2.5.1 Single tuning parameter for analytical workloads

To enable BLU Acceleration, a single registry variable, `DB2_WORKLOAD=ANALYTICS`, is required. Setting this registry variable to `ANALYTICS` informs DB2 that the database is used for analytic workloads. It automatically configures relevant configuration parameters to optimize existing hardware capacity and performance of analytic workloads. For example, when recommended settings are auto-configured on a database, intraquery parallelism is enabled (to use multi-core parallelism) and the default table organization (`DFT_TABLE_ORG`) is set to `COLUMN`. Utility heap is set to `AUTOMATIC`. Other relevant sort parameters are set to a higher value to optimize the existing hardware. Automated workload management is BLU-aware, optimizing server utilization for analytics concurrency. Last but not least, automatic space reclamation is enabled.

For details of the database behavior that is affected by `DB2_WORKLOAD=ANALYTICS`, see 2.6.1, "Changes applied with DB2_WORKLOAD=ANALYTICS" on page 31. This single-touch automatic database tuning saves effort from manual settings.

For the `DB2_WORKLOAD` registry parameter to become active, the DB2 instance must be restarted after setting `DB2_WORKLOAD=ANALYTICS`. As part of a new database creation process, the configuration advisor automatically applies all required BLU Acceleration settings to optimize analytic workloads. Optionally, this can also be applied explicitly through the `AUTOCONFIGURE` keyword as part of the database creation. For existing databases, analytics-optimized settings can be applied through a separate `AUTOCONFIGURE` statement. More details are explained in 2.5.4, "Existing database deployments" on page 28.

If for any reason you do not want to use the single-touch, instance-level parameter `DB2_WORKLOAD=ANALYTICS` (for example when using multiple databases within the same instance and for different purposes, or when running a mixed workload against the same database with significant transactional queries happening at the same time), see 2.6, "Configuration preferred practices for BLU Acceleration deployment" on page 31 for parameters that you might want to set manually to best suit your environment.

> **Preferred practice:** Create each BLU Accelerated database in its own separate instance.
>
> The configuration change suggested by the `AUTOCONFIGURE` option, which is also called *Configuration Advisor*, accounts for only one database per instance. This means multiple databases should be put into separate instances to be more independent when using auto-configure to tune initial database manager and database configuration parameters.

For more information about this topic, see the "Column-organized tables" information in the IBM Knowledge Center:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.dbobj.doc/doc/c0060592.html?lang=en

## 2.5.2 New database deployments

This section describes the steps to create a new BLU Accelerated database. In a typical scenario for analytical workloads, a DB2 instance holds a single database. We suggest assigning the majority of the instance memory to this database.

If you intend to have a BLU-enabled database and other non-analytics databases within the same instance, see 2.5.3, "Multiple mixed-workload databases in a single instance" on page 27.

When you create a new database, create it with a `UNICODE` code set, `IDENTITY` collation, and `automatic storage management`. These are the default settings and do not need to be specified explicitly.

By default, the configuration advisor is run automatically and assigns up to 25% instance memory for the database when a new database is created. Example 2-2 overwrites this default through the `autoconfigure` keyword with using `mem_percent 80`, which assigns 80% of the instance memory to this single active database in the instance. The `AUTOCONFIGURE` keyword also recommends the enablement of Self Tuning Memory Manager (STMM), allowing various database memory parameters to adjust to workload requirement as needed. At the same time, `AUTOCONFIGURE` applies all tuning parameters to the instance and to the database (`APPLY DB AND DBM`). Then, the new database is optimized for analytic workloads based on available hardware. At this point, you can start creating and loading your tables as listed in this example.

*Example 2-2 Activating BLU Acceleration in a new database*

```
db2set DB2_WORKLOAD=ANALYTICS
db2stop
db2start
db2 CREATE DB GS_DB ON /dbpath/gs_db AUTOCONFIGURE USING mem_percent 80
APPLY DB AND DBM
db2 CONNECT TO GS_DB
db2 CREATE TABLE GOSALESDW.EMP_SUMMARY_FACT (...)
db2 LOAD FROM EMP_SUMMARY_FACT.DEL OF DEL REPLACE INTO
GOSALESDW.EMP_SUMMARY_FACT
```

If you have existing table and object creation DDL statements from an existing data warehouse, you can reuse them without making changes to the table creation statements. With the `DB2_WORKLOAD=ANALYTICS` registry variable set and one of its underlying settings `DFT_TABLE_ORG=COLUMN` in effect, all tables are created as column-organized tables by default.

Having said that, both column and row organization tables can coexist in the same database and even in the same table space. The new `CREATE TABLE` statement includes a new `ORGANIZE BY` keyword. It allows tables to be created with an explicit definition of either `ORGANIZE BY ROW` or `ORGANIZE BY COLUMN`. This easily overrides the value assigned to the `DFT_TABLE_ORG` parameter.

After all the tables are created, you can start loading tables. No new syntax of data loading is required, as demonstrated in Example 2-2.

For more detail about creating column-organized table and loading data, see 2.6.3, “Creating column-organized tables” on page 35 and 2.6.4, “Using insert and ingest with column-organized tables” on page 38.

## 2.5.3 Multiple mixed-workload databases in a single instance

Although the preferred practice is to create a BLU-accelerated database in its own instance, it is possible to create and enable a BLU database in an instance that has multiple databases, if required. This section demonstrates how to create a new database with BLU Acceleration to optimize analytic workloads while keeping other existing or future non-analytics databases in the same instance untouched.

`DB2_WORKLOAD` is an instance-level registry variable. When set to `ANALYTICS`, it enables intra-partition parallelism because BLU Acceleration requires these parallelism settings to process. If more than one database is in the same instance and not all of the databases are intended for analytic workloads, you can follow Example 2-3 to create an analytics database while allowing other databases to use their own separate configuration for memory and intra-partition parallelism.

In this example, we first set the `DB2_WORKLOAD=ANALYTICS` single-touch registry variable and create a new database that optimizes the settings for analytic workloads. The procedure shown in Example 2-3 includes an `ALTER WORKLOAD` statement. It enables the maximum degree of parallelism equivalent to a `DFT_DEGREE` with a value of `ANY` to enable intra-partition parallelism. Then, unsets the `DB2_WORKLOAD` instance-level registry variable and restarts the instance, so that this parameter does not affect other databases in the same instance.

*Example 2-3 Mixing BLU-accelerated and non-analytics databases in a single instance*

```
db2set DB2_WORKLOAD=ANALYTICS
db2stop
db2start
db2 CREATE DB GS_DB ON /dbpath/gs_db AUTOCONFIGURE APPLY DB ONLY
db2 CONNECT TO GS_DB
```

```
db2 ALTER WORKLOAD sysdefaultuserworkload MAXIMUM DEGREE 32767
db2set DB2_WORKLOAD=
db2 TERMINATE
db2stop
db2start
db2 CONNECT TO GS_DB
db2 CREATE TABLE GOSALESDW.EMP_SUMMARY_FACT (...)
db2 LOAD FROM EMP_SUMMARY_FACT.DEL OF DEL REPLACE INTO
GOSALESDW.EMP_SUMMARY_FACT
```

## 2.5.4 Existing database deployments

An existing DB2 10.5 database installed on a single node and using the required settings of `UNICODE` code set and `IDENTITY` collation can be optimized for BLU Acceleration. There is no method for changing the code set and collation of an existing database. If you have an existing database with a different code set or collation, see 2.5.2, "New database deployments" on page 26 for steps to create a new database with the correct code set and collation to take advantage of BLU Acceleration.

For deployment, we suggest using the latest fix pack. At the time of this writing, the latest fix pack is Fix Pack 4 (Cancun release). Check the following link for the latest available fix pack level:

http://www-969.ibm.com/software/reports/compatibility/clarity-reports/report/html/softwareReqsForProductByComponent?deliverableId=254B4BA0C5F011E18183F12B0925FE36&duComponent=Server_8388E5E04AD611E2A6D822020925FE1B

If you are currently using a major release earlier than 10.5, follow the steps in 2.5.5, "Upgrade from a previous release to DB2 10.5" on page 30 to upgrade DB2 to the suggested version.

Example 2-4 shows the commands to activating the BLU Acceleration in an existing database. The first step is to set optimal defaults for analytical workloads using the single-touch `DB2_WORKLOAD` registry variable. Then, restart the DB2 instance. The tuning parameters behind the `DB2_WORKLOAD` registry variable become active only at database creation or when manually applying it using the `autoconfigure` statement. The assumption for this example is an already created database, so we specify the autoconfigure command explicitly.

*Example 2-4 Activating BLU Acceleration in an existing database*

```
db2set DB2_WORKLOAD=ANALYTICS
db2stop
```

```
db2start
db2 connect to GS_DB
db2 alter workload sysdefaultuserworkload maximum degree default
db2 autoconfigure apply db only
```

> **Note:** Use `db2 autoconfigure apply db only` to apply analytics-optimized settings to the current database without affecting other databases in the same instance.

## Converting table spaces to automatic storage management

BLU Acceleration provides simplicity and allows column-organized tables to coexist with row-organized tables in existing table spaces. With computing trends toward autonomics, BLU Acceleration only works with automatic storage table spaces.

If your database is not already using automatic storage management, you can now create a new table space, or convert and rebalance existing table spaces. Example 2-5 also shows the creation of a new table space. Note that the new table space creation statement is using the generally suggested page size and extent size for BLU Acceleration.

*Example 2-5 Creating a new automatic storage table space*

```
db2 CREATE TABLESPACE newtbsp PAGESIZE 32 K EXTENTSIZE 4
```

Example 2-6 shows the conversion of an existing table space to use automatic storage management.

*Example 2-6 Converting table spaces to automatic storage management*

```
db2 CREATE STOGROUP ibmcolstogrp ON '/data1' SET AS DEFAULT
db2 ALTER TABLESPACE oldtbsp MANAGED BY AUTOMATIC STORAGE
db2 ALTER TABLESPACE oldtbsp REBALANCE
```

## Converting a row-organized table to column-organized

Existing tables created traditionally with row organization can be converted using the **`db2convert`** tool shown in Example 2-7. You can find more preferred practices considerations on converting row-organized tables to column-organized tables stored in **`newtbsp`**, the newly created table space, in Example 2-5.

*Example 2-7 Converting a row-organized table to column-organized*

```
db2convert –d GS_DB -z GOSALESDW -t EMP_SUMMARY_FACT -ts newtbsp
```

### Creating and loading new tables

The **LOAD** command syntax and input data source for loading column-organized tables do not differ compared to loading data into row-organized tables. Column-organized data loading takes the same sources in row-organized files, such as delimited files.

Example 2-8 shows how to create and load a new table.

*Example 2-8 Populating new tables in an existing database*

```
db2 "CREATE TABLE GOSALESDW.EMP_SUMMARY_FACT (...)"
db2 LOAD FROM EMP_SUMMARY_FACT.DEL OF DEL REPLACE INTO
GOSALESDW.EMP_SUMMARY_FACT
```

With BLU Acceleration, column-organized tables are always compressed through the use of column-level and page-level compression dictionaries. Both are used transparently to map frequently occurring patterns to shorter symbols. Additionally, less space is typically required compared to row-organized tables as there is no need for indexes and aggregates.

When loading data into column-organized tables using an initial `LOAD` operation the incoming data is scanned and a compression dictionary is built for each column based on a histogram analysis. The dictionaries are then used to reduce the number of pages needed to store the same amount of data. This initial load employs the new Analyze load phase specific to column-organized tables.

Subsequent `LOAD` operations, `INGEST` operations and regular table `INSERTs` add page-level compression dictionaries to further compress data by exploiting new data patterns not found in the data used to populate the table initially.

When using a single load operation, metadata is created (synopsis tables and constraints). Also, statistics are collected automatically within the process. DBAs do not need to run **RUNSTATS** after loading data into column-organized tables.

## 2.5.5 Upgrade from a previous release to DB2 10.5

Upgrading from the previous version of DB2 release to DB2 10.5 requires the database to already be using a **UNICODE** code set and **IDENTITY** collation. If this is not the case, a new database must be created after the DB2 instance is upgraded to DB2 10.5 and activated for BLU Acceleration.

Use the following steps to upgrade an existing database:

1. Run the **db2chkupgrade** tool to verify if existing `GS_DB101` database can be upgraded:

   ```
   db2ckupgrade GS_DB101 -l db2ckupgrade.log
   ```

2. Upgrade the db2inst1 DB2 instance by using the **db2iupgrade** utility:

```
db2iupgrade db2inst1
```

3. Upgrade the databases by using one of the following methods:

   – In-place upgrade using the **UPGRADE DATABASE** command:

   ```
   db2 UPGRADE DATABASE GS_DB101 REBINDALL
   ```

   – If you do not choose to upgrade the database in place, you can restore a database from a DB2 backup image taken from an older DB2 release. This can be accomplished using the following **RESTORE** command. As part of the database restore, the database is then automatically upgraded to the DB2 10.5 level.

   ```
   db2 RESTORE DATABASE GS_DB101 FROM /backups TAKEN AT
   2014XXXXXXXXXX
   ```

4. Now, the database can be activated for BLU Acceleration. Follow the same steps described in 2.5.4, “Existing database deployments” on page 28.

## 2.6 Configuration preferred practices for BLU Acceleration deployment

This section explains some preferred practices for BLU Acceleration deployment.

### 2.6.1 Changes applied with DB2_WORKLOAD=ANALYTICS

When activating DB2 with BLU Acceleration by setting the single aggregate DB2 registry parameter, `DB2_WORKLOAD` to `ANALYTICS`, this new value causes the configuration advisor (`AUTOCONFIGURE`) to optimize various parameters for analytics workloads as listed:

- Database parameters:
  – `DFT_DEGREE=ANY`
  – `DFT_TABLE_ORG=COLUMN`
  – `PAGESIZE=32768`
  – `DFT_EXTENT_SZ=4`
  – `SORTHEAP`= [higher than default optimized for current hardware; self-tuning disabled]
  – `SHEAPTHRES_SHR`=[higher than default optimized for current hardware; self-tuning disabled]

  - UTIL_HEAP_SZ=AUTOMATIC (and at least 1 000 000 pages)
  - CATALOGCACHE_SZ=[higher than default optimized for current hardware]
  - AUTO_REORG=ON
- Workload management (WLM) objects:
  - Work Action SYSMAPMANAGEDQUERIES=Y
  - SYSDEFAULTMANAGEDSUBCLASS Service Subclass is enabled
  - SYSDEFAULTCONCURRENT threshold = [value optimized for current hardware]

## DFT_DEGREE=ANY

BLU Accelerated databases require intra-partition parallelism to be enabled. When configuring DB2_WORKLOAD=ANALYTICS, all incoming connections to a database are implicitly enabled with intra-partition parallelism, which is controlled through a workload management database object (WLM). This mechanism operates independently from the INTRA_PARALLEL instance-wide configuration parameter. Intra-partition parallelism allows DB2 to divide single database operations into multiple parts that can be executed in parallel threads on multiple processor cores.

Every new database (with or without DB2_WORKLOAD=ANALYTICS set) includes the same standard WLM object, a workload definition that is named SYSDEFAULTUSERWORKLOAD. All database connections are, by default, routed through this workload definition that applies certain attributes to the connections.

One of the attributes for the WLM object is MAXIMUM DEGREE DEFAULT. In this context, DEFAULT instructs DB2 to use a default value for the degree of parallelism. With DB2_WORKLOAD=ANALYTICS set, this directly translates into intra-partition parallelism to be enabled. When DB2_WORKLOAD is *not* set, intra-partition parallelism is disabled. If DB2_WORKLOAD is not set or if it is set to a value other than ANALYTICS, intra-partition parallelism can still be enabled on a per database level by altering SYSDEFAULTUSERWORKLOAD, as shown in Example 2-3 on page 27.

With intrapartition parallelism, the SQL optimizer can choose the degree of query parallelism based on the configured maximum degree. The degree of parallelism can be controlled on an instance level using the databases manager configuration parameter MAX_QUERYDEGREE, which by default, is set to ANY (numerical value of -1). On a database level, the degree of parallelism can be further controlled through the database parameter DFT_DEGREE, which by default is set to 1 and with DB2_WORKLOAD=ANALYTICS set to ANY (numerical value of -1). In addition, the degree can be controlled using the CURRENT DEGREE register or through binding the application with a specific value.

At run time, you can use the **SET RUNTIME DEGREE** command. An application can also use the stored procedure **ADMIN_SET_INTRA_PARALLEL** to enable or disable parallelism for its connection beginning with the next transaction.

For more details about managing and controlling intra parallelism, either instance wide or on a per statement level, see the DB2 10.5 documentation:

http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.partition.doc/doc/t0004886.html

## DFT_TABLE_ORG=COLUMN

The DFT_TABLE_ORG=COLUMN setting switches the default table organization to be implicitly organized by columns versus rows. An ORGANIZED BY [ROW | COLUMN] keyword in the CREATE TABLE statement syntax can be used to override that default.

## PAGESIZE=32768 and DFT_EXTENT_SZ=4

As suggested for BLU Acceleration, the database-wide default parameters have the following settings:

- The page size (PAGESIZE) is set to 32 KB.
- The extent size (DFT_EXTENT_SZ) is set to 4 pages.

These settings are assumed by DB2 if they are not specified during the creation of table spaces or buffer pools.

## SHEAPTHRES_SHR, SORTHEAP, and CATALOGCACHE_SZ

The database memory parameters CATALOGCACHE_SZ, SORTHEAP, and SHEAPTHRES_SHR are set to a value that is higher than the default and optimized for the hardware in use. Self-tuning must be disabled for both SHEAPTHRES_SHR and SORTHEAP. In addition, SHEAPTHRES database manager configuration parameter must be set to a value of 0 so that all sort memory consumers for the database is determined by the SHEAPTHRES_SHR value instead of private sort memory.

## UTIL_HEAP_SZ=AUTOMATIC

To assist the build process of the compression dictionary during table load operations, UTIL_HEAP_SZ should be set to at least 1 000 000 pages and AUTOMATIC.

## AUTO_REORG=ON

Automatic space reclamation is performed for column-organized tables and applies to BLU by setting the database parameters AUTO_MAINT=ON and AUTO_REORG=ON.

### Automatic workload management

Several other WLM objects are created and set to maximize throughput in your database on your hardware to process every running query when many large analytic type queries are submitted. For workload management in a BLU Acceleration environment, see 6.6, “Workload management” on page 205.

## 2.6.2 Memory distribution

In DB2 10.5 with BLU Acceleration, specifically after the auto-configure operation is applied (`DB2_WORKLOAD=ANALYTICS`), specific sort memory parameters are set to a fixed value and must be tuned manually. To achieve dramatic performance improvements for analytic workloads, BLU Acceleration use a set of innovative algorithms to join and group column-organized data in extremely fast manner. These algorithms utilize hashing techniques, which requires sort memory to process. Typically, because of the nature of analytics queries, sort memory becomes more important because more complex queries often require more working memory. The following operations commonly require sort memory:

- Hash joins
- Block index AND’ing
- Merge joins
- Dynamic bitmaps (used for index ANDing and star joins)
- Partial early distinct operations
- Partial early aggregation operations
- Hashed `GROUP BY`s
- Columnar vector buffering

In terms of sort memory parameters, we first look at the database-specific parameter, `SHEAPTHRES_SHR`, which defines the total amount of database shared memory that can be used by operations using sort memory (when the `SHEAPTHRES` database manager configuration parameter is set to `0`). The `SORTHEAP` database parameter configures how much of that overall memory (`SHEAPTHRES_SHR`) DB2 can be dedicated to each of these operations.

For workloads that exhibit lower query concurrency (less than 20 concurrent queries), we suggest using the `DB2_WORKLOAD=ANALYTICS` autoconfigured distribution of memory, which is as much memory for buffer pools as for sort memory with a total of approximately 80% of the *available database memory*. This means half of that memory, about 40% of the total database memory, should be made available to sort memory, which can be set through the `SHEAPTHRES_SHR` database configuration parameter. Then, the other half of that memory should be allocated to those buffer pools that are frequently accessed by the queries of your workload.

You can look up the *available database memory* through the DATABASE_MEMORY database configuration parameter. However, this memory can possibly be increased to the value of the INSTANCE_MEMORY database manager configuration parameter, which specifies the global memory available to all databases in the same DB2 instance. Keep in mind that if you are currently in a development or testing phase without a workload constantly running, the self-tuning memory manager might have tuned down the memory allocation for the database (DATABASE_MEMORY). In this case, more unused memory is likely available that should be considered as *available database memory*.

With increasing query concurrency, the suggestion is to manually tune the database configuration by decreasing the total buffer pool memory in favor of sort memory (SHEAPTHRES_SHR), because queries likely benefit from increased sort memory. Consider this example:

- Set 40% of available database memory to SHEAPTHRES_SHR, and 40% to buffer pool for less than 20 concurrent workloads.
- Set 50% of available database memory to SHEAPTHRES_SHR, and lower buffer pool to 25% for higher concurrency.

Further testing and tuning should be made depending on your own workloads and requirements.

If not enough overall sort memory is available, your workload may benefit additionally from reducing the memory heap (database configuration parameter SORTHEAP) that is available to each operation requiring sort. Typical SHEAPTHRES_SHR:SORTHEAP ratio usually lies between 1:5 to 1:20. Examples of SHEAPTHRES_SHR and SORTHEAP ratios are as follows:

- Set SORTHEAP to a value of (SHEAPTHRES_SHR/5) for low concurrency
- Set SORTHEAP to a value of (SHEAPTHRES_SHR/20) for higher concurrency

Using these examples as baseline, further adjustments should be made depending on your own workloads and environment.

Analytic queries are often more complex and can take advantage of higher than default memory values for STMTHEAP, which defines how much heap memory is available during the statement preparation. If you see queries returning SQL0437W with reason code 1, STMTHEAP should be increased by 50% in the tuning steps.

### 2.6.3 Creating column-organized tables

Creating a column-organized table is straightforward. Existing DDL statements to create tables can remain the same if the database is created or autoconfigured with DB2_WORKLOAD=ANALYTICS enabled. The reason is that the default table organization setting (DFT_TABLE_ORG database parameter) was set to COLUMN.

This means, when it is not explicitly specified, `CREATE TABLE` creates new tables in column-organized format by default. To create a row-organized table in such a configured database, specify the `ORGANIZED BY ROW` clause in your `CREATE TABLE` statement. Column-organized tables are created in automatic storage table spaces.

### Optional: Table space setup

As of DB2 9.1, databases are created by using automatic storage, by default. Many new autonomic DB2 features, going forward, are supported only on automatic storage, as are column-organized tables.

DB2 includes a default large table space, called `USERSPACE1`, for storing user-defined data and objects. This can be sufficient for test environments. However, for users who are looking to deploy BLU Acceleration in production and for better manageability, we suggest organizing your data in your own table spaces.

Creating an automatic storage table space is simple. Example 2-9 shows a command to create a new automatic storage table space. For column-organized tables, we suggest using 32 K page size and an extent size of 4 pages. This suggestion is already part of the default settings when the database is created in analytics mode; therefore no extra option is needed.

*Example 2-9 Creating an automatic storage table space*

```
db2 create tablespace GOSALES_COLTS
```

**Tip:** Keep your fact tables and dimension tables in separate table spaces to maximize performance and manageability. For instance, with fact tables in their own table spaces, storage groups, and buffer pools, you can independently manage your backup and restore of fact tables.

### CREATE TABLE statement

When creating tables, the `ORGANIZE BY` clause can be used to explicitly specify the table organization you want. For existing DB2 users, you can optionally use the **`db2look`** utility to generate a DDL script. For example, you can explicitly define specific tables in either format despite the default organization format configured with `DFT_TABLE_ORG`.

You can create a new column-organized table using `CREATE TABLE`. Because the table organization is defaulted to `COLUMN` when `DB2_WORKLOAD=ANALYTICS`, specifying the `ORGANIZE BY` clause in the statement is not required. In Example 2-10 on page 37, we create a column-organized table in `GOSALES_COLTS` using the `CREATE TABLE` statement. The `ORGANIZE BY COLUMN` clause here is purely optional as the database was configured for an `ANALYTICS` workload.

*Example 2-10 CREATE TABLE statement*

```
CREATE TABLE GOSALESDW.EMP_SUMMARY_FACT (
   ORGANIZATION_KEY INTEGER NOT NULL,
   EMPLOYEE_KEY INTEGER NOT NULL,
   SALARY DECIMAL(19,2) NOT NULL,
   VACATION_DAYS_TAKEN DOUBLE NOT NULL)
IN GOSALES_COLTS
ORGANIZE BY COLUMN;
```

> **Tip:** Put as many tables as possible into column-organized format for an environment with a purely analytics or OLAP workload. This includes typical data marts or data warehouse workloads with extensive range scans, grouping, joins, or aggregation.

### *INDEX IN clause*

As with row-organized tables, it is a common database preferred practice to keep indexes in their own table spaces. In BLU Acceleration, no user-defined indexes are required, but it is possible to define informational primary key or unique constraints as they can be beneficial to the query optimizer. Informational (not enforced) keys require less storage and time to create. Also, all index objects in BLU Acceleration are maintained by DB2 automatically, without a DBAs intervention.

> **Tip:** Take advantage of not enforced key constraints in your environment. Keep indexes in separate table spaces.

Users who have primary keys or unique constraints in their tables can optionally group index objects in a separate table space by using the `INDEX IN` clause, as demonstrated in Example 2-11.

*Example 2-11 CREATE TABLE statement with keys*

```
CREATE TABLE GOSALESDW.MRK_PRODUCT_SURVEY_FACT (
   BRANCH_KEY INTEGER NOT NULL,
   PRODUCT_KEY INTEGER NOT NULL,
   PRODUCT_SURVEY_KEY INTEGER NOT NULL
   PRODUCT_TOPIC_SCORE DOUBLE
   PRIMARY KEY (PRODUCT_KEY) )
IN GOSALES_COLTS INDEX IN GOSALES_IDXTS
ORGANIZE BY COLUMN;
```

## 2.6.4 Using insert and ingest with column-organized tables

DB2's user interfaces for adding data to column-organized tables are identical to the traditional row-organized tables. While initial LOAD operations on empty tables create column-level compression dictionaries to compress data, subsequent LOADs, table inserts, and inserting data using the INGEST utility in addition create page-level compression dictionaries automatically to further improve compression rates on column-organized tables. The following examples merely demonstrate the usual user interface known for loading data into row-organized tables.

Example 2-12 shows an example for an insert on a column organized table and how to verify the number of pages used with the percentage of pages saved with page-level compression dictionary.

*Example 2-12 Insert into a column-organized table creates page-level dictionary*

```
CREATE TABLE "DB2BLU  "."FLIGHTS"  (
  "origin_airport" VARCHAR(5 OCTETS) NOT NULL ,
  "destination_airport" VARCHAR(5 OCTETS) NOT NULL ,
  "flights" BIGINT NOT NULL ,
  "passengers" BIGINT ,
  "month" BIGINT )
 IN "USERSPACE1"
 ORGANIZE BY COLUMN;


[db2blu@db2pure06 scripts]$ db2 "INSERT INTO FLIGHTS_INSERT SELECT *
FROM FLIGHTS"


[db2blu@db2pure06 ~]$ db2 runstats on table FLIGHTS_INSERT
  with distribution on all columns and detailed indexes all;

[db2blu@db2pure06 scripts]$ db2 -tvf page_level_dict_check.sql
SELECT substr(tabschema,1,20) AS TABSCHEMA, substr(tabname,1,20) AS
TABNAME,CARD,NPAGES,PCTPAGESSAVED FROM SYSSTAT.TABLES WHERE
TABSCHEMA='DB2BLU'

TABSCHEMA            TABNAME              CARD                 NPAGES               PCTPAGESSAVED
-------------------- -------------------- -------------------- -------------------- -------------
DB2BLU               FLIGHTS                           3606803                  719            87
DB2BLU               FLIGHTS_INSERT                    3606803                 1294            68
```

In Example 2-13 on page 39, the same operation is done using ingest instead of insert. It is possible to check how the table benefits from page-level compression with a lower number of pages and 70% percent of pages saved with this compression technique.

*Example 2-13 Ingest into a column-organized tables creates page-level dictionary*

```
[db2blu@db2pure06 ~]$ db2 "ingest from file flights_with_colnames.csv
format delimited insert into FLIGHTS_INGEST"
SQL2979I  The ingest utility is starting at "14/08/2014
22:49:07.780995".
SQL2914I  The ingest utility has started the following ingest job:
"DB21005:20140814.224907.780995:00002:00020".

Number of rows read         = 3606803
Number of rows inserted     = 3606803
Number of rows rejected     = 0

SQL2902I  The ingest utility completed at timestamp "14/08/2014
22:49:23.514394". Number of errors: "0". Number of warnings: "0".

[db2blu@db2pure06 ~]$db2 runstats on table FLIGHTS_INSERT
  with distribution on all columns and detailed indexes all;

[db2blu@db2pure06 scripts]$ db2 “SELECT substr(tabschema,1,20) AS
TABSCHEMA, substr(tabname,1,20) AS TABNAME,CARD,NPAGES,PCTPAGESSAVED
FROM SYSSTAT.TABLES WHERE TABSCHEMA='DB2BLU'”

TABSCHEMA            TABNAME              CARD                 NPAGES               PCTPAGESSAVED
-------------------- -------------------- -------------------- -------------------- -------------
DB2BLU               FLIGHTS                           3606803                  719            87
DB2BLU               FLIGHTS_INGEST                    3606803                 1474            70
DB2BLU               FLIGHTS_INSERT                    3606803                 1294            68
```

## 2.6.5 Converting tables to column-organized tables

By using the **db2convert** tool included in DB2 10.5, the tables that are originally created in row-organized tables can be directly converted in an automated fashion. The **db2convert** tool uses the ADMIN_MOVE_TABLE stored procedure to move data between the source table and a target table that it automatically defines as column-organized. During this operation, it removes unnecessary objects such as indexes and materialized query tables (MQTs). User-defined tables can be selected for the conversion process by using the following criteria:

- Single tables (**-t** parameter)
- All tables created by a specific user (**-u** parameter)
- All tables associated with a particular schema (**-z** parameter)
- All tables in the database (**-d** parameter).

Alternatively, a target table space that differs from the source can be specified using the **-ts** parameter.

Example 2-14 through Example 2-17 demonstrate these parameters.

*Example 2-14 Converting an existing user-defined table to column-organized format*

```
$ db2convert -d GS_DB -z GOSALESDW -t EMP_SUMMARY_FACT
```

*Example 2-15 Converting all tables created by user martin in different table space*

```
$ db2convert -d GS_DB -u martin -ts coltbsp
```

*Example 2-16 Converting all user-defined tables in a given schema*

```
$ db2convert -d GS_DB -z GOSALESDW
```

*Example 2-17 Converting all user-defined tables in a database*

```
$ db2convert -d GS_DB
```

For more information about **db2convert**, see the following resource:

http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0060728.html

## 2.7 Configuration preferred practices for HADR

This section explains some general preferred practices for deploying HADR, including deploying HADR for column-organized tables.

### 2.7.1 Column-organized tables now support high availability and disaster recovery

DB2 High Availability and Disaster Recovery (HADR) has been available for a long time and it offers an effortless solution for high availability (HD) and disaster recovery (DR) for your DB2 database. Now the DB2 Cancun Release 10.5.0.4 release extends the HADR feature to column-organized tables.

HADR works in the same way with column-organized tables as with row-organized tables and it supports the same functionality and synchronization modes. In this section, we describe how HADR in DB2 works and how it is configured.

### Introduction

Whether you are looking to make your database highly available or ready for disaster recovery, HADR is the DB2 feature to use. HADR ships database transaction logs at runtime between a primary database and a standby database. The standby database receives logs from the primary database and replays these logs in real time to keep the standby database in sync with the primary database.

### Easy setup

To set up HADR, you can simply use a backup image from the primary HADR database to establish a standby HADR database. The configuration for HADR is also fairly simple as all is done on the database configuration parameters. After you have the necessary configuration updated and the backup restored on the standby database, you can start the HADR with one of the four synchronization modes available:

- SYNC
- NEARSYNC
- ASYNC
- SUPERASYNC

### Roles

In a high availability and disaster recovery scenario, an HADR setup has different types of roles for each mode according to the context of each database server.

The basic HADR scenario has two databases on two database servers. The source database, where applications are connecting to, is known as the *primary* database. The target database, where transaction logs coming from the primary database are applied, is known as the *standby* database. As its name says, the role of the target database is to be in synchronization with the primary database. in case of a disaster recovery or just a failure on the primary database, the standby server can become active and applications can seamlessly connect to it.

In addition to this basic scenario, HADR also allows to have additional standby databases for higher availability and for reporting purposes. In a multiple standby mode, it is possible to have up to three standby databases. One of these is the principal HADR standby database and any other standby database is known as the *auxiliary* standby.

## 2.7.2 Column-organized tables and HADR synchronization modes

Users can decide the level of protection from potential loss of data by specifying one of the four HADR synchronization modes:

- Synchronous (SYNC)

  This mode provides the greatest protection against transaction loss and it also has the longest transaction response time when compared to the other three modes.

  Log write is considered successful only when the data is written on the primary log after receiving an acknowledgement to confirm that it is also written to the log files of the standby database. In this mode, there can be no data loss while HADR is in Peer state.

- Near-synchronous (NEARSYNC)

  This is the default mode for an HADR set up. This mode provides a shorter transaction response time compared to the synchronous mode but it also results in a slightly less protection against a possible transaction loss.

  Log write is successful only when the log buffer on the primary is written to log files on the primary and an acknowledgement is received when the log buffer is received on the standby.

- Asynchronous (ASYNC)

  The ASYNC mode offers a shorter transaction response time when compared to SYNC and NEARSYNC, but it also might cause greater transaction loss in case of a failure of the primary database.

  Log write is considered successful when logs are written to disk on the primary and the log data is sent through the network TCP/IP socket to the standby. In this mode, data loss can occur as there is no acknowledgement if the log data is received on the standby or not.

- Super-asynchronous (SUPERASYNC)

  The transaction response time in this mode is the shortest compared to the others, but this also result in the highest probability of transaction loss if the primary database fails. This can be useful to avoid transactions to be blocked or experience higher response time due to network traffic.

  Log writes are considered as successful when the log records are written to log files on the primary database. The primary database does not wait for any acknowledgement from the standby database and transactions are marked as committed even before the log data is sent over to the network.

DB2 Cancun Release 10.5.0.4 with BLU Acceleration supports all these HADR synchronization modes for databases with column-organized tables. The only feature not supported so far for column-organized tables is the possibility to use the standby database in read-only, known as “read on standby” (RoS).

Figure 2-1 illustrates the different HADR synchronization modes supported in DB2.

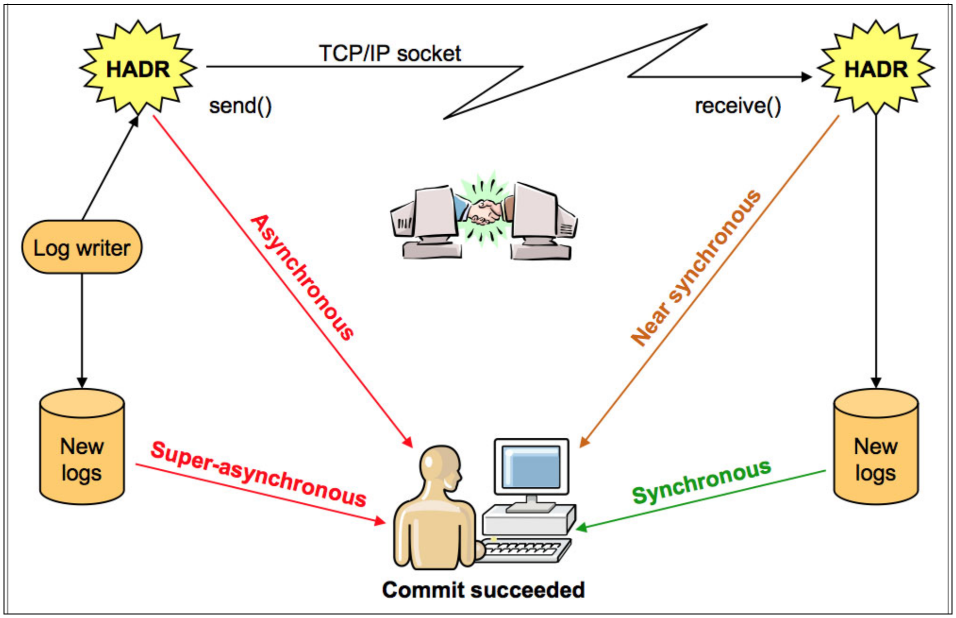



*Figure 2-1 DB2 10.5 supported synchronization modes*

## 2.7.3 Configuring primary and standby databases

One of the benefits of HADR is its simplicity to set up and configure to enable high availability and disaster recovery capabilities for your mission critical database. In failover situations, you can just recover your database from a planned or unplanned failure by just using one DB2 command.

### Requirements for setting up HADR

To set up HADR, there are some requirements to be met by the environment:

- The operating system and DB2 version and level must be the same on the primary server and any standby server.
- DB2 must have the same bit size (Intel, Power).
- TCP/IP interface must be available between the HADR primary server and the standby server.
- The database name must be the same on the primary and any standby server

The following steps are involved to enable HADR on your database based on a basic scenario with one primary and one standby database:

1. Determine the information for your environment.
2. Update the database configuration on your primary database.

3. Take a backup image of your primary database.
4. Copy and restore the backup image on your standby database server.
5. Update the database configuration on your standby database.
6. Start HADR on the standby database.
7. Start HADR on the primary database.

## Primary database

The information required to set up HADR in your environment is basically as follows:

- Standby and primary host name
- Standby and primary instance name
- Standby and primary service port for HADR
- Synchronization mode to be used

When you know this information, you can start to update the configuration on the primary database. You use DB2 commands to update the HADR related parameters that establish the communication between primary and standby databases.

To enable HADR, complete the following steps on the primary database server:

1. Activate log archive for your database. Use the following commands to enable log archive and specify where the archive log would be placed (LOGARCHMETH1):

   ```
   db2 "update db cfg for BLUDB using LOGRETAIN recovery"
   db2 "update db cfg for BLUDB using LOGARCHMETH1 DISK:/space/BLUDB/ARCH"
   ```

2. Update the database configuration parameters that tell DB2 how to connect the primary- standby pair. Here you specify both primary and standby host names, connection ports and the HADR synchronization mode required.

   ```
   db2 "'UPDATE DB CFG FOR BLUDB USING
        HADR_LOCAL_HOST $PRIMARY_HOSTNAME
        HADR_LOCAL_SVC $PRIMARY_PORT
        HADR_TARGET_LIST  {$STANDBY_HOSTNAME}
        HADR_REMOTE_HOST  $STANDBY_HOSTNAME
        HADR_REMOTE_SVC   $STANDBY_PORT
        HADR_REMOTE_INST  $STANDBY_INSTANCE
        LOGINDEXBUILD ON
        HADR_SYNCMODE     $MODE'"
   ```

3. Take a backup from your primary database to build the standby database.

   ```
   db2 "backup database $DBNAME online to $BACKUP_DIR"
   ```

   When the backup is complete, take note of the timestamp because that will be used during the restore step.

### Standby database

On the database server where you want to deploy the standby database, you only need to have a DB2 instance with same operating system, same version and level code as the primary database server.

Based on that, you just have to restore the backup image taken before from the primary that has to be copied over for the restore.

Use the following command to restore the database to build the standby database:

```
db2 "restore database $DBNAME taken at $BACKUP_TIMESTAMP from
$ARCH_DIR"
```

Just after the database restore, the next step is to update the database configuration for the standby database. Use the following command to update the HADR related parameters needed to establish a network connection between the standby and the primary databases:

```
db2 "'UPDATE DB CFG FOR BLUDB USING
     HADR_LOCAL_HOST $STANDBY_HOSTNAME
     HADR_LOCAL_SVC $STANDBY_PORT
     HADR_TARGET_LIST  $PRIMARY_HOSTNAME
     HADR_REMOTE_HOST  $STANDBY_HOSTNAME
     HADR_REMOTE_SVC   $PRIMARY_PORT
     HADR_REMOTE_INST  $PRIMARY_INSTANCE
     LOGINDEXBUILD ON
     HADR_SYNCMODE     $MODE'"
```

With this step completed, the standby is now ready to be activated and HADR can start to ship transaction logs from the primary to the standby database.

### Starting HADR

One of the features often appreciated by clients is the simplicity to set up and configure HADR for high availability or disaster recovery.

This is also shown during the process to start HADR, where only one command needs to be run on each standby and primary database.

#### *Starting HADR on the standby*

The following command must be run on the standby database to start it as standby:

```
db2 start hadr on database $DBNAME as standby
```

After HADR is started on the standby, the standby database is ready to start to receive transaction logs from the primary.

### *Starting HADR on the primary*

On the primary, run the following command so DB2 starts HADR as primary and starts to send the transaction log to the standby database according to the synchronization mode selected during the configuration steps:

```
db2 start hadr on database $DBNAME as primary
```

## 2.7.4 Automatic Client Reroute and HADR

Now that HADR is started, the primary database is shipping transaction logs to the standby database. At the same time, the standby database is applying them so it can be in sync with the primary based on the synchronization mode selected for the environment.

But what if the primary database suffers an unplanned failure and suddenly becomes not available? HADR is used for these scenarios and the standby database can be made the new primary to recover from the failure on the primary.

A really useful DB2 feature that comes to help in these kind of situations is Automatic Client Reroute. You can define an alternate server to each of your primary and standby databases. The alternate server information is retrieved and refreshed by DB2 clients connecting to the database server. By connecting the client to alternate server once, the client will have the alternate server information to reroute when needed.

Alternate servers can be enabled on primary and standby using the following commands:

- Primary pointing to standby:

  ```
  db2 UPDATE ALTERNATE SERVER FOR DATABASE $DATABASE USING HOSTNAME
  $STANDBY PORT $INSTANCE_PORT
  ```

- Standby pointing to primary:

  ```
  db2 UPDATE ALTERNATE SERVER FOR DATABASE $DATABASE USING HOSTNAME
  $MEMBER PORT $INSTANCE_PORT
  ```

Having these settings configured on both databases means that in case of an unplanned failure on the primary database, you can just do a failover to the standby. When the applications fail to connect to the primary database, they make use of the alternate server parameter to connect to the new primary database just when it becomes available.

### 2.7.5 Switching the roles in your HADR environment

As the name says, HADR feature in DB2 offers a high level of protection for your database. A standby database where all data from the primary is being replicated is useful not only in the event of disaster recovery but also for maintenance purposes. With few simple DB2 commands, you can switch the role for the standby database to become the primary weather you have an planned or unplanned failure (failover) in your environment.

#### *Unplanned failure*

In the event of an unplanned failure affecting your primary database, it is possible to easily switch the primary role to the standby database to restore the database service as soon as you realize about the failure.

In the event of an unplanned failure, the DB2 command to perform a takeover from the standby specifies the “BY FORCE” parameter that lets DB2 know that it does not have to wait any acknowledge from the actual primary before switching the role (as the primary could be down and might not respond).

```
DB2 TAKEOVER HADR ON DATABASE $DATABASE BY FORCE
```

After the command is run, it just takes few seconds to complete and the new primary database will be available to attend to clients and applications requests.

#### *Planned takeover*

HADR is also useful to handle upgrades or failures smoothly in your environment and a planned takeover can be very useful for a planned temporary downtime on the primary database.

With an easy command on the standby, it is possible to switch the primary role to the standby database so it becomes the new primary, while the original primary has its role changed to standby:

```
db2 takeover hadr on database $DATABASE
```

At this point, clients and application can connect to the new primary and the new standby starts to receive the data replicated from the primary.

### *Automated takeover using Tivoli System Automation*

It is also possible to automate the process to perform a takeover using Tivoli System Automation for Multiplatforms.

For further details, more information is available here:

http://www.ibm.com/developerworks/data/tutorials/dm-1009db2hadr/index.html?ca=drs

## 2.7.6 Checking HADR status

To check the actual status of HADR configuration, you can use the `mon_get_hadr` table function or the **db2pd** command.

Example 2-18 shows the table function example. Perform this command on the primary database server.

*Example 2-18 Checking HADR status with table function*

```
db2 "select HADR_ROLE,HADR_SYNCMODE, HADR_STATE,
varchar(PRIMARY_MEMBER_HOST,30)
as PRIMARY_MEMBER_HOST, varchar(STANDBY_MEMBER_HOST,30) as
STANDBY_MEMBER_HOST from table (mon_get_hadr(-2))"

HADR_ROLE     HADR_SYNCMODE HADR_STATE              PRIMARY_MEMBER_HOST
STANDBY_MEMBER_HOST
------------- ------------- -----------------------
----------------------------- -----------------------------
PRIMARY       SYNC          PEER                    db2pure06
db2pure04

  1 record(s) selected.
```

The following **db2pd** command syntax is used for checking the HADR status:

```
db2pd -hadr -db $DATABASE
```

When issuing this command on the primary, it shows information for both the primary and standby. When issuing this command on the standby, the output is just for that specific standby.

Example 2-19 shows an example of checking HADR status on the primary with **db2pd**. The first row HADR_ROLE = PRIMARY indicates that the output is for the primary database.

*Example 2-19 Checking HADR status on the primary with db2pd*

```
[db2blu@db2pure06 ~]$ db2pd -hadr -db bludb

Database Member 0 -- Database BLUDB -- Active -- Up 0 days 00:08:57 -- Date
2014-07-28-12.43.29.955740

                            HADR_ROLE = PRIMARY
                          REPLAY_TYPE = PHYSICAL
                        HADR_SYNCMODE = SYNC
                           STANDBY_ID = 1
                        LOG_STREAM_ID = 0
                           HADR_STATE = PEER
                           HADR_FLAGS =
                  PRIMARY_MEMBER_HOST = db2pure06
                     PRIMARY_INSTANCE = db2blu
                       PRIMARY_MEMBER = 0
                  STANDBY_MEMBER_HOST = db2pure04
                     STANDBY_INSTANCE = db2blu
                       STANDBY_MEMBER = 0
                  HADR_CONNECT_STATUS = CONNECTED
             HADR_CONNECT_STATUS_TIME = 28/07/2014 12:38:37.693945 (1406547517)
          HEARTBEAT_INTERVAL(seconds) = 30
                     HEARTBEAT_MISSED = 0
                   HEARTBEAT_EXPECTED = 13
                HADR_TIMEOUT(seconds) = 120
        TIME_SINCE_LAST_RECV(seconds) = 22
             PEER_WAIT_LIMIT(seconds) = 0
           LOG_HADR_WAIT_CUR(seconds) = 0,000
    LOG_HADR_WAIT_RECENT_AVG(seconds) = 0,005752
   LOG_HADR_WAIT_ACCUMULATED(seconds) = 0,012
                  LOG_HADR_WAIT_COUNT = 2
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 19800
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
            PRIMARY_LOG_FILE,PAGE,POS = S0001007.LOG, 0, 12679490414
            STANDBY_LOG_FILE,PAGE,POS = S0001007.LOG, 0, 12679490414
                  HADR_LOG_GAP(bytes) = 0
     STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0001007.LOG, 0, 12679490414
       STANDBY_RECV_REPLAY_GAP(bytes) = 0
                     PRIMARY_LOG_TIME = 28/07/2014 12:34:56.000000 (1406547296)
                     STANDBY_LOG_TIME = 28/07/2014 12:34:56.000000 (1406547296)
              STANDBY_REPLAY_LOG_TIME = 28/07/2014 12:34:56.000000 (1406547296)
         STANDBY_RECV_BUF_SIZE(pages) = 4304
             STANDBY_RECV_BUF_PERCENT = 0
           STANDBY_SPOOL_LIMIT(pages) = 460000
                STANDBY_SPOOL_PERCENT = 0
                   STANDBY_ERROR_TIME = NULL
                 PEER_WINDOW(seconds) = 0
             READS_ON_STANDBY_ENABLED = N
```

Example 2-20 shows the **db2pd** command run on the standby database server.

*Example 2-20 Checking the status of the standby server with db2pd*

```
[db2blu@db2pure04 ~]$ db2pd -hadr -db bludb

Database Member 0 -- Database BLUDB -- Standby -- Up 0 days 00:05:26 -- Date
2014-07-28-12.44.03.587977

                            HADR_ROLE = STANDBY
                          REPLAY_TYPE = PHYSICAL
                        HADR_SYNCMODE = SYNC
                           STANDBY_ID = 0
                        LOG_STREAM_ID = 0
                           HADR_STATE = PEER
                           HADR_FLAGS =
                  PRIMARY_MEMBER_HOST = db2pure06
                     PRIMARY_INSTANCE = db2blu
                       PRIMARY_MEMBER = 0
                  STANDBY_MEMBER_HOST = db2pure04
                     STANDBY_INSTANCE = db2blu
                       STANDBY_MEMBER = 0
                  HADR_CONNECT_STATUS = CONNECTED
             HADR_CONNECT_STATUS_TIME = 28/07/2014 12:38:37.696446 (1406547517)
          HEARTBEAT_INTERVAL(seconds) = 30
                     HEARTBEAT_MISSED = 0
                   HEARTBEAT_EXPECTED = 10
                HADR_TIMEOUT(seconds) = 120
        TIME_SINCE_LAST_RECV(seconds) = 19
             PEER_WAIT_LIMIT(seconds) = 0
           LOG_HADR_WAIT_CUR(seconds) = 0,000
    LOG_HADR_WAIT_RECENT_AVG(seconds) = 0,005752
   LOG_HADR_WAIT_ACCUMULATED(seconds) = 0,012
                  LOG_HADR_WAIT_COUNT = 2
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 19800
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
            PRIMARY_LOG_FILE,PAGE,POS = S0001007.LOG, 2, 12679500171
            STANDBY_LOG_FILE,PAGE,POS = S0001007.LOG, 2, 12679500171
                  HADR_LOG_GAP(bytes) = 0
     STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0001007.LOG, 2, 12679500171
       STANDBY_RECV_REPLAY_GAP(bytes) = 0
                     PRIMARY_LOG_TIME = 28/07/2014 12:43:45.000000 (1406547825)
                     STANDBY_LOG_TIME = 28/07/2014 12:43:45.000000 (1406547825)
              STANDBY_REPLAY_LOG_TIME = 28/07/2014 12:43:45.000000 (1406547825)
         STANDBY_RECV_BUF_SIZE(pages) = 4304
             STANDBY_RECV_BUF_PERCENT = 0
           STANDBY_SPOOL_LIMIT(pages) = 460000
                STANDBY_SPOOL_PERCENT = 0
                   STANDBY_ERROR_TIME = NULL
                 PEER_WINDOW(seconds) = 0
             READS_ON_STANDBY_ENABLED = N
```

### 2.7.7 Considerations for fix pack upgrades in an HADR environment

When applying a fix pack upgrade in a DB2 HADR environment, there are some considerations to be made. The procedure of upgrading the DB2 software is done by the rolling update process and this keeps the database service available during the upgrade. This means that the database stays up while each database server is being upgraded, one at a time.

In case of an HADR environment, the upgrade starts with the standby member. After the upgrade is done on the standby member, a takeover is performed to switch the primary role to the recently upgraded member. At this point, after all the clients and applications are connected to the standby database (that is, using automatic client reroute), the original primary can be upgraded. After the original primary (now standby) is upgraded, HADR synchronization can continue and once the pair becomes in peer state again, the upgrade procedure is complete.

For more details and the detailed instructions to perform a rolling update in a HADR environment, see the following website:

http://www-01.ibm.com/support/knowledgecenter/#!/SSEPGG_10.5.0/com.ibm.db2.luw.admin.ha.doc/doc/t0011766.html

### 2.7.8 HADR best practices

HADR is a proven technology that offers high availability and disaster recovery under various DB2 environments.

Here are a few suggestions for an HADR environment:

- Use dedicated and high performing disks or file systems for the database logs.

  Database changes are logged in a single log stream, so it is important to have enough I/O capacity available on the file system.

- Make the location of archived logs accessible to both the primary and standby databases.

  The primary database is the one that archives logs, but after a takeover, the new primary database (original standby database) starts to archive logs as well. During a takeover, the step to catch up and synchronize both databases could require to retrieve archive logs. Using a shared location for primary and standby simplifies the configuration and also reduces the network traffic caused by archive logs.

- Use a dedicated network for the HADR primary-standby connection.

  Though each of synchronization modes might have different network bandwidth requirements, the performance and stability of HADR rely on the network communication channel. In particular, the bandwidth of this network should meet certain requirements based on the log generation rate.

- Consider using multiple network adapters.

  For high availability, multiple network adapters should be used to ensure that the failure of a single network adapter does not result in the loss of network.

- Consider using a virtual IP for the database server.

  By using a virtual IP for the database server, the client applications can always connect to the same virtual IP address independently from which member actually has the primary role.

- Consider using automatic client reroute.

  Instead of a virtual IP, a different option to consider is to use the automatic client reroute feature built in DB2 engine. In the event of a failover, a DB2 client is redirected from the primary server to the standby server.

For more information about the HADR best practices, see *High Availability Disaster Recovery* at this website:

https://ibm.biz/Bdx2Ga

# 3

# Planning and deployment of BLU Acceleration shadow tables for mixed workload environments

In this chapter, we introduce the DB2 shadow table feature that consolidates DB2 online transaction processing (OLTP) with BLU-accelerated online analytic processing (OLAP) optimizations into a single Online Transactional Analytical Processing (OLTAP) database.

The following topics are covered:

- Overview
- Prerequisites
- Deployment
- Configuration preferred practices
- Operational preferred practices
- Shadow tables and HADR

## 3.1 Overview

DB2 with BLU Acceleration introduces a new innovative feature that allows businesses to further simplify their IT landscape. Businesses can now perform complex analytic processing within their OLTP environments. Shadow tables exploit the revolutionary performance of DB2 with BLU Acceleration technologies to make analytic processing faster, simpler, and easier for OLTP systems.

### 3.1.1 Data environments with mixed workloads

Traditionally, businesses have relied on both OLTP and OLAP systems to meet their business requirements. OLTP systems process simple, ongoing transactions such as, creating and processing orders and the processing of data in real-time. OLAP systems are used for more complex processing of aggregated and multi-dimensional data from multiple sources to support business intelligence. Each system works independently to provide a key part of the overall data processing needs.

There is a growing need to combine the processing of daily transactions with continuous or on-demand analytics, to improve decision making in key business areas. These environments require a convergence of OLTP and OLAP workloads into a new hybrid system that processes both. OLTAP systems have the ability to run complex analytic queries on real-time transactional data. By bringing together the processing of day-to-day transactions and complex analytics, OLTAP systems reduce the complexity of processing mixed workloads, making the system more agile with minimal trade-off.

Figure 3-1 on page 55 shows how DB2 handles different type of workloads. For purely transactional processing, DB2 pureScale is optimized to provide continuous availability and exceptional scalability. When working with data warehouses, data mining, or other OLAP queries, the DB2 Database Partitioning Feature (DPF) provides the performance boost to query processing, roll-in and roll-out of large data sets. The ground-breaking in-memory, column-organized processing in DB2 with BLU Acceleration further enhances the processing of analytic workloads. Now, DB2 introduces a new feature to truly transform how mixed workloads are handled. Systems that require analytic reporting in a predominantly OLTP environment, can now benefit from columnar processing technologies for their row-organized data.

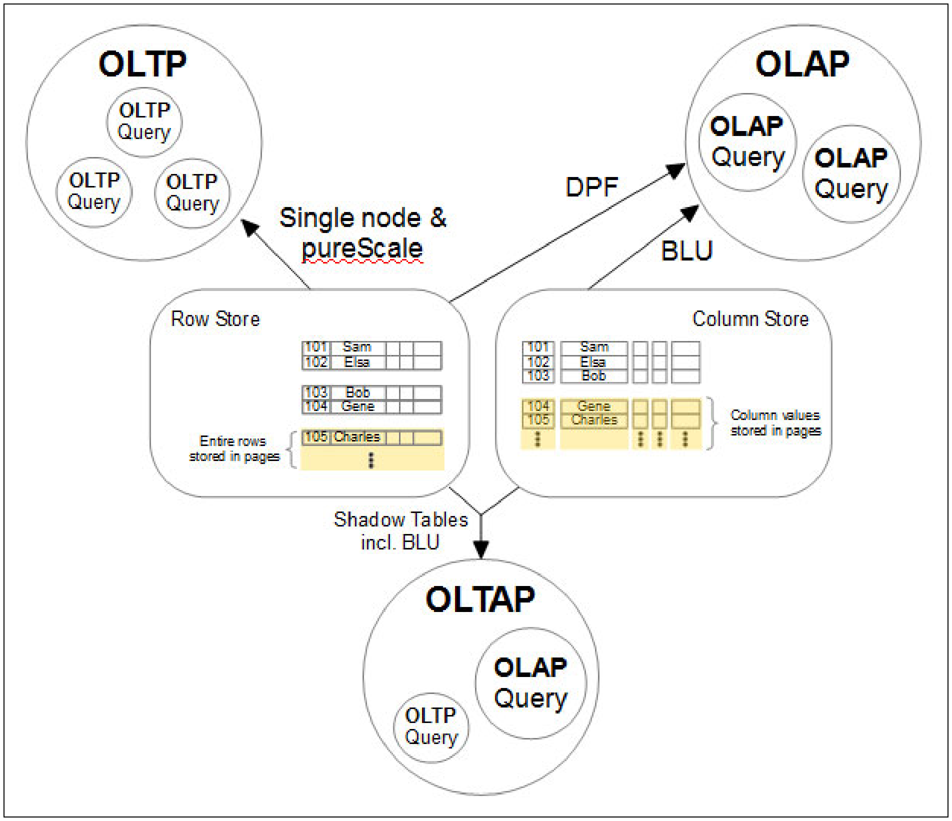



*Figure 3-1 Workloads and DB2*

Before shadow tables, mixed workloads within predominantly OLTP systems used multiple indices to allow for OLAP workloads to be run alongside OLTP workloads. The new shadow table feature introduced with DB2 Cancun Release 10.5.0.4 uses BLU Acceleration to eliminate the overhead that is involved in the traditional handling of mixed workloads.

## 3.1.2 DB2 with BLU Acceleration and shadow tables

DB2 with BLU Acceleration technologies allow mixed workloads to be processed faster by combining traditional row-organized with column-organized tables in the same database. Hybrid workloads demand high transactional rates (OLTP) while also reporting on the same data using more complex queries (OLAP). DB2 offers the ability to use an OLTP and OLAP optimized query execution environment at the same time using the BLU Acceleration technologies.

By allowing table data to be stored as both row and column-organized tables, the DB2 optimizer is able to choose the best suitable execution environment for different queries that access the same table data. This new feature illustrated in Figure 3-2 is called *shadow tables*. Shadow tables allow an existing source table organized by row to be "shadowed" by a column-organized table. The shadow table is maintained by replication from the source table. Being column-organized, it builds the foundation to be used with the column data engine that is part of the BLU Acceleration technologies. With the use of shadow tables, OLTP queries, continue to rely on row-organized tables, while OLAP queries have the advantage of being transparently re-routed to the column-organized "shadows".

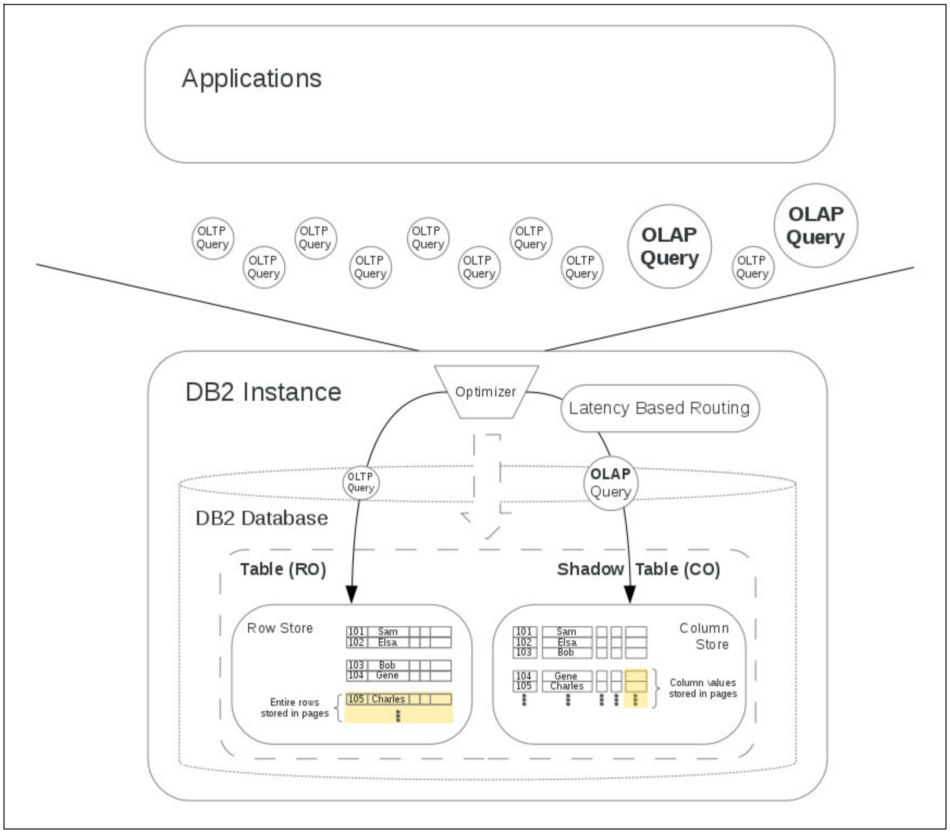



*Figure 3-2 Shadow tables overview*

From a usage perspective, these database changes are transparent to the application. Applications would continue to formally query the same row-organized table while not requiring any changes to their SQL statement text. When a query is submitted to the database engine, the DB2 optimizer chooses which execution environment to use based on the nature of the query, the estimated cost and the user-defined latency limits of the replication. When suitable, it will transparently reroute (and rewrite) the query to run against the shadow table, so it can use BLU Acceleration to heavily parallelize the execution. The end result is that queries, either OLAP or OLTP in nature, can always be run using an optimal execution environment within a single database.

Shadow table definitions include either all columns or just a subset of columns of their source tables. Replication from the source to the shadow tables is maintained through InfoSphere Data Replication Change Data Capture (InfoSphere CDC) Version 10.2.1 as shown in Figure 3-3. InfoSphere CDC captures changes to source tables by reading the database's transaction log stream. It then applies the changes asynchronously to the shadow table and if configured to do so, can feed latency information back to DB2. The DB2 optimizer can then utilize a user set time limit to base the routing decision on how current the data in the shadow tables is.

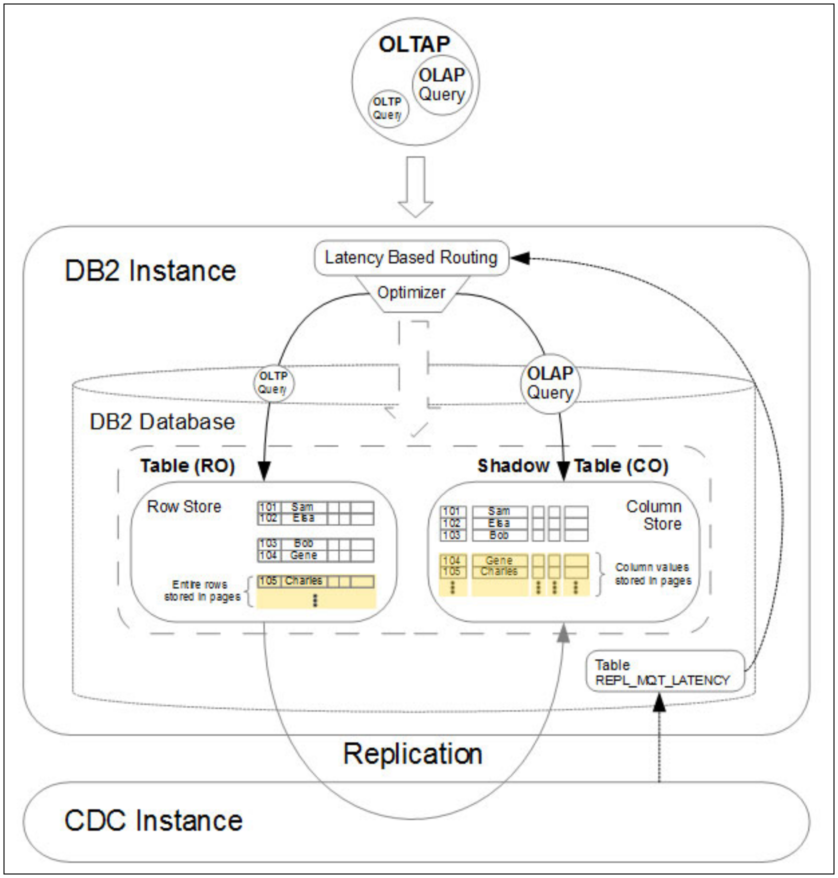



*Figure 3-3 Latency based routing and InfoSphere CDC Replication*

Figure 3-4 shows DB2 and the InfoSphere CDC components in more detail. This replication solution uses a single InfoSphere CDC instance and Datastore for the source and target side. The InfoSphere CDC Replication Engine for DB2 for Linux, UNIX, and Windows (DB2 for LUW) performs the database transaction log reads to capture changes on the source table and to apply them to the target table. The InfoSphere CDC Access Server maintains the access control metadata used by the Replication Engine, that is, database connection details, and InfoSphere CDC-specific user accounts used to secure the management access.

Replication can be managed through a command-line based tool CHCCLP that is part of the InfoSphere CDC Access Server or through a graphical front-end called the InfoSphere CDC Management Console. By using these tools tables can be subscribed for replication according to table mappings that define the source and target table associations.

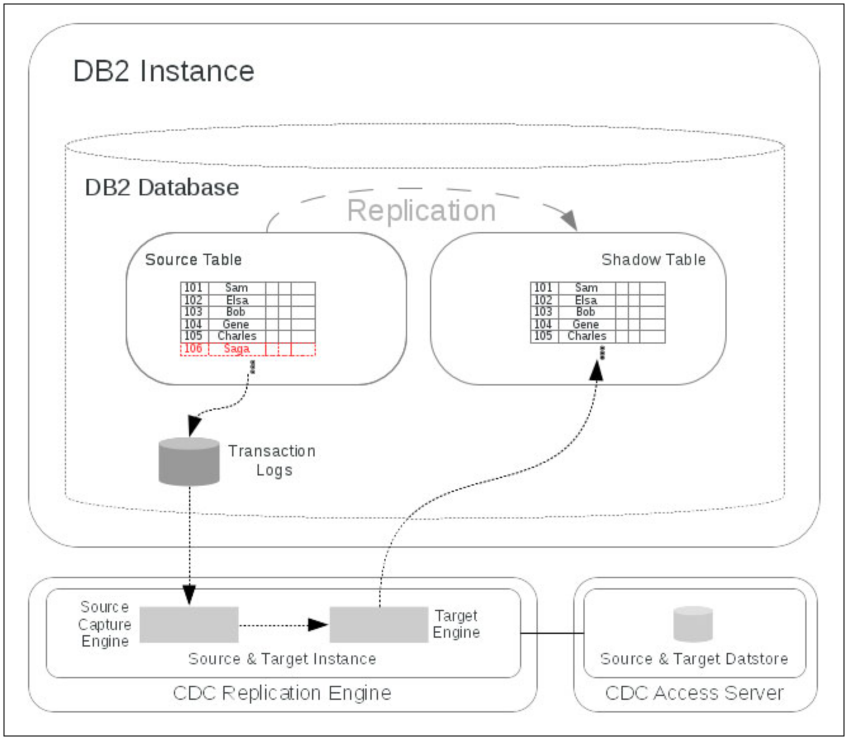



*Figure 3-4 DB2 and InfoSphere CDC Components*

## 3.2 Prerequisites

To use shadow tables in your environment, you must install and configure InfoSphere CDC Replication Engine, InfoSphere CDC Access Server to work with DB2 with BLU Acceleration. This section covers the necessary requirements for installation and information about the tasks that must be completed before using InfoSphere CDC for maintaining shadow tables. The user must complete all of the tasks before installing and configuring InfoSphere CDC.

The components for the replication solution can be installed in the same host or separate hosts. The topology suggested comprises these elements:

- The InfoSphere CDC Replication Engine and InfoSphere CDC Access Server installed on the same server as the DB2 instance.
- The InfoSphere CDC Management Console, a graphical replication management tool, installed on a dedicated Windows server or workstation.

### 3.2.1 Supported versions, platforms, and system requirements

The following list provides information about the supported platforms and software versions for the shadow tables feature:

- IBM DB2 for Linux, UNIX, and Windows (DB2 for LUW) Cancun Release 10.5.0.4
  - Supported platforms are listed in 2.4.1, “DB2 system requirements” on page 20.
- IBM InfoSphere Data Replication V10.2.1
  - Change Data Capture for DB2 for LUW with Interim Fix 12 or later releases
  - InfoSphere CDC Access Server V10.2.1 Interim Fix 5 and later releases
  - (Optional) InfoSphere CDC Management Console (Interim Fix 5 and later releases)

For system requirements for InfoSphere Data Replication V10.2.1, go to the following web address:

http://www-01.ibm.com/support/docview.wss?uid=swg27039669

### 3.2.2 User accounts and groups

Before installation, the user accounts listed in Table 3-1 are to be created. A separate operating system user is to be defined for the DB2 instance and the InfoSphere CDC installations.

*Table 3-1 User accounts and groups*

| Users | Description | Primary group (example) | Secondary groups (example) |
|---|---|---|---|
| db2inst1 | DB2 instance owner created during a DB2 instance creation. | Default SYSADM group (that is, the default db2iadm1) | - |
| cdcuser | InfoSphere CDC user that installs, runs and configures the InfoSphere CDC for DB2 LUW Replication Engine and the CDC Access Server.<br>This user must have access to the local database directory for the DB2 instance relevant for replication, for example, db2-instance-dir/db2-instance/NODE0000/sqldbdir.<br>To access DB2, this user requires to be authorized with DATAACCESS and either SYSADM or DBADM. | cdc-user-group | - |

## 3.2.3 Paths, storage space requirements, and permissions

Table 3-2 shows the suggested installation paths and space requirements for the shadow table feature.

*Table 3-2 Path, storage space, and permissions requirements*

| Description | Path Examples | Minimum Space Requirements | Permissions |
|---|---|---|---|
| Total space requirements for temporary installation files | | 3.5 GB | |
| DB2 Product Install Files (compressed + extracted) | | 3 GB | Read/Write to all users |
| InfoSphere CDC Access Server (compressed + extracted) | | 370 MB | |
| InfoSphere CDC Replication Engine (compressed + extracted) | | 370 MB | |
| | /tmp | 2 GB | DEFAULTS |
| | /var | 512 MB | DEFAULTS |

| Description | Path Examples | Minimum Space Requirements | Permissions |
|---|---|---|---|
| DB2 for LUW | | | |
| DB2 Product Installation | /opt/ibm/db2/v10.5 | 1.2 GB | DEFAULTS |
| DB2 Instance | /home/db2inst1 | 50 MB | DEFAULTS |
| DB2 Database path | /db2/db2instl1_db1 | variable[a] | Read/Write to DB2 SYSADM and readable to the cdc user |
| InfoSphere Change Data Capture | | | |
| InfoSphere CDC Replication Engine Installation | /home/cdc/opt/ReplicationEngine forIBMDB2 | 260 MB | DEFAULTS |
| InfoSphere CDC Access Server Installation | /home/cdc/opt/AccessServer | 360 MB | DEFAULTS |
| InfoSphere CDC Refresh Loader Path | For better I/O performance, specify a path on a separate file system. | variable[a] | Read/Write for DB2 instance owner and cdc user |

a. See 3.4, “Configuration preferred practices” on page 78.

### 3.2.4 Selecting candidates for shadow tables

You can use the Optim Query Workload Tuner, discussed in Chapter 4, “Optim Query Workload Tuner and BLU Acceleration” on page 137, to analyze existing queries and tables to determine which source tables would benefit from the shadow table feature.

## 3.3 Deployment

This section covers the installation and configuration process for setting up and using IBM InfoSphere CDC to maintain and administer shadow tables. We assume that all prerequisites have been met and that necessary user accounts and file paths are created as indicated in 3.2, “Prerequisites” on page 59.

### 3.3.1 Roadmap

The tasks involved in setting up and administering your shadow tables are outlined in the following list:

1. Install DB2 for LUW Cancun Release 10.5.04 as root user.
2. Create a DB2 instance.
3. Create and configure a database with the required parameters from Table 3-3 on page 65.
4. Install and configure InfoSphere CDC.
   a. Install InfoSphere CDC Access Server Version 10.2.1 with Interim Fix 5 or later.
   b. Install InfoSphere CDC for DB2 for LUW Version 10.2.1 with Interim Fix 12.
   c. (Optional) Install the InfoSphere CDC Management Console Version 10.2.1 with Interim Fix 5 or later.
   d. Create the InfoSphere CDC for DB2 LUW Replication Engine instance.
   e. Configure a data store.
5. Create shadow tables.
6. Configure and begin replication to shadow tables.
   a. Create an InfoSphere CDC subscription and table mappings.
   b. Start the InfoSphere CDC subscription and replicate data to a shadow table.
7. Enable query routing to shadow tables.
   a. Enable query routing within a DB2 command line processor (CLP) session.
   b. Enable query routing at the application level or using connect procedure.
8. Perform administration tasks.
   a. Monitor your InfoSphere CDC subscription.
   b. Tune the performance of your database system.

### 3.3.2 Installation

This section identifies the basic DB2 and InfoSphere CDC product installation documentation for the InfoSphere CDC for DB2 LUW (Replication Engine) and the InfoSphere CDC Access Server. It only describes the configuration. Make sure to use the versions shown in 3.2.1, “Supported versions, platforms, and system requirements” on page 59.

After the basic InfoSphere CDC product installation is complete, you can use Table 3-4 on page 68 as an overview of the most important configuration parameters, their descriptions, and suggested value examples. It is suggested to leave all other values at defaults.

As shown in 3.2, “Prerequisites” on page 59, it is suggested to install and run the InfoSphere CDC for DB2 LUW (Replication Engine) and the InfoSphere CDC Access Server on the same host, sharing the same user for both installations. Suggestions for installation paths with required minimum permissions and space requirements can be taken from Table 3-2 on page 60.

### DB2 for LUW

For details about how to install DB2 Cancun Release 10.5.0.4, see the IBM Knowledge Center at the following web addresses:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.qb.server.doc/doc/c0024080.html

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.1.0/com.ibm.db2.luw.qb.server.doc/doc/c0059810.html?lang=en

### InfoSphere CDC Access Server

Refer to the following documentation to learn how to install the InfoSphere CDC Access Server Version 10.2.1 Interim Fix 5 or later:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.installingasandmc.doc/concepts/installingaccessserver.html?lang=en

After installed the Access Server, process must be started using the **dmaccessserver** command. Its documentation can be found at this web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.mcadminguide.doc/refs/dmaccessserver.html?lang=en

On UNIX and Linux environments, start the Access Server as background process and independent of the terminal as shown in Example 3-1 on page 63.

*Example 3-1 Starting the InfoSphere CDC Access Server on UNIX and Linux*

```
# Run these system commands as CDC user, that is, cdcuser

cd [...]InfoSphereChangeDataCapture/AccessServer/bin/
nohup ./dmaccessserver &
```

### InfoSphere CDC Replication Engine

The InfoSphere CDC product installation procedures for the InfoSphere CDC Replication Engine for DB2 LUW are explained in the official InfoSphere CDC documentation, found at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/concepts/installingorupgradinginfospherecdc.html?lang=en

Ensure that the InfoSphere CDC for DB2 LUW base version 10.2.1 is installed prior to installing the Interim Fix 12 or later.

> **Note:** When installation is complete, do not start the InfoSphere CDC configuration tool.

### InfoSphere CDC Management Console

The InfoSphere CDC instance can be managed using the command-line interface CHCCLP directly from the InfoSphere CDC Access Server. Alternatively, there is the option to use the graphical InfoSphere CDC Management Console on the Windows platform. To install the Management Console, consult the installation documentation, found at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.installingasandmc.doc/concepts/installingmanagementconsole.html?lang=en

### Database and InfoSphere CDC Instance

There is a single DB2 instance to CDC instance setup used for shadow tables. The shadow table feature is designed to operate within a single database and replicate data using a single InfoSphere CDC instance for all shadow tables within this database. Before configuring the InfoSphere CDC instance, the database must be configured to enable it for shadow tables. This is detailed in 3.3.3, “Database and database manager configuration for shadow tables” on page 64.

## 3.3.3 Database and database manager configuration for shadow tables

In a DB2 database that uses shadow tables, the database is configured assuming that the predominant workload is OLTP in nature with changes made to optimize it for shadow tables. Before the creation of shadow tables, you must ensure that the database is configured for their use.

Some of the prerequisite configuration parameters and registry variables that must be in effect are listed in Table 3-3. See 3.4.2, “DB2 configuration parameters” on page 79 for more details.

*Table 3-3 Summary of required DB2 configuration parameters*

| Attribute | Value | Description |
|---|---|---|
| Code set | UTF-8 | Support codepage and collating sequence for databases with column-organized tables. |
| Collating Sequence | IDENTITY<br>IDENTITY_16BIT | |
| DB2_WORKLOAD | *NOT* set to ANALYTICS | In an OLTAP environment with shadow tables, a predominant workload is still assumed to be OLTP. |
| “Intra-Partition Parallelism” | Application level | Enabling intra-partition parallelism on the application level is a preferred method for shadow table environments. |
| sortheap | *Fixed* value only | Column-organized data processing requires large values for sortheap and sheapthres_shr (set to some fraction of database_memory). |
| sheapthres_shr | | |
| instance_memory | *AUTOMATIC* | Size the memory allocation of the DB2 instance to about 75-95% of the main memory available to the operating system. |
| util_heap_sz | *AUTOMATIC* with high base value | Large value to support the LOAD command. |
| logarchmeth1 | LOGRETAIN or other archiving method | Primary log archive method is required for InfoSphere CDC replication. |
| pagesize | 32K | Retain default page size for database, and set table space level page size to 32K for table space containing column-organized tables |

**Note:** Sizing guidelines for memory parameters are mentioned in 3.4.2, “DB2 configuration parameters” on page 79.

As discussed in Table 3-3 on page 65, the values for memory parameters must be larger than those that are typically used in OLTP configurations. The mixed OLTAP environment must be tailored for both row-organized and column-organized data. Although a larger `sheapthres_shr` value is expected for any system that uses column-organized tables, a larger `sortheap` value could potentially cause access plan changes for existing OLTP queries. If you want to avoid such changes, you can override the `sortheap` value by specifying the `OPT_SORTHEAP_EXCEPT_COL` token and a value for the `DB2_EXTENDED_OPTIMIZATION` registry variable (Example 3-2). This override value only affects compiler query optimization for generating plans that do not reference any column-organized table and will not dictate the amount of actual memory that is available at run time.

*Example 3-2 Set sortheap value for row-organized tables only*

```
# Run this DB2 system command as DB2 instance user

db2set DB2_EXTENDED_OPTIMIZATION="OPT_SORTHEAP_EXCEPT_COL 10000"
```

**Note:** Before setting appropriate values for the `util_heap_sz`, `sheapthres_shr`, and `sortheap` database configuration parameters, record the original values in case you want to revert back to them later.

## 3.3.4 Shadow table creation

To create a shadow table, the source table candidate must be selected. The process of selecting an appropriate source table is discussed in Chapter 4, “Optim Query Workload Tuner and BLU Acceleration” on page 137.

When creating a shadow table, the source table can be replicated entirely to its column-organized counterpart or you can choose to only replicate a subset of columns that are relevant to your query workload. Example 3-3 on page 66 shows the DDL for a row-organized table named ROWT on which a shadow table is created.

**Note**: The source table must have a primary key constraint or unique constraint whose column set must be included in the select list for the shadow table.

*Example 3-3 DDL of source (row-organized) table*

```
-- Execute this SQL statement from DB2 CLP or Data Studio as DB2 user
```

```
create table rowt(
  c1 integer not null,
c2 integer,
  data char(254)
) organize by row;
alter table rowt
  add constraint rowt_pk primary key(c1);
```

To create a shadow table, issue the CREATE TABLE statement with the MAINTAINED BY REPLICATION clause. This clause identifies the table as a shadow table. Example 3-4 shows the CREATE TABLE statement.

*Example 3-4 Creating table*

```
-- Execute this SQL statement from DB2 CLP or Data Studio as DB2 user

create table rowt_shadow as
   (select * from rowt)
data initially deferred
refresh deferred
enable query optimization
maintained by replication
organize by column
```

Upon creation, the new shadow table is in the *set integrity pending* state. Issue the SET INTEGRITY statement, specifying the IMMEDIATE UNCHECKED clause, to bring the shadow table out of the pending state, as shown in Example 3-5.

*Example 3-5 Bring shadow table out of pending state*

```
-- Execute this SQL statement from DB2 CLP or Data Studio as DB2 user

set integrity for rowt_shadow
   all immediate unchecked
```

A primary key that matches the enforced primary or unique constraint on the source table (in both column list and key sequence) is required on the newly created shadow table. A one-to-one mapping is required for each row in both tables. Issue the ALTER TABLE statement to define a primary key on the shadow table, as shown in Example 3-6 on page 67.

*Example 3-6 Add primary key constraint on shadow table*

```
-- Execute this SQL statement from DB2 CLP or Data Studio
```

```
alter table rowt_shadow
   add constraint rowt_shadow_pk primary key (c1);
```

### 3.3.5 Replication setup

All changes made on the source tables can be replicated to shadow tables through the use of the InfoSphere Change Data Capture product. InfoSphere CDC uses the InfoSphere CDC for DB2 for LUW Replication Engine component to capture changes made to the tables in the source database and to apply them to the tables in the target database. In this solution, the source and target database are one-and-the-same.

The additional InfoSphere CDC Access Server component stores and manages the datastore, which holds metadata representing the connection details that are needed by the InfoSphere CDC Replication Engine to connect to the actual database. The InfoSphere CDC Access Server also manages InfoSphere CDC-specific credentials used to authenticate users that manage the InfoSphere CDC replication through either the graphical InfoSphere CDC Management Console or the command-line driven CHCCLP utility that is built into the Access Server.

After the InfoSphere CDC Access Server (AS) and the InfoSphere CDC Replication Engine (RE) components are installed, you can use the RE command **dmconfigurets** to add and start a new instance of the InfoSphere CDC Replication Engine and configure the parameters from Table 3-4. The command is in the following directory:

`[...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/bin/`

For information about how to invoke this command, see the InfoSphere CDC documentation at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/tasks/addanewinstance_windows.html?lang=en

*Table 3-4 Examples of most relevant InfoSphere CDC instance configuration parameters*

| Parameter | Example | Description |
|---|---|---|
| Name | cdcinst1 | This is a name of the instance you want to create. |
| Staging Store Disk Quota (GB) | variable | Staging storage on the source. |
| DB2 instance | db2inst1 | Contains the database with the tables that are to be replicated. |

| Parameter | Example | Description |
|---|---|---|
| Database Name | oltpdb | This is the name of the database that you want to replicate data to or from and contains all of the tables for replication. |
| Username | db2inst1 | This is a user name for the specified database; use the same user name that you used while adding an Access Server connection. |
| Password | <password to db2inst1> | This is a password for the user specified in Username. |
| Metadata schema | db2inst1 | This is a schema name used by the InfoSphere CDC for DB2 for metadata table; typically the same as the Username. |
| Refresh loader path | /db2/db2inst1_db1/cdc_refresh_loader | The path "cdc_refresh_loader" is used later for a staging location for table refreshes with InfoSphere CDC data replication. |

As the next step, configure the newly created instance with the following InfoSphere CDC instance parameters. The settings shown in Example 3-7 are explained in 3.4.3, "InfoSphere CDC instance configuration parameters" on page 85. The command **dmset** is used to set these parameters. The documentation for this command can be found at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/refs/dmset.html?lang=en

*Example 3-7 Setting InfoSphere CDC instance parameters*

```
# Execute these system commands as CDC user, that is, cdcuser
cd [...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/bin/
./dmset -I cdcinst1
acceptable_latency_in_seconds_for_column_organized_tables=5
./dmset -I cdcinst1 maintain_replication_mqt_latency_table=true
./dmset -I mirror_auto_restart_interval_minutes=2
```

After configuring, start the CDC instance by using the **dmts64** command. On UNIX and Linux platforms, we suggest to use the **nohup** command to have this process continue running in the background without a terminal attachment. For more details about these two commands, see the following web addresses:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/refs/dmts32.html?lang=en
or
http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/refs/dmts64.html?lang=en

Example 3-8 shows how to start the instance with the **dmts64** command.

*Example 3-8 Starting the CDC instance*

```
# Execute these system commands as CDC user, that is, cdcuser

cd [...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/bin/
nohup ./dmts64 -I cdcinst1 &
```

The suggested configuration used in Example 3-7 on page 69 implies the use of the latency based routing algorithm that is part of the Shadow Table feature. This configuration parameter instructs InfoSphere CDC to propagate the replication's latency details to DB2 by writing it to a database table, `SYSTOOLS.REPL_MQT_LATENCY`, which must be created manually. The DB2 Optimizer uses this information to decide when data in the shadow table is current enough to reroute queries from the source to the shadow table. Example 3-9 shows how to create this table in a custom table space "TBSP4K" using a SQL statement to DB2 system stored procedure call.

*Example 3-9 Creating the SYSTOOLS.REPL_MQT_LATENCY table*

```
-- Execute this SQL statement from DB2 CLP or Data Studio

CALL SYSPROC.SYSINSTALLOBJECTS('REPL_MQT', 'C', 'TBSP4K', CAST (NULL
AS VARCHAR(128))) ;
```

**Note**: Specify the same table space that contains the InfoSphere CDC TS_BOOKMARK table so that these tables can be backed up, restored, and rolled forward together.

Within the InfoSphere CDC Access Server, you must create a single datastore that identifies the replication engine to the Access Server. You can then create an Access Server admin ID and configure a datastore connection which grants access for the datastore, previously created, to this admin ID. The Access Server admin ID controls who can replicate what data. Table 3-5 lists the suggested parameters.

*Table 3-5 Access Server parameters*

| Parameter | Example value | Description |
|---|---|---|
| Admin user creation using the **dmcreateuser** command | | |
| Username | admin | Name of the user |
| Fullname | N/A | Full name for the user |

| Parameter | Example value | Description |
|---|---|---|
| Description | N/A | Description for the user |
| Password | **`<See Note below>`** | Password for the user |
| Role | sysadmin | One of SYSADMIN, ADMIN, OPERATOR or MONITOR (case insensitive) |
| Manage | true | User has access manager privileges (TRUE/FALSE, case insensitive) |
| ChangePassword | false | Password must be changed on first login (TRUE/FALSE, case insensitive) |
| PasswordNeverExpires | false | Password never expires (TRUE/FALSE, case insensitive) |
| Datastore creation using the **`dmcreatedatastore`** command | | |
| DatastoreName | ds1 | Name of the datastore |
| Description | db2inst1 OLTPDB Shadow Tables | Description for the user |
| Hostname | **`<Hostname or IP Address of the database server>`** | Host name of the datastore Example uses "prod01" |
| Port | 10901 | Port of the datastore |
| Connection mapping using the **`dmaddconnection`** command | | |
| DatastoreName | ds1 | Name of the datastore |
| UserName | db2inst1 | Name of the user |
| Database | OLTPDB | Database name |
| DatabaseOwner | db2inst1 | Database owner name |
| DatabasePassword | **`<See Note below>`** | Database password |
| alwaysPrompt | false | Always prompts with the connection dialog at login (TRUE/FALSE, case insensitive) |

| Parameter | Example value | Description |
|---|---|---|
| showParams | true | Displays the connection parameters in the connection dialog (TRUE/FALSE, case insensitive) |
| writeProtected | false | Makes the connection parameters write protected (TRUE/FALSE, case insensitive) |
| saveParams | true | Saves the connection parameters input in the connection dialog (TRUE/FALSE, case insensitive) |

**Note**: Ensure the password is at least six characters long and contains at least two alphabetic characters and at least two non-alphabetic characters to avoid an error later when trying to connect to the Access Server.

Run the commands in Example 3-10 to create an admin ID using the parameter values suggested in Table 3-5 on page 70. The commands are accessible from the `bin` subfolder of your InfoSphere CDC Access Server installation.

*Example 3-10 Configuring Access Server*

```
# Execute these system commands as CDC user, that is, cdcuser

cd [...]InfoSphereChangeDataCapture/AccessServer/bin/
dmcreateuser admin "N/A" "N/A" p@ssw0rd sysadmin true false false
dmcreatedatastore ds1 "db2inst1 OLTPDB Shadow Tables" prod01 10901
dmaddconnection admin ds1 oltpdb db2inst1 pa$$w0rds false true false
true
```

## 3.3.6 Subscriptions and table mappings

InfoSphere CDC uses subscriptions to logically group source and target table mappings. With the shadow tables feature, all table mappings must be placed in a single subscription. Subscriptions can be created using the graphical Management Console or interactively and non-interactively using the command-line driven CHCCLP utility.

Example 3-11 on page 73 shows the interactive use of the CHCCLP utility to create a subscription “sub1” in our previously created datastore “ds1”. In the example, we also marked the new subscription as persistent.

Persistent subscriptions are automatically restarted when the replication engine is restarted after shut down or after the subscription has ended with a recoverable error.

*Example 3-11 Creating a InfoSphere CDC subscription*

```
# Execute these system commands as CDC user, that is, as cdcuser

cd [...]InfoSphereChangeDataCapture/AccessServer/bin/
./chcclp
   set verbose;
   connect server hostname prod01 port 10101 username admin password
p@ssw0rd;
   connect datastore name ds1;
   add subscription name sub1 persistency true;
   disconnect datastore name ds1;
   disconnect server;
   exit;
```

**Note:** Make sure to only create one synchronization point by creating a single subscription for all shadow tables in the same database.

For each source and target (shadow) table mapping, you can use the command in Example 3-12 to map the source table to the shadow table.

*Example 3-12 Creating a InfoSphere CDC Table Mapping*

```
# Execute these system commands as CDC user, that is cdcuser

cd [...]InfoSphereChangeDataCapture/AccessServer/bin/
./chcclp
   set verbose;
   connect server hostname prod01 port 10101 username admin password
p@ssw0rd;
   connect datastore name ds1;
   select subscription name sub1;
   add table mapping sourceSchema BASE sourceTable ROWT targetSchema
BASE targetTable ROWT_SHADOW targetIndexMode index targetIndexName
BASE.ROWT_SHADOW_PK type standard method mirror;
   start mirroring method continuous;
   disconnect datastore name ds1;
   disconnect server;
   exit;
```

## 3.3.7 InfoSphere CDC event notification setup

As an operational preferred practice, we suggest to set up event notifications as part of the shadow table deployment to monitor replication on an on-ongoing basis. It is imperative to have any communication regarding interruptions in replication to be passed on to the database administrator as there are no CDC warnings or messages to be expected in DB2 diagnostic logs. In the event that replication is disrupted, queries continue to run against the source table, but might not route to shadow tables. This can occur when the last replication update applied to the shadow table is older than the user-defined latency as described in "Query routing to shadow tables" on page 81, causing queries to run against source tables without using BLU Acceleration.

To set up InfoSphere CDC event notifications, complete the following steps:

1. Determine the SMTP server name by consulting the network administrator.
2. Log in to the InfoSphere CDC Management Console as the CDC Access Server admin user.
3. Stop mirroring and end replication on the subscription for your shadow tables.

   InfoSphere CDC does not allow any configuration changes while a subscription is active.

   a. Click **Monitoring** → **Subscriptions**.

   b. Right-click the subscription, select **End Replication** → **Normal**, and click **OK**.
4. Set email notifications at the datastore level:

   a. Click **Configuration** and navigate to **Datastores** on the bottom left pane.

   b. Right-click the **ds1** data store for your shadow tables and select **Notifications**.

   c. Click the **Source** tab.

   d. Within Notifications-ds1, select the first category from Notification Categories, for which you want to enable notifications. We select **Scrape/Refresh** → **Error** as shown in Figure 3-5 on page 75.

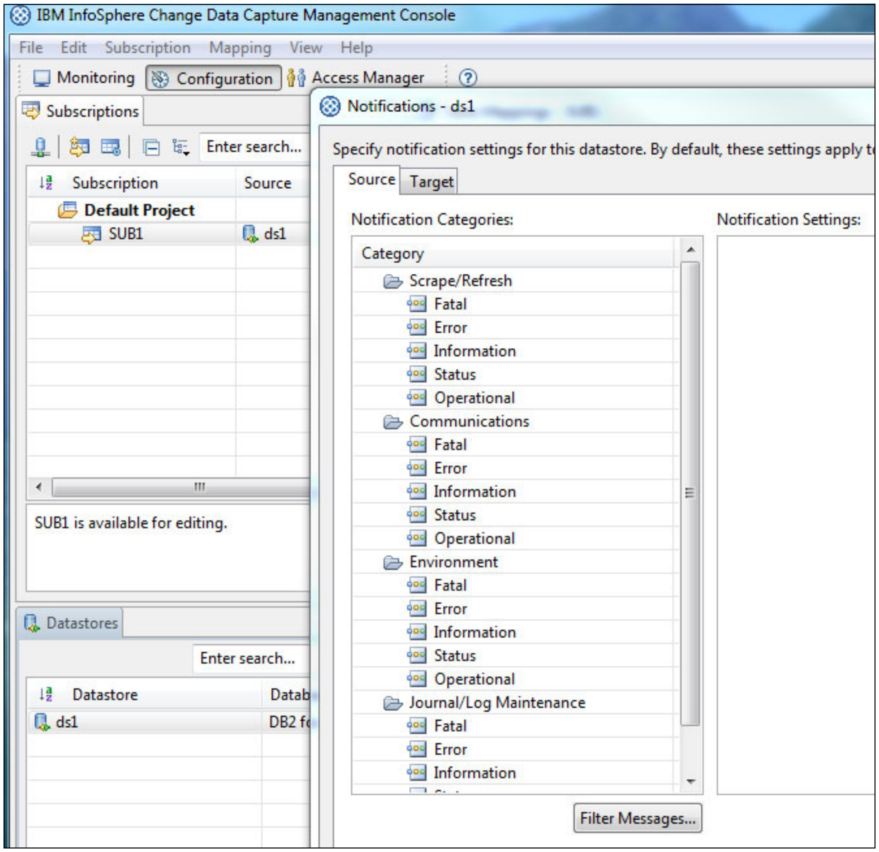



*Figure 3-5 Set up events notifications for ds1*

e. Select the **INTERNET MAIL** box, and enter the following information:

- SMTP Mail Host: Enter the name of the outgoing mail server on the host name field.
- Alert Account: Enter the email addresses or distribution list that you want to notify.
- Sender Mail Address: Enter the email address.
- Sender Mail Password: Enter the password of the sender.

f. Click **Copy Setting** to copy the notification settings to all other notification categories that apply.

g. As the source and the target are the same data store for shadow tables, scroll down to the Target section to copy notification settings to its notification categories.

h. Select the **Apply** → **Information** notification category to copy the notification settings and click **OK** to enable notifications for the subscription

i. Click **OK** to the notifications window and return to the main Configuration view.

j. Set the latency threshold notification by right-clicking the **SUB1** subscription in the subscriptions sub-pane and select **Latency Thresholds**, as shown in Figure 3-6.

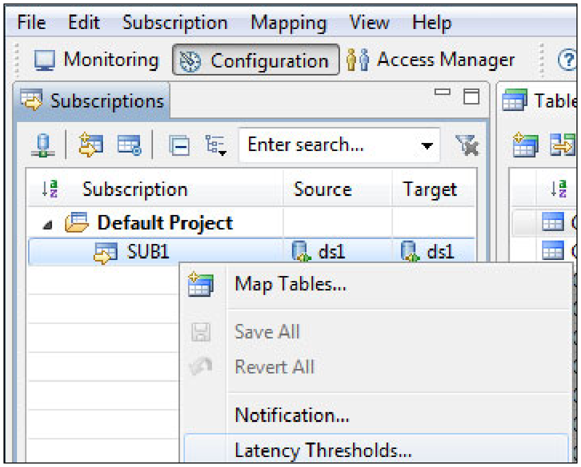



*Figure 3-6 Setting latency threshold notifications*

i. Check **Notify when latency crosses these thresholds**.

ii. Specify a warning threshold in minutes. A good value is 75% of the value of the CURRENT REFRESH AGE special registry.

iii. Specify a problem threshold in minutes. The value must be greater than the warning threshold. A good value is 90% of the CURRENT REFRESH AGE.

iv. Click **OK**. Do not click Set Notification, because you specified all the required notification settings in step 4 on page 74.

5. Set event filtering to disable event notification for events that are not important for monitoring. Enabling Target Informational events to capture latency threshold notifications returns notifications for event IDs that you might not be interested in:

   a. Click **Configuration** → **Subscriptions**.

   b. Right-click the subscription and select **Notifications**.

   c. In the Notification Categories window, click the **Target** tab.

d. Click **Datastore defaults** → **Filter Messages**. The event ID filtering applies to all the categories of the target datastore.

e. In the list, enter the event IDs that you do not want to be notified of (one event on each line), and select the **Do not send these messages** option.

f. Filter the same event IDs for the source datastore, because it is the same as the target datastore.

6. Start mirroring to start receiving event notifications by email:

   a. Click **Configuration** → **Subscriptions**.

   b. Right-click the subscription name and select **Start Mirroring**.

   c. In the Start Mirroring window, specify the mirroring method by selecting **Continuous**, and click **OK**.

For more details about setting up event notifications and event listings, see the CDC documentation at the following web address:

`http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdc doc.mcadminguide.doc/concepts/settingUpNotifications.html?lang=en`

### 3.3.8 Connection setup

DB2 features a general purpose mechanism that can be used to automate the configuration of those connections intended to submit analytical queries that benefit from shadow tables. It allows a user-defined stored procedure to be registered with the database configuration parameter `CONNECT_PROC`, so that each time a database connection is established the database, the stored procedure is run automatically within the connection's first transaction.

The stored procedure can be designed to identify a connection based on custom attributes, for example, the connection-configured application name. It can also prepare those connections for BLU Acceleration and configure parameters important to shadow tables. This typically includes enabling per-connection intra-partition parallelism and parameters that fine-tune the automatic query routing to shadow tables.

Example 3-13 on page 78 demonstrates such a stored procedure based on a set application name "bizreport". See "Query routing to shadow tables" on page 81 for more details.

*Example 3-13 CONNECT_PROC stored procedure*

```
create or replace procedure DBGUEST1.REPL_MQT_SETUP()
begin
  declare applname varchar(128);

  set applname = (SELECT APPLICATION_NAME FROM
          TABLE(MON_GET_CONNECTION(mon_get_application_handle(), -1)));

  if (applname = 'bizreport') then
    call sysproc.admin_set_intra_parallel('yes');
    set current degree 'ANY';
    set current maintained types replication;
    set current refresh age 500;
  end if;
end@
```

All users IDs that are in need of running the “bizreport” workload must be granted the EXECUTE privilege. Example 3-14 shows the SQL statements run through the **db2** command to deploy the stored procedure from Example 3-13 on page 78 after storing its text into a file called `connect_proc.ddl` and to grant the EXECUTE privilege to the public.

*Example 3-14 Deploying the stored procedure*

```
# Execute these system commands as DB2 user with DBADM authority

db2 td@ -v -f connect_proc.ddl
db2 “grant execute on procedure DBGUEST1.REPL_MQT_SETUP to public”
```

### 3.3.9 Setup validation

For setup validation, use the steps described in “Checking query routing to shadow tables” on page 106 and “Troubleshooting query routing using EXPLAIN” on page 107.

## 3.4 Configuration preferred practices

This section discusses DB2 and InfoSphere CDC configuration options, their meanings, and how they relate to each other. It gives insight into DB2 and InfoSphere CDC memory sizing and tuning, storage considerations, the operation of latency based query routing, and the start and shut down procedures.

## 3.4.1 DB2 and InfoSphere CDC sizing guidelines

We suggest to start the sizing of physical memory and the number of processor cores based on the current needs of the OLTP part of your workload. With the shadow table feature, we suggest to start with a minimum of 3-4 GB additional main memory for InfoSphere CDC. Also consider additional main memory for DB2 shared memory as needed for queries using BLU Acceleration against shadow tables based on the data volume, query complexity, and concurrency.

For the DB2 with BLU Acceleration preferred practices, see Chapter 2, " Planning and deployment of BLU Acceleration" on page 17 and the DB2 for LUW best practices document "*Optimizing analytic workloads using DB2 10.5 with BLU Acceleration*", found at the following web address:

https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/0fc2f498-7b3e-4285-8881-2b6c0490ceb9/page/ecbdd0a5-58d2-4166-8cd5-5f186c82c222/attachment/625dbbda-afce-40a3-b079-af3541a93d74/media/DB2BP_BLU_Acceleration_0514.pdf

InfoSphere CDC performance tuning is discussed in the InfoSphere documentation, found at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.performancetuning.doc/concepts/serverperformancetuning.html?lang=en

## 3.4.2 DB2 configuration parameters

The following DB2 configuration parameters are important to tune your mixed workload environment for the shadow tables feature.

### Database creation

A general requirement to use DB2 with BLU Acceleration is that the database must support automatic storage, the `UNICODE` code set and `IDENTITY` (or `IDENTITY_16BIT`) collation. All these settings are implicit default values for databases created in DB2 10.5.

### Intra-partition parallelism

Intra-partition parallelism is vital to enable BLU Acceleration. This form of parallelism allows DB2 to run a suitable query much faster by splitting into multiple parts and by running these parts in parallel by using multiple threads. In a mixed workload environment such as the shadow table feature that is designed to support OLTP and OLAP workloads at the same time, it should only be enabled on a connection basis to not negatively affect queries that are not suitable for this execution environment. We strongly suggest to use the mechanism represented by the `CONNECT_PROC` database configuration parameter and the `SYSPROC.ADMIN_SET_INTRA_PARALLEL` procedure to enable intra-partition parallelism as explained in 3.3.8, "Connection setup" on page 77.

### *SYSPROC.ADMIN_SET_INTRA_PARALLEL*

This DB2 built-in stored procedure can enable (through argument value "yes") and disable (through argument value "no") the intra-partition parallelism for the current connection as shown in Example 3-15. In the shadow table context, we advise to use the method described in 3.3.8, "Connection setup" on page 77 to enable this form of parallelism automatically and transparent to the application code when possible.

*Example 3-15 SQL statement to enable Intra-Partition Parallelism using the stored procedure SYSPROC.ADMIN_SET_INTRA_PARALLEL*

```
call sysproc.admin_set_intra_parallel('yes');
```

### *DB2_WORKLOAD, INTRA_PARALLEL*

Use the DB2 registry parameter `DB2_WORKLOAD` only with a value *other than* `ANALYTICS` to support the OLTP portion of the workload running against your database. When setting up a database tuned for an analytics workload environment using the BLU-Acceleration technologies, this registry parameter is often set to a value of `ANALYTICS`. However, among other settings, this would enable intra-partition parallelism for all connections by default, even those that run OLTP-type queries that might not be suitable for this execution environment. For the same reason, we suggest against enabling the instance wide database manager configuration parameter (DBM CFG) `INTRA_PARALLEL`.

See "SYSPROC.ADMIN_SET_INTRA_PARALLEL" on page 80 for more details.

### *MAX_QUERYDEGREE, DFT_DEGREE, and DEGREE*

The degree of intra-partition parallelism can be controlled through the DBM CFG parameters `MAX_QUERYDEGREE`, database configuration (DB CFG) parameter `DFT_DEGREE`, and the precompile option `DEGREE`. We suggest to use the default values, so that DB2 can choose a suitable level of parallelism.

## General parameters

This section includes the general parameters that are important to tune in your mixed workload environment for the shadow tables feature.

### *LOGARCHMETH1*

The InfoSphere Change Data Capture component replicates data from transactions that it captures from the database's transaction log. To replicate information from non-active logs, the database's transaction logs must be at least retained. Set the `LOGARCHMETH1` DB CFG parameter to a value of `LOGRETAIN` or other (DB2) supported log retention methods, such as DISK, TSM, and so on.

### *CONNECT_PROC*

This DB CFG parameter allows a user-defined stored procedure to be registered and triggers the database to call that stored procedure on an incoming database connection. As shown in 3.3.8, "Connection setup" on page 77, this function can be used to automatically and transparent to the application configure specific connections to enable the use of shadow tables.

### *DFT_TABLE_ORG*

With the main focus on OLTP-type queries, we suggest to set this DB CFG parameter to the default of row organization. This means all `CREATE TABLE` statements that are not explicitly specifying the `ORGANIZE BY` keyword will implicitly create row-organized tables.

### *Page size*

We suggest to use 32 K page size for shadow tables, just as for regular column-organized tables. Traditional row-organized tables can be sized as usual. There is no clear default page size that would be useful to set the DB CFG parameter `PAGESIZE` to. Create the database with the most common page size required for your tables.

### *Extent size*

A good default for extent size for shadow and column-organized tables is a value of 4 pages. Otherwise, the general DB2 best practices can be followed to tune this parameter. As with the page size, there is no clear all round value in a mixed workload environment that could be used to set for the DB CFG parameter `DFT_EXTENT_SZ`, to indicate an implicit, default extent size.

## Query routing to shadow tables

Example 3-16 shows the sample SQL statements required to enable latency based query routing to shadow tables. The following sections discuss the individual parameters included in these statements.

*Example 3-16 SQL statements enable query routing to shadow tables*

```
call sysproc.admin_set_intra_parallel('yes');
set current degree 'ANY';
set current maintained types replication ;
set current refresh age 500 ;
```

### *CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION*

This special register defines the types of tables that can be considered by the optimizer when processing dynamic SQL queries. Make sure to only include the value `REPLICATION`.

### *CURRENT REFRESH AGE <value>*

This special register defines the acceptable age of replicated data that allows the shadow tables with this data to still be considered for query execution. Use a value other than 0 to enable query routing. A value of `ANY` ignores latency information and always route the shadow table. The value you choose should be made based on your business requirements on how current the data needs to be.

This parameter specifies a time stamp duration value with a data type of `DECIMAL(20,6)`. The value must be 0, 99 999 999 999 999 (ANY), or a valid time stamp duration between that range. The valid format for the range is yyyymmddhhmmss.nnnnnn,

where:

- yyyy is the number of years and can have values in the range of 0 - 9999 (inclusive)
- mm is the number of months and can have values in the range of 0 - 11 (inclusive)
- dd is the number of days and can have values in the range of 0 - 30 (inclusive)
- hh is the number of hours and can have values in the range of 0 - 23 (inclusive)
- mm is the number of minutes and can have values in the range of 0 - 59 (inclusive)
- ss is the number of seconds and can have values in the range of 0 - 59 (inclusive)
- nnnnnn is the number of fractional seconds (the fractional seconds portion of the value is ignored and therefore can be any value)

The leading zeros for the entire value and the trailing fractional seconds do not need to be included. However, individual elements that have another element to the left must include the zeros such as using 500 to represent 5 minutes, and 0 seconds.

### *CURRENT QUERY OPTIMIZATION <value>*

Set a query optimization value of 2, 5-9. The default value is 5.

## Memory parameters

This section describe the memory related configuration parameters that are important to tune in your mixed workload environment for the shadow tables feature.

### *DB2_EXTENDED_OPTIMIZATION*

The processing of analytical workloads uses more operations to benefit from increased sort memory heaps as queries are more complex and often join and aggregate more tables. From a memory tuning perspective you can distinguish between the sort memory available to transactional workloads running against row-organized tables and analytical queries using column-organized tables. The sort memory heap considered by the SQL compiler and optimizer for column-organized tables can be set through overall DB CFG parameter `SORTHEAP`. By adding the text `OPT_SORTHEAP_EXCEPT_COL`=<value> to the `DB2_EXTENDED_OPTIMIZATION` registry variable, the value specified here limits the amount of sort heap memory for queries not running against column-organized tables.

### *INSTANCE_MEMORY*

We suggest to use the `AUTOMATIC` setting for this DBM CFG configuration parameter because it sizes the memory allocation of the DB2 instance to about 75-95% of the main memory available to the operating system.

### *DATABASE_MEMORY*

This DB CFG parameter can be left at the default `AUTOMATIC` setting. While setting it to automatic, you might want to consider setting an initially size value at the same time that is big enough to hold all memory areas.

A good starting point to divide the overall database memory into the biggest memory areas is as follows:

- Use about 40-50% of the total available database memory (`DATABASE_MEMORY`) for the total amount of shared memory (`SHEAPTHRESH_SHR`)
  - As a starting point, assign 5-20% of the total amount of shared sort memory (`SHEAPTHRES_SHR`) to the sort heap available to each SORT operation (`SORTHEAP`)
  - With an increasing number of connections, you might benefit from reducing the `SORTHEAP` parameter to allow more SORT operations succeed
  - Increase `SORTHEAP` with increasing query complexity when SORT operations are spilling to disk
- Use about 40% of the remaining database memory for buffer pools
  - Associated 10% of that memory to buffer pools associated to table spaces storing column-organized tables only.
  - The remaining 90% should be assigned to buffer pools associated with table spaces storing traditional tables.

### *SHEAPTHRES*

You can disable private memory sorts to restrict the total amount of sort memory available to the value specified by the `SHEAPTHRES_SHR` DBM CFG parameter.

### *SHEAPTHRES_SHR*

This DB CFG parameter controls the total amount of sort memory available to all SORT operations. See “DATABASE_MEMORY” on page 83 about how to size this parameter.

### *SORTHEAP*

The amount of sort memory available to each SORT operation can be set with this DB CFG parameter. For sizing guidelines, see “DATABASE_MEMORY” on page 83.

### *DB_MEM_THRESH*

We suggest to configure DB2 to not release any unused database shared memory by setting the `DB_MEM_THRESH` DB CFG parameter to a value of 100.

### *CATALOGCACHE_SZ*

DB2 maintains a synopsis table with each column-organized table to allow data ranges to be easily skipped. This requires lookups on synopsis tables that are faster when this data is kept within the catalog cache. Compared to a purely OLTP-type database we suggest to increase this memory DB CFG parameter by about 10% to keep more of this data in cache.

### *UTIL_HEAP_SZ (DB CFG)*

In the context of the Shadow Table feature this parameter affects data LOAD operations as they are used for replicating data to shadow tables by InfoSphere CDC. The suggestion is to set to AUTOMATIC with a starting value of at least 1 million (4 K pages). Starting with 128 GB of main memory use a value of 4 million (4 K pages). Increase the DB CFG parameter `UTIL_HEAP_SZ` while adding more shadow tables as there will be more concurrent LOAD operations requiring this memory heap.

## Default query concurrency management

DB2 has a built-in concurrency control as part of its Workload Management feature that can protect the system to be overloaded from fewer long running queries by prioritizing the more frequent short running queries. A default query concurrency mechanism that is part of the `SYSDEFAULTMANAGEDSUBCLASS` service subclass can be put in place by enabling the `SYSDEFAULTCONCURRENT` threshold using the statement shown in Example 3-17.

*Example 3-17 SQL statement to enable the default query concurrency management*

```
alter threshold SYSDEFAULTCONCURRENT enable ;
```

### Automatic maintenance

This section describes the automatic maintenance related configuration parameters that are important to tune in your mixed workload environment for the shadow tables feature.

#### *AUTO_RUNSTATS (DB CFG)*

By default, automatic maintenance is enabled, which includes automatic runtime statistics collection. If you prefer to perform this task manually, we suggest to specify the `WITH DISTRIBUTION AND SAMPLED DETAILED INDEX ALL` clause with the **`RUNSTATS`** command. This will collect both basic statistics and distribution statistics on columns, and extended statistics on all table indexes.

#### *AUTO_REORG (DB CFG)*

This DB CFG parameter can enable automatic table and index reorganization for a database, by default **`AUTO_REORG`** is not enabled. To manually perform these operations specify the **`RECLAIM EXTENTS`** clause with the **`REORG`** command on indexes and tables to reclaim extents that are not being used. Use the value of the `RECLAIMABLE_SPACE` column that is returned by a query to the `ADMINTABINFO` administrative view or the `ADMIN_GET_TAB_INFO` table function. This value is the amount of free space that was determined the last time that you ran the **`RUNSTATS`** command.

## 3.4.3 InfoSphere CDC instance configuration parameters

The following parameters can be set using the **`dmset`** command as shown in Example 3-7 on page 69.

### `acceptable_latency_in_seconds_for_column_organized_tables`

This value indicates the intended latency in seconds to InfoSphere CDC when replicating to column-organized shadow tables. This value should be less than the value of the `CURRENT REFRESH AGE` special register.

### `maintain_replication_mqt_latency_table`

When set to a value of true, InfoSphere CDC attempts to maintain latency information in a table `SYSTOOLS.REPL_MQT_LATENCY`. The table must be first created through a DB2 stored procedure, shown in Example .

*Example 3-18 SQL statement to create the DB2 table holding replication latency information*

```
CALL SYSPROC.SYSINSTALLOBJECTS('REPL_MQT', 'C', CAST (NULL AS
VARCHAR(128)),CAST (NULL AS VARCHAR(128))) ;
```

**mirror_auto_restart_interval_minutes**

You can define the interval in minutes before InfoSphere CDC attempts to automatically restart continuous mirroring for persistent subscriptions. See Example 3-7 on page 69 for details.

## 3.4.4 Startup and shutdown procedures

The startup and shutdown of the components used in this solution should occur in a specific order to allow the replication from source to shadow tables to properly start and stop.

Before starting the InfoSphere CDC components, make sure the DB2 instance managing the database that contains the tables mapped for replication is started. Then start the InfoSphere CDC Access Server followed by the InfoSphere CDC Replication Engine instance as shown in Example 3-19.

*Example 3-19 Console output showing a sample DB2 and InfoSphere CDC startup procedure*

```
[martin@imde01 ~]$ su - db2inst1
[db2inst1@imde01 ~]$ db2start
04/28/2014 17:26:04 0 0 SQL1063N DB2START processing was successful.
SQL1063N DB2START processing was successful.
[db2inst1@imde01 ~]$ db2 activate db MyDB
DB20000I  The ACTIVATE DATABASE command completed successfully.
[db2inst1@imde01 ~]$ db2 connect to MyDB && db2 unquiesce db
   Database Connection Information

 Database server        = DB2/LINUXX8664 10.5.4
 SQL authorization ID   = DB2INST1
 Local database alias   = MyDB

DB20000I  The UNQUIESCE DATABASE command completed successfully.

[db2inst1@imde01 ~]$ su - cdcuser
[cdcuser@imde01 ~]$ nohup [...]/AccessServer/bin/dmaccessserver &
[cdcuser@imde01 ~]$ nohup [...]/ReplicationEngineforIBMDB2/bin/dmts64
-I cdcinst1&
```

As part of the shutting down procedure, first stop replication, then shut down the InfoSphere CDC, and finally deactivate the database.

Example 3-20 shows the execution of the **dmendreplication** command while logged in as InfoSphere CDC user to stop all (-A) replication for the instance (-I) “cdcinst1” employing the default method (-c). Similarly, the instance is shut down using the **dmshutdown** command. Finally, after logged in as DB2 instance owner, you can **quiesce** and **deactivate** the database.

*Example 3-20 Console output showing a sample InfoSphere CDC and DB2 shutdown procedure*

```
[db2inst1@imde01 ~]$ su - cdcuser

[cdcuser@imde01 ~]$
[...]/ReplicationEngineforIBMDB2/bin/dmendreplication -I cdcinst1 -c -A

[cdcuser@imde01 ~]$ [...]/ReplicationEngineforIBMDB2/bin/dmshutdown -I
cdcinst1

[cdcuser@imde01 ~]$ [...]/AccessServer/bin/dmshutdownserver

[cdcuser@imde01 ~]$ su - db2inst1
[db2inst1@imde01 ~]$ db2 connect to MyDB && db2 quiesce db immediate

   Database Connection Information

 Database server        = DB2/LINUXX8664 10.5.4
 SQL authorization ID   = DB2INST1
 Local database alias   = MyDB

DB20000I  The QUIESCE DATABASE command completed successfully.
[db2inst1@imde01 ~]$ db2 terminate
DB20000I  The TERMINATE command completed successfully.
[db2inst1@imde01 ~]$ db2 deactivate db MyDB
DB20000I  The DEACTIVATE DATABASE command completed successfully.
```

**Note:** Depending on the size of the CDC pending-job queue, time to complete the **dmendreplication** command can vary. A useful command to add immediately after **dmendreplication** is **dmshowevents** to check the list of pending jobs:

```
dmshowevents -I cdcinst1 -a -c 10
```

### 3.4.5 Buffer pool and storage considerations

We suggest to structure row and column-organized tables in separate buffer pools, table spaces, and storage groups associated with different I/O subsystems for better management and performance due to different I/O access patterns. Example 3-21 shows a full set of DB2 statements that can be used to implement this setup. The end result is a set of buffer pools and table spaces in various page sizes from 4 K-32 K to store row-organized tables typically used for OLTP workloads and a 32 K buffer pool table space storing column-organized tables for column-organized tables.

*Example 3-21 SQL statements to create dedicated storage groups, table spaces, and buffer pools*

```
CREATE BUFFERPOOL bp4k SIZE AUTOMATIC PAGESIZE 4 K ;
CREATE BUFFERPOOL bp8k SIZE AUTOMATIC PAGESIZE 8 K ;
CREATE BUFFERPOOL bp16k SIZE AUTOMATIC PAGESIZE 16 K ;
CREATE BUFFERPOOL bp32k SIZE AUTOMATIC PAGESIZE 32 K ;
CREATE BUFFERPOOL bp32kcol SIZE <~4% of SHEAPTHRES_SHR> AUTOMATIC
PAGESIZE 32 K ;
CREATE STOGROUP row_sg ON '/db2/instv105_oltpdb/datafs_row1' ;
CREATE STOGROUP col_sg ON '/db2/instv105_oltpdb/datafs_col1' ;
CREATE TABLESPACE tbsp4k PAGESIZE 4k MANAGED BY AUTOMATIC STORAGE USING
STOGROUP row_sg BUFFERPOOL bp4k ;
CREATE TABLESPACE tbsp8k PAGESIZE 8k MANAGED BY AUTOMATIC STORAGE USING
STOGROUP row_sg BUFFERPOOL bp8k ;
CREATE TABLESPACE tbsp16k PAGESIZE 16k MANAGED BY AUTOMATIC STORAGE
USING STOGROUP row_sg BUFFERPOOL bp16k ;
CREATE TABLESPACE tbsp32k PAGESIZE 32k MANAGED BY AUTOMATIC STORAGE
USING STOGROUP row_sg BUFFERPOOL bp32k ;
CREATE TABLESPACE tbsp32kcol PAGESIZE 32k MANAGED BY AUTOMATIC STORAGE
USING STOGROUP col_sg BUFFERPOOL bp32kcol ;
```

The example uses a method to store table spaces in different storage groups based on their use of either storing row-organized or column-organized tables. This way, row-organized tables can be stored in the table spaces `tbsp4k`, `tbsp8k`, `tbsp16k`, and `tbsp32k` that are using the I/O subsystem associated with the storage group row_sg.

The additional table space called `tbsp32kcol` can be used with column-organized tables only and is part of the storage group col_sg. For best performance, the file system paths used for the storage groups row_sg and col_sg should correspond with file systems (mount points) on different I/O subsystems. When needed, additional storage paths can be separately added to each storage group later to extend its available space. By doing so DB2 can automatically stripe table spaces across the multiple paths.

Note that buffer pools are also divided to allow for better tuning in a mixed OLTAP workload environment. Just like the table spaces and storage groups, we suggest to also use dedicated buffer pools for column-organized tables.

The size of buffer pools associated with table spaces storing specifically column-organized tables should be increased compared to typical transactional sizes as BLU Acceleration technologies can considerably benefit from keeping as much pages in memory as possible. We suggest to assign a total of about 40% of the database shared memory (`SHEAPTHRES_SHR`) to buffer pools in general while using 10% of that memory for buffer pools dedicated to column-organized tables and the rest of all other buffer pools (for row-organized tables).

### 3.4.6 Enabling latency based routing

The routing (rewrite) of queries by the DB2 optimizer allows a query formally written to access the row-organized table to be run transparently against a shadow table under specific conditions.

The routing decision is generally based on replication latency. It compares the user-defined latency with the actual replication latency information supplied to DB2 by InfoSphere Change Data Capture. The optimizer takes the user-defined latency from the value stored in the `CURRENT REFRESH AGE` special register while it can read the actual replication latency information from the table `SYSTOOLS.REPL_MQT_LATENCY`, which is maintained by InfoSphere CDC.

If the actual latency is below the user-defined latency, the optimizer considers the data stored in the shadow table to be current enough compared to its source table. In this case, the statement text of the query gets rewritten to use the name of the shadow table instead of its source table. Due to the column organization of the shadow table, the query can be optimized to be run by the Column Data Engine that incorporates the BLU Acceleration optimizations.

This entire routing mechanism depends on multiple conditions and parameters:

- A shadow table must be in place, which specifically is a column-organized materialized query table that is maintained by (external) replication and enabled for query optimization.
- Replication through InfoSphere CDC must be in continuous mirroring mode, which implies, that there is a subscription in place that maps the source table to a shadow table. InfoSphere CDC might be instructed to maintain the replication latency table (InfoSphere CDC instance parameter `maintain_replication_mqt_latency_table`).

- The optimizer must be directed to only consider tables maintained by replication. This can be done explicitly through the special register `CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION` or implicitly through the database configuration parameter `DFT_MTTB_TYPES`.
- The optimizer requires a user-defined latency limit defined through the special register `CURRENT REFRESH AGE`.
- The optimizer must be advised of a valid query optimization level (levels 2 or 5-9) through the special register `CURRENT QUERY OPTIMIZATION`.
- Intra-partition parallelism needs to be enabled for the connection submitting the query (using stored procedure `SYSPROC.ADMIN_SET_INTRA_PARALLEL`). The value for the special register `CURRENT DEGREE` specifying the degree of intra-partition parallelism should be set to `ANY` to allow the database manager to determine a suitable value.

> **Note:** Businesses that can tolerate latency can use shadow tables without having to record and rely on latency information within DB2. They do not need to use the REPL_MQT_LATENCY table. In that case, queries set the DEFAULT REFRESH AGE to “NY” and the DB2 Optimizer does not take latency into consideration.

Example 3-22 lists sample statements to enable a connection for latency based query routing. These statements can be either set manually or automatically as described in 3.3.8, “Connection setup” on page 77.

*Example 3-22 SQL statements to enable connections for latency based query routing*

```
CALL SYSPROC.ADMIN_SET_INTRA_PARALLEL('YES') ;
SET CURRENT DEGREE 'ANY' ;
SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION REPLICATION ;
SET CURRENT REFRESH AGE 500 ;
```

### Enabling latency-based routing in Cognos

Users that use Cognos BI as their analytic reporting application can optionally enable routing to shadow tables, on the application level. Cognos BI allows users to pass on database commands to be run as it opens a connection session to a data source by using command blocks.

The following example demonstrates how to define the Cognos command block to enable Cognos connection sessions for latency based query routing to shadow tables.

1. Launch **IBM Cognos Administration** studio.
2. Click the **Configuration** tab and select **Data Source Connections** on the left menu.
3. A list of data source connections are listed. Click the **Set Properties** icon ( ) beside the data source connection that you want latency based query routing enabled, as shown in Figure 3-7.

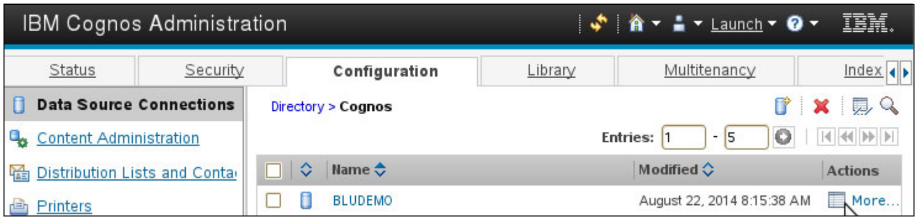



*Figure 3-7 Set Properties icon for a DB2 data source connection*

4. In the Set Properties panel, click the **Connection** tab. A list of database events that can have a Cognos XML command block specified is displayed. Set the **Open session commands** by clicking **Set...** or **Edit...** beside it, as shown in Figure 3-8.

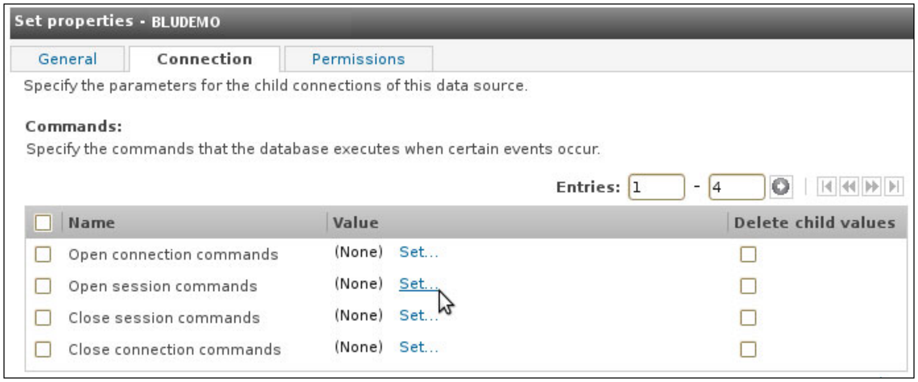



*Figure 3-8 Set open session commands for a Cognos data source connection*

5. Cognos command blocks are formatted in XML. Example 3-23 on page 92 shows a sample command block syntax to enable latency based routing every time Cognos opens a database session to the DB2 data source.

*Example 3-23 Cognos command block example to enable latency based routing*

```
<commandBlock>
     <commands>
          <sqlCommand>
               <sql>
                    CALL ADMIN_SET_INTRA_PARALLEL('YES');
               </sql>
          </sqlCommand>
          <sqlCommand>
               <sql>
                    SET CURRENT DEGREE 'ANY'
               </sql>
          </sqlCommand>
          <sqlCommand>
               <sql>
                    SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION
REPLICATION
               </sql>
          </sqlCommand>
          <sqlCommand>
               <sql>
                    SET CURRENT REFRESH AGE 500
               </sql>
          </sqlCommand>
     </commands>
</commandBlock>
```

6. Click **OK** to leave the open session commands window and return to the connection properties.
7. Click **OK** to leave the connection Set Properties panel and return to IBM Cognos Administration.

With the sample Cognos open session command block demonstrated in this section, latency based routing to shadow tables is set every time Cognos opens a database session to the database.

## 3.4.7 Optimizing compression dictionaries

Shadow tables are fundamentally column-organized tables that, due to their implementation, always apply data compression. Columnar table compression works by creating dictionaries on a column level and on a page level when data is loaded or inserted into the table. During a table load operation, all column values are being analyzed for frequently occurring patterns from which the column-level dictionaries are built. For column-level compression dictionaries to be optimized, make sure there is a larger number of rows being initially loaded with data that is representative enough to optimize the column-level compression dictionary.

In the context of shadow tables, we suggest that it can be beneficial for the row-organized source table to already contain a large row set to initially load the shadow table with this set of rows to optimize the compression dictionary. This can happen either manually through a LOAD operation followed by a `SET INTEGRITY ... ALL IMMEDIATE UNCHECKED` command or by directly setting up and starting the InfoSphere CDC replication that uses load operations against the shadow tables.

The load operation employed by InfoSphere CDC uses the fast load apply mode to perform a refresh operation on a target table. InfoSphere CDC first lines up a number of rows defined by the `fastload_refresh_commit_after_max_operations` system parameter and then loads this data using bulk loads into the shadow table using the db2Load API.

The suggestion is to size the `fastload_refresh_commit_after_max_operations` to the largest source table row count or the maximum supported value for the system parameter (2147483647) so that all that data is used to refresh your shadow table in a single bulk load operation. When the row count of the source table is larger than the parameter value, InfoSphere CDC has to split the operation into multiple committed bulk loads, where only the initial load operation would contribute to the quality of the column-level compression dictionary.

In Example 3-24, we set the mentioned parameter to 10 million rows.

*Example 3-24 Modifying the FASTLOAD_REFRESH_COMMIT_AFTER_MAX_OPERATIONS system parameter*

```
# Execute this system command as CDC user, i.e. cdcuser

cd [...]/ReplicationEngineforIBMDB2/bin/
dmset -I cdcinst1 fastload_refresh_commit_after_max_operations=10000000
```

## 3.4.8 Latency and throughput

InfoSphere CDC replication requires changes in the source tables to be asynchronously applied to their respective shadow tables. When shadowing multiple row-organized source tables in your environment, the change apply process can be a performance bottleneck. Fast apply is an InfoSphere CDC product feature that optimizes throughput and reducing latency.

Fast apply provides four different algorithms (or modes) to choose from that are appropriate for varying replication needs. In this section, we talk about two modes. For more information about each fast apply mode and how each is appropriate for your needs, refer to the InfoSphere CDC documentation.

To improve the performance of your shadow tables, it is suggested that you enable the **Group By Table** fast apply mode for your subscription. In this mode, InfoSphere CDC will reorder a set of operations creating lists of operations for each table, and then attempt to apply them to the target tables. The following list outlines the steps to enable fast apply using the InfoSphere CDC Management Console:

1. Log in to the InfoSphere CDC Management Console, select **Configuration** → **Subscriptions**, right-click the subscription, and select **End Replication**.
2. Select **Normal** and click **OK**.
3. Right-click the subscription and select **User Exits**. A user exit lets you define a set of actions that InfoSphere CDC can run before or after a database event occurs on a specified table.
4. Specify **`com.datamirror.ts.target.publication.userexit.fastapply.GroupByTable`** as the class name and **`10000`** as the parameter. The value signifies the number of transactions grouped together as a unit of work.
5. Select **Configuration** → **Subscriptions**, right- click the subscription, and select **Start Mirroring**.
6. Select **Continuous** and click **OK**.

If your environment still experiences latency issues, you can change the fast apply mode to **Parallelize by table.** It is similar to **Group by table**, but the reordered operations are applied concurrently using multiple database connections. With this mode, it is important to monitor the replication activity of tables to determine parallel connections that do not impede performance. When enabling this mode using the InfoSphere CDC Management console, you have to select the number of database connections and the threshold value as shown in Example 3-25.

*Example 3-25 Parallelize by table, set with 3 connections and 10000 transactions*

```
class name
com.datamirror.ts.target.publication.userexit.fastapply.ParallelizeByTable
Parameter
3:10000
```

**Note:** Using fast apply to improve performance will often result in an increased resource footprint of InfoSphere CDC in form of processor and memory. You should keep this in mind when deciding if this feature is appropriate for your environment. The suggested minimum memory allocated for the InfoSphere CDC instance is 3.5 GB.

## 3.5 Operational preferred practices

Using shadow tables in an OLTAP environment leads to certain changes in how database operations are carried out. This section covers certain guidelines and suggestions when performing regular administration tasks such as database backup, restore, reorganization, monitoring, and redesign.

### 3.5.1 Backup and recovery

When using the shadow table feature, the source tables and their column-organized counterparts should be backed up together. To ensure that complete backups are created, perform a backup on the full set of table spaces that contain the source-shadow table pairs and their indexes. It is optional to include the table space that contains the InfoSphere CDC metadata tables in the backup image, although it is strongly suggested that it be included for correct recovery from unintended corruption to the table space.

As part of the backup and recovery strategy, it is preferred to back up the InfoSphere CDC replication configuration data for your CDC instance. The replication configuration data is stored in a metadata database, as files within the `cdc-installation-dir`. This data differs from the metadata tables that are stored in the DB2 database. You should always back up the metadata after the initial setup of the subscription for shadow tables and when there are changes to your subscription such as adding or dropping a table mapping.

Example 3-26 shows the **dmbackupmd** command, which creates the backup within the InfoSphere CDC installation directory under `[...]/cdc-installation-dir/cdc-replication-engine-files/instance/cdc-instance/conf/backup`.

*Example 3-26 CDC backup command*

```
cd [...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/bin/
./dmbackupmd -I cdcinst1
```

With the replication configuration data backed up, we should back up all the table spaces that contain the source-shadow pair and their relevant information. To back up all the necessary table spaces, first verify the required table spaces by running the following query on the system catalog, as shown in Example 3-27 on page 96.

*Example 3-27 SQL statement to list all table spaces containing shadow tables*

```
WITH TABLELIST as (
SELECT TABSCHEMA, TABNAME
FROM SYSCAT.TABLES
WHERE SUBSTR(PROPERTY,23,1) = 'Y'
OR TABNAME IN ('TS_AUTH', 'TS_BOOKMARK', 'TS_CONFAUD',
'TS_DDLAUD')
UNION SELECT BSCHEMA as TABSCHEMA, BNAME as TABNAME
FROM SYSCAT.TABLES, SYSCAT.TABDEP
WHERE SUBSTR(SYSCAT.TABLES.PROPERTY,23,1) = 'Y'
AND SYSCAT.TABLES.TABNAME = SYSCAT.TABDEP.TABNAME
AND SYSCAT.TABLES.TABSCHEMA = SYSCAT.TABLES.TABSCHEMA
)
SELECT substr(SYSCAT.TABLES.TABSCHEMA,1,30) as TABSCHEMA,
substr(SYSCAT.TABLES.TABNAME,1,30) as TABNAME,
substr(TBSPACE,1,30) as TBSPACE,
substr(INDEX_TBSPACE,1,30) as INDEX_TBSPACE,
substr(LONG_TBSPACE,1,30) as LONG_TBSPACE
FROM SYSCAT.TABLES, TABLELIST
WHERE SYSCAT.TABLES.TABNAME = TABLELIST.TABNAME
AND SYSCAT.TABLES.TABSCHEMA =
TABLELIST.TABSCHEMA;
```

Use the **BACKUP DATABASE** DB2 command to make sure that the backup includes the full set of table spaces that contain the tables and indexes associated with your source-shadow table pair. Table spaces corresponding to one or more source-shadow pairs can be backed up individually or together in one image.

**Note:** it is important that the table spaces for a source table and its associated shadow table are in the same backup image.

Example 3-28 shows the **BACKUP DATABASE** command for a single source-shadow table pair when the source table t_row is in a SOURCE table space, and the shadow table t_shadow is in a SHADOW table space along with the REPL table space holding all InfoSphere CDC metadata tables.

*Example 3-28 DB2 statement to back up source-shadow table pairs through table spaces level backups*

```
BACKUP DATABASE oltpdb TABLESPACE (SOURCE, SHADOW, REPL) ONLINE ;
```

## 3.5.2 Restoring table space backups for shadow tables

Table spaces that contain the source-shadow pair must always be restored together, although they can be restored independently of other source-shadow table pairs. When you restore table spaces, use a backup image that contains the full set of table spaces that are involved in the source-shadow pairing.

If the source table space, alone, is restored to a previous point in time, it can cause the shadow table to be placed in a drop-pending state. Else, if the table space that contains the shadow table is independently restored, the shadow table is placed in an integrity-pending state.

**Note:** As mentioned earlier, the backup and recovery of the table space containing the InfoSphere CDC metadata tables is optional and only required in cases of table space corruption, such as disk failure. If you recover the table space that contains the metadata tables, you must flag all source tables for refresh before you start mirroring.

To restore the table spaces for your source-shadow pairing, InfoSphere CDC replication must be stopped. Example 3-29 shows the command that stops replication for a subscription that is named SUB1 in an InfoSphere CDC instance named cdcinst1.This command has to be issued as the InfoSphere CDC user.

*Example 3-29 Stopping replication of a specific subscription*

```
# Execute these system commands as CDC user, i.e. cdcuser

cd [...]/ReplicationEngineforIBMDB2/bin/
dmendreplication -I cdcinst1 -s SUB1
```

Next, you ar required to reset the values in the **`SYSTOOLS.REPL_MQT_LATENCY`** table within the database `oltpdb`, by running the command shown in Example 3-30. This prevents inaccurate latency information from being communicated to DB2 from the latency table.

*Example 3-30 SQL statement to reset values in repl_mqt_latency*

```
UPDATE SYSTOOLS.REPL_MQT_LATENCY SET (COMMIT_POINT,DELAY_OFFSET) = (0,0)
```

Now we are ready to restore the table spaces for our source-shadow pairs. Example 3-31 describes the **`RESTORE DATABASE`** command run when the source table t_row is in a SOURCE table space, shadow table t_shadow is in a SHADOW table space. After the restore is complete, the table spaces will be in a rollforward pending state.

*Example 3-31 Restore table spaces*

```
RESTORE DATABASE oltpdb TABLESPACE (SOURCE, SHADOW) ONLINE
```

To take the table spaces out of the rollforward pending state, you will run the ROLLFORWARD DATABASE command. It is possible to roll forward to a specific point in time or to the end of logs. Example 3-32 takes the table spaces out of the rollforward pending state without replaying recent changes from the log files.

*Example 3-32 Rollforward database to end of backup*

```
ROLLFORWARD DATABASE oltpdb TO END OF BACKUP AND STOP TABLESPACE
(SOURCE, SHADOW, REPL) ONLINE
```

> **Note:** Unless rollforward was to end of logs, all restored table spaces are now in a backup-pending state and are not accessible until a new backup image is captured.

With the table spaces restored, if necessary, the InfoSphere CDC replication configuration can be restored, next. The InfoSphere CDC replication configuration should be restored to a point in time as close as possible to that of the restored table spaces. Restoring from this backup ensures that the table descriptions that are stored in the replication configuration data match the table information in the DB2 system catalog. The InfoSphere CDC metadata that was backed up, as part of the backup process, is restored within the InfoSphere CDC instance. Example 3-33 copies the files that are associated with restore point *t1* into the instance directory for the instance *cdcinst1*.

*Example 3-33 Restore metadata content for InfoSphere CDC instance*

```
cp
[...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/instance/c
dcinst1/conf/backup/t1/*
[...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/instance/c
dcinst1/conf/
```

When you restore the InfoSphere CDC replication configuration, the current instance configuration is overwritten by the version that is captured by the **dmbackupmd** backup command.

> **Note:** It is not mandatory to restore the replication configuration of your InfoSphere CDC instance. However, when restoring the shadow tables to a point in time in the past, it is imperative that you restore the replication configuration data from a backup that was made at that point in time.

If the CDC metadata tables were restored, we need to flag the newly restored source tables for refresh. Example 3-34 flags the source table DTW.T_ROW in a subscription that is named SUB1 for the InfoSphere CDC instance cdcinst1 for refresh.This command is issued as the InfoSphere CDC user.

*Example 3-34 Flag the source tables for refresh*

```
# Execute this system command as CDC user, that is, cdcuser

cd [...]/ReplicationEngineforIBMDB2/bin/
dmflagforrefresh -I cdcinst1 -s SUB1 -t DTW.T_ROW
```

To complete the restore process, restart the InfoSphere CDC replication by using the **dmstartmirror** command. Example 3-35 shows how to restart the subscription SUB1.

*Example 3-35 Restart replication for subscription SUB1*

```
# Execute this system command as CDC user, that is, cdcuser

dmstartmirror -I cdcinst1 -s SUB1
```

> **Note:** For Windows users, they can stop/restart the subscription and flag tables for refresh using the InfoSphere CDC Management Console, instead of using the command line.

### 3.5.3 Moving InfoSphere CDC metadata tables to a specific table space

If you want to move the InfoSphere CDC metadata tables to a specific table space of your choice, use the ADMIN_MOVE_TABLE procedure. Example 3-36 shows an example of a query to display the table space where the metadata tables were created.

*Example 3-36 SQL statement to query to display the table space of the metadata tables*

```
SELECT TBSPACE
FROM SYSCAT.TABLESPACES
WHERE TBSPACEID IN (
SELECT TBSPACEID
FROM SYSCAT.TABLES
WHERE TABNAME LIKE ÊTS_%Ê
AND TABSCHEMA = db2-metadata-schema-name);
```

Move the metadata tables with the ADMIN_MOVE_TABLE procedure to a specific table space of your choice. Example 3-37 shows how to move the TS_AUTH metadata table to the SHADOW table space.

*Example 3-37 SQL statement to move meta-data tables to table space SHADOW*

```
CALL ADMIN_MOVE_TABLE('db2-metadata-schema-name', 'TS_AUTH',
'SHADOW','SHADOW','SHADOW','','','','','','MOVE');
```

## 3.5.4 Reorganizing shadow tables

Similar to regular column-organized tables, it is suggested to enable automatic reorganization for shadow tables. If automatic reorganization is not enabled, then the `REORG` command with the `RECLAIM EXTENTS` option must be run to reorganize shadow tables. Example 3-38 shows how to manually reorganize the T_SHADOW table. It is good practice to update statistics after reorganizing your tables, as shown under 3.5.5, "Table statistics".

*Example 3-38 Manually reorganizing the T_SHADOW table*

```
REORG TABLE DTW.T_SHADOW RECLAIM EXTENTS
```

## 3.5.5 Table statistics

Up-to-date table statistics, including shadow table statistics are vital for query performance. The optimizer needs accurate statistics to make well-informed decisions when determining execution plans. It is advisable to keep the `auto_runstats` database configuration parameter enabled for up-to-date database statistics.

If the parameter is not enabled, you must manually perform `RUNSTATS` on shadow tables after they are initially populated with data and periodically thereafter. shows an example of collecting statistics on a shadow table with the appropriate options.

*Example 3-39 SQL statement to update table statistics for shadow table*

```
RUNSTATS ON TABLE t_shadow WITH DISTRIBUTION ON ALL COLUMNS AND
DETAILED INDEXES ALL;
```

## 3.5.6 Monitoring replication

As shadow tables use InfoSphere CDC components for replication, you will want to monitor and analyze the replication activities to ensure that data is consistent across your source and shadow table. You can use the InfoSphere CDC Management Console or the command line CHCCLP to better understand the various characteristics of your replication environment and diagnose potential problems.

The Management Console supports the monitoring of replication operations and latencies, the collection of various statistics about source and target data stores, and event detection and notification.

### Monitoring replication using CHCCLP

The InfoSphere CDC user can use CHCCLP to monitor the status of the InfoSphere CDC subscriptions and collect information that can assist in resolving any issues that might arise with the replication setup. Example 3-40 shows the commands to launch CHCCLP. CHCCLP is launched from the `bin` folder within the InfoSphere CDC access server installation folder.

*Example 3-40 Run CHCCLP for monitoring InfoSphere CDC replication*

```
# Execute these system commands as CDC user, i.e. cdcuser

cd [...]InfoSphereChangeDataCapture/AccessServer/bin/
./chcclp
   chcclp session set to cdc;
   set verbose;
[CDC1004]: Verbose output enabled.
   connect server hostname linux-host port 10101 username admin password
p@ssw0rd;
[CDC1101]: Connected to Access Server linux-host@10101.
   connect datastore name ds1;
[CDC1201]: Connected to datastore ds1.
   select subscription name SUB1;
[CDC1308]: Current subscription context set to SUB1 from datastore ds1
to datastore ds1.
```

Table 3-6 provides descriptions of some important commands that can be run using CHCCLP. If a name is not provided with commands, the subscription that is currently identified as the context will be used.

*Table 3-6 Commands to check replication state using CHCCLP*

| Command | Description |
|---|---|
| `monitor replication` | Monitors the replication state and latency of subscriptions. |
| `monitor subscription activity` | Shows activity metrics for current subscription since mirroring started. |
| `monitor subscription latency` | Retrieves the latency information for current subscription e.g latency threshold, latency in seconds. |
| `show latency thresholds` | Shows the latency threshold for the subscription, if it is enabled. It is the value for how long it takes InfoSphere CDC to apply data to the target datastore. |
| `monitor subscription busy tables name SUB1` | Shows statistics around the busiest tables within the named subscription. Shows number of bytes replicated, and percentage of activity for each table. |
| `show subscription event details` | Displays the full text of truncated messages and event details for the subscription. |
| `list subscription events count [1..100..]` | Lists the events of the shadow table subscription. |

## Monitoring using InfoSphere CDC Management Console

You can use the InfoSphere CDC Management Console to monitor your InfoSphere CDC subscription. It can provide information about events and system messages that relate to the subscription for your shadow tables. The Management Console is composed of a number of windows and tabs that are referred to as perspectives and views.

To monitor the InfoSphere CDC subscription, log in to the InfoSphere CDC Management Console with the InfoSphere CDC Access Server user. After logging in, you see three perspectives, the Configuration perspective, the Monitoring perspective, and the Access Manager perspective, from which you can access a number of different views depending on your role and access level.

Select **Monitoring** → **Subscriptions** as shown in Figure 3-9.

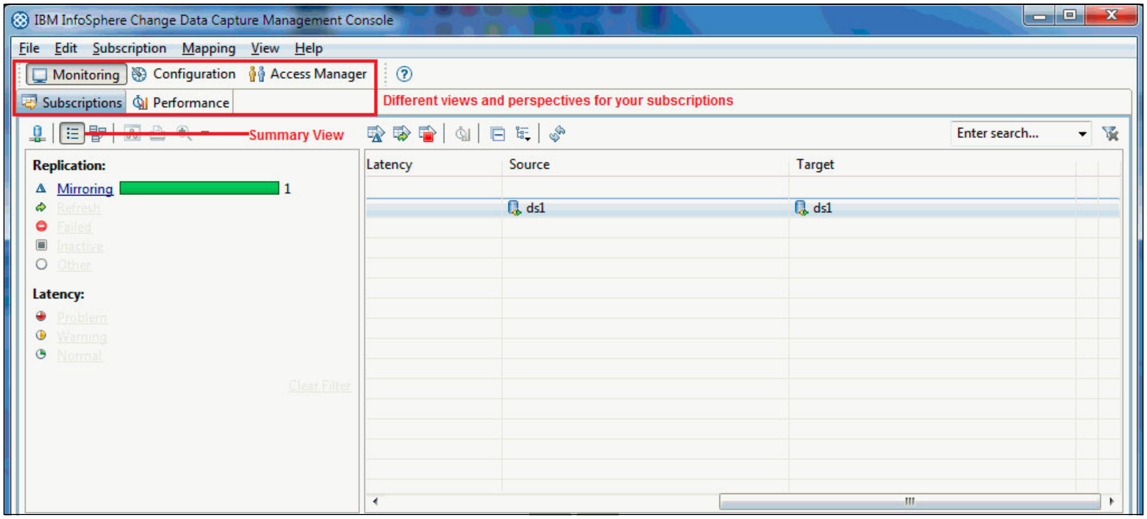



*Figure 3-9 InfoSphere CDC Management Console*

The subscription summary page shown displays the status of your subscription, which should be in mirroring mode. This mode displays a summary of replication states and latency on the left, and a list of subscriptions on the right.

To gather information regarding your replication activity, right-click the **SUB1** subscription **SUB1** and select **Collect Statistics**. Figure 3-10 shows the statistics pane on the bottom of the main window.

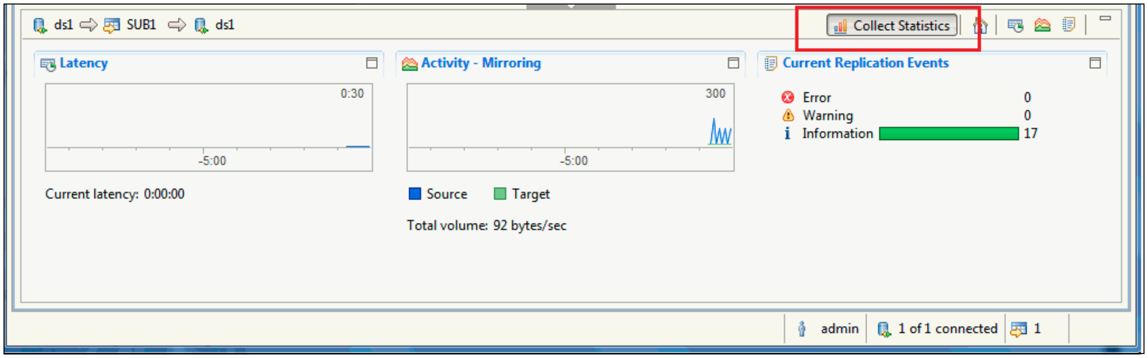
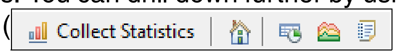



*Figure 3-10 Enable statistics on subscription to gather information*

The summary page shows a graphical view of latencies, activities, and event counts. You can drill down further by using one of the three views in the summary pane ( 

).

- Latency view shows detailed replication latency information, as seen in Figure 3-11.

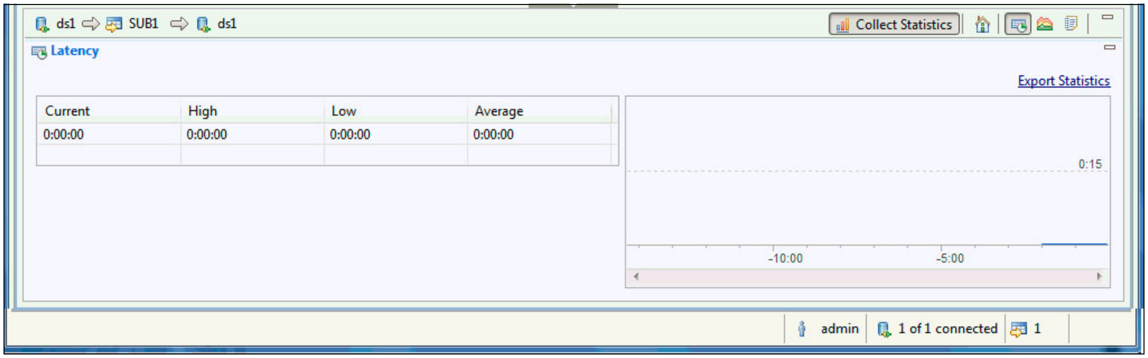



*Figure 3-11 Latency view*

- Activity view shows subscription activity statistics.The chart in Figure 3-12 shows the amount of data (in bytes) or the number of apply operations that are being processed by InfoSphere CDC.

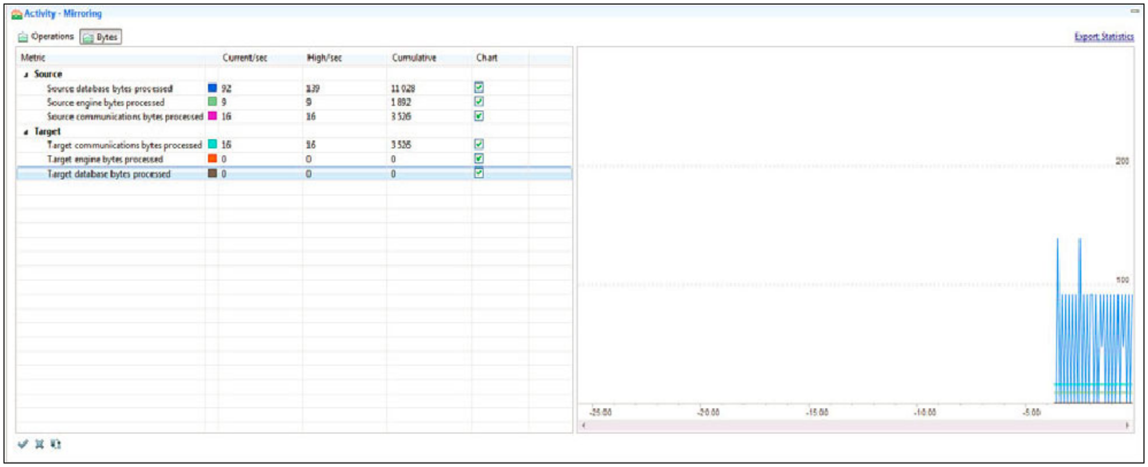



*Figure 3-12 Activity view*

- Events view shows a sortable view of subscription events. This view can be used to get more information regarding particular events that lead to a subscription failing or behaving abnormally. Events for subscription SUB1 are shown in Figure 3-13 on page 105.

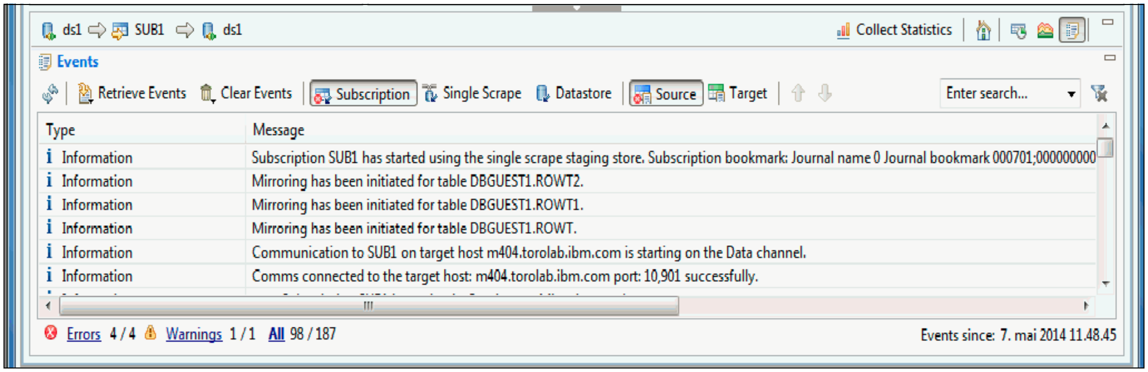



*Figure 3-13 Events log for current subscription*

To view the busiest table in your subscription, right-click the subscription in the main view and select **Show in performance view**. When the view opens, right-click the subscription and select **Busy tables..**

## Analyzing subscription events

To get more detail about a subset of subscription events, click **Retrieve Events** in the events view of the InfoSphere CDC Management Console and select an appropriate length of time (such as the last 24 hours). The Events pane is populated with information regarding each event. To read the full message, you can right-click an event and select **Event Details**. You can also get to the events view, by right-clicking the subscription for shadow tables in the summary page and selecting **Show Event Details**.

To further improve monitoring capabilities, set up event notifications against your subscription. Notifications are most useful when you are troubleshooting replication activities and you want to detect particular events that are happening in the source and target datastores, and subscriptions. To set up notifications, you must stop mirroring and end replication on the subscription for shadow tables, as described in 3.3.7, “InfoSphere CDC event notification setup” on page 74.

To learn more about the process of monitoring your subscription and setting event notifications, refer to the IBM Knowledge Center, found at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.mcadminguide.doc/concepts/monitoring.html?lang=en

## 3.5.7 Troubleshooting shadow tables

When troubleshooting issues with shadow tables, the best approach is to verify both the DB2 database and InfoSphere Change Data Capture. We suggest that the user checks the status of replication and its settings in DB2 and InfoSphere CDC to resolve problems that are specific to shadow tables. This section discusses important issues to verify when encountering any problems with shadow tables.

The following is a list of some reasons why shadow tables might not be used in the query plan for a statement:

- Special registers allowing queries to use shadow tables might not be set correctly: See "Checking DB2 special register settings" on page 109.
- Latency might be out of date: See "Checking current latency status" on page 110.
- The EXPLAIN utility might be used to determine reasons why a shadow table might not have been chosen, for example, query references row-organized tables with no associated shadow tables: See "Troubleshooting query routing using EXPLAIN" on page 107.
- Replication might be stopped: See "Diagnosing issues with InfoSphere CDC replication" on page 111.

### Checking query routing to shadow tables

Monitoring table functions can be used to gather information about which tables were used in the query plan for statements that are in cached memory.

Run the query shown in Example 3-41 to view a list of all the objects (tables or indexes) that are used in the query plan for a statement. The *statement-text* in italics is where the query statement would be placed.

*Example 3-41 List of objects used in query plan*

```
SELECT DISTINCT stmt.EXECUTABLE_ID, OBJECT_TYPE,
  substr(OBJECT_NAME,1,25) as OBJECT_NAME,
  substr(OBJECT_SCHEMA,1,25) as OBJECT_SCHEMA
FROM
  table(MON_GET_PKG_CACHE_STMT(NULL,NULL,NULL,-1)) AS stmt,

table(MON_GET_SECTION_OBJECT(stmt.EXECUTABLE_ID,NULL,NULL,NULL,NULL,-1)
) AS object
WHERE
  STMT_TEXT = 'statement-text';
```

Example 3-42 shows the edited result for a query similar to the preceding query run against a statement.

*Example 3-42 Result for a query against objects used in a query plan*

```
EXECUTABLE_ID OBJECT_TYPE   OBJECT_NAME   OBJECT_SCHEMA
-------------- -----------   ------------  --------------
x'0100..7################' TA            T1_SHADOW     ONLINE
  1 record(s) selected.
```

To view the list of statements that are currently cached in memory where a particular shadow table was used, use the query shown in Example 3-43.

*Example 3-43 List of statements for shadow table t_shadow in schema DWH*

```
SELECT DISTINCT
  object.EXECUTABLE_ID,
  substr(STMT_TEXT,1,200) as STMT_TEXT
FROM
  table(MON_GET_SECTION_OBJECT(NULL,NULL,'DWH',NULL,'T_SHADOW',-1)) AS
object,
  table(MON_GET_PKG_CACHE_STMT(object.EXECUTABLE_ID,NULL,NULL,-1)) AS
stmt
```

If the statement is no longer cached in memory, or more detailed information is required regarding the query plan, use the EXPLAIN utility.

## Troubleshooting query routing using EXPLAIN

The EXPLAIN utility can be used to diagnose issues that result in shadow tables not being used for query optimization. The EXPLAIN output for a query that is being assessed contains information that includes extended diagnostic information that can be used when troubleshooting query routing issues.

To use the EXPLAIN utility, you must go through a series of steps to create the explain tables, enable explain capture, collect the diagnostic information about the query in question, and format the output. These steps are outlined in the DB2 Knowledge Center at:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.cmd.doc/doc/r0002353.html

The following examples in this section outline information returned by the EXPLAIN utility for shadow tables and the reasons they might occur.

- Queries that run under isolation level RS or RR are not routed to shadow tables. Example 3-44 shows that the isolation level of the materialized query table prevented the table from being used in the query.

*Example 3-44 Isolation level does not allow routing to shadow tables*

```
Extended Diagnostic Information:
--------------------------------
Diagnostic Identifier:  1
Diagnostic Details:     EXP0053W
The materialized query table "DTW"."T_SHADOW" was not considered for
querymatching because the isolation level of the query is higher
than the isolation level of the materialized query table.
```

- Queries are routed to shadow tables only when all source tables referenced in the query have associated shadow tables. Example 3-45 shows the message that is returned when a query references some row-organized tables which do not have shadow tables associated.

*Example 3-45 Query references some source tables with no shadow tables*

```
Extended Diagnostic Information:
--------------------------------
Diagnostic Identifier:  1
Diagnostic Details:     EXP0076W
No materialized query table matching was performed on the statement
during query rewrite because there is a shadow table defined on at
least one, but not every, row-organized table referenced in the
query...
```

- Example 3-46 output shows the message that is returned when query routing to the shadow table does not happen because the current latency between the source table and the shadow table does not fall within the user-defined limit that is specified by the value of the `CURRENT REFRESH AGE` special register.

*Example 3-46 Current latency between source and shadow tables out of limit*

```
Extended Diagnostic Information:
--------------------------------
Diagnostic Identifier:  1
Diagnostic Details:     EXP0087W
No materialized query table matching was performed on the statement
during query rewrite because there is a shadow table defined on one
of the tables in the query and the current replication latency is
larger than the time value specified in the CURRENT REFRESH AGE
special register.
```

### Checking DB2 special register settings

Special registers are used to store information for applications that can be referenced in SQL statements. To ensure that shadow tables are considered for query optimization, the `CURRENT REFRESH AGE` and `CURRENT MAINTAINED TABLE TYPES` special registers are to be set to appropriate values for your connection as discussed in 3.4.2, “DB2 configuration parameters” on page 79.

> **Note:** Values returned might not be the same as the settings applied to concurrent applications, as these settings are applied at the connection level.

To check the value for `CURRENT REFRESH AGE` while a query was in process, issue the query as show in Example 3-47.

*Example 3-47 SQL statement to query the current refresh age*

```
SELECT
  stmt.EXECUTABLE_ID,
  ( SELECT substr(VALUE,1,24)
    FROM table(COMPILATION_ENV(COMP_ENV_DESC))
    WHERE NAME = 'REFRESH_AGE'
  ) as REFRESH_AGE,
  ( SELECT substr(VALUE,1,24)
    FROM table(COMPILATION_ENV(COMP_ENV_DESC))
    WHERE NAME = 'MAINTAINED_TABLE_TYPE'
  ) as MAINTAINED_TYPES
from
  table(MON_GET_PKG_CACHE_STMT(null,null,null,-1)) AS stmt
where
  STMT_TEXT = statement_text;
```

Example 3-48 shows the output of a query similar to the one shown in Example 3-47.

*Example 3-48 Query output for current refresh age value for query statement*

```
EXECUTABLE_ID REFRESH_AGE              MAINTAINED_TYPES
----------------------- ------------------------ ---------------------
x'0100######## 000#########3188' +00000000001000.000000   REPLICATION

  1 record(s) selected.
```

> **Note:** The QUERY OPTIMIZATION level must also be set to 2 or a value greater than or equal to 5.

## Checking current latency status

InfoSphere CDC replication must be up to date for shadow tables to be considered by the optimizer. It is advisable to compare the values in the latency table REPL_MQT_LATENCY with the current time stamp to ensure that the latency is not out of date.

Example 3-49 shows the query that can be used to check latency.

*Example 3-49 SQL statement to query to check latency and its sample output*

```
SELECT
  COMMIT_POINT, DELAY_OFFSET,
  int( CURRENT TIMESTAMP - (
      timestamp('1970-01-01') + (COMMIT_POINT - DELAY_OFFSET) seconds +
CURRENT TIMEZONE
  ) ) as LATENCY
FROM
  SYSTOOLS.REPL_MQT_LATENCY;


COMMIT_POINT         DELAY_OFFSET         LATENCY
-------------------- -------------------- -----------
          1405089689                    0          28

  1 record(s) selected.
```

Check the latency table when troubleshooting issues regarding InfoSphere CDC sending latency information to DB2. If there is an issue in sending information or access issues with InfoSphere CDC and the latency table, this can be verified with the latency table. If there is a row in the latency table, with the value 0 in the COMMIT_POINT column, you should verify the conditions, shown in Table 3-7, that might causing the issue.

*Table 3-7 Issues with latency and their resolution*

| Issue | Resolution |
|---|---|
| Subscription does not start: latency table does not exist but the `maintain_replication_mqt_latency_table` registry parameter is set to *true.* | Create the `SYSTOOLS.REPL_MQT_LATENCY` table and then start the subscription. |
| InfoSphere CDC not sending information to DB2: InfoSphere CDC was started when the `maintain_replication_mqt_latency_table` registry parameter was set to *false*. | Stop the subscription, set the `maintain_replication_mqt_latency_table` registry to true, and then restart the subscription. |

| Issue | Resolution |
|---|---|
| InfoSphere CDC is not sending information to DB2: InfoSphere CDC user does not have permissions to update latency table. | Ensure the correct permissions are set for InfoSphere CDC user. |
| Latency table is not being updated: The table space that contains the latency table is not in a state where it can be updated. For example, the latency table is in the backup-pending state. | Ensure that the table space is not in backup-pending state. |
| InfoSphere CDC replication has stopped or table is not accessible: If calculated LATENCY value is large but DELAY_OFFSET is small. | Verify access to table and proceed to diagnose issues that might be causing the DELAY_OFFSET value changes. |
| InfoSphere CDC is lagging behind the changes to the source tables: The value of `DELAY_OFFSET` is large or increasing. | Check issues that might cause the replication to be slow. |

## Diagnosing issues with InfoSphere CDC replication

Queries might not be routed to shadow tables if InfoSphere CDC replication stops. To verify the status of InfoSphere CDC replication using the command line, run the commands in the following examples as the InfoSphere CDC user. The examples use cdcinst1 as the InfoSphere CDC instance name and SUB1 as the subscription name.

> **Note:** To learn more about the usage of the InfoSphere CDC commands, add "-?" after each command name, for example, **`./dmgetsubscriptionstatus -?`**.

Example 3-50 shows that the InfoSphere CDC user navigates to the bin folder for InfoSphere CDC replication installation and issues a command to monitor subscription status.

*Example 3-50 Check the status of the InfoSphere CDC subscription*

```
# Execute these system commands as CDC user, that is, cdcuser

cd [...]InfoSphereChangeDataCapture/ReplicationEngineforIBMDB/bin
./dmgetsubscriptionstatus -I cdcinst1 -p -s SUB1
Subscription : SUB1
Status : Running
```

Example 3-51 shows that the InfoSphere CDC user can run the **dmshowevents** command to check the InfoSphere CDC event log for a summary of errors that might affect replication. You can use this command as an alternative to the Event Log view in Management Console as seen in 3.5.6, “Monitoring replication” on page 101. The command shown here limits the number of events reported to a maximum of 100.

*Example 3-51 Check InfoSphere CDC event log for details regarding subscription events*

```
# Execute these system commands as CDC user, that is, cdcuser

cd [...]InfoSphereChangeDataCapture/ReplicationEngineforIBMDB/bin
./dmshowevents -I cdcinst1 -s SUB1 -c 100
TIME|AGENTTYPE|SUBSCRIPTION|EVENTID|SEVERITY|EVENTPROGRAM|EVENTTEXT
2014-05-01 16:04:38.483000000000|S|SUB1|2922|Information| |Subscription SUB1 has
started using the single scrape staging store. Subscription bookmark: Journal name 0 Journal
bookmark
000701;000000000000112500000000004039e;5362a8d4000000000000;000000000000000
0112500000000 0004039e Staging store oldest bookmark: Journal name 0 Journal bookmark
INVALIDBOOKMARK Staging store newest bookmark: Journal name 0 Journal bookmark
INVALIDBOOKMARK
2014-05-01 16:04:38.470000000000|S|SUB1|44|Information||Mirroring has been initiated for
table INSTV105.T1.
```

To get more information regarding the datastore objects, the user can issue the **dmgetstagingstorestatus** command to check the status of the InfoSphere CDC datastore objects. If replication is not running, restart replication by using the **dmstartmirror** command, as shown in Example 3-52.

*Example 3-52 Restart replication*

```
# Execute this system command as CDC user, i.e. cdcuser

cd [...]InfoSphereChangeDataCapture/ReplicationEngineforIBMDB/bin
./dmstartmirror -I cdcinst1 -s SUB1
```

> **Note:** See 3.5.6, “Monitoring replication” on page 101 for more detail regarding gathering information and diagnosing issues with InfoSphere CDC.

### Checking the values of InfoSphere CDC system parameters

Verify that InfoSphere CDC system parameters are set to appropriate values so that query plans can use shadow tables. View the current settings by issuing the **dmset** command as the InfoSphere CDC user, and make sure the values are as set under 3.3.5, "Replication setup" on page 68. Example 3-53 on page 113 shows how to run **dmset** command.

*Example 3-53 Using dmset command*

```
cd [...]/InfoSphereChangeDataCapture/ReplicationEngineforIBMDB2/bin/
./dmset -I cdcinst1
```

The value of the system parameter, `acceptable_latency_in_seconds_for_column_organized_tables`, must be less than the value of the `CURRENT REFRESH AGE` special register. The `maintain_replication_mqt_latency_table` parameter should be set to true, and the `mirror_auto_restart_interval_minutes` system parameter should be set to a value greater than 0 (a value of 1 or 2 is preferred for most environments).

## 3.5.8 Restoring replication when restricted DDL is applied to a table

InfoSphere CDC maintains and uses metadata that describes source and shadow tables; any changes caused to the structure of the source table results in a disruption in replication. As a result of DDL changes to the source table, InfoSphere CDC can no longer read the database log records during mirroring. In this scenario, the metadata must be updated and the source tables have to be reorganized before replication can continue.

Stop replication using the InfoSphere CDC Management Console, or command line and run the required DDL operations on the source table after dropping the current associated shadow table. After the DDL operations are applied, reorganize the source table, and then re-create the shadow table. Example 3-54 on page 114 shows the sequence of steps discussed.

*Example 3-54 SQL statements to apply DDL changes to source table and restore replication*

```
drop db DTW.T_SHADOW ;

-- Perform DDL operations on the source table
alter table DTW.T_ROW alter column i1 set data type bigint ;

--Reorganize the table after the alter command
reorg table DTW.T_ROW ;

--Recreate the shadow table
create table DTW.T_SHADOW as
      (select * from DTW.T_ROW)
    data initially deferred
    refresh deferred
    maintained by replication
    organize by column;
set integrity for DTW.T_SHADOW all immediate unchecked ;
alter table DTW.T_SHADOW add constraint DTW.T_SHADOW_pk primary key
(pk);
```

After the Shadow table is recreated, it is necessary to update the table definitions in the InfoSphere CDC data store. If the DDL statement applied to the source table is *not* RENAME TABLE, update table definitions by using the Management Console or command line. Here, we discuss the Management Console.

Navigate to tab **Configuration** → **Datastores**, right-click the datastore object, and select **Replication Tables**. Within Replication Tables, select the source table and click **Update** and **Close**.

If the DDL statement applied to the source table is RENAME TABLE, you are required to delete the table mapping and re-create it because InfoSphere CDC must reference the name change. To do so, navigate to tab **Configuration** → **Table Mappings**, right-click the table mapping for the shadow table, and select **Delete** and click **Yes**. In our example, the DTW.T_ROW/ DTW.T_SHADOW table mapping is deleted. After deletion, re-create the table mapping for the shadow table with the new source table name.

Resume mirroring after the process is complete. Using the Management Console, navigate to **Configuration** → **Subscriptions**, right-click the subscription, and select **Start Mirroring**. Select **Continuous** and click **OK**.

### 3.5.9 Preventing disruption to replication with restrictive DDL applied to the source table

InfoSphere CDC replication can be impacted and stopped by certain DDL operations or a load being applied to the source table. To prevent this situation, follow the procedure described in this section.

Stop replication using the CDC Management console, or the command line, and apply the DDL operations. In case of DDL changes, complete the DDL operation and reorganize the source table. Example 3-55 shows an example of the DDL changes. In case of a load, complete the load.

*Example 3-55 SQL statements to apply DDL changes to source table and restore replication*

```
alter table DTW.T_ROW add column i3 integer ;
reorg table DTW.T_ROW ;
```

After the DDL changes are applied, update table definitions by using the Management Console or command line. Here, we discuss the Management Console.

Navigate to tab **Configuration** → **Datastores**, right-click the data store, and select **Replication Tables**. Within Replication Tables, select the source table and click **Update** and **Close**.

After the definitions are updated, the table mappings must be refreshed to start replication. Navigate to **Configuration** → **Table Mappings**, right-click the table mapping for the source-shadow, DTW.T_ROW/ DTW.T_SHADOW, and **select Flag for Refresh** and click **Yes**. Select **Standard Refresh** as the refresh option.

Resume mirroring after the process is complete. Using the Management Console, navigate to **Configuration** → **Subscriptions**, right-click the subscription, and select **Start Mirroring**. Select **Continuous**.

## 3.6 Shadow tables and HADR

Shadow tables in an HADR environment is based on the combination of the shadow tables feature with the support for HADR in column-organized tables. Figure 3-14 shows a topology where the shadow tables are implemented in an HADR environment.

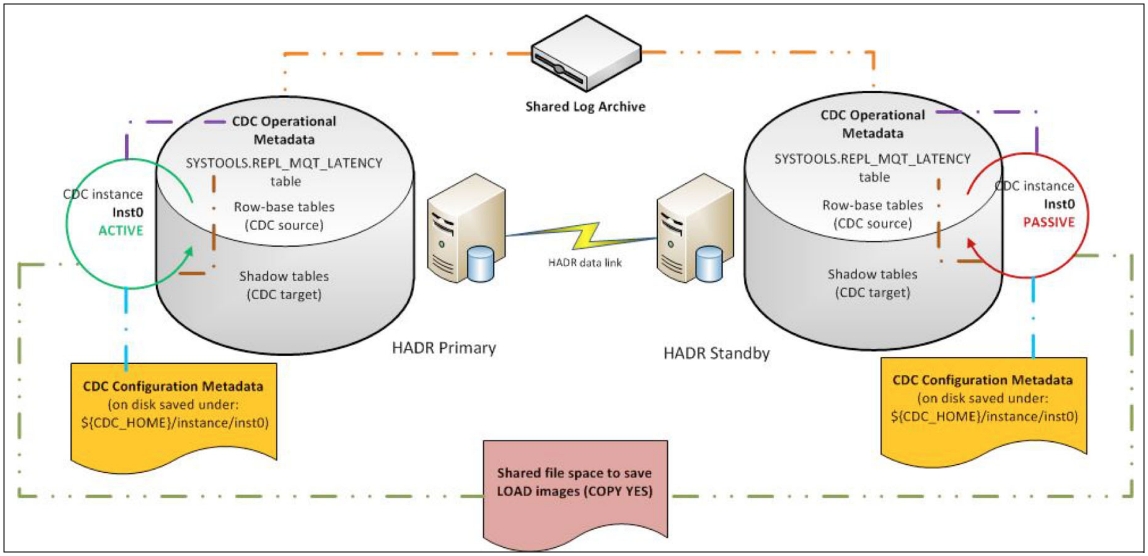



*Figure 3-14 Topology of an HADR environment with shadow tables*

This implementation includes the following components:

- DB2 HADR primary server:
  - DB2 instance including the HADR primary database
  - InfoSphere CDC instance (Access Server and Replication Engine for DB2) and metadata
- DB2 HADR standby server:
  - DB2 instance including the HADR standby database
  - InfoSphere CDC passive instance (Access Server and Replication Engine for DB2) and metadata
- Shared elements

As part of this initial setup, it is suggested to use a shared log archive. When using HADR with shadow tables, a shared archive is even more essential to ensure that CDC log capture has access to all of the log files and the possibility that CDC will invoke log retrieve after a failover is quite high.

These elements must be accessible to all servers included in the configuration (shared storage, shared file system, or GPFS™ file system):

- Shared log archive for DB2 for HADR (shared file system or IBM Tivoli Storage Manager (TSM) can be used)
- Shared file space for InfoSphere CDC load refresh images

## 3.6.1 Shadow tables configuration

As part of shadow tables configuration, InfoSphere CDC must be installed on all HADR servers including primary, standby, and any additional auxiliary standby.

As seen in 3.3, “Deployment” on page 61, to install InfoSphere CDC, the following components are used:

- InfoSphere CDC Access Server
- InfoSphere CDC Replication Engine for DB2
- InfoSphere CDC Management Console

There are some considerations for the deployment of these three components when shadow tables are deployed in an HADR environment.

### InfoSphere CDC Access Server

In the disaster recovery (DR) scenarios where DB2 with HADR is used, we suggest using a virtual IP address that always corresponds to the HADR primary server. In case of failover, using virtual IP can make the transition easier and it would be even simpler for applications to connect to the active database.

In an HADR configuration with shadow tables defined, using a separate InfoSphere CDC Access Server is a simpler approach for operation. The Access Server can monitor the InfoSphere CDC instances on both HADR servers. Because the Access Server is not part of the HADR configuration, it will not be affected when there is failure on the HADR servers.

When shadow tables are implemented in an HADR environment, there are different options to deploy the InfoSphere CDC Access Server.

- InfoSphere CDC Access Server collocated with InfoSphere CDC Replication Engine on each HADR server
- InfoSphere CDC Access Server deployed outside of the HADR cluster
- InfoSphere CDC Access Server deployed outside of the HADR cluster using a virtual IP address for the primary database

### InfoSphere CDC Replication Engine for DB2

Because each HADR server might become primary at some point, InfoSphere CDC Replication Engine is configured on each HADR server (primary, standby, and auxiliary standby, if any).

**InfoSphere CDC Management Console**

This InfoSphere CDC component can be installed on a notebook, workstation, or any server or virtual machine. Therefore, deploy the InfoSphere CDC Management Console in a separate system other than the HADR servers.

## 3.6.2 HADR considerations

The preferable HADR configuration practices are applicable to an HADR environment with shadow tables. Some of the practices are described in 2.7.8, “HADR best practices” on page 51. For an HADR with a shadow table environment, the following items are even more important:

- Use dedicated, high performing disks and file systems for the database logs.
- Use a shared location for DB2 archive logs accessible to primary and standby.
- Have a dedicated network for the HADR primary-standby connection.
- Consider using a virtual IP for the database server.
- Consider using automatic client reroute.

For more information about the HADR best practices, see *High Availability Disaster Recovery* at the following web address:

https://ibm.biz/Bdx2Ga

## 3.6.3 InfoSphere CDC considerations

This section explains certain considerations regarding InfoSphere CDC metadata and how it is managed when shadow tables are deployed in an HADR environment.

With InfoSphere CDC, there are two types of metadata that are used to synchronize the source and target tables:

- Operational metadata

  The operational metadata is information such as the instance signature and bookmarks. This metadata is stored in the database and is replicated by HADR to the standby.

- Configuration metadata

  The configuration metadata is information such as subscriptions and mapped tables. This metadata is stored in the CDC home directory, so it is not replicated by HADR. Therefore, after you have set up HADR with shadow tables, any configuration changes made to the shadow tables on the primary must be propagated to the standby using a scheduled CDC metadata backup-and-copy.

### Operational metadata

The operational metadata used in InfoSphere CDC is stored inside the DB2 database. This situation means that operational metadata is synchronized and replicated to the standby database as part of the HADR replication, so there is no need to take care of this metadata separately.

### Configuration metadata

The configuration metadata for InfoSphere CDC includes table mapping change information and table modification information. This information is stored in a file system directory on the InfoSphere CDC instance deployed on the HADR primary server. These types of changes on table mappings or changes to tables (such as adding or dropping a column) must be kept synchronized across all InfoSphere CDC instances deployed on the HADR standby servers. These InfoSphere CDC instances are idle instances that must have the latest copy of the configuration metadata to ensure that no configuration or subscription details are lost during failover.

### InfoSphere CDC instance

In a scenario with shadow tables and HADR, you must set up InfoSphere CDC on each standby database of the HADR environment. Each standby database is kept synchronized with the primary database. In the primary database, an active InfoSphere CDC instance maintains the shadow table functionality for the primary database. Because the standby databases do not require the shadow table feature when in the standby mode, the InfoSphere CDC instance on each standby database is deactivated or in passive mode. The metadata for InfoSphere CDC is replicated across all standby databases. When a takeover occurred, the InfoSphere CDC instance on the standby database that becomes the new primary is activated with the most recent configuration metadata.

## 3.6.4 Standby server considerations

The configuration of shadow tables in an HADR environment is similar to what must be done in a standard shadow table configuration. There are, however, some considerations related to the standby database server.

### Prerequisites

Other than the prerequisites already mentioned for shadow tables and InfoSphere CDC in general, there are few other items applicable to the standby database configuration and setup.

The InfoSphere CDC instance on each standby database server remains in passive mode and stopped. The configuration metadata is kept in synchronization with the active InfoSphere CDC instance on the primary database server. Because the operational metadata is replicated with HADR, the operational metadata is the same as on the primary instance.

On the DB2 standby instance, the following parameters must be kept the same as on the primary server:

- DB2 instance and database name
- HADR service port (SVCENAME)
- Connection parameters (user, password, port)
- Log location (LOGARCHMETH1)

**Shared location:** Use a shared location for archive logs that are accessible from all servers in the HADR environment.

For the InfoSphere CDC instance on a standby server, the following prerequisites apply:

- Identical InfoSphere CDC installation location ($CDC_HOME) as on the primary instance.
- InfoSphere CDC refresh loader directory on the primary server must be accessible from all HADR servers. The DB2 instance owner and the InfoSphere CDC user must have read and write permission for this directory.
- InfoSphere CDC user name (that is, `cdcuser`) must be the same as the user name on the primary server instance. For Linux and UNIX environments, the user ID must be the same (that is, 1000).
- In Linux and UNIX environments, InfoSphere CDC user must have the same primary group (and group ID) as the DB2 instance owner (that is db2iadm1) on each server (primary and standby).

These prerequisites ensure that all the InfoSphere CDC instances on a DB2 standby instance are ready to become active if the DB2 standby database is changed to the primary database.

### Restrictions

There are no additional restrictions for HADR with shadow tables other than the shadow tables restrictions and the HADR support for read on standby on column-organized tables.

To avoid any potential incompatibility in the metadata from different versions, use the same version of InfoSphere CDC on all HADR servers.

## 3.6.5 Installation and configuration

This section explains how to configure and enable shadow tables in an HADR environment.

### HADR initial state

An HADR environment must be in place first before setting up the shadow tables. For the HADR configuration, see 2.7.3, “Configuring primary and standby databases” on page 43.

For this section, the assumption is that the HADR is already set up and its configuration and failover were verified already. Another possible scenario is to have already enabled shadow tables on the HADR primary server only. In that particular case, to perform the HADR configuration, it is easier to start with a full backup of the primary database. It will already include the InfoSphere CDC metadata information and that backup can be restored on the standby database as the step to configure HADR.

### InfoSphere CDC installation

In an HADR configuration, the InfoSphere CDC component “InfoSphere CDC for DB2 for Linux, UNIX, and Windows” must be installed on all HADR servers (primary, standby, and any auxiliary standby).

For installation details, see 3.3, “Deployment” on page 61.

The following items should be noted when InfoSphere CDC for DB2 uses the same location for InfoSphere CDC on each server ($CDC_HOME):

- Run InfoSphere CDC configuration tool only on the active InfoSphere CDC instance after the installation.
- On Linux and UNIX platforms, the user for InfoSphere CDC installation must be the same on all HADR servers.

  User ID and group ID for the InfoSphere CDC user must match on all HADR servers (primary, standby and auxiliary standby if applicable).
- The InfoSphere CDC user primary group on each HADR server must be the same primary group as the DB2 instance owner.
- The InfoSphere CDC user profile (`/home/cdcuser/.bashrc`) must be modified to alter the default privileges for directory creation as shown in Example 3-56.

*Example 3-56 umask setting to be added to the InfoSphere CDC user profile*

```
[cdcuser@db2pure04 cdcuser]$ cat /home/cdcuser/.bashrc
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
   . /etc/bashrc
fi

# User specific aliases and functions
#UMASK ADDED TO ALLOW DB2 USER access CDC_REFRESH_LOADER_BACKUP_PATH
umask 007
```

### InfoSphere CDC Access Server

As described in 3.6.1, “Shadow tables configuration” on page 117, there are three possible deployment scenarios for InfoSphere CDC Access Server. Because on the InfoSphere CDC Access Server, a datastore is defined pointing to the InfoSphere CDC Replication Engine located on the actual HADR primary server, the following must be considered when configuring the InfoSphere CDC Access Server according to these three scenarios:

- InfoSphere CDC Access Server collocated with InfoSphere CDC Replication Engine on each HADR server

  In this case, each datastore can be configured using the host name “localhost”, because the InfoSphere CDC Replication Engine is always on the same server as the InfoSphere CDC Access Server.

- InfoSphere CDC Access Server deployed outside of the HADR cluster

  In this case, the datastore on the InfoSphere CDC Access Server is configured using the IP address or host name of the InfoSphere CDC Replication Engine located on the HADR primary server. However, after a failover, this datastore definition must be updated to use the InfoSphere CDC Replication Engine on the new HADR primary server.

- InfoSphere CDC Access Server deployed outside of the HADR cluster using a virtual IP address for the primary database

  When a virtual IP address is used to point to the current HADR primary server, the datastore definition uses this virtual IP address. With this configuration, after the datastore is set to use the Virtual IP address, it will always point to the correct InfoSphere CDC Replication Engine on the current HADR primary server.

After InfoSphere CDC and its components are installed, you need to configure the active InfoSphere CDC instance, located on the HADR primary server.

## InfoSphere CDC configuration on the primary instance

The initial InfoSphere CDC active instance is configured on the HADR primary server. Complete the following steps to configure the InfoSphere CDC active instance:

1. Run InfoSphere CDC configuration tool, **`dmconfigurets`**.
2. Create an InfoSphere CDC instance on the primary HADR server.
3. During the instance configuration, a file system path must be selected for the InfoSphere CDC “refresh loader path”. This file system path must be created on the primary server and on all the standby HADR servers.

The InfoSphere CDC configuration metadata is created under the product home directory. The operational metadata is created inside the database itself so the configuration metadata is synchronized and replicated to all standby servers. The operational metadata is replicated to all standby servers as part of the HADR replication.

For more details about adding a new instance of InfoSphere CDC for DB2 for LUW (Windows), see the IBM Knowledge Center at the following web address:

http://www-01.ibm.com/support/knowledgecenter/SSTRGZ_10.2.1/com.ibm.cdcdoc.cdcfordb2luw.doc/tasks/addanewinstance_windows.html?lang=en

## Configuring Refresh Loader for InfoSphere CDC

In the InfoSphere CDC refresh process, the default option is to use LOAD with the NONRECOVERABLE option. To avoid placing the table space where the table is in backup pending mode, it is necessary to use LOAD with the option COPY YES. With this change, you can avoid having the table space in backup pending state but still retain the ability to roll forward logs on the HADR standby database.

> **CDC system parameters:** The InfoSphere CDC system parameters must be set before starting the InfoSphere CDC instance.

Use the command in Example 3-57 to apply the COPY YES option in the InfoSphere CDC refresh.

*Example 3-57* dmset options for InfoSphere CDC refresh loader for HADR

```
dmset -I <cdc_inst_name> refresh_loader_with_backup_copy=true
dmset -I <cdc_inst_name> refresh_loader_cleanup_backup_copy=false
dmset -I <cdc_inst_name>
refresh_loader_backup_path=<cdc_refresh_loader_path>
```

It is important to mention that both InfoSphere CDC and DB2 users require read and write permissions to the directory where the load files are stored, <cdc_refresh_loader_backup_path>. This directory can be the same directory specified earlier during the instance creation of `cdc_refresh_loader`.

For this directory path, the file system must use the same mount point on all HADR servers (that is, `/cdc/refresh_loader`) and must be accessible from all HADR standby nodes (primary, standby, and any auxiliary standby).

Each time the InfoSphere CDC Replication Engine creates a new directory under the <cdc_refresh_loader_path>/BACKUP_COPY/ directory for DB2 load copies, the directory is created with the InfoSphere CDC user ownership. The directory is used to keep the LOAD copies across the different HADR servers.

Because DB2 needs read and write privileges on the directory, the DB2 instance owner and the InfoSphere user must have the same primary group on all HADR servers. In addition, it is necessary to ensure that the DB2 user under the same primary group as the InfoSphere CDC user can access the LOAD directory created by InfoSphere CDC. To allow this situation, the InfoSphere CDC user profile has to be set as indicated in Example 3-56 on page 122 on each HADR server (primary, standby and auxiliary standby if any).

### Starting InfoSphere CDC instance on the primary server

At this point, you can start the InfoSphere CDC instance on the active server by following the steps described in 3.3.5, “Replication setup” on page 68.

When both the Access Server and the Replication Engine are started, the next step is to create a single subscription and map the tables. Use “One-to-One mapping” and “Standard” as the mapping type. For more details about these operations, see 3.3.6, “Subscriptions and table mappings” on page 72.

The next step after setting up the mapping is to start mirroring. Select “Continuous” as the mirroring method.

When the table mapping under the subscription becomes active, all shadow tables are refreshed and ready.

## Setting InfoSphere CDC instance on the standby server

To set up an InfoSphere CDC instance on the standby server, you can copy the instance directory from the primary server to each HADR standby server.

The following steps set up the InfoSphere CDC instance on the standby server:

1. Stop replication and the InfoSphere CDC instance on the primary server.

   You can use InfoSphere CDC Management Console or the InfoSphere CDC for DB2 commands to stop replication and the InfoSphere CDC instance.

   The following InfoSphere CDC for DB2 commands stop replication and the InfoSphere CDC instance:

   – Stop InfoSphere CDC replication (Example 3-58).

     *Example 3-58 Command to stop InfoSphere CDC replication on the subscription*

     ```
     dmendreplication -I <cdc_inst_name> -c -s <cdc_subscription_name>
     ```

     The “-c” option specifies that InfoSphere CDC end replication in a controlled manner. This option lets InfoSphere CDC complete all in-progress operations and apply pending changes to the target table.

   – Stop InfoSphere CDC instance (Example 3-59).

     *Example 3-59 Command to shutdown InfoSphere CDC instance*

     ```
     dmshutdown -I <cdc_inst_name>
     ```

   When using the command instead of the Management Console, verify that the subscription has stopped by checking the InfoSphere CDC instance logs in `$CDC_HOME/instance/<cdc_inst_name>/log`.

- Copy the entire instance directory, `$CDC_HOME/instance` from the primary server to each standby server.

  One way to do this is by creating a tar file and copying it over to each HADR standby server and extracting the tar file into the `$CDC_HOME/instance` directory.

  Example 3-60 shows the command to compress the active InfoSphere CDC instance directory from the HADR primary server.

  *Example 3-60 Compress the instance directory*

  ```
  cd $CDC_HOME/instance/
  tar -zcvf cdc_inst_<instance_name>.tar.gz <instance_name>
  ```

Example 3-61 shows the command to uncompress the tar file on the HADR standby server.

*Example 3-61 Uncompress the instance directory*

```
cd $CDC_HOME/instance/
tar -zxvf cdc_instance<instance_name>.tar.gz
```

## Verifying InfoSphere CDC and HADR configuration

After the active InfoSphere CDC instance is configured, verify that the actual configuration is functional by completing the following tasks:

1. Perform an HADR takeover (TAKEOVER HADR) and verify that HADR is in the “PEER” state. For information about how to check HADR status, see 2.7.3, “Configuring primary and standby databases” on page 43.

   On the HADR standby server, run the following command to start the takeover process:

   **`db2 takeover hadr on database $DATABASE`**

2. Start the InfoSphere CDC instance on the standby server using the following command:

   **`nohup ./dmts64 -I cdcinst1 &`**

   The instance name is the same as the one configured on the primary HADR server.

3. Change host name and IP address of the connection to the data source and update the subscriptions using that data source connections.
4. Start InfoSphere CDC replication using the command line or the InfoSphere CDC Management Console and verify that all table mappings become active.

After verifying that the InfoSphere CDC and HADR configuration is working as expected, you can failback to the old primary to use it as the HADR primary server.

To failback to the old primary server, complete the following steps:

1. Stop the replication and InfoSphere CDC on the new primary server.
2. Failback HADR to the old primary.

   From the old primary, enter the following command to start the fallback process:

   **`db2 takeover hadr on database $DATABASE`**

3. Verify that HADR is in the PEER state from the primary database.
4. When the HADR is in the PEER state, start InfoSphere CDC, activate the subscription, and perform the steps required to start mirroring.

At this point, the main configuration for shadow tables in an HADR environment can be considered completed.

## 3.6.6 Shadow tables configuration and metadata

An important consideration when implementing shadow tables for an HADR environment is how the InfoSphere CDC metadata is kept in synchronization with the primary server on each standby server.

There are two types of InfoSphere CDC metadata. The operational metadata is stored within the database and it is replicated to each standby server as part of HADR replication. For InfoSphere CDC configuration metadata (table mapping change information), the replication must be implemented manually to ensure that each standby server has the latest copy of the configuration metadata used on the primary.

To maintain the configuration metadata, consider the following two methods:

- On-demand metadata synchronization
- Programmed synchronization

### On-demand metadata synchronization

On the HADR primary server, the configuration metadata can be synchronized to each HADR standby server using **scp**, **rsync**, or similar commands. Example 3-62 shows how to synchronize the configuration metadata with the **rsync** command.

*Example 3-62 Synchronization for InfoSphere CDC configuration metadata*

```
export INSTANCE=cdcinst1
export CDC_HOME=/home/cdcuser/bin/ReplicationEngineforIBMDB2/
rsync -avz $CDC_HOME/instance/$INSTANCE/conf/md*
cdcuser@hadrstandby:$CDC_HOME/instance/$INSTANCE/conf
```

You also can use the **dmbackupmd** command to back up configuration metadata to the backup directory located in `$CDC_HOME/instance/<instance_name>/bX`, as shown in Example 3-63. This directory can have multiple backup files; use the latest backup file.

*Example 3-63 Taking a backup of the configuration metadata with dmbackupmd*

```
[cdcuser@db2pure06 bin]$ ./dmbackupmd -I cdcinst1
[cdcuser@db2pure06 bin]$ cd
/home/cdcuser/bin/ReplicationEngineforIBMDB2/instance/cdcinst1/conf/bac
kup/
[cdcuser@db2pure06 backup]$ ls -ltrh
total 144K
drwxrwx--- 2 cdcuser db2iadm1 4.0K Aug 11 10:27 b1
-rw-rw---- 1 cdcuser db2iadm1 135K Aug 11 10:29 dmbackupdir.tar.gz
drwxrwx--- 2 cdcuser db2iadm1 4.0K Aug 15 12:53 b2
[cdcuser@db2pure06 backup]$ cd b2/
[cdcuser@db2pure06 b2]$ ls
md$1.wal  md.dbn
```

Compress the backup directory in a tar file to synchronize it with each standby server as shown in Example 3-64.

*Example 3-64 Compressing and transferring the metadata backup to the standby server*

```
cd $CDC_HOME/instance/<instance_name>/conf/backup
tar -zcvf cdcdmbackupdir.tar.gz bnX
(Note: X refers to the most recent directory)

scp cdcdmbackupdir.tar.gz
cdcuser@db2shadowt:$CDC_HOME/bin/RE/instance/cdcinst1/backup
```

After the configuration metadata backup file is copied to each HADR standby server, uncompress the backup file by using the command shown in Example 3-65.

*Example 3-65 Replacing standby configuration metadata on the standby*

```
cd $CDC_HOME/instance/<instance_name>/conf/
tar -zxvf dmbackupdir.tar.gz
```

Consider keeping the log history synchronized across the standby servers by copying the database event file in `$CDC_HOME/instance/<instance_name>/events/*` to the standby servers. Use the same method of copying the configuration metadata.

## Programmed synchronization

You can also create a script and schedule a job to automate the configuration metadata synchronization process. Example 3-66 shows a script with the **rsync** command for synchronizing the configuration metadata.

*Example 3-66 Sample script to synchronize configuration metadata*

```
# cdc_md_sync.sh
#Sample script for instance metadata synchronization
#
#!/bin/bash
export INSTANCE=cdcinst1
export CDC_HOME=/home/cdcuser/bin/ReplicationEngineforIBMDB2/
rsync -avz $CDC_HOME/instance/$INSTANCE/conf/md*
cdcuser@hadrstandby:$CDC_HOME/instance/$INSTANCE/conf
```

Any job scheduler such as, **cron** can run the script. When you run the script with a scheduler, understand that the remote login to the standby system might be required.

The following **cron** command runs the synchronization script every15 minutes:

```
*/15 * * * * /home/cdcuser/cdc_md_sync.sh
```

The frequency set for this synchronization determines the gap in the configuration metadata between the primary InfoSphere CDC instance and the standby InfoSphere CDC instance.

## 3.6.7 Considerations for an HADR role switch

In addition to the steps required to perform a role switch (or takeover) in an HADR environment, some extra steps are required to failover the InfoSphere CDC active configuration to the new active HADR server.

### Planned failover

In case of a planned failover (takeover), use the following steps to stop InfoSphere CDC on the active server (HADR primary) and start it again on the passive server (new active):

1. Stop replication for the subscription on the HADR primary server where the InfoSphere CDC active instance is located. Run the following command:

   ```
   [cdcuser@db2pure06 bin]$ ./dmendreplication -I cdcinst1 -c -s SUB1
   Command for SUB1 successfully executed
   ```

2. Stop the active InfoSphere CDC instance on the HADR primary using the following command and wait until it finishes gracefully.

```
[cdcuser@db2pure06 bin]$ ./dmshutdown -I cdcinst1
```

The messages you should receive are:

```
Waiting for the single scrape change log to be persisted to disk.
This may take a while...
IBM InfoSphere Change Data Capture has successfully shutdown.
```

3. From the HADR standby server, perform the DB2 HADR takeover command to failover to the HADR standby server.

```
[cdcuser@db2pure04 bin]$db2 takeover hadr on database $DATABASE
```

4. After the HADR takeover is completed and the new primary is started, restore the latest copy of InfoSphere CDC configuration metadata from the old primary as described in 3.6.4, “Standby server considerations” on page 119.
5. Start the InfoSphere CDC instance on the new HADR primary server (ensure that configuration metadata was restored before this step).

```
nohup ./dmts64 -I cdcinst1 &
```

6. Update the host name or IP address of the connection to the datastore using the Management Console and update the subscriptions using the new datastore connection details.
7. Clear the staging store for the instance using the **dmclearstagingstore** command:

```
[cdcuser@db2pure04 bin]$ ./dmclearstagingstore -I cdcinst1
Command executed successfully
```

8. Restart the subscription, selecting the Start Mirror option from Management Console or using the **dmstartmirror** command from the command line.

```
[cdcuser@db2pure06 bin]$ ./dmstartmirror -I cdcinst1 -s SUB1
Command for SUB1 successfully executed
```

### Unplanned failover

For an unplanned failover where the HADR primary server is not reachable or fails, the failover procedure is the same as what is described in “Planned failover” on page 129 except for the first two steps. The HADR takeover command must use the BY FORCE option.

In the unplanned failover scenario when the HADR primary server is unavailable, it is important to consider that the InfoSphere CDC configuration metadata is the last copy of what was synchronized to the former standby server (now primary) before the failover event. If any table-mapping in the subscription was changed since that last synchronization, it is necessary to reapply these changes before clearing the staging store and restarting the subscriptions.

### 3.6.8 Considerations for InfoSphere CDC Access Server after a failover

After a failover procedure on the environment, take these considerations according to the three deployment scenarios of InfoSphere CDC Access Server:

- InfoSphere CDC Access Server collocated with InfoSphere CDC Replication Engine on each HADR server

  In this scenario, the datastore already points to the InfoSphere CDC Replication Engine on the same server. There is no need to update the datastore properties as each datastore is local to the Replication Engine.

  It is necessary, however, to update the connection IP address or host name when connecting with InfoSphere CDC Management Console after a failover. In this scenario, InfoSphere CDC Management Console points to the InfoSphere CDC Access Server on the previous HADR primary server (now standby). Therefore, the IP address or host name of the new primary server that hosts the InfoSphere CDC Access Server should be used when connecting from the InfoSphere Management console after a failover.

- InfoSphere CDC Access Server deployed outside of the HADR cluster

  In this scenario, the InfoSphere CDC Access Server is deployed outside of the HADR cluster using a third server. After failover, the datastore definition still points to the InfoSphere CDC Replication Engine located on the old primary HADR server, now HADR standby. Therefore, the datastore connection must be updated to point to the correct InfoSphere CDC Replication Engine on the new HADR primary server.

  For this scenario, it is not necessary to update the connection defined on the InfoSphere CDC Management Console. Using a unique InfoSphere CDC Access Server external to the HADR cluster, the connection defined on InfoSphere CDC Management Console is always pointing to the only InfoSphere CDC Access Server defined on the environment.

- InfoSphere CDC Access Server deployed outside of the HADR cluster using a virtual IP address for the primary database

  In this scenario, no change is required to the datastore. After a failover, the virtual IP address points to the actual HADR primary server. InfoSphere CDC Access Server datastore is defined used this virtual IP address so it will automatically point to the actual HADR primary and its CDC Replication Engine.

  It is also not necessary to update the connection defined on the InfoSphere CDC Management Console. Because there is a single InfoSphere CDC Access Server external to the HADR cluster, the connection defined here always points to the only InfoSphere CDC Access Server defined on the environment.

### Updating InfoSphere CDC Access Server datastore connection

When a failover is performed on an HADR environment where shadow tables are used, it might be necessary to update the datastore connection defined on InfoSphere CDC Access Server to point to the InfoSphere CDC Replication Engine located on the current HADR primary server.

To update the connection defined in the datastores, use the following steps:

1. Using the InfoSphere CDC Management Console, connect to the InfoSphere CDC Access Server. In case of multiple InfoSphere Access Servers, connect to the one deployed on the current HADR primary Server (Figure 3-15).

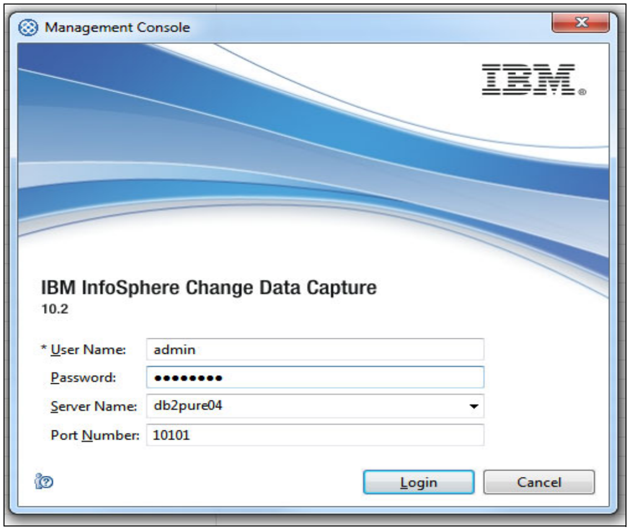



*Figure 3-15 Login to connect to the new active Access Server (on HADR primary)*

2. On the datastore properties panel (Figure 3-16), select **Update Related Subscriptions** to update the datastore properties and its subscription.

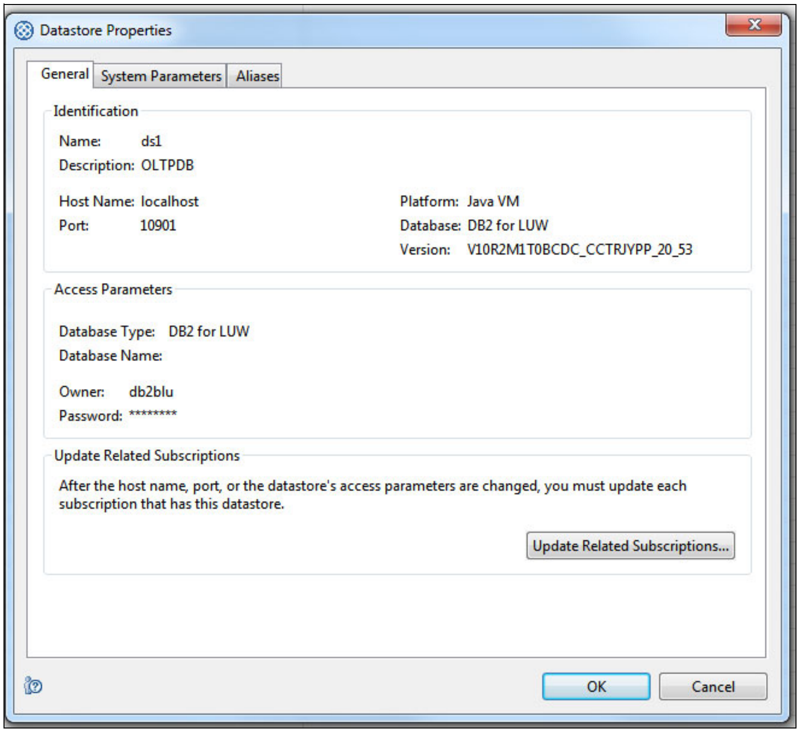



*Figure 3-16 Datastore properties panel*

3. The tool might show the change on the host name detected on the subscription (Example 3-17). If necessary, update the host name or IP address to point to the InfoSphere CDC Replication Engine located on the current HADR primary.

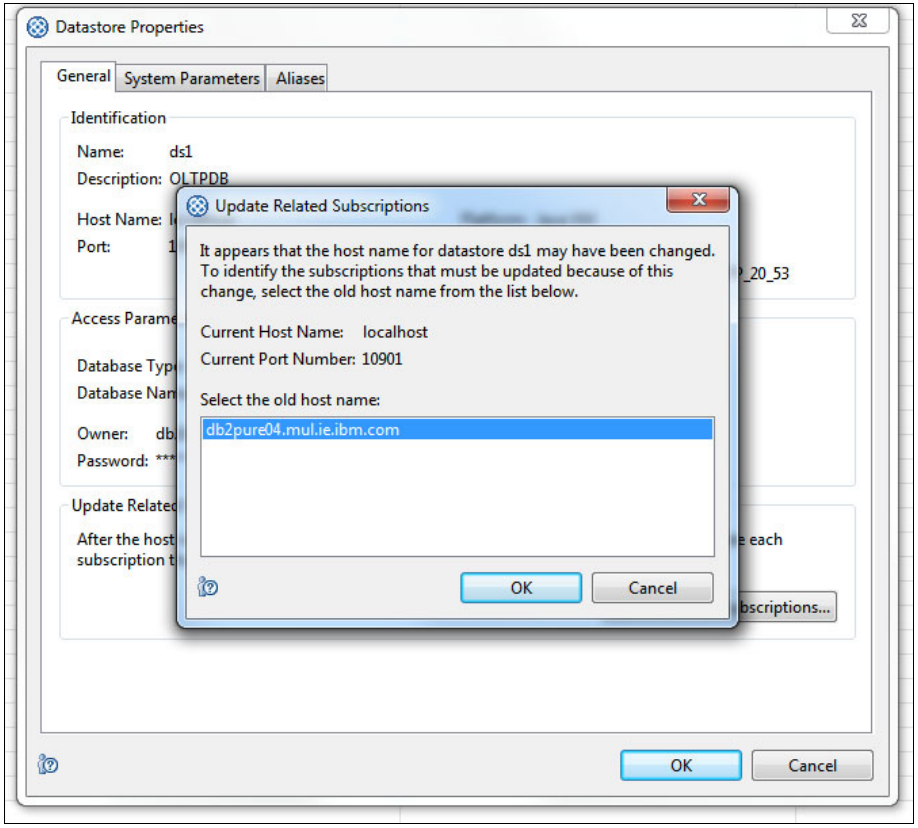



*Figure 3-17 Update Related Subscriptions message notifies about the host name change*

4. After the subscription is updated, the Management Console shows that the subscription is in an inactive state. To activate the subscription, select the subscription and use the **Start Mirroring** option as shown in Figure 3-18.

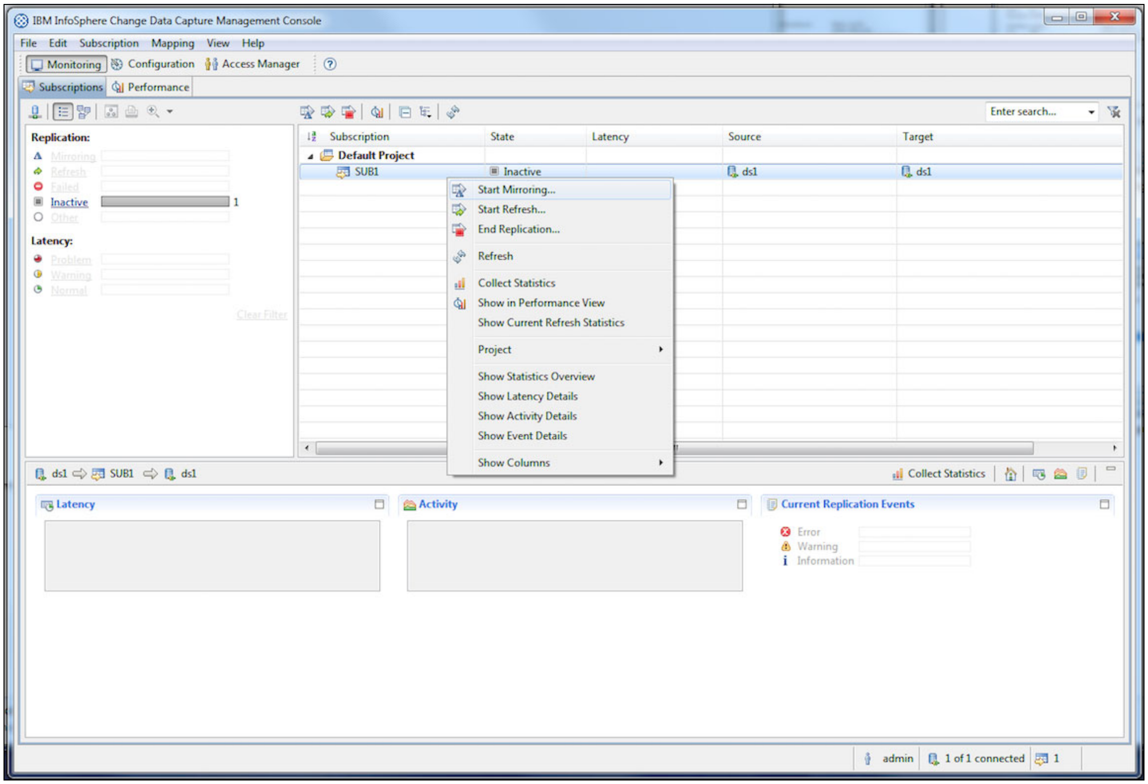



*Figure 3-18 Starting the replication on the subscription by using the Start Mirroring option*

When InfoSphere CDC mirroring starts to synchronize the source table with its shadow table, all functionality of the shadow table feature becomes available on the new HADR primary database.

From this point, applications and queries can benefit from the advantage of shadow tables after either a planned or an unplanned failover.

4

# Optim Query Workload Tuner and BLU Acceleration

This chapter demonstrates how IBM InfoSphere Optim Query Workload Tuner is used with DB2 with BLU Acceleration. If you have an existing DB2 9.7 or 10.1 data warehouse and are debating if a near-term upgrade to DB2 10.5 with BLU Acceleration is worthwhile, this tool can help you make decisions by estimating the performance increase for your analytic workloads when running with BLU Acceleration. If you have mixed workloads for your data environment and considering making column-organized tables as an alternative to speed your analytic workloads, this tool can also be used to help select column-organized table candidates for defining shadow table copies or for conversion to optimize analytic performance.

The following topics are covered:

- Planning and testing BLU Acceleration with IBM InfoSphere Optim Query Workload
- How the Workload Table Organization Advisor works
- Prerequisites
- Preparing an empty DB2 10.5 database with current objects and statistics using db2look
- Running the conversion recommendations from the advisor
- Optional: Selecting your own candidate tables for conversion analysis

# 4.1 Planning and testing BLU Acceleration with IBM InfoSphere Optim Query Workload

DB2 with BLU Acceleration delivers a set of breakthrough technologies and can improve performance of complex analytic queries dramatically. Testing with BLU Acceleration can be simple. With just a few commands, you can create an analytics-workload database and start loading your data to explore the benefits of BLU Acceleration in your environment.

In a data warehouse or data mart environment dedicated for analytic workloads, one can easily test and witness the performance improvements from loading all user data (or all tables that are referenced in complex analytic workloads) into BLU Acceleration column-organized tables. It is not necessary to use the Workload Table Organization Advisor (WTOA) in Optim Query Workload Tuner (OQWT) for column-organized table candidate recommendations. Although a mixture of row-organized tables and column-organized tables can coexist in the same database, BLU Acceleration provides best performance by joining column-organized tables of the same type.

Optim Query Workload Tuner is licensed with DB2 Advanced Editions. In the new version of Optim Query Workload Tuner 4.1.1, the GUI capability is enhanced to support BLU Acceleration. It comes with a Workload Table Organization Advisor that helps make decisions on table organization based on provided workloads.

In this chapter, we review two common use cases where you might find Optim Query Workload Tuner useful.

## 4.1.1 Use case 1: Databases with mixed workloads

One of the many unique advantages of DB2 with BLU Acceleration is that traditional row-organized tables and BLU Accelerated column-organized tables can coexist within one database. Businesses might need to run up-to-the-minute analytics directly from OLTP databases. With BLU Acceleration supported in mixed workloads environments, you can choose to convert or load only the analytic-focused tables into column-organized tables to speed analytic insights.

In cases where the same tables are accessed in both OLTP and OLAP workloads, users can benefit from the shadow table capability included in DB2 Cancun Release 10.5.0.4. Users can create column-organized shadow table copies with data replicated asynchronously from row-organized tables. As a result, OLTP workloads have the same regular transactional performance from accessing the original row-organized tables, while OLAP workloads referencing the same tables can be rerouted to column-organized shadow table copies for enhanced analytic performance.

If you have mixed workloads, you might choose to use Optim Query Workload Tuner 4.1.1 to help make test decisions and plan for a BLU Acceleration adoption. Workload Table Organization Advisor helps analyze ongoing workloads, and recommends table candidates that might benefit from storing in column-organized format. Users can then decide whether they should convert all recommended tables into column-organized tables or define those as shadow table copies in the same environment.

### 4.1.2 Use case 2: DB2 10.5 with BLU Acceleration upgrade

Mixed workloads environment is not the only use cases for Optim Query Workload Tuner. For existing DB2 9.7 or 10.1 data warehousing users, you can use Optim Query Workload Tuner to help decide whether a near-term upgrade to DB2 10.5 with BLU Acceleration is worthwhile for you. Workload Table Organization Advisor in Optim Query Workload Tuner uses cost-based approach to determine if existing row-organized tables would benefit from conversion to column-organized tables for a provided workload. It helps evaluate and understand the possible estimated performance gain when using BLU Acceleration. This helps make upgrade decisions easier.

## 4.2 How the Workload Table Organization Advisor works

When the advisor is run, the tool first determines the cost estimation for the workload with existing row-organized tables. For tables that are referenced in the provided workload, it virtually converts qualifying row-based tables into column-organized format, then compares the cost differences of the queries before and after the virtual column-organized conversion. It then provides column-organized table candidate recommendations if they have performance improvements higher than a user-defined minimum performance improvement threshold.

For those tables that BLU Acceleration does not support (for example, those that contain LOB and XML columns), the tool might recommend as shadow table candidates if DB2 engine finds improved performance for the supported columns to be stored as shadow tables.

Because the tool performs virtual conversion in the analysis, it does require the database being analyzed to have BLU Acceleration enabled. With that said, it is not necessary to have all data reside in the BLU-enabled DB2 10.5 database. A common scenario is to use **db2look** to replicate the current database catalog and statistics into a new, empty DB2 10.5 database with BLU Acceleration enabled. With the tool, you can capture a representative workload from existing database for the analysis. The advisor can then perform the workload table organization analysis using the replicated database catalog and statistics info.

Figure 4-1 shows a high-level workflow to capture workloads from current DB2 database and analyze the performance gains moving to DB2 10.5 with BLU Acceleration.

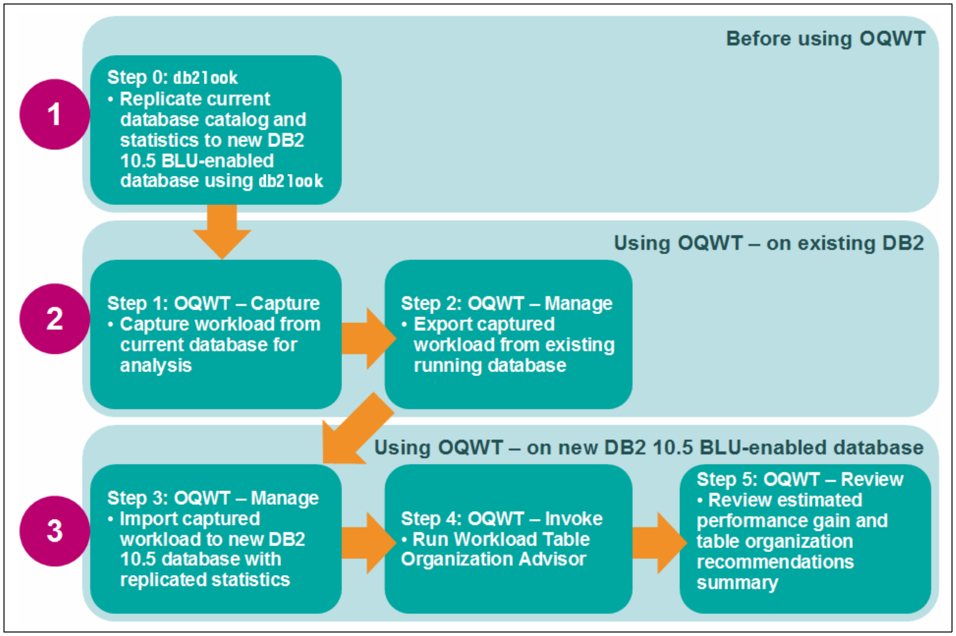



*Figure 4-1 Workflow for using Workload Table Organization Advisor in Optim Query Workload Tuner*

Of course, for those who already have a test database loaded in DB2 10.5, you can always run the Workload Table Organization Advisor directly from the test database without using **db2look**. Remember to enable BLU Acceleration parameters in the test database for Workload Table Organization Advisor to work properly.

## 4.2.1 Scenario used in this chapter

In this chapter, we demonstrate Optim Query Workload Tuner with a mixed workload use case. We use a Sample Outdoors database from Cognos, GS_DB, our scenario. Throughout the example, GS_DB101 represents a DB2 10.1 OLTP database that is currently being used; GS_DB105 represents a new, “empty” DB2 10.5 database that has the OLTP database catalog and statistics replicated.

The following major steps are involved in analyzing workloads with Optim Query Workload Tuner:

- Step 1: Capturing existing workloads for analysis

- Step 2: Managing a list of captured workloads
- Step 3: Running the Workload Table Organization Advisor
- Step 4: Reviewing the table organization summary

To avoid changes and impact in the currently running DB2 10.1 OLTP database, we first run steps 1 and 2 against the current database to capture and save a representative workload. Consequently, we run steps 2 to 4 to generate analysis using the replicated catalog and statistics on the DB2 10.5 database.

**Note:** Although this scenario uses Cognos BI workloads as an example, DB2 with BLU Acceleration is compatible with other business intelligence solutions.

## 4.3 Prerequisites

To use Workload Table Organization Advisor offered in Optim Query Workload Tuner, you must install the software. The following software is used for this demonstration:

- IBM InfoSphere Optim Query Workload Tuner 4.1.1 (entitled with any DB2 Advanced Editions)

  This includes the following two components from the Optim Query Workload Tuner installer:

  – IBM Data Studio client 4.1
  – A license activation kit for InfoSphere Optim Query Workload Tuner 4.1

  For more information about installing Optim Query Workload Tuner, go to the following location:

  `http://pic.dhe.ibm.com/infocenter/dstudio/v4r1/index.jsp?topic=%2Fcom.ibm.datatools.qrytune.installconfig.doc%2Ftopics%2Fioqwt41_top.html`

- IBM DB2 10.5 for Linux, UNIX, and Windows

  For a full list of DB2 editions that support BLU Acceleration, see Chapter 2, " Planning and deployment of BLU Acceleration" on page 17.

- IBM Cognos Business Intelligence Server 10.2 or 10.2.1

  If you have another business intelligence (BI) software solution in your current infrastructure, you can optionally replace the Cognos BI Server component with your own BI solution. Of course, you are welcome to try the Cognos BI solution as an alternative for your BI environment. All DB2 10.5 with BLU Acceleration supported editions include license entitlements to five authorized entitlement of Cognos BI.

For more information, see the DB2 10.5 with BLU Acceleration announcement letter:

http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&appname=iSource&supplier=897&letternum=ENUS213-210

## 4.4 Preparing an empty DB2 10.5 database with current objects and statistics using db2look

This section describes the preparation steps for users who are looking to use Workload Table Organization Advisor in Optim Query Workload Tuner without making any changes to the current non-BLU environment. For users who are planning to run the analysis on the same database, you can skip this preparation and go directly to 4.5, "Step 1: Capturing existing workloads for analysis" on page 144.

The Workload Table Organization Advisor in Optim Query Workload Tuner is the advisor to estimate performance improvements of the workload in row-organized or column-organized tables. The tool requires database catalog and statistics to complete the analysis. Therefore, it is not necessary to migrate all the data. The **db2look** command allows us to quickly mimic the current database with similar structures and statistics in an "empty" DB2 10.5 with BLU Acceleration database for analysis.

Complete the following steps:

1. Before extracting database objects and statistics on the current database in an existing DB2 database, ensure that statistics are updated. Otherwise, use **RUNSTATS** to update statistics of database objects in the system catalog before proceeding to the next steps.
2. On the current database, run the **db2look** command to extract the database object definitions and statistics into a file. Example 4-1 illustrates a sample **db2look** command that extracts all user-defined database objects and statistics along with registry variables into an output file.

   *Example 4-1 Sample db2look command*

   ```
   db2look -d GS_DB101 -a -e -m -l -x -f -o db2look.sql
   ```

   The command uses the following parameters:

   | | |
   |---|---|
   | -d GS_DB101 | Specifies the name of the database (in our example, GS_DB101). |
   | -a -e | Generates the DDL statements for all objects in the database. |

| | |
|---|---|
| `-m` | Runs in mimic mode; generates `UPDATE` statements to replicate statistics on tables, views, columns, and indexes. |
| `-l` | Generates DDL statements for user-defined objects such as table spaces, storage groups and buffer pools. |
| `-x` | Generates DDL statements for authorizations (optional for this use case). |
| `-f` | Generates statements to reproduce configuration parameters and registry variables that affect the query optimizer. |
| `-o db2look.sql` | Writes all generated statements to an output file (in our example, `db2look.sql`). |

3. In a DB2 10.5 test environment, create a new database instance and set `DB2_WORKLOAD=ANALYTICS`. Then, create a new database with the default analytics settings, as demonstrated in Example 4-2. For further details about how to create a new DB2 10.5 with BLU Acceleration database, see 2.5.2, "New database deployments" on page 26.

*Example 4-2 Create a new, empty DB2 with BLU Acceleration database*

```
db2set DB2_WORKLOAD=ANALYTICS
db2stop
db2start
db2 create db GS_DB105
```

4. Review the `db2look.sql` output file generated from step 2 on page 142. The script has `CONNECT TO` statements that connect to a database and re-create the extracted objects and statistics. Modify the script to connect to the newly created DB2 10.5 database from step 3. Review the same file and comment out all **`DBM CFG`** and **`DB CFG`** parameters that might disable BLU Acceleration. For parameters that are suggested in BLU Acceleration, see to 2.6, "Configuration preferred practices for BLU Acceleration deployment" on page 31.
5. On the newly created DB2 10.5 with BLU Acceleration database, run the `db2look.sql` script modified from step 4. Example 4-3 demonstrates a command to execute the updated `db2look.sql` script and reproduce the database objects and statistics.

*Example 4-3 Execute previously modified db2look script to reproduce database objects and statistics on new database*

```
db2 -tvf db2look.sql
```

You now have a DB2 10.5 with BLU Acceleration database that mimics your current database objects and statistics from an existing DB2 database. You are now ready to run Workload Table Organization Advisor and estimate performance increase of your current workload with DB2 10.5 with BLU Acceleration.

## 4.5 Step 1: Capturing existing workloads for analysis

Before Optim Query Workload Tuner can provide table organization analysis and recommendations on the current workload, you must first provide the tool a good representation of your existing workload for analysis.

Optim Query Workload Tuner can capture existing workloads from various sources. For example from package cache, event monitor tables, a SQL procedure, triggers that use to compile SQL statements, user-defined functions with compiled SQL statements, views, a SQL file and so on. In our example, we use the DB2 *package cache* to capture our current BI workloads. Any incoming dynamic and static SQL coming through DB2 database manager is regularly recorded in the DB2 package cache. This helps benefit different applications agents that access the same queries. In case of dynamic SQL query statements, it helps reduce the internal compilation costs in DB2. Because package cache has limited size, only recent statements are found here.

In our scenario, our Cognos BI users have been running their usual analytic workloads (that is, Cognos Dynamic Cube sample reports in our case) from Cognos. Analytic queries that require access to the underlying database are collected in the DB2 package cache. By capturing workloads in DB2 package cache, we capture the usual workloads executed by our BI users.

> **Tip:** You can also use other products in the IBM Optim tools suite with BLU Acceleration. For instance, you can use Optim Workload Replay to capture existing workloads as you always used to. Optim Query Workload Tuner can take those workloads, captured by Optim Workload Replay, to run table organization analysis. For more information, see this web page:
>
> https://www.ibm.com/developerworks/data/library/techarticle/dm-1405-db2blureplay/
>
> You can also use the Optim Performance Manager to monitor your workload performance.

To capture existing workloads for table organization analysis, follow these demonstration steps:

1. Open Data Studio with a new workload location.
2. By default, Data Studio opens the Database Administration perspective. Click the **Open Perspective** icon ( ) at the top-right corner. Select **IBM Query Tuning** in the Open Perspective window, as shown in Figure 4-2.

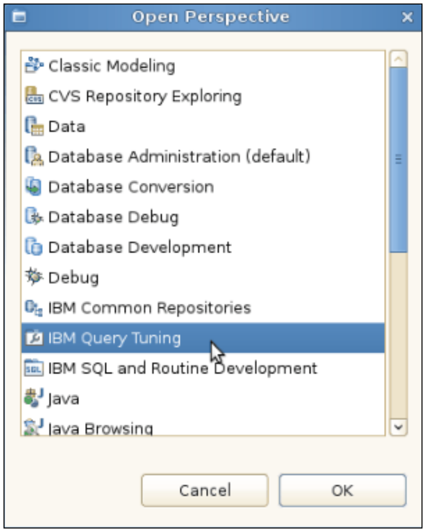



*Figure 4-2 Open IBM Query Tuning perspective*

3. From the Data Source Explorer panel, connect to a database by right-clicking the database name, clicking **Connect**, and providing database connection information.

4. In Data Source Explorer, right-click the database connection from the previous step, and select **Analyze and Tune** → **Start Tuning**, as shown in Figure 4-3.

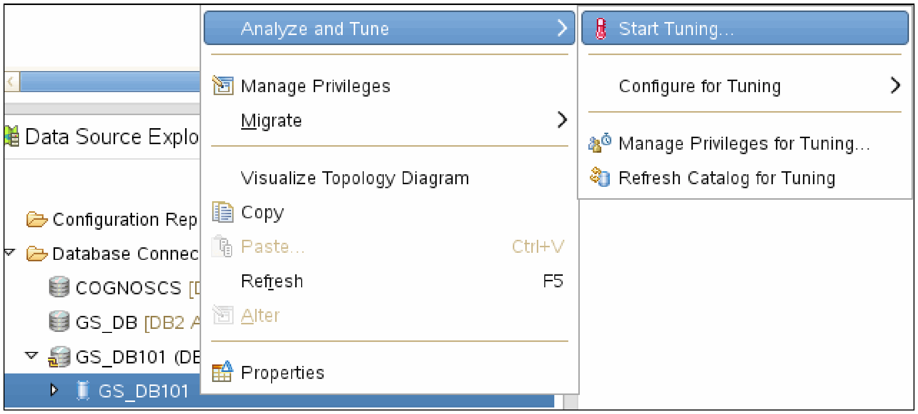



*Figure 4-3 Start Tuning option for database*

If the Analyze and Tune option is not available, ensure that Optim Query Workload Tuner License is enabled on the target database. By default, the Optim Query Workload Tuner License Activation Kit is included with the installation package. After the activation kit is installed, you can configure your DB2 databases for query workload tuning using IBM Data Studio client, as follows:

a. Run the **`License.sh`** file (on Linux) or **`License.bat`** file (on Windows) from the IBM Data Studio 4.1 installation directory. By default, Optim Query Workload Tuner Activation Kit is in the following directory, where `/opt/IBM/DS4.1.0/` is the default Data Studio installation directory (`DS_installdir`):

- On Linux:

```
<DS_installdir>/QueryTunerServerConfig/all_features/LUW/License
```

- On Windows:

```
<DS_installdir>\QueryTunerServerConfig\all_features\LUW\License
```

b. In Data Studio, right-click the database and select **Analyze and Tune** → **Configure for Tuning** → **Guided Configuration**.

For more information about configuring databases for query workload tuning and the necessary authorities and privileges, go to this web page:

http://pic.dhe.ibm.com/infocenter/dstudio/v4r1/topic/com.ibm.datatools.qrytune.installconfig.doc/topics/oqwt_configureluw.html

5. The Query Tuner Workflow Assistant panel opens, with the **2. Capture** view displayed, and is ready to capture workloads from various sources. Under DB2 for Linux, UNIX, and Windows Sources, select **Package Cache**. In this example, we capture workloads from the DB2 *Package Cache* for analysis (Figure 4-4). You can also select other workload source types available in the list.

   For details of each source types available for workload capturing, see the following web page:

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

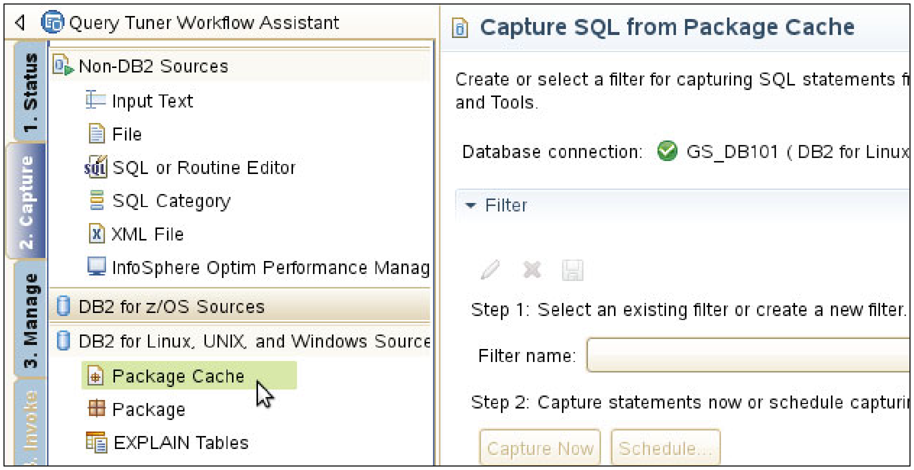



*Figure 4-4 Capture workloads from the DB2 Package Cache*

6. On the panel at the right side, the Capture SQL from Package Cache page opens. As shown in Figure 4-5, click **New** (under Step 1) to define a filter for collecting SQL statements from the DB2 package cache.

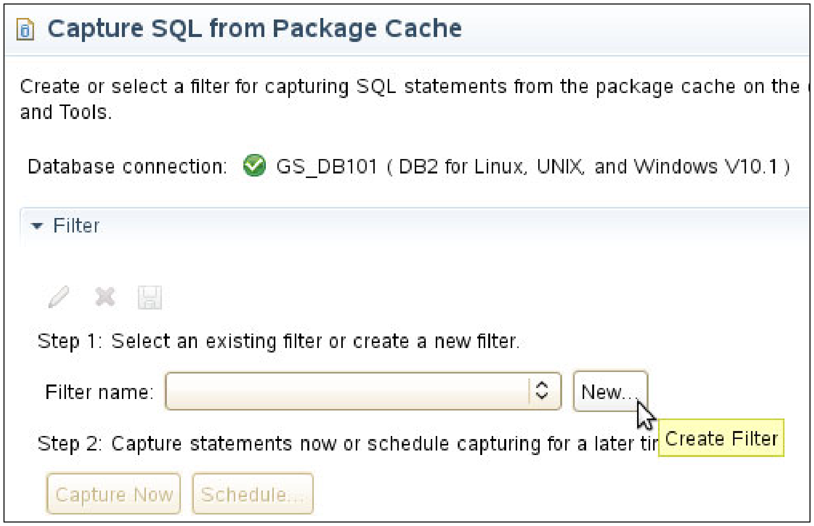



*Figure 4-5 Define new or existing filter for capturing workloads*

7. You can select a variety of filters. For example, dynamic or static statements, statements that run over certain milliseconds, statements that have been executed over a specific number of times, and so on.

   If you have an analytics environment and are looking to analyze possible performance improvements for only long, complex analytic queries, you can define a filter on statement execution time. Figure 4-6 illustrates filter definition with **STMT_EXEC_TIME > 60000 ms** as a demonstration, that is, any statements that have spent over 1 minute to execute. You can specify a different filter according to your environment and the workload type you want to capture.

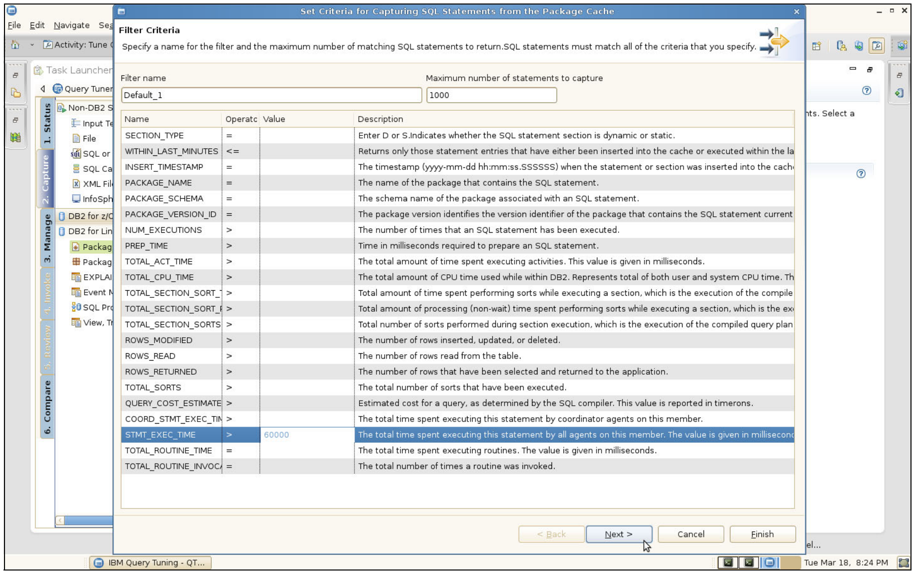


*Figure 4-6 Defining a filter on STMT_EXEC_TIME*

8. Go through the wizard to define the filters you want for capturing workloads. Then, click **Capture Now (**under Step 2) to start capturing workloads from DB2 package cache, as demonstrated in Figure 4-7.

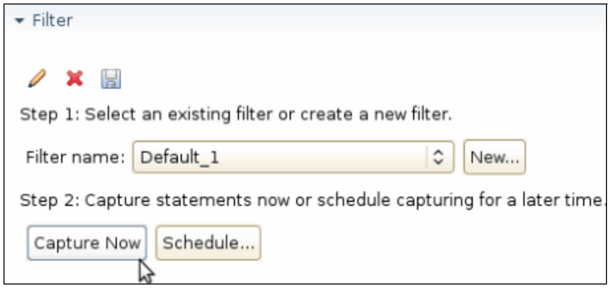



*Figure 4-7 Starting capturing workloads*

9. SQL statements that match the defined filter are displayed, as demonstrated in Figure 4-8.

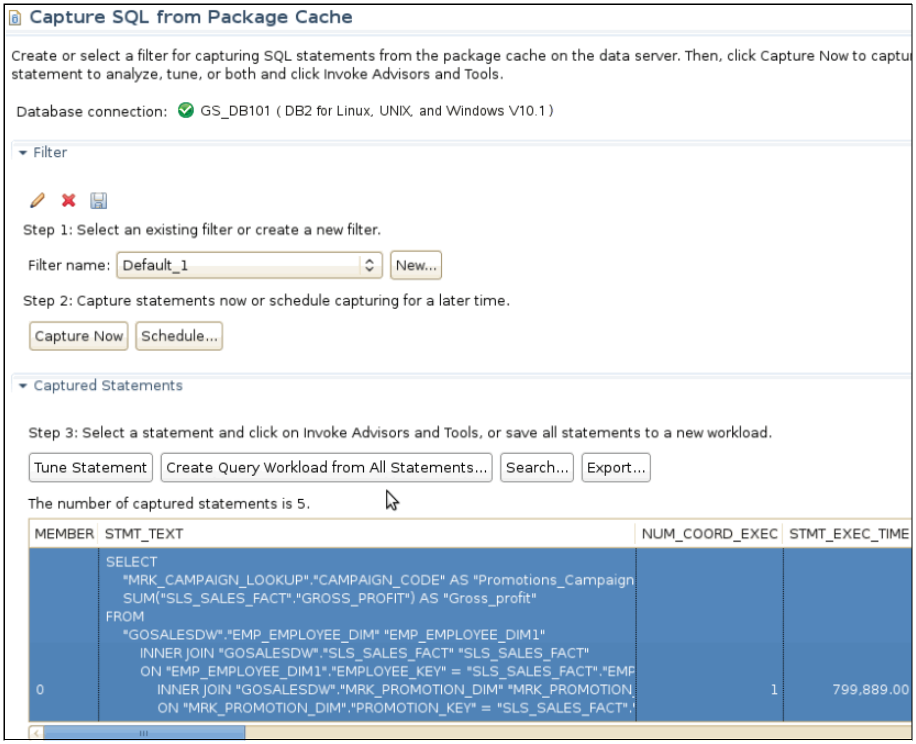



*Figure 4-8 Captured workloads*

10. Users can save all captured statements as a workload and proceed with the advisor tools. Click **Save All to Workload** to save the workload as a unique name, as shown in Figure 4-9.

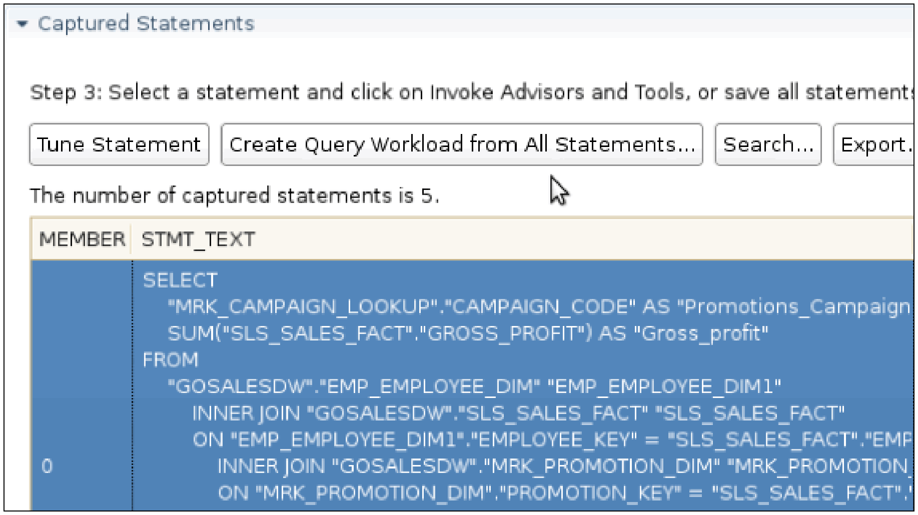



*Figure 4-9 Save captured statements to workload*

## 4.6 Step 2: Managing a list of captured workloads

After capturing representative workloads from a currently running database, you can export the captured workloads from the *Manage* tab. Subsequently, start a new Optim Query Workload Tuner project that connects to the DB2 10.5 database created in 4.4, “Preparing an empty DB2 10.5 database with current objects and statistics using db2look” on page 142, then import the captured workloads in the new project for further analysis.

For cases where the query workload analysis should be run from the same database (for example, the mixed workload use case), you can skip to 4.6.2, “Invoking Workload Table Organization Advisor” on page 153 for procedures to start the Workload Table Organization Advisor.

## 4.6.1 Exporting and importing captured workloads

In the **3. Manage** section of the workflow assistant, you can manage the list of previously captured workloads and invoke the desired advisors from here. For users who want to analyze the performance gains moving to DB2 10.5 with BLU Acceleration, complete the following steps to import captured workloads in a DB2 10.5 Optim Query Workload Tuner project:

1. To export captured workloads, in the **3. Manage** section, click **Export Workload** as illustrated in Figure 4-10. A pop-up window prompts for the location where the captured workloads should be exported. Save the workload in a `.zip` file.

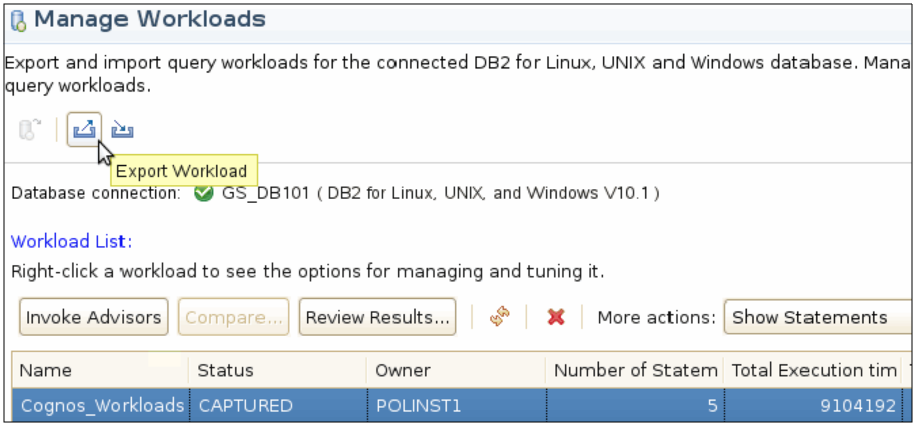



*Figure 4-10 Export captured workloads*

2. Follow step 4 on page 146 in 4.5, “Step 1: Capturing existing workloads for analysis” on page 144 to connect to a DB2 10.5 database and start a new Optim Query Workload Tuner project. To import captured workloads when a new Optim Query Workload Tuner project is connected to a DB2 10.5 database, click **Import Workload** in the **3. Manage** section, as illustrated in Figure 4-11. A pop-up window prompts for the workload to import. Open the previously exported workload from step 1 to import.

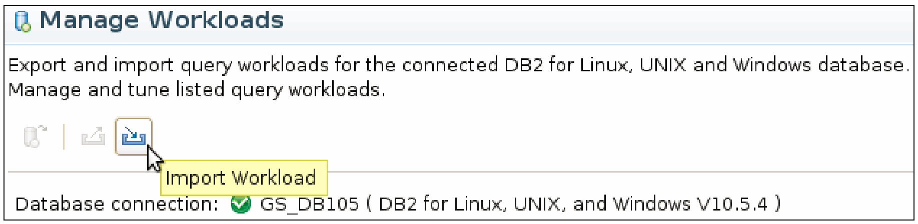



*Figure 4-11 Import previously captured workloads*

After the captured workload is imported into an Optim Query Workload Tuner project connected to a DB2 10.5 database, you are ready to run the Workload Table Organization Advisor to start the analysis.

## 4.6.2 Invoking Workload Table Organization Advisor

To start the Workload Table Organization Advisor, in the Manage view, select the captured workload and click **Invoke Advisors** to start the advisor tools, as shown in Figure 4-12. Alternatively, you can also select the workload and right-click, select **Invoke Workload Advisor and Tools** from the pop-up menu.

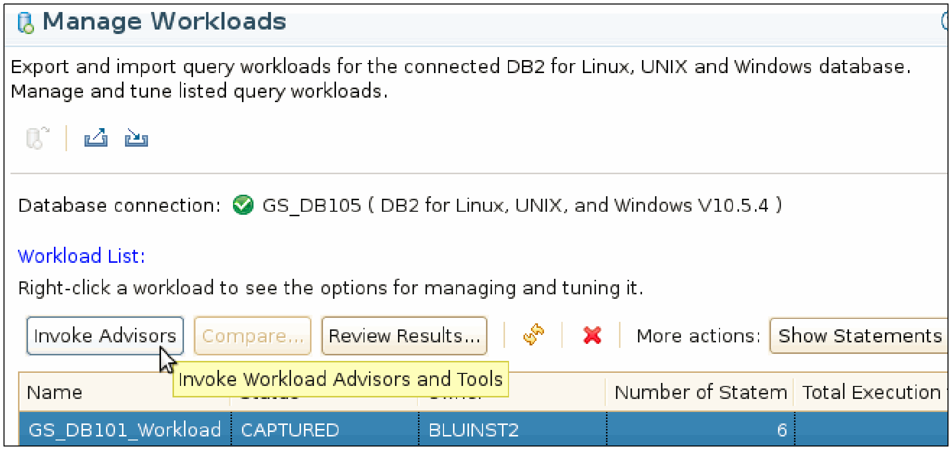



*Figure 4-12 Invoke Advisors for the selected workload*

## 4.7 Step 3: Running the Workload Table Organization Advisor

The *Invoke* view opens after you selected the Invoke Advisors option for your captured workload. In this view, you can select the options of which advisors to run for your workload. This is where you start the Workload Table Organization Advisor.

> **Note:** By default, Workload Table Organization Advisor provides recommendations for column-organized table candidates when estimated performance improvement of the workload is greater than a minimum threshold of 20%. To customize this minimum threshold for your analysis, go to **4. Invoke** → **Workload** → **Set Advisor Options** to change the **Minimum threshold for estimated performance improvement** setting before running Workload Table Organization Advisor.

Use the following steps to run Workload Table Organization Advisor:

1. The **4. Invoke** section opens after you start Workload Advisor as demonstrated in 4.6, "Step 2: Managing a list of captured workloads" on page 151. Select **Re-collect EXPLAIN information before running workload advisors** in the Run Workload Advisors page. This ensures that the advisors have access to latest EXPLAIN information for SQL statements in the workload. Click **Select What to Run**, as shown in Figure 4-13. A window opens where you can select which of the advisors to run for this workload.

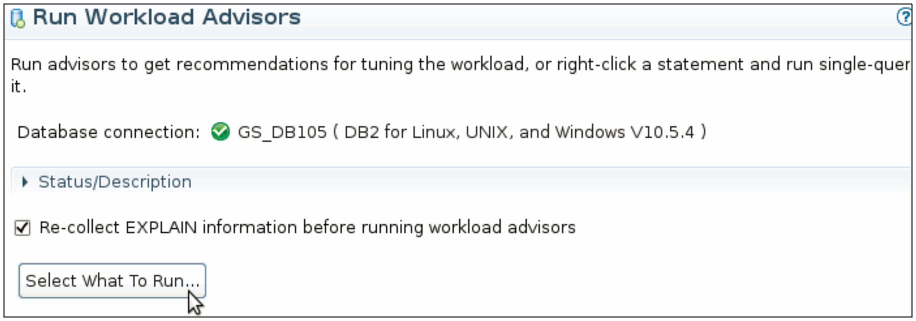



*Figure 4-13 Select what (advisor) to run for the captured workload*

2. In the Select Activities panel, select **Table Organization**, as demonstrated in Figure 4-14. The Workload Table Organization Advisor opens. It runs an analysis against current workload and generates recommendations for candidate tables that can benefit the workload when storing in column-organized tables rather than in row-organized.

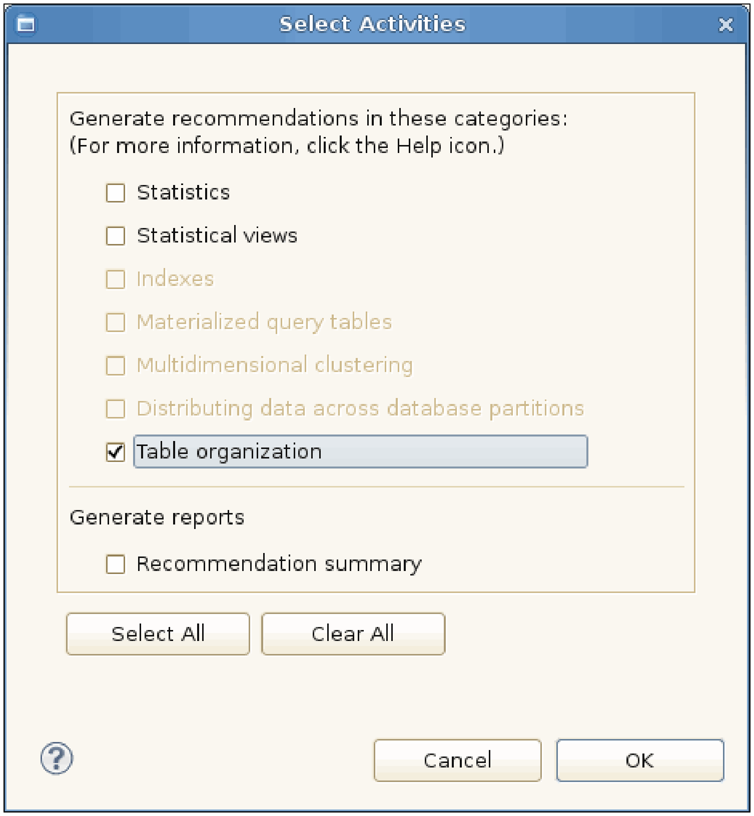



*Figure 4-14 Selecting the Table organization advisor*

**Note:** When **Table organization** is selected, some advisors are disabled. These advisors are not applicable to column-organized tables, therefore they cannot be run with Workload Table Organization Advisor at the same time. For example, if tables are recommended as column-organized by Workload Table Organization Advisor, indexes are not required and its relevant option disabled. BLU Acceleration automatically creates and maintains metadata objects within DB2 to achieve fast performance and great storage savings.

**Important:** If you are running Workload Table Organization Advisor on the same database you captured workload on, before using the advisor to collect table organization recommendations, run statistics advisor twice to ensure that the Workload Table Organization Advisor has the latest RUNSTATS statistics for the analysis. This ensures more accurate estimates for the workload analysis. *Note that this is not required when advisor is running against an empty database with statistics replicated from another existing database.*

To run statistics advisor, select **Statistics** in the Select Activities panel (Figure 4-14 on page 155). The advisor provides recommended **`RUNSTATS`** commands if statistics are required. Follow the wizard to run **`RUNSTATS`** using the Statistics Advisor tool. Repeat a second time until the advisor gives no new recommendations for statistics. Subsequently invoke Workload Table Organization Advisor for the actual column-organized table recommendation analysis.

3. Optionally, specify the filters in the Collect EXPLAIN Information panel and click **Start EXPLAIN** to continue. If in doubt, leave it blank and click **Start EXPLAIN**.
4. You might see a warning recommending that the Workload Statistics Advisor be run twice before Workload Table Organization Advisor. If you ensured that the catalog statistics are updated at this point, read the warning and click **Continue**. This is a reminder to ensure that statistics information is most updated before running Workload Table Organization Advisor.

# 4.8 Step 4: Reviewing the table organization summary

When Workload Table Organization Advisor is running, it *virtually* converts tables from row-organized to column-organized and generates cost estimates before and after the conversion. When the cost improvement is better than the user-defined minimum performance improvement threshold, the advisor provides recommendations on ideal candidates to store in column-organized format for the provided workload. For those tables that cannot be converted to column organized (such as tables that contain LOB/XML columns), the advisor analyzes whether the supported columns are ideal candidates as shadow tables. When the advisor finishes the analysis, you are ready to review the advisor summary.

To review the table organization summary, continue with these steps:

1. The **5. Review** tab opens after the advisor completes running. On the **Review Workload Advisor Recommendations** page, summary of the advisor results are listed under the **Summary** tab. In this example (Figure 4-15), new recommendations are generated for table organization. You can double-click **Table Organization** under the Summary tab or click the **Table organization** tab to view the detailed recommendations.

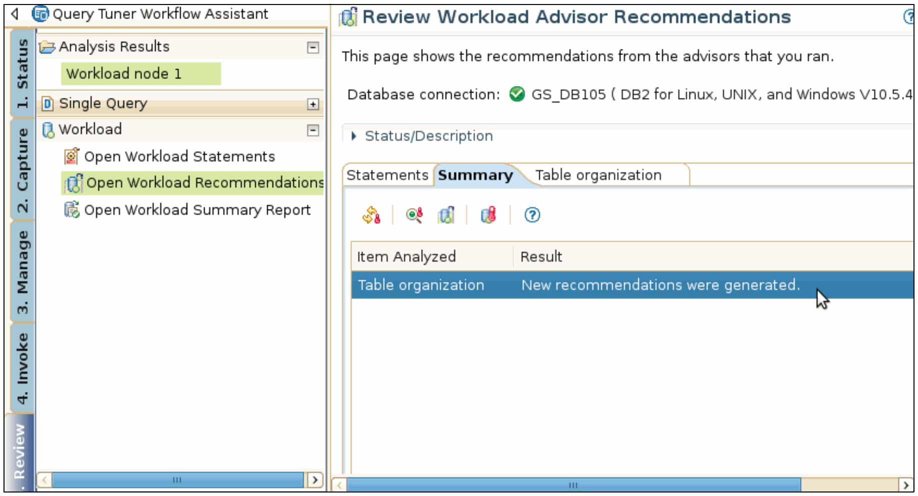



*Figure 4-15 Workload Advisor Recommendations summary view*

2. The **Table organization** tab shows a list of tables. These tables are identified as candidate tables that can benefit from column-organization. In this scenario (Figure 4-16 on page 158), all ten tables found in the workload are recommended to be converted as column-organized tables. An estimated performance improvement is displayed at the top of the page. In this example, advisor estimated that storing the listed candidates in column-organized tables will likely gain estimated performance improvement of 74%.

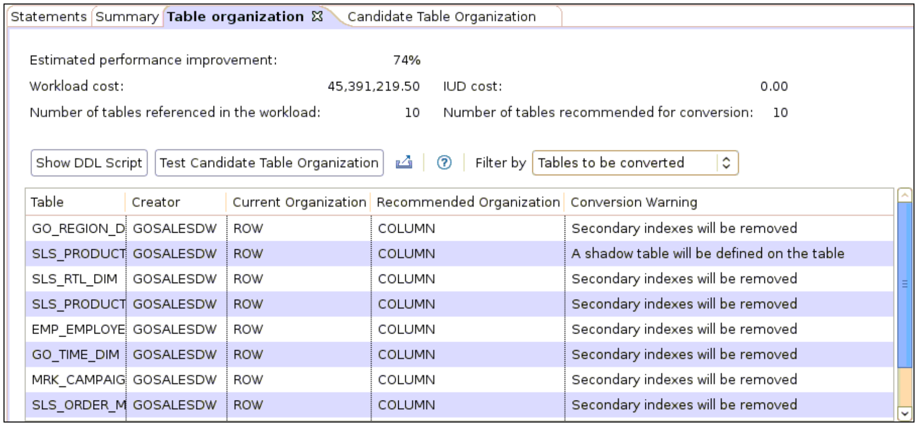



*Figure 4-16 Table Organization Advisor Results*

If you are evaluating only the estimated performance gain to evaluate if a near-term upgrade to DB2 10.5 with BLU Acceleration is worthwhile, you can stop here and use this possible improvement estimation of your workload as part of your decision making. Because this estimated number is generated from estimated costs from DB2 engine when it mimics column-organized tables in the catalog, the actual performance when data is loaded into column-organized tables can be different.

Proceed with a regular test cycle to check the performance improvement with data loaded. If the database you are considering to upgrade is purely for analytic workloads, attempt to test load all your data being used in your queries into column-organized tables, even though the advisor might only suggest a subset as candidate tables. Queries generally perform optimally with tables of the same type.

Continue with the next steps if you are using the tool to select a list of candidate column-organized tables in your mixed workloads environment.

3. The top grid of the **Table organization** tab (Figure 4-17 on page 159) shows a list of recommended column-organized tables, along with the *Conversion warning* about the suggested row- to column-organization conversion. These changes are related to the characteristics of BLU Acceleration. For example, user-defined indexes are not required in BLU Acceleration, therefore, any existing secondary indexes are removed when a row- to column-organization conversion takes place. Enforced referential integrity (RI) constraints are changed to `NOT ENFORCED`. These not-enforced key constraints require less storage and time to create, and are beneficial to the BLU Acceleration query optimizer.

By default, this list is filtered to only display the list of tables recommended for column-organization tables. To view all the tables referenced in the workload, or tables that are recommended to keep their original table organization, you can optionally select the desired options from the “Filter by” drop-down list.

4. If you scroll the list of advisor-recommended tables to the right, as demonstrated in Figure 4-17, you see these columns:

   – The *Findings* column indicates the reasons why the list of tables are suggested to be converted (or not). A common finding is improved performance for column-organized table recommendation.

   – The *IUDM Statements* column specifies the number of INSERT, UPDATE, DELETE, and MERGE statements that reference the table in the workload.

   – The *Cumulative Total Cost* column indicates the total CPU cost spent on the tables in the captured workload.

   – The *Reference to Table* column states the number of times the table is referenced in the captured workload.

| References to Table | Cumulative Total Cost | IUDM Statements | Finding |
|---|---|---|---|
| 3 | 86,661,418.00 | 0 | Conversion would improve performance. |
| 1 | 29,480,748.00 | 0 | Adding a shadow table can help analytical queries on the tabl |
| 3 | 86,661,418.00 | 0 | Conversion would improve performance. |
| 2 | 57,852,856.00 | 0 | Conversion would improve performance. |
| 6 | 171,597,890.00 | 0 | Conversion would improve performance. |
| 3 | 85,808,778.00 | 0 | Conversion would improve performance. |
| 1 | 27,820,066.00 | 0 | Conversion would improve performance. |
| 2 | 57,232,596.00 | 0 | Conversion would improve performance. |

*Figure 4-17 Recommended column-organized table candidates and advisor findings*

You might see some recommendations are listed as row- to column-organized conversion candidates, and some are listed as shadow table candidates. For any tables that can benefit from column-organized store, Workload Table Organization Advisor first recommends *converting* tables from row-organized into column-organized. Subsequently it considers the rest as possible *shadow table* candidates if applicable. Usually when the table is not compatible for a conversion (for example, when table has LOB, XML columns), or when the advisor finds that there are enough OLTP workloads referencing the table. In the advisor, no tables are recommended as both column-organized table conversion and shadow table candidates. But it is possible to define shadow tables for those that are recommended for a conversion.

**Tip:** For users who have mixed workloads and converting tables into column-organized is not an option in an OLTP environment, consider defining shadow tables for all the recommended column-organized candidates instead of an actual row to column-organized conversion. This will leave the original row-organized tables unaffected in OLTP performance, while allowing DB2 optimizer to reroute complex analytic workloads to column-organized shadow table copies to speed analytic reporting.

**Note:** Defining shadow tables requires an InfoSphere Change Data Capture configuration. For more details, see Chapter 3, “Planning and deployment of BLU Acceleration shadow tables for mixed workload environments” on page 53.

5. In the lower tab labeled **SQL Statements Affected** (Figure 4-18), all SQL statements in the workload that are affected if these tables are converted to column-organized tables are shown. In the grid, you can see the statement texts and the number of times the SQL statement is run in the captured workload, the weight cost of the SQL statement in the entire captured workload, estimated percentage of performance gain, and the estimated costs of statement execution before and after conversion. This order of the list is shown by descending weight cost, that is, the computing cost of the SQL statement out of the entire captured workload.

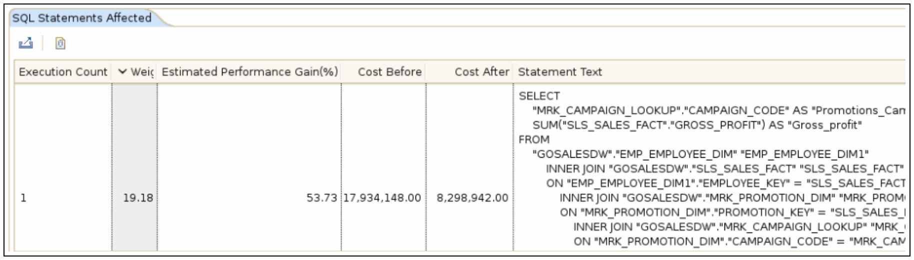



*Figure 4-18 SQL Statements Affected tab*

## 4.9 Running the conversion recommendations from the advisor

The tool not only provides table organization recommendations, but it can also generate a DDL script to execute the recommended changes.

After reviewing the detailed Workload Table Organization Advisor results, you can generate a DDL script of the recommended changes for further BLU Acceleration testing. From here, you can save the script, customize, and run it on your BLU Acceleration test environment when ready. For users who are already connected to their BLU Acceleration test database using the tool, you can modify and execute the table conversion script within IBM Data Studio. During a row- to column-organized table conversion, the original tables remain accessible online. It is important to note that data server configuration, IBM InfoSphere Change Data Capture configuration, subscription, and table mappings are required to use Shadow Tables.

Alternatively, after you have made decisions based on the advisor recommendations, you can manually create shadow tables or to convert existing row-organized tables to column-organized format on your own. For instructions on creating shadow tables, see Chapter 3, “Planning and deployment of BLU Acceleration shadow tables for mixed workload environments” on page 53. For manual conversion, you can use the **db2convert** command, as described in 2.6.5, “Converting tables to column-organized tables” on page 39. This command calls the same **ADMIN_MOVE_TABLE** stored procedure as in the script generated by Workload Table Organization Advisor.

If you prefer to continue with manual analysis, you can optionally select your own list of tables for further analysis by the tool, as described in 4.10, “Optional: Selecting your own candidate tables for conversion analysis” on page 165.

**Important note for table organization conversion:** Row-organized to column-organized table conversion is a one-direction conversion. That is, after a row-organized table is converted to column-organized, no DB2 utilities or tools support the backward conversion to the original row-organized format. To undo the conversion, unload the data from the converted column-organized tables, then reload into new row-organized tables.

A good practice is to perform a full backup of the database or the affected table spaces before a row- to column-organized conversion. An alternative to a conversion is *create-and-load*. That is, create a new set of selected tables in a new column-organized table format, and load data into the new column-organized tables, as described in 2.6.5, “Converting tables to column-organized tables” on page 39.

**Important note for shadow tables creation:** InfoSphere Change Data Capture configuration is required to properly map and replicate shadow tables from original row-organized tables. For more details, see Chapter 3, “Planning and deployment of BLU Acceleration shadow tables for mixed workload environments” on page 53.

When you are ready to convert the recommended tables to column-organized tables, complete the following steps:

1. Use **Show DDL Script** to generate a script that can be used to convert the list of recommended tables from row-organized to column-organized, as shown in Figure 4-19.

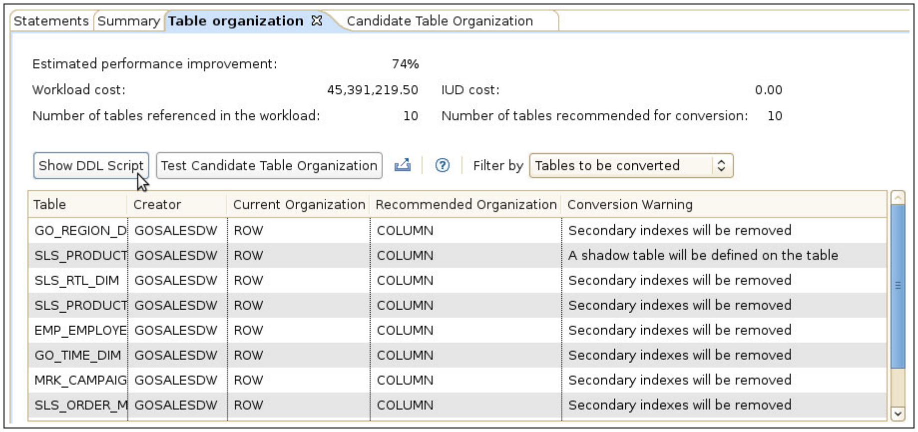



*Figure 4-19 Generate DDL script for recommended column-organized table conversion*

2. The Show DDL Script panel opens as illustrated in Figure 4-20. In this panel, review the column-organized table conversion command options. Here, you can choose the table conversion type, target table spaces for the column-organized tables, and constraint option. Note the **Constraint Option** section. All enforced referential integrity and check constraints should be converted to `NOT ENFORCED` for any new column-organized tables.

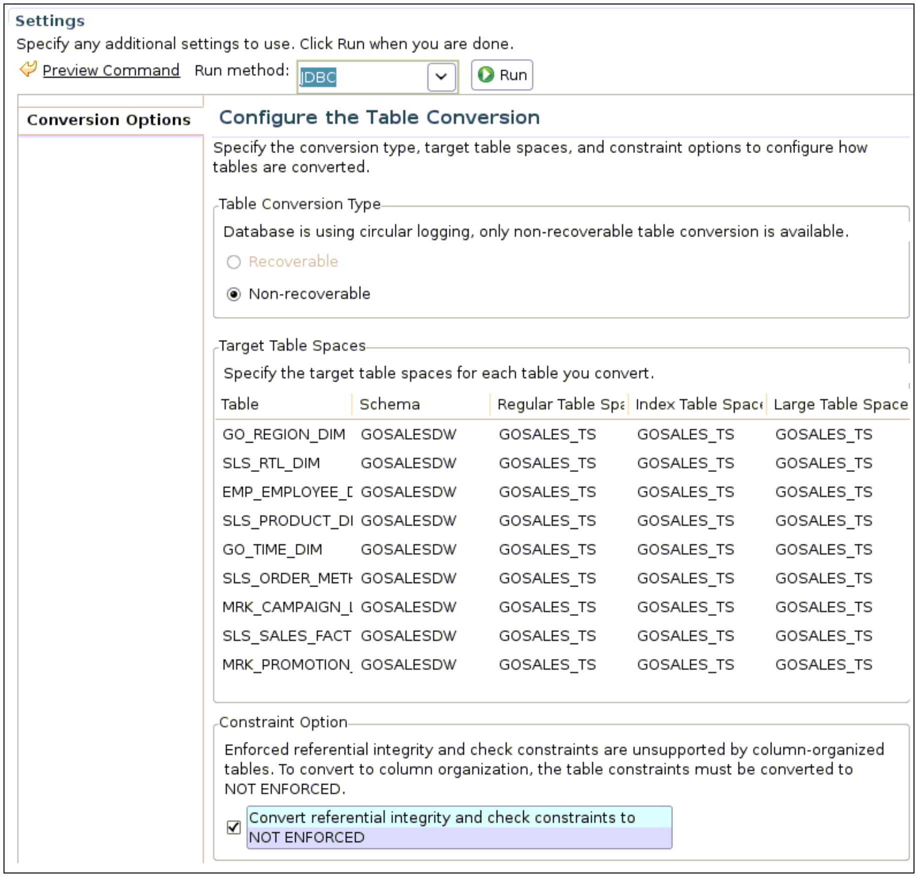



*Figure 4-20 Show DDL Script panel - Column-organized table conversion options*

Click **Preview command** to preview the generated DDL script.

3. The command section opens at the bottom, as illustrated in Figure 4-21.

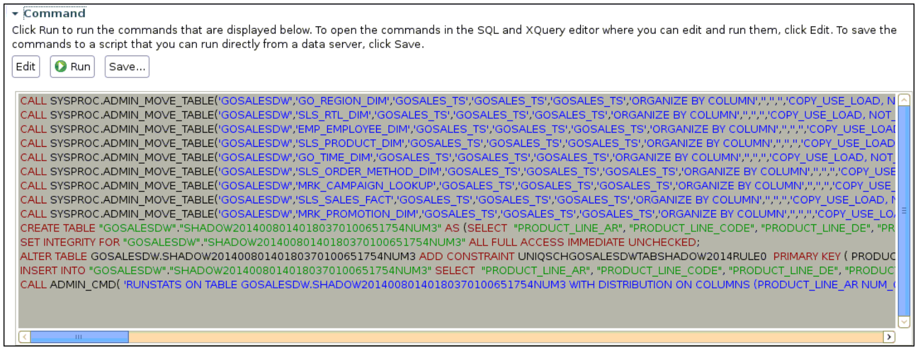



*Figure 4-21 Preview command in Show DDL Script panel*

DDL statements for the recommended conversion are displayed.

For suggested row-organized to column-organized table conversions, the generated statements call the **ADMIN_MOVE_TABLE** stored procedure to convert row-organized tables to column-organized tables. It is the same procedure that is used behind the scenes in the **db2convert** command.

For recommended shadow tables, the script generates the **db2 create table** commands to create the replicated shadow tables, **set integrity** statements to bring them out of check pending state, If there are any primary key or unique constraints, they will be added. The script also generates **insert** statements and **runstats** commands to update the shadow tables. IBM InfoSphere Change Data Capture configuration is required to set up proper mapping and replication of the shadow tables. For detailed procedures about how BLU Acceleration shadow tables are set up, see Chapter 3, “Planning and deployment of BLU Acceleration shadow tables for mixed workload environments” on page 53.

Click **Save...** to save the DDL and customize for your BLU Acceleration tests.

Optionally, users who are already connected to test databases can click **Run** to execute the column-organized table conversions in place. It is important to note that row-organized to column-organized table conversions (**ADMIN_MOVE_TABLE**) are permanent. You cannot convert column-organized tables back to row-organized table.

# 4.10 Optional: Selecting your own candidate tables for conversion analysis

Workload Table Organization Advisor also comes with a test candidate feature. This feature allows users to virtually convert table candidates to column-organized tables, or to virtually define shadow tables. It then generates an estimated performance improvement for review.

To mark your own tables for further analysis, follow these steps:

1. Click **Test Candidate Table Organization** to start manually selecting candidates for the analysis, as shown in Figure 4-22.

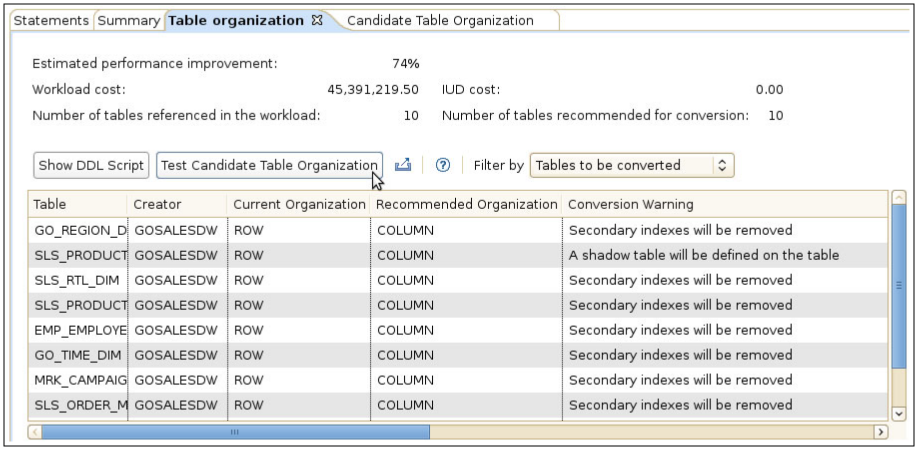



*Figure 4-22 Test Candidate Table Organization button*

2. The **Workload Test Candidate Table Organization** page opens (Figure 4-23). All tables referenced by the captured workload are listed. To manually change the table candidate type, click a table, then select **Change Candidate Table Type**.

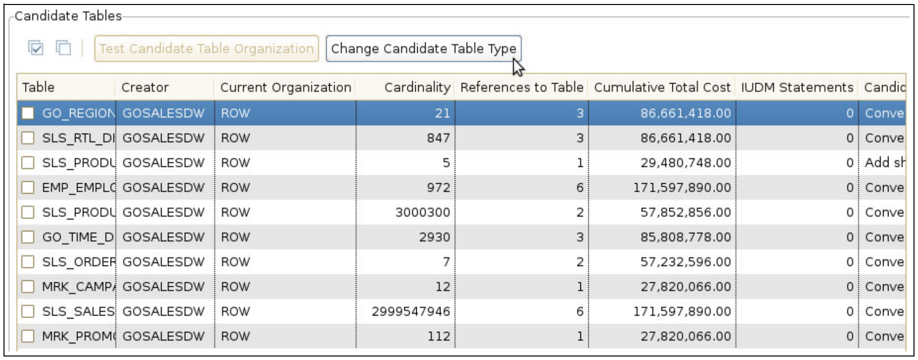



*Figure 4-23 Change candidate table type*

**Tip:** When selecting your own tables for column-organized table conversion, consider that tables with the following properties generally benefit from column-organized tables:

- *Large number of rows:* You have a large table with a high number of rows.
- *References to table*: Many SQL statements from the analyzed workload access this table.
- *High cumulative total cost*: Significant computing cost is spent on this table.
- *Low IDUM value*: The table has few or none of INSERT, UPDATE, DELETE, and MERGE operations.
- *Low column access*: A small number of columns are accessed by queries in analyzed workload.

Also, keep in mind that generally queries perform faster joining tables of the same type.

3. A pop-up window allows users to select the table type to virtually test for the selected table. In Figure 4-24, we choose to define all listed tables as shadow tables as an example.

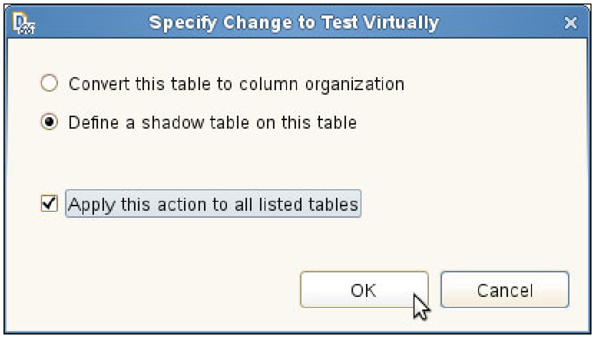
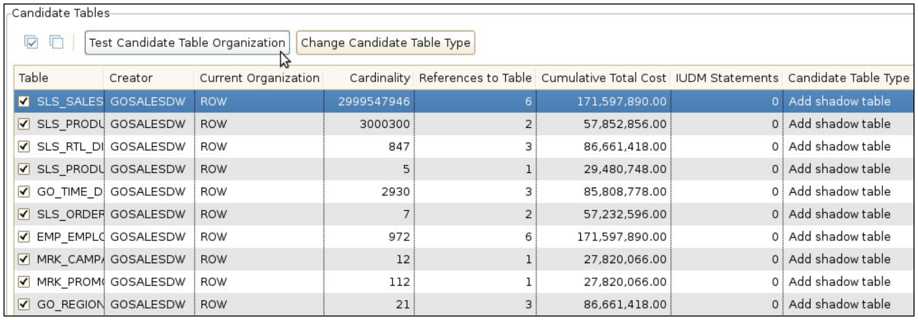



*Figure 4-24 Specify table type to test virtually*

4. After you select the tables you want, click **Test Candidate Table Organization** to start the analysis (Figure 4-25).




| Table | Creator | Current Organization | Cardinality | References to Table | Cumulative Total Cost | IUDM Statements | Candidate Table Type |
|---|---|---|---|---|---|---|---|
| SLS_SALES | GOSALESDW | ROW | 2999547946 | 6 | 171,597,890.00 | 0 | Add shadow table |
| SLS_PRODU | GOSALESDW | ROW | 3000300 | 2 | 57,852,856.00 | 0 | Add shadow table |
| SLS_RTL_DI | GOSALESDW | ROW | 847 | 3 | 86,661,418.00 | 0 | Add shadow table |
| SLS_PRODU | GOSALESDW | ROW | 5 | 1 | 29,480,748.00 | 0 | Add shadow table |
| GO_TIME_D | GOSALESDW | ROW | 2930 | 3 | 85,808,778.00 | 0 | Add shadow table |
| SLS_ORDER | GOSALESDW | ROW | 7 | 2 | 57,232,596.00 | 0 | Add shadow table |
| EMP_EMPLO | GOSALESDW | ROW | 972 | 6 | 171,597,890.00 | 0 | Add shadow table |
| MRK_CAMP/ | GOSALESDW | ROW | 12 | 1 | 27,820,066.00 | 0 | Add shadow table |
| MRK_PROM( | GOSALESDW | ROW | 112 | 1 | 27,820,066.00 | 0 | Add shadow table |
| GO_REGION | GOSALESDW | ROW | 21 | 3 | 86,661,418.00 | 0 | Add shadow table |

*Figure 4-25 Test Candidate Table Organization*

5. When the test candidate table organization analysis is completed, the Review Workload Advisor Recommendations page opens. Under the Summary tab, a list of previously run analysis results are shown. In the following example (Figure 4-26), new results are generated for Candidate Table Organization. Double-click **Candidate Table Organization** in the list, or click **Candidate Table Organization** on top to review the detailed analysis results.

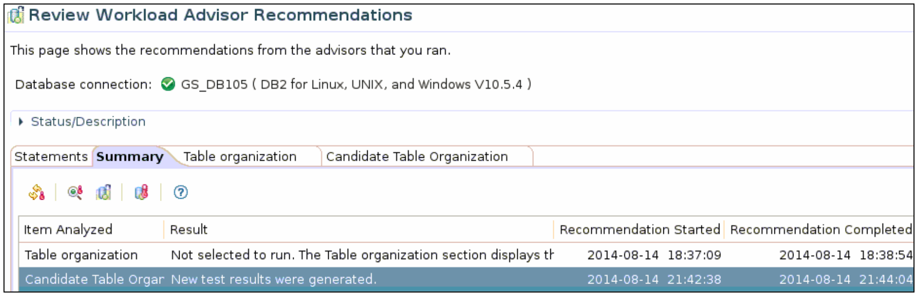



*Figure 4-26 Review Candidate Table Organization analysis results*

6. In the **Candidate Table Organization** tab, an estimated performance improvement is shown with similar details as described in 4.8, “Step 4: Reviewing the table organization summary” on page 157. In this scenario, (Figure 4-27), the manual analysis achieves 71% performance improvement for the given workload.

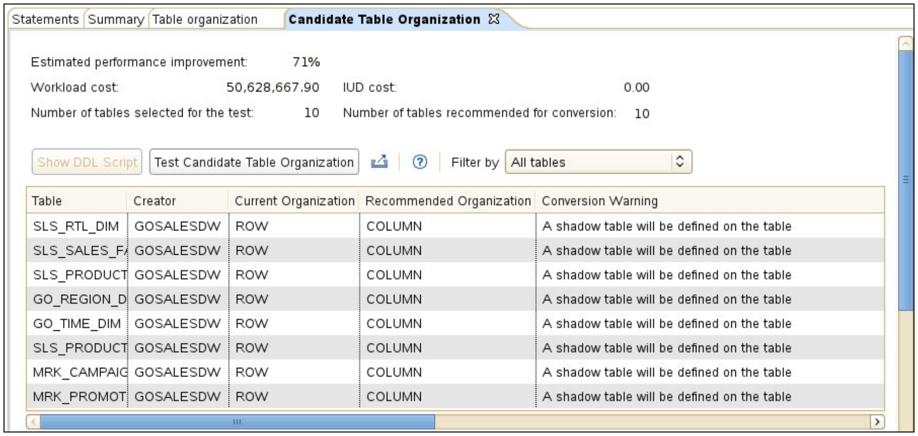



*Figure 4-27 Candidate Table Organization analysis results*

7. You can optionally continue to test your own selection of candidate table organization analysis until a decision is made by repeating steps from 4.10, “Optional: Selecting your own candidate tables for conversion analysis” on page 165 until you are ready to proceed to carry out a BLU Acceleration test.

5

# Performance test with a Cognos BI example

With the new technologies such as parallel vector processing, core-friendly parallelism, and scan-friendly memory caching, the DB2 with BLU Acceleration feature optimizes the best use of the hardware in your existing infrastructure for analytics. With adaptive compression and data skipping capabilities on columnar store, BLU Acceleration provides another level of performance for complex analytic workloads.

After a BLU Acceleration deployment, you might be eager to test the immediate performance gain that BLU Acceleration can provide to your analytic workloads. Although users are open to using their own testing tools, in this chapter, we demonstrate a built-in DB2 utility and a Cognos tool (*specific for Cognos users*) that you might find helpful to measure your query performance for testing or evaluation purposes.

Although this chapter uses Cognos BI workloads as an example, DB2 with BLU Acceleration can also benefit other analytic workloads generated by other business intelligence vendors.

The following topics are covered:

- Testing your new column-organized tables
- DB2 benchmark tool: db2batch command
- Cognos Dynamic Query Analyzer
- Conclusion

# 5.1 Testing your new column-organized tables

Testing is an essential and important phase in a deployment project. After you convert, load, or replicate your data into an analytic-optimized column store in DB2, you might be excited to experience the performance gains and storage saving for your environment. Users can use their own favorite benchmarking tool for testing their workload performance before or after their BLU-deployment. A helpful alternative is to use the built-in DB2 utility or a Cognos BI tool to help you measure performance results of your workloads.

In 5.2, “DB2 benchmark tool: db2batch command” on page 171, we use the built-in **`db2batch`** command to quickly demonstrate the performance comparisons of our Cognos BI workload, against original row-organized tables and the new BLU column-organized tables. In 5.3, “Cognos Dynamic Query Analyzer” on page 175, we perform a similar comparison with Cognos Dynamic Query Analyzer.

Besides the **`db2batch`** utility, DB2 also includes a variety of monitoring metrics that you can also use for determining query run times. For more details about these monitoring metrics, or for observing storage savings and compression rates of your new BLU column-organized tables, see Chapter 6, “Post-deployment of DB2 with BLU Acceleration” on page 187.

> **Note:** Performance measuring results illustrated in this chapter are for functional demonstration only. They are not being used as a benchmarking reference.

## 5.1.1 Scenario environment

Here we describe the lab environment for the examples used in this section.

### Hardware used in our scenario

One of the many advantages of BLU Acceleration technologies is that acquiring new hardware to run BLU is unnecessary. For best performance, we advise you to follow the suggested hardware from 2.4, “Prerequisites” on page 20. For this book, we used an existing System x3650 with non-solid-state drives (non-SSDs) available in our lab to complete our functional scenario. The system had less than the minimum memory recommended for the number of cores in it. Even without the SSDs and the recommended memory requirements, we observed a 6 - 20 times improvement in query response time during the test.

### Sample workload used in our scenario

The workload we used in this example is mainly composed of six GO Data Warehouse Sales queries from the Cognos Dynamic Cubes samples pack, as illustrated in Figure 5-1 on page 171. These Dynamic Cubes samples are based on the sample database model from the Cognos sample `GS_DB` database. In our example, our sales fact table, `GOSALESDW.SLS_SALES_FACT`, is expanded to 3,000,000,000 (three billion) rows for the sole purpose of this functional demonstration.

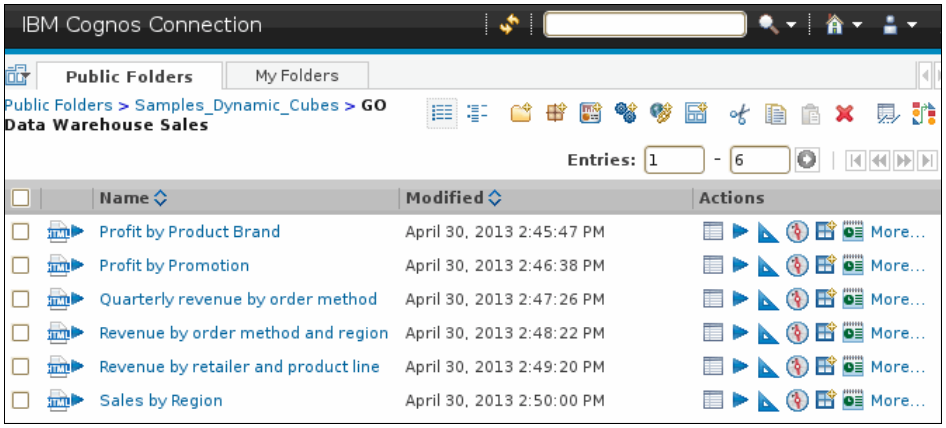



*Figure 5-1 Sample queries used in this example*

For more details about the IBM Cognos Business Intelligence Sample Outdoors samples and setup instructions, visit the following website:

http://www-01.ibm.com/support/knowledgecenter/SSEP7J_10.2.1/com.ibm.swg.ba.cognos.inst_cr_winux.10.2.1.doc/c_instsamplesoverview.html?lang=en

## 5.2 DB2 benchmark tool: db2batch command

DB2 provides a **db2batch** command that is used for benchmarking SQL or XQuery statements in all editions. This built-in DB2 command takes queries from either a flat or a standard input. Then, it dynamically prepares the statements and returns the query results and the required time to run the provided statements.

Before and after loading the Cognos BI sample data into BLU column-organized tables, we run the **db2batch** command to capture the performance results of the captured Cognos sample query workloads, as demonstrated in Example 5-1.

*Example 5-1 db2batch benchmark tool example*

```
db2batch -d GS_DB -f DQworkloads.sql -iso CS -r results.out,summary.out
```

In Example 5-1, we specify several options for the **db2batch** benchmarking tool command:

`-d GS_DB` Specifies the name of the database (in our example, `GS_DB`)

`-f DQworkloads.sql` This input file contains the SQL statement to be run by the tool dynamically (in our case, `DQworkloads.sql`)

`-iso CS` Sets the cursor isolation level to cursor stability (`CS` for ODBC read committed). This determines how data is locked and isolated from other processes while being accessed. Another available isolation level is uncommitted read (`UR`)

`-r results.out,summary.out`

No spaces exist between the two file names and the comma.

Specifies the query results to be returned in the `results.out` file; and the performance summary to be returned in a separate file called `summary.out`

With a few options to the **db2batch** command, we can determine the performance information about the query workload on the database level.

**Tip:** To reduce impact of writing to disk during the performance benchmark, put a limit on the number of rows that should be written to the results output file. To do so, use the `--#SET ROWS_OUT <#_of_rows_in_resultsfile>` control option in the workload SQL input file. For more information, see the following web page:

https://www-304.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.cmd.doc/doc/r0002043.html?cp=SSEPGG_10.5.0%2F3-5-2-6-14&lang=en

Example 5-2 shows an excerpt of the `DQworkloads.sql` input file. The following control option is added to limit the number of rows written to the results output file:

```
--SET ROWS_OUT <#>
```

*Example 5-2 Example workload SQL input file for db2batch*

```
--#SET ROWS_OUT 10
SELECT "SLS_PRODUCT_DIM"."PRODUCT_BRAND_KEY" AS
"Product_brand_Product_brand_key",
        "GO_TIME_DIM2"."CURRENT_YEAR" AS "Time_Year",
        SUM("SLS_SALES_FACT"."GROSS_PROFIT") AS "Gross_profit"
FROM "GOSALESDW"."EMP_EMPLOYEE_DIM" "EMP_EMPLOYEE_DIM1" INNER JOIN
"GOSALESDW"."SLS_SALES_FACT" "SLS_SALES_FACT" ON
"EMP_EMPLOYEE_DIM1"."EMPLOYEE_KEY" = "SLS_SALES_FACT"."EMPLOYEE_KEY"
INNER JOIN "GOSALESDW"."GO_TIME_DIM" "GO_TIME_DIM2" ON
"GO_TIME_DIM2"."DAY_KEY" = "SLS_SALES_FACT"."ORDER_DAY_KEY" INNER JOIN
"GOSALESDW"."SLS_PRODUCT_DIM" "SLS_PRODUCT_DIM" ON
"SLS_PRODUCT_DIM"."PRODUCT_KEY" = "SLS_SALES_FACT"."PRODUCT_KEY" WHERE
"EMP_EMPLOYEE_DIM1"."EMPLOYEE_KEY" BETWEEN 4001 AND 4972 GROUP BY
"SLS_PRODUCT_DIM"."PRODUCT_BRAND_KEY", "GO_TIME_DIM2"."CURRENT_YEAR";

SELECT ...
```

## 5.2.1 Before BLU-conversion results

In Example 5-1 on page 172, we ran a **db2batch** command to test the captured workload against the row-organized Cognos sample `GS_DB` database before our BLU Acceleration conversion. Example 5-3 shows a sample **db2batch** output from the query run against the row-organized database. From the **db2batch** output summary, we find the following approximations of time spent:

- Approximately 1398 seconds (approximately 23 minutes) were spent for each query.
- A total of approximately 8389 seconds (approximately 2.3 hours) was spent to complete all six queries in the captured workload.

*Example 5-3 db2batch results against row-organized tables in our example*

```
* Summary Table:

Type      Number      Repetitions Total Time (s) Min Time (s)   Max Time (s)
--------- ----------- ----------- -------------- -------------- --------------
Statement           1           1    1355.823464    1355.823464    1355.823464
Statement           2           1    1612.169303    1612.169303    1612.169303
Statement           3           1    1352.828554    1352.828554    1352.828554
Statement           4           1    1340.276011    1340.276011    1340.276011
Statement           5           1    1339.085966    1339.085966    1339.085966
Statement           6           1    1388.847591    1388.847591    1388.847591

Arithmetic Mean Geometric Mean Row(s) Fetched Row(s) Output
--------------- -------------- -------------- -------------
1355.823464    1355.823464           100           10
1612.169303    1612.169303            11           10
1352.828554    1352.828554            91           10
1340.276011    1340.276011            35           10
1339.085966    1339.085966           575           10
1388.847591    1388.847591            20           10

* Total Entries:              6
* Total Time:              8389.030889 seconds
* Minimum Time:            1339.085966 seconds
* Maximum Time:            1612.169303 seconds
* Arithmetic Mean Time:    1398.171815 seconds
* Geometric Mean Time:     1395.036726 seconds
```

## 5.2.2 After BLU-conversion results

Example 5-4 on page 175 shows the sample output captured the performance information after the sample database GS_DB is converted to BLU column-organized tables. In our results, we find the following information:

- Each query completed in less than 300 seconds (less than 5 minutes) that would otherwise complete in 23 minutes.
- The fastest-performed query completed in 80 seconds against converted column-organized tables, as opposed to the same query taking 1612 seconds, being the longest-performed query before the conversion. This is approximately 20 times faster than querying from data stored in a traditionally OLTP-optimized row-organized platform.
- The entire workload of six queries took approximately 1337 seconds (approximately 22 minutes) that would otherwise take 8389 seconds (approximately 2.3 hours) to complete.

*Example 5-4 db2batch results against column-organized tables in our example*

```
* Summary Table:

Type      Number      Repetitions Total Time (s) Min Time (s)   Max Time (s)
--------- ----------- ----------- -------------- -------------- --------------
Statement          1           1     298.558335     298.558335     298.558335
Statement          2           1      80.128576      80.128576      80.128576
Statement          3           1     234.080254     234.080254     234.080254
Statement          4           1     197.261276     197.261276     197.261276
Statement          5           1     266.079743     266.079743     266.079743
Statement          6           1     260.751941     260.751941     260.751941

Arithmetic Mean Geometric Mean Row(s) Fetched Row(s) Output
--------------- -------------- -------------- -------------
1355.823464    1355.823464            100            10
1612.169303    1612.169303             11            10
1352.828554    1352.828554             91            10
1340.276011    1340.276011             35            10
1339.085966    1339.085966            575            10
1388.847591    1388.847591             20            10

* Total Entries:              6
* Total Time:              1336.860125 seconds
* Minimum Time:              80.128576 seconds
* Maximum Time:             298.558335 seconds
* Arithmetic Mean Time:     222.810021 seconds
* Geometric Mean Time:      206.099577 seconds
```

With a simple **db2batch** command, we can obtain an overview of the run times for a given query workload.

## 5.3 Cognos Dynamic Query Analyzer

For Cognos BI developers, Cognos Dynamic Query Analyzer (DQA) can be a useful tool to measure and analyze query workload performance before and after a BLU Acceleration deployment. Cognos Dynamic Query Analyzer provides a graphical interface for reviewing Cognos query logs. Logs are visualized in a graphical representation format, helping you understand the costs and how the queries are run from the view of a Cognos query engine. Dynamic Query Analyzer is the main tool to visualize queries in Dynamic Query Mode, but it can also be used for reports that are not based on dynamic cubes.

In this section, we continue the performance review before and after our BLU Acceleration deployment using Cognos Dynamic Query Analyzer. For more details about the sample data set we used in our scenario, see “Sample workload used in our scenario” on page 171.

## 5.3.1 Quick configuration of Cognos Dynamic Query Analyzer

Before we demonstrate our query results in Dynamic Query Analyzer, several items must be set up:

- Enable log tracing on dynamic queries, so that when any user generates a report, a query log trace is also generated for later review.
- Specify several settings in Dynamic Query Analyzer to connect to the Cognos instance and pull the query logs from the Cognos server.

This is a simplified demonstration of the preconfiguration to use Dynamic Query Analyzer. For more details about configuring Dynamic Query Analyzer, see the following web page:

http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp?topic=%2Fcom.ibm.swg.ba.cognos.ig_dqa.10.2.0.doc%2Ft_instdqa.html

You can download Cognos Dynamic Query Analyzer from IBM Passport Advantage® or IBM PartnerWorld® Software Access Catalog.

### Enable query execution trace in Cognos Administration

Use the following steps to gather log tracing for query requests run under a QueryService:

1. Log in as a Cognos administrator. In the Cognos Connection web interface (Figure 5-2). Click **IBM Cognos Administration** to launch the administration view.

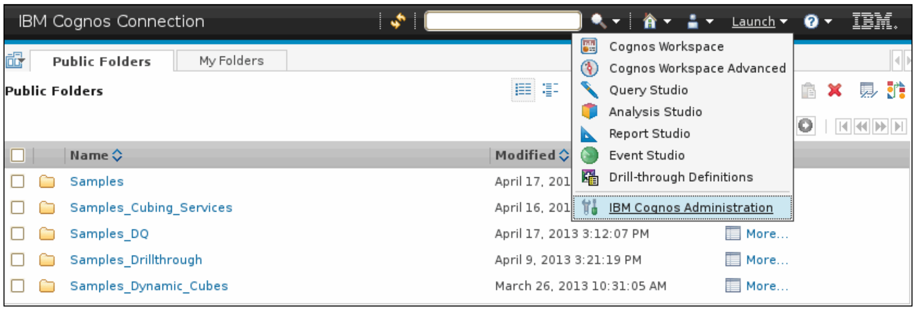



*Figure 5-2 Launching IBM Cognos Administration from Cognos Connection*

2. In IBM Cognos Administration, select **Status** → **System** → **<name of target Cognos Server>** → **<name of target Cognos dispatcher>** → **Query Service** → **Set properties**, as shown in Figure 5-3.

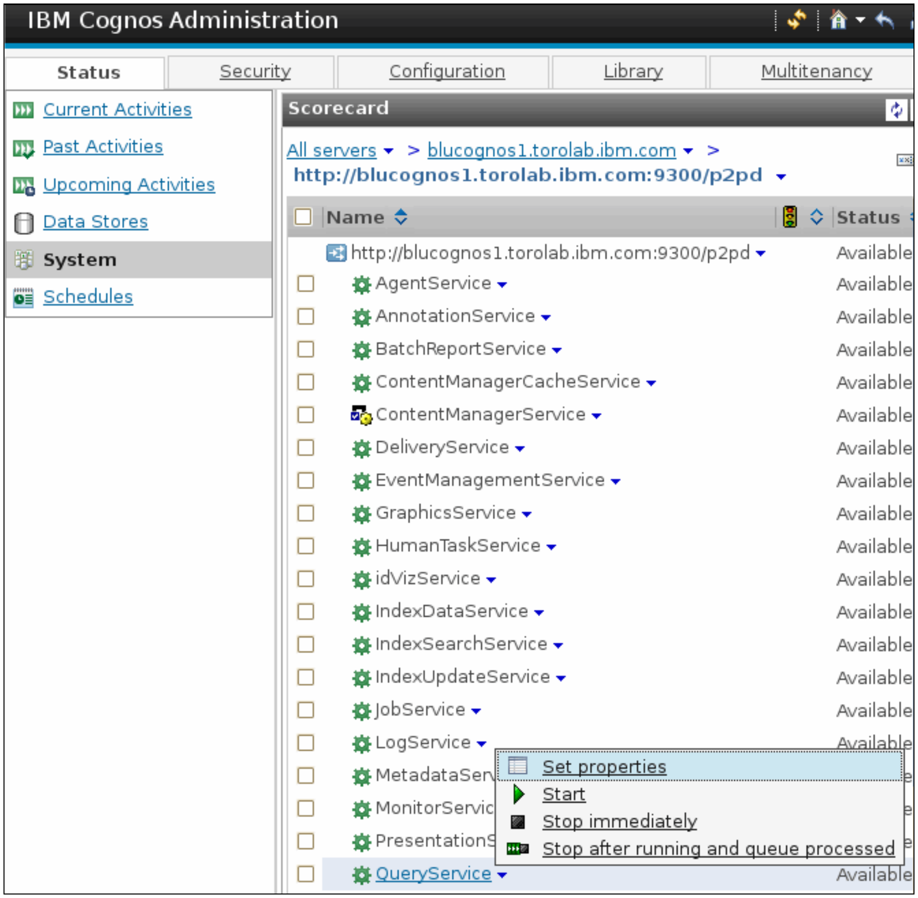



*Figure 5-3 Set properties for QueryService of the target Cognos server or dispatcher*

3. In the Set properties panel, select **Settings**. Then, locate the **Enable query execution trace** property, and select the check box under the Value column, as shown in Figure 5-4 on page 178. This enables query execution tracing and gathers the log and trace information for all requests to this QueryService. When query execution tracing is complete, you can return to this panel and disable the same property.

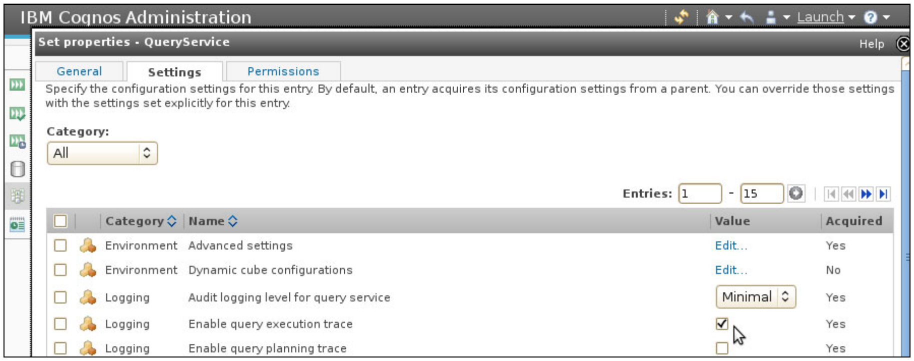



*Figure 5-4 Enable Query execution trace for designated QueryService*

## Cognos Dynamic Query Analyzer configurations

Follow these steps to set preferences in Cognos Dynamic Query Analyzer to connect to your current Cognos instance. If required, you can optionally set the default log directory URI to pull your query logs from a web server instead of a local file systems.

1. In Cognos Dynamic Query Analyzer, select **Window** → **Preferences**. In the left navigation pane (Figure 5-5), select **Cognos Server**. Enter your Cognos Dispatcher and Gateway URIs, then click **Apply**. This makes sure that Cognos DQA is connected to your current Cognos instance.

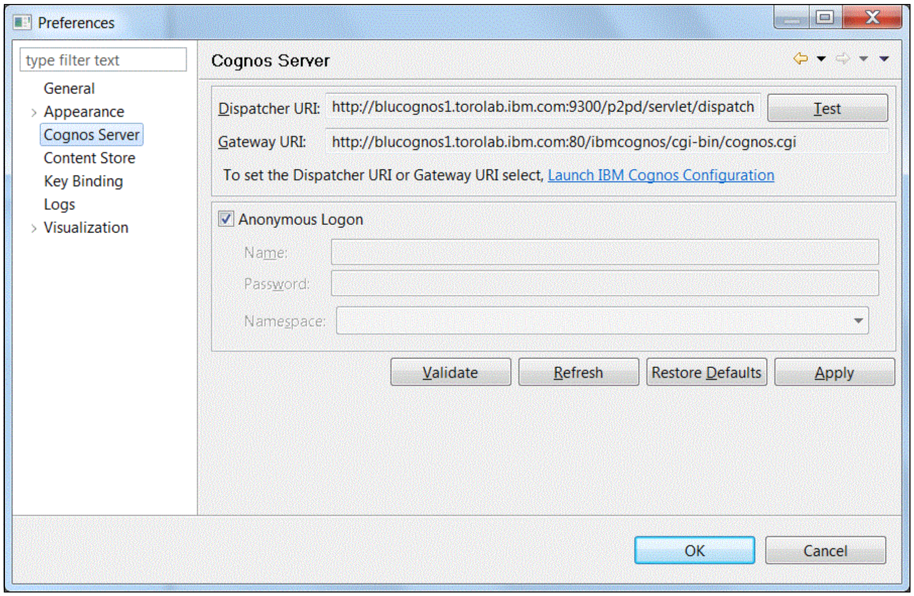



*Figure 5-5 Dynamic Query Analyzer: set Cognos Server preferences*

2. In our scenario, we access our Cognos query execution logs from the Cognos web server. Therefore, we can optionally set the logging folder URI in the DQA Preferences settings. Select **Logs** in the left navigation pane (Figure 5-6), then enter the web server path that contains your Cognos query logs in the **Log directory URI** field. Click **Apply** and **OK** when finished.

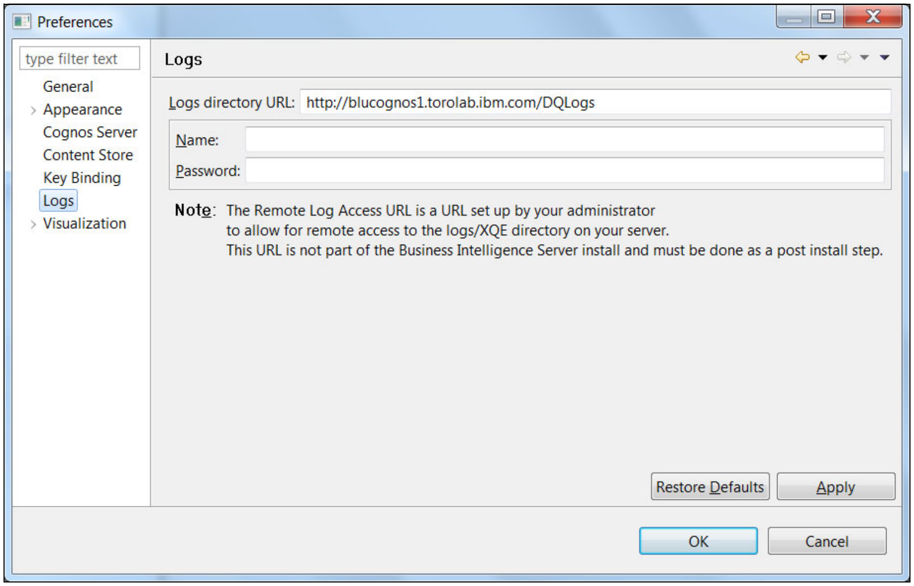



*Figure 5-6 Dynamic Query Analyzer: set Logs directory URL preferences*

## 5.3.2 Opening query execution trace logs in DQA

Before this step, our users ran a few Cognos reports from Cognos connection against our three billion rows expanded `GS_DB` database for testing purposes. Reports were run before and after the database was converted to use BLU Acceleration as a comparison.

After enabling query execution trace and setting up Dynamic Query Analyzer, we are ready to review the query trace logs generated by the reports that ran previously. Query trace logs are generated in the `XQE` directory in the Cognos log directory as the reports are run.

Use the following steps to review the query trace logs:

1. Launch **Cognos Dynamic Query Analyzer**. Click **Open Log**.
2. In the Open Log dialog (Figure 5-7), select the query log to open. In our scenario, we open our query logs from the Cognos web server that we setup earlier. Other open log source options include previously opened files, or local file directory.

   In Figure 5-7, we open two sets of logs from the same query that we executed before and after the BLU deployment for a comparison review.

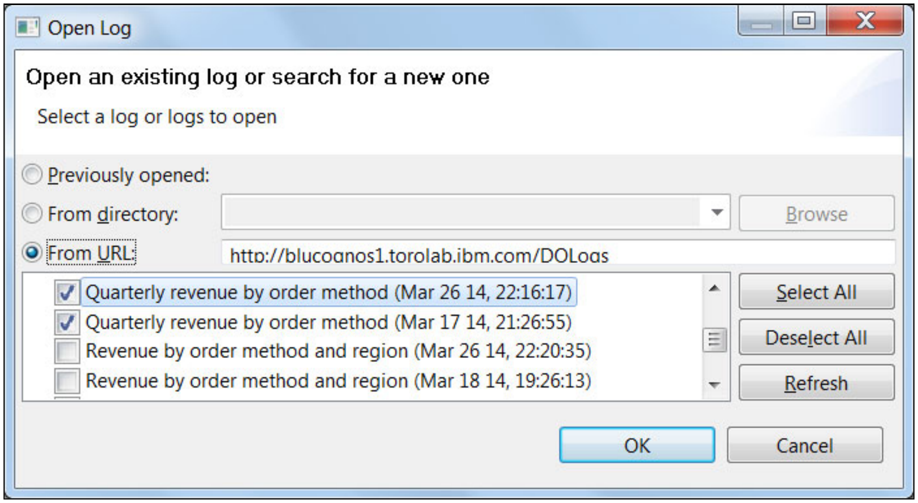



*Figure 5-7 Open Log from web server URL*

3. Dynamic Query Analyzer opens the selected query logs in several views similar to Figure 5-8. By default, the following views are shown:
   - Summary: Displays the overview information including the data source type, and total time that is needed to execute the query.
   - Query: Displays the full query statement being run.
   - Navigation: Shows the execution tree of the query in graphical format.
   - Properties: Displays the detailed properties of a selected node.

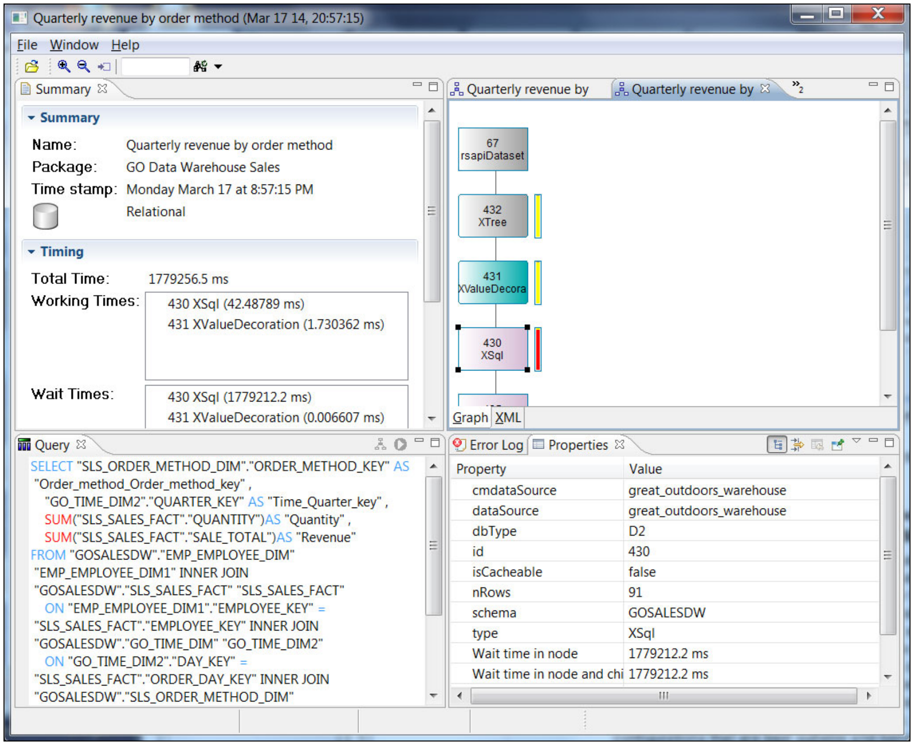



*Figure 5-8 Dynamic Query Analyzer window-based views*

## 5.3.3 Log summary before BLU Acceleration deployment

In our example, we focus on the timing Summary view of our query logs, specifically the performance numbers against the underlying database.

- Before converting the Cognos sample database to column-organized tables, the *Quarterly revenue by order method* query was run and query execution logs were captured. From the captured logs, Figure 5-9 shows the Summary trace of the relational data source component involved in the dynamic cube query. We observe that the total time spent in QueryService to fetch data from the source relational database is approximately 1,779,256 ms, which is approximately 30 minutes.

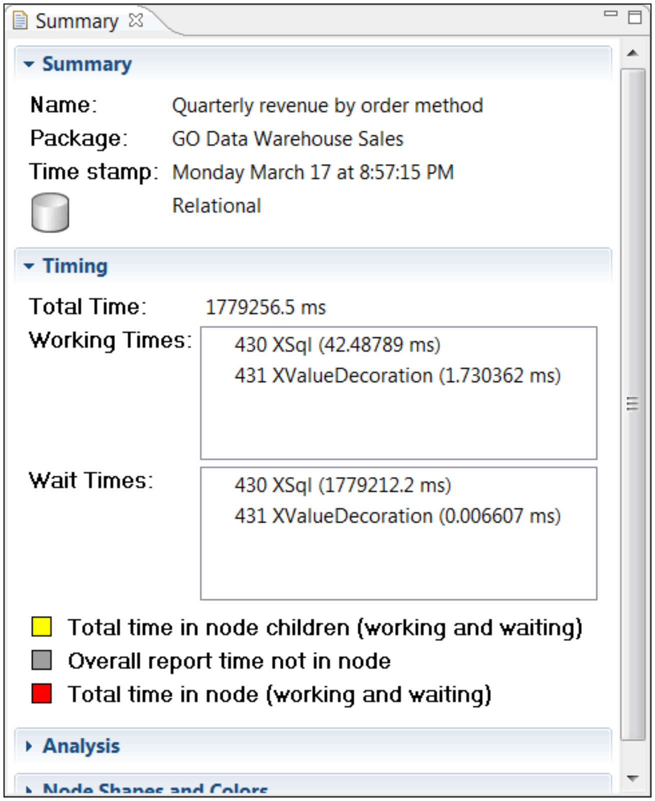



*Figure 5-9 Timing summary against row-organized database*

- You can review the logs in more depth and determine the actual time spent on querying the underlying database for a specific SQL component in the query. In the navigation view, click the **Xsql** node; the node details are displayed in the Properties view. The execution trace (Figure 5-10) shows that 91 rows were returned (`nRows`) from the SQL query. Overall, approximately 1779 seconds (approximately 30 minutes) were spent to complete the query against the source (row-organized) database.

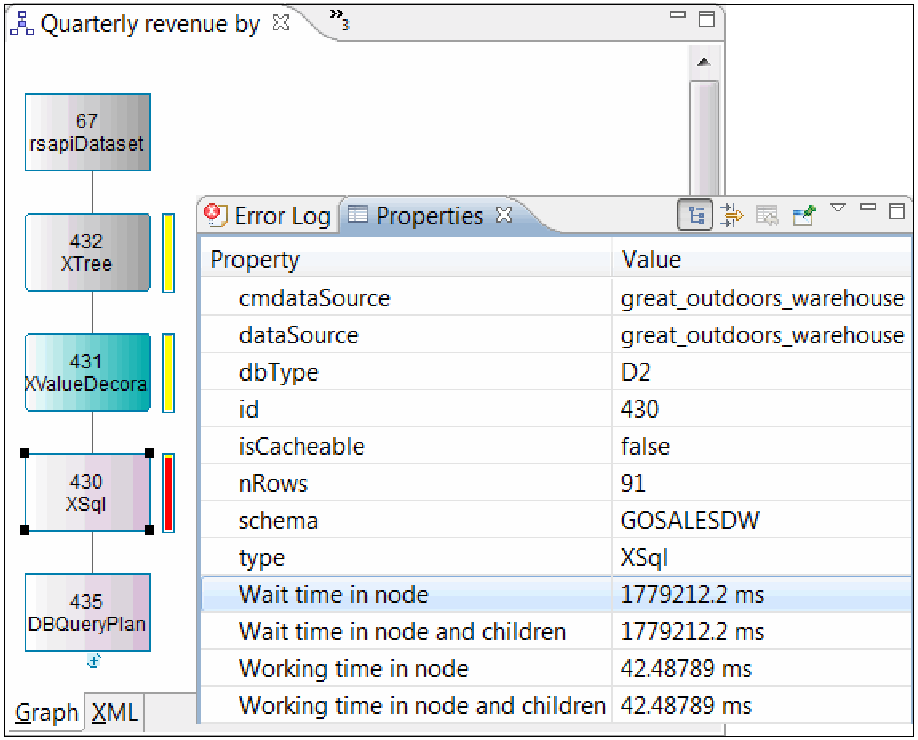



*Figure 5-10 Xsql node and its properties against row-organized database*

## 5.3.4 Log summary after BLU Acceleration deployment

After the Cognos Go Sales Warehouse database is converted into column-organized tables, we run the same sample dynamic cube *Quarterly revenue by order method* query. With Dynamic Query Analyzer, we review the log details. The sample outputs in this section are measured before any data is loaded in memory. Typically, you run the workload twice to experience the query response time when data is loaded in memory. In our scenario, reports returned instantly in the second run.

Figure 5-11 shows the timing summary for the relational data source component of the log, from the same dynamic cube query running against the converted column-organized tables. The log captured the total time spent querying from relational database at 277 717 ms, which is approximately 4.6 minutes. This is about 6.4 times faster than the same dynamic cube query against row-organized tables.

**Note:** This is a functional demonstration only, with workloads tested on hardware with less than minimum recommendations.

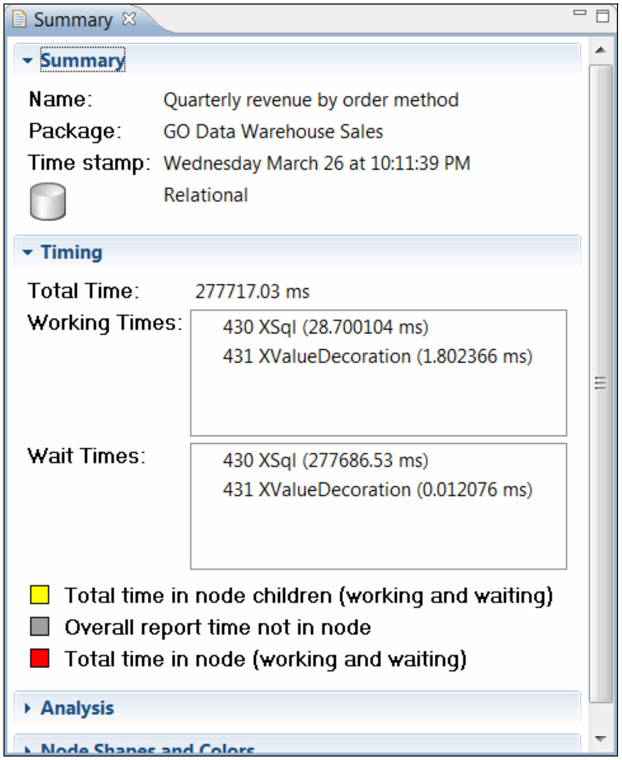



*Figure 5-11 Timing summary against column-organized database*

Again, we drill-down further to the same Xsql node properties that we reviewed previously for a direct comparison of SQL execution time from the database before and after BLU conversion. Figure 5-12 illustrates the Xsql node and its properties from a query that is executed after a BLU conversion. From the execution trace, we observe that the same SQL query completed in approximately 278 seconds on the database, returning 91 rows (`nRows`). This is 6.4 times faster than the same dynamic cube query against a row-organized database.

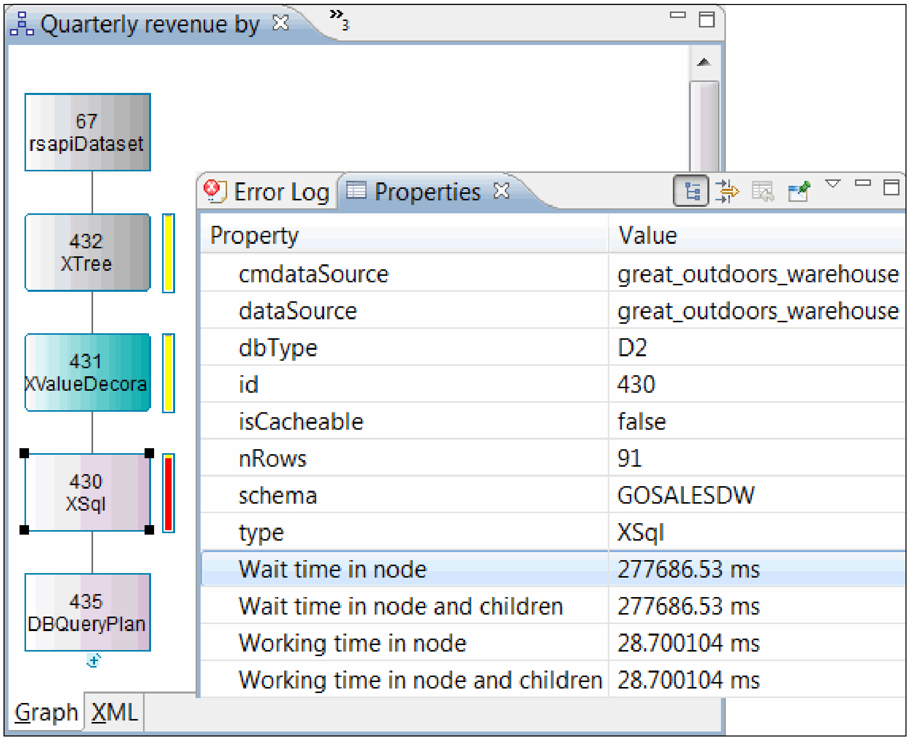



*Figure 5-12 Xsql node and its properties against column-organized database*

## 5.4 Conclusion

In this chapter, we demonstrated a DB2 benchmark utility and a Cognos tool that can be useful to measure workload performance before and after a BLU deployment for testing. Various other DB2 monitoring elements are available to perform a more thorough validation of a BLU Acceleration deployment. See Chapter 6, “Post-deployment of DB2 with BLU Acceleration” on page 187.

6

# Post-deployment of DB2 with BLU Acceleration

In this chapter, we provide information about monitoring elements that you can observe from a DB2 with BLU Acceleration environment. We discuss how you can analyze and ensure that the environment is set up correctly for maximum performance. To demonstrate these concepts, we use the Cognos Sample Outdoors Warehouse example.

The following topics are covered:

- Post-deployment of BLU Acceleration
- Table organization catalog information
- BLU Acceleration metadata objects
- Storage savings
- Memory utilization for column data processing
- Workload management
- Query optimization

# 6.1 Post-deployment of BLU Acceleration

With simplicity and ease of use being the main key objectives of the BLU Acceleration design, typical relational analytics database deployment tasks are greatly simplified. Database administrators no longer need to explicitly create user-defined indexes, or significantly tune databases to reach a state that is ready for production with excellent performance results.

With DB2 BLU Acceleration, analytic environments are automatically optimized with available hardware and are combined into a single registry variable, `DB2_WORKLOAD=ANALYTICS`. Basically create your column-organized tables, and then *load and go*. Chapter 5, "Performance test with a Cognos BI example" on page 169 showed decent performance results after data was loaded into column-organized tables, without any manual tuning effort.

The deployment of DB2 with BLU Acceleration is made easy through autonomics; the maintenance of BLU Acceleration is also straightforward. For DBAs, the DB2 process model is the same as you are used to. The same DB2 administration skills, APIs, backup and recovery protocols, and so on can all be kept as before. Also, you no longer must do many of the maintenance tasks because DB2 automatically does these for you.

Although DB2 with BLU Acceleration is easy to use with significantly less manual administration items, understanding how the database is configured and performing is useful. After a deployment, a DBA naturally wants to obtain the performance gain in the current workload with BLU Acceleration tables. DBAs can observe new monitoring elements to ensure that a deployed BLU Acceleration environment is working the way it should. This chapter focuses on these monitoring elements and some interesting BLU Acceleration administration tasks useful for DBAs.

# 6.2 Table organization catalog information

After deployment, the first item that you can check is the table organization of the converted tables listed in the catalog. DB2 10.5 adds a new column, `TABLEORG`, to the `SYSCAT.TABLES` catalog view. This single character value, displays the organization of each table: the letter "R" for row-organization or the letter "C" for column-organization.

Use the query in Example 6-1 on page 189 to see if your tables were created in BLU Accelerated columnar format.

*Example 6-1 checking if table is in BLU Accelerated columnar format*

```
SELECT
   substr(tabschema, 1, 30) AS TABSCHEMA,
   substr(tabname, 1, 30) AS TABNAME,
   tableorg
FROM syscat.tables
   WHERE tabschema ='<tabschema_name>';
```

Example 6-2 shows an excerpt of the output from the Cognos sample database after a BLU Acceleration deployment. The `TABLEORG` column shows "C" for all the column-organized tables in the `GOSALESDW` schema. The `TABLEORG` for those tables in the `GOSALES` schema are denoted as "R" to indicate traditional row-organized tables.

*Example 6-2 Sample output of TABLEORG*

```
TABSCHEMA            TABNAME                        TABLEORG
-------------------- ------------------------------ --------
GOSALESDW            SLS_RTL_DIM                    C
GOSALESDW            SLS_SALES_FACT                 C
GOSALESDW            SLS_SALES_ORDER_DIM            C
GOSALESDW            SLS_SALES_TARG_FACT            C
GOSALES              BRANCH                         R
GOSALES              CONVERSION_RATE                R
GOSALES              COUNTRY                        R
GOSALES              CURRENCY_LOOKUP                R
```

# 6.3 BLU Acceleration metadata objects

To ensure optimal performance with BLU Acceleration, the DB2 engine creates and maintains metadata objects in the background. Although no DBA intervention is required for these metadata objects, it is beneficial to know about them.

## 6.3.1 Synopsis tables

One of the metadata object that gets created for BLU Acceleration is a data object called a *synopsis table*. It is created and populated automatically upon creation of a BLU Acceleration columnar table. This internal metadata table is maintained automatically during subsequent loads, imports, inserts, updates, and deletes of the column organized table for data-skipping purposes.

Before analyzing the structure of a synopsis table, first understand the synopsis table and how it is built. In DB2 10.5, when column-organized tables are created, a companion table (synopsis table) is also created by DB2, automatically. The synopsis table tracks the minimum and maximum column data values for each range of 1024 rows. The range of values that we store are specifically, from non-character columns and those character columns that are part of its primary or foreign key definitions. The size of the synopsis table is approximately 0.1% the size of the base table.

> **Note:** Although not a requirement, loading pre-sorted data helps to cluster data and increase efficiency of synopsis tables.

Example 6-3 shows a synopsis table from a test environment. Note the structure of the table. Minimum and maximum values of non-character or key columns are stored, with the `TSNMIN` and `TSNMAX` values that indicate the row ranges (that is, tuple sequence numbers). In this example, when BLU Acceleration runs a query that looks for orders made in February 2014, it goes straight to the second two ranges of tuple sequence numbers (1024 - 3071) without having to scan through all the data.

*Example 6-3 Sample excerpt of the output from a synopsis table*

```
SHIP_DAY_KEYMIN SHIP_DAY_KEYMAX TSNMIN               TSNMAX
--------------- --------------- -------------------- --------------------
       20040113        20040115                    0                 1023
       20040120        20040223                 1024                 2047
       20040224        20040225                 2048                 3071
       20040416        20040426                 3072                 4095
       20040513        20061023                 4096                 5119
       20061025        20061026                 5120                 6143
       20061027        20061030                 6144                 7167
       20061204        20061205                 7168                 8191
       20061211        20061211                 8192                 9215
```

Synopsis tables are always created in the SYSIBM schema, with a naming convention like `SYN%_<tablename>`. To obtain a list of all synopsis tables that are created in the database, simply query a list of tables in the SYSIBM schema with its naming pattern, as shown in Example 6-4.

*Example 6-4 Sample query to return a list of synopsis tables in database*

```
select TABNAME, COLCOUNT
from SYSCAT.TABLES
where TABSCHEMA='SYSIBM' and TABNAME like 'SYN%';
```

Alternatively, to determine the name of the synopsis table that is created for a specific column-organized table, you can query the SYSCAT.TABLES catalog view by using the command shown in Example 6-5.

*Example 6-5 Sample command and output to determine name of a synopsis table*

```
select substr(TABSCHEMA,1,30) AS TABSCHEMA, substr(TABNAME,1,40) AS
TABNAME
from SYSCAT.TABLES
where TABNAME like '%SLS_SALES_FACT';

TABSCHEMA                      TABNAME
------------------------------ ----------------------------------------
GOSALESDW                      SLS_SALES_FACT
SYSIBM                         SYN140325171836089285_SLS_SALES_FACT

  2 record(s) selected.
```

Alternatively and to achieve similar results, you can query the SYSCAT.TABDEP catalog view as shown in Example 6-6. A new value to the dependent object metric is added to signify synopsis table objects (dtype='7').

*Example 6-6 Synopsis Table Name Query*

```
SELECT
   bschema AS BLU_SCHEMA,
   bname AS BLU_TABLENAME,
   tabschema AS SYN_SCHEMA,
   tabname AS SYN_TABLENAME
FROM syscat.tabdep
WHERE dtype = '7';
```

Example 6-7 shows an excerpt of the output of this query.

*Example 6-7 Sample output of Synopsis tables from SYSCAT.TABDEP catalog view*

```
BLU_SCHEMA BLU_TABLENAME        SYN_SCHEMA SYN_TABLENAME
---------- -------------------- ---------- ------------------------------------
GOSALESDW  SLS_RTL_DIM          SYSIBM     SYN140325171835493928_SLS_RTL_DIM
GOSALESDW  SLS_SALES_FACT       SYSIBM     SYN140325171836089285_SLS_SALES_FACT
```

**Tip:** Synopsis tables only stores data value ranges for non-character columns or key columns. If a column with date values is a frequently queried column in a BLU Accelerated database, and it is currently defined as a character column, consider defining the column in the DATE data type instead. This ensures that the date column is included in the synopsis table and data-skipping is considered during a query process.

### 6.3.2 Pagemap indexes

DB2 with BLU Acceleration also automatically creates a data object called a *pagemap index*. It is also automatically maintained by the DB2 engine and is used by the engine to get the physical location of pages.

Example 6-8 shows a query that returns the associated pagemap index that is created for a particular BLU Acceleration table.

*Example 6-8 Finding pagemap index query*

```
SELECT
   tabname,
   indname,
   indextype
FROM syscat.indexes
WHERE tabschema ='GOSALESDW'
ORDER BY tabname,indname;
```

Example 6-9 shows output. The `CPMA` index type represents column-organized page map (CPMA) index.

*Example 6-9 Output of CPMA index types*

```
TABNAME                        INDNAME              INDEXTYPE
------------------------------ -------------------- ---------
SLS_RTL_DIM                    SQL140325171835667 38 CPMA
SLS_SALES_FACT                 SQL14032517183626671 CPMA
SLS_SALES_ORDER_DIM            SQL14032517183685697 CPMA
SLS_SALES_TARG_FACT            SQL14032517183743802 CPMA
```

## 6.4 Storage savings

BLU Acceleration uses several compression algorithms to compress data, thus delivering significant storage savings. Besides saving storage, BLU Acceleration adaptive compression also helps improve query performance. The better your column-organized tables are compressed, the more data you can store in memory, thus reducing disk I/0 and improving query response time.

BLU Acceleration also has a unique vector-processing engine, which operates on compressed values rather than uncompressed values. This unique feature delivers much quicker query response times.

### 6.4.1 Table-level compression rates

Starting at a table level first, we examine the size and compression rates of BLU Acceleration tables. In particular, seeing the number of data pages that are saved through columnar compression is useful. DB2 10.5 creates the following monitor elements for columnar based data:

- `COL_OBJECT_L_SIZE`: Amount of disk space logically allocated for the column-organized data in the table, reported in kilobytes.
- `COL_OBJECT_P_SIZE`: Amount of disk space physically allocated for the column-organized data in the table, reported in kilobytes.
- `COL_OBJECT_L_PAGES`: Number of logical pages used on disk by column-organized data contained in this table.

These elements are reported by the `ADMIN_GET_TAB_INFO` table function, a standard administrative routine in DB2 10.5.

Also, an existing compression monitor element `PCTPAGESSAVED` is adapted for columnar tables. You can find the value of `PCTPAGESSAVED` for each BLU Acceleration table in the `SYSCAT.TABLES` catalog view. It provides an approximate percentage of pages saved in a table as a result of compression.

For row-based tables, `NPAGES` is normally used to determine the table size. However, for column-based BLU Acceleration tables, because `NPAGES` does not account for metadata or empty pages, it will underestimate the actual space usage, especially for small tables. You can obtain a more accurate value by adding the following key metric values from `ADMIN_GET_TAB_INFO`:

```
TOT_COL_STOR_SIZE = COL_OBJECT_P_SIZE + DATA_OBJECT_P_SIZE +
INDEX_OBJECT_P_SIZE
```

This equation sums physical size of associated objects for a column-organized table:

- `COL_OBJECT_P_SIZE` is the physical size for column-organized data objects, including column-organized user data and empty pages.

- DATA_OBJECT_P_SIZE is the physical size for metadata objects including compression dictionaries created by DB2.
- INDEX_OBJECT_P_SIZE is the physical size for all index objects, including non-enforced keys and unique constraints if there is any, and pagemap indexes which is a metadata object that BLU uses to locate physical location of pages.

Example 6-10 is a query to retrieve the computed size (AS COLSIZE) of the column-organized tables along with the computed compression ratio (AS COMPRATIO).

*Example 6-10 Table level compression analysis*

```
SELECT
   substr(a.tabschema,1,20) AS TABSCHEMA,
   substr(a.tabname,1,20) AS TABNAME,
   a.card,
   b.data_object_p_size+b.index_object_p_size+b.col_object_p_size AS
COLSIZE,
   a.pctpagessaved,
   DEC(1.0 / (1.0 - pctpagessaved/100.0),5,2) AS COMPRATIO
FROM
   syscat.tables a,
   sysibmadm.admintabinfo b
WHERE a.tabname = b.tabname
AND a.tabschema = b.tabschema
AND a.tabschema = 'GOSALESDW';
```

Example 6-11 shows an excerpt of the output.

*Example 6-11 PCTPAGESSAVED and compression ratio sample output*

```
TABSCHEMA  TABNAME              CARD       COLSIZE   PCTPAGESSAVED COMPRATIO
---------- -------------------- ---------- --------- ------------- ---------
GOSALESDW  SLS_SALES_FACT       2999547946 158704640            83      5.88
GOSALESDW  SLS_SALES_TARG_FACT  233625     9728                 83      5.88
GOSALESDW  DIST_PRODUCT_FORECAS 129096     4736                 84      6.25
GOSALESDW  MRK_PRODUCT_SURVEY_F 165074     5888                 86      7.14
GOSALESDW  SLS_PRODUCT_DIM      3000300    107520               86      7.14
```

In this example, we obtained a compression ratio (COMPRATIO) of approximately 5 - 7x, reducing storage and at the same time fitting more data into memory cache. If you find that PCTPAGESAVED has a value of -1, statistics might be outdated.

You can update the statistics by running the following DB2 command-line processor (CLP) command on the table in question:

```
db2 "RUNSTATS ON TABLE <BLU_TABLENAME> ON ALL COLUMNS WITH DISTRIBUTION
ON ALL COLUMNS AND INDEXES ALL ALLOW WRITE ACCESS"
```

## 6.4.2 Column-level compression rates

Now that you have an idea of the column-organized tables storage footprint, and how effective the compression algorithms are, examining the tables further can help you understand how the data values are encoded at the column level. This section explores how to obtain the percentage of values that are encoded as a result of compression, for each column in our BLU Acceleration column-organized tables.

DB2 10.5 offers the `PCTENCODED` column in the `SYSCAT.COLUMNS` catalog view. It represents the percentage of values that are encoded as a result of compression for individual columns in a column-organized table. This information is collected and maintained as part of the automatic statistics gathering feature in BLU Acceleration.

By checking the value for each column in your columnar tables, you can measure and identify which columns have been optimally compressed. You can also determine if any column values were left uncompressed because of an insufficient utility heap during the data load. The SQL statement shown in Example 6-12 queries the column information and the percentage of values encoded in a Cognos sample `GOSALESDW.SLS_SALES_FACT` table.

*Example 6-12 Column level compression analysis*

```
SELECT
   substr(tabname,1,20) AS TABNAME,
   substr(colname,1,20) AS COLNAME,
   substr(typename,1,10) AS TYPENAME,
   length,
   pctencoded
FROM
   syscat.columns
WHERE tabschema ='GOSALESDW' and tabname='SLS_SALES_FACT'
ORDER BY 1,5 DESC;
```

Example 6-13 on page 196 shows output of the query. In this example, all columns in the `GOSALESDW.SLS_SALES_FACT` table are 100% encoded. This means, all column values are encoded by the compression dictionaries. This is the most ideal compression to achieve for best performance.

*Example 6-13 Sample output of PCTENCODED*

```
TABNAME              COLNAME              TYPENAME   LENGTH      PCTENCODED
-------------------- -------------------- ---------- ----------- ----------
SLS_SALES_FACT       ORGANIZATION_KEY     INTEGER              4        100
SLS_SALES_FACT       CLOSE_DAY_KEY        INTEGER              4        100
SLS_SALES_FACT       ORDER_DAY_KEY        INTEGER              4        100
SLS_SALES_FACT       QUANTITY             BIGINT               8        100
SLS_SALES_FACT       PRODUCT_KEY          INTEGER              4        100
SLS_SALES_FACT       EMPLOYEE_KEY         INTEGER              4        100
SLS_SALES_FACT       GROSS_MARGIN         DOUBLE               8        100
SLS_SALES_FACT       RETAILER_KEY         INTEGER              4        100
SLS_SALES_FACT       PROMOTION_KEY        INTEGER              4        100
SLS_SALES_FACT       RETAILER_SITE_KEY    INTEGER              4        100
SLS_SALES_FACT       SALE_TOTAL           DECIMAL             19        100
SLS_SALES_FACT       SALES_ORDER_KEY      INTEGER              4        100
SLS_SALES_FACT       GROSS_PROFIT         DECIMAL             19        100
SLS_SALES_FACT       UNIT_COST            DECIMAL             19        100
SLS_SALES_FACT       ORDER_METHOD_KEY     INTEGER              4        100
SLS_SALES_FACT       UNIT_PRICE           DECIMAL             19        100
SLS_SALES_FACT       UNIT_SALE_PRICE      DECIMAL             19        100
SLS_SALES_FACT       SHIP_DAY_KEY         INTEGER              4        100

  18 record(s) selected.
```

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

If you see any columns or tables, where PCTENCODED has a value of -1, verify the following information:

- The table is organized by column.
- The statistics were collected and are up to date.
- The data was loaded into the table.

## 6.4.3 Automatic space reclamation

When data is deleted from a column-organized table, ideally, we want to return the pages, on which the deleted data resided, to table space storage, where they can later be reused by any table in the table space.

By setting the DB2_WORKLOAD registry variable to ANALYTICS, a default policy is applied, and the AUTO_REORG database configuration parameter is set so that automatic reclamation is active for all column-organized tables.

For environments where DB2_WORKLOAD registry variable is not set, you can set AUTO_REORG database configuration parameter to ON to enable automatic column-organized tables space reclamation. If for any reason you prefer to keep the AUTO_REORG configuration parameter disabled, you can manually reclaim space by using the **REORG TABLE** command and specifying the RECLAIM EXTENTS option for that command. As with most background processes, BLU Acceleration uses an automated approach.

If required, you can monitor the progress of a table reorganization operation with the reorganization monitoring infrastructure. The ADMIN_GET_TAB_INFO table function returns an estimate of the amount of reclaimable space on the table, which you can use to determine when a table reorganization operation is necessary.

To determine whether any space can be reclaimed, you might use the query shown in Example 6-14, which returns the amount of reclaimable space for a BLU Acceleration column-organized table.

*Example 6-14 Estimate reclaimable space*

```
SELECT
   tabname,
   reclaimable_space
FROM table(sysproc.admin_get_tab_info('GOSALESDW','SLS_SALES_FACT'));
```

Example 6-15 shows output where no space needs to be reclaimed. This is as expected in an environment where AUTO_REORG is enabled. A DB2 daemon runs in the background and frequently checks and reclaims any empty extents that can be reclaimed automatically.

*Example 6-15 Reclaimable space for a table*

```
TABNAME                         RECLAIMABLE_SPACE
------------------------------ --------------------
SLS_SALES_FACT                                     0

  1 record(s) selected.
```

## 6.5 Memory utilization for column data processing

Another key feature of DB2 with BLU Acceleration is scan-friendly memory caching. Most database systems have memory caching, but they tend to be configured for transaction processing rather than analytical. Transaction processing systems tend to use paging algorithms, such as least recently used

(LRU) and most recently used (MRU). This usually results in keeping the data most recently referenced, in memory and getting rid of older, less frequently accessed data from memory to disk.

DB2 with BLU Acceleration introduces a new scan-friendly, analytics-optimized, page-replacement algorithm. It detects data access patterns, keeps hot data in buffer pools as long as possible, and minimizes I/O. It also works well with traditional row-based algorithms in mixed workload environments. As with most admin tasks with BLU Acceleration, the DBA does not have to do any tasks here, BLU Acceleration automatically adapts the way it caches data based on the organization of the table being accessed. No optimization hints, no configuration parameters to set, it all happens automatically.

### 6.5.1 Column-organized hit ratio in buffer pools

Although BLU Acceleration delivers accelerated performance using unique analytics-optimized algorithms and buffer pool caching, a prudent task is to always monitor the memory aspects of your database. Understand how well your workloads use buffer pool reads versus physical reads.

Table A-1 on page 409 indicates new monitor elements that are added to DB2 10.5 and that you can use to monitor buffer pool and prefetch usage by column-organized tables. Rather than individually querying each of these metric values, DB2 10.5 automatically creates administrative views that provide a condensed representation of the monitor elements, displaying the highlights, and providing several standard calculations that are of most interest, such as buffer pool hit ratio. These views are similar to table functions because they return data in table format. However, unlike table functions, they do not require any input parameters.

The `MON_BP_UTILIZATION` administrative view returns the following columnar buffer pool information:

- `COL_PHYSICAL_READS`

  This is the number of column-organized table data pages read from the physical table space containers.

- `COL_HIT_RATIO_PERCENT`

  This is the column-organized table data hit ratio (percentage of time that the database manager does not have to load a page from disk for the request).

You can use the SQL query in Example 6-16 on page 199 to observe how the buffer pool is performing.

*Example 6-16 SQL query for observing the buffer pool performance*

```
SELECT
   varchar(bp_name,20) AS BUFFER_POOL,
   col_physical_reads,
   col_hit_ratio_percent
FROM
   sysibmadm.mon_bp_utilization;
```

Example 6-17 shows output from a Cognos Sample Outdoors sample database. In this example, the `GOSALES_BP` where the column-organized GOSALESDW schema resides has 96.9% `COL_HIT_RATIO_PERCENT`. This means for approximately 96.9% of the time, the workload was able to process the column-organized data from the buffer pools without reading the physical disk. Ideally, we want the `COL_PHYISICAL_READS` to be as low as possible, and the `COL_HIT_RATIO_PERCENT` to be as close to 100% as possible.

*Example 6-17 Bufferpool columnar performance metrics*

```
BUFFER_POOL          COL_PHYSICAL_READS   COL_HIT_RATIO_PERCENT
-------------------- -------------------- ---------------------
IBMDEFAULTBP                            0                     -
GOSALES_BP                        9413531                 96.90
IBMSYSTEMBP4K                           0                     -
IBMSYSTEMBP8K                           0                     -
IBMSYSTEMBP16K                          0                     -
IBMSYSTEMBP32K                          0                     -

  6 record(s) selected.
```

## 6.5.2 Prefetcher performance

The prefetch logic for queries that access column-organized tables is used to asynchronously fetch only those pages that each thread reads for each column that is accessed during query execution. If the pages for a particular column are consistently available in the buffer pool, prefetching for that column is disabled until the pages are being read synchronously, at which time prefetching for that column is enabled again.

Prefetch monitor elements (listed in Table A-1 on page 409) can help you track the volume of requests for data in column-organized tables that are submitted to prefetchers, and the number of pages that prefetchers skipped reading because the pages were in memory. Efficient prefetching of data in column-organized tables is important for mitigating the I/O costs of data scans.

You can use the SQL query in Example 6-18 to observe how the prefetchers are performing. A reference to what each column represents is in Table A-1 on page 409. Typically, we are looking for small values to be returned for prefetch waits and prefetch wait times, which indicates minimal I/O as pages already in the buffer pool.

*Example 6-18 SQL query for observing prefetch performance*

```
SELECT
   bp_name,
   pool_queued_async_col_reqs AS COL_REQS,
   pool_queued_async_col_pages AS COL_PAGES,
   pool_failed_async_col_reqs AS FAILED_COL_REQS,
   skipped_prefetch_col_p_reads AS SKIPPED_P_READS,
   prefetch_wait_time,
   prefetch_waits
FROM table(sysproc.mon_get_bufferpool('',-1));
```

Example 6-19 shows output of the query. In this example, we focus on the GOSALES_BP buffer pool that handles the sample Cognos analytic workloads for the test. From this output, we observe a total of 13,594,221 (`pool_queued_async_col_reqs AS COL_REQS`) column-organized pages that are requested from the workload.

Of the total of pages requested, approximately 5,368,715 pages (`skipped_prefetch_col_p_reads AS SKIPPED_P_READS`) are skipped from physical reads because those pages are already in the buffer pool. This means that approximately 40% of data was already found in the buffer pool as the Cognos workload was run.

We examine the `pool_queued_async_col_reqs` element (`AS COL_REQS` in output). There are total of 3,146,216 column-organized requests from the workload. Of all the requests, only 280657 (`prefetch_waits`) requests need to wait for the prefetcher to load data to buffer pool. This means, approximately 9% of the total requests require a wait from the prefetcher. This can be a result of the DB2 prefetcher being able to detect and determine the correct type of prefetch to use.

If we take the `prefetch_wait_time` divided by the `prefetch_waits`, we get the average time spent for prefetcher to load data onto the buffer pool. From our example, only about 35 milliseconds are spent for each prefetch. Ideally, you want as short a prefetch wait time as possible in an optimal environment.

*Example 6-19 Prefetcher performance sample output*

```
BP_NAME        COL_REQS COL_PAGES FAILED_COL_REQS SKIPPED_P_READS PREFETCH_WAIT_TIME PREFETCH_WAITS
-------------- -------- --------- --------------- --------------- ------------------ --------------
IBMDEFAULTBP   0        0         0               0               179                25
```

```
GOSALES_BP     3146216  13594221  0               5368715         9845548            280657
IBMSYSTEMBP4K  0        0         0               0               0                  0
IBMSYSTEMBP8K  0        0         0               0               0                  0
IBMSYSTEMBP16K 0        0         0               0               0                  0
IBMSYSTEMBP32K 0        0         0               0               0                  0

  6 record(s) selected.
```

## 6.5.3 Monitoring sort memory usage

Monitoring sort memory usage is important to ensure that BLU Acceleration is performing optimally against your analytical queries. In earlier chapters, we learned that BLU Acceleration uses a set of innovative algorithms to join and group column-organized data in an exceptionally fast manner, resulting in the dramatic performance improvements in analytic workloads. These algorithms utilize hashing techniques, which in turn, require sort memory to process. Therefore, BLU Acceleration does typically require more sort memory than traditional row-organized databases. The primary sort memory consumers are `JOIN`s and `GROUP BY` operations.

In an environment with BLU Acceleration, there are a few factors that affect performance of sort memory consumers:

- `SORTHEAP` database configuration parameter

  `SORTEHAP` is generally set to a higher number with self-tuning disabled when BLU Acceleration is enabled on a database. This parameter limits the maximum number of memory pages that sort heap memory can use.

- **`SHEAPTHRES_SHR`** database configuration parameter

  `SHEAPTHRES_SHR` is also set to a higher number with self-tuning disabled when BLU Acceleration is enabled on a database. Together with the `SHEAPTHRES` database manager parameter set to 0, this parameter specifies the total amount of database shared memory that is available for sort memory consumers at any given time. It controls the shared sort memory consumption at the database level. When overall sort memory usage approaches the `SHEAPTHRES_SHR` limit, memory throttles and sort memory requests might get less memory than required. This can lead to data spilling into temporary tables and performance degrade. When overall memory usage exceeds the `SHEAPTHRES_SHR` limit, queries can fail.

- The number of sort memory consumers running concurrently.

To monitor and ensure that sort memory usage is optimal in your BLU Acceleration environment, you can monitor behaviors of various sort memory consumers and the concurrency.

## Monitoring sort usage

The following monitoring elements return the overall sort memory usage for a database:

- `SORT_SHRHEAP_ALLOCATED`

  Total amount of shared sort memory currently allocated in the database. When no active sort is running at a given time, `SORT_SHRHEAP_ALLOCATED` reports a value of 0.

- `SORT_SHRHEAP_TOP`

  The high watermark for database-wide shared sort memory.

Example 6-20 shows a SQL query that reports the currently allocated shared sort heap at a given time and the high watermark shared sort heap usage. The high watermark value gives a baseline on the maximum sort memory that the system requires for the workload.

*Example 6-20 Currently allocated and maximum shared sort heap usage*

```
SELECT SORT_SHRHEAP_ALLOCATED, SORT_SHRHEAP_TOP
FROM TABLE(MON_GET_DATABASE(-1));

SORT_SHRHEAP_ALLOCATED SORT_SHRHEAP_TOP
---------------------- --------------------
                 51690                71009

  1 record(s) selected.
```

## Monitoring active sort operations

The following monitoring elements report the currently active sort operations being executed on the database:

- `ACTIVE_SORTS`

  Number of sorts in the database that are currently running and consuming sort heap memory.

- `ACTIVE_HASH_JOINS`

  Total number of hash joins that are currently running and consuming sort heap memory.

- `ACTIVE_OLAP_FUNCS`

  Total number of OLAP functions that are currently running and consuming sort heap memory.

- `ACTIVE_HASH_GRPBYS`

  Total number of `GROUP BY` operations that use hashing as their grouping method that are currently running and consuming sort heap memory.

Example 6-21 shows a SQL query that reports the total number of active sort operations currently running and consuming sort memory:

*Example 6-21 Total number of active sort operations currently running and consuming sort memory*

```
SELECT
   (ACTIVE_SORTS + ACTIVE_HASH_JOINS + ACTIVE_OLAP_FUNCS +
ACTIVE_HASH_GRPBYS) AS TOTAL_ACTIVE_SORTS
FROM TABLE(MON_GET_DATABASE(-1));

TOTAL_ACTIVE_SORTS
--------------------
                   6

  1 record(s) selected.
```

Using these monitoring elements, you can determine the maximum number of concurrent sort memory operations. If concurrency is high, consider a lower ratio of `SORTHEAP` and `SHEAPTHRES_SHR` database configuration parameters. For example:

- Set `SORTHEAP` to a value of (`SHEAPTHRES_SHR/5`) for low concurrency
- Set `SORTHEAP` to a value of (`SHEAPTHRES_SHR/20`) for higher concurrency

### Monitoring total sort memory and overflows

The following monitoring elements report the total number sort memory consumers executed on the database:

- `TOTAL_SORTS`

  Total number of sorts that have been executed.

- `TOTAL_HASH_JOINS`

  Total number of hash joins executed.

- `TOTAL_OLAP_FUNCS`

  Total number of OLAP functions executed.

- `TOTAL_HASH_GRPBYS`

  Total number of hashed `GROUP BY` operations.

The following monitoring elements report the total number of sort overflows on the database. When there are sort overflows, results are spilled to the system temporary table which causes undesirable disk access. This degrades performance and should be avoided.

- `SORT_OVERFLOWS`

  The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.

- `HASH_JOIN_OVERFLOWS`

  The number of times that hash join data exceeded the available sort heap space.

- `OLAP_FUNC_OVERFLOWS`

  The number of times that OLAP function data exceeded the available sort heap space.

- `HASH_GRPBY_OVERFLOWS`

  The number of times that `GROUP BY` operations use hashing as their grouping method and exceeded the available sort heap memory.

Example 6-22 shows an SQL query that reports the total sort usage and the number of sort operations overflowed.

*Example 6-22 Total sort consumers and number of sort operations overflowed*

```
SELECT
   (TOTAL_SORTS + TOTAL_HASH_JOINS + TOTAL_OLAP_FUNCS +
TOTAL_HASH_GRPBYS) AS TOTAL_SORT_CONSUMERS,
   (SORT_OVERFLOWS + HASH_JOIN_OVERFLOWS + OLAP_FUNC_OVERFLOWS +
HASH_GRPBY_OVERFLOWS) AS TOTAL_SORT_CONSUMER_OVERFLOWS
FROM TABLE (MON_GET_DATABASE(-1));

TOTAL_SORT_CONSUMERS TOTAL_SORT_CONSUMER_OVERFLOWS
-------------------- -----------------------------
                 178                             0

  1 record(s) selected.
```

Optionally, you can use the two values to compute the percentage of sort memory operations that spilled to disk. To do so, divide the total sort memory consumers overflowed by the total number of sorts used. If the sort overflow percentage is high, there is a large percentage of sort operations throttled and spilled to disk.

In this case, consider the following possibilities:

- Increase the SHEAPTHRES_SHR database configuration parameter so that more memory is available for concurrent sort operations.
- Adjust the Workload Management concurrency limits for the workload to reduce the number of concurrently executing sort operations.

For more details of monitoring sort consumer overflows, see this web page:

http://www.ibm.com/developerworks/data/library/techarticle/dm-1407monitor-bluaccel/index.html.

# 6.6 Workload management

From DB2 9.5, a set of features was introduced into the DB2 engine, in the form of DB2 Workload Manager (WLM). With DB2 WLM, you can treat separate workloads (applications, users, and so on) differently, and provide them with different execution environments to run in. It improves the workload management process by ensuring priority workloads get the most resources allocated to them.

## 6.6.1 Automatic workload management

DB2 10.5 with BLU Acceleration builds on DB2 Workload Manager (WLM) and includes automated query resource consumption controls to deliver even higher performance to your analytic database environments. With BLU Acceleration, there is a threshold limit on the number of “heavyweight” queries that can run against a database at any one time. This ensures that heavier workloads that use column-organized data will not overload the system and affect the lighter weight queries.

Several default workload management objects are created for new or converted BLU Acceleration databases. Particularly, a new service subclass called SYSDEFAULTMANAGEDSUBCLASS is enabled for analytics workload. It is specifically for handling heavy, long running queries and is bounded by the SYSDEFAULTCONCURRENT threshold. Table 6-1 summarizes the WLM components for this service subclass.

*Table 6-1 WLM components for the SYSDEFAULTMANAGEDSUBCLASS*

| WLM component | Name | Description |
|---|---|---|
| service subclass | SYSDEFAULTMANAGEDSUBCLASS | Controls and manages heavyweight queries. |
| threshold | SYSDEFAULTCONCURRENT | Controls the number of concurrently running queries that are running in the SYSDEFAULTMANAGEDSUBCLASS. |

| WLM component | Name | Description |
|---|---|---|
| work class | SYSMANAGEDQUERIES | Identifies the class of heavyweight queries to control. This includes queries that are classified as READ DML (a work type for work classes) and that exceed a timeron threshold that reflects heavier queries. |
| work class set | SYSDEFAULTUSERWCS | Identifies the class of heavyweight queries to control. |

These workload management objects help control the processing of light and heavyweight queries. Figure 6-1 illustrates how the automatic WLM process works for analytic workloads.

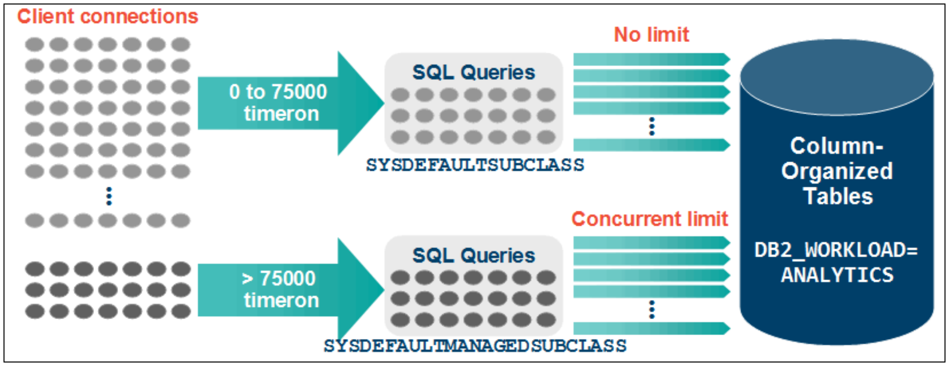



*Figure 6-1 Automatic WLM process for BLU Acceleration*

Any read-only queries submitted to the system are categorized based on the query cost estimate generated by DB2 cost-based optimizer into a managed and unmanaged query classes. Lightweight queries that are below a defined cost (and non-read-only activities) are categorized as unmanaged and are allowed to enter the system with no admission control. This avoids a common problem with queuing approaches in which the response time of small tasks that require modest resources can suffer disproportionately when queued behind larger, more expensive tasks.

For heavyweight queries above the timeron cost, DB2 categorizes these as managed, and applies an admission control algorithm that operates based on the processor parallelism of the underlying hardware. When a certain number of managed queries are running on the server, further submitted queries are queued if the current number of long-running queries is at the upper limit to optimize the current hardware. This is done transparently to the user, who sees only the resulting performance benefit from a reduction in spilling and resource contention among queries.

### Automatic SYSDEFAULTCONCURRENT threshold for heavy queries

The threshold limit, SYSDEFAULTCONCURRENT, is automatically enabled on new databases if you set the value of the DB2_WORKLOAD registry variable to ANALYTICS. Its value is calculated during database creation or when database is autoconfigured, based on system hardware attributes such as the number of CPU sockets, CPU cores, and threads per core.

This limit is captured in the SYSCAT.THRESHOLDS catalog view. To retrieve this limit, you can use the query shown in Example 6-23.

*Example 6-23 WLM threshold*

```
SELECT
   substr(thresholdname,1,25) AS THRESHOLDNAME,
   SMALLINT(maxvalue) AS MAXVALUE,
   decode(enforcement,'W','WORKLOAD','D','DATABASE','P','PARTITION',
'OTHER') AS ENFORCEMENT
FROM syscat.thresholds
WHERE thresholdname='SYSDEFAULTCONCURRENT';
```

Example 6-24 shows output of the query. In this example, the maximum number of heavy queries that can run at the same time is 14, enforced on database level.

*Example 6-24 System WLM concurrent limit*

```
THRESHOLDNAME             MAXVALUE ENFORCEMENT
------------------------- -------- -----------
SYSDEFAULTCONCURRENT            14 DATABASE

  1 record(s) selected.
```

### Changing threshold when hardware specs has changed

If your hardware specification has changed since SYSDEFAULTCONCURRENT threshold was last auto-configured, you can use the configuration advisor to generate a new threshold suitable for your new hardware. Example 6-25 shows an AUTOCONFIGURE APPLY NONE command to review the recommended changes:

*Example 6-25 Review autoconfigure recommended values*

```
db2 AUTOCONFIGURE APPLY NONE

Current and Recommended Values for System WLM Objects

Description                                 Current Value  Recommended
Value
-----------------------------------------------------------------------
------
Work Action SYSMAPMANAGEDQUERIES Enabled  = Y             Y
Work Action Set SYSDEFAULTUSERWAS Enabled = Y             Y
Work Class SYSMANAGEDQUERIES Timeroncost  = 1.50000E+05   1.50000E+05
Threshold SYSDEFAULTCONCURRENT Enabled    = Y             Y
Threshold SYSDEFAULTCONCURRENT Maxvalue   = 14            16
```

If the recommended value has changed, you can use the command in Example 6-26 to change the current SYSDEFAULTCONCURRENT threshold to a recommended value. This example demonstrates changing the threshold to a recommended value of 18 as an example.

*Example 6-26 Changing SYSDEFAULTCONCURRENT threshold*

```
db2 ALTER THRESHOLD SYSDEFAULTCONCURRENT
  WHEN CONCURRENTDBCOORDACTIVITIES > 16 STOP EXECUTION
```

## Display total CPU and queue time for each service subclass

If you want to drill-down further to observe how each workload management service class is performing, specifically, the wait time that is spent in the SYSDEFAULTMANAGEDSUBCLASS where heavy queries are run, use the command shown in Example 6-27.

*Example 6-27 SQL statement to determine CPU costs and total queue time in each service subclass*

```
SELECT
   service_superclass_name,
   service_subclass_name,
   sum(total_cpu_time) as TOTAL_CPU,
   sum(wlm_queue_time_total) as TOTAL_QUEUE_TIME
FROM TABLE(MON_GET_SERVICE_SUBCLASS('','',-2)) AS t
GROUP BY service_superclass_name, service_subclass_name
ORDER BY total_cpu desc;
```

This command returns the total CPU time, and the total queue time for every WLM service class, ordered by CPU time.

Example 6-28 shows output. The `WLM_QUEUE_TIME_TOTAL` metric indicates the accumulated time that queries wait on concurrency threshold. Ideally, `WLM_QUEUE_TIME_TOTAL` should be as low as possible.

*Example 6-28 Service subclasses and their total CPU costs and queue time*

```
SERVICE_SUPERCLASS_NAME      SERVICE_SUBCLASS_NAME        TOTAL_CPU    TOTAL_QUEUE_TIME
---------------------------- ---------------------------- ------------ ------------------
SYSDEFAULTMAINTENANCECLASS   SYSDEFAULTSUBCLASS           65179497     0
SYSDEFAULTUSERCLASS          SYSDEFAULTMANAGEDSUBCLASS    35227720522  0
SYSDEFAULTUSERCLASS          SYSDEFAULTSUBCLASS           198413509    0
SYSDEFAULTSYSTEMCLASS        SYSDEFAULTSUBCLASS           0            0

  4 record(s) selected.
```

If your application workload or business requirements have changed and you observe that the `WLM_QUEUE_TIME_TOTAL` of the `SYSDEFAULTMANAGEDSUBCLASS` is accumulated, this means that a large amount of queries are categorized as heavy queries and are bounded by a concurrency limit.

In this case, where the system is under-utilized, consider increasing the timeron cost minimum for the `SYSMANAGEDQUERIES` class. This action routes the less heavy queries back to the default subclass and minimizes the wait time for those subset of queries (`SYSDEFAULTSUBCLASS`).

Alternatively, consider increasing `the SYSDEFAULTCONCURRENT` threshold if the distribution of the service classes appears to be reasonable. Likewise, you can decrease the metrics if the system appears to be over-utilized.

## 6.7 Query optimization

SQL queries that are submitted within a DB2 with BLU Acceleration environment are processed as usual by the DB2 industry-leading, cost-based optimizer and query rewrite engine. For DBAs, being able to generate an *explain* plan, and do cost-analysis, remains the same.

The explain facility is invoked by issuing the EXPLAIN statement, which captures information about the access plan chosen for a specific explainable statement and writes this information to explain tables. You must create the explain tables before issuing the EXPLAIN statement. For further information about the explain facility, see the following web page:

http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp?topic=%2Fcom.ibm.db2.luw.admin.perf.doc%2Fdoc%2Fc0005137.html

Besides the **db2expln** and **db2exfmt** engine tools, comprehensive support is also available for explain in the new version of Optim Query Workload Tuner and Optim Performance manager. All these utilities and tools are enabled for BLU Acceleration.

## 6.7.1 CTQ operator

What is unique to BLU Acceleration queries is a columnar-table-queue (CTQ) operator for query execution plans. It represents a runtime boundary in the query execution plan. It indicates a transition between column-organized data processing and row-organized data processing. Anything below the CTQ operator in the query access plan is run on encoded, compressed column-organized data. Anything above the CTQ operator is run on non-encoded data.

Example 6-29 shows a statement to generate an explain plan for a query.

*Example 6-29 Sample command to generate explain plan for query*

```
db2expln -d GS_DB -f workload.sql -t -z ";" -g > explain.out
```

The following parameters are used:

| | |
|---|---|
| -d GS_DB | Specifies the name of the database (in our example, GS_DB). |
| -f workloads.sql | Is the input file containing the SQL statement to generate an explain plan for (in our case, workloads.sql). |
| -t | Sends the **db2expln** output to the terminal. |
| -z ";" | Specifies the semi-colon character as the SQL statement separator. |
| -g | Shows optimizer plan graphs in the output. |
| > explain.out | Redirects output from terminal to file. |

Figure 6-2 shows a sample BLU query explain plan. Note the CTQ operator. We can see that most operators lie below this CTQ boundary, which indicates that our query is optimized for columnar processing.

```
Access Plan:
-----------
     Total Cost:              79342.5
     Query Degree:            1

                Rows
               RETURN
               (   1)
                Cost
                 I/O
                 |
                  5
                CTQ
               (   2)
               79342.5
               14985.9
                 |
                  5
               GRPBY
               (   3)
               79342.5
               14985.9
                 |
               60553.6
               HSJOIN
               (   4)
               79341.6
               14985.9
         /-------+-------\
     60553.6           2.99942e+06
     TBSCAN              TBSCAN
     (   5)              (   6)
     20638.6             58685.8
     3925.53             11060.4
       |                   |
     1.5e+06           6.00122e+06
 CO-TABLE: DB2INST1  CO-TABLE: DB2INST1
     ORDERS             LINEITEM
       Q2                  Q1
```

*Figure 6-2 An example of a good query execution plan for column-organized tables*

Ideally, you want as much of the plan run below the CTQ operator. In good execution plans for column-organized tables, the majority of operators are below the CTQ operator, and only a few rows flow through the CTQ operator (Table 6-2).

*Table 6-2 CTQ recommendations*

| **Optimal plan** | **Suboptimal plan** |
|---|---|
| One or few CTQ operators | Many CTQs |
| Few operators above CTQ | Many operators above CTQ |
| Operators above CTQ work on few rows | Operators above CTQ work on many rows |
| Few rows flow through the CTQ | Many rows flow through the CTQ |

In DB2 10.5, the following examples are of some operators that are optimized for column-organized tables:

- Table scan operators
- Hash-based join operators
- Hash-based group by operators
- Hash based unique operators

## 6.7.2 Time spent on column-organized table processing

Time-spent monitor elements provide information about how the DB2 database manager spends time processing column-organized tables. The time-spent elements are broadly categorized into wait time elements and processing time elements. The columnar-related monitor elements listed in Table 6-3 are added to the time-spent monitoring hierarchy.

*Table 6-3 Time spent on queries metrics in the Time area*

| **Elements** | **Description** |
|---|---|
| TOTAL_SECTION_TIME | Total column-organized section time monitor element: Represents the total time agents spent performing section execution. The value is given in milliseconds. |
| TOTAL_COL_TIME | Total column-organized time monitor element: Represents the total elapsed time over all column-organized processing subagents. |
| TOTAL_SECTION_PROC_TIME | Total column-organized section process time monitor element: Represents the total amount of processing time agents spent performing section execution. Processing time does not include wait time. The value is given in milliseconds. |

| Elements | Description |
|---|---|
| TOTAL_COL_PROC_TIME | Total column-organized processing time monitor element: Represents the subset of this total elapsed time in which the column-organized processing subagents were not idle on a measured wait time (for example, lock wait or IO). |
| TOTAL_COL_EXECUTIONS | Total column-organized executions monitor element: Represents the total number of times that data in column-organized tables was accessed during statement execution. |

You can use these elements, in the context of other “time-spent” monitor elements, to determine how much time was spent, per thread, in performing column-organized data processing. For example, if you want to know what portion of a query was run in column-organized form and what portion was executed in row-organized form, you can compare `TOTAL_COL_TIME` to `TOTAL_SECTION_TIME`. A large ratio suggests an optimal execution plan for BLU Acceleration.

Use the query in Example 6-30 to obtain the time spent on column-organized data processing.

*Example 6-30 Columnar processing times*

```
SELECT
   total_col_time AS TOTAL_COL_TIME,
   total_col_executions AS TOTAL_COL_EXECUTIONS,
   total_section_time  AS TOTAL_SECTION_TIME
FROM
   table(mon_get_database(-1));
```

Your query output should be like the output in Example 6-31.

*Example 6-31 Time spent on queries key metrics*

```
TOTAL_COL_TIME       TOTAL_COL_EXECUTIONS TOTAL_SECTION_TIME
-------------------- -------------------- --------------------
           103679097                  121            114185091

  1 record(s) selected.
```

Notice that most of the total time was spent on column-organized data processing (103,656,167 ms out of 114,111,844 ms in the example). This means that our query is using the column-organized data processing advantages of DB2 with BLU Acceleration for the majority of the elapsed processing time.

### 6.7.3 Observing query performance

The `MON_GET_PKG CACHE_STMT` table function returns report metrics that relate to key database metrics, such as I/O server efficiency, processing time for authentication, statistics generation, and statement execution. You can use this table function to identify possible problematic static and dynamic SQL statements for both row and columnar query processing.

`MON_GET_PKG CACHE_STMT` returns a point-in-time view of SQL statements in the database package cache. This has a similar query from Example 6-30 on page 213, except that this is reported per SQL query execution. This allows you to examine the aggregated metrics for a particular SQL statement, and to quickly determine the performance of an executed query. The query in Example 6-32 collects execution times of the statements from the package cache by using the `MON_GET_PKG_CACHE_STMT` table function.

*Example 6-32 Statement execution times*

```
SELECT
   varchar(stmt_text,30) AS STATEMENT,
   substr(total_cpu_time/num_exec_with_metrics,1,12) AS AVG_CPU_TIME,
   total_col_time,
   total_section_time,
   rows_read,
   rows_returned
FROM
   table(mon_get_pkg_cache_stmt( NULL, NULL, NULL, -1))
WHERE num_exec_with_metrics <> 0;
```

Example 6-33 shows output from a Cognos sample workload. It shows that the query section took 76861 ms to execute (`TOTAL_SECTION_TIME`). Out of the total section time, 73471 ms was spent in column-processing (`TOTAL_COL_TIME`). That is, about 96% of the time was spent in column-processing and using BLU Acceleration technologies to optimize the query.

*Example 6-33 Total column-processing time versus total section time for queries*

```
STATEMENT                      AVG_CPU_TIME TOTAL_COL_TIME
TOTAL_SECTION_TIME
------------------------------ ------------ --------------
------------------
select "T0"."C0" "C0" , "T0"." 928702      73471        76861

ROWS_READ  ROWS_RETURNED
---------- -------------
239968     420
```

## 6.7.4 Average number of columns referenced in workload

A new monitor element, TAB_ORGANIZATION, reports information about the organization of data in a table and is returned by the MON_GET_TABLE table function.

BLU Acceleration works best on queries that access only a subset of table columns. This is because of the nature of column-organized table processing. In an environment where majority or all of the workloads tend to access all columns in the tables, traditional row-organized table might achieve more efficient performance rather than scanning different column-organized tables individually.

To determine the average number of columns that a query accesses, you can divide NUM_COLUMNS_REFERENCED by the SECTION_EXEC_WITH_COL_REFERENCES monitor element. If this average is much less than the number of columns in the table, the query is accessing only a small subset of the columns. We can confirm that the workload favors column organization.

Example 6-34 shows how to assess if our workload is columnar in nature.

*Example 6-34 Query for assessing if a workload is columnar in nature*

```
SELECT
   varchar(b.tabname,25) AS TABNAME,
   num_columns_referenced AS NUM_COLUMNS_REFERENCED,
   section_exec_with_col_references AS TOTAL_SECS,
   cast(num_columns_referenced*1.0/section_exec_with_col_references as
decimal(2,1)) AS COL_PER_SEC,
   colcount AS COLCOUNT,
   card AS ROWCOUNT
FROM
   table(mon_get_table('<BLU_SCHEMA>', '<BLU_TABLENAME>',-1)) a,
   syscat.tables b
WHERE a.tabschema = b.tabschema
AND a.tabname = b.tabname
AND section_exec_with_col_references > 0
ORDER BY 2 DESC;
```

Example 6-35 on page 216 shows the result of the query against a series of Cognos Sample Outdoor Warehouse sample analytic workloads. You can see in this test case, that the average number of columns that a particular query accesses in our workload is using the column data organization and we are not scanning all individual column-organized tables for the query.

*Example 6-35 Columns per query section versus total column count in tables*

```
TABNAME          NUM_COLUMNS_REFERENCED SECTIONS COL_PER_SEC COLCOUNT ROWCOUNT
---------------- ---------------------- -------- ----------- -------- ----------
GO_TIME_DIM                         275 38               7.2       73 2930
GO_REGION_DIM                       239 34               7.0       66 21
SLS_RTL_DIM                         187 32               5.8       88 847
SLS_PRODUCT_DIM                     184 28               6.5       14 3000300
SLS_SALES_FACT                      184 40               4.6       18 2999547946
```

Example 6-35 shows that the total column count of each tables (`card AS COLCOUNT`) is significantly larger than the average number of columns being queried per section (computed `COL_PER_SEC`). Therefore, we can conclude that this series of executed workload found in the package cache is ideal for BLU Acceleration.

### 6.7.5 The MONREPORT module

DB2 10.5 also has several modules you can use to generate reports containing a variety of monitoring information.

One of these modules, the `MONREPORT` module, provides a set of procedures for retrieving a variety of monitoring data and generating text reports. In DB2 10.5, this module was updated to include columnar monitor elements.

In particular, the procedures that were updated are as follows:

- `MONREPORT.CONNECTION`: This procedure outputs a summary report based on connection information from `MON_GET_CONNECTION`.
- `MONREPORT.DBSUMMARY`: This procedure outputs a summary report, based on `MON_GET_SERVICE_SUBCLASS`, `MON_GET_CONNECTION` and `MON_GET_WORKLOAD`.

You use the following DB2 commands to call these procedures:

```
db2 call monreport.connection;
db2 call monreport.dbsummary;
```

The report contains a good amount of monitor information for the entire database and also key performance indicators for each connection, workload, service class, and database member.

For a sample output of the database summary report from monreport, see A.1, “Sample Monreport output” on page 412.

7

# Oracle compatibility for BLU Acceleration

The DB2 SQL Compatibility feature was first introduced in DB2 9.7. It enables applications that are written for Oracle to run natively on DB2. Client code and stored procedures that rely on PL/SQL, data types, error handling, and other Oracle-specific functionalities are natively supported by DB2. For users who are moving from Oracle databases to DB2 with BLU Acceleration, this feature simplifies and makes the conversion process easier.

This chapter discusses the SQL Compatibility feature for BLU Acceleration. We demonstrate the benefits of adopting DB2 with BLU Acceleration for Oracle applications.

The following topics are covered:

- DB2 SQL Compatibility feature overview
- Deploying DB2 BLU Acceleration with SQL Compatibility
- Migrating Oracle to DB2 with BLU Acceleration

# 7.1 DB2 SQL Compatibility feature overview

Since version 9.7, DB2 includes a built-in DB2 SQL Compatibility feature which supports running Oracle applications natively. It has extensive support for PL/SQL procedural language, Oracle data types, scalar functions, built-in packages, locking mechanisms, SQL*Plus, and much more. With DB2 SQL Compatibility, applications built for Oracle database can run on DB2 with minimal or no changes. Current Oracle DBAs and developers can keep their current skills while working with DB2 with BLU Acceleration. For example, current DBAs can use the PL/SQL skills to work with DB2. Existing test cases can also be reused on DB2 with few or no changes.

The SQL Compatibility feature is included in DB2 with BLU Acceleration Cancun Release (10.5.0.4) and later. Oracle databases can now easily be converted into BLU Acceleration to take advantage of the benefits of the DB2 dynamic in-memory column-organized store and to speed analytic workloads.

In this section, we go through different Oracle compatibility features that are covered in DB2.

## 7.1.1 Data types

Data types define how a database engine handles each type of data and how it is stored and represented. Even though there is an American National Standards Institute (ANSI) standard for data types, most relational database management system (RDBMS) vendors implement their own set of data types. Data type mismatch can be a challenge during a RDBMS migration.

DB2 supports the Oracle nonstandard data types, such as numeric (for example, NUMBER), strings (for example, VARCHAR2), and dates (for example, DATE). More complex types that are commonly used in Oracle PL/SQL are also supported. This approach makes it possible for any Oracle object definition to be supported during a migration process.

## 7.1.2 SQL standard

There is also an ANSI standard on SQL definitions. However, some RDBMS vendors also come with their own SQL implementation standard. The differences in vendor specific SQL syntaxes can make a migration process complex.

With the SQL Compatibility feature, DB2 supports Oracle keywords and semantics and dramatically reduce the need to convert different SQL syntaxes. Examples of Oracle keywords supported are: `DUAL`, `OUTER JOIN` operator (+), `TRUNCATE TABLE`, character literals, collection methods, and Oracle data dictionaries.

## 7.1.3 PL/SQL

DB2 engine includes a SQL PL compiler that natively supports compilation of Oracle PL/SQL. It generates virtual machine code at the runtime engine level and does not involve any translation. Because of that situation, compiled PL/SQL and SQL PL code have the same level of performance and efficiency as any other native DB2 language element. DBAs and developers can continue working with the language they are familiar with. Package application vendors can also use the same source code against both Oracle and DB2.

## 7.1.4 Concurrency management

Concurrency or locking management is another item that requires attention during a migration process or evaluation.

Traditionally, DB2 cursor stability (CS) isolation level is being used in most production environments because it balances data integrity and data access performance. Cursor stability waits for the pending transaction changes and ensures that the data being read is committed. Since version 9.7, DB2 has added a *Currently Committed* database parameter that can change the behavior of the CS isolation level. If currently committed parameter (**`cur_commit`**) is turned on, DB2 retrieves the currently committed version of a locked row from the log buffer and the writer does not block the reader.

Figure 7-1 shows a concurrency scenario where there are two sessions looking for the same data. User 1 is updating data and user 2 is trying to read the same data before the application commits the change made by user 1. Traditionally, user 2 would have to wait for user 1's change to commit. With currently committed, user 2 reads the originally committed data from the log buffer and does not have to wait for session one to commit its change.

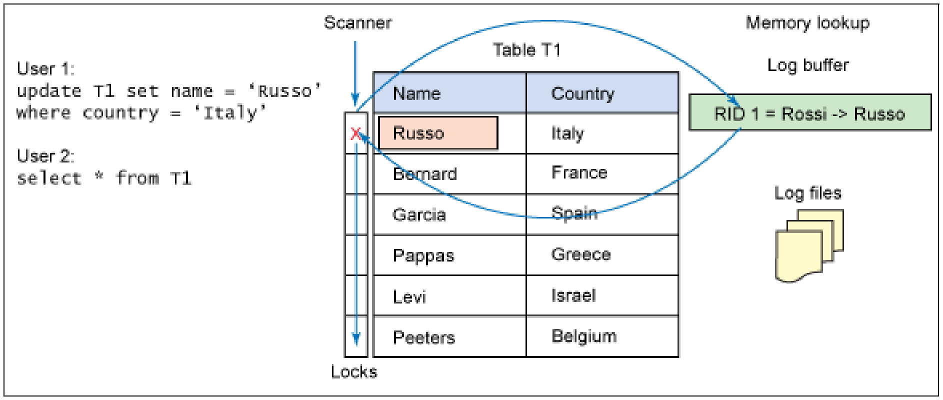



*Figure 7-1 Concurrency behavior with currently committed feature enabled*

## 7.1.5 Built-in functions and packages

Most RDBMS offer built-in functions, packages, and modules to provide pre-built functionality and enrich the SQL user experience. DB2 is like these offerings and offers its own built-in functions. In addition, with the SQL Compatibility feature, DB2 also supports the most used Oracle built-in functions and packages.

### Built-in functions

With SQL Compatibility feature turned on, DB2 extends the support for Oracle built-in functions and increases the compatibility with Oracle applications.

The following list includes examples of Oracle built-in functions that are available in DB2:

- TO_DATE
- TO_CHAR
- TO_NUMBER
- NVL
- NVL2
- SUBSTR
- INSTR
- LISTAGG
- LTRIM
- LPAD

### Built-in packages or modules

Many Oracle applications use built-in packages, commonly called Database Management System (DBMS). Table 7-1 lists the most used built-in packages in Oracle applications that are supported in DB2.

*Table 7-1 Built-in packages or modules available in DB2*

| Package name | Description |
|---|---|
| DBMS_OUTPUT | Provides basic reporting capabilities that can be switched on or off from the command line |
| UTL_FILE | A module that allows working with files on the DB2 server |
| DBMS_SQL | A package that provides an SQL API to perform dynamic SQL in addition to the existing `EXECUTE` and `EXECUTE IMMEDIATE` statements |
| UTL_MAIL | A module that allows sending email notifications from SQL |
| UTL_SMTP | A lower-level API similar to UTL_MAIL that provides SMTP integration |
| DBMS_ALERT | A package that, when used, allows different sessions to semaphore one another |
| DBMS_PIPE | A module that allows sessions to send each other data |
| DBMS_JOB | Provides a compatible API that integrates with DB2's task scheduler |
| DBMS_LOB | An Oracle API for LOB processing that echoes DB2's built-in LOB functions |
| DBMS_UTILITY | A collection of various procedures used in applications |
| DBMS_DDL | Functions to obfuscate PL/SQL objects to protect vendor IP |
| UTL_DIR | Provides a set of routines to maintain directory aliases used by UTL_FILE |

## 7.1.6 SQL*Plus scripting

Oracle application developers and database administrators (DBAs) often use the SQL*Plus command line processor and develop SQL*Plus scripts. DB2 provides a command line utility called *CLPPlus* that provides similar functionality to SQL*Plus.

With CLPPlus, Oracle professionals who are familiar with the SQL*Plus syntaxes do not need to do any modification on their scripts or learn a new tool. They can continue to use their SQL*Plus skills to do column formatting, reporting functions, variable substitution, control variables, and use SQL*Plus-compatible command options in DB2 CLPPlus.

### 7.1.7 Oracle Call Interface (OCI) and Pro*C

Oracle Call Interface (OCI) and Pro*C are two application programming interfaces (APIs) used to access Oracle databases through applications developed in C or C++ programming languages.

DB2 also has its own implementations to provide support for development using C and C++, known as *DB2CI*. DB2CI provides more than 150 OCI APIs and supports existing OCI applications that work with DB2 databases. In addition, Embedded SQL is also available in DB2 and covers the same functionality as Pro*C.

### 7.1.8 Miscellaneous features and capabilities

The following Oracle features and capabilities are supported and understood by DB2:

- DUAL
- OUTER JOIN OPERATOR (+)
- ANONYMOUS BLOCKS
- TRUNCATE TABLE
- CREATE TEMPORARY TABLES
- RECORD and COLLECTION TYPES
- COLLECTION METHODS
- Oracle Data Dictionaries
- Oracle Database Links
- INOUT parameters
- ROWNUM
- Hierarchical queries (CONNECT BY)
- Implicit casting

For a complete reference about the SQL Compatibility feature, see *Oracle to DB2 Conversion Guide: Compatibility Made Easy*, SG24-7736 at this web address:

http://www.redbooks.ibm.com/abstracts/sg247736.html

# 7.2 Deploying DB2 BLU Acceleration with SQL Compatibility

DB2 with BLU Acceleration speeds the use of analytics and helps organizations gain faster business insights with innovative technologies. With the SQL Compatibility feature, current Oracle database users can move their data marts and warehouses to DB2 easily to take advantage of the unique benefits provided by BLU Acceleration.

Enabling DB2 SQL Compatibility feature is a simple and quick process. It is activated and controlled by one single registry variable.

This section describes how to create and enable a BLU Acceleration database with SQL Compatibility.

## 7.2.1 Enabling SQL Compatibility for BLU Acceleration

As we have seen in 2.5, “Deployment” on page 24, BLU Acceleration can be activated by setting a single register variable `DB2_WORKLOAD=ANALYTICS`. This variable automatically optimizes a database instance and a database for analytics workload.

### DB2_COMPATIBILITY_VECTOR registry variable

Similar to BLU Acceleration, the **DB2_COMPATIBILITY_VECTOR** is a single registry variable that enables one or more DB2 Compatibility features.

To enable SQL Compatibility feature for Oracle applications, set the register variable to **ORA**, shown in Example 7-1.

*Example 7-1 Setting SQL Compatibility register variable*

```
db2set DB2_COMPATIBILITY_VECTOR=ORA
```

### Creating a BLU Acceleration database with SQL Compatibility

To enable Oracle compatibility and BLU Acceleration for a database, simply set two register variables, **DB2_COMPATIBILITY_VECTOR=ORA** and **DB2_WORKLOAD=ANALYTICS** and run a few commands. Example 7-2 shows the commands to create a database enabled for Oracle compatibility and BLU Acceleration.

**Set variables:** The DB2_COMPATIBILITY_VECTOR variable must be set before the database creation. It indicates to DB2 that the database should be created in SQL Compatibility mode. After the database is created, the compatibility mode cannot be modified for that same database.

Set both DB2_COMPATIBILITY_VECTOR and DB2_WORKLOAD registry variables before database creation to enable the BLU Acceleration database for SQL Compatibility feature.

*Example 7-2 Creating DB2 BLU database with SQL Compatibility*

```
db2set DB2_COMPATIBILITY_VECTOR=ORA
db2set DB2_DEFERRED_PREPARE_SEMANTICS=YES
db2set DB2_ATS_ENABLE=YES
db2set DB2_WORKLOAD=ANALYTICS
db2stop
db2start
db2 CREATE DATABASE sales
db2 CONNECT TO sales
db2 UPDATE DB CFG USING auto_reval deferred_force
db2 UPDATE DB CFG USING decflt_rounding round_half_up
TERMINATE
db2 DEACTIVATE DATABASE sales
db2 ACTIVATE DATABASE sales
```

As demonstrated in Example 7-2, in addition to the main register variables, DB2_COMPATIBILITY_VECTOR and DB2_WORKLOAD, there are other register variables and database parameters to be configured to set up SQL Compatibility for BLU Acceleration appropriately.

### *DB2_DEFERRED_PREPARE_SEMANTICS*

The register variable DB2_DEFERRED_PREPARE_SEMANTICS works with DB2_COMPATIBILITY_VECTOR to enhance compatibility between Oracle applications and DB2 such as those written in Java. By setting this registry variable to YES, dynamic SQL statements are not evaluated at the PREPARE step, but rather are evaluated on OPEN or EXECUTE calls. You can use this setting to take advantage of the DB2 implicit data type casting feature. It also avoids errors that might otherwise occur during the PREPARE step when untyped parameter markers are present.

### *DB2_ATS_ENABLE*

If you plan to use Oracle scheduled jobs through the use of DBMS_JOB built-in package, you must set the register variable **DB2_ATS_ENABLE=YES**, which activates the Administrative Task Scheduler facility. This facility automates the execution of a scheduled task and enables administrative tasks in DB2. Setting DB2_ATS_ENABLE is optional for SQL Compatibility and should be set only if you want to use DBMS_JOB to handle scheduled jobs.

### *AUTO_REVAL*

The AUTO_REVAL parameter controls the automatic objects revalidation behavior when invalid objects are encountered. The default value is DEFERRED that means if an object such as, a view or function is invalidated for any reason, an attempt to revalidate the object happens automatically the next time it is referenced. An example of invalidation is dropping of an underlined table.

Changing the AUTO_REVAL parameter value to DEFERRED_FORCE allows new objects to be created successfully, even though they might depend on invalid objects.

### *DECFLT_ROUNDING*

The DECFLT_ROUNDING database configuration parameter specifies the rounding mode for a decimal floating point (DECFLOAT). This parameter defaults to round-half-even. You can set it to round-half-up to match the Oracle rounding mode more closely.

After updating the database configuration parameters, you must deactivate and activate the database to commit the changes.

**Best practice:** It is a best practice to set the PAGE SIZE to 32K in the CREATE DATABASE statement for SQL Compatibility enabled databases. This setting is to ensure large rows moving from source Oracle database so they fit into DB2 target tables. When DB2_WORKLOAD is set to ANALYTICS, default page size is set to 32K. It is not necessary to explicitly specify the 32K pagesize in a CREATE DATABASE statement.

### 7.2.2 Verifying SQL Compatibility set up

To verify if SQL Compatibility is correctly enabled on a database, review the following database configuration parameters using the **db2 get db cfg** command.

Figure 7-2 lists a few parameters that should be set to ON (turned on) when the database is in Oracle Compatibility mode.

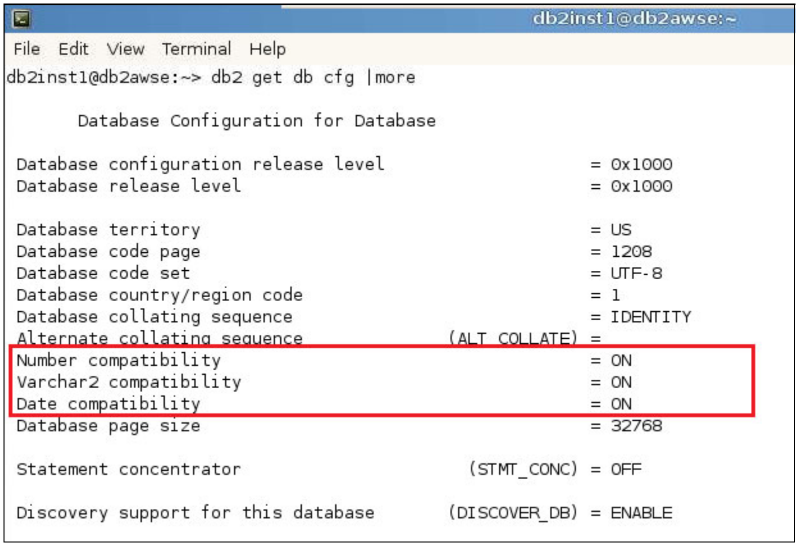



*Figure 7-2 Verify whether Oracle compatibility is set*

> **Verify parameters:** To verify the parameters that are configured in a BLU Accelerated database, refer to 2.6, “Configuration preferred practices for BLU Acceleration deployment” on page 31.

## 7.3 Migrating Oracle to DB2 with BLU Acceleration

Migrating or converting databases from one DBMS to another can be a complex project. Items such as different data types, data isolation behavior, specific SQL syntaxes, and database procedural language can involve a tremendous amount of effort.

With the SQL Compatibility feature, DB2 is approximately 98% compatible with Oracle's syntax and behavior. Along with the conversion tool, IBM Database Conversion Workbench (DCW), the migration process from Oracle to DB2 becomes easily manageable and migration effort is dramatically reduced.

This section shows a complete conversion process scenario using the compatibility features explained in 7.1, "DB2 SQL Compatibility feature overview" on page 218. We also describe the different conversion process phases based on numerous successful client experiences.

This section does not cover the Data Studio and DCW plug-in installations. It focuses on the migration process only.

For a complete guidance about how to install and configure Data Studio and DCW, see *Oracle to DB2 Conversion Guide: Compatibility Made Easy*, SG24-7736 at the following web address:

http://www.redbooks.ibm.com/abstracts/sg247736.html

## 7.3.1 A general conversion process overview

Before going through a database conversion scenario using DCW, we explain the different phases involved in a database conversion process:

- *DDL extraction*: Data Definition Language (DDL) is the syntax for defining and altering data structures of a database. DDL statements are used to create, modify, and drop database objects, such as tables, indexes, users, and routines. DDL extraction is the process of obtaining the required DDL statements from the source database so existing database objects can be replicated on the target database.
- *Assessment and conversion*: The assessment and conversion phase involves examining the DDL statements of the source database to appropriately plan for the conversion project. Traditionally, this phase involved manually reviewing and converting thousands of DDL statements. This phase is typically the most challenging and time consuming aspect of a database migration. IBM DCW helps streamline this process with the Compatibility Report and Auto-Convert feature, which dramatically reduces the work effort required.
- *Data movement*: This is the last phase of the conversion process and usually occurs after all compatibility issues between the source and target database objects are resolved. Data movement involves replicating the appropriate source database objects on the target database, mapping the data from the source database, and then migrating data into the target database.

Figure 7-3 shows a diagram of a complete Oracle to DB2 database conversion process. Each of these processes are streamlined by IBM Database Conversion Workbench (DCW) tools.

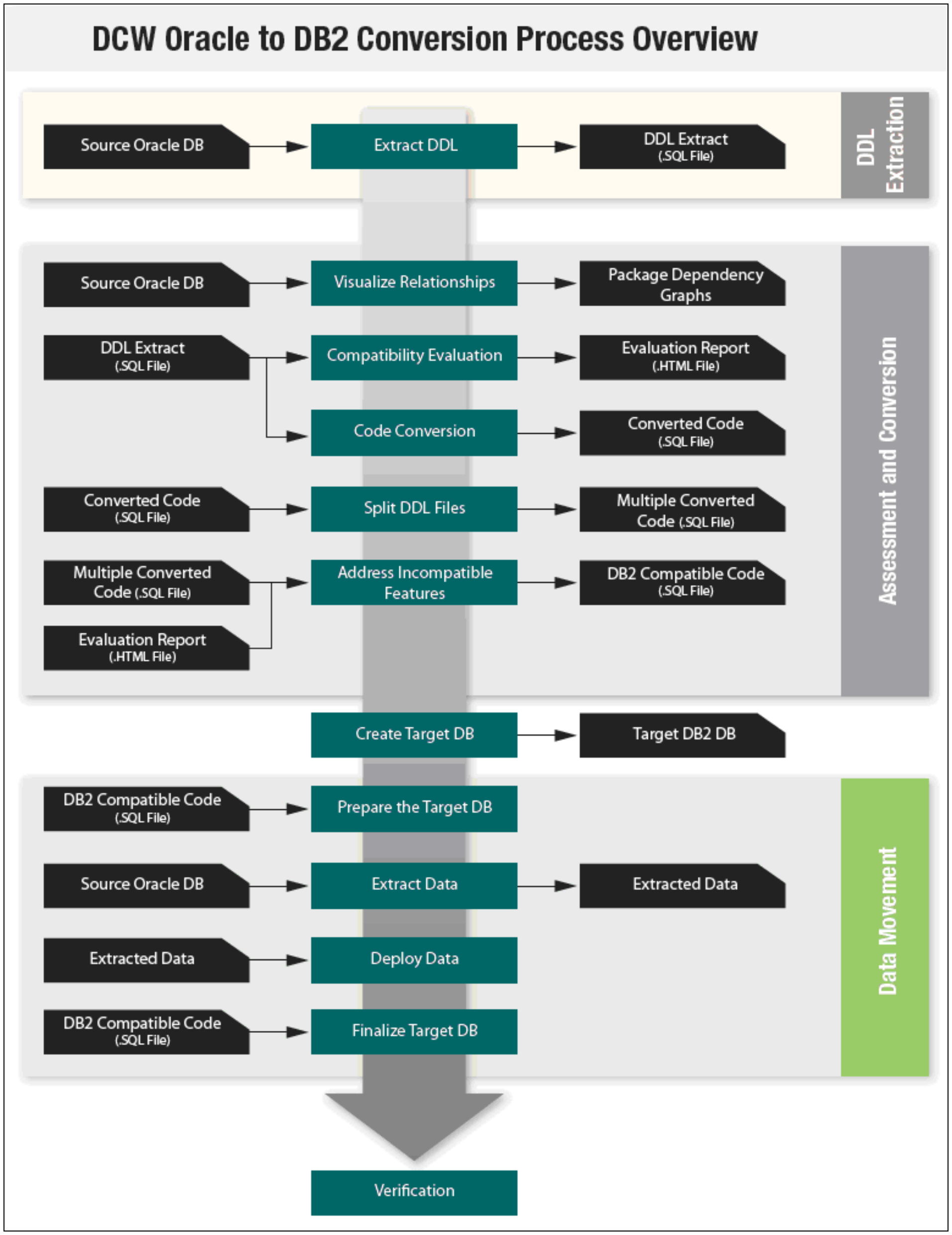



*Figure 7-3 Oracle to DB2 conversion process using DCW*

## 7.3.2 IBM Data Conversion Workbench tools

The IBM Database Conversion Workbench (DCW) provides an end-to-end solution through an integrated workbench of tools that facilitate your conversion to DB2. DCW is a no-charge plug-in that can easily be installed into IBM Data Studio, adding database migration capabilities. Whether you are converting to DB2 from another RDBMS or migrating from one version of DB2 to another version, DCW provides an easy-to-use framework to take you through the conversion process.

DCW combines many of the tools that are used for database conversion into a single graphical user environment. It follows processes that are based on several leading practices that are identified by IBM data migration consultants.

DCW provides the following benefits to the user:

- A common graphical interface with a uniform appearance for all phases of conversion
- A consolidated process with defined steps that are based on leading industry practices
- Faster data conversion by using automated tools
- Wizards that provide guidance through all the conversion steps

The conversion features in DCW can be used together to carry a user through all of the steps of a database conversion or, if the user prefers, each of the features can be used separately.

DCW offers several step-by-step functions to facilitate the conversion process:

- *DCW Task Launcher*: This integrated interface launches the various steps of the conversion process. The Task Launcher contains steps which, when clicked, launches the appropriate wizards and help topics.
- *DDL Extraction*: This function extracts the DDL of the objects in the source Oracle database. DDL is used by DCW to analyze the source database and provides compatibility assessment, conversion, and object creation on the target DB2 database. Various methods of extraction are available.
- *Compatibility Evaluation*: This function provides a report of the estimated compatibility (expressed as a percentage) of Oracle SQL and PL/SQL statements with DB2 10.1 or DB2 10.5. It outlines the major issues (if any) with the conversion and highlights code that must be fixed manually.
- *Code Conversion*: This function auto-converts known Oracle syntax to DB2-compatible syntax. This activity helps streamline what can otherwise be a long process.
- *Split DDL*: This function splits a single DDL file into multiple files that are organized by object types. This function provides more intuitive code organization by breaking down large DDL files into smaller and more manageable components.
- *Package Visualizer*: This function generates a dependency graph of objects in the source database.
- *Data Movement*: This function extracts and loads data from the source Oracle database to the target DB2 database. Various methods of extraction and loading are available.

Refer to the Database Conversion Workbench community website to download DCW documentation, user guides, manuals, and the plug-in available at this web address:

`http://www.ibm.com/developerworks/data/ibmdcw`

## 7.3.3 A migration scenario using DCW

For demonstration purposes, this section illustrates an Oracle data warehouse database containing a sales data mart migrating into DB2 with BLU Acceleration. This data mart is composed of a one fact table with multiple dimensions tables that are being represented as a star schema. The tables are located inside the database schema GOSALESDW. Figure 7-4 presents a diagram of the database star schema.

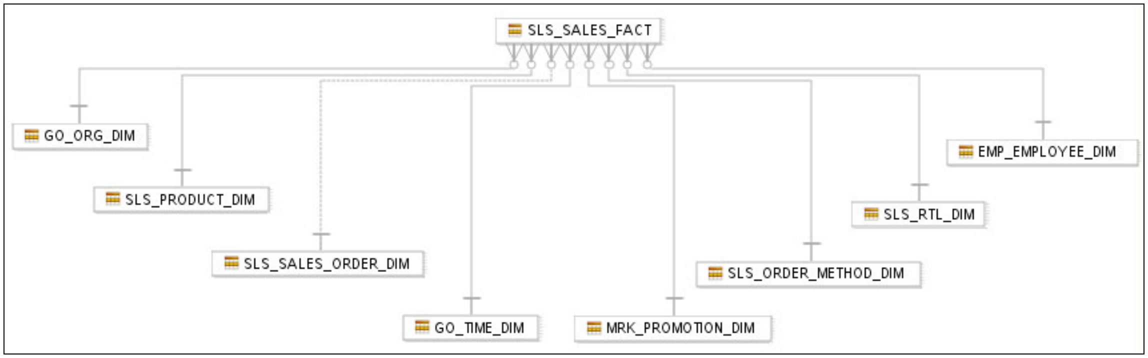



*Figure 7-4 Sample database star schema model*

> **Note:** Before preceding with the migration procedure, you must create the BLU Acceleration target database to which the Oracle objects and data are being migrated. In 7.2.1, "Enabling SQL Compatibility for BLU Acceleration" on page 223, we explain how to create a DB2 database with BLU Acceleration enabled with SQL Compatibility.

### Compatibility evaluation for BLU Acceleration

The first task in a migration project is to check if there are any Oracle features in use that are not yet supported by DB2. DCW can create a report listing all these incompatibilities by analyzing the DDL statements from the source database. If you do not have the DDL statements available, DCW can be used to connect to the source Oracle database and perform a DDL extraction.

In this section, we describe the steps to extract the Oracle DDL statements and generate a report on Oracle to DB2 with BLU Acceleration compatibility using DCW.

We assume a DB2 BLU Accelerated target database with SQL Compatibility enabled has been created following the guidelines described in 7.2.1, "Enabling SQL Compatibility for BLU Acceleration" on page 223.

### *Extracting Oracle DDL using DCW*

DCW provides a tool to extract Oracle database objects definitions into DDL statements. Follow these steps to extract Oracle DDLs:

1. Create a new DCW project:

   a. In Data Studio, navigate to menu **File** → **New** → **Project** and choose the **DCW project** wizard.

   b. Enter the **Project Name**, choose the Source Type as **Oracle**, and the Target database as **DB2 LUW 10.5 with BLU** (Figure 7-5).

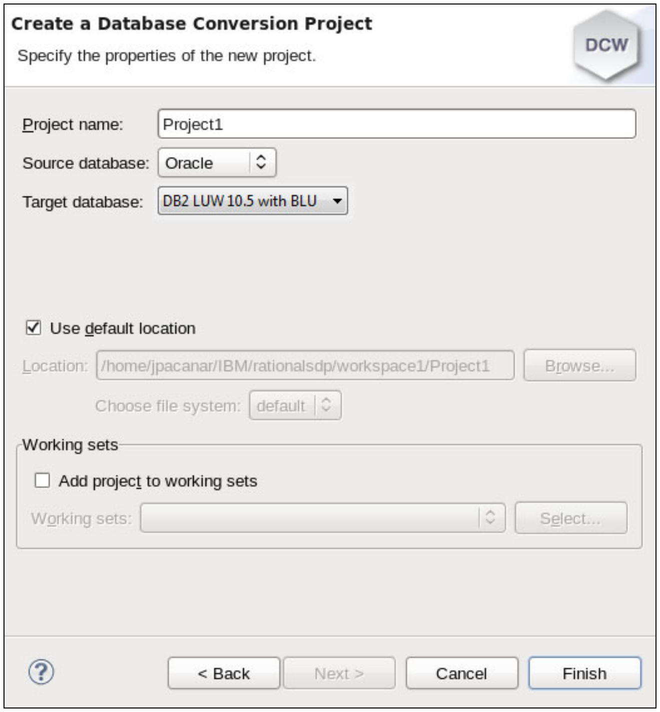



*Figure 7-5 Creating a new DCW Project*

   c. Click **Finish** and you have a new project in the Project Explorer panel.

2. Extract DDL statements from the source Oracle database:

   a. In the Project Explorer view, right-click the project name created in step 1 on page 232. Select **Database Conversion** → **Extract DDL**. The DDL Extraction wizard is displayed.

   b. Select the extraction method and click **Finish**. In this case, we are selecting the extraction through a connection (Figure 7-6). For a large database, we suggest to generate a DDL extraction script instead.

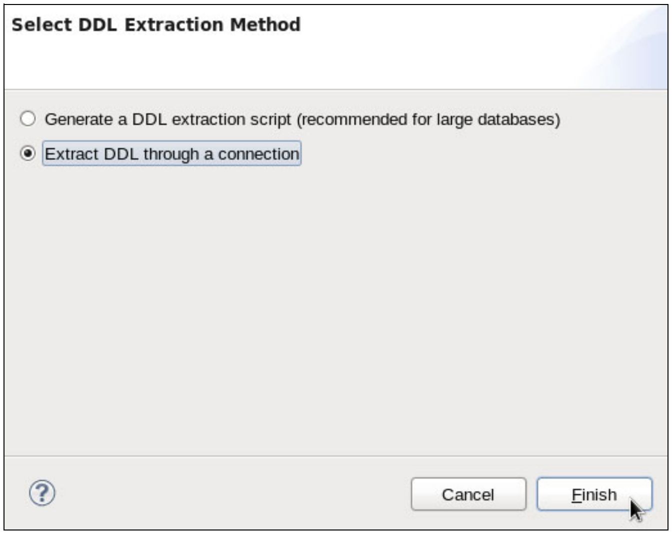



*Figure 7-6 DDL Extraction method window*

   c. A new wizard window opens with your choice in the last step. For an extraction through a connection, specify the connection for the source Oracle database and then specify one or more database schemas to extract. Click **Next**.

   d. Choose the options and the objects to be extracted and click **Next**.

   e. The process creates the extracted script and asks you to review and save it. Depending on the number of objects, it could take a few minutes to create the script. Click **Next**.

f. Review the summary window that shows the setting chosen and click **Finish** shown in Figure 7-7. This ends the process and you have the extracted script saved to your DCW project.

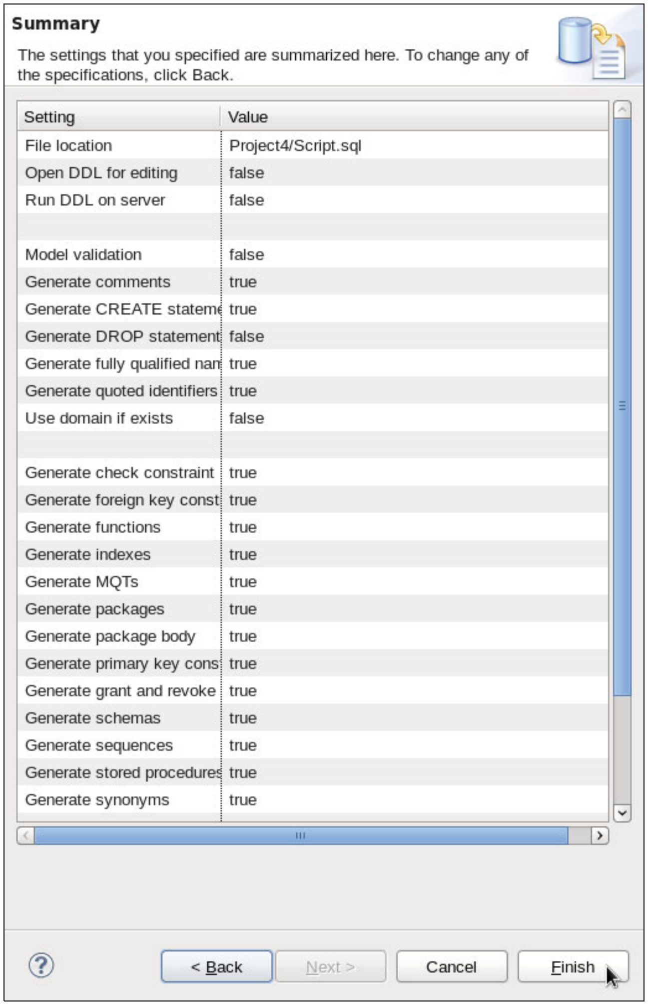



*Figure 7-7 Extraction DDL summary window*

### *Compatibility evaluation*

DCW can evaluate Oracle DDL statements, generate a Compatibility Evaluation report, and identify any incompatibilities between the source Oracle and target DB2 databases. The report lists any features or syntaxes that would result in errors if the DDL is executed on DB2 with BLU Acceleration.

**Conversion project:** When you select "DB2 10.5 with BLU" as your target for the conversion project, DCW assumes that all tables in the DDL will be column-organized when performing the compatibility evaluation and code conversion.

The Compatibility Evaluation report presents two different types of incompatibilities:

- The first type includes incompatibilities that are not supported by DB2 regardless if BLU Acceleration is enabled or not.
- The second type includes incompatibilities that are specific to BLU Acceleration. They should only be considered if you are using column-organized tables, otherwise, they can be ignored. For example, BLU Acceleration does not support the Character Large OBject (CLOB) data type on a column-organized table, DCW reports any tables that have CLOB columns. If the table will not be created as column-organized, you can simply ignore the entry in the report.

Complete the following steps to start evaluating compatibility of the source Oracle database and DB2 with BLU Acceleration:

1. In DCW Project, right-click the DDL script generated. Select **Database Conversion** → **Evaluate Compatibility.** The Evaluate Compatibility wizard opens. Select **DB2 10.5 BLU** as the target database as illustrated in Figure 7-8.

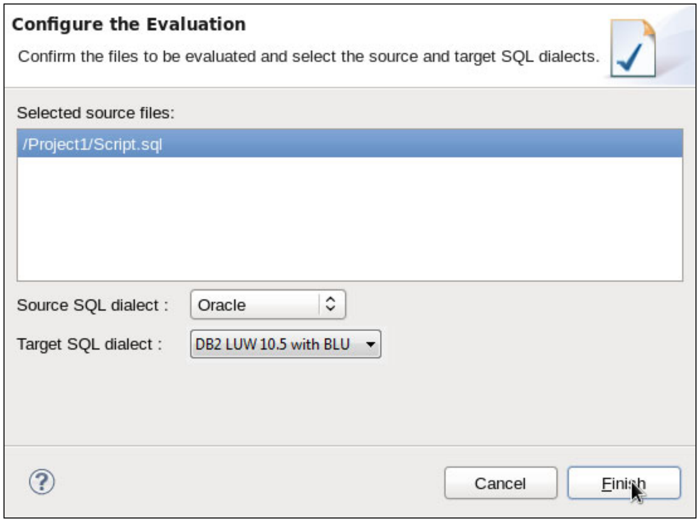



*Figure 7-8 Evaluate compatibility wizard*

2. Click **Finish** and a progress windows opens. A message box appears indicating compatibility evaluation is successful (Figure 7-9).

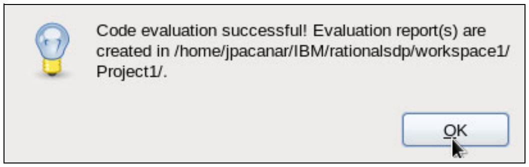
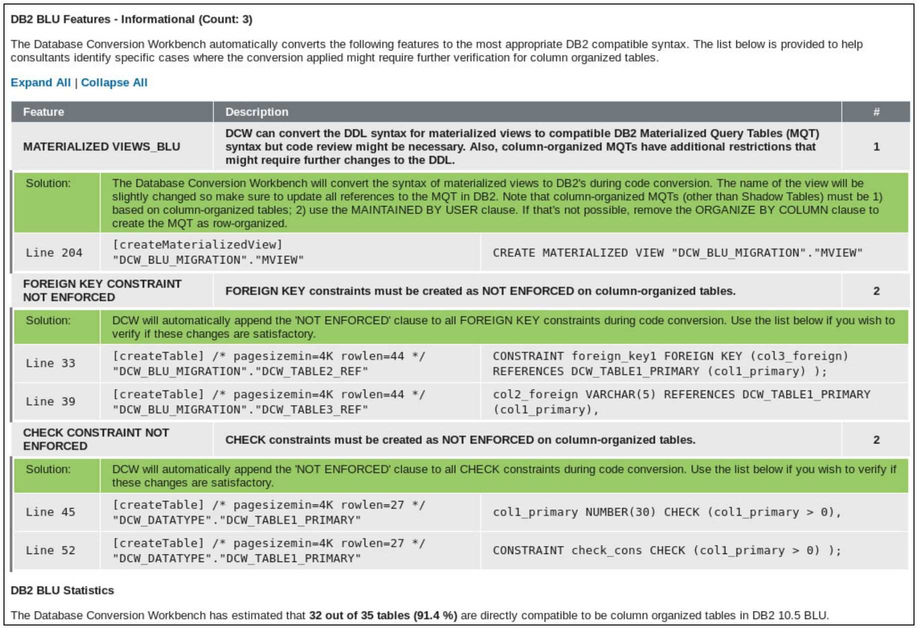



*Figure 7-9 Evaluation report generated successfully*

3. The process generates an encrypted XML file with file extension *xmle*. It lists the incompatibilities found during the evaluation. To get a nicely formatted HTML report, attach and email the *xmle* file to *askdcw@ca.ibm.com*.

4. Review and analyze the evaluation report generated, perform corrections on the possible incompatibilities found, if any. The report shows general incompatibilities when moving Oracle to DB2 and specific issues when using BLU Acceleration. The BLU Acceleration incompatibilities are the same as reported by the **db2convert** utility when migrating from a regular DB2 row-based model to column-organized BLU Acceleration model. For each issue found, DCW offers a possible solution. Figure 7-10 shows a sample compatibility evaluation report with the issues found and possible solutions. To see the solutions, click the issue to expand it.



**DB2 BLU Features - Informational (Count: 3)**

The Database Conversion Workbench automatically converts the following features to the most appropriate DB2 compatible syntax. The list below is provided to help consultants identify specific cases where the conversion applied might require further verification for column organized tables.

**Expand All** | **Collapse All**

| Feature | Description | | # |
|---|---|---|---|
| **MATERIALIZED VIEWS_BLU** | **DCW can convert the DDL syntax for materialized views to compatible DB2 Materialized Query Tables (MQT) syntax but code review might be necessary. Also, column-organized MQTs have additional restrictions that might require further changes to the DDL.** | | **1** |
| Solution: | The Database Conversion Workbench will convert the syntax of materialized views to DB2's during code conversion. The name of the view will be slightly changed so make sure to update all references to the MQT in DB2. Note that column-organized MQTs (other than Shadow Tables) must be 1) based on column-organized tables; 2) use the MAINTAINED BY USER clause. If that's not possible, remove the ORGANIZE BY COLUMN clause to create the MQT as row-organized. | | |
| `Line 204` | `[createMaterializedView] "DCW_BLU_MIGRATION"."MVIEW"` | `CREATE MATERIALIZED VIEW "DCW_BLU_MIGRATION"."MVIEW"` | |
| **FOREIGN KEY CONSTRAINT NOT ENFORCED** | **FOREIGN KEY constraints must be created as NOT ENFORCED on column-organized tables.** | | **2** |
| Solution: | DCW will automatically append the 'NOT ENFORCED' clause to all FOREIGN KEY constraints during code conversion. Use the list below if you wish to verify if these changes are satisfactory. | | |
| `Line 33` | `[createTable] /* pagesizemin=4K rowlen=44 */ "DCW_BLU_MIGRATION"."DCW_TABLE2_REF"` | `CONSTRAINT foreign_key1 FOREIGN KEY (col3_foreign) REFERENCES DCW_TABLE1_PRIMARY (col1_primary) );` | |
| `Line 39` | `[createTable] /* pagesizemin=4K rowlen=44 */ "DCW_BLU_MIGRATION"."DCW_TABLE3_REF"` | `col2_foreign VARCHAR(5) REFERENCES DCW_TABLE1_PRIMARY (col1_primary),` | |
| **CHECK CONSTRAINT NOT ENFORCED** | **CHECK constraints must be created as NOT ENFORCED on column-organized tables.** | | **2** |
| Solution: | DCW will automatically append the 'NOT ENFORCED' clause to all CHECK constraints during code conversion. Use the list below if you wish to verify if these changes are satisfactory. | | |
| `Line 45` | `[createTable] /* pagesizemin=4K rowlen=27 */ "DCW_DATATYPE"."DCW_TABLE1_PRIMARY"` | `col1_primary NUMBER(30) CHECK (col1_primary > 0),` | |
| `Line 52` | `[createTable] /* pagesizemin=4K rowlen=27 */ "DCW_DATATYPE"."DCW_TABLE1_PRIMARY"` | `CONSTRAINT check_cons CHECK (col1_primary > 0) );` | |

**DB2 BLU Statistics**

The Database Conversion Workbench has estimated that **32 out of 35 tables (91.4 %)** are directly compatible to be column organized tables in DB2 10.5 BLU.

*Figure 7-10 DB2 BLU Compatibility Evaluation report*

## Database objects conversion

IBM Database Conversion Workbench provides the capability to convert Oracle syntaxes that are not yet supported by the SQL Compatibility feature into DB2 syntaxes. DCW together with the DB2 SQL Compatibility feature greatly simplifies the database conversion process.

This section demonstrates how to convert Oracle database objects and deploy them into a DB2 with BLU Acceleration database.

### *DDL conversion*

The Code Conversion wizard allows you to convert one or more files containing SQL statements in Oracle syntax to the specified target DB2 syntax.

The following steps describe how to convert source Oracle database objects into DB2 syntaxes:

1. Select the existing **DCW Project** and select the DDL script extracted from "Extracting Oracle DDL using DCW" on page 232. Right-click **Database Conversion** → **Convert Code**. The Code Conversion wizard window opens as shown in Figure 7-11.

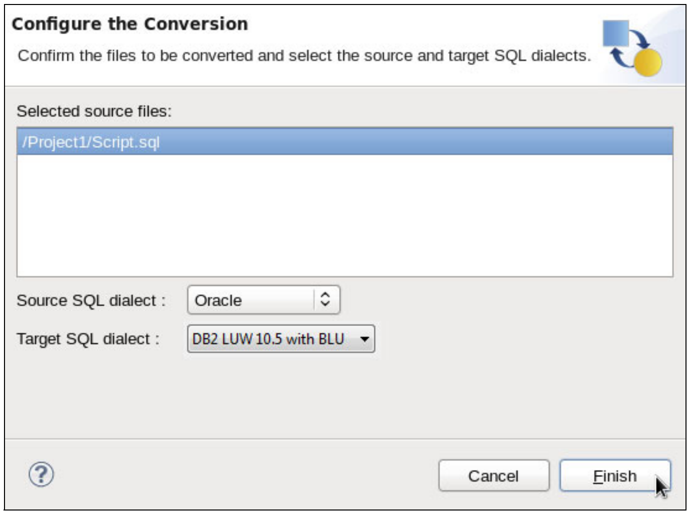



*Figure 7-11 Code conversion wizard*

2. Click **Finish** and wait for conversion processing. A new converted script is created in the project tree in the naming convention of `<Script name>-<timestamp>_converted.sql`.

3. Double-click the converted script and review the content. Verify the converted objects including messages from the DCW evaluation process defined by the comment lines `/* *** DCW Evaluation ... */`.

### *DDL splitting*

To facilitate objects deployment, DCW offers the functionality to split the converted script file into multiple parts. The split process creates multiple files organized by different types of objects. For example, one file for tables creation statements, one file for index creation statements, and so on. This provides a more intuitive way to deploy the objects in a correct sequence of dependency. It also becomes more manageable to make different changes as needed:

1. To split a converted DDL script file, right-click the converted script under the **DCW Project**. Select **Database Conversion** → **Split DDL Files**. The split DDL wizard window opens. Follow instructions in the wizard and click **Finish**.
2. A new entry appears under the project, divided into multiple scripts organized by object types.

   Figure 7-12 illustrates the different DDL files.

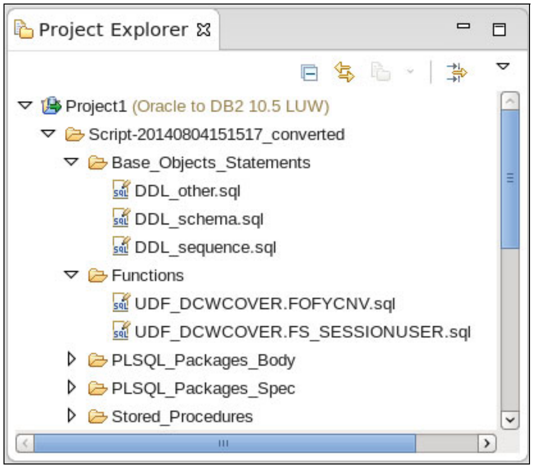



*Figure 7-12 Converted DDL script split for multiple scripts*

### *DDL deployment*

After all DDL scripts are converted into DB2 syntax, the next step is to execute the converted DDL scripts and create the objects in the target DB2 database:

1. Choose one or more converted database objects creation scripts that you want to run, right-click and select **Database Conversion** → **Run DDL Files**. This action calls the Run DDL Files wizard and displays the list of scripts to run.

2. Select the statement delimiter of the DDL file from the drop-down list, then click **Next**, as illustrated in Figure 7-13. Choose the target database and click **Finish** to run the DDL scripts.

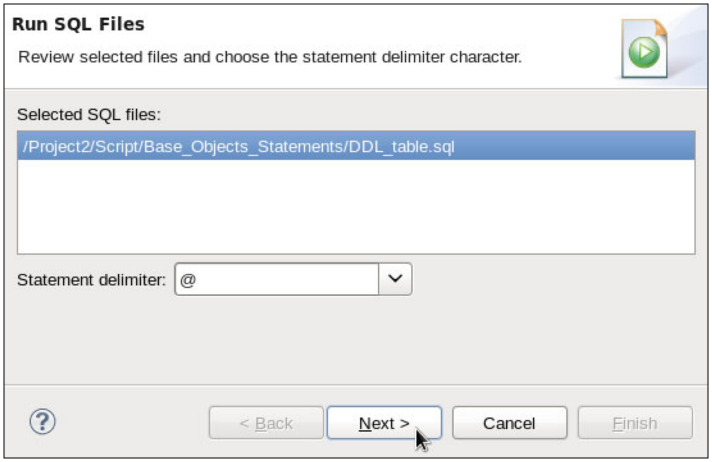



*Figure 7-13 Running and deploying DDL scripts*

3. The results are displayed in the **SQL Results** tab. Repeat the steps in this section until all desired Oracle objects are deployed in the target DB2 with BLU Acceleration database.

> **PL/SQL conversion:** Details on PL/SQL conversion or other tasks are out of the scope of this book. For further information, see *Oracle to DB2 Conversion Guide: Compatibility Made Easy*, SG24-7736 available at this web address:
>
> http://www.redbooks.ibm.com/abstracts/sg247736.html

## Data movement using DCW

When all the database objects are deployed in the DB2 with BLU Acceleration database, you are ready to start the data movement process.

IBM Database Conversion Workbench provides various data movement methods, including flat files, connection pipes, database federation, and IBM InfoSphere CDC. In this section, we demonstrate the most common data movement option that is using flat files. This option involves extracting data from source Oracle database into delimited flat files and loading data from flat files into the target DB2 database. Keep in mind that this two-step load process requires additional temporary disk spaces to store temporary flat files.

In this section, we use the same **DCW Project** created in the previous sections, unload data from the source Oracle database and load into the target DB2 with BLU Acceleration database.

### *Extracting data*

To extract data, users might require certain administrative privileges on the source Oracle database. With these credentials, DCW can establish a JDBC connection with the source database and fetch data through simple `SELECT * FROM` table SQL queries. During the data extraction, load scripts for the flat files are generated.

1. Right-click the **DCW Project** and choose **Database Conversion** → **Extract Data**. This opens the Extract Data wizard.
2. Specify a connection to the source Oracle database.
3. Select one or more schemas that you want to move and enter the output directory for the flat files and logs, as demonstrated in Figure 7-14. Click **Next**.

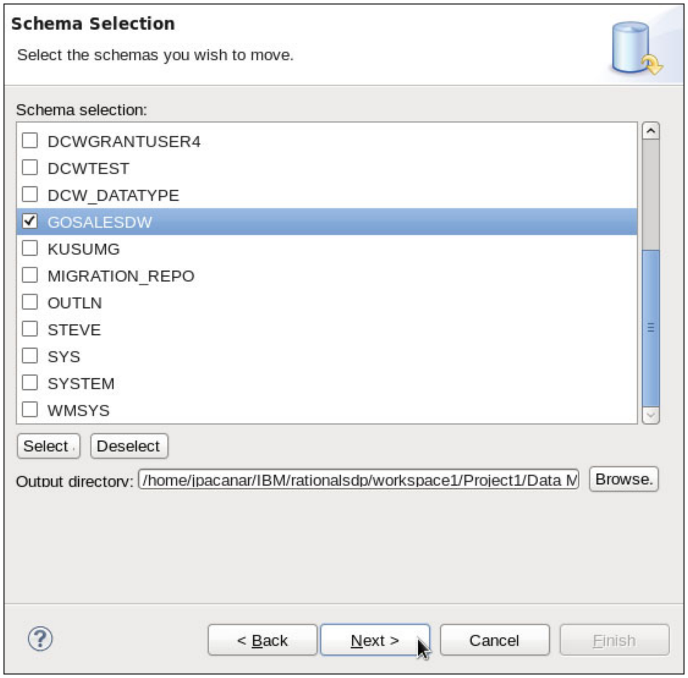



*Figure 7-14 Choosing schemas to extract data*

4. Select the tables to move, as illustrated in Figure 7-15.

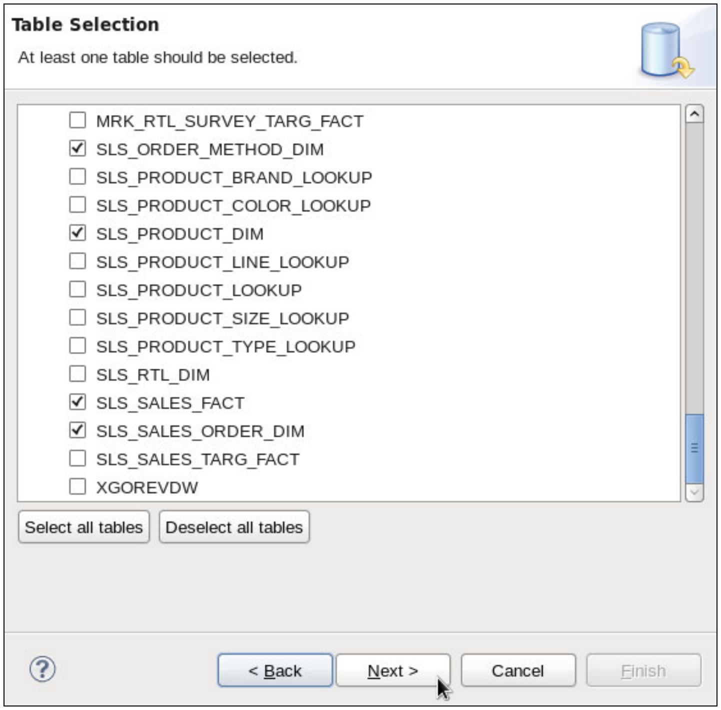



*Figure 7-15 Selecting tables to unload the data*

5. Optionally specify the desired data movement configuration parameters. Click **Next**, as illustrated in Figure 7-16.

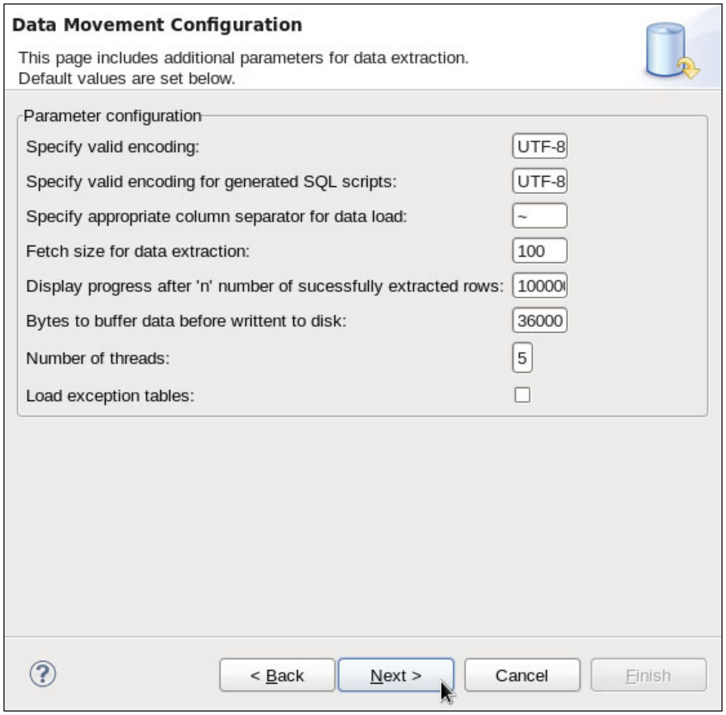



*Figure 7-16 Data movement configuration parameters*

6. Review the configuration summary for the data movement process and click **Finish** to begin extracting data, as demonstrated in Figure 7-17.

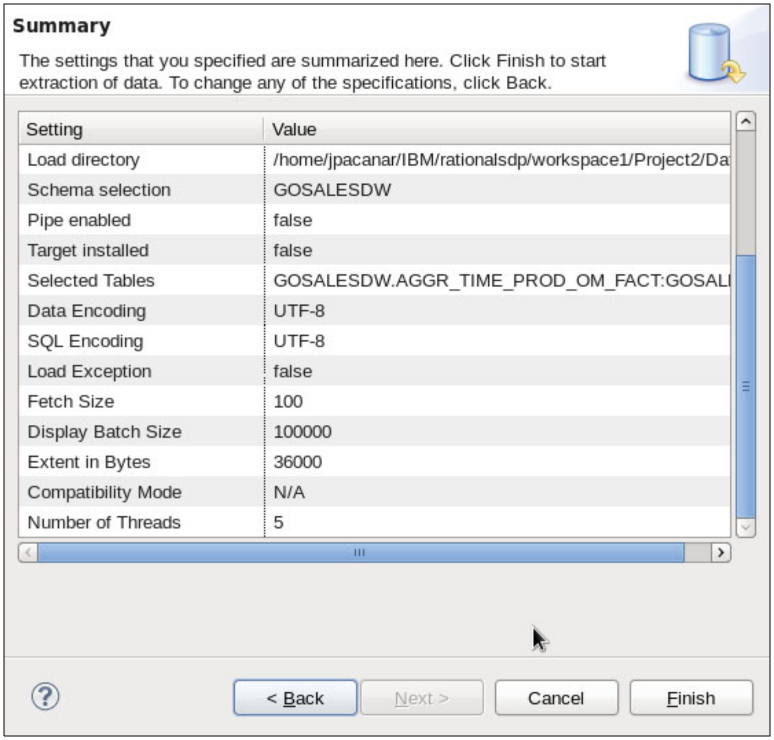



*Figure 7-17 Data movement configuration summary*

7. When the process is finished, a data movement log is displayed. Review the extraction results in the log.

### *Loading data*

After data is extracted to flat files, the data can be loaded into the target DB2 with BLU Acceleration database. The process to load data using DCW is similar to data extraction. Before you perform the load, the prerequisites to load data into the target database are as follows:

- The user who runs the LOAD requires LOAD privileges on the target database.
- DCW must be installed on the target database server.

Proceed with the following steps to start loading data from flat files generated in "Extracting data" on page 240:

1. Right-click the **DCW Project** and select **Database Conversion** → **Load Data**. This opens the Load Data wizard.
2. You are prompted to select the target database. Select the target DB2 with BLU Acceleration database and click **Next**.
3. In the next window, select the load directory where the generated flat files are located. Click **Next**.
4. Optionally, you can check the post-deployment options and click **Next**.
5. Review the summary and click **Finish**. This initiates the data load process and a progress window appears. Review the process results when it completes.
6. When the load is finished, a load status window appears. Review the status and verify the number of rows from the source and target. Ensure that all rows have been successfully loaded.

## 7.3.4 Conclusion

After the database conversion is completed and all data has been migrated to DB2, modify your applications to connect to the new DB2 with BLU Acceleration database instead of Oracle. The steps can vary depending on the programming language, database drivers, and clients used. This activity is out of the scope of this book.

Although a data migration might appear to be a complex project, moving an Oracle database into DB2 with BLU Acceleration can be a lot simpler than you think. With the SQL Compatibility feature, existing code and staff skills can be reused on DB2. BLU Acceleration significantly improves analytic performance and greatly enhances user experience. It also offers significant cost savings with flexible licensing terms from IBM and further lowers database maintenance efforts and ownership costs.

# 8

# DB2 with BLU Acceleration and SAP integration

This chapter contains information about the integration of DB2 for Linux, UNIX, and Windows (DB2) with BLU Acceleration into SAP Business Warehouse (SAP BW) and into the SAP BW near-line storage solution (NLS). We list the SAP releases in which BLU Acceleration is supported and the SAP and DB2 software prerequisites.

We discuss ABAP Dictionary integration of BLU Acceleration and how to monitor the DB2 columnar data engine in the SAP DBA Cockpit, as well as the usage of BLU Acceleration in SAP BW in detail. We explain which SAP BW objects BLU Acceleration can be used with, how to create new SAP BW objects that use column-organized tables, and how to convert existing SAP BW objects from row-organized to column-organized tables.

We provide information about installing SAP BW systems on DB2 10.5 and about the handling of BLU Acceleration in migrations of SAP BW systems to DB2 10.5. Results from IBM internal lab tests give a description of performance and compression benefits that you can achieve with BLU Acceleration in SAP BW systems. We also provide preferred practices and suggestions for DB2 parameter settings when BLU Acceleration is used in SAP BW.

We explain is the usage of BLU Acceleration in SAP's near-line storage solution on DB2 in detail, showing for which SAP BW objects in the near-line storage archive BLU Acceleration can be used, and how to create new data archiving processes (DAPs) that use column-organized tables in the near-line storage database.

Finally, we demonstrate installing an NLS database on DB2 10.5 and the preferred practices for DB2 parameter settings when using BLU Acceleration in SAP BW NLS systems.

The following topics are covered:

- Introduction to SAP Business Warehouse (BW)
- Prerequisites and restrictions for using BLU Acceleration in SAP BW
- BLU Acceleration support in the ABAP Dictionary
- BLU Acceleration support in the DBA Cockpit
- BLU Acceleration support in SAP BW
- Conversion of SAP BW objects to column-organized tables
- Deployment
- Performance of SAP BW with BLU Acceleration
- BLU Acceleration for SAP near-line storage solution on DB2 (NLS)

## 8.1 Introduction to SAP Business Warehouse (BW)

This section briefly describes the elements of the SAP BW information model. The information in this section is an excerpt from the SAP Help Portal for SAP BW (**Technology** → **SAP NetWeaver Platform** → **7.3** → **Modeling** → **Enterprise Data Warehouse Layer**)[1].

SAP BW is part of the SAP NetWeaver platform. Key capabilities of SAP BW are the buildup and management of enterprise, and departmental data warehouses for business planning, enterprise reporting and analysis. SAP BW can extract master data and transactional data from various data sources and store it in the data warehouse. This data flow is defined by the SAP BW information model (Figure 8-1).

Furthermore, SAP BW is the basis technology for several other SAP products such as SAP Strategic Enterprise Management (SAP SEM), SAP Business Planning and Consolidation (SAP BPC).

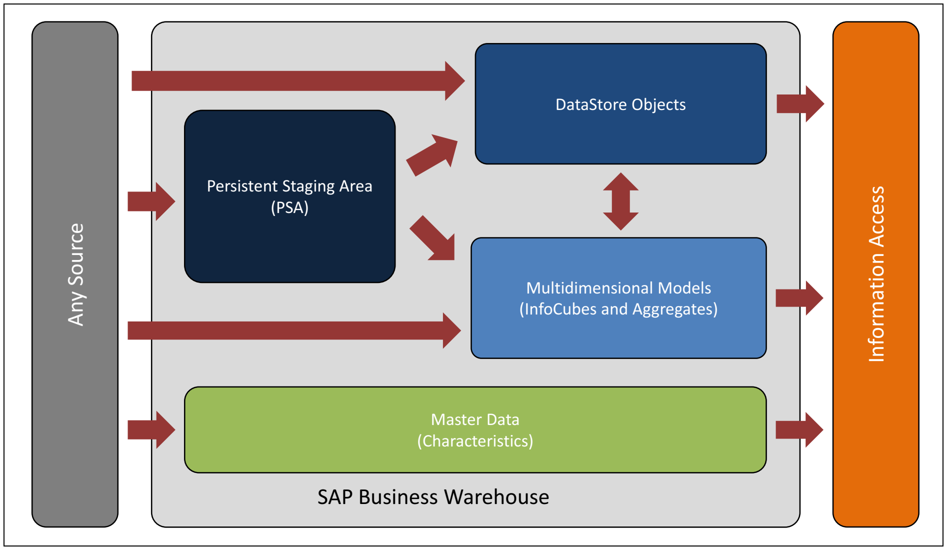



*Figure 8-1 SAP BW information model*

[1] https://help.sap.com/saphelp_nw73/helpdata/en/d5/cd581d7ebe41e88cb6022202c956fb/content.htm?frameset=/en/32/5e81c7f25e44abb6053251c0c763f7/frameset.htm

### 8.1.1 Persistent Staging Area (PSA)

Usually, data extracted from source systems is first stored directly in the Persistent Staging Area (PSA) as it was received from the source system. PSA is the inbound storage area in SAP BW for data from the source systems. PSA is implemented with one transparent database table per data source.

### 8.1.2 InfoObjects

InfoObjects are the smallest information units in SAP BW. They can be divided into three groups:

- Sets of *characteristics*, modeled as master data references, form entities. Such a reference remains unchanged, even if the corresponding value (attribute) is changed. Additional time characteristics are used to model the state of an entity over the course of time.
- Numerical attributes of entities are modeled as *key figures*. They are used to describe their state at a certain point of time.
- *Units* and other *technical characteristics* are used for processing and transformation purposes.

InfoObjects for characteristics consist of several database tables. They contain information about time-independent and time-dependent attributes, hierarchies, and texts in the languages needed. The central table for characteristics is the surrogate identifiers (SIDs) table that associates the characteristics with an integer called SID. That is used as foreign key to the characteristic in the dimension tables of InfoCubes.

### 8.1.3 DataStore Objects (DSOs)

DSOs store consolidated and cleansed transactional or master data on an atomic level. SAP BW offers three types of DSOs:

- Standard
- Write-optimized
- Transactional

Write-optimized and transactional DSOs consist of one database table that is called the DSO active table.

Standard DSOs consist of three database tables:

- The activation queue table
- The active table
- The change log table

The data that is loaded into standard DSOs is first stored in the activation queue table. The data in the activation queue table is not visible in the data warehouse unless it has been processed by a complex procedure that is called DSO data activation. Data activation merges the data into the active table and writes before and after images of the data into the change log table. The change log table can be used to roll back activated data if the data is not correct or other issues occur.

### 8.1.4 InfoCubes

InfoCubes are used for multidimensional data modeling according to the star schema. There are several types of InfoCubes. Two of them are introduced in this section:

- Standard InfoCubes
- Flat InfoCubes

#### Standard InfoCubes

SAP BW uses the enhanced star schema as shown in Figure 8-2 on page 250, which is partly normalized. An InfoCube consists of two fact tables, F and E. They contain recently loaded and already compressed data. To provide a single point of access, a UNION ALL view is defined on these two tables. The fact tables contain the key figures and foreign keys to up to 16 dimension tables. The dimension tables contain an integer number DIMID as primary key and the SIDs to the characteristics that belong to the dimension.

The F fact table is used for staging new data into the InfoCube. Each record contains the ID of the package with which the record was loaded. To accelerate the load process, data from one package is loaded to the F fact table in blocks. Although a certain combination of characteristics will be unique in each block, there can be duplicates in different blocks and packages. By the assignment of loaded data to packages, incorrect or mistakenly loaded data can be deleted easily as long as it resides in the F fact table.

The process of bringing the data from the F fact table to the E fact table is called *InfoCube compression*. In the course of this, the relation to the package is removed and multiple occurrences of characteristic combinations are condensed. Thus the F fact table contains only unique combinations of characteristics. Compressed packages are deleted from the F fact table after compression is completed.

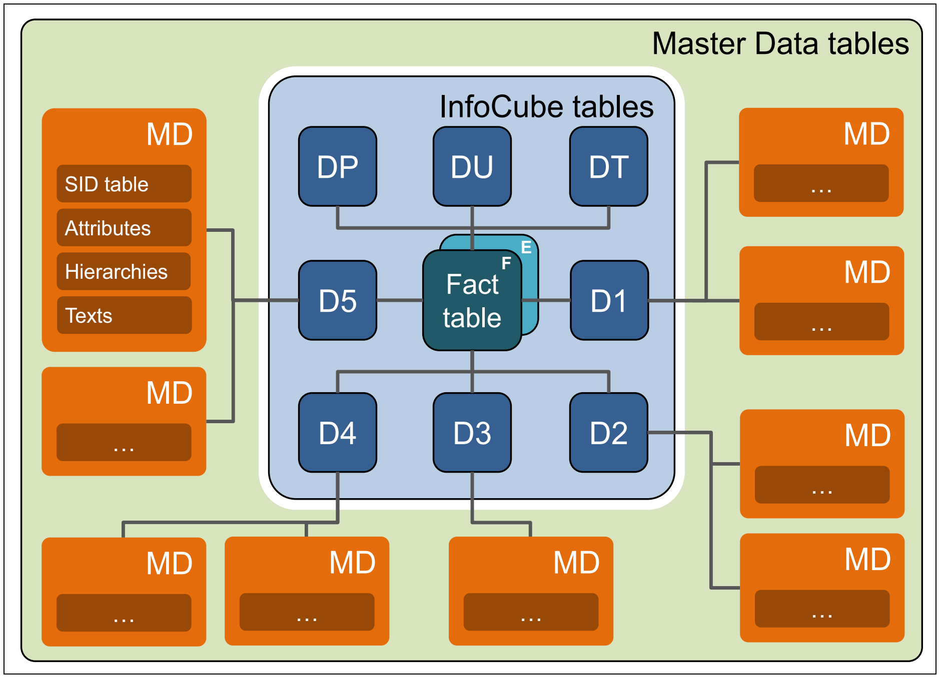



*Figure 8-2 SAP BW extended star schema*

## InfoCube query processing

The SAP BW OLAP processor maps the query to the underlying InfoProvider. It processes reporting queries and translates them into a series of SQL statements that are sent to the database. Depending on the query, the OLAP processor might run additional complex processing to combine the results of the SQL queries into the final result that is sent back to the user.

A typical BW InfoCube query involves joins between the fact, dimension, and master data tables, as shown in Example 8-1.

*Example 8-1 Sample SAP BW Query*

```
-- Projected columns
    SELECT "X1"."S__OINDUSTRY"        AS "S____1272"
         , "DU"."SID_OBASE_UOM"       AS "S____1277"
         , "DU"."SID_OSTAT_CURR"      AS "S____1278"
         , "X1"."S__OCOUNTRY"         AS "S____1270"
         , SUM (  "F"."CRMEM_CST"  )  AS "Z____1279"
         , SUM (  "F"."CRMEM_QTY"  )  AS "Z____1280"
         , SUM (  "F"."CRMEM_VAL"  )  AS "Z____1281"
         , COUNT( * )                 AS "Z____031"

-- F fact table
    FROM "/BIO/FOSD_C01" "F“

-- Dimension tables
    JOIN "/BIO/DOSD_C01U" "DU" ON "F“ ."KEY_OSD_C01U" = "DU"."DIMID“
    JOIN "/BIO/DOSD_C01T" "DT" ON "F“ ."KEY_OSD_C01T" = "DT"."DIMID“
    JOIN "/BIO/DOSD_C01P" "DP" ON "F“ ."KEY_OSD_C01P" = "DP"."DIMID“
    JOIN "/BIO/DOSD_C013" "D3" ON "F“ ."KEY_OSD_C013" = "D3"."DIMID“
    JOIN "/BIO/DOSD_C011" "D1" ON "F“ ."KEY_OSD_C011" = "D1"."DIMID"

-- Master data attribute table
    JOIN "/BIO/XCUSTOMER" "X1" ON "D1"."SID_OSOLD_TO" = "X1"."SID“

-- Predicates
   WHERE ( (      ( ( "DT"."SID_OCALMONTH" IN ( 201001, 201101, 201201,
                                               201301, 201401 )    ) )
             AND  ( ( "DP"."SID_OCHNGID"      = 0                  ) )
             AND  ( ( "D3"."SID_ODISTR_CHAN" IN ( 9, 7, 5, 3 )     ) )
             AND  ( ( "D3"."SID_ODIVISION"   IN ( 3, 5, 7, 9 )     ) )
             AND  ( ( "DP"."SID_ORECORDTP"    =   0                ) )
             AND  ( ( "DP"."SID_OREQUID"     <= 536                ) )
             AND  ( ( "X1"."S__OINDUSTRY"    IN ( 2, 4, 6, 9 )     ) )
         ) )
         AND  "X1"."OBJVERS" = 'A‘
```

```
-- Aggregation
GROUP BY "X1"."S__OINDUSTRY"
       , "DU"."SID_OBASE_UOM"
       , "DU"."SID_OSTAT_CURR"
       , "X1"."S__OCOUNTRY"
;
```

In this case, the fact table, the F fact table `/BIO/FOSD_C01`, is joined to several dimension tables, `/BIO/DOSD_C01x`, and the table of dimension 1, `/BIO/DOSD_C011`, is joined to a master data attribute table, `/BIO/XCUSTOMER`, to extract information about customers in certain industries.

SAP BW query processing might involve SQL queries that generate intermediate temporary results that are stored in database tables. Consecutive SQL queries join these tables to the InfoCube tables. The following examples make use of such tables:

- Evaluation of master data hierarchies
- Storage of a large number of master data SIDs, considered as too large for an IN-list in the SQL query
- Pre-materialization of joins between a dimension table and master data tables when the SAP BW query involves a large number of tables to be joined

### Flat InfoCubes

Flat InfoCubes are similar to standard InfoCubes. However, they use a simplified schema as shown in Figure 8-3. The E fact table and all dimension tables other than the package dimension table are eliminated from the original schema. The SIDs of the characteristics are directly stored in the fact table. For databases other than SAP HANA, flat InfoCubes are only available in SAP BW 7.40 Support Package 8 and higher.

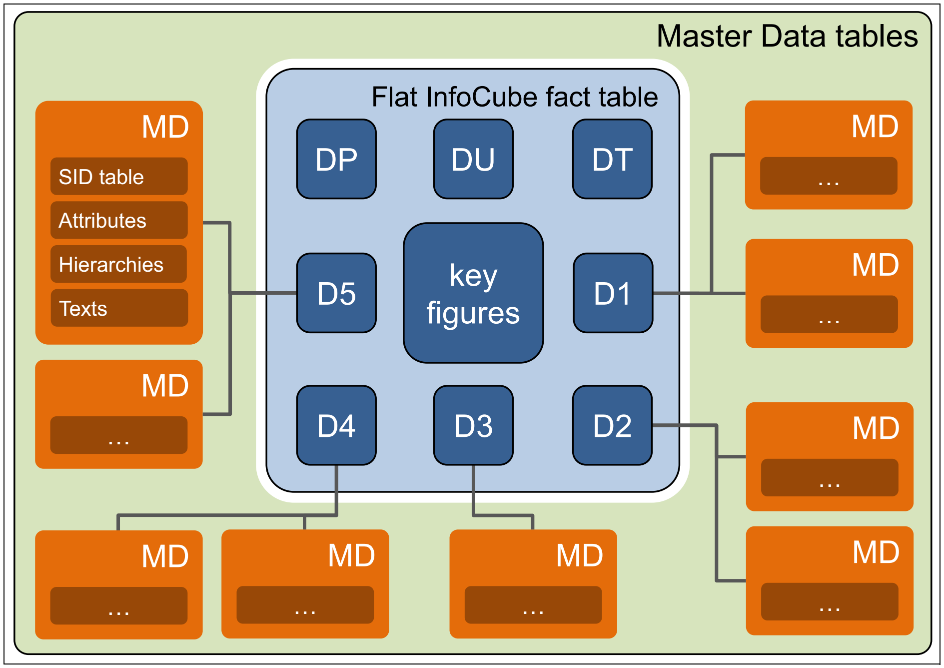



*Figure 8-3 Flat InfoCube*

From the storage point of view, one aspect is that this denormalization increases the redundancy of data in the fact table significantly; the other aspect is that two columns per dimension are dropped which were required for the foreign key relation. Furthermore, the compression algorithm of DB2 with BLU Acceleration can reach high compression rates on redundant data. Thus, in most cases, the impact of the denormalization on the total storage consumption is low.

The number of columns of the fact table is considerably increased by the denormalization. Using BLU Acceleration this does not affect the query performance, because only those columns are read, which are actually required.

You can run InfoCube compression on flat InfoCubes. Compressed and uncompressed data are stored in the same fact table (the F fact table).

Flat InfoCubes have the following advantages:

- Reporting queries might run faster because less tables have to be joined.
- Data propagation into flat InfoCubes might run faster because the dimension tables do not need to be maintained. This is especially true in cases where some dimension tables are large.

- Administration and modelling simplified because line items do not have to be handled separately and the logical assignment of characteristics to dimensions can be changed without any effort.

In SAP BW, the following restrictions apply to flat InfoCubes:

- Flat InfoCubes cannot be stored in the SAP Business Warehouse Accelerator (BWA).
- Aggregates cannot be created for flat InfoCubes.

### Aggregates and SAP Business Warehouse Accelerator

To increase speed of reporting, SAP BW allows the creation of aggregates for non flat InfoCubes. Aggregates contain pre-calculated aggregated data from the InfoCubes. They are comparable to materialized query tables but are completely managed by SAP BW. The SAP BW OLAP processor decides when to route a query to an SAP BW aggregate. The creation and maintenance of aggregates is a complex and time-consuming task that takes a large portion of the extract, transform, and load (ETL) processing time.

SAP offers the SAP Business Warehouse Accelerator (SAP BW Accelerator) for speeding up reporting performance without the need to create aggregates. SAP BW Accelerator runs on a separate hardware and does not support flat InfoCubes. InfoCubes and master data can be replicated into SAP BW Accelerator. Starting with SAP BW 7.30, InfoCubes that reside only in SAP BW Accelerator can be created. When InfoCubes and master data are replicated, the SAP BW Accelerator data must be updated when new data is loaded into SAP BW. This creates ETL processing overhead.

## 8.1.5 BLU Acceleration benefits for SAP BW

Creating column-organized database tables for SAP BW objects provides the following benefits:

- Reporting queries run faster on column-organized tables. Performance improvements vary and depend on the particular query, but results from customer proof of concepts show that a factor greater than ten can be achieved in many cases.
- These performance improvements can be achieved with little or no database tuning effort. Table statistics are collected automatically. Table reorganization, the creation of indexes for specific queries, statistical views, or database hints are usually not needed.

- Because of the much faster query performance, aggregates might not be required any more, or at least the number of aggregates can be reduced significantly. Even the SAP BW Accelerator might become obsolete. This greatly improves the ETL processing time because aggregate rollup and updating the SAP BW Accelerator data is no longer required.
- Except for primary key and unique constraints, column-organized tables have no indexes. Thus, during ETL processing, hardly any time needs to be spent on index maintenance and index logging. This significantly reduces the SAP ETL processing time and the active log space consumption, especially for ETL operations such as InfoCube data propagation and compression. Also data deletion might run much faster, especially when MDC rollout is not applicable on a certain object.
- The compression ratio of column-organized tables is much higher than the ratio achieved with adaptive compression on row-organized tables. In addition, space requirements for indexes are drastically reduced.
- Modelling of InfoCubes is simplified by the use of flat InfoCubes because the logical assignment of characteristics to dimensions can be changed without any effort.

## 8.2 Prerequisites and restrictions for using BLU Acceleration in SAP BW

This section discusses the following topics:

- Important SAP documentation and SAP Notes that you should read
- Prerequisites and restrictions for using DB2 BLU Acceleration in SAP BW
- Required SAP Support Packages and DBSL kernel patches
- Required SAP Kernel parameter settings

### 8.2.1 Important SAP documentation and SAP notes

Before you use BLU Acceleration in SAP BW and SAP NLS on DB2, see the available SAP documentation:

- Database administration guide:

  `http://service.sap.com/instguidesnw`

  Select <***Your SAP NetWeaver Main Release***> → **Operations** → **Database-Specific Guides.**

  The name of the guide is: *SAP Business Warehouse on IBM DB2 for Linux, UNIX, and Windows: Administration Tasks*.

- How-to guide:

  http://service.sap.com/instguidesnw

  Select ***<Your SAP NetWeaver Main Release>*** → **Operations** → **Database-Specific Guides**

  The name of the guide is: *Enabling SAP Business Warehouse Systems to Use IBM DB2 for Linux, UNIX, and Windows as Near-Line Storage (NLS)*.

- Database upgrade guide:

  *Database Upgrade Guide Upgrading to Version 10.5 of IBM DB2 for Linux, UNIX, and Windows*[2]:

  http://service.sap.com/instguides

  **Database Upgrades** → **DB2 UDB** → **Upgrade to Version 10.5 of IBM DB2 for LUW**

SAP Notes[3] with general information about the enablement of DB2 10.5 with BLU Acceleration are listed in Table 8-1.

*Table 8-1 SAP Notes regarding the enablement of BLU Acceleration in SAP Applications*

| SAP Note | Description |
|---|---|
| 1555903 | Gives an overview on the supported DB2 features. |
| 1851853 | Describes the SAP software prerequisites for using DB2 10.5 in general for SAP applications. The note also provides an overview of the new DB2 10.5 features that are relevant for SAP, including BLU Acceleration. |
| 1851832 | Contains a section about DB2 parameter settings for BLU Acceleration. |
| 1819734 | Describes prerequisites and restrictions for using BLU Acceleration in the SAP environment (includes corrections). |

The SAP Notes in Table 8-2 contain corrections and advises that you must install to enable BLU Acceleration for the supported SAP BW based applications. You can install these SAP Notes using SAP transaction SNOTE.

2 See http://service.sap.com/~sapidb/011000358700000843422013E
3 See http://service.sap.com/notes (authentication required)

*Table 8-2 SAP Notes regarding BLU Acceleration of SAP BW based Applications*

| SAP Note | Description |
|---|---|
| 1889656 | Contains mandatory fixes you must apply before you use BLU Acceleration in SAP BW. With the Support Packages installed, which are listed in 8.2.2, "Prerequisites and restrictions for using DB2 BLU Acceleration" on page 258, this is only necessary for SAP BW 7.0 Support Package 32. |
| 1825340 | Describes prerequisites and the procedure for enabling BLU Acceleration in SAP BW. |
| 2034090 | Contains recommendations and best practices for SAP BW objects with BLU Acceleration created on DB2 10.5 FP3aSAP or earlier. |
| 1911087 | Contains an advise regarding a DBSL patch. |
| 1834310 | Describes how to enable BLU Acceleration for SAP's near-line storage solution on DB2 (includes corrections). |
| 1957946 | Describes potential issues with column-organized InfoCubes. This note is only relevant if you use DB2 10.5 FP3aSAP. If you often make structural changes to InfoCubes, for example, by adding key figures or by adding or changing dimensions, you should upgrade to DB2 Cancun Release 10.5.0.4. |

The list of SAP Notes listed in Table 8-3 contain further improvements and code fixes. They should be installed as well to achieve the best functionality.

*Table 8-3 SAP Notes with further improvements and fixes*

| SAP Note | Description |
|---|---|
| 1964464 | Contains specific fixes for the conversion report from row-organized to column-organized tables. This is needed if you want to convert existing InfoCubes to BLU on DB2 10.5 FP3aSAP or earlier. |
| 1996587<br>2022487 | Contain code fixes for the InfoCube compression of the accelerated InfoCubes with BLU Acceleration. |
| 1979563 | Contains a fix for eliminating mistakenly created warning messages in the SAP work process trace files. |
| 2019648 | Handles issues that might occur when you transport a BLU InfoCube to a system that does not use automatic storage or does not use reclaimable storage tablespaces. |
| 2020190 | Contains corrections for BLU InfoCubes with dimensions that contain more than 15 characteristics |

Starting with DB2 Cancun Release 10.5.0.4, several additional features have been added, which can be enabled by applying the SAP Notes in Table 8-4.

*Table 8-4 SAP Notes for DB2 Cancun Release 10.5.0.4 specific features*

| SAP Note | Description |
|---|---|
| 1997314 | Required to use BLU Acceleration for DataStore Objects, InfoObjects, and PSA tables. This is only supported as of DB2 Cancun Release 10.5.0.4. |
| 2038632 | Required for using flat InfoCubes in SAP BW 7.40 Support Package 8 or higher. This is only supported as of DB2 Cancun Release 10.5.0.4. |
| 1947559<br>1969500 | These notes are required for SAP system migrations. They enable the generation of BW object tables directly as BLU tables without the need to run conversions to BLU in the migration target system later. This is only supported when your DB2 target system is set up on DB2 Cancun Release 10.5.0.4 or higher. |

## 8.2.2 Prerequisites and restrictions for using DB2 BLU Acceleration

BLU Acceleration is currently only supported in SAP BW, SAP Strategic Enterprise Management (SAP SEM), and SAP NLS on DB2. BLU Acceleration is not supported in other SAP applications, including applications that are based on SAP BW, such as SAP Supply Chain Management (SAP SCM). The latest list of supported products can be found in *SAP Note 1819734*.

The following additional restrictions for your DB2 database apply:

- You need at least DB2 10.5 Fix Pack 3aSAP. The preferred version is DB2 Cancun Release 10.5.0.4 or later, which provides additional significant optimizations and performance improvements for SAP and enables the BLU implementation for DSOs, InfoObjects, and PSA tables.
- BLU DSOs, InfoObjects, and PSA tables are only supported as of DB2 Cancun Release 10.5.0.4 (see *SAP Note 1997314*). All InfoObject types except for key figures (for which no database tables are created) are supported.
- Flat InfoCubes are only supported as of DB2 Cancun Release 10.5.0.4 and SAP BW 7.40 Support Package 8.
- Your database server runs on platform that supports BLU Acceleration. At the time of writing these are AIX or Linux on the X86_64 platform.
- Your database uses Unicode.
- You do not use DB2 pureScale and do not use the DB2 Database Partitioning Feature (DPF).

- If you use DB2 10.5 Fix Pack 3aSAP you do not use HADR. This restriction does not exist any more in DB2 Cancun Release 10.5.0.4 release.
- Your database is set up with DB2 automatic storage.
- You provide reclaimable storage tablespaces for the BLU tables.

### 8.2.3 Required Support Packages and SAP DBSL Patches

SAP BW 7.0 and later are supported. The lowest preferred Support Packages for each SAP BW release are shown in Table 8-5. With DB2 Cancun Release 10.5.0.4 and these Support Packages and additional SAP notes that have to be installed on top, BLU Acceleration for InfoCubes, DataStore objects, InfoObjects, PSA tables, and BW temporary tables is available. Furthermore, it is possible to directly create column-organized tables in SAP BW system migrations to DB2 10.5.

*Table 8-5 Supported SAP BW releases and Support Packages*

| SAP NetWeaver release | SAP BASIS Support Package | SAP BW Support Package |
|---|---|---|
| 7.0 | 30 | 32 |
| 7.01 | 15 | 15 |
| 7.02 | 15 | 15 |
| 7.11 | 13 | 13 |
| 7.30 | 11 | 11 |
| 7.31/7.03 | 11 | 11 |
| 7.40 | 6 | 6 |

These support packages contain ABAP code that is specific to DB2 in the SAP Basis and SAP BW components that enable the use of BLU Acceleration and extensions to the DBA Cockpit for monitoring and administering SAP DB2 databases. They are prerequisites for using BLU Acceleration in SAP BW. For more details, see *SAP Notes 1819734* and *1825340*.

We suggest that you also install the latest available SAP kernel patches and SAP Database Shared Library (DBSL). The minimum required DBSL patch level for using DB2 Cancun Release 10.5.0.4 with BLU Acceleration in SAP BW and SAP's near-line storage solution on DB2 can be found in *SAP Note 2056603*. These patch levels are higher than the levels listed in the SAP database upgrade guide: *Upgrading to Version 10.5 of IBM DB2 for Linux, UNIX, and Windows*.

### 8.2.4 Required SAP kernel parameter settings

As of DB2 Cancun Release 10.5.0.4 and SAP BW support InfoObjects, the following SAP kernel parameters are required:

- *dbs/db6/dbsl_ur_to_cc=**

  This parameter sets the default DB2 isolation level that is used in the SAP system to Currently Committed. This applies to all SQL statements that otherwise run with isolation level UR (uncommitted read) by default. This parameter is needed to avoid potential data inconsistencies during the update of existing master data and insertion of new master data in the SAP BW system.

- *dbs/db6/deny_cde_field_extension=1*

  This parameter prevents operations in the SAP BW system that extend the length of a VARCHAR column in a column-organized table. Extending the length of a VARCHAR column in a column-organized table is not supported in DB2 10.5 and causes SQL error SQL1667N. (`The operation failed because the operation is not supported with the type of the specified table. Specified table: <tabschema>.<tabname>. Table type: “ORGANIZE BY COLUMN”. Operation: “ALTER TABLE”. SQLSTATE=42858`). When the SAP profile parameter is set, the SAP ABAP Dictionary triggers a table conversion if extending the length of a VARCHAR column in a column-organized table is requested.

# 8.3 BLU Acceleration support in the ABAP Dictionary

In the ABAP Dictionary, storage parameters are used to identify column-organized tables. The property, including the column-organized table, can be retrieved from the database and displayed in the “Storage Parameter” window of the SAP data dictionary.

In the example in Figure 8-4, the `/BIC/FICBLU01` table is a column-organized table. Use the following steps to look up the storage parameters of the transaction table SE11:

1. Enter the table name in the Database table field and select **Display**.

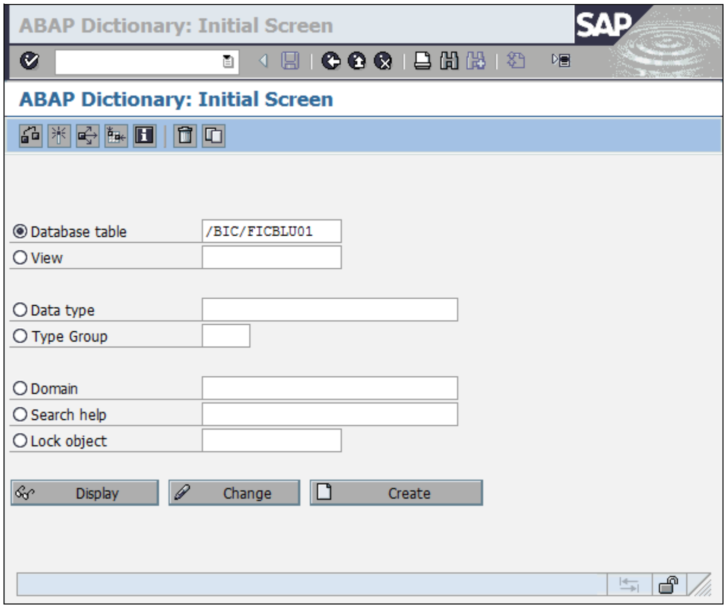



*Figure 8-4 ABAP Dictionary initial window*

2. In the "Dictionary: Display" table window, select **Utilities** → **Database Object** → **Database Utility** from the menu (Figure 8-5).

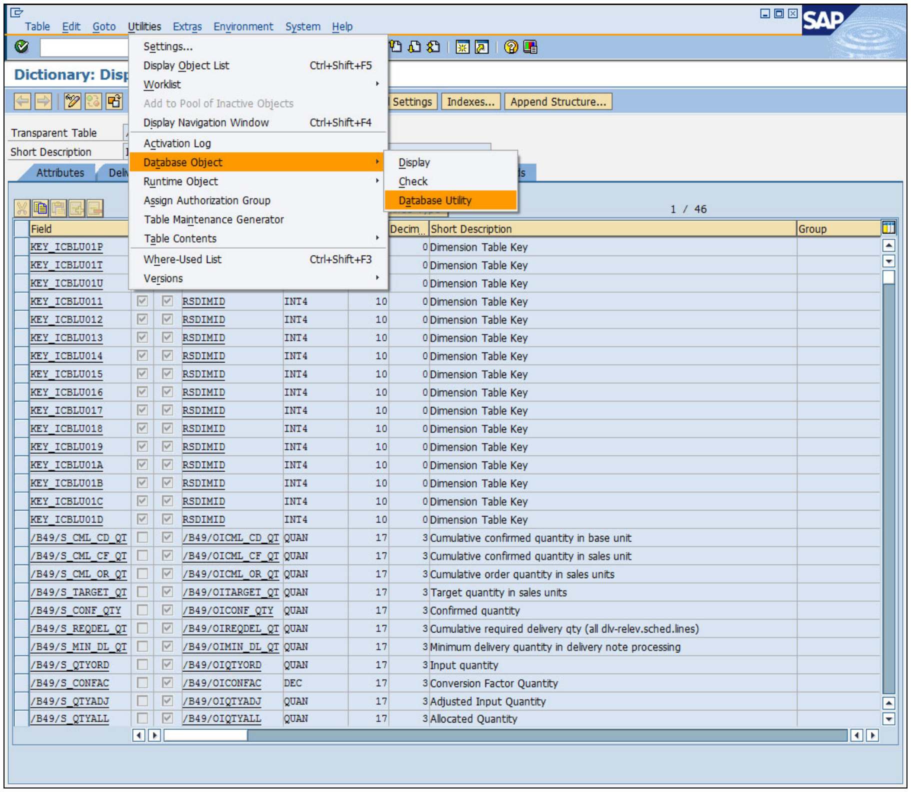


*Figure 8-5 ABAP Dictionary: retrieval of table properties*

3. In the “ABAP Dictionary: Utility for Database Tables” window, select **Storage Parameters**, as shown in Figure 8-6.

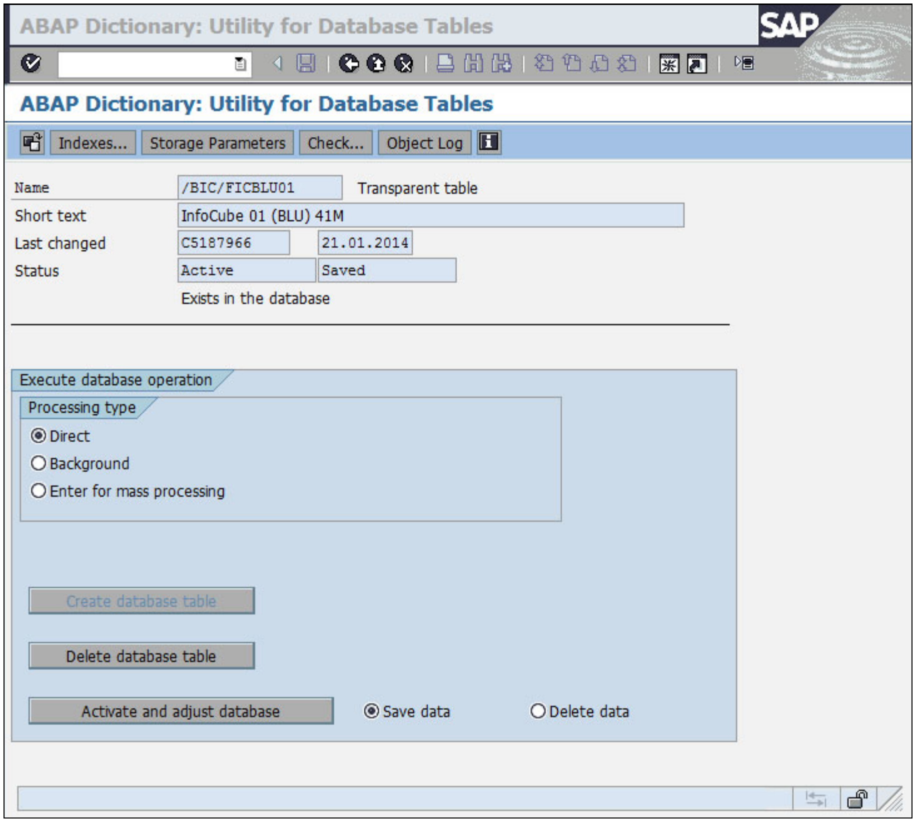



*Figure 8-6 ABAP Dictionary: retrieval of table database storage parameters*

The “Storage parameters: (display and maintain)” window (Figure 8-7) shows the table properties that are retrieved from the DB2 system catalog. The last entry with the `OPTIONS` label is `ORGANIZE BY COLUMN`, which shows that the table is organized column-oriented.

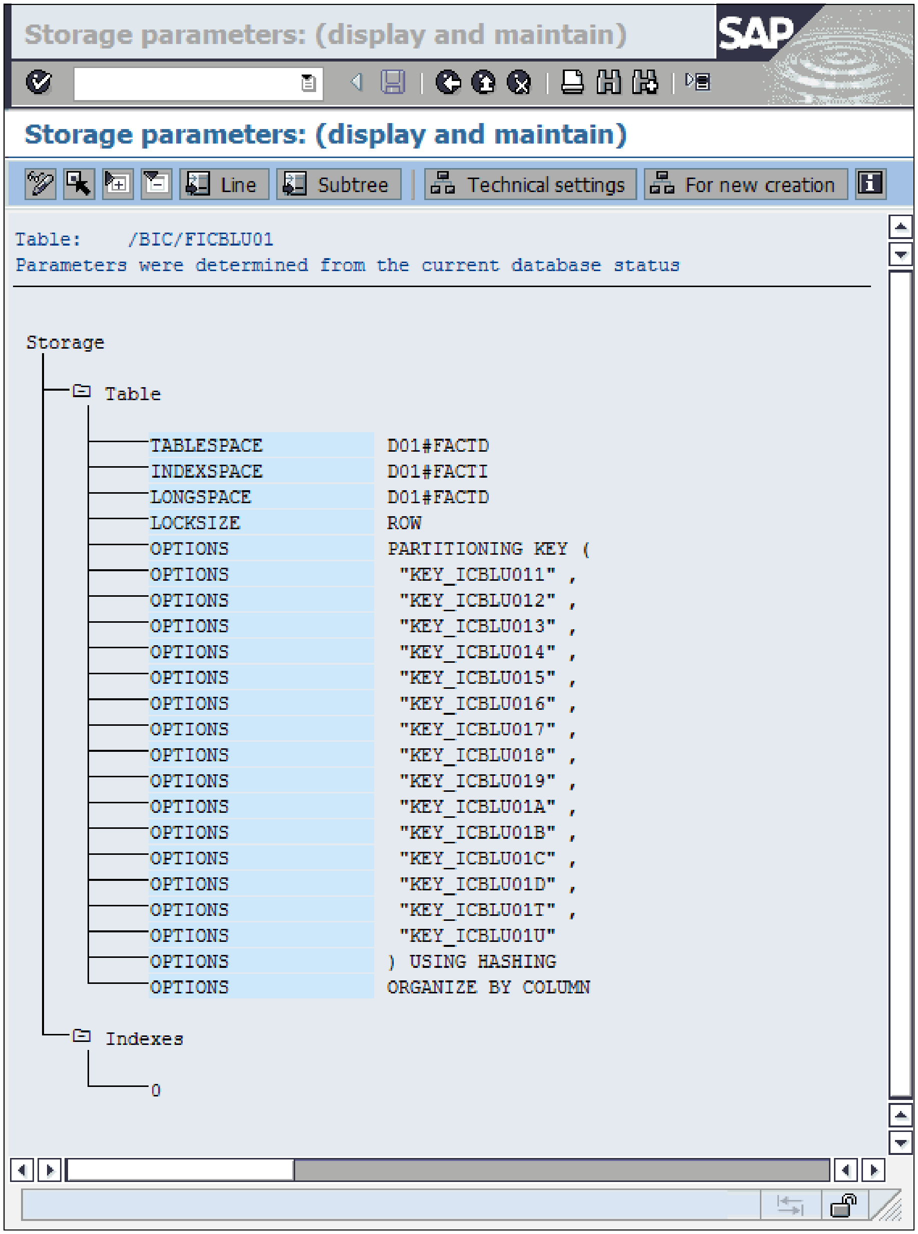



*Figure 8-7 Table storage parameters of a column-organized SAP BW F fact table*

**Notes:**

- Although BLU Acceleration does not require DPF for parallel query processing, DB2 accepts the specification of a distribution key. In SAP BW, distribution keys are always created for InfoCube fact, DSO, and PSA tables.

  SAP uses the term *PARTITIONING KEY* for distribution keys in the ABAP Dictionary.
- SAP NetWeaver 7.40 introduces an option in the ABAP Dictionary to specify whether a table should be created in Column Store or in Row Store. This option does not trigger creation of column-organized tables in DB2.

## 8.4 BLU Acceleration support in the DBA Cockpit

The DBA Cockpit has been enhanced with the DB2 monitoring elements for column-organized tables.

The DBA Cockpit extensions for BLU Acceleration are available in the SAP NetWeaver releases and Support Packages listed in the second column of Table 8-6. With these Support Packages, time spent analysis, the BLU Acceleration buffer pool, and I/O metrics are only available in the Web Dynpro user interface. When you upgrade to the Support Packages listed in column three, these metrics are included in the SAP GUI user interface.

*Table 8-6 Availability of DBA Cockpit enhancements for BLU Acceleration in the SAP Web Dynpro and SAP GUI user interfaces*

| SAP NetWeaver release | SAP BASIS Support Package for BLU Acceleration metrics in Web Dynpro | SAP BASIS Support Package for BLU Acceleration metrics in SAP GUI |
|---|---|---|
| 7.02 | 14 | 15 |
| 7.30 | 10 | 11 |
| 7.31 | 9 | 11 |
| 7.40 | 4 | 6 |

We suggest that you also implement *SAP Note 1456402*. This SAP Note contains a patch collection for the DBA Cockpit that is updated regularly. You can install the most recent DBA Cockpit patches by reinstalling the patch collection SAP Note. When you reinstall the SAP Note, only the delta between your current state and the most recently available patches is installed.

When you monitor your SAP systems using SAP Solution Manager, you need SAP Solution Manager 7.1 Support Package 11 to get the BLU Acceleration metrics.

For SAP BW 7.0 and SAP BW 7.01, a compatibility patch for DB2 10.5 is available in the Support Packages that are required for using BLU Acceleration in SAP BW. This patch handles only changes in the DB2 monitoring interfaces that are used in the DBA Cockpit. It does not provide any extra metrics for BLU Acceleration in the DBA Cockpit of SAP NetWeaver 7.0 and 7.01.

With the DBA Cockpit enhancements for BLU Acceleration, you can obtain the following information:

- Individual table information, including whether the table is column-organized
- Database information, including the databases that contain column-organized tables
- Columnar data processing information as part of time spent analysis
- Buffer pool hit ratio, buffer pool read and write activity, and prefetcher activity for the whole database, buffer pools, table spaces, and applications

In the following sections, we provide examples of BLU Acceleration monitoring using the DBA Cockpit. The windows in the figures are from an SAP BW 7.40 system running on Support Package stack 5.

## 8.4.1 Checking whether individual tables in SAP database are column-organized

In the DBACOCKPIT transaction, choose **Space** → **Single Table Analysis** in the navigation frame (Figure 8-8). Enter the table name in the Name field of the Space: Table and Indexes Details window and press Enter. You find the table organization in the Table Organization field on the System Catalog tab.

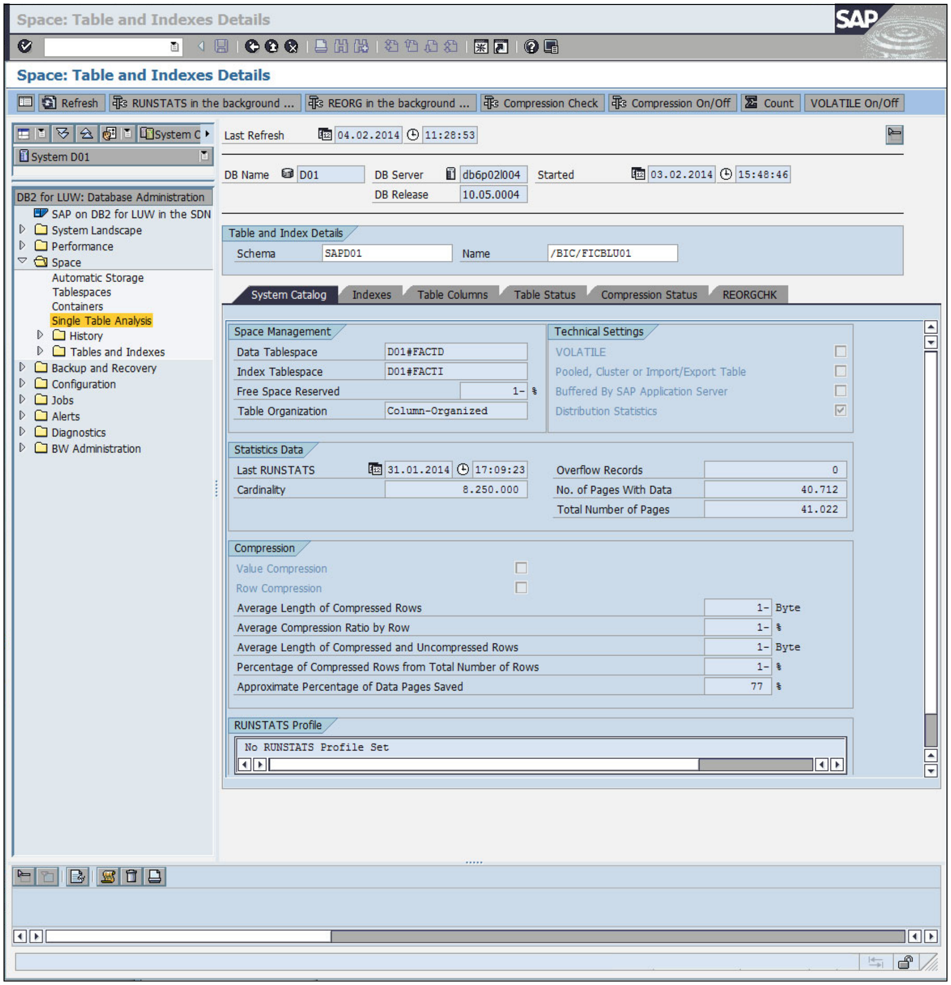



*Figure 8-8 DBA Cockpit: properties of a single table*

## 8.4.2 Checking if SAP database contains column-organized tables

In the DBACOCKPIT transaction, choose **Configuration** → **Parameter Check** in the navigation frame (Figure 8-9). On the Check Environment tab in the Parameter Check window, find the System Characteristics panel. If the CDE attribute has the value YES, the database contains column-organized tables.

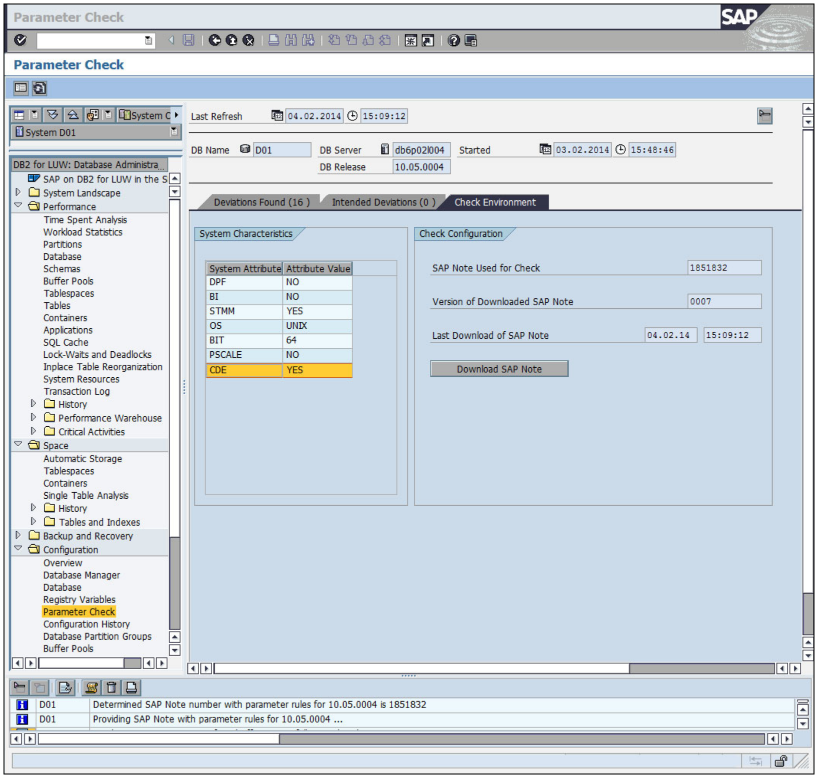



*Figure 8-9 DBA Cockpit: DB2 configuration parameter check window*

## 8.4.3 Monitoring columnar data processing time in the SAP database

Time spent analysis was originally only available in the Web Dynpro user interface. With the service packs listed Table 8-6 on page 265 it is available in SAP GUI as well. The configuration of the Web Dynpro user interface is described in *SAP Note 1245200*.

Choose **Performance** → **Time Spent Analysis** in the navigation frame of the DBA Cockpit. A window similar to the one shown in Figure 8-10 opens. Enter the time frame for which you want to retrieve the information. The window is updated and shows how much time DB2 spent in various activities, including the columnar processing time in the selected time frame.

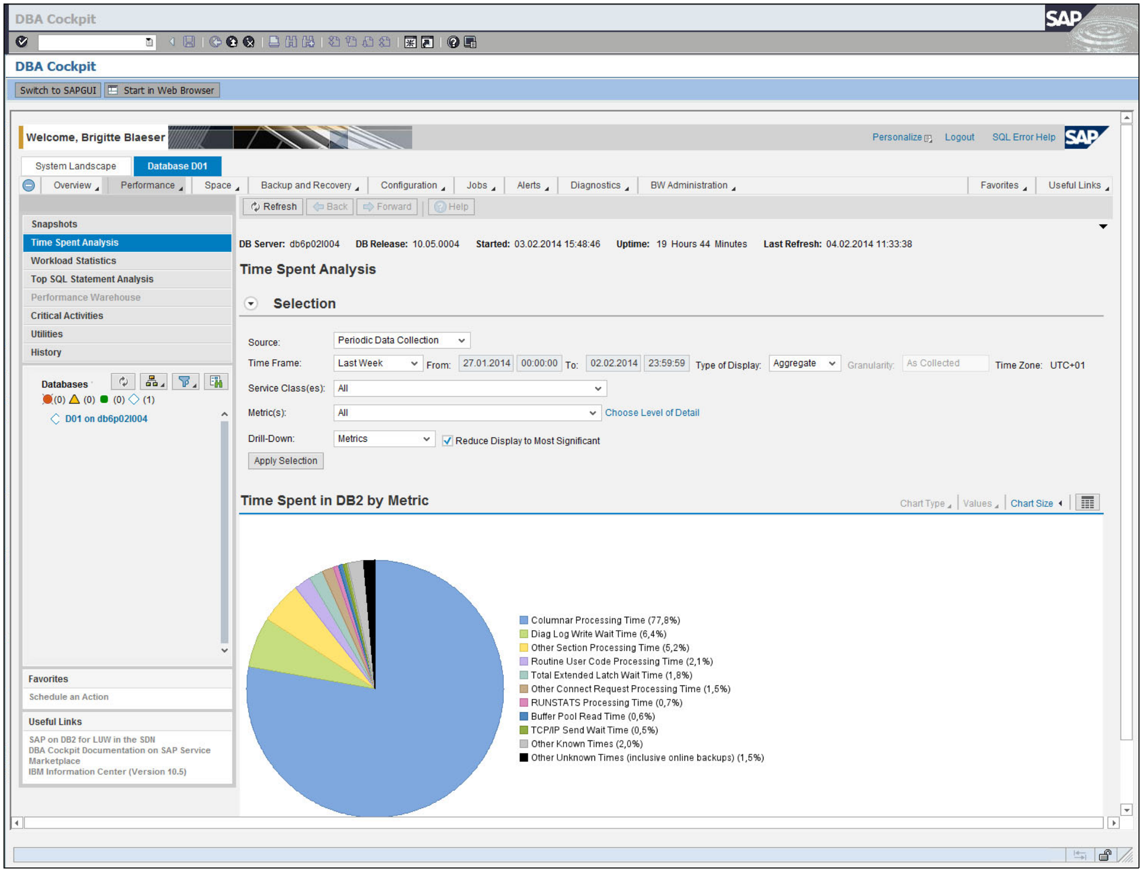



*Figure 8-10 DBA Cockpit: Analysis of time spent in DB2*

### 8.4.4 Monitoring columnar processing-related prefetcher and buffer pool activity in the SAP database

You can monitor buffer pool and prefetcher activity at database, workload, table space, and application level.

Choose **Performance** → **(Snapshots** → **)**[4] **Database** in the DBA Cockpit. Select the time frame for which you want to display the information. On the "Buffer Pool" tab in the "History Details" area (Figure 8-11), you see the logical and physical reads, the physical writes, synchronous reads and writes, and temporary logical and physical reads on column-organized tables in the "Columnar" section. In the "Buffer Quality" section, you see the buffer pool hit ratio for column-organized tables.

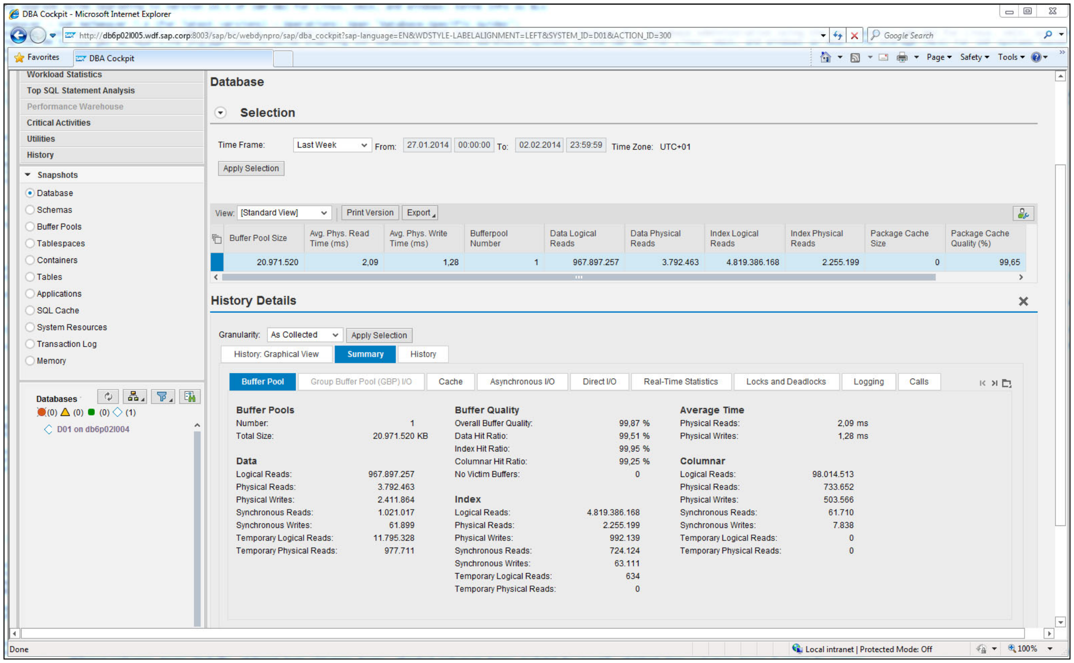


*Figure 8-11 DBA Cockpit: Buffer pool metrics*

4 In newer service packs, the menu has been rearranged.

As shown in Figure 8-12, when you switch to the “Asynchronous I/O” tab, you find separate counters for column-organized tables in these sections:

- Prefetcher I/O
- Total No. of Skipped Prefetch Pages
- Skipped Prefetch Pages Loaded by Agents in UOW sections.

For example, the “Prefetcher I/O” section shows “Columnar Prefetch Requests”, which is the number of column-organized prefetch requests successfully added to the prefetch queue. This corresponds to the pool_queued_async_col_reqs DB2 monitoring element.

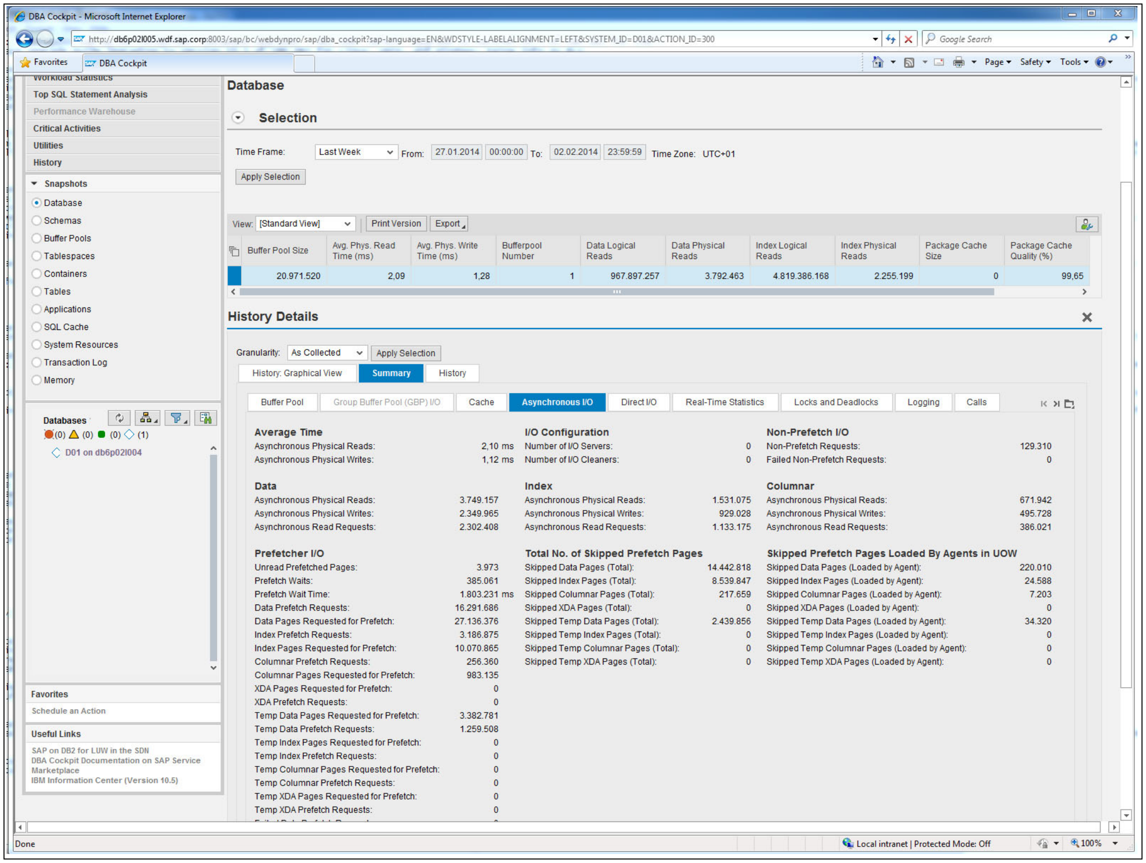


*Figure 8-12 DBA Cockpit: Asynchronous I/O metrics*

To retrieve the buffer pool and prefetcher information for individual table spaces, buffer pools, and for the current unit of work of an application, choose **Performance** → **(Snapshots** → **) Tablespaces**, **Bufferpools**, or **Applications** in the DBA Cockpit navigation frame.

## 8.5 BLU Acceleration support in SAP BW

You can do the following operations in SAP BW:

- In the SAP Data Warehousing Workbench, you can create InfoCubes, DSOs, InfoObjects, and PSA tables as column-organized tables in the database. The following list indicates the supported BW object types:
  - Basic cumulative InfoCubes
  - Basic non-cumulative InfoCubes
  - Multi-Cubes with underlying column-organized basic InfoCubes
  - Multi-Provider which consolidate several InfoProvider
  - In SAP BW 7.30 and higher, semantically partitioned InfoCubes
  - As of DB2 Cancun Release 10.5.0.4 BLU Acceleration is also supported for the following BW objects:
    - Standard DSOs
    - Write-optimized DSOs
    - InfoObjects (characteristics, time characteristics, units), through a new RSADMIN parameter
    - PSA tables, through a new RSADMIN parameter

  You cannot create column-organized real-time InfoCubes and transactional DSOs.
- As of SAP BW 7.40 Support Package 8 and DB2 Cancun Release 10.5.0.4, you can work with flat InfoCubes. Flat InfoCubes have a simplified star schema with only one fact table and no dimension tables except for the package dimension table. You can create flat InfoCubes and you can convert existing standard InfoCubes to flat InfoCubes (see 8.5.2, “Flat InfoCubes in SAP BW 7.40” on page 297).
- DB6CONV is enhanced with the option to convert BW objects (InfoCubes, DSOs, InfoObjects, and PSA tables) from row-organized to column-organized tables. You can convert selected BW objects or all BW objects in your system (see 8.6, “Conversion of SAP BW objects to column-organized tables” on page 350). Thus, the SAP_CDE_CONVERSION report is discontinued with DB2 Cancun Release 10.5.0.4.
- You can set an SAP BW RSADMIN configuration parameter that causes new InfoObject SID and attribute SID tables to be created as column-organized tables.
- You can set an SAP BW RSADMIN configuration parameter that causes new PSA tables to be created as column-organized tables.

- You can set an SAP BW RSADMIN configuration parameter that causes new SAP BW temporary tables to be created as column-organized tables.

In this section, we explain in detail how to perform these operations.

## 8.5.1 Column-organized standard InfoCubes in SAP BW

This section shows how to create a new standard InfoCube that uses column-organized tables in the Data Warehousing Workbench of SAP BW. We illustrate the procedure with an example in an SAP BW 7.40 system in which the BI DataMart Benchmark InfoCubes have been installed (InfoArea BI Benchmark). The example works in exactly the same way in the other SAP BW releases in which BLU Acceleration is supported.

Follow these steps to create a new InfoCube that uses column-organized tables in the Data Warehousing Workbench of SAP BW:

1. Call transaction RSA1 to get to the Data Warehousing Workbench and create the InfoArea `BLU Acceleration Example InfoArea` (Figure 8-13).

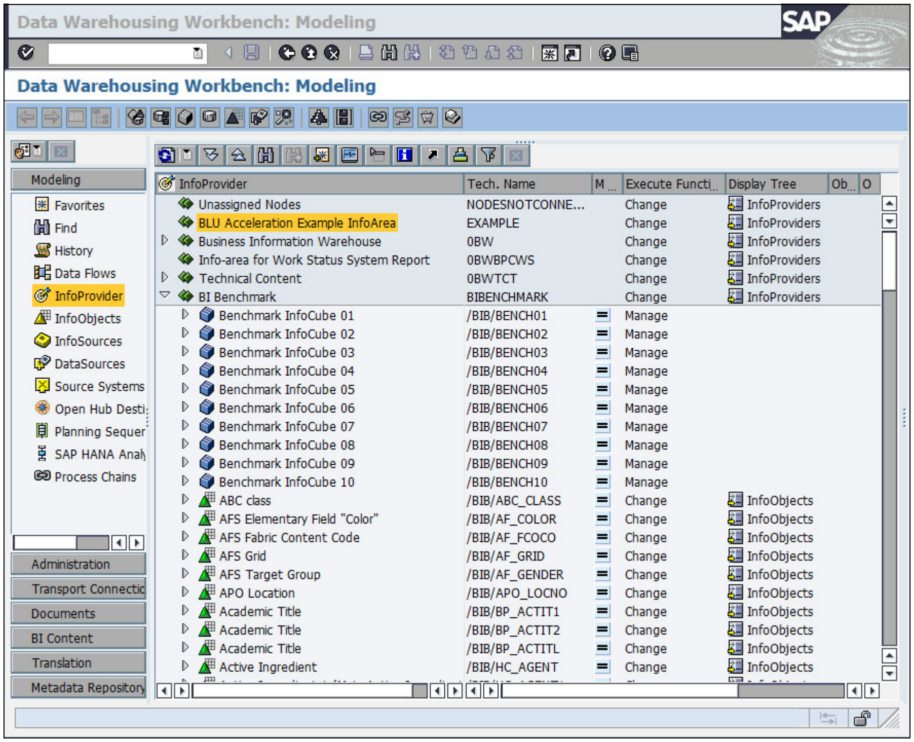


*Figure 8-13 SAP Data Warehousing Workbench main window*

2. In BI Benchmark, select **Benchmark InfoCube 01** (technical name is `/BIB/BENCH01`), and choose **Copy** from the context menu (Figure 8-14).

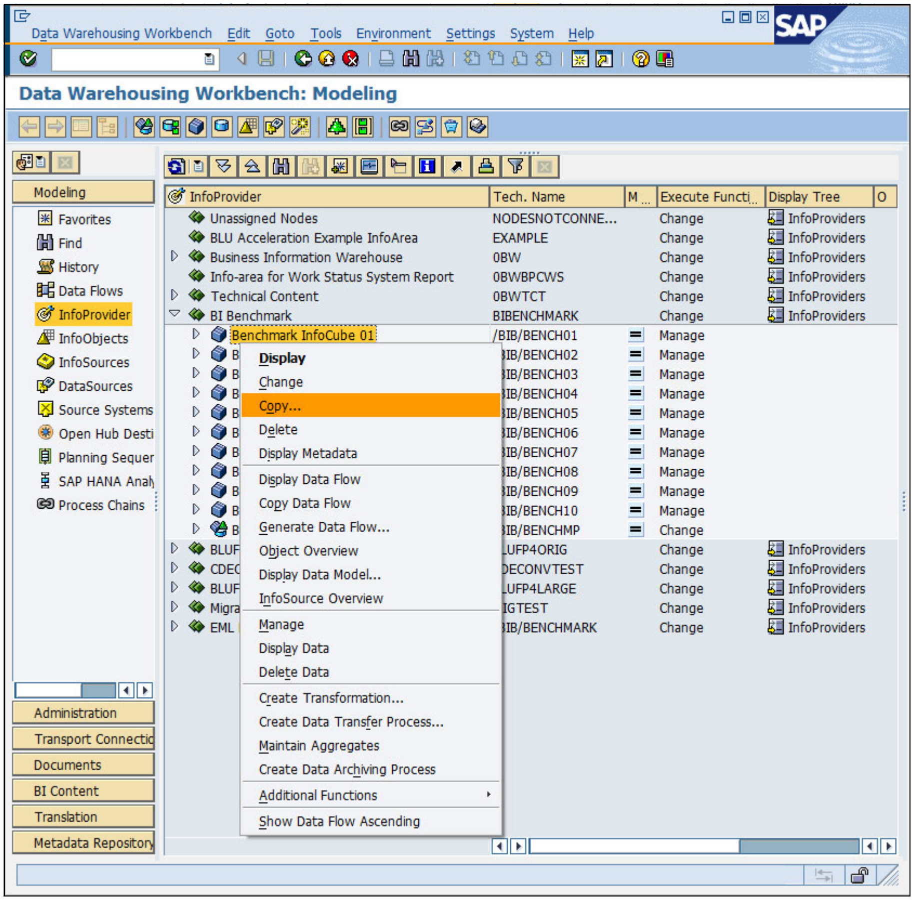



*Figure 8-14 SAP Data Warehousing Workbench: Copying an InfoCube*

3. In the Edit InfoCube window, enter `BLUCUBE1` as the technical name, `BLU Demo InfoCube 1` as the description, and select **EXAMPLE (BLU Acceleration Example InfoArea)** as the InfoArea. Click the **Create** icon (Figure 8-15).

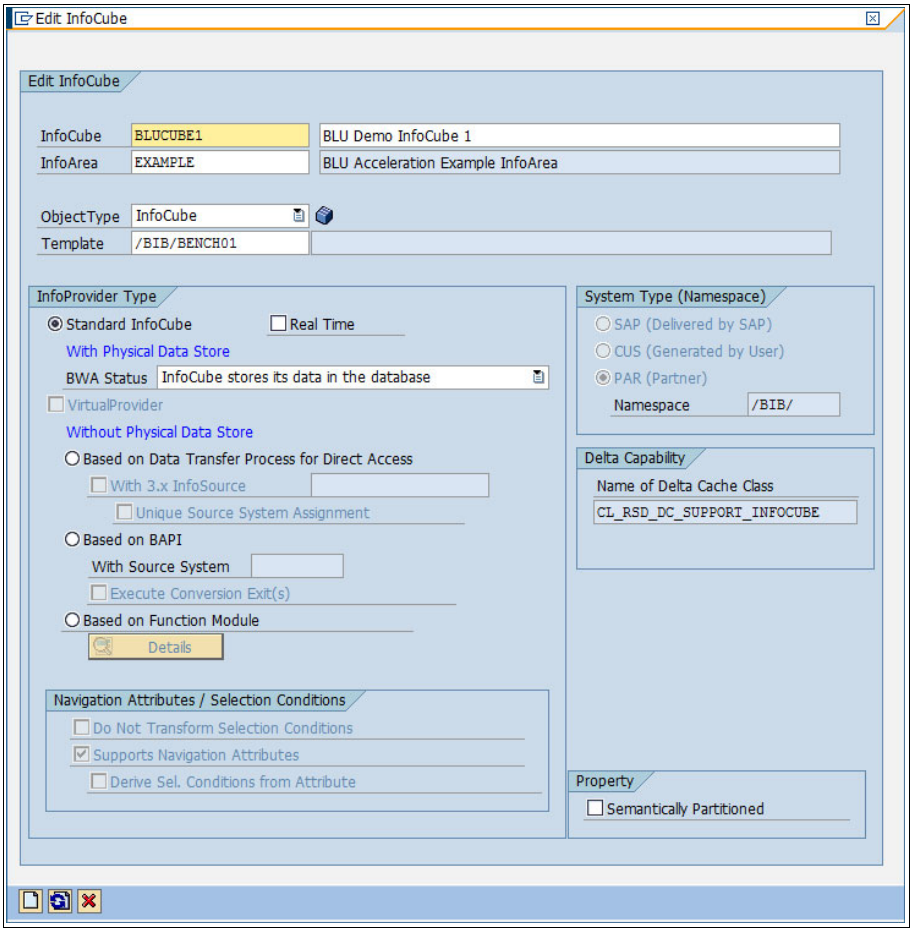



*Figure 8-15 SAP Data Warehousing Workbench: Editing InfoCube window*

4. The Edit InfoCube window in the Data Warehousing Workbench opens. From the menu, choose **Extras** → **DB Performance** → **Clustering** (Figure 8-16).

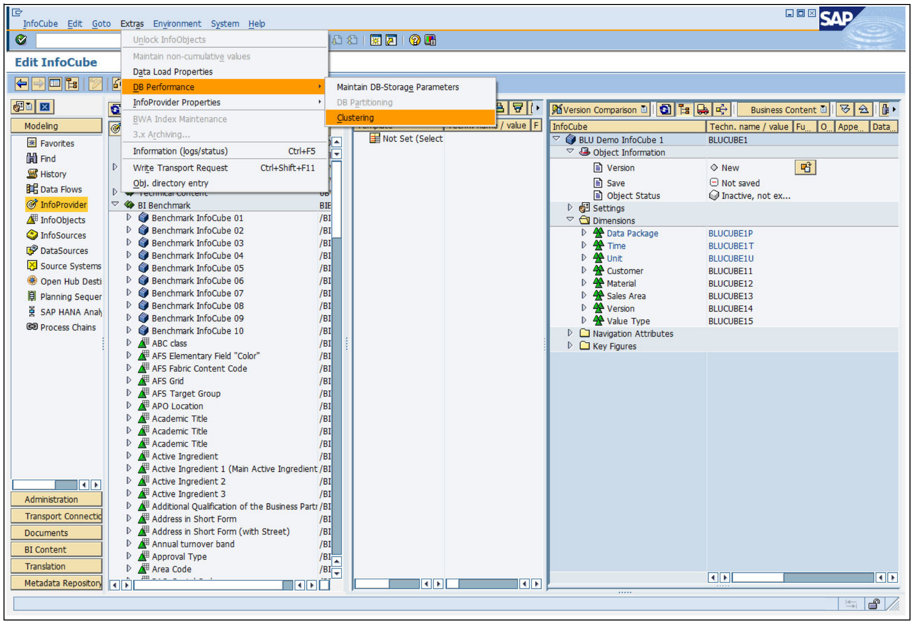



*Figure 8-16 SAP Data Warehousing Workbench: Specifying DB2-specific properties*

5. In the Selection of Clustering dialog window, select **Column-Organized** and then click the check mark icon to continue (Figure 8-17).

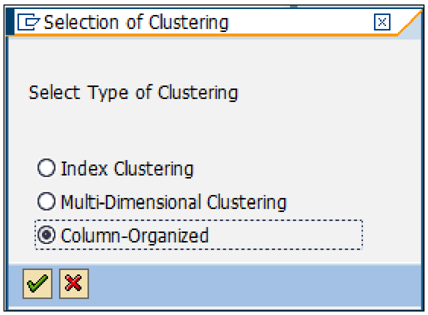



*Figure 8-17 SAP Data Warehousing Workbench: Select column-organized tables*

6. Save and activate the InfoCube.

7. Inspect the tables:

   In transaction RSA1, you can inspect the tables of an InfoCube by choosing **Extras** → **Information (logs/status)** from the menu (Figure 8-18).

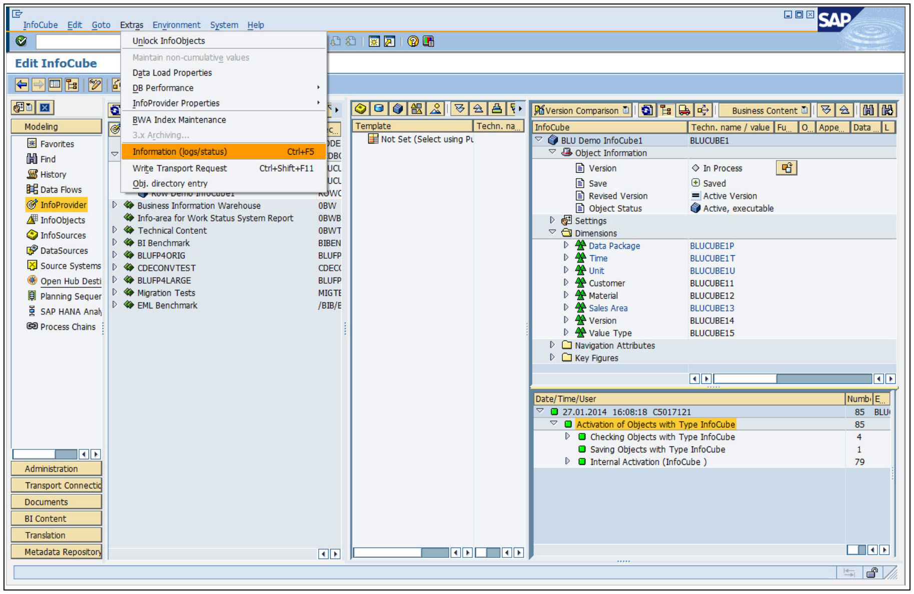



*Figure 8-18 SAP Data Warehousing Workbench: Accessing protocol and status information*

   In the Info Selection dialog window, click **Dictionary/DB Status** (Figure 8-19).

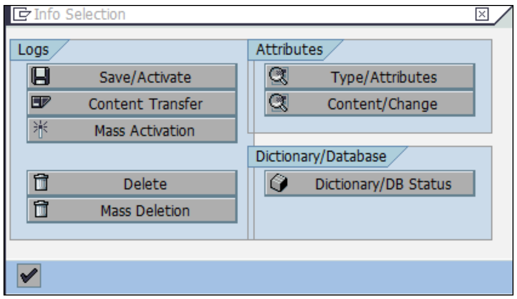



*Figure 8-19 Accessing ABAP Dictionary for the tables of an InfoCube*

The “Status Information” window displays the tables of the InfoCube. To look up the storage parameters for the table, double-click a table to get to the ABAP Dictionary information for the table (Figure 8-20).

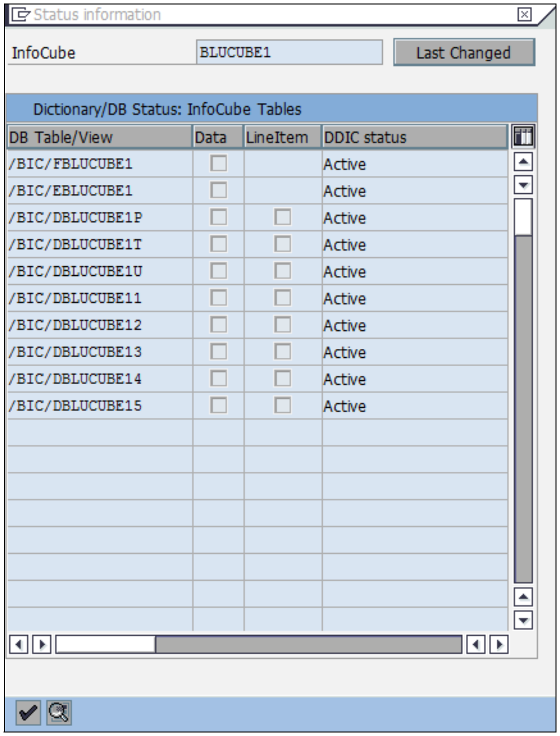



*Figure 8-20 ABAP Dictionary information for a InfoCube table*

When you work with BLU Acceleration InfoCubes, consider the following information:

- When you copy an existing InfoCube, the target InfoCube inherits the clustering and table organization settings of the source InfoCube. You can change the settings of the target InfoCube in the “Selection of Clustering” window in the Data Warehousing Workbench before you activate the target InfoCube.
- You might no longer need aggregates when you work with BLU Acceleration InfoCubes because the query performance on the basic InfoCubes is fast enough. However, you can create aggregates for BLU Acceleration InfoCubes just as for row-organized InfoCubes if needed. The tables for any aggregates of BLU Acceleration InfoCubes are also created as column-organized tables.

- When you install InfoCubes from SAP BI Content, they are created with the default clustering settings for DB2, which is index clustering. If you want to create BLU Acceleration InfoCubes, you must change the clustering settings in the Data Warehousing Workbench after you install the InfoCubes, and then reactivate the InfoCubes.
- When you transport an InfoCube from your development system into the production system, the behavior is as follows:
  - If the InfoCube already exists in the production system and contains data, the current clustering settings and table organization are preserved.
  - If the InfoCube does not yet exist in the production system or does not contain any data, the clustering and table organization settings from the development system are used. If the tables exist, they are dropped and re-created.
- When you create a semantically partitioned InfoCube, the clustering settings that you select for the semantically partitioned InfoCube are propagated to the partitions of the InfoCube. Therefore, when you choose to create column-organized tables, the tables of all partitions of the semantically partitioned InfoCube are created automatically as column-organized.

## Table layout of column-organized InfoCubes

This section shows the differences in the layout of row-organized and column-organized InfoCube tables.

For comparison, we also create `ROWCUBE1`, a row-organized InfoCube. Here we use multidimensional clustering (MDC) on the package dimension key column of the InfoCube F fact table and on the time characteristic calendar month on the InfoCube F and E fact tables:

1. You select these clustering settings for the row-organized InfoCube in SAP BW by choosing **Multi-Dimensional Clustering** in the “Selection of Clustering” dialog window (Figure 8-21).

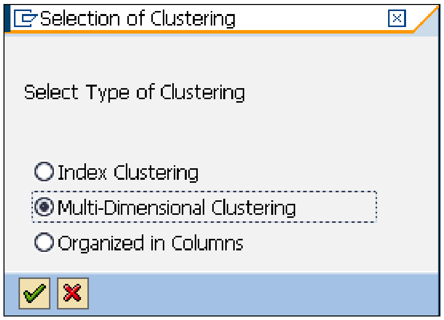



*Figure 8-21 SAP Data Warehousing Workbench: Creating a sample row-organized InfoCube*

2. Then make selections in the “Multi-Dimensional Clustering” dialog window (Figure 8-22).

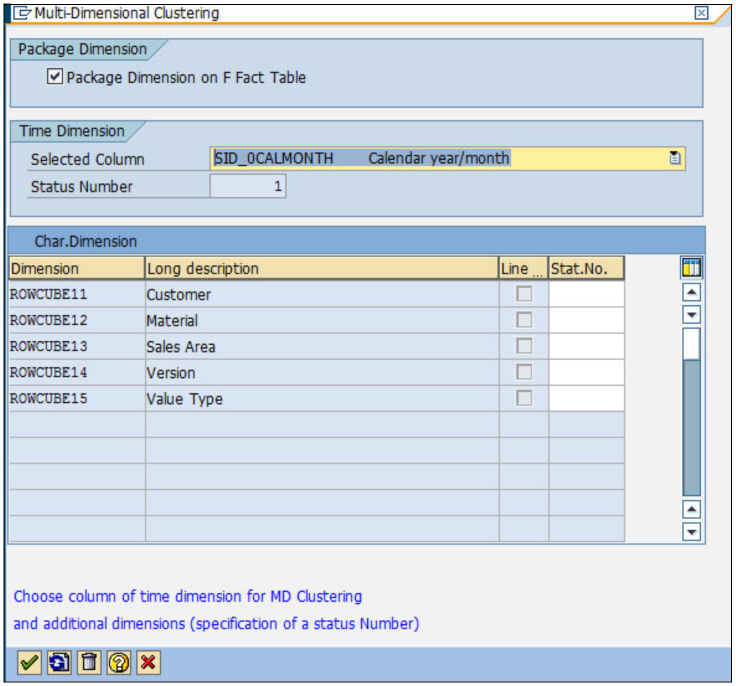



*Figure 8-22 Specification of multi-dimensional clustering for the sample row-organized InfoCube*

3. Save and activate the InfoCube. After that, the tables listed in Figure 8-23 exist for ROWCUBE1 InfoCube.

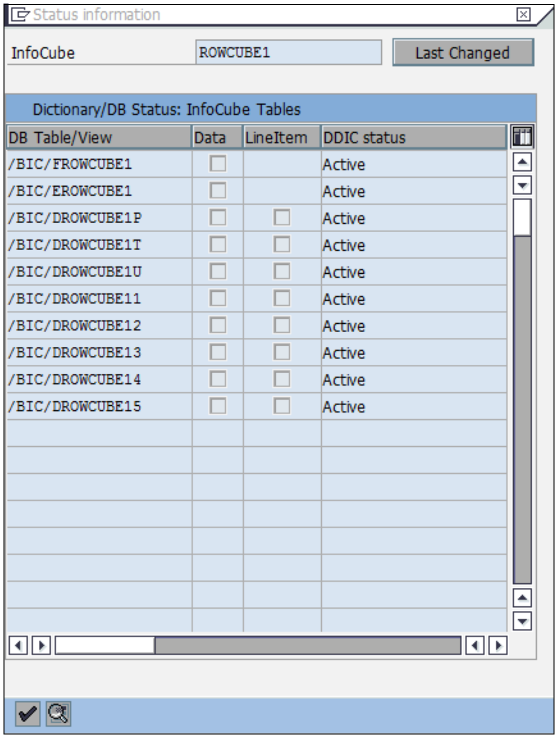



*Figure 8-23 Tables of the sample row-organized InfoCube*

### *Comparison of the F fact tables*

You can use the DB2 **db2look** tool to extract the DDL for the two fact tables:

```
db2look -d D01 -a -e -t "/BIC/FBLUCUBE1"
```

Example 8-2 shows the DDL of the table.

*Example 8-2 DDL of column-organized F fact table*

```
CREATE TABLE "SAPD01  "."/BIC/FBLUCUBE1"  (
                  "KEY_BLUCUBE1P" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE1T" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE1U" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE11" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE12" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE13" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE14" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE15" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
```

```
 "/B49/S_CRMEM_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_CRMEM_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INCORDCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INCORDQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INCORDVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INVCD_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INVCD_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_INVCD_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_OPORDQTYB" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_OPORDVALS" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_ORD_ITEMS" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_RTNSCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_RTNSQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_RTNSVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
 "/B49/S_RTNS_ITEM" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 )
DISTRIBUTE BY HASH("KEY_BLUCUBE11",
"KEY_BLUCUBE12",
"KEY_BLUCUBE13",
"KEY_BLUCUBE14",
"KEY_BLUCUBE15",
"KEY_BLUCUBE1T",
"KEY_BLUCUBE1U")
  IN "D01#FACTD" INDEX IN "D01#FACTI"
ORGANIZE BY COLUMN;
```

`ORGANIZE BY COLUMN` shows that the table is a column-organized table. The table has no indexes.

Together with each column-organized table, DB2 automatically creates a corresponding synopsis table. The synopsis table is automatically maintained when data is inserted or loaded into the column-organized table and is used for data skipping during SQL query execution. Space consumption and the cost for updating the synopsis tables when the data in the column-organized table is changed are much lower than that for secondary indexes. Synopsis tables reside in the SYSIBM schema use the following naming convention:

`SYN<timestamp>_<column-organized table name>.`

You can retrieve the synopsis table for /BIC/FBLUCUBE1 with the SQL statement as shown in Example 8-3.

*Example 8-3 Determine name of synopsis table for column-organized F fact table*

```
db2 => SELECT TABNAMEFROM SYSCAT.TABLES WHERE TABNAME LIKE ‘SYN%_/BIC/FBLUCUBE1’

TABNAME
---------------------------------------
SYN140205190245349822_/BIC/FBLUCUBE1
```

You can retrieve the layout of the synopsis table with the DB2 **DESCRIBE** command, as shown in Example 8-4.

*Example 8-4 Structure of synopsis table of column-organized F fact table*

```
db2 => DESCRIBE TABLE SYSIBM.SYN140205190245349822_/BIC/FBLUCUBE1

                                Data type                     Column
Column name                     schema    Data type name      Length     Scale Nulls
------------------------------- --------- ------------------- ---------- ----- ------
KEY_BLUCUBE1PMIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE1PMAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE1TMIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE1TMAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE1UMIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE1UMAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE11MIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE11MAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE12MIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE12MAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE13MIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE13MAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE14MIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE14MAX                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE15MIN                SYSIBM    INTEGER                      4     0 No
KEY_BLUCUBE15MAX                SYSIBM    INTEGER                      4     0 No
/B49/S_CRMEM_CSTMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_CRMEM_CSTMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_CRMEM_QTYMIN             SYSIBM    DECIMAL                     17     3 No
/B49/S_CRMEM_QTYMAX             SYSIBM    DECIMAL                     17     3 No
/B49/S_CRMEM_VALMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_CRMEM_VALMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_INCORDCSTMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_INCORDCSTMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_INCORDQTYMIN             SYSIBM    DECIMAL                     17     3 No
/B49/S_INCORDQTYMAX             SYSIBM    DECIMAL                     17     3 No
/B49/S_INCORDVALMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_INCORDVALMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_INVCD_CSTMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_INVCD_CSTMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_INVCD_QTYMIN             SYSIBM    DECIMAL                     17     3 No
/B49/S_INVCD_QTYMAX             SYSIBM    DECIMAL                     17     3 No
/B49/S_INVCD_VALMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_INVCD_VALMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_OPORDQTYBMIN             SYSIBM    DECIMAL                     17     3 No
/B49/S_OPORDQTYBMAX             SYSIBM    DECIMAL                     17     3 No
/B49/S_OPORDVALSMIN             SYSIBM    DECIMAL                     17     2 No
/B49/S_OPORDVALSMAX             SYSIBM    DECIMAL                     17     2 No
/B49/S_ORD_ITEMSMIN             SYSIBM    DECIMAL                     17     3 No
/B49/S_ORD_ITEMSMAX             SYSIBM    DECIMAL                     17     3 No
/B49/S_RTNSCSTMIN               SYSIBM    DECIMAL                     17     2 No
/B49/S_RTNSCSTMAX               SYSIBM    DECIMAL                     17     2 No
/B49/S_RTNSQTYMIN               SYSIBM    DECIMAL                     17     3 No
/B49/S_RTNSQTYMAX               SYSIBM    DECIMAL                     17     3 No
```

```
/B49/S_RTNSVALMIN                SYSIBM    DECIMAL                    17      2 No
/B49/S_RTNSVALMAX                SYSIBM    DECIMAL                    17      2 No
/B49/S_RTNS_ITEMMIN              SYSIBM    DECIMAL                    17      3 No
/B49/S_RTNS_ITEMMAX              SYSIBM    DECIMAL                    17      3 No
TSNMIN                           SYSIBM    BIGINT                      8      0 No
TSNMAX                           SYSIBM    BIGINT                      8      0 No
```

Space consumption and performance of synopsis tables can be monitored in the DBA Cockpit. Choose, for example, **Tables and Indexes** → **Top Space Consumers**, in the DBA Cockpit navigation frame, and set the table schema filter to **SYSIBM** and the table name filter to **SYN***. Space consumption information about synopsis tables is displayed in the table on the right (Figure 8-24).

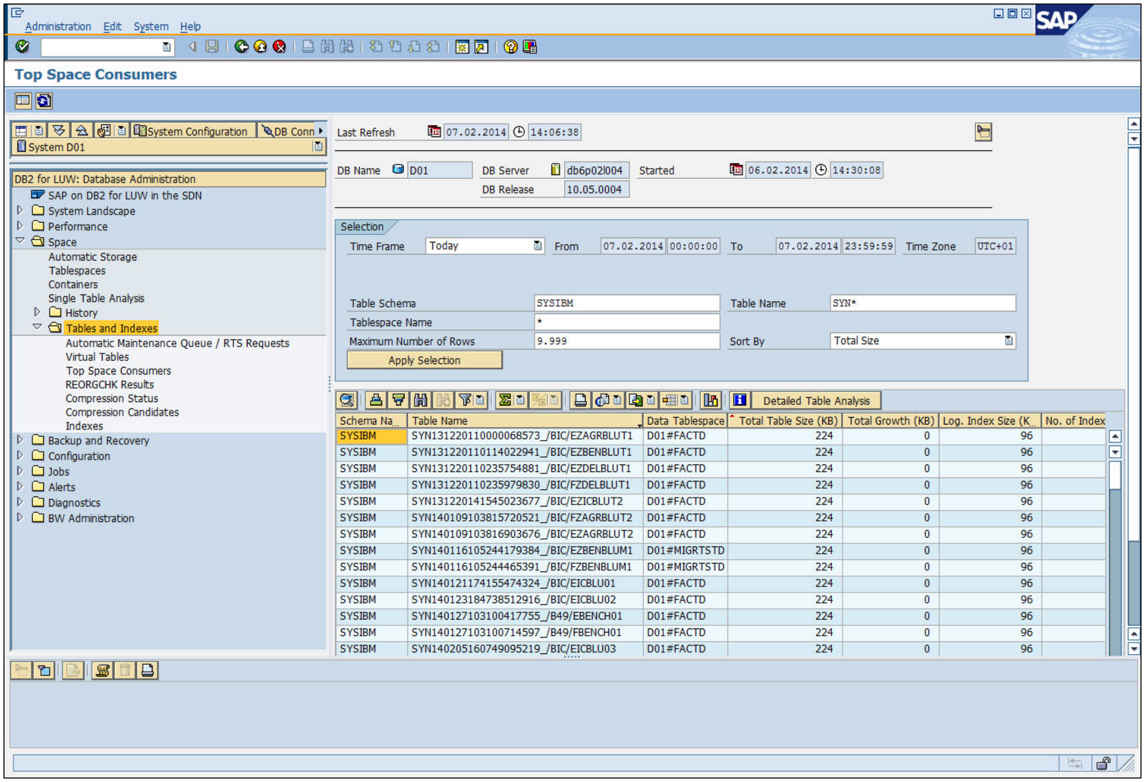



*Figure 8-24   DBA Cockpit: Space consumption information for synopsis tables*

Now run **db2look** for the row-organized table:

```
db2look -d D01 -a -e -t "/BIC/FROWCUBE1"
```

Example 8-5 shows the DDL for the table.

*Example 8-5 DDL of row-organized F fact table*

```
CREATE TABLE "SAPD01  "."/BIC/FROWCUBE1"  (
                  "KEY_ROWCUBE1P" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE1T" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE1U" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE11" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE12" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE13" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE14" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_ROWCUBE15" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "SID_0CALMONTH" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_OPORDQTYB" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_OPORDVALS" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_ORD_ITEMS" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_RTNSCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_RTNSQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_RTNSVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_RTNS_ITEM" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 )
                 COMPRESS YES ADAPTIVE
                 DISTRIBUTE BY HASH("KEY_ROWCUBE11",
                 "KEY_ROWCUBE12",
                 "KEY_ROWCUBE13",
                 "KEY_ROWCUBE14",
                 "KEY_ROWCUBE15",
                 "KEY_ROWCUBE1T",
                 "KEY_ROWCUBE1U")
                   IN "D01#FACTD" INDEX IN "D01#FACTI"
                 ORGANIZE BY ROW USING (
                  ( "KEY_ROWCUBE1P" ) ,
                  ( "SID_0CALMONTH" ) )
                 ;
ALTER TABLE "SAPD01  "."/BIC/FROWCUBE1" LOCKSIZE BLOCKINSERT;

CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~020" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE1T" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~040" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE11" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;
```

```
CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~050" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE12" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~060" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE13" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~070" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE14" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/FROWCUBE1~080" ON "SAPD01  "."/BIC/FROWCUBE1"
                ("KEY_ROWCUBE15" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;
```

The table has two MDC dimensions and six single-column indexes that must be maintained during INSERT/UPDATE/DELETE and LOAD operations.

### *Comparison of the E fact tables*

You can use the DB2 **db2look** tool to extract the DDL for the two fact tables.

```
db2look -d D01 -a -e -t "/BIC/EBLUCUBE1"
```

Example 8-6 shows the DDL for the table.

*Example 8-6   DDL of column-organized E fact table*

```
CREATE TABLE "SAPD01  "."/BIC/EBLUCUBE1"  (
                  "KEY_BLUCUBE1P" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE1T" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE1U" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE11" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE12" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE13" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE14" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "KEY_BLUCUBE15" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_CRMEM_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INCORDVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_INVCD_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_OPORDQTYB" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
```

```
                 "/B49/S_OPORDVALS" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_ORD_ITEMS" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_RTNSCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_RTNSQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_RTNSVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_RTNS_ITEM" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 )
                DISTRIBUTE BY HASH("KEY_BLUCUBE11",
                "KEY_BLUCUBE12",
                "KEY_BLUCUBE13",
                "KEY_BLUCUBE14",
                "KEY_BLUCUBE15",
                "KEY_BLUCUBE1T",
                "KEY_BLUCUBE1U")
                  IN "D01#FACTD" INDEX IN "D01#FACTI"
                ORGANIZE BY COLUMN;

ALTER TABLE "SAPD01  "."/BIC/EBLUCUBE1"
        ADD CONSTRAINT "/BIC/EBLUCUBE1~P" UNIQUE
                ("KEY_BLUCUBE1T",
                 "KEY_BLUCUBE11",
                 "KEY_BLUCUBE12",
                 "KEY_BLUCUBE13",
                 "KEY_BLUCUBE14",
                 "KEY_BLUCUBE15",
                 "KEY_BLUCUBE1U",
                 "KEY_BLUCUBE1P");
```

The table has one UNIQUE constraint. In addition, a synopsis table is created in schema SYSIBM.

Use the DB2 **db2look** tool to extract the DDL of the second row-organized fact table:

```
db2look -d D01 -a -e -t "/BIC/EROWCUBE1"
```

Example 8-7 shows the DDL for the table.

*Example 8-7 DDL for the row-organized fact table*

```
CREATE TABLE "SAPD01  "."/BIC/EROWCUBE1"  (
                 "KEY_ROWCUBE1P" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE1T" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE1U" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE11" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE12" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE13" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE14" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "KEY_ROWCUBE15" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "SID_0CALMONTH" INTEGER NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_CRMEM_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_CRMEM_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_CRMEM_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
                 "/B49/S_INCORDCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
```

```
          "/B49/S_INCORDQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_INCORDVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_INVCD_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_INVCD_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_INVCD_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_OPORDQTYB" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_OPORDVALS" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_ORD_ITEMS" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_RTNSCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_RTNSQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_RTNSVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
          "/B49/S_RTNS_ITEM" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 )
         COMPRESS YES ADAPTIVE
         DISTRIBUTE BY HASH("KEY_ROWCUBE11",
         "KEY_ROWCUBE12",
         "KEY_ROWCUBE13",
         "KEY_ROWCUBE14",
         "KEY_ROWCUBE15",
         "KEY_ROWCUBE1T",
         "KEY_ROWCUBE1U")
           IN "D01#FACTD" INDEX IN "D01#FACTI"
         ORGANIZE BY ROW USING (
          ( "SID_0CALMONTH" ) )
         ;

ALTER TABLE "SAPD01  "."/BIC/EROWCUBE1" LOCKSIZE BLOCKINSERT;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~020" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE1T" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~040" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE11" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~050" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE12" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~060" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE13" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~070" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE14" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/EROWCUBE1~080" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE15" ASC)
```

```
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE UNIQUE INDEX "SAPD01  "."/BIC/EROWCUBE1~P" ON "SAPD01  "."/BIC/EROWCUBE1"
                ("KEY_ROWCUBE1T" ASC,
                 "KEY_ROWCUBE11" ASC,
                 "KEY_ROWCUBE12" ASC,
                 "KEY_ROWCUBE13" ASC,
                 "KEY_ROWCUBE14" ASC,
                 "KEY_ROWCUBE15" ASC,
                 "KEY_ROWCUBE1U" ASC,
                 "KEY_ROWCUBE1P" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;
```

The table has one MDC dimension, six single-column indexes, and one unique multi-column index. The multi-column index corresponds to the unique constraint of the column-organized table.

### *Comparison of dimension tables*

As an example, we use the tables created for dimension 3 of InfoCubes `/BIC/DBLUCUBE13` and `/BIC/DROWCUBE13`.

We use the DB2 **`db2look`** tool to extract the DDL for the column-organized dimension table:

```
db2look -d D01 -a -e -t "/BIC/DBLUCUBE13"
```

Example 8-8 shows the DDL for the table.

*Example 8-8   DDL of column-organized dimension table*

```
CREATE TABLE "SAPD01  "."/BIC/DBLUCUBE13"  (
                  "DIMID" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_DISTR_CHA" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_DIVISION" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_SALESORG" INTEGER NOT NULL WITH DEFAULT 0 )
                 IN "D01#DIMD" INDEX IN "D01#DIMI"
                 ORGANIZE BY COLUMN;

ALTER TABLE "SAPD01  "."/BIC/DBLUCUBE13"
        ADD CONSTRAINT "/BIC/DBLUCUBE13~0" PRIMARY KEY
                ("DIMID");

ALTER TABLE "SAPD01  "."/BIC/DBLUCUBE13"
        ADD CONSTRAINT "/BIC/DBLUCUBE13~99" UNIQUE
                ("/B49/S_DISTR_CHA",
                 "/B49/S_DIVISION",
                 "/B49/S_SALESORG",
                 "DIMID");
```

The table has one primary key constraint and a unique constraint /BIC/DBLUCUBE13~99. This constraint is not created on row-organized dimension tables. It is used during ETL processing when new data is inserted into the InfoCube.

We use the DB2 **db2look** tool to extract the DDL for the row-organized dimension table.

```
db2look -d D01 -a -e -t "/BIC/DBLUCUBE13"
```

Example 8-9 shows the DDL for the table.

*Example 8-9 DDL of row-organized dimension table*

```
CREATE TABLE "SAPD01  "."/BIC/DROWCUBE13"  (
                  "DIMID" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_DISTR_CHA" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_DIVISION" INTEGER NOT NULL WITH DEFAULT 0 ,
                  "/B49/S_SALESORG" INTEGER NOT NULL WITH DEFAULT 0 )
                 COMPRESS YES ADAPTIVE
                 IN "D01#DIMD" INDEX IN "D01#DIMI"
                 ORGANIZE BY ROW;

-- DDL Statements for Indexes on Table "SAPD01  "."/BIC/DROWCUBE13"
CREATE UNIQUE INDEX "SAPD01  "."/BIC/DROWCUBE13~0" ON "SAPD01  "."/BIC/DROWCUBE13"
                ("DIMID" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;
-- DDL Statements for Primary Key on Table "SAPD01  "."/BIC/DROWCUBE13"
ALTER TABLE "SAPD01  "."/BIC/DROWCUBE13"
        ADD CONSTRAINT "/BIC/DROWCUBE13~0" PRIMARY KEY
                ("DIMID");

CREATE INDEX "SAPD01  "."/BIC/DROWCUBE13~01" ON "SAPD01  "."/BIC/DROWCUBE13"
                ("/B49/S_DISTR_CHA" ASC,
                 "/B49/S_DIVISION" ASC,
                 "/B49/S_SALESORG" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/DROWCUBE13~02" ON "SAPD01  "."/BIC/DROWCUBE13"
                ("/B49/S_DIVISION" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;

CREATE INDEX "SAPD01  "."/BIC/DROWCUBE13~03" ON "SAPD01  "."/BIC/DROWCUBE13"
                ("/B49/S_SALESORG" ASC)
                PCTFREE 0 COMPRESS YES
                INCLUDE NULL KEYS ALLOW REVERSE SCANS;
```

The row-organized table has the same primary key constraint as the column-organized table and three additional indexes. There is no correspondence to the unique constraint /BIC/DBLUCUBE13~99.

Due to the structure and access patterns of dimensions tables, data skipping brings no benefit. Thus, as of DB2 Cancun Release 10.5.0.4, synopsis tables are no longer created for InfoCube dimension tables. This is achieved with the DB2 registry setting `DB2_CDE_WITHOUT_SYNOPSIS=%:/%/D%` which is part of DB2_WORKLOAD=SAP.

If you have created column-organized InfoCubes with DB2 10.5 FP3aSAP or lower, you can remove the dimension synopsis tables with the following steps:

1. Identify the dimension tables for which synopsis tables exist with the following SQL statement:

   ```
   SELECT
   VARCHAR(TABNAME,40) AS SYNOPSIS,
   SUBSTR(TABNAME,23,20) AS DIMENSION
   FROM
   SYSCAT.TABLES
   WHERE
   TABSCHEMA='SYSIBM' AND
   TABNAME LIKE 'SYN%/%/D%'
   ```

   For example, the output might contain entries similar to Example 8-10 for the dimension tables of BLUCUBE1.

   *Example 8-10 Example output*

   ```
   SYNOPSIS                                 DIMENSION
   ---------------------------------------- --------------------
   SYN140723110501110392_/BIC/DBLUCUBE13    /BIC/DBLUCUBE13
   SYN140723110501383201_/BIC/DBLUCUBE14    /BIC/DBLUCUBE14
   SYN140723110500543747_/BIC/DBLUCUBE11    /BIC/DBLUCUBE11
   SYN140723110500936031_/BIC/DBLUCUBE12    /BIC/DBLUCUBE12
   SYN140723110501571646_/BIC/DBLUCUBE15    /BIC/DBLUCUBE15
   SYN140723110501770345_/BIC/DBLUCUBE1P    /BIC/DBLUCUBE1P
   SYN140723110501952166_/BIC/DBLUCUBE1T    /BIC/DBLUCUBE1T
   SYN140723110502188692_/BIC/DBLUCUBE1U    /BIC/DBLUCUBE1U
   ```

2. Run `ADMIN_MOVE_TABLE` for each dimension table, where <schema> is the table schema and <table> is the table name, as follows:

   ```
   CALL
   SYSPROC.ADMIN_MOVE_TABLE('<schema>','<table>','','','','','','','','COPY_US
   E_LOAD','MOVE')
   ```

   With this step, the dimension tables are copied to new column-organized tables without synopsis tables.

## InfoCube index, statistics, and space management

When you load data into InfoCubes, you can drop fact table indexes before the data load and re-create them after. For row-organized InfoCubes with many indexes defined on the fact tables, re-creating the fact table indexes might take a long time if the fact table is large.

Because BLU Acceleration needs only primary key and unique constraints but not secondary indexes, the drop and re-create index operations in SAP BW do not perform any work. Therefore, they are fast. If have dropping and re-creating fact table indexes included in your process chains, modifications are necessary. These operations return immediately.

The SAP BW administration window for InfoCubes also contains the options to drop and to re-create InfoCube indexes. You get there when you call up the context menu of an InfoCube and select **Manage**, as shown in Figure 8-25.

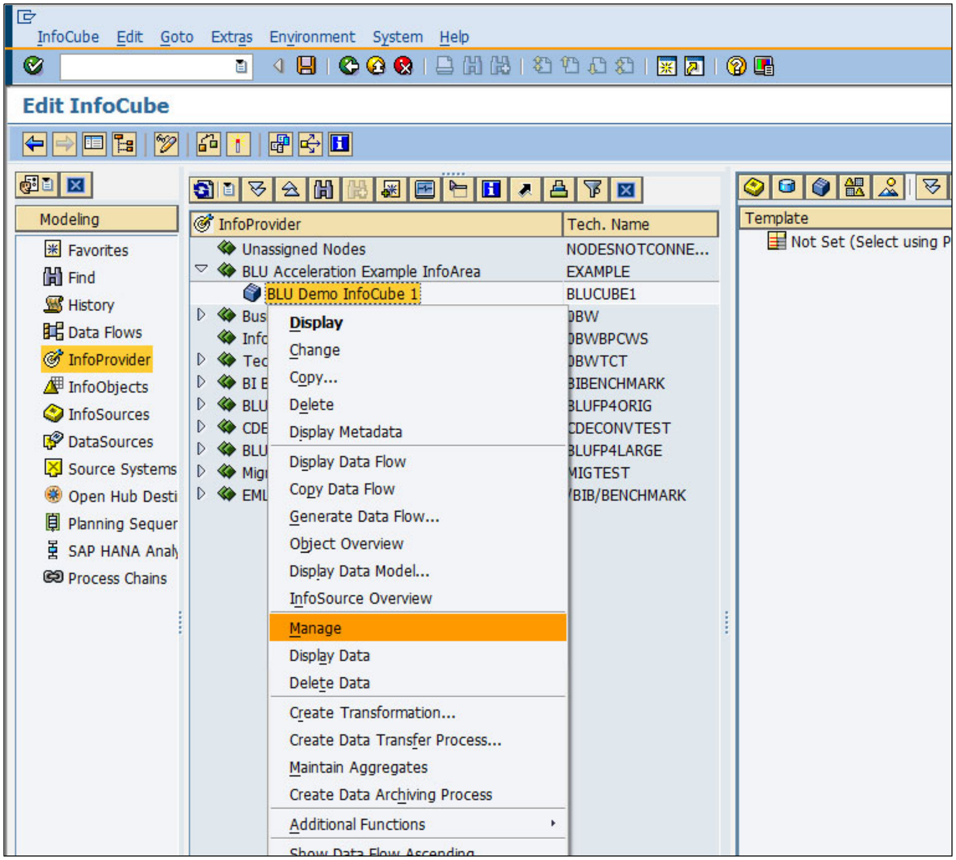



*Figure 8-25 Call-up of the InfoCube administration window in the SAP BW Datawarehousing Workbench*

Figure 8-26 shows the “Performance” tab page of the InfoCube administration window that has the following options:

- Delete DB Indexes (Now)
- Repair DB Indexes (Now)
- Create DB Index (Batch)
- Delete DB Index (Batch)

These options perform no work on BLU Acceleration InfoCubes.

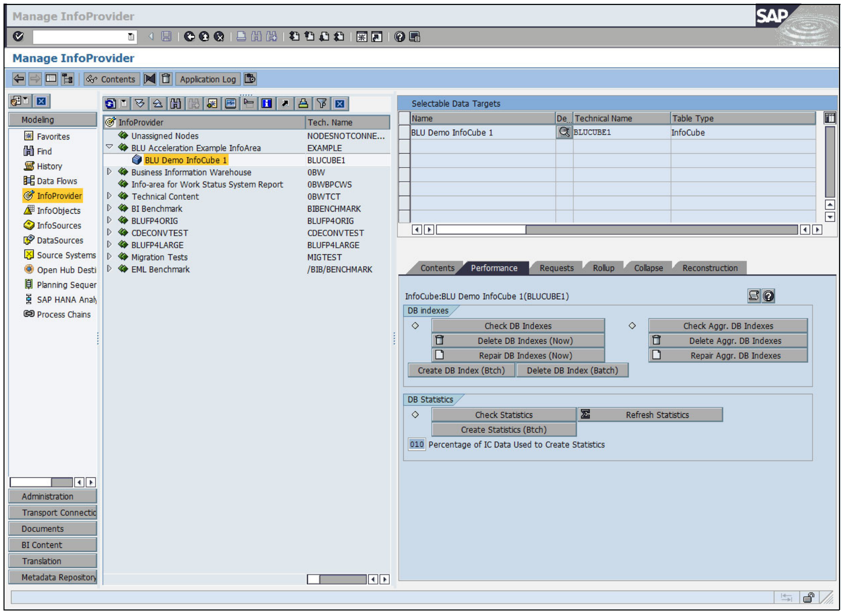


*Figure 8-26 InfoCube index and statistics management panel*

As for row-organized tables, DB2 collects statistics on BLU Acceleration tables automatically. You might still want to include statistics collection into your data load process chains to make sure that InfoCube statistics are up-to-date immediately after new data was loaded.

SAP BW transaction RSRV provides an option to check and repair the indexes of an InfoCube. You can run this check also for BLU Acceleration InfoCubes. Figure 8-27 shows “Database indexes of InfoCube and its Aggregates” is run for the BLU Acceleration InfoCube BLUCUBE1.

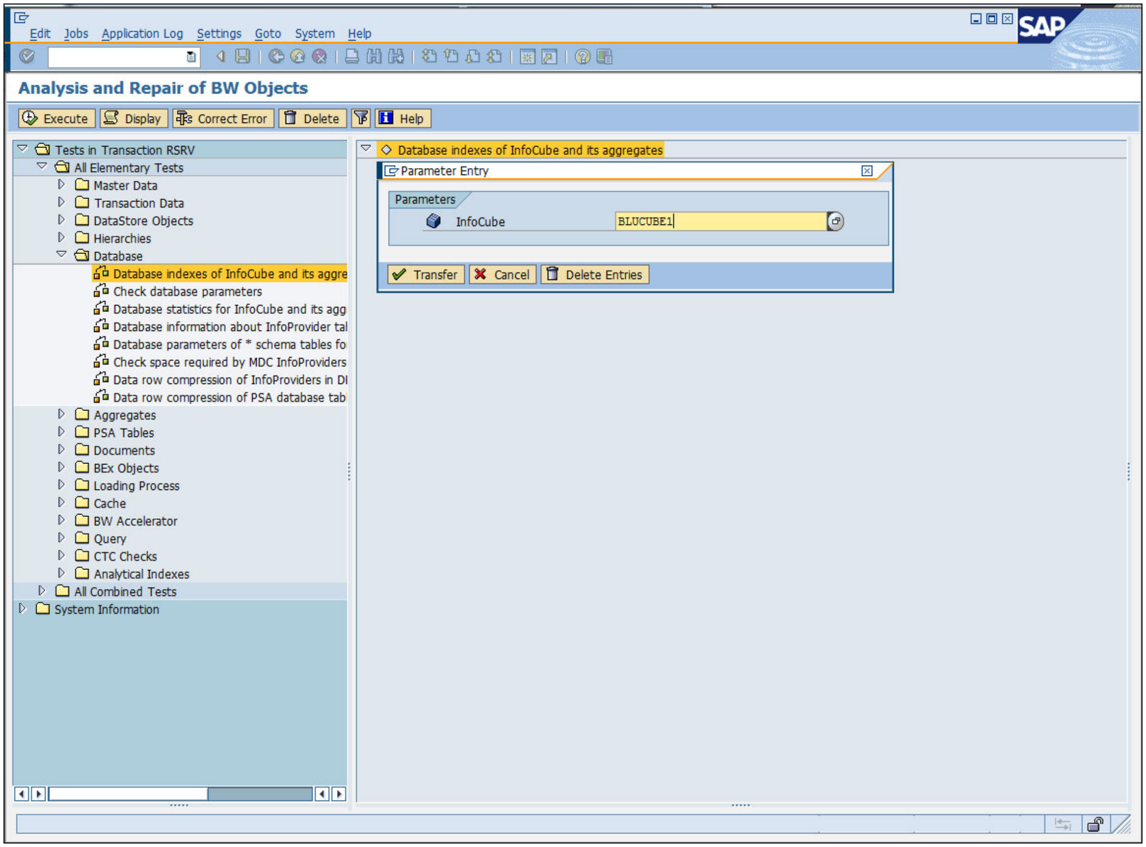



*Figure 8-27 SAP BW Transaction RSRV: InfoCube index check and repair function*

The result is shown in Figure 8-28. Only the unique constraint that is defined on the E fact table and the unique constraints that are defined on the dimension tables are listed.

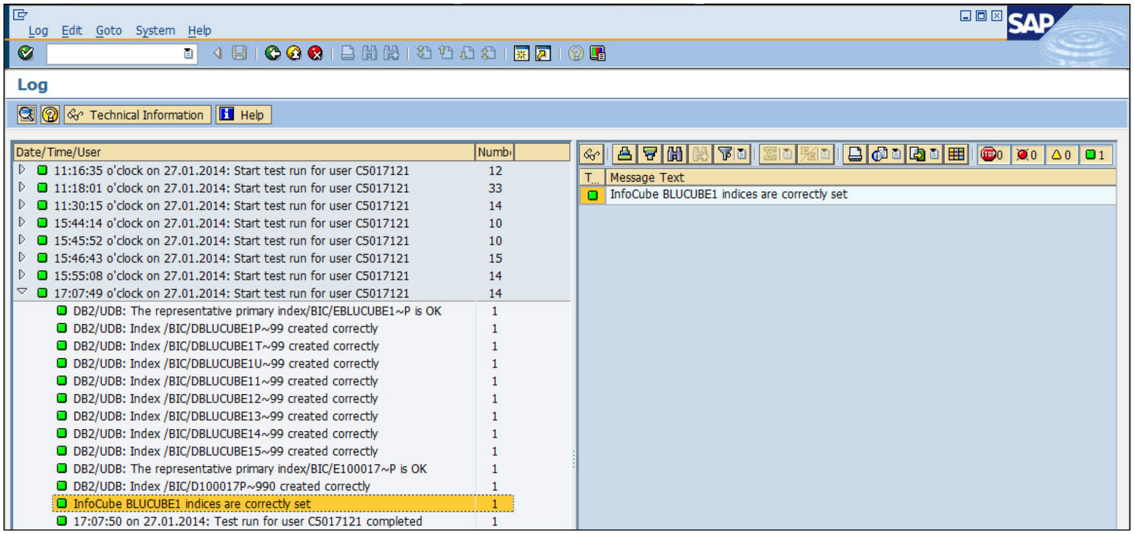



*Figure 8-28 Result of index check for sample BLU Acceleration InfoCube*

Figure 8-29 shows the output for the row-organized InfoCube ROWCUBE1 in comparison. For this InfoCube, many more indexes are listed.

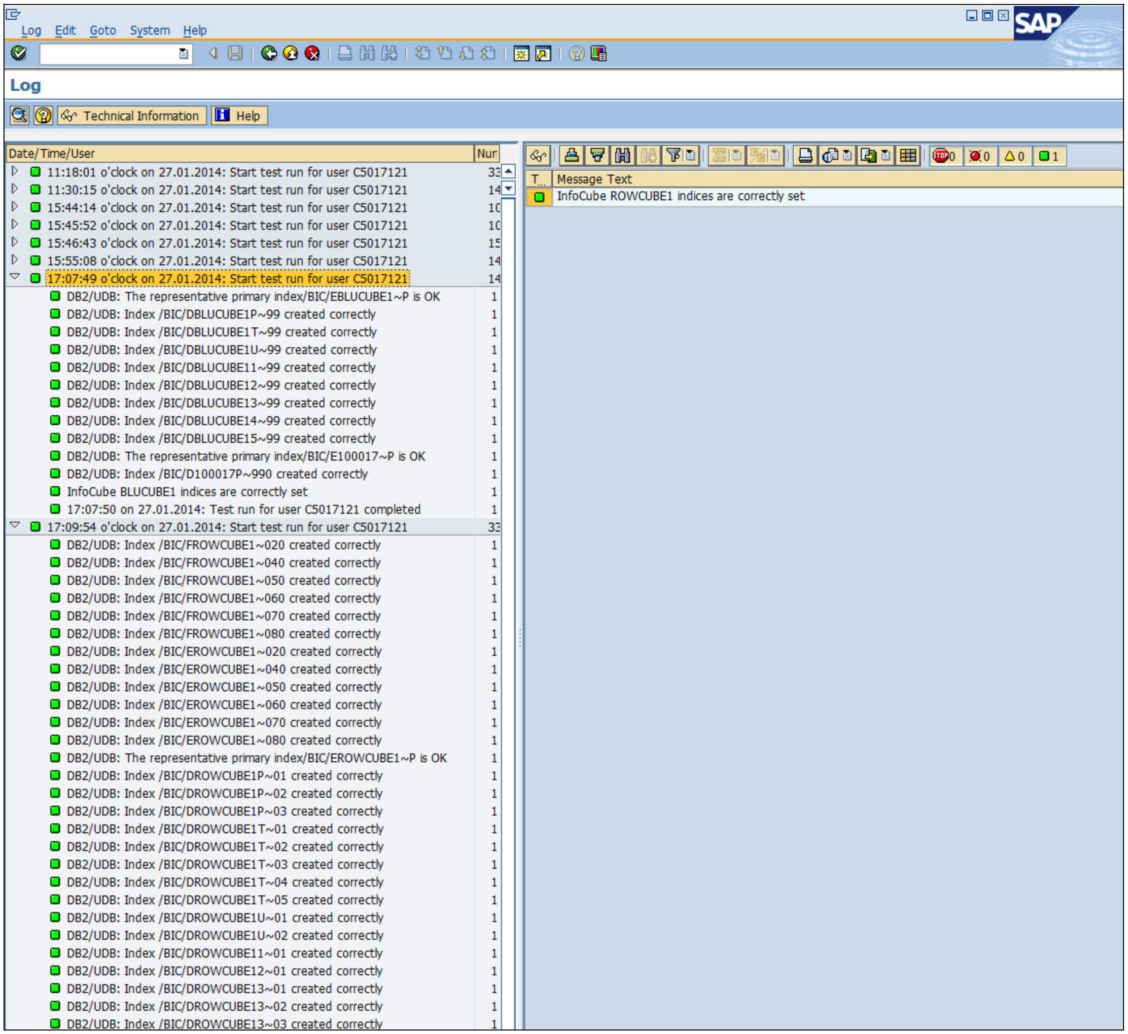


Figure 8-29 Comparison of index check output of BLU Acceleration InfoCube with row-organized InfoCube

## 8.5.2 Flat InfoCubes in SAP BW 7.40

This section provides information about flat InfoCubes, which become available for DB2 as of SAP BW 7.40 Support Package 8. We show how to create flat InfoCubes and how to convert existing standard InfoCubes to flat InfoCubes. We illustrate the procedure with an example in an SAP BW 7.40 system in which the BI DataMart Benchmark InfoCubes have been installed (InfoArea BI Benchmark).

Note the following important information about flat InfoCubes:

- Flat InfoCubes are only supported with DB2 for LUW Cancun Release 10.5.0.4 and higher. The tables of flat InfoCubes are always created as column-organized tables. For optimal performance of reporting on flat InfoCubes the InfoObjects that are referenced by the InfoCubes should also be implemented with column-organized tables (see 8.5.4, “Column-Organized InfoObjects in SAP BW” on page 332).
- Both cumulative and non-cumulative InfoCubes can be created as flat InfoCubes. However, DB2 does not support the creation of real-time InfoCubes as flat InfoCubes.
- Flat InfoCubes cannot be created directly in the Data Warehousing Workbench. You must create a standard InfoCube first and then convert it to flat with report RSDU_REPART_UI.
- You can also convert existing standard InfoCubes that contain data to flat InfoCubes with report RSDU_REPART_UI.
- When you install InfoCubes from the SAP BI Content, they are created as non-flat InfoCubes. If you want these InfoCubes to be flat you need to convert them with report RSDU_REPART_UI.
- If you want to create a flat semantically partitioned InfoCube, you must first create a non-flat InfoCube and then convert the InfoCubes of which the semantically partitioned InfoCube consists to flat InfoCubes with report RSDU_REPART_UI.
- When you create a flat InfoCube and transport it to another SAP BW system, the following is created in the target system:
  - If the InfoCube already exists in the target system, the target system settings are preserved, that is:
    - If the InfoCube is flat in the target system it remains flat.
    - If the InfoCube is non-flat in the target system it remains non-flat.
  - If the InfoCube does not yet exist in the target system it is created as non-flat InfoCube and has to be converted to flat with report RSDU_REPART_UI.

Follow these steps to create a new flat InfoCube:

1. Create a new InfoCube in the Data Warehousing Workbench. New InfoCubes are always created as standard non-flat InfoCubes with two fact tables and up to 16 dimension tables.
2. Run report RSDU_REPART_UI to schedule and run a batch job that converts the standard InfoCube into a flat InfoCube.

In the following example, the creation of a flat InfoCube is shown that illustrates these steps in more detail:

1. Call transaction **RSA1** to get to the Data Warehousing Workbench. In the InfoArea BI Benchmark, select **`Benchmark InfoCube 01`** (technical name is `/BIB/BENCH01`), and choose **Copy** from the context menu (Figure 8-30).

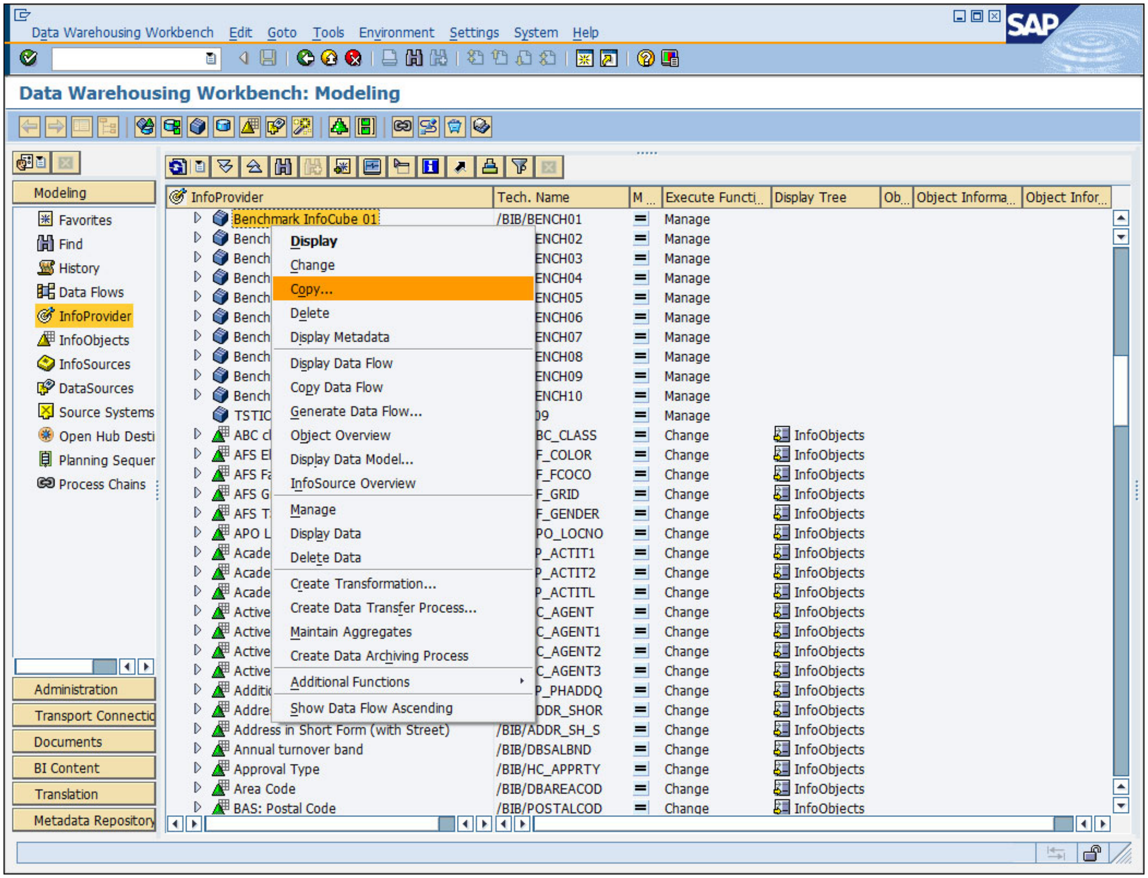



*Figure 8-30 Create a copy of InfoCube /BIB/BENCH01*

2. In the Edit InfoCube window, enter `FLATCUBE1` as the technical name, `BLU Flat Demo InfoCube 1` as the description, and select **EXAMPLE (BLU Acceleration Example InfoArea)** as the InfoArea. Click the **Create** icon (Figure 8-31).

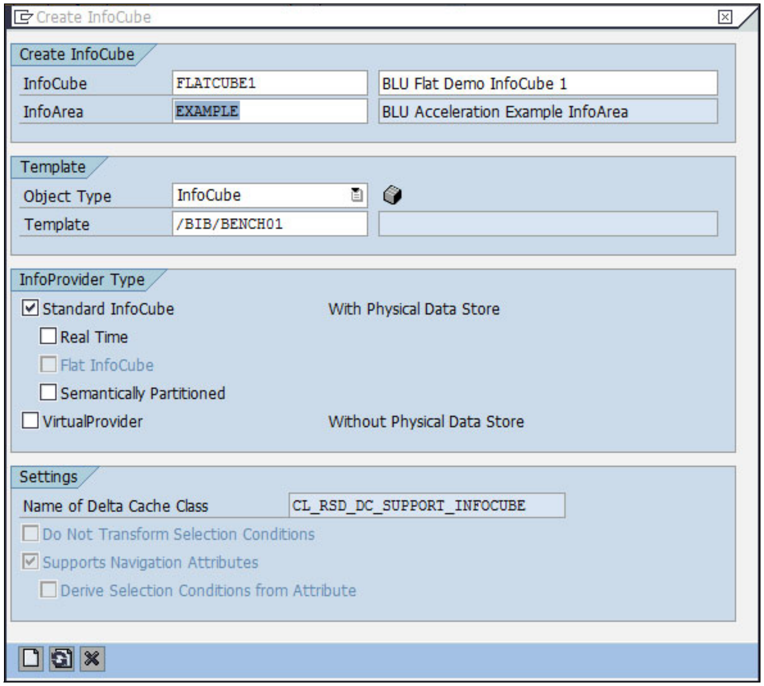



*Figure 8-31 Create InfoCube window for InfoCube FLATCUBE1*

3. The "Edit InfoCube" window in the Data Warehousing Workbench opens. Save and activate the InfoCube (Figure 8-32 on page 300). You do not need to choose between index clustering, multi-dimensional clustering (MDC) or column-organized tables in the Clustering dialog because it is not important how the InfoCube tables are created. When you convert the InfoCube to a flat InfoCube new column-organized tables are created for the InfoCube.

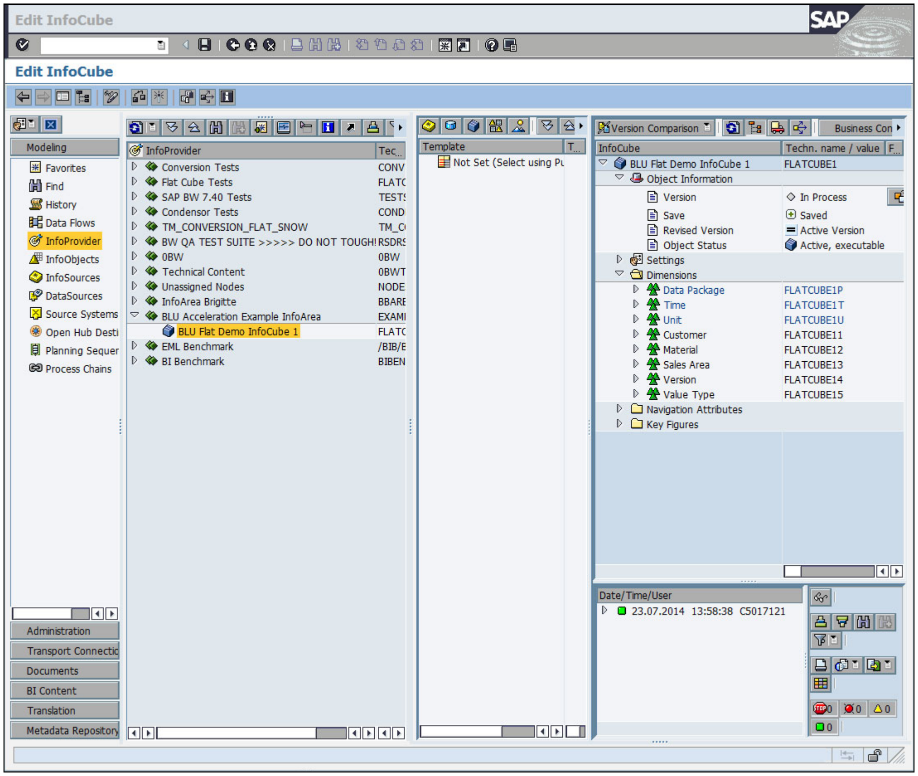



*Figure 8-32 Edit InfoCube window for InfoCube FLATCUBE1*

4. First the InfoCube is created as an enhanced star schema InfoCube with two fact and 16 dimension tables. You can check this by displaying the list of database tables that were created for the InfoCube: Choose **Extras** → **Information (logs/status)** from the menu. The “Info Selection” pop-up window is shown. Choose **Dictionary/DB Status** (Figure 8-33).

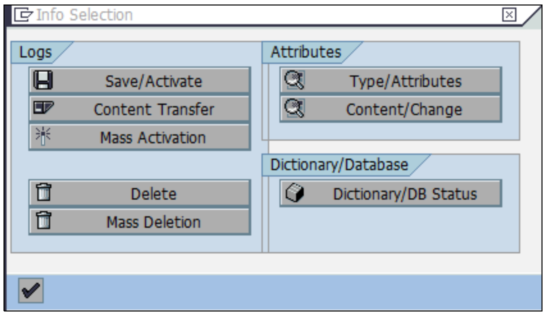



*Figure 8-33 Info Selection window for InfoCube FLATCUBE1*

5. You see the list of tables (Figure 8-34).

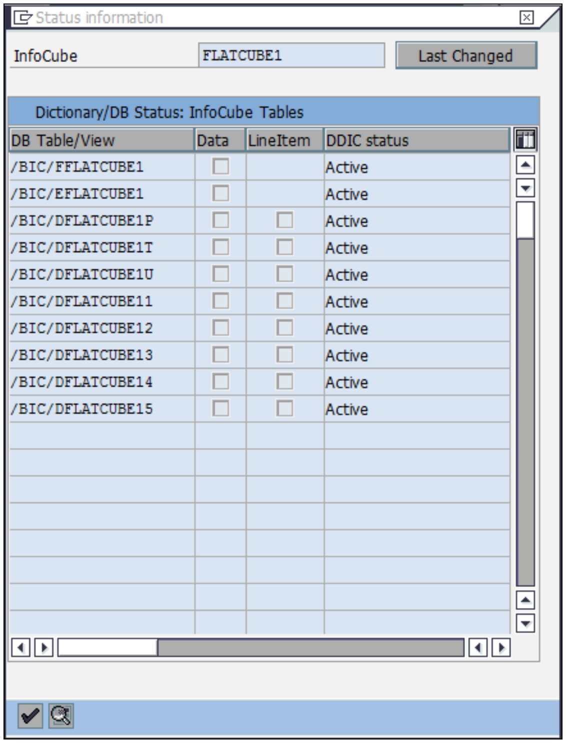
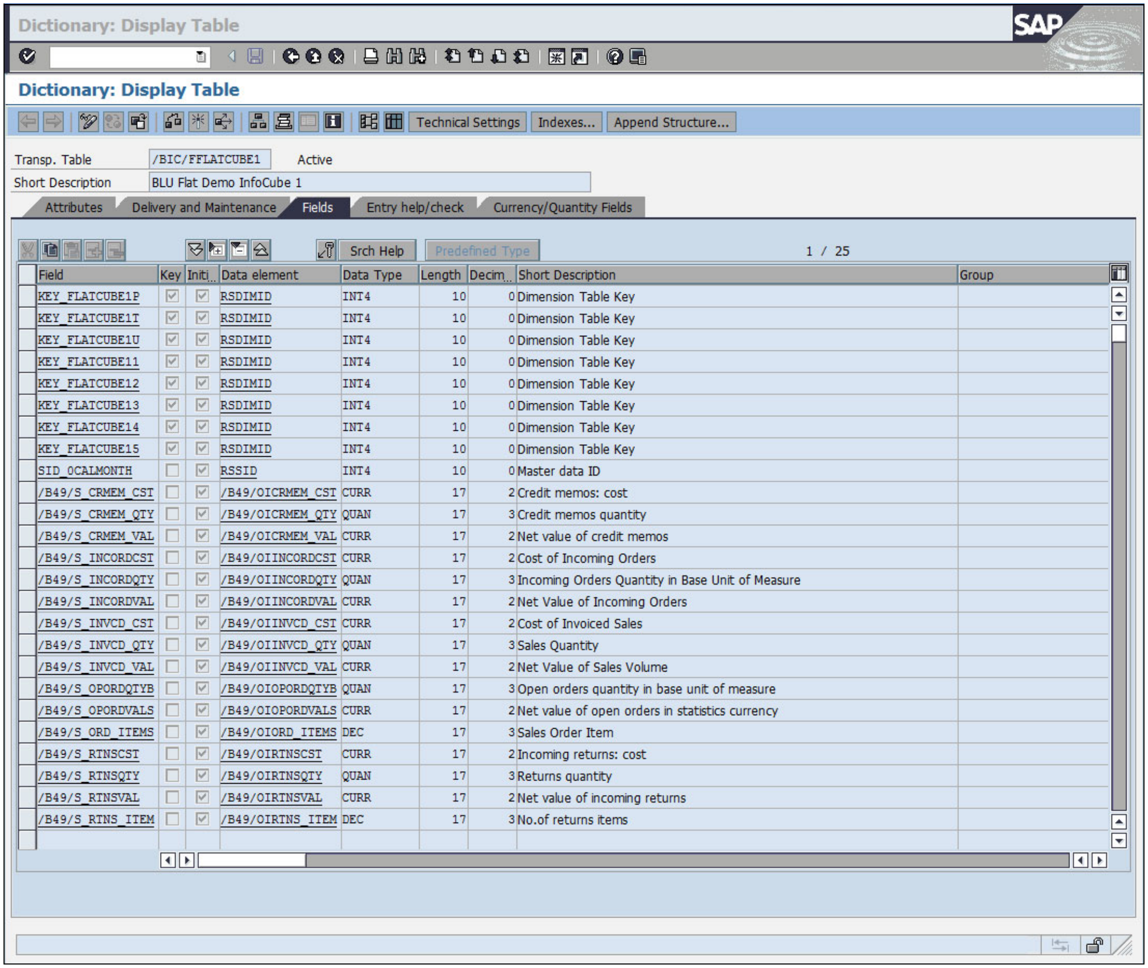



*Figure 8-34 List of database tables of InfoCube FLATCUBE1 before conversion to a flat InfoCube*

6. Double-click the F fact table **/BIC/FFLATCUBE1**. In the Dictionary: Display Table window, you see that the table has 8 dimension key columns and 16 key figures (Figure 8-35).




| Field | Key | Initi... | Data element | Data Type | Length | Decim... | Short Description | Group |
|---|---|---|---|---|---|---|---|---|
| KEY_FLATCUBE1P | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE1T | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE1U | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE11 | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE12 | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE13 | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE14 | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| KEY_FLATCUBE15 | ☑ | ☑ | RSDIMID | INT4 | 10 | 0 | Dimension Table Key | |
| SID_0CALMONTH | ☐ | ☑ | RSSID | INT4 | 10 | 0 | Master data ID | |
| /B49/S_CRMEM_CST | ☐ | ☑ | /B49/OICRMEM_CST | CURR | 17 | 2 | Credit memos: cost | |
| /B49/S_CRMEM_QTY | ☐ | ☑ | /B49/OICRMEM_QTY | QUAN | 17 | 3 | Credit memos quantity | |
| /B49/S_CRMEM_VAL | ☐ | ☑ | /B49/OICRMEM_VAL | CURR | 17 | 2 | Net value of credit memos | |
| /B49/S_INCORDCST | ☐ | ☑ | /B49/OIINCORDCST | CURR | 17 | 2 | Cost of Incoming Orders | |
| /B49/S_INCORDQTY | ☐ | ☑ | /B49/OIINCORDQTY | QUAN | 17 | 3 | Incoming Orders Quantity in Base Unit of Measure | |
| /B49/S_INCORDVAL | ☐ | ☑ | /B49/OIINCORDVAL | CURR | 17 | 2 | Net Value of Incoming Orders | |
| /B49/S_INVCD_CST | ☐ | ☑ | /B49/OIINVCD_CST | CURR | 17 | 2 | Cost of Invoiced Sales | |
| /B49/S_INVCD_QTY | ☐ | ☑ | /B49/OIINVCD_QTY | QUAN | 17 | 3 | Sales Quantity | |
| /B49/S_INVCD_VAL | ☐ | ☑ | /B49/OIINVCD_VAL | CURR | 17 | 2 | Net Value of Sales Volume | |
| /B49/S_OPORDQTYB | ☐ | ☑ | /B49/OIOPORDQTYB | QUAN | 17 | 3 | Open orders quantity in base unit of measure | |
| /B49/S_OPORDVALS | ☐ | ☑ | /B49/OIOPORDVALS | CURR | 17 | 2 | Net value of open orders in statistics currency | |
| /B49/S_ORD_ITEMS | ☐ | ☑ | /B49/OIORD_ITEMS | DEC | 17 | 3 | Sales Order Item | |
| /B49/S_RTNSCST | ☐ | ☑ | /B49/OIRTNSCST | CURR | 17 | 2 | Incoming returns: cost | |
| /B49/S_RTNSQTY | ☐ | ☑ | /B49/OIRTNSQTY | QUAN | 17 | 3 | Returns quantity | |
| /B49/S_RTNSVAL | ☐ | ☑ | /B49/OIRTNSVAL | CURR | 17 | 2 | Net value of incoming returns | |
| /B49/S_RTNS_ITEM | ☐ | ☑ | /B49/OIRTNS_ITEM | DEC | 17 | 3 | No.of returns items | |

*Figure 8-35 Display of F fact table of InfoCube FLATCUBE1 before conversion to a flat InfoCube*

7. You have to convert the InfoCube to a flat InfoCube using report RSDU_REPART_UI: Make sure that you switch from Edit mode to Display mode in the Data Warehousing Workbench or that you close the Data Warehousing Workbench so that the InfoCube is not locked in the SAP BW system. Call transaction SE38 and run report RSDU_REPART_UI (Figure 8-36).

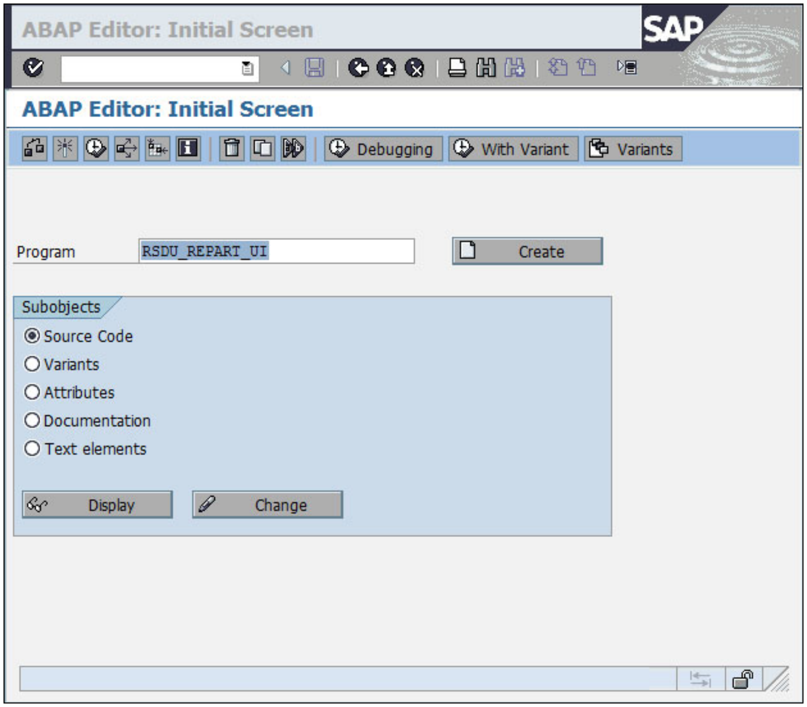



*Figure 8-36 Report for converting standard InfoCubes to flat InfoCubes*

8. Enter the name of the InfoCube (`FLATCUBE1`) in the InfoCube entry field and select **Non-flat to flat** conversion. For the conversion, a batch job is created that you can schedule. Choose **Initialize** (Figure 8-37 on page 304).

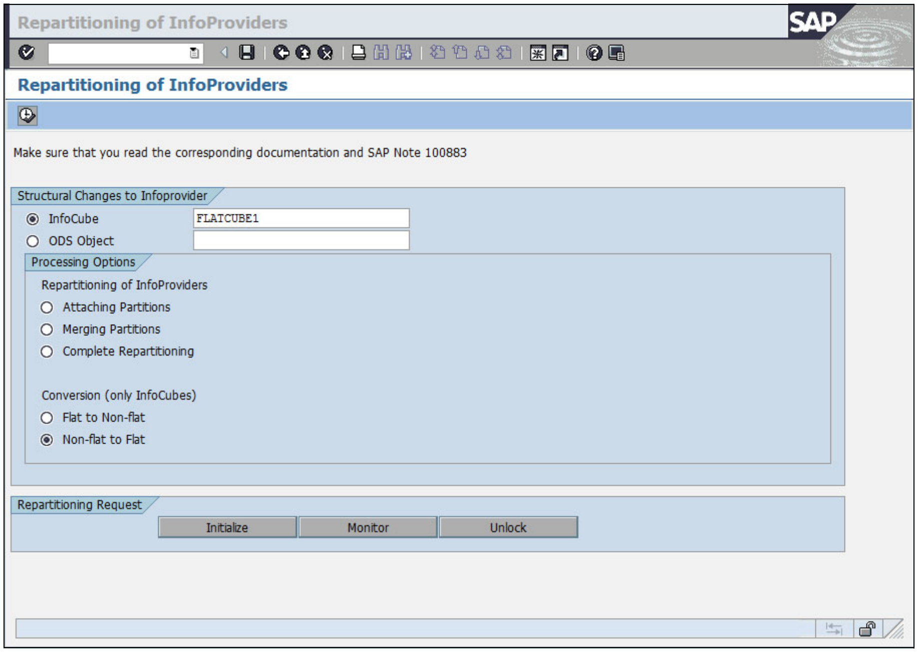



*Figure 8-37 Initialize flat InfoCube conversion*

9. Several pop-up windows with questions are shown. These questions are important when you convert InfoCubes that contain data to flat InfoCubes. In the current example, where the InfoCube is new and thus empty, you can answer these questions with **Yes**:

   – Confirm that you have taken a database backup (Figure 8-38).

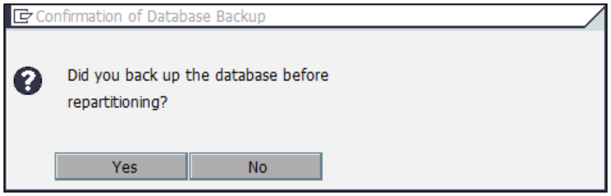



*Figure 8-38 Inquiry about database backup*

This is important when you convert existing InfoCubes with data to flat InfoCubes. During the conversion, new database tables are created, the data is copied to the new tables and the old tables are dropped. When you have a database backup, you can always go back to the state before the conversion was started.

– Confirm that you have compressed all requests that contain historical movements (Inquiry about compression of requests with historical movements). See Figure 8-39.

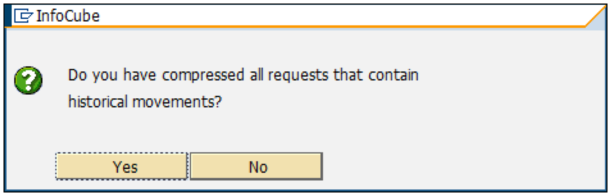



*Figure 8-39 Inquiry about compression of requests with historical movements*

This is important when you convert existing InfoCubes with data to flat InfoCubes. When you compress historical movements of non-flat InfoCubes, you set the "No Marker Update" flag in the compression window. For flat InfoCubes, this is handled differently: you define directly in the Data Transfer Process (DTP) whether you load requests that contain historical movements. However, this only has an effect for requests that are newly loaded after the conversion of the InfoCube to a flat InfoCube. For requests that already reside in the F fact table, it is no longer possible to detect and correctly compress requests that contain historical movements after the InfoCube conversion to a flat InfoCube.

> **Note:** When you convert existing InfoCubes with data, the report checks whether any uncompressed requests with package dimension ID 1 or 2 exist. If this is the case you must compress these requests before the conversion. The package dimension IDs 1 and 2 are reserved for marker records of non-cumulative InfoCubes and for historical movement data in flat InfoCubes.

10. The information message shown in Figure 8-40 is displayed.

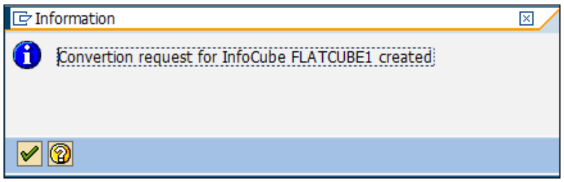



*Figure 8-40 Message about flat InfoCube conversion job creation*

When you confirm the message, the “Start Time” window in Figure 8-41 is shown.

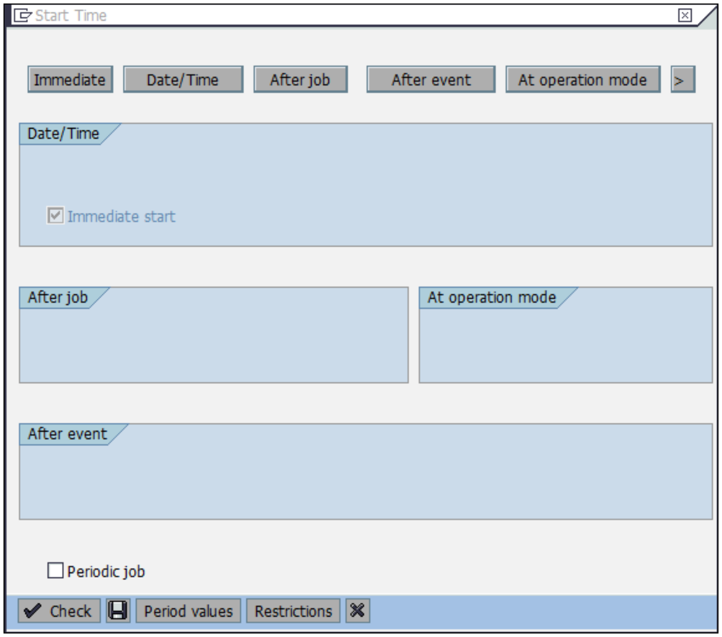



*Figure 8-41 Schedule flat InfoCube conversion job*

Choose **Immediate** as start time for the conversion batch job and choose **Enter**. The following information message is shown (Figure 8-42).

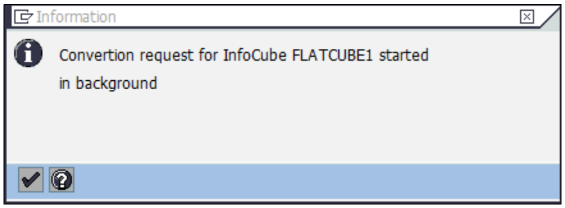



*Figure 8-42 Message about start of flat InfoCube conversion job*

11.On the window of report RSDU_REPART_UI, choose **Monitor** to track the progress of the conversion until it is finished successfully (Figure 8-43).

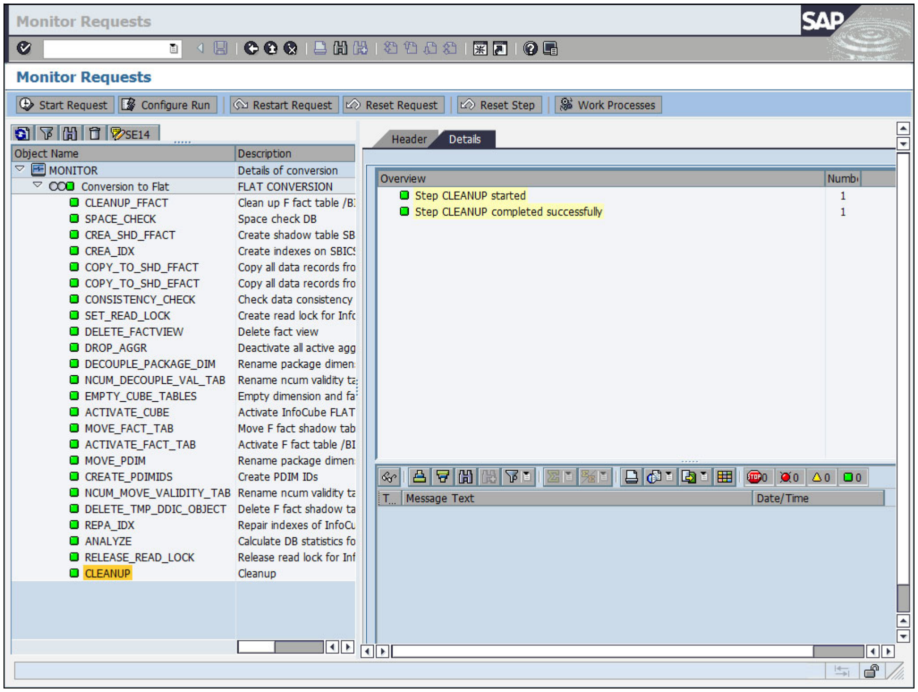



Figure 8-43 Successful conversion of InfoCube FLATCUBE1 to a flat InfoCube

12. Go back to the Data Warehousing Workbench and display the InfoCube FLATCUBE1 (Figure 8-44).

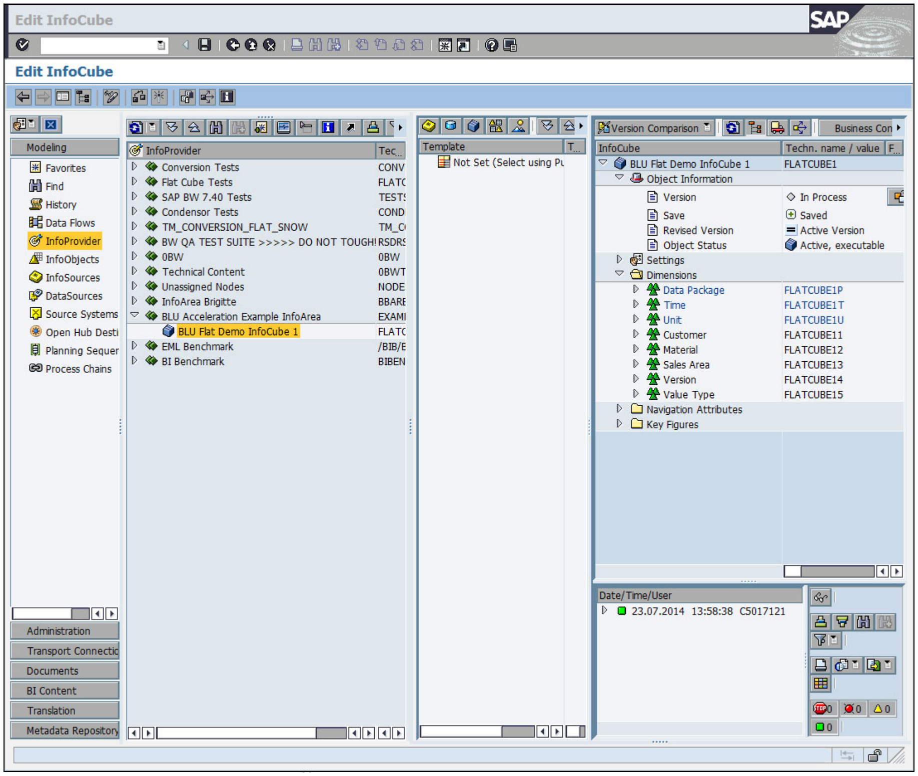


Figure 8-44 Edit InfoCube window for flat InfoCube FLATCUBE1

13. Choose **Extras** → **Information (logs/status)** from the menu. From the **Info Selection** pop-up window, select **Dictionary/DB status**. Now the list of InfoCube tables only contains two tables: the F fact table and the package dimension table (Figure 8-45).

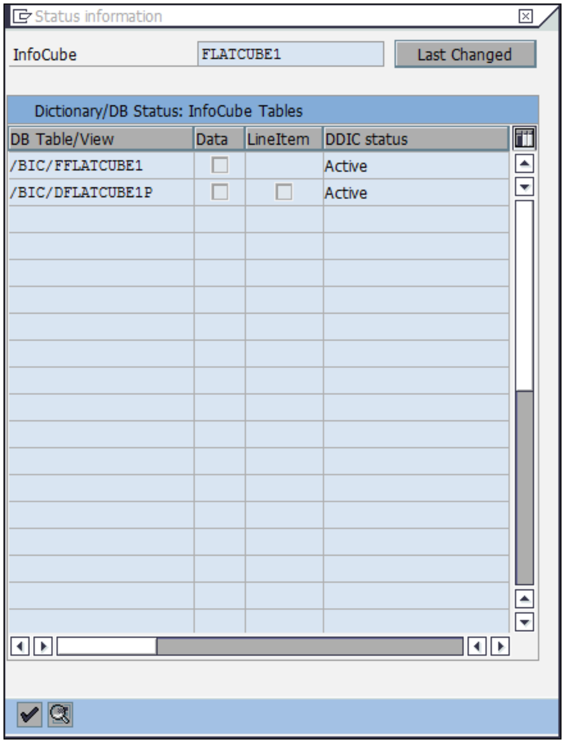



*Figure 8-45 Overview of database tables of flat InfoCube FLATCUBE1*

14. Double-click the F fact table **/BIC/FFLATCUBE1** to get to the SAP Data Dictionary.

    The table fields contain the package dimension key column, 14 columns for the SIDs of all the characteristics, and the 16 key figure columns (Figure 8-46).

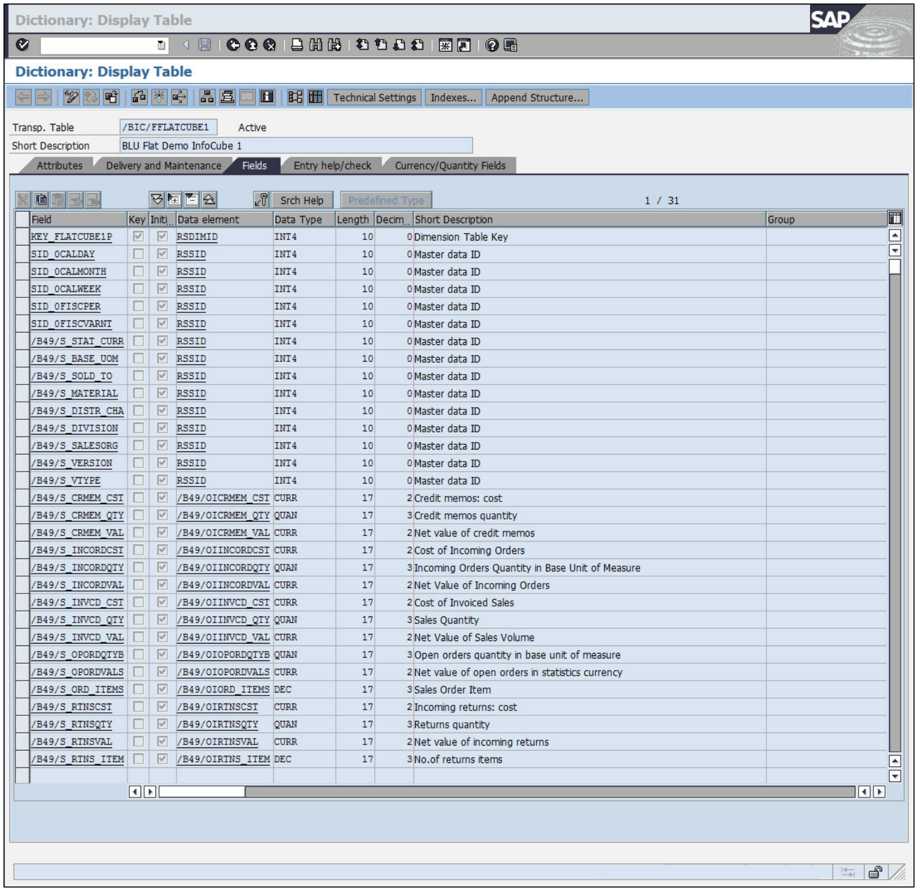



*Figure 8-46 Display table /BIC/FFLATCUBE1*

15. Choose **Utilities** → **Database Object** → **Database Utility** from the menu and select **Storage Parameters**. In the storage parameters window, you see that the table is column-organized (Figure 8-47).

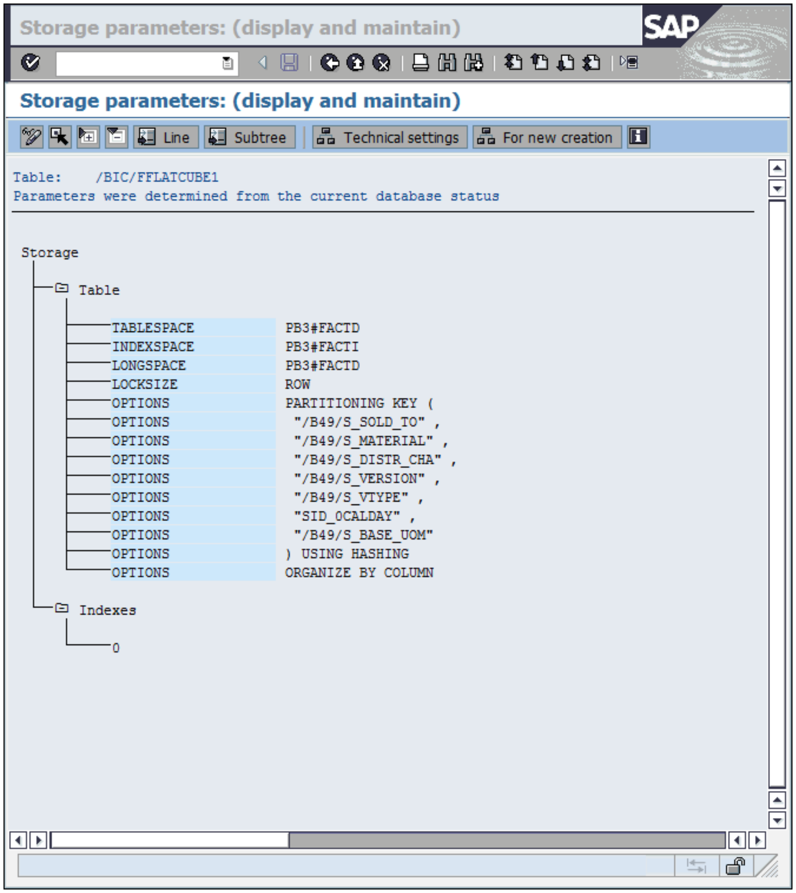



*Figure 8-47 Storage parameters of fact table of flat InfoCube FLATCUBE1*

16. Repeat steps 14 and 15 for the package dimension table. You see that the package dimension table is also column-organized.

## Table layout of flat InfoCubes

You can use the DB2 **db2look** tool to extract the DDL for the fact table:

```
db2look -d D01 -a -e -t "/BIC/FFLATCUBE1"
```

Example 8-11 shows the DDL of the table.

*Example 8-11 DDL describing the fact table of a flat InfoCube*

```
CREATE TABLE "SAPPB3  "."/BIC/FFLATCUBE1"  (
         "KEY_FLATCUBE1P" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_BASE_UOM" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_DISTR_CHA" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_DIVISION" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_MATERIAL" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_SALESORG" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_SOLD_TO" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_STAT_CURR" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_VERSION" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_VTYPE" INTEGER NOT NULL WITH DEFAULT 0 ,
         "SID_0CALDAY" INTEGER NOT NULL WITH DEFAULT 0 ,
         "SID_0CALMONTH" INTEGER NOT NULL WITH DEFAULT 0 ,
         "SID_0CALWEEK" INTEGER NOT NULL WITH DEFAULT 0 ,
         "SID_0FISCPER" INTEGER NOT NULL WITH DEFAULT 0 ,
         "SID_0FISCVARNT" INTEGER NOT NULL WITH DEFAULT 0 ,
         "/B49/S_CRMEM_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_CRMEM_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_CRMEM_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INCORDCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INCORDQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INCORDVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INVCD_CST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INVCD_QTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_INVCD_VAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_OPORDQTYB" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_OPORDVALS" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_ORD_ITEMS" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_RTNSCST" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_RTNSQTY" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_RTNSVAL" DECIMAL(17,2) NOT NULL WITH DEFAULT 0 ,
         "/B49/S_RTNS_ITEM" DECIMAL(17,3) NOT NULL WITH DEFAULT 0 )
      DISTRIBUTE BY HASH("/B49/S_SOLD_TO",
         "/B49/S_MATERIAL",
         "/B49/S_DISTR_CHA",
         "/B49/S_VERSION",
         "/B49/S_VTYPE",
         "SID_0CALDAY",
         "/B49/S_BASE_UOM")
      IN "PB3#FACTD" INDEX IN "PB3#FACTI"
      ORGANIZE BY COLUMN;
```

Use the following command to get the DDL of the package dimension table.

```
db2look -d D01 -a -e -t "/BIC/FFLATCUBE1"
```

Example 8-12 shows the result.

*Example 8-12 DDL describing the package dimension table of a flat InfoCube*

```
CREATE TABLE "SAPPB3  "."/BIC/DFLATCUBE1P"  (
  "DIMID" INTEGER NOT NULL WITH DEFAULT 0 ,
  "SID_0CHNGID" INTEGER NOT NULL WITH DEFAULT 0 ,
  "SID_0RECORDTP" INTEGER NOT NULL WITH DEFAULT 0 ,
  "SID_0REQUID" INTEGER NOT NULL WITH DEFAULT 0 )
IN "PB3#DIMD" INDEX IN "PB3#DIMI"
ORGANIZE BY COLUMN;

-- DDL Statements for Primary Key on Table "SAPPB3  "."/BIC/DFLATCUBE1P"
ALTER TABLE "SAPPB3  "."/BIC/DFLATCUBE1P"
  ADD CONSTRAINT "/BIC/DFLATCUBE1P~0" PRIMARY KEY
      ("DIMID");

-- DDL Statements for Unique Constraints on Table "SAPPB3  "."/BIC/DFLATCUBE1P"
ALTER TABLE "SAPPB3  "."/BIC/DFLATCUBE1P"
  ADD CONSTRAINT "/BIC/DFLATCUBE1P99" UNIQUE
      ("SID_0CHNGID",
       "SID_0RECORDTP",
       "SID0REQUID");
```

## Flat InfoCube index, statistics, and space management

The fact table of flat InfoCubes has no primary key or unique constraint and no secondary indexes. Therefore overhead for index management does not occur.

The “Performance” tab of the SAP BW administration window for flat InfoCubes (Figure 8-48) does not contain any options for aggregates because aggregates cannot be defined on flat InfoCubes.

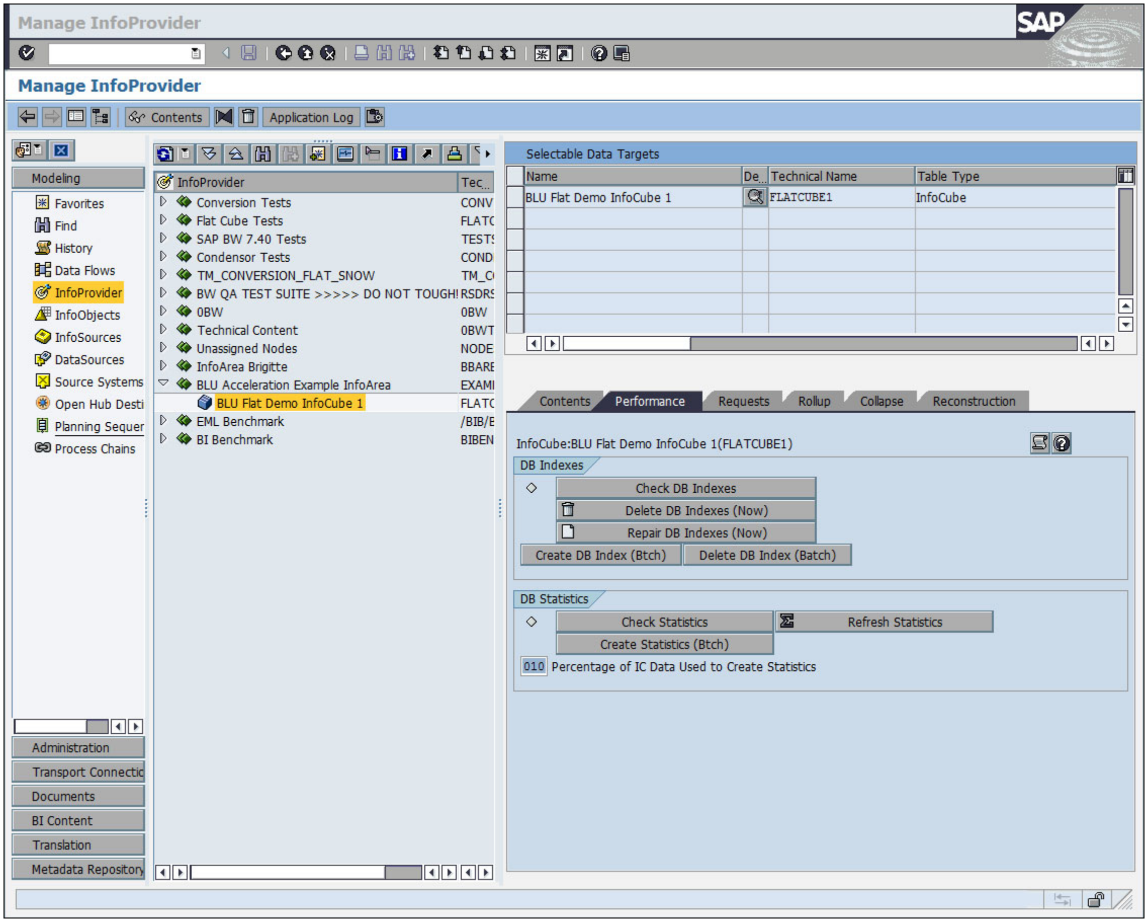



*Figure 8-48 Manage flat InfoCube FLATCUBE1 - Performance tab*

The “DB indexes” options perform no work on flat InfoCubes.

SAP BW transaction RSRV provides an option to check and repair the indexes of an InfoCube. You can run this check also for flat InfoCubes. Figure 8-49 shows “Database indexes of InfoCube and its Aggregates” is run for the BLU Acceleration InfoCube FLATCUBE1.

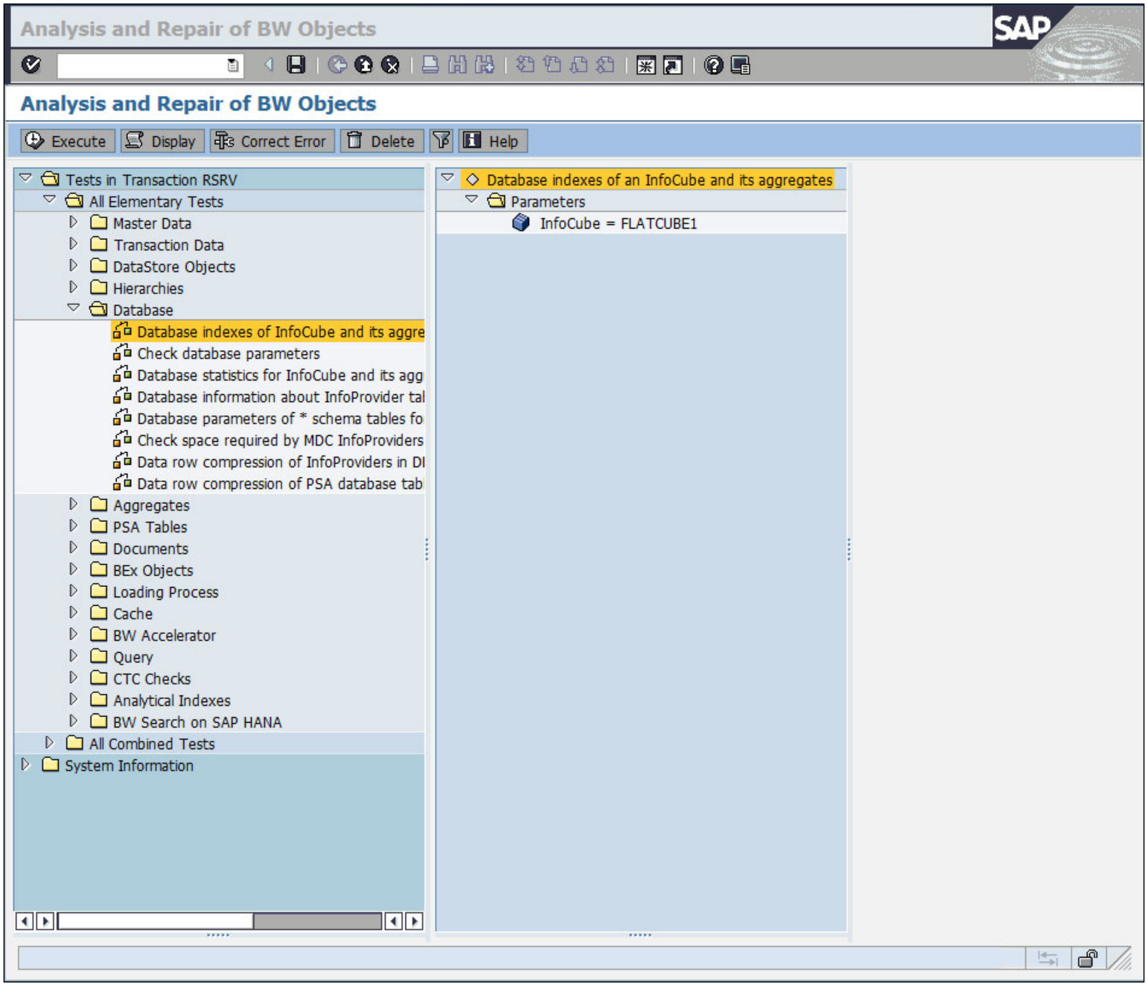



*Figure 8-49 Check database indexes of flat InfoCube FLATCUBE1*

The result is shown in Figure 8-50. Only the unique constraint that is defined on the package dimension table is listed.

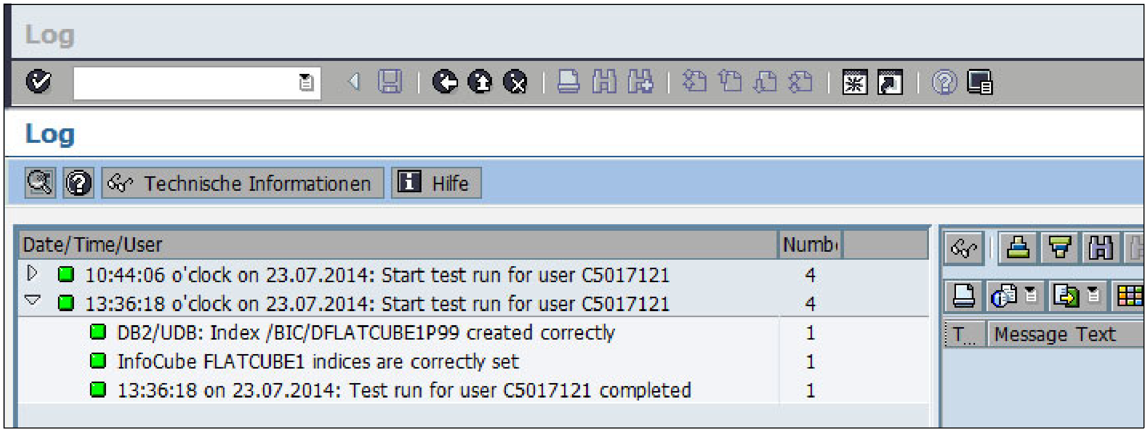



*Figure 8-50 Result of database index check for flat InfoCube FLATCUBE1*

## 8.5.3 Column-Organized DataStore Objects (DSOs) in SAP BW

In this section, we show how to create standard and write-optimized DSOs such that the active table is created as column-organized table. We illustrate the procedure with an example in an SAP BW 7.40 system in which the BI DataMart Benchmark InfoCubes have been installed (InfoArea BI Benchmark). The example works in exactly the same way in the other SAP BW releases in which BLU Acceleration for DSOs is supported.

The creation of DSOs with a column-organized active table is supported as of DB2 for LUW Cancun Release 10.5.0.4 and in SAP BW as of SAP BW 7.0. For detailed information about the required SAP Support Packages and SAP Notes see 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW”.

Creating DSOs with a column-organized active table can have the following advantages:

- Compression of the active table can be improved because column-organized tables usually compress better than row-organized tables.
- Reporting queries on DataStore objects run faster. You can run more queries directly on DataStore objects. This can eliminate the need for creating InfoCubes which saves both storage space and ETL processing time for propagating data from DataStore objects into InfoCubes. For optimal performance of reporting on DSOs, the InfoObjects that occur in the DSOs should also be implemented with column-organized tables (see 8.5.4, “Column-Organized InfoObjects in SAP BW” on page 332).

Both standard and write-optimized DSOs can be created with column-organized active tables. The activation queue and change log tables of standard DSOs are still created row-organized.

You can convert existing DSOs with row-organized active tables to column-organized with DB6CONV (see 8.6.1, “Conversion of SAP BW objects to column-organized tables with DB2 Cancun Release 10.5.0.4 and DB6CONV V6” on page 351).

## Creating column-organized standard DSO

We create a new standard DSO from a DataSource as follows:

1. Call transaction RSA1 and select DataSources from the Modelling window. Open the “Benchmark” folder, select DataSource **BENCH_DTP_CUBE1** and call up **Generate Data Flow** from the context menu (Figure 8-51).

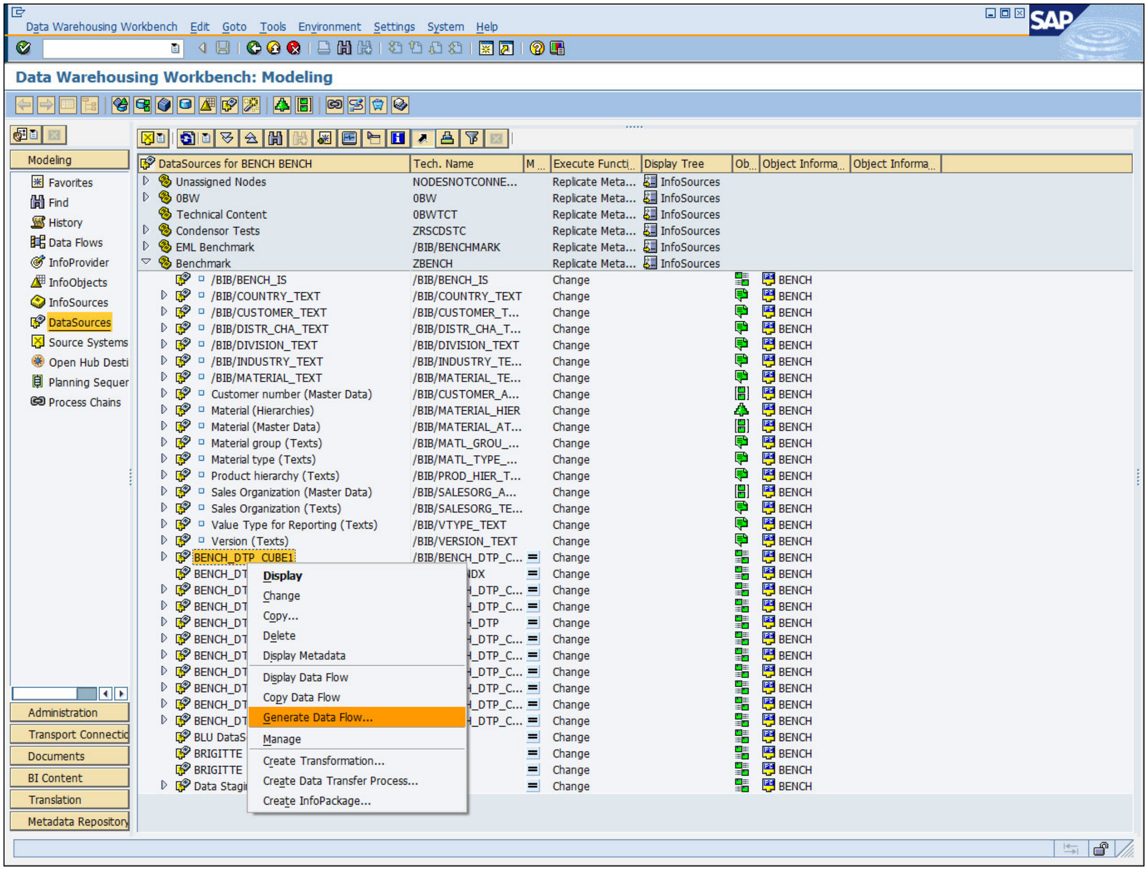



*Figure 8-51 Generate new DSO from DataSource with Data Flow Generation*

2. In the Data Flow Generation Wizard, select **DataStore Object** as InfoProvider type, **Standard** as Subtype, enter **BLUDSO1** as InfoProvider name, **BLU DEMO DSO 1** as description, and **EXAMPLE** as InfoArea, and choose **Continue** (Figure 8-52).

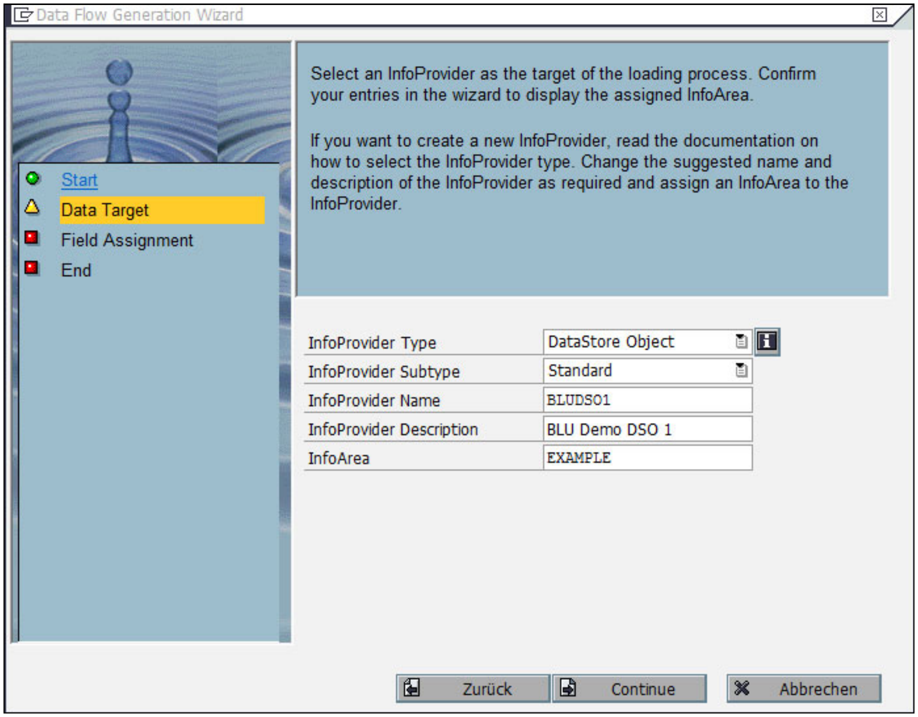



*Figure 8-52 Define new target DSO of Data Flow*

3. In the Field Assignment window, choose **Suggestion** (Figure 8-53).

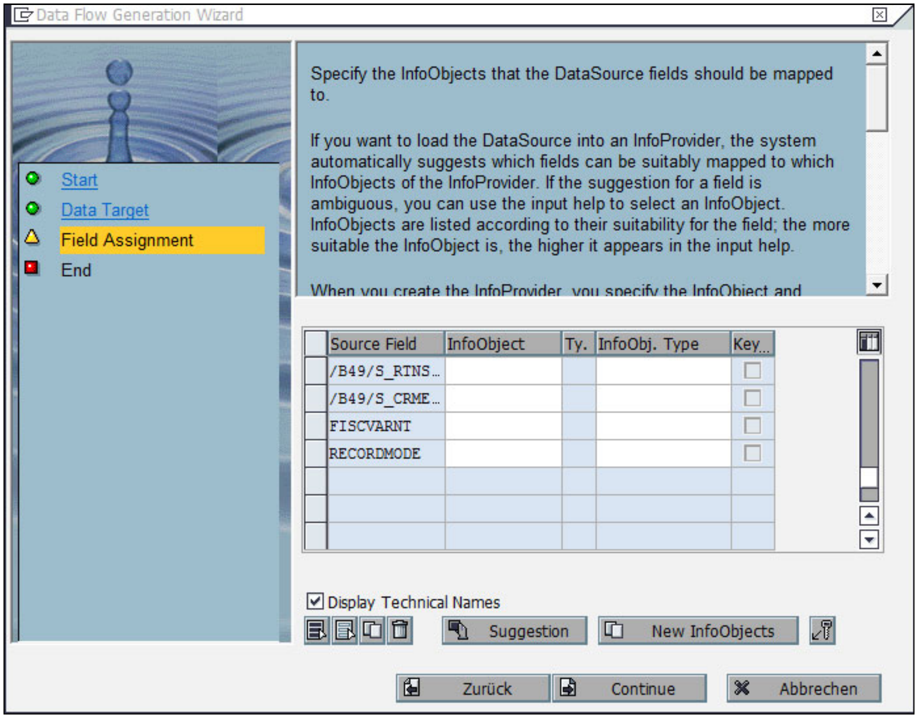



*Figure 8-53 Field assignment window of Data Flow Generation wizard*

4. Choose **Continue** when the InfoObject and InfoObject Type columns are filled (Figure 8-54).

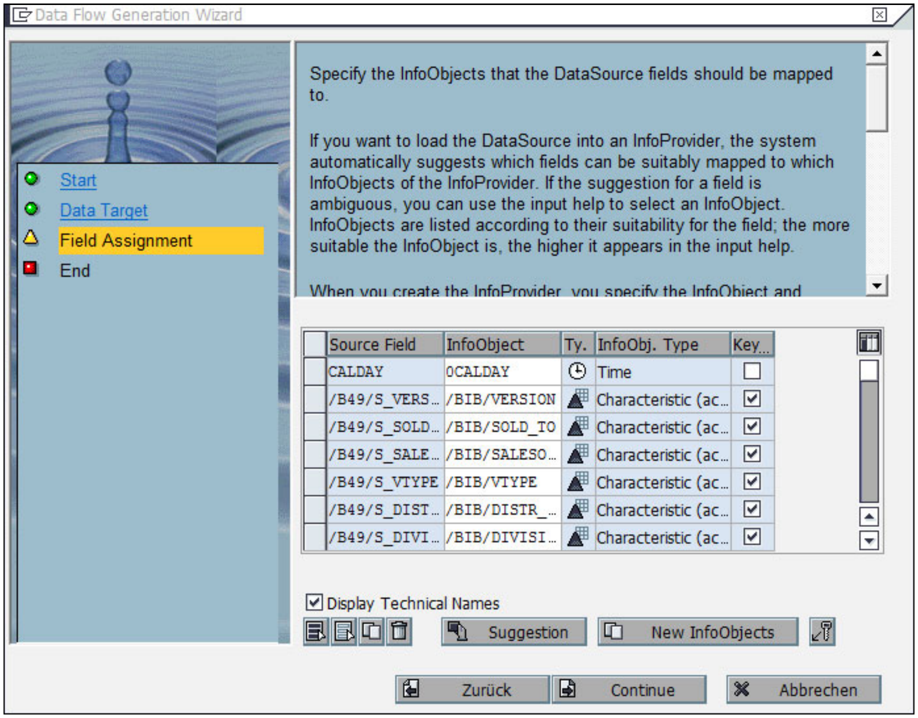



*Figure 8-54 Define InfoObects of new DSO in Data Flow Generation wizard*

5. In the “End” window, choose **Complete** (Figure 8-55).

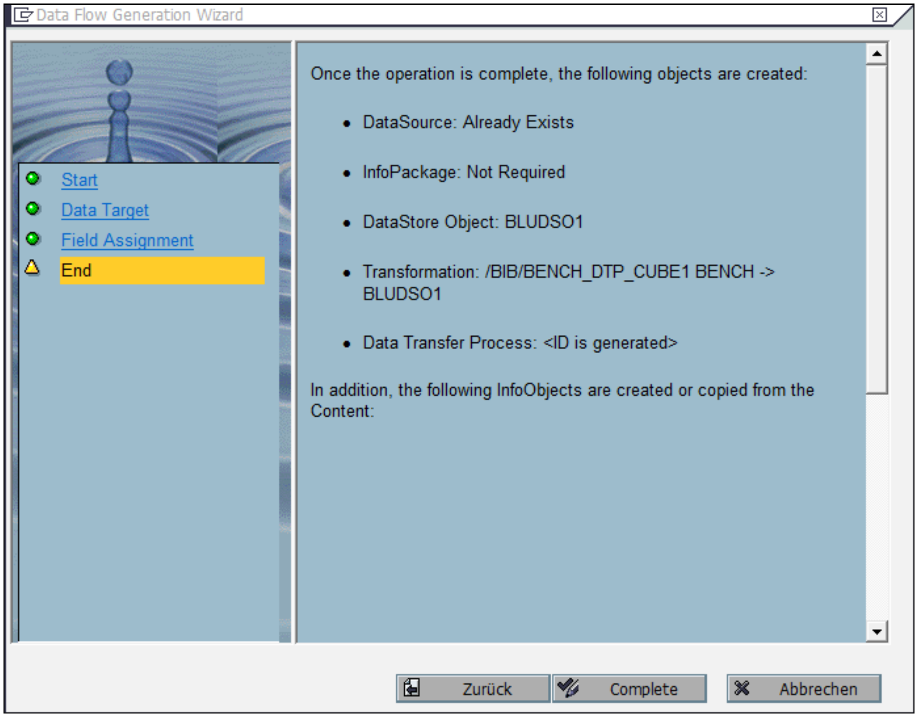



*Figure 8-55 Final window of Data Flow Generation wizard*

6. In transaction RSA1, select InfoProvider from the Modelling window. In the "EXAMPLE InfoArea", you find the newly created standard DSO. Open the DSO in Edit mode (Figure 8-56).

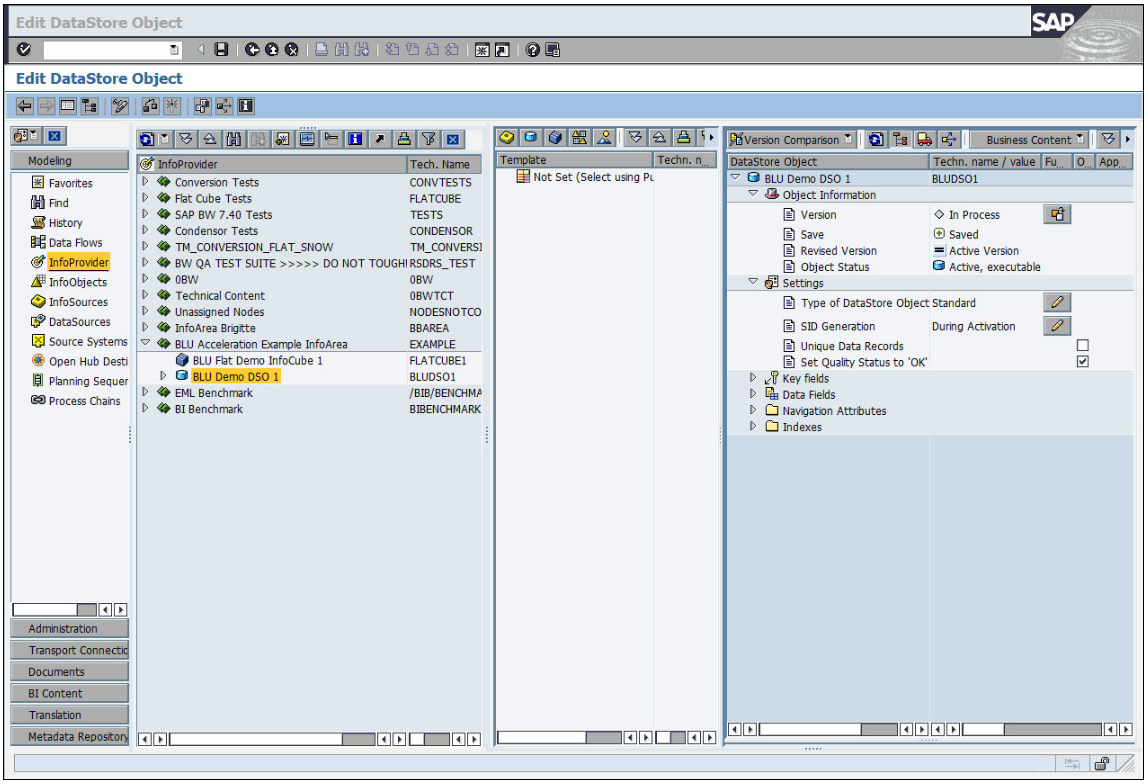


*Figure 8-56 Edit standard DSO*

7. Select **Extras** → **DB Performance** → **Clustering** from the menu (Figure 8-57).

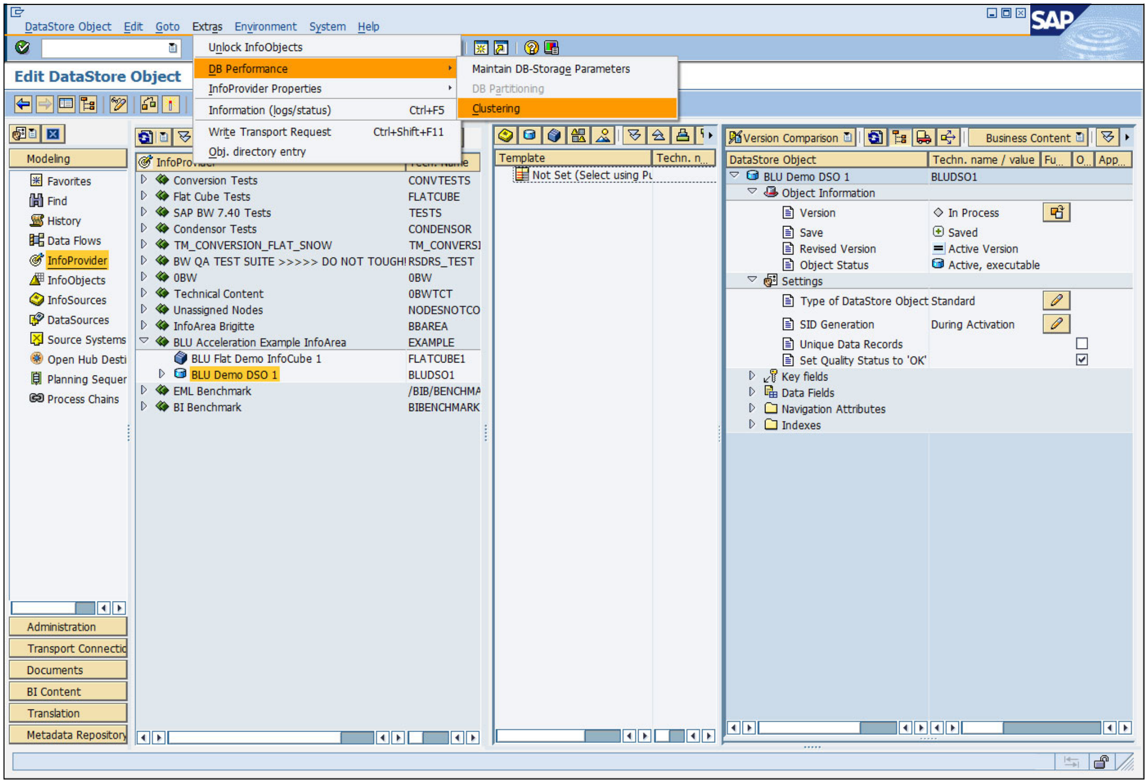



*Figure 8-57 Define clustering for standard DSO*

8. In the Clustering window, choose **Column-Organized** (Figure 8-58).

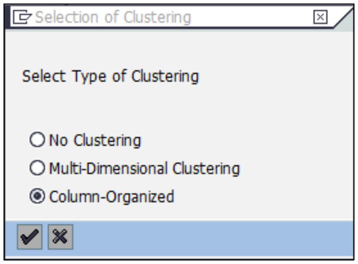



*Figure 8-58 Clustering window of standard DSO*

9. Save and activate the DSO. Then select **Extras** → **Information (logs/status)** from the menu (Figure 8-59 on page 324).

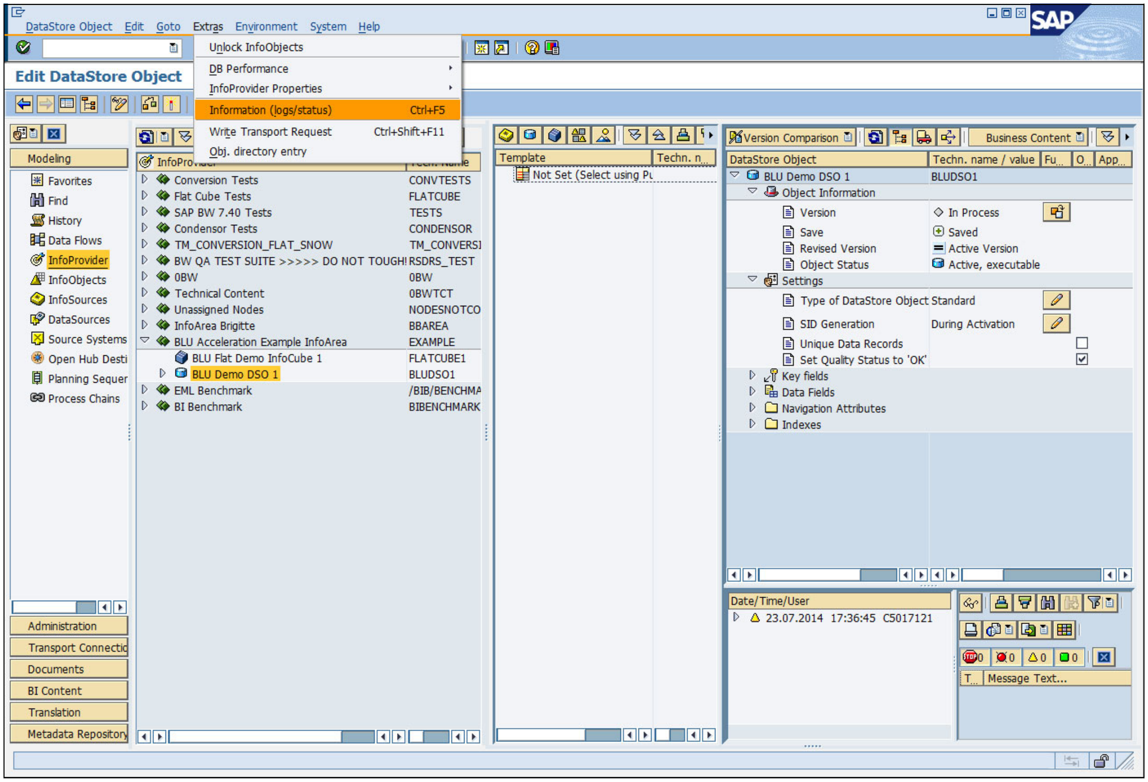



*Figure 8-59 Show database tables of standard DSO*

10. In the Info Selection window, choose **Dictionary/DB status** (Figure 8-60).

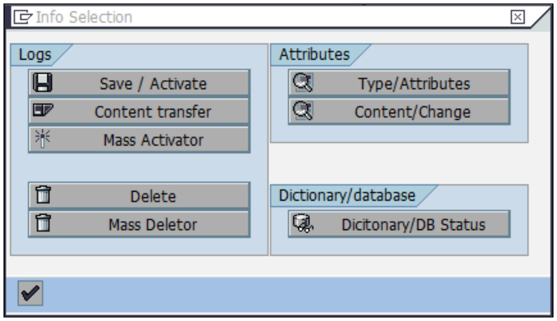



*Figure 8-60 Info Selection window for standard DSO*

11. In the Status information window, double-click the name of the “Active Table” **/BIC/ABLUDSO100** (Figure 8-61) to get to the SAP Data Dictionary.

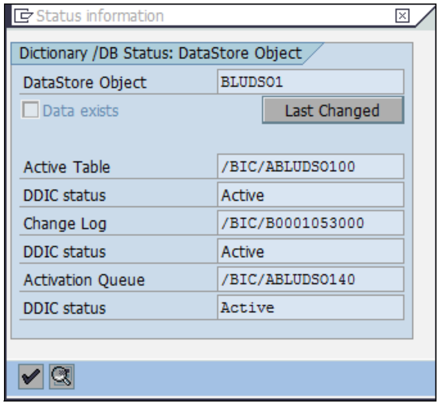



*Figure 8-61 Database tables of standard DSO*

In the SAP Data Dictionary, look up the storage parameters of the table as described in step 2 on page 262. The storage parameters show that the table is column-organized (Figure 8-62).

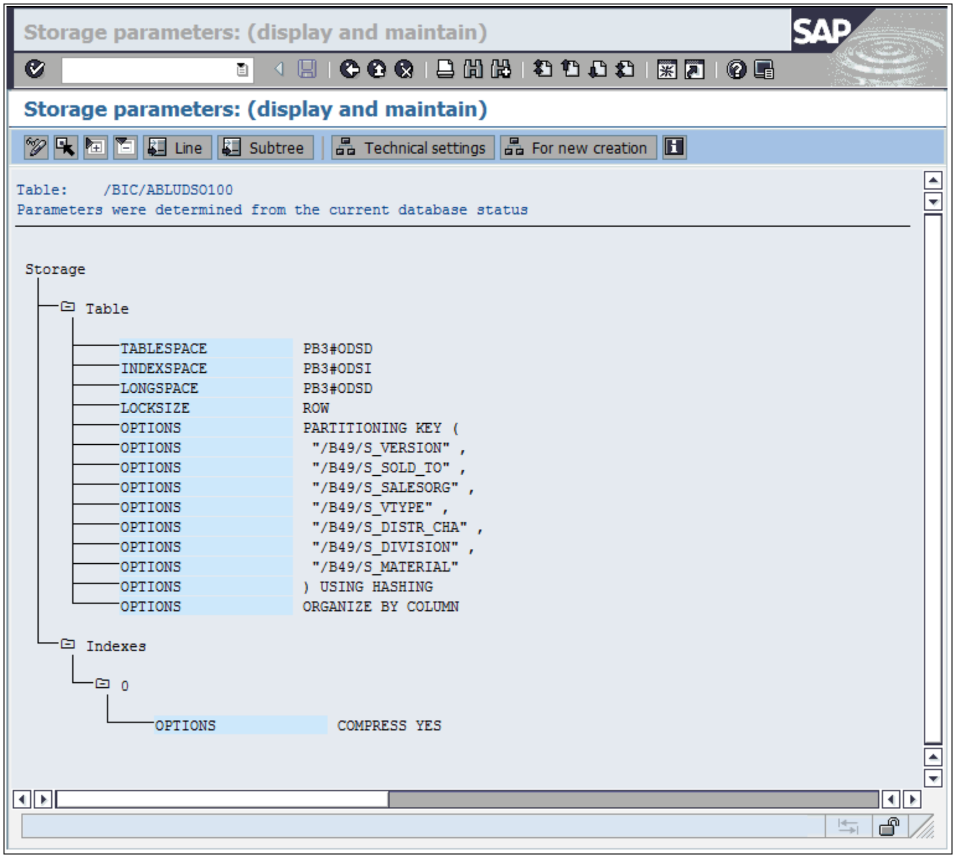



*Figure 8-62 Storage parameters of BLU standard DSO active table*

## Creating column-organized write-optimized DSO

We create a write-optimized DSO by copying the standard DSO BLUDSO1 that we created in the previous chapter as follows:

1. In transaction RSA1, select standard DSO **BLUDSO1** and choose **Copy...** from the context menu (Figure 8-63 on page 327).

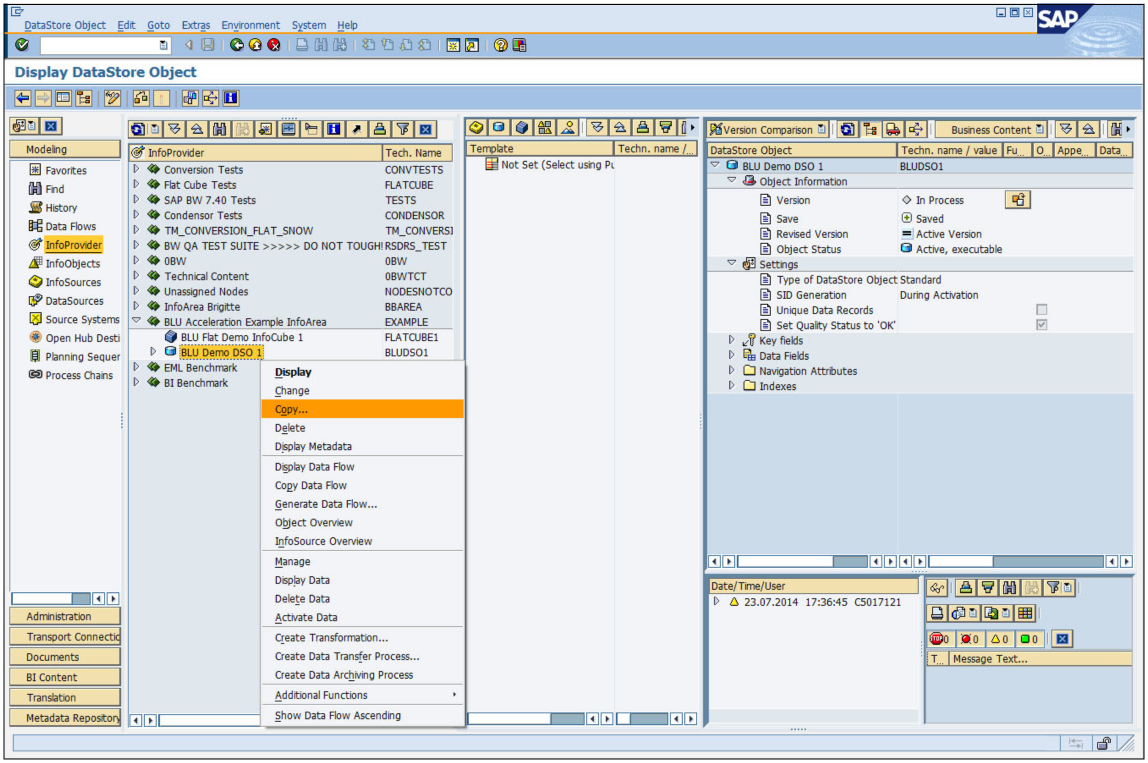



*Figure 8-63 Copy DSO*

2. In the Create DataStore Object window enter `BLUWDSO1` as technical name and "BLU Demo write-optimized DSO 1" as description and click the **Create** icon (Figure 8-64).

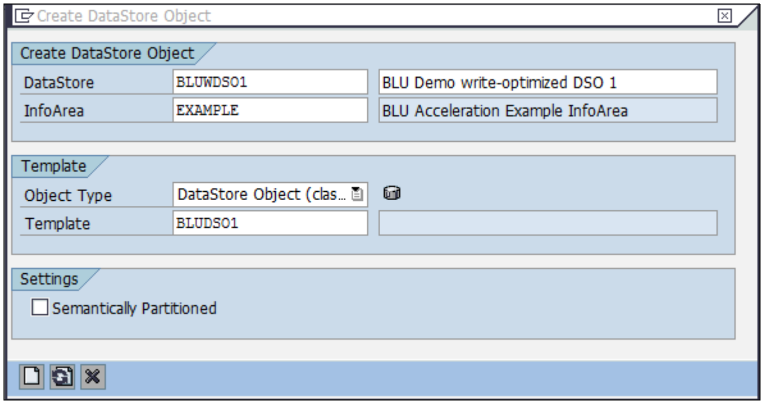



*Figure 8-64 Create DataStore Object window*

3. In the “Edit DataStore Object” window change the type of the DSO in the “Settings” folder from “Standard” to “Write Optimized” (Figure 8-65).

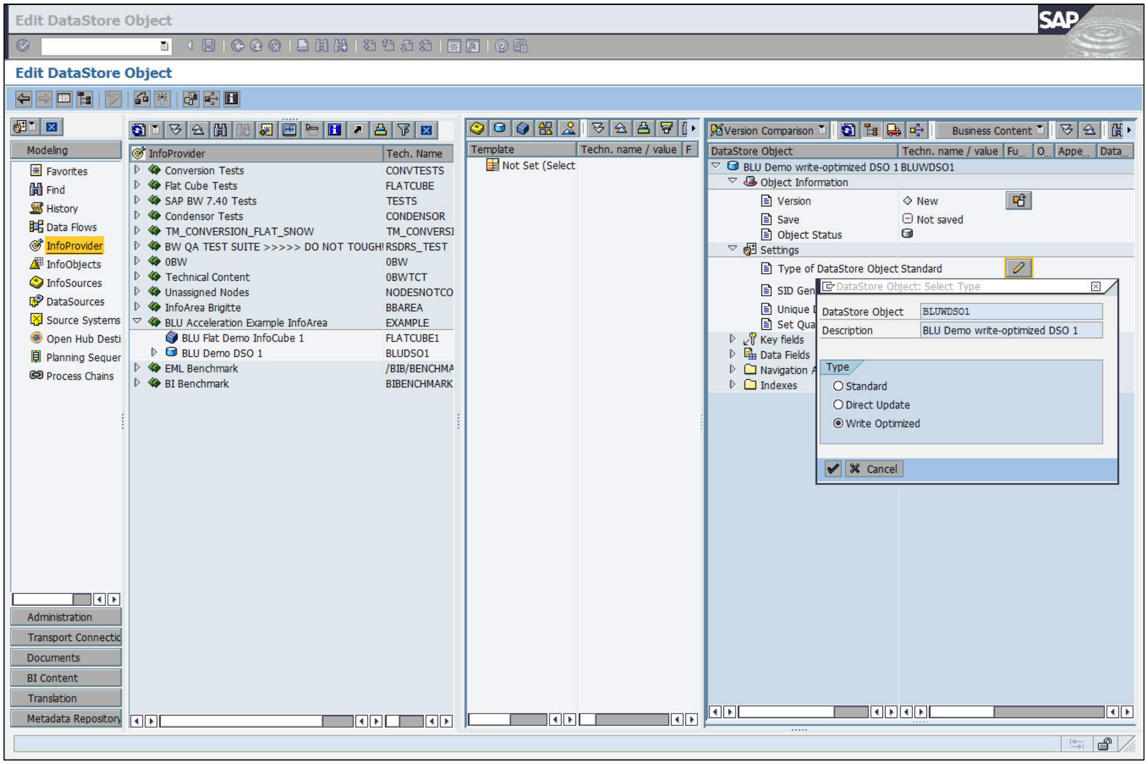


*Figure 8-65 Change DSO type to write-optimized from standard*

4. Choose **Extras** → **DB Performance** → **Clustering** from the menu (Figure 8-66).

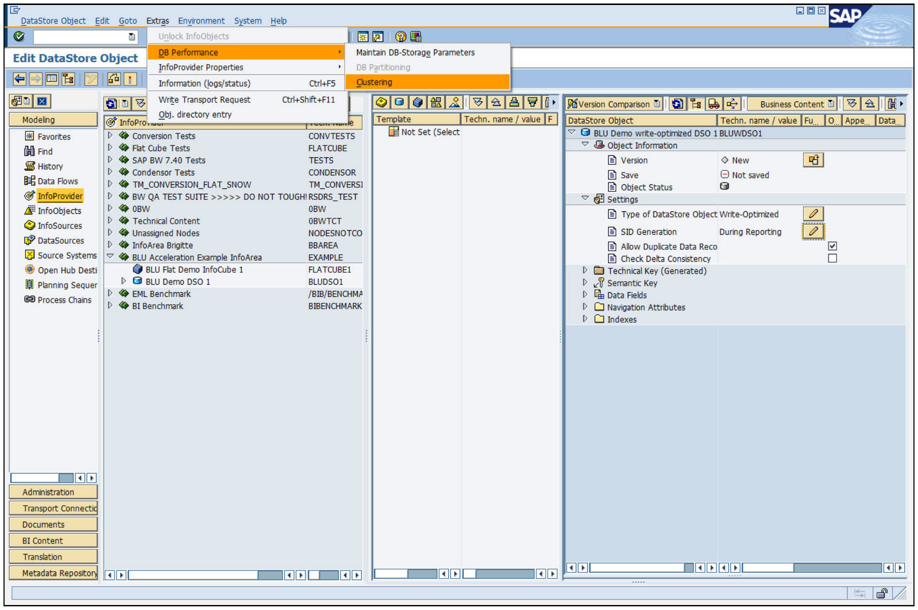



*Figure 8-66 Define clustering for write-optimized DSOs*

5. In the “Selection of Clustering” window, select **Column-Organized** (Figure 8-67).

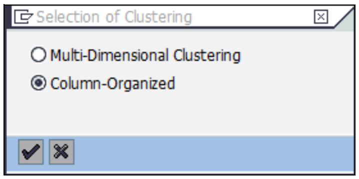



*Figure 8-67 Clustering window for write-optimized DSOs*

6. Save and activate the write-optimized DSO and select **Extras** → **Information (logs/status)** from the menu (Figure 8-68 on page 330).

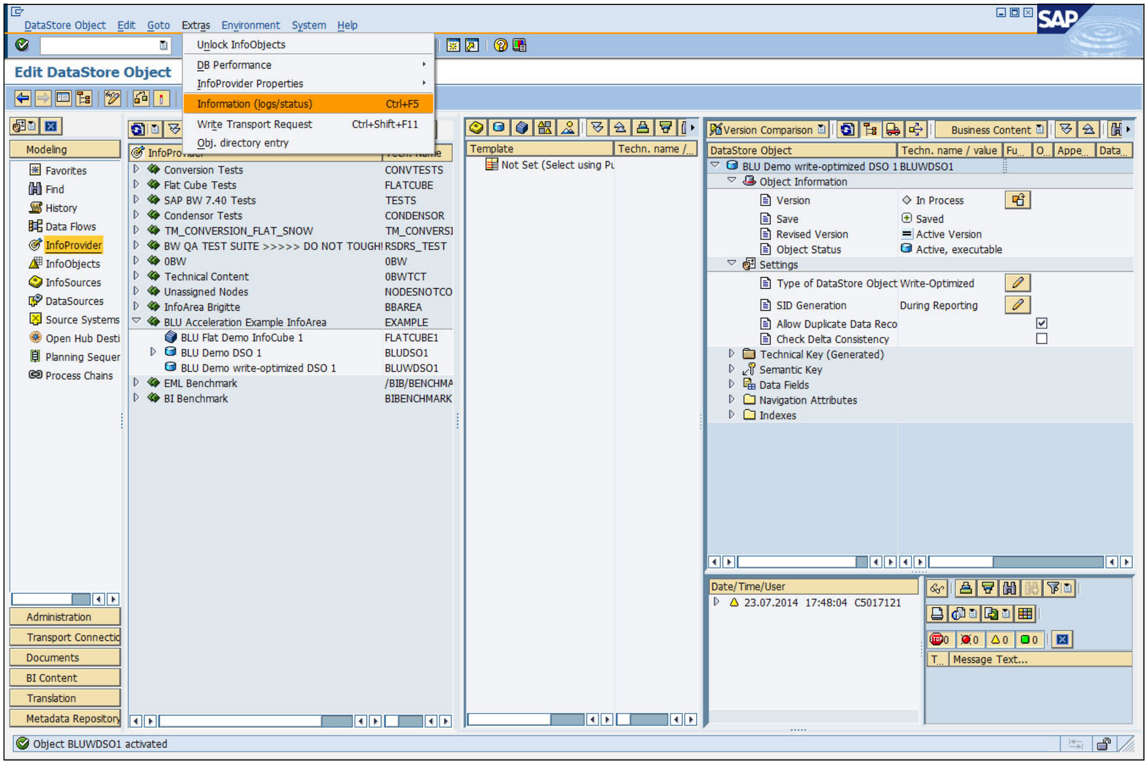



*Figure 8-68 Prepare to display database tables of write-optimized DSO*

7. In the “Info Selection” window choose **Dictionary/DB Status** (Figure 8-69).

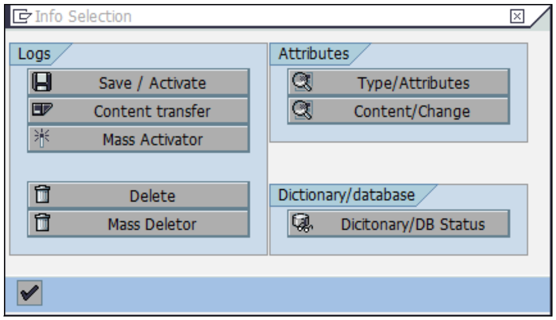



*Figure 8-69 Info Selection window for write-optimized DSO*

8. In the “Status information” window double-click the name of the “Active Table” to get to the SAP Data Dictionary (Figure 8-70 on page 331).

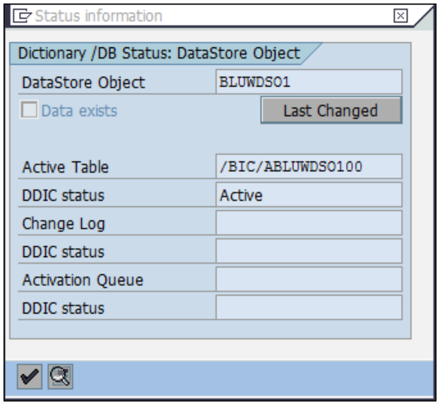



*Figure 8-70 Overview of database tables of write-optimized DSO*

9. Look up the storage parameters for the table as described in step 2 on page 262. The storage parameters show that the table is column-organized (Figure 8-71 on page 331).

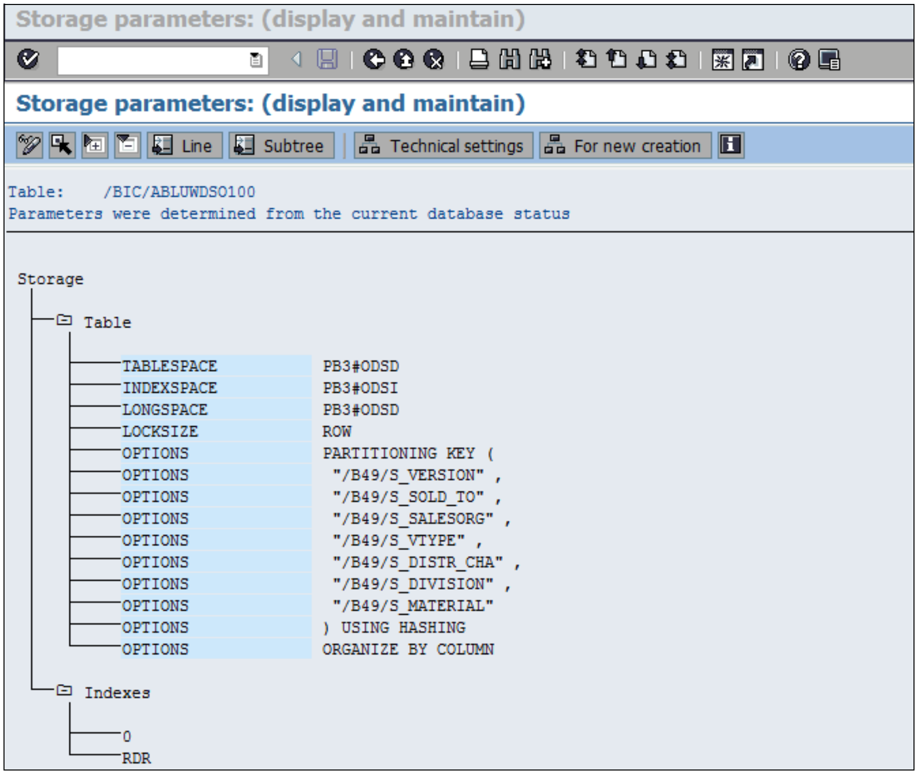



*Figure 8-71 Storage parameters of column-organized write-optimized DSO*

### 8.5.4 Column-Organized InfoObjects in SAP BW

In this section, we show how to create InfoObjects, such as the InfoObject tables that occur in SAP BW reporting queries, as column-organized tables.

We illustrate the procedure with an example in an SAP BW 7.40 system in which the BI DataMart Benchmark InfoCubes have been installed (InfoArea BI Benchmark). The example works in exactly the same way in the other SAP BW releases in which BLU Acceleration is supported.

The creation of InfoObjects with column-organized tables is supported as of DB2 for LUW Cancun Release 10.5.0.4 and in SAP BW as of SAP BW 7.0. For detailed information about the required SAP Support Packages, see 8.2, "Prerequisites and restrictions for using BLU Acceleration in SAP BW".

Creating InfoObjects with column-organized tables can improve query performance when the InfoObject tables are joined with column-organized InfoCube or DSO tables. You can convert existing InfoObjects that use row-organized tables to column-organized tables with DB6CONV (see 8.6.1, "Conversion of SAP BW objects to column-organized tables with DB2 Cancun Release 10.5.0.4 and DB6CONV V6" on page 351).

From the tables that are created for an InfoObject, only the SID table (S table) and the time-dependent and time-independent attribute SID tables (X and Y tables) are created as column-organized tables. These are the tables that frequently occur in SAP BW reporting queries. The text table, the time-dependent and time independent attribute tables, and the hierarchy tables remain row-organized.

By default, SAP BW InfoObjects are created with row-organized tables only. SAP BW provides the RSADMIN configuration parameter, DB6_IOBJ_USE_CDE, to create SID and attribute SID tables as column-organized tables. You set this parameter as follows:

1. Call transaction **SE38** and run the SAP_RSADMIN_MAINTAIN report.
2. In the "OBJECT" field, enter `DB6_IOBJ_USE_CDE`. In the "VALUE" field, enter `YES`, and then click the **Execute** icon (Figure 8-72).

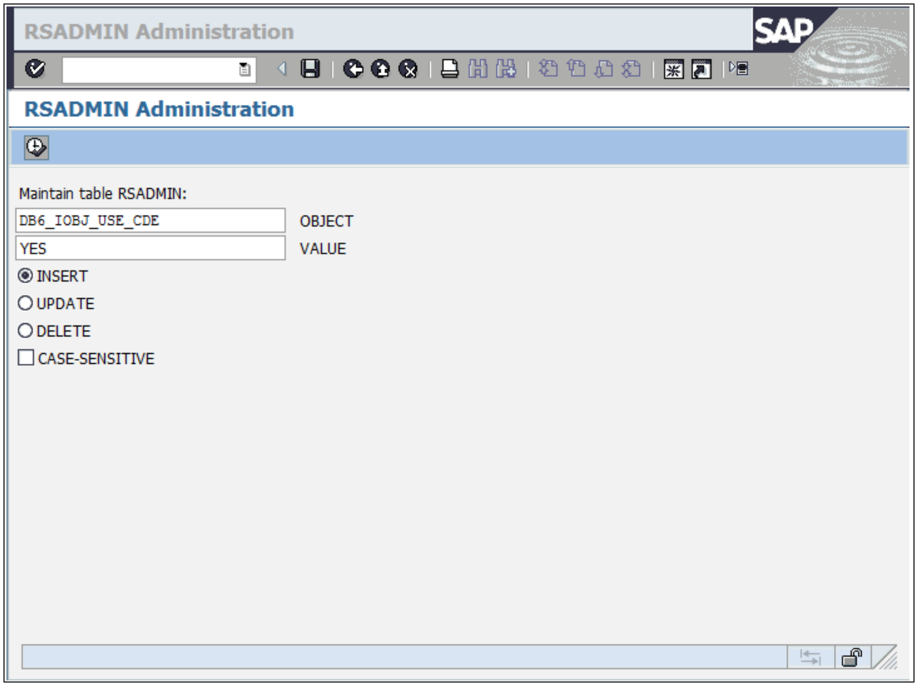



*Figure 8-72 Report SAP_RSADMIN_MAINTAIN: Set option to create SAP BW InfoObject tables column-organized*

The information shown in Figure 8-73 is displayed.

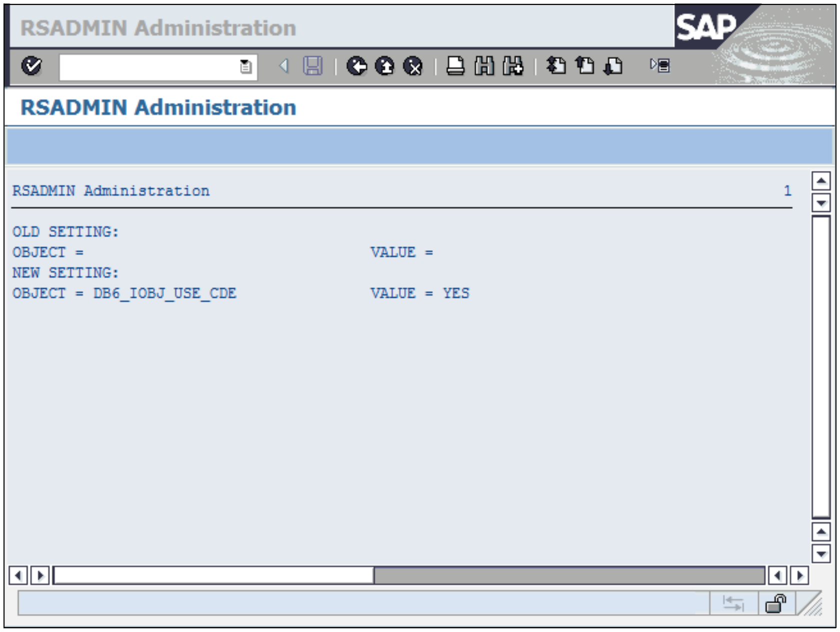



*Figure 8-73 Report SAP_RSADMIN_MAINTAIN: Result of setting the option to create SAP BW InfoObject tables column-organized*

Now create a copy of the `/BIB/MATERIAL` InfoObject as follows:

1. Call transaction RSD1.
2. Select **Characteristic** and enter **`BLUMATL`** as the InfoObject name (see Figure 8-74).

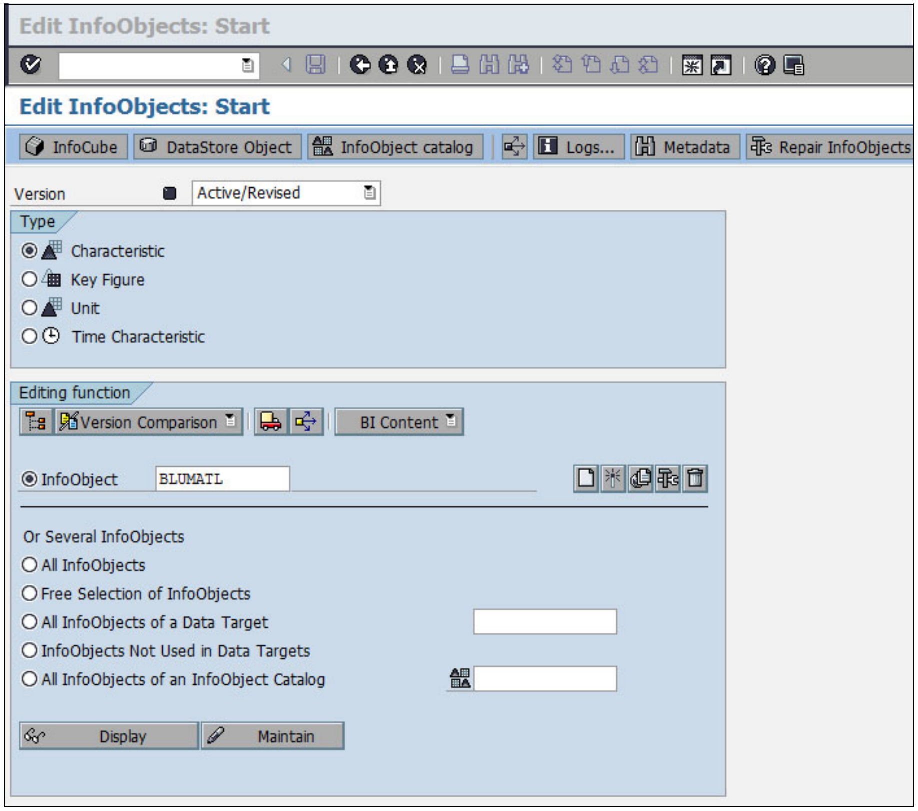


Figure 8-74 Edit InfoObjects: Start window

3. Click the **Copy** icon. In the pop-up window (Figure 8-75) enter “BLU Material InfoObject” as Long Description and `/BIB/MATERIAL` in the “Template” entry field.

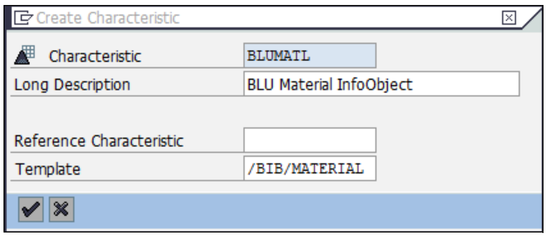


Figure 8-75 Create Characteristic window

4. The “Create Characteristic BLUMATL: Detail” window is displayed (see Figure 8-76). Save and activate the InfoObject.

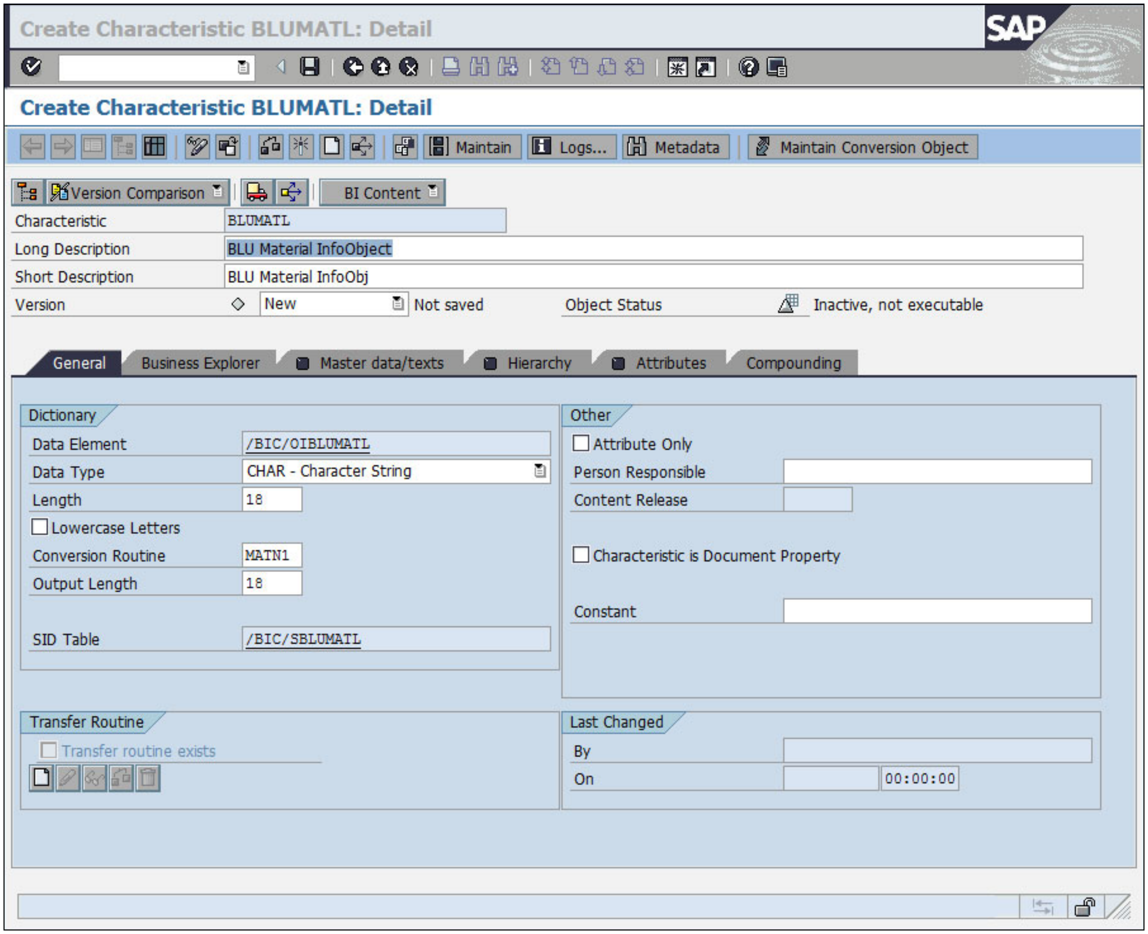



*Figure 8-76 Create BLU Material InfoObject Detail window*

5. Select **Extras** → **Database tables** → **Status...** from the menu (Figure 8-77).

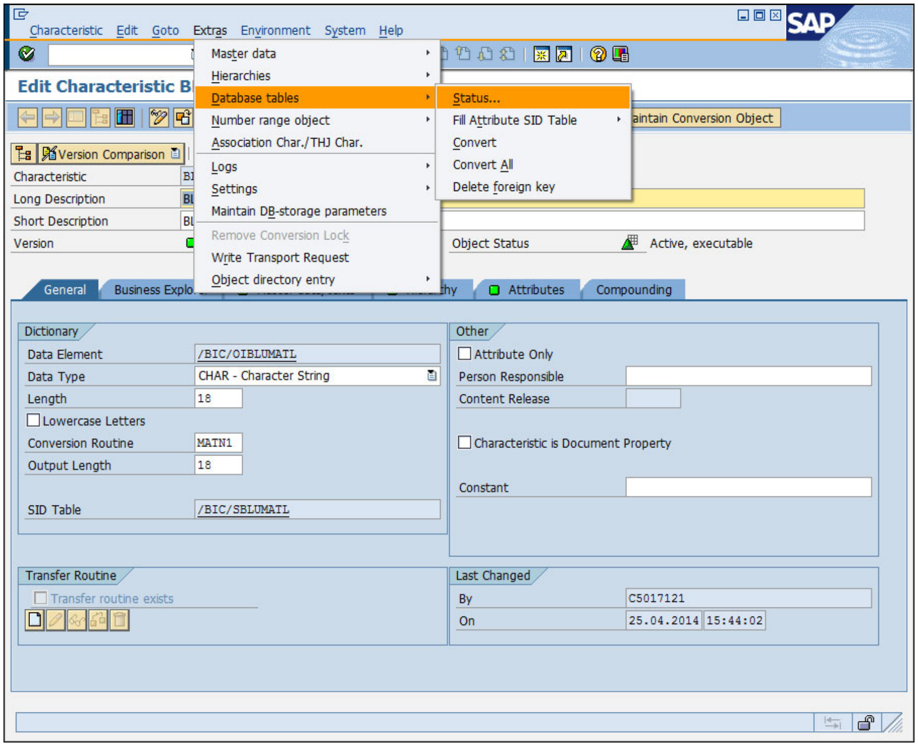


*Figure 8-77 BLU Material InfoObject Edit window*

Information about the tables created for the InfoObject is displayed (Figure 8-78).

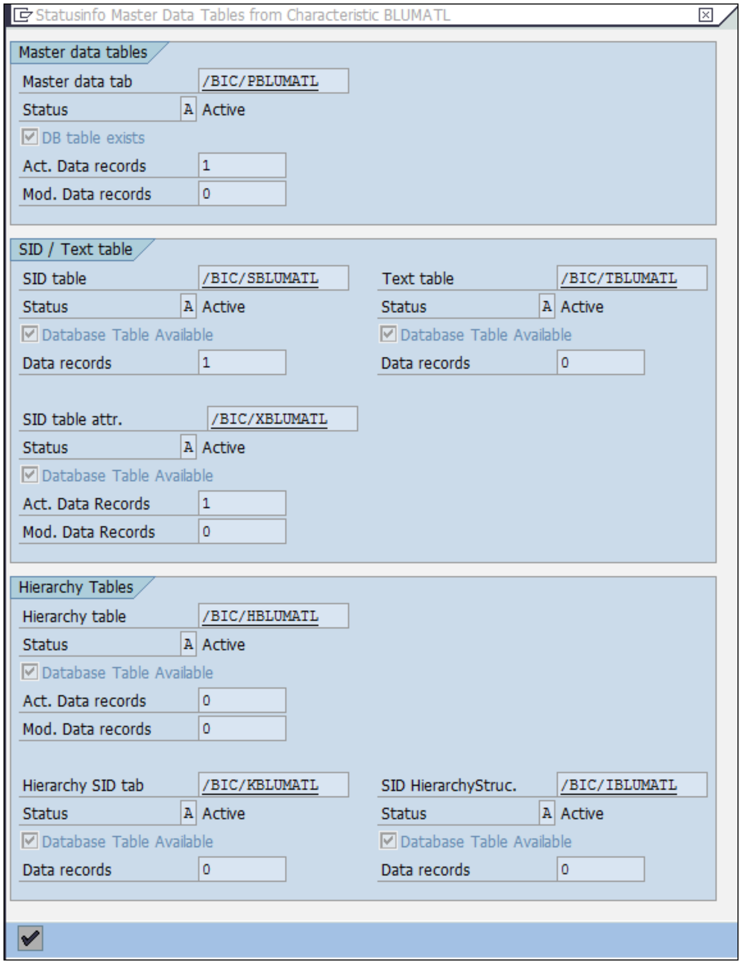



*Figure 8-78 Overview of tables of BLU Material InfoObject*

6. Double-click the SID table name **/BIC/SBLUMATL**. In the SAP Data Dictionary look up the storage parameters with the Database Utility. In the storage parameters you see that the table is column-organized (Figure 8-79).

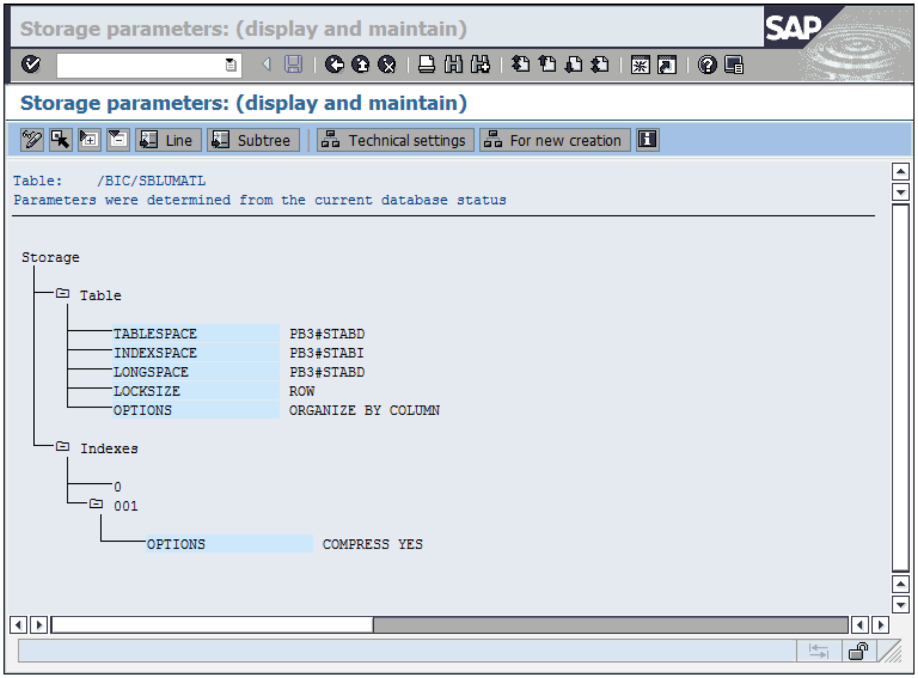



*Figure 8-79 Storage parameters of BLU Material SID table*

7. Repeat step 6 on page 339 for the tables `/BIC/XBLUMATL` and `/BIC/PBLUMATL`. You see that table `/BIC/XBLUMATL` is also column-organized (Figure 8-81) and that table `/BIC/PBLUMATL` is row-organized (Figure 8-80). All other tables of the InfoObject are also row-organized.

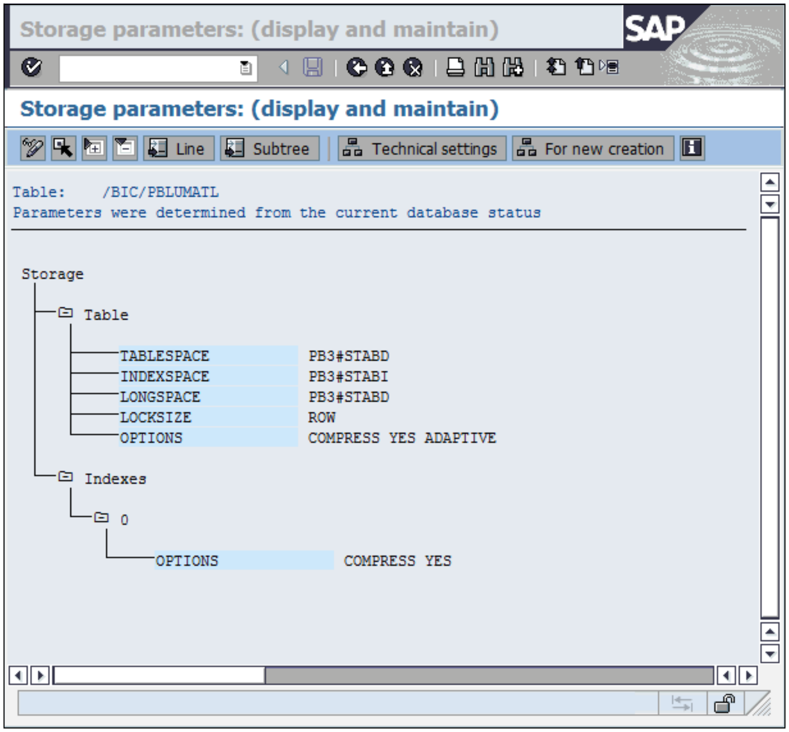



*Figure 8-80 Storage parameters of BLU Material display attribute table*

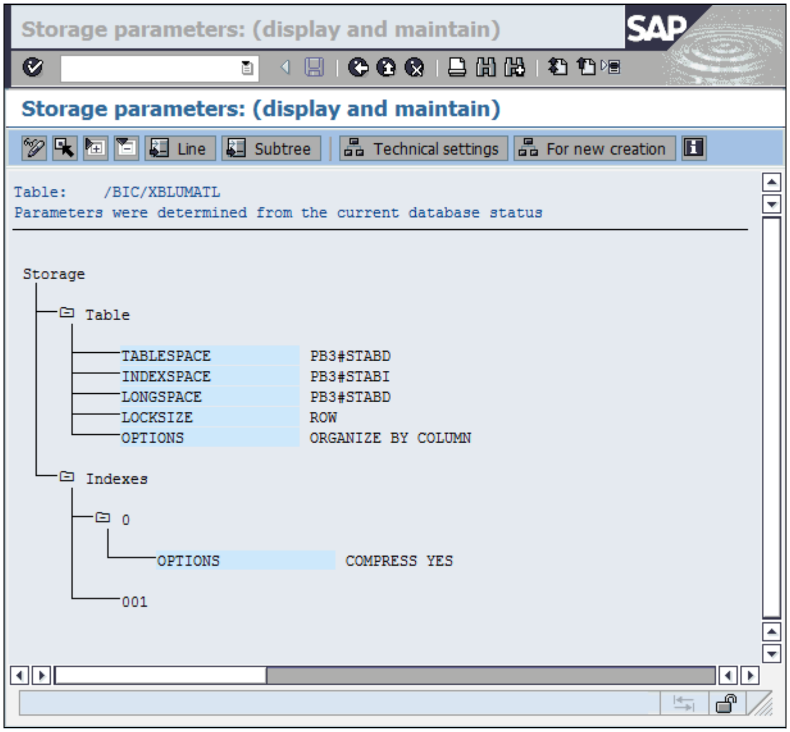



*Figure 8-81 Storage parameters of BLU Material attribute SID table*

## 8.5.5 Column-organized PSA tables in SAP BW

In this section, we show how to create PSA tables as column-organized tables.

We illustrate the procedure with an example in an SAP BW 7.40 system in which the EML Benchmark InfoCubes and DataStore objects have been installed (InfoArea EML Benchmark). The example works in exactly the same way in the other SAP BW releases in which BLU Acceleration is supported.

The creation of column-organized PSA tables is supported as of DB2 for LUW Cancun Release 10.5.0.4 and in SAP BW as of SAP BW 7.0. For detailed information about the required SAP Support Packages, see 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW”.

Creating PSA tables as column-organized tables might improve compression because column-organized tables usually compress better than row-organized tables. You can convert existing row-organized PSA tables to column-organized tables with DB6CONV (see 8.6.1, “Conversion of SAP BW objects to column-organized tables with DB2 Cancun Release 10.5.0.4 and DB6CONV V6” on page 351).

By default, PSA tables are created row-organized. SAP BW provides the RSADMIN configuration parameter `DB6_PSA_USE_CDE` to create PSA tables as column-organized tables. You set this parameter as follows:

1. Call transaction **SE38** and run the SAP_RSADMIN_MAINTAIN report.
2. In the “OBJECT” field, enter `DB6_PSA_USE_CDE`. In the “VALUE” field, enter `YES`, and then click the **Execute** icon (Figure 8-82).

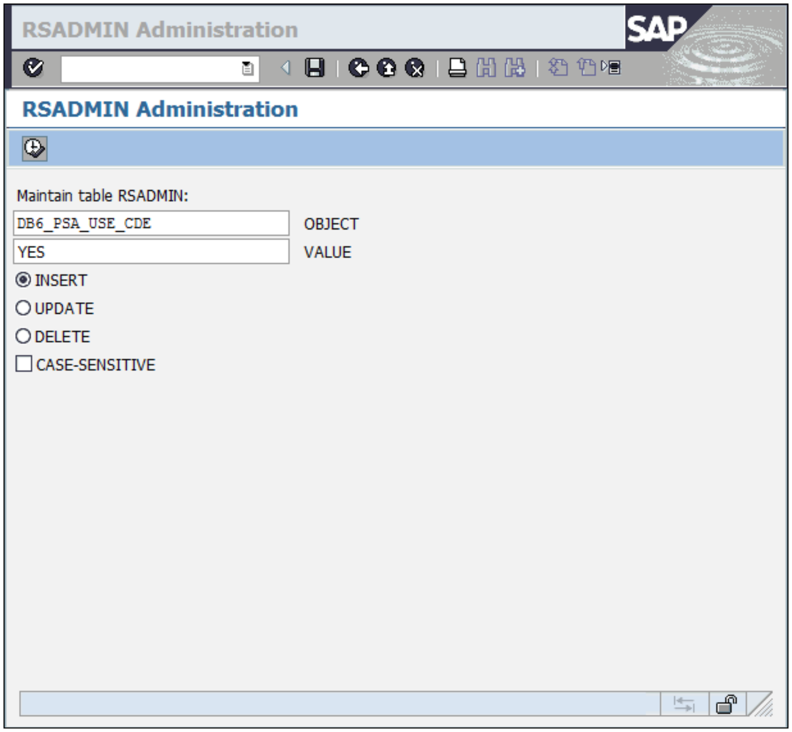



*Figure 8-82 Report SAP_RSADMIN_MAINTAIN: Set option to create SAP BW PSA tables column-organized*

The information shown in Figure 8-83 is displayed.

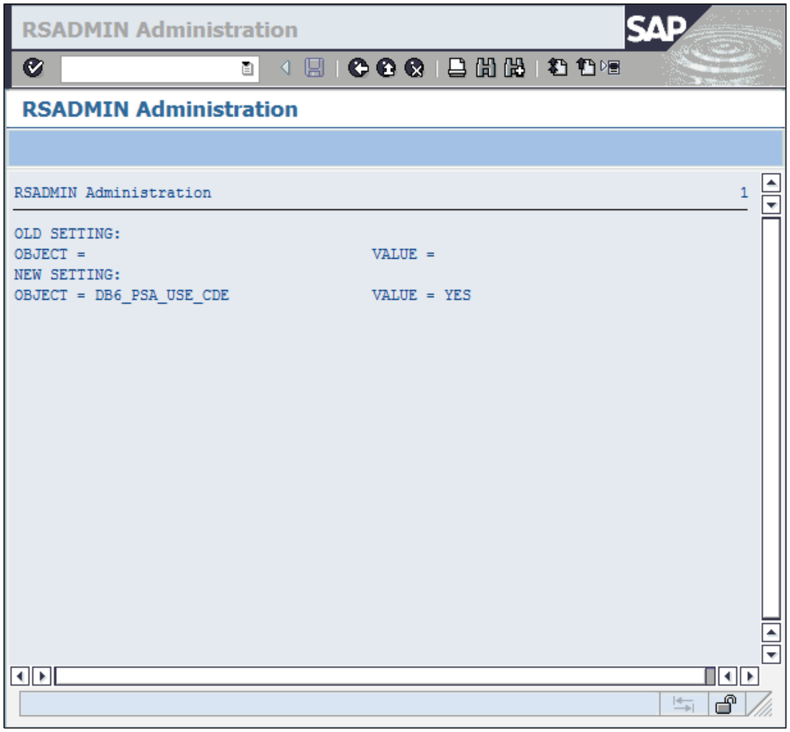



*Figure 8-83 Report SAP_RSADMIN_MAINTAIN: Result of setting the option to create SAP BW PSA tables column-organized*

Now you can create a new DataSource as follows:

1. Call transaction RSA1 and select **DataSources** in the Modelling window. Open the **Benchmark** folder, select DataSource **BENCH_DTP_CUBE1**, and choose **Copy** from the context menu (Figure 8-84).

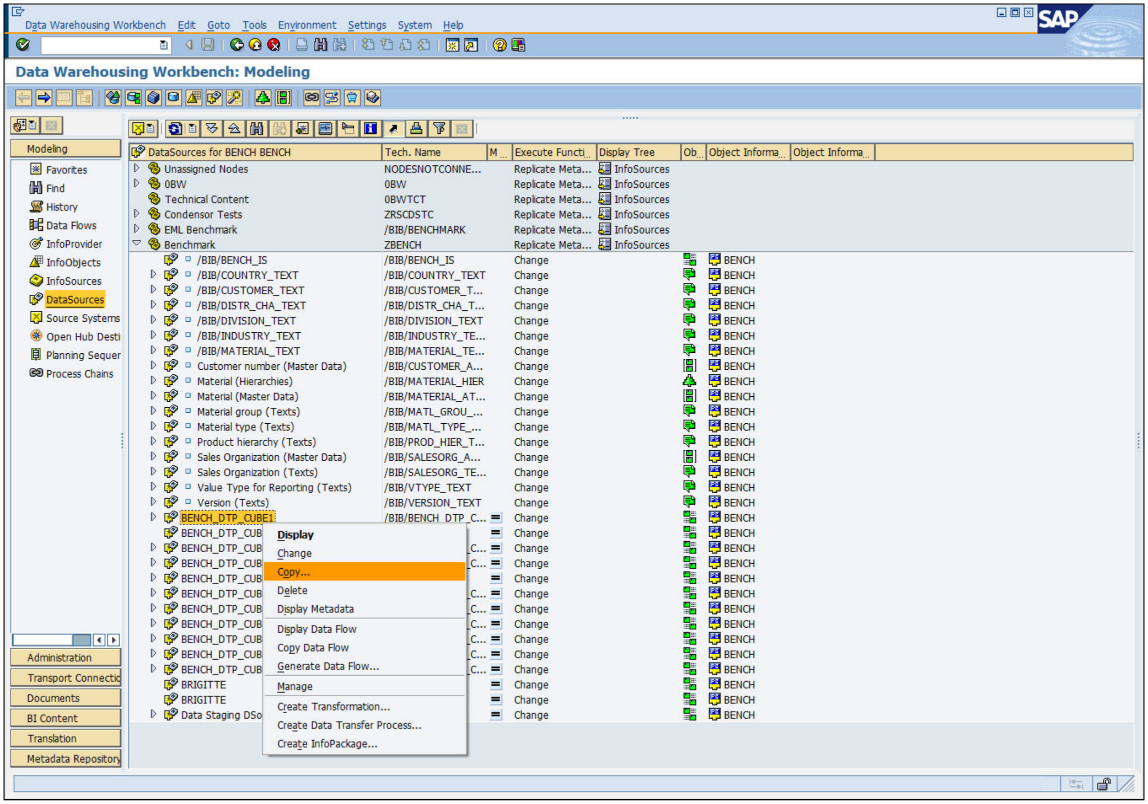



*Figure 8-84 Copy an existing SAP BW DataSource*

2. In the Target frame enter `BLUDS1` as “DataSource” name and `BENCH` as “Source system” and press **Enter** (Figure 8-85).

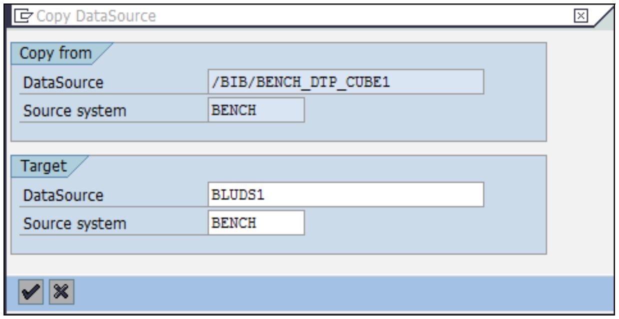



*Figure 8-85 Copy DataSource window*

Enter “**BLU DataSource 1**” as short, medium, and long description and save and activate the DataSource (Figure 8-86).

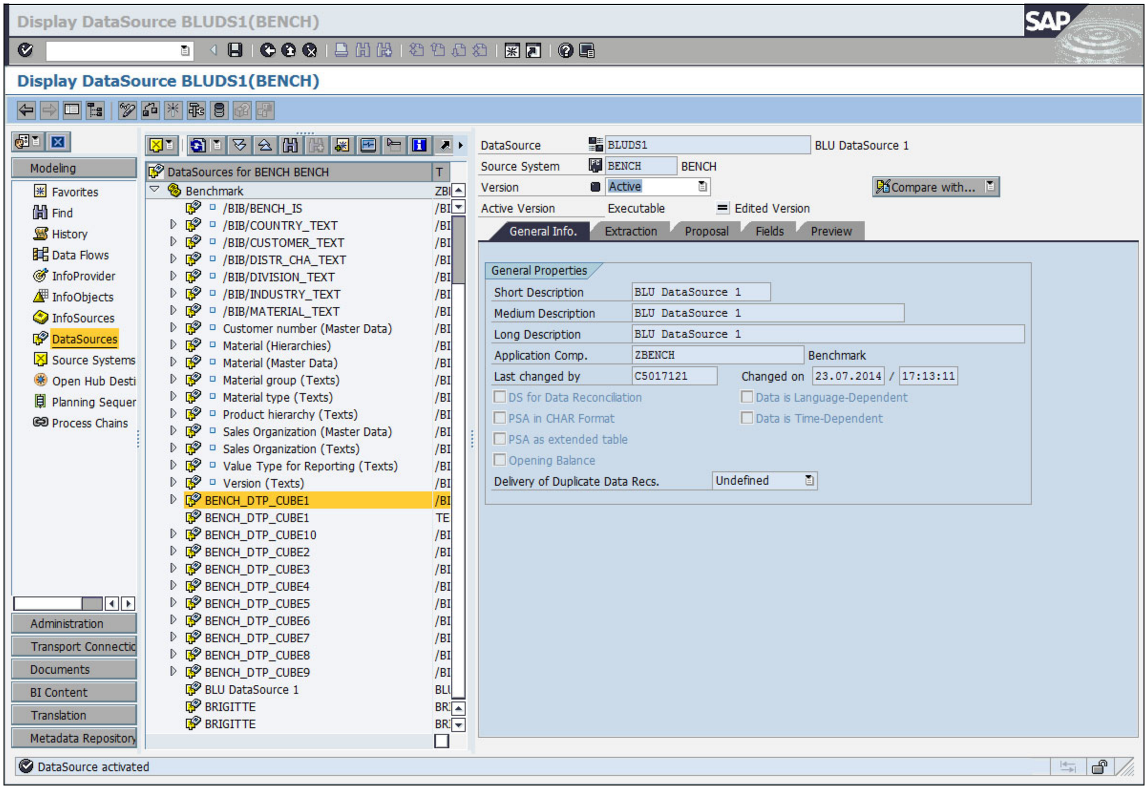



*Figure 8-86 Properties of an SAP BW DataSource*

3. Select **Goto** → **Technical Attributes** from the menu. The technical attributes window is shown (Figure 8-87).

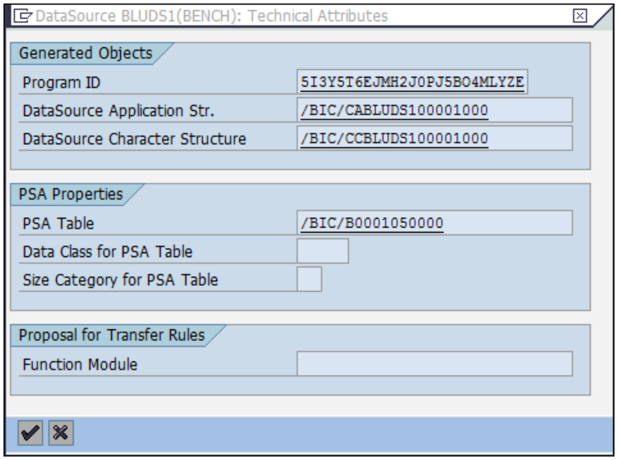



*Figure 8-87 Technical Attributes of an SAP BW DataSource*

4. Double-click the PSA table to get to the SAP Data Dictionary and look up the storage parameters of the table as described in step 2 on page 262. The storage parameters show that the table is column-organized (Figure 8-88).

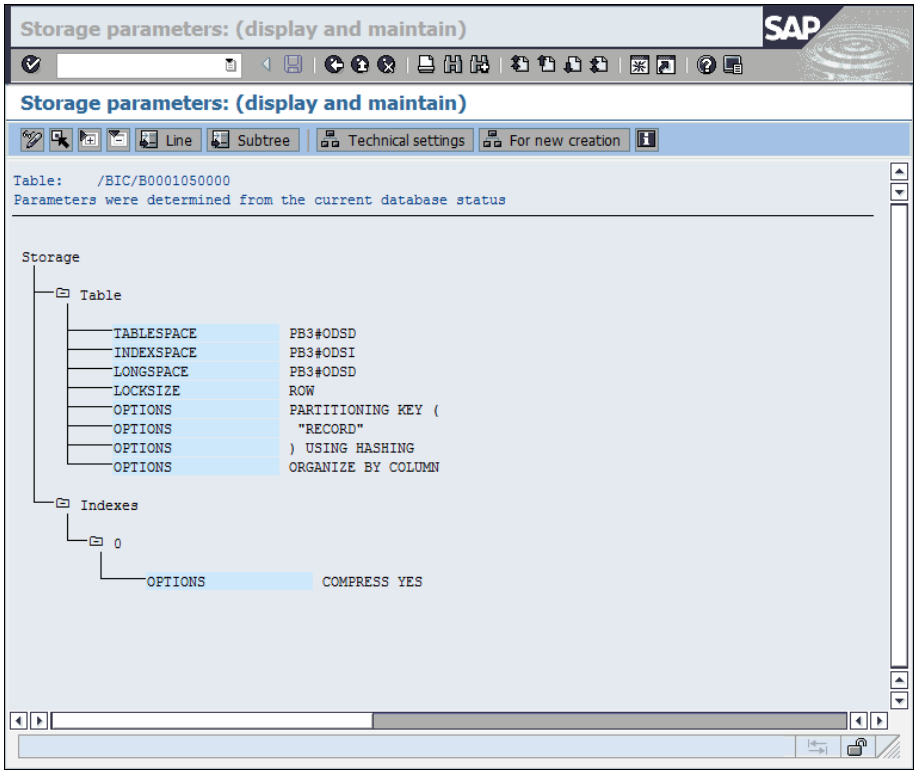



*Figure 8-88 Storage parameters of BLU PSA table*

## 8.5.6 Column-organized temporary tables in SAP BW

When running SAP BW queries, intermediate results are often calculated and stored in database tables. These tables are called “SAP BW temporary tables”. The following temporary table types for InfoCube queries are the most important:

- Temporary hierarchy (02) tables (naming convention is `/BI0/02<number>`)

  These tables store results from the evaluation of master data hierarchies. The data in these tables can be used by multiple SAP BW queries and is therefore preserved for a longer period of time.

- EQSID (06) tables (naming convention is `/BI0/06<number>`)

  These tables are used to store SIDs of master data. They are used to avoid large IN-lists in the WHERE condition of SQL queries.

- Temporary pre-materialization (0P) tables (naming convention is `/BI0/0P<number>`):

  These tables are used to store the results of joins of master data and dimension tables. They are used for SQL queries that join a large number of tables. These queries are split in a way that the result of the join of each dimension table with master data tables is calculated separately and stored in a 0P table. A final SQL query then joins the fact tables with the 0P tables.

Creating SAP BW temporary tables as column-organized tables can improve the performance of queries on BLU Acceleration InfoCubes. By default, SAP BW temporary tables are created as row-organized tables. SAP BW provides the RSADMIN configuration parameter, `DB6_TMP_USE_CDE`, to create SAP BW temporary tables as column-organized tables. You set this parameter as follows:

1. Call transaction **SE38** and run the SAP_RSADMIN_MAINTAIN report.
2. In the “OBJECT” field, enter `DB6_TMP_USE_CDE`. In the VALUE field, enter `YES`, and then click the **Execute** icon (Figure 8-89).

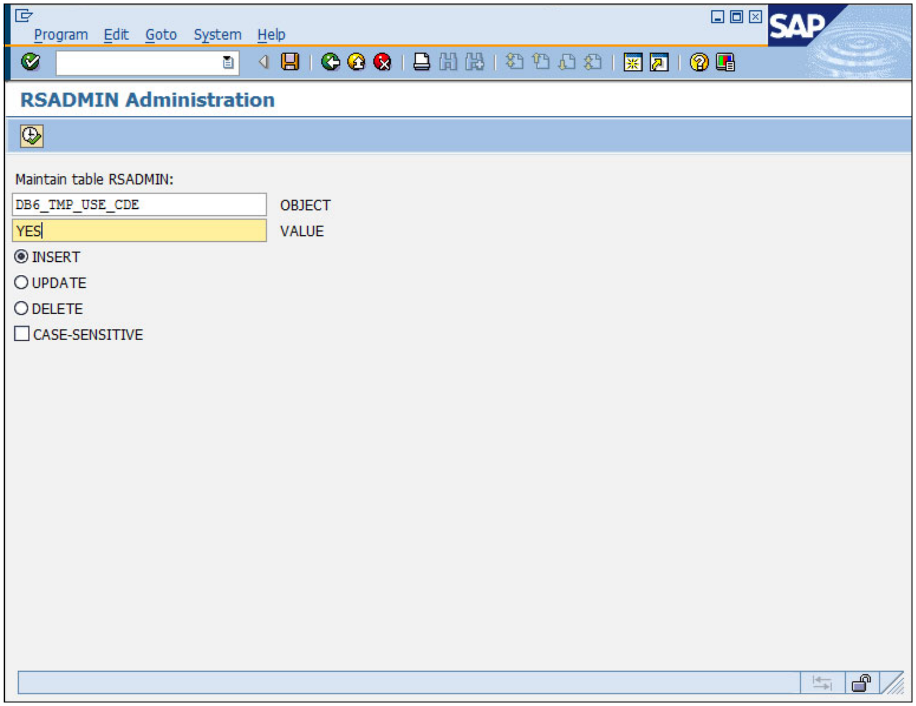



*Figure 8-89 Report SAP_RSADMIN_MAINTAIN: Set option to create SAP BW temporary tables column-organized*

The information shown in Figure 8-90 is displayed.

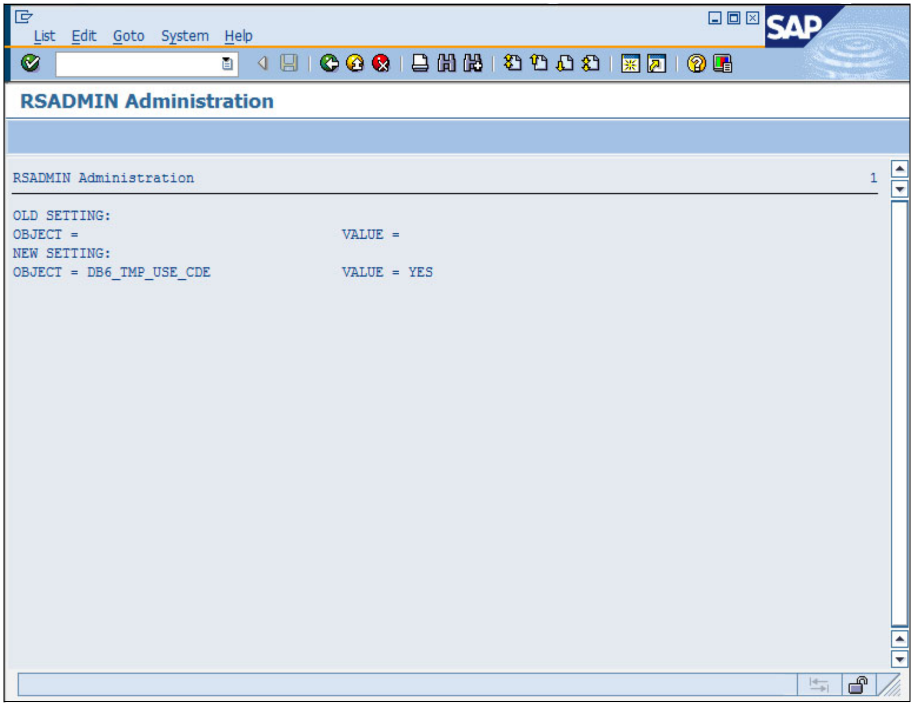



*Figure 8-90 Report SAP_RSADMIN_MAINTAIN: Result of setting the option to create SAP BW temporary tables column-organized*

Consider this information about column-organized temporary tables in SAP BW:

- SAP BW temporary tables are visible in the ABAP Dictionary. You can check whether a temporary table is a column-organized table by inspecting the storage parameters of the table. See 8.3, “BLU Acceleration support in the ABAP Dictionary” on page 261.
- Changes to the RSADMIN parameter, `DB6_TMP_USE_CDE`, have an effect only when new SAP BW temporary tables are created. There is no effect on existing tables. Temporary hierarchy tables and EQSID tables are managed in a pool and reused. Therefore, a large number of row-organized temporary tables might exist already when you set the RSADMIN parameter. When you start working with BLU Acceleration InfoCubes, drop existing temporary tables and set the RSADMIN parameter so that new temporary tables are created as column-organized tables. You can drop temporary tables with the SAP_DROP_TMPTABLES report. Before you drop temporary tables, make sure that you read *SAP Note 1139396*.

When you have a mixture of row-organized and column-organized InfoCubes in your system and generate SAP BW temporary tables as column-organized tables, these column-organized temporary tables might be used in queries on row-organized InfoCubes. This might cause a small performance penalty. However, this is outweighed by the performance benefit for queries on column-organized InfoCubes.

## 8.6 Conversion of SAP BW objects to column-organized tables

You can convert existing InfoCubes, DSOs, InfoObjects, and PSA tables from row-organized to column-organized tables with the conversion report DB6CONV. The conversion of standard column-organized InfoCubes to flat ones has to be done separately as described in 8.5.2, “Flat InfoCubes in SAP BW 7.40” on page 297.

When you have installed DB2 10.5 FP3SAP BLU Acceleration is only supported for InfoCubes. In this case you must use the report SAP_CDE_CONVERSION_DB6 to convert existing InfoCubes to column-organized tables. SAP_CDE_CONVERSION_DB6 creates a table conversion job that you can schedule and run with DB6CONV. This is described in detail in 8.6.2, “Conversion of InfoCubes to column-organized tables before DB2 Cancun Release 10.5.0.4”.

When you have installed DB2 Cancun Release 10.5.0.4 and DB6CONV V6 you can generate conversion jobs for InfoCubes, DSOs, InfoObjects, and PSA tables directly with the DB6CONV tool. You do not need the report SAP_CDE_CONVERSION_DB6.

### 8.6.1 Conversion of SAP BW objects to column-organized tables with DB2 Cancun Release 10.5.0.4 and DB6CONV V6

DB6CONV V6 provides a new tab page for the conversion of SAP BW object tables from row-organized to column-organized and vice versa. You can convert single, selected, or all InfoCubes, DSOs, InfoObjects, and PSA tables. DB6CONV collects the tables of the selected BW objects that need to be converted and puts them into a DB6CONV job.

You can schedule and run the job like any other DB6CONV table or table space conversion jobs:

1. Call transaction **SE38** and run report DB6CONV, as shown in Figure 8-91.

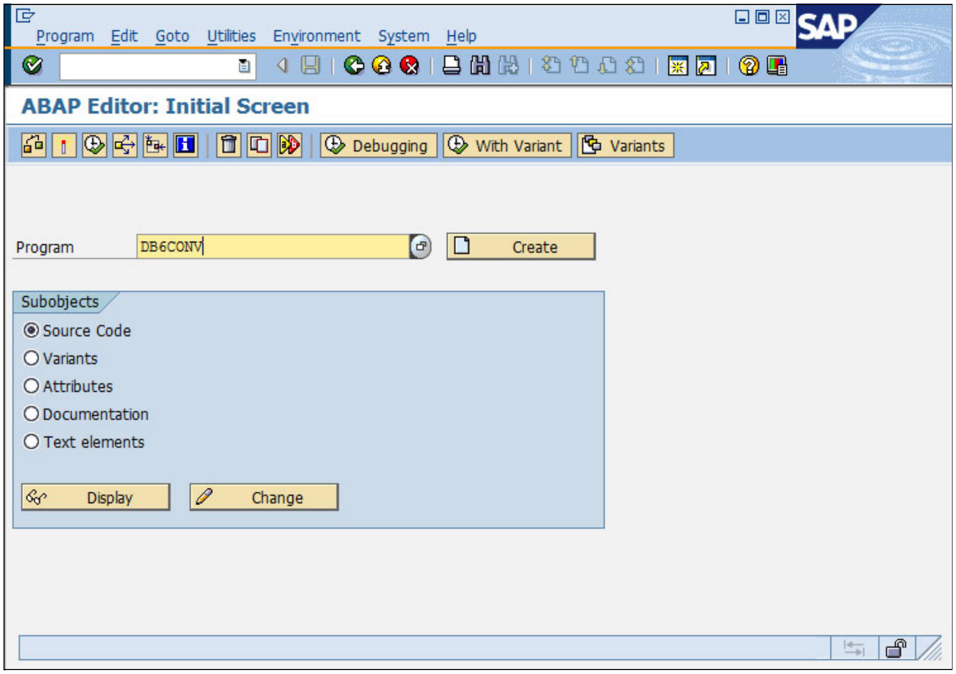



*Figure 8-91 Invocation of report DB6CONV*

The DB6CONV conversion window is shown (see Figure 8-92).

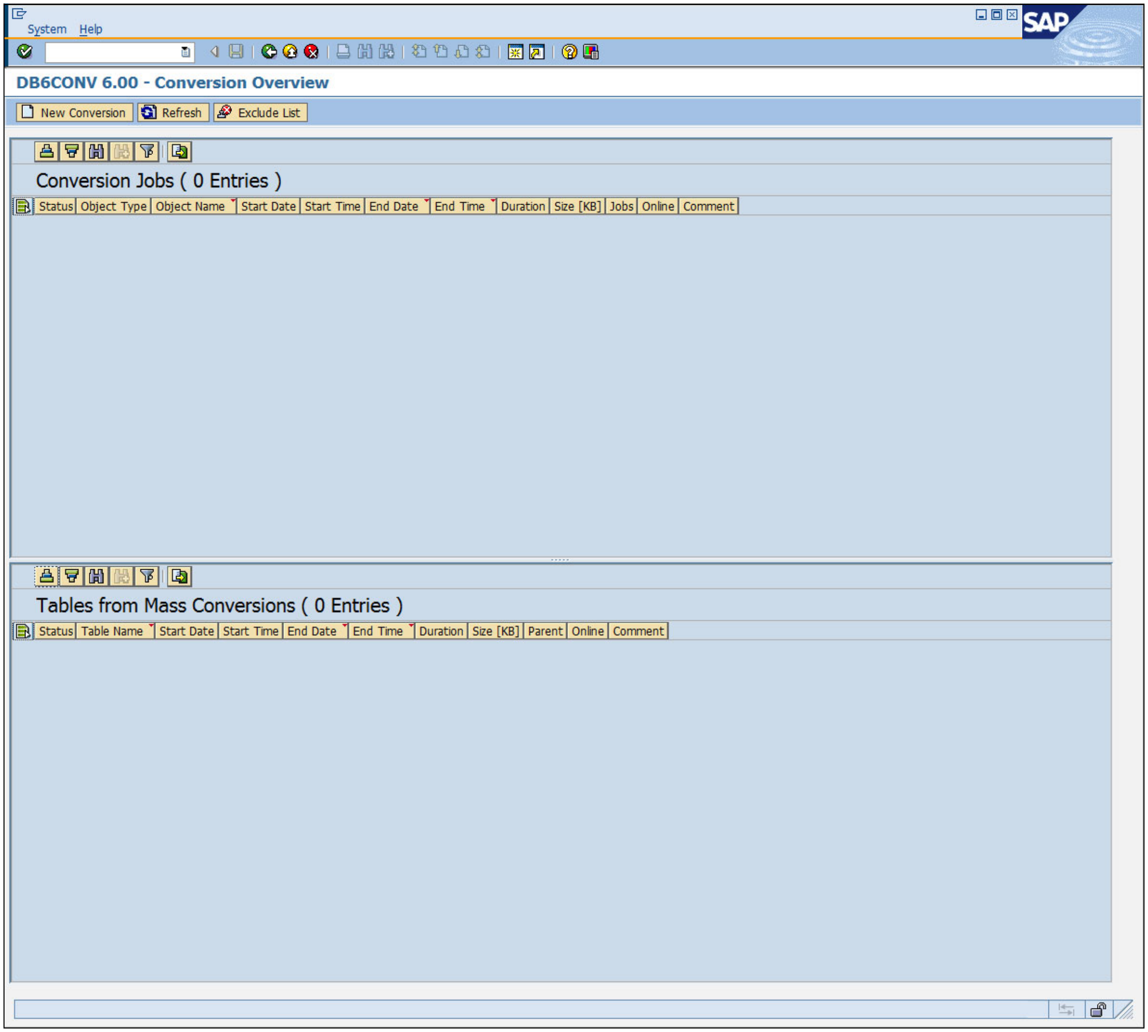



*Figure 8-92 DB6CONV Tool - Conversion jobs overview*

2. Choose **New Conversion** and go to the **BW Conversions** tab page (Figure 8-93).

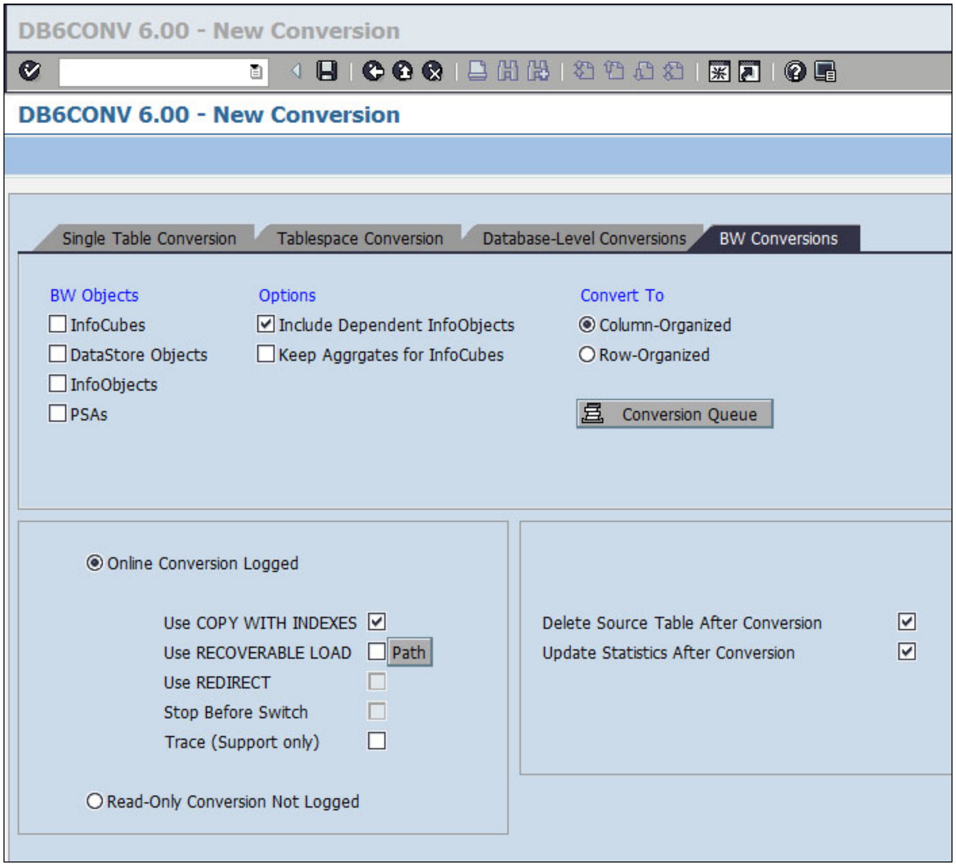



*Figure 8-93 DB6CONV Tool - BW Conversions*

3. You can convert InfoCubes, DSOs, InfoObjects, and PSA tables from row-organized to column-organized tables and vice versa. The following options are available:

   – *Include Dependent InfoObjects:* If you use this option, the InfoObjects that are referenced in the InfoCubes and DSOs to be converted are included in the conversion job. Inclusion of InfoObjects is the default because reporting performance usually improves if the InfoObject tables that are referenced in the queries have the same layout as the InfoCube or DSO tables. If you do not want to convert InfoObjects, you must deselect this option.

- *Keep Aggregates for InfoCubes:* For conversions from row-organized to column-organized tables, the default is not to keep the InfoCube aggregates. In most cases the aggregates are no longer needed because the InfoCube queries run fast enough on the InfoCube itself. Therefore the aggregates are deactivated when the InfoCube tables have been converted to column-organized tables. The definitions of the aggregates are kept in the SAP BW metadata so that selected aggregates can be re-activated and filled if needed.

For conversions of column-organized to row-organized tables, the default is to keep the InfoCube aggregates. In such a conversion, the aggregate tables are also converted to row-organized tables.

The following conversion methods are offered:

- *Online Conversion Logged*: With this conversion method, DB6CONV creates the target table and then calls the AMDIN_MOVE_TABLE stored procedure to create the indexes (optional) and copy the data. The following options for ADMIN_MOVE_TABLE are available:
  - *Use COPY WITH INDEXES:* This option is the default and should not be changed. ADMIN_MOVE_TABLE takes care of the conversion of unique and primary key indexes on row-organized tables to BLU constraints and vice versa. Non-unique indexes on the row-organized tables are not created on target column-organized tables.
  - *Use RECOVERABLE LOAD:* When this option is selected, ADMIN_MOVE_TABLE uses LOAD WITH COPY YES to copy the data. Choose **Path** to specify a location for the LOAD COPY files.
  - *Trace (Support Only):* With this option, an ADMIN_MOVE_TABLE trace in the db2dump directory is created. This can be useful to identify issues in support situations, but it generates a significant performance overhead as well.
- *Read-Only Conversion Not Logged:* With this conversion method, DB6CONV creates the target table and its indexes or constraints and uses the non-recoverable DB2 LOAD to copy the data from the source to the target tables.

In the following example, you convert two row-organized InfoCubes to column-organized without InfoObjects:

1. Select **InfoCubes**, de-select Include Dependent InfoObjects, and choose **Conversion Queue** (Figure 8-94).

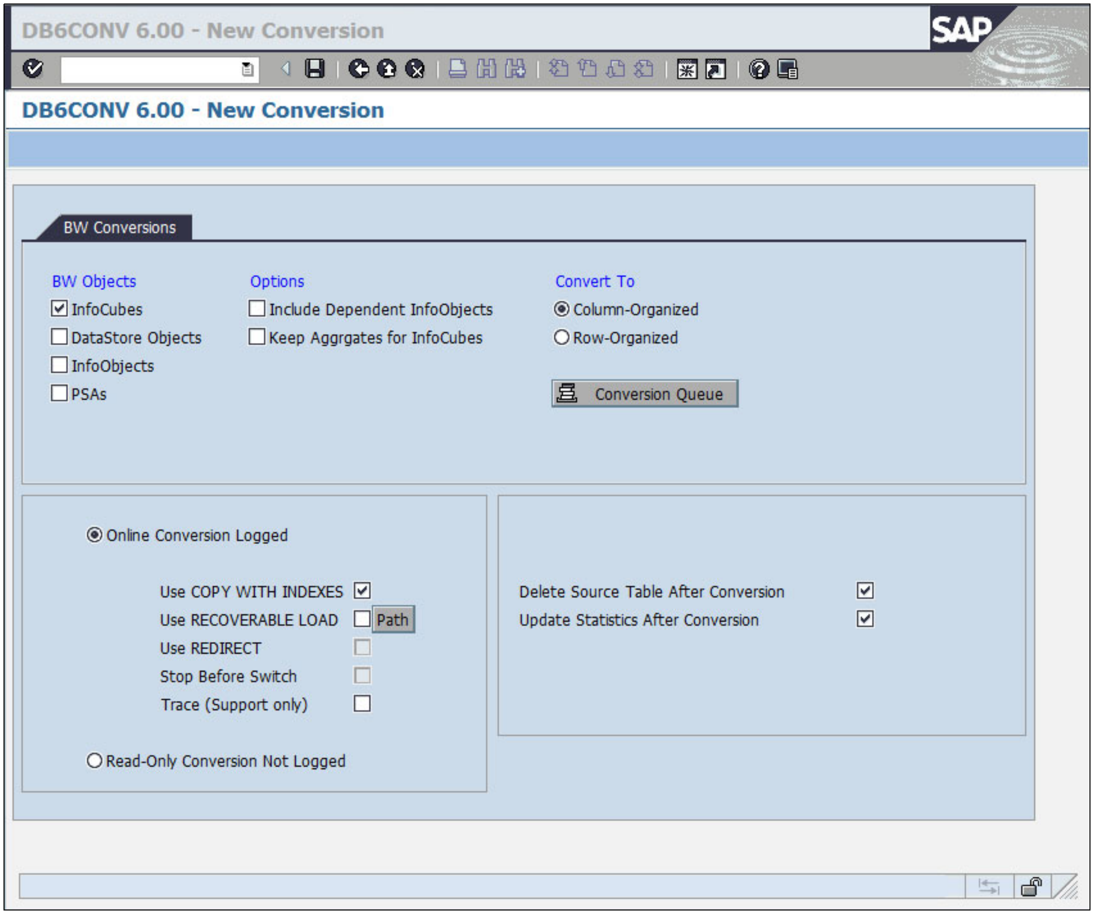



*Figure 8-94 DB6CONV - Conversion of InfoCube tables to column-organized tables*

2. A pop-up window with all basic row-organized InfoCubes in the system is shown (Figure 8-95). In the header, the number of InfoCubes and the number of database tables and their overall size is displayed.

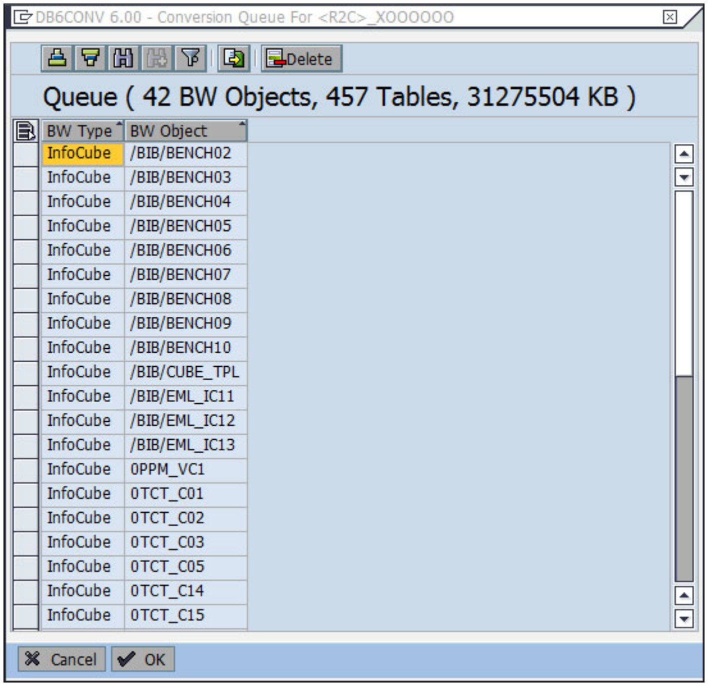



*Figure 8-95 DB6CONV - Conversion candidate InfoCubes*

3. If you do not want to convert all InfoCubes in the list, you must select the InfoCubes that should not be converted and delete them from the list. In this example, you want to convert only the InfoCubes `ROWCUBE1` and `ROWCUBE2`.
4. Click the icon in the top left corner of the list to select all list elements, then search for `ROWCUBE1` and `ROWCUBE2` (binoculars icon) and de-select these two InfoCubes only (Figure 8-96).

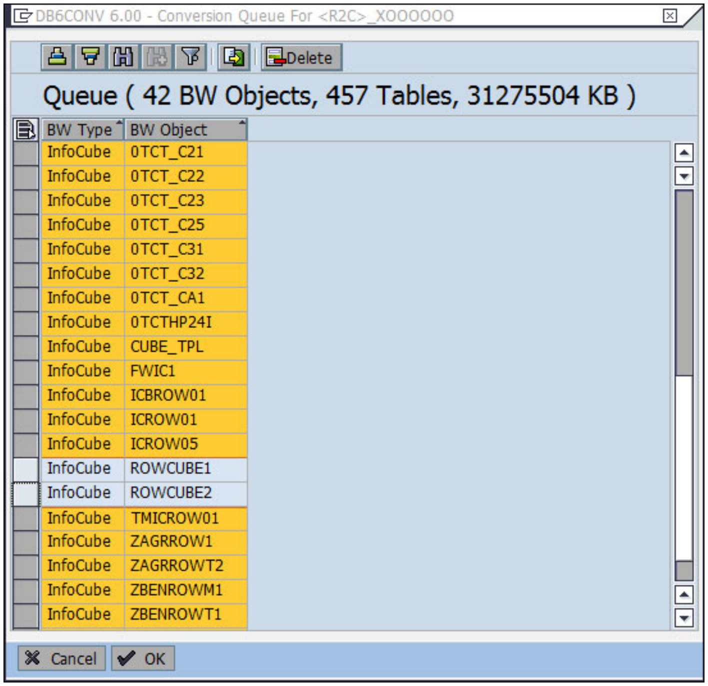



*Figure 8-96 DB6CONV - Selection of BW Objects to be excluded*

5. Then choose **Delete** to delete all InfoCubes except `ROWCUBE1` and `ROWCUBE2` from the list (Figure 8-97).

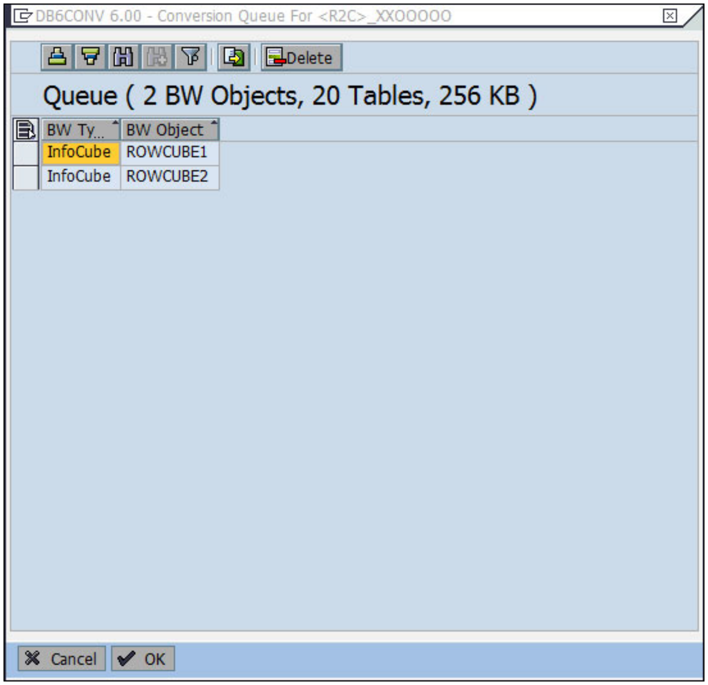



*Figure 8-97   DB6CONV - Selected conversion candidates*

6. When you double-click one of the two InfoCubes, the list of its database tables is displayed in a separate pop-up window (Figure 8-98).

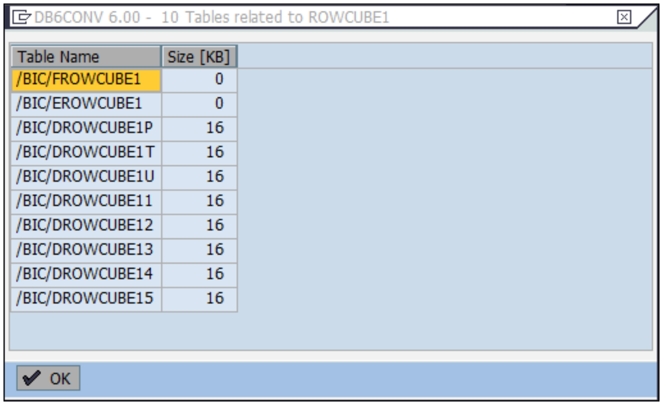



*Figure 8-98   DB6CONV - List of tables of a sample InfoCube to be converted*

7. Choose **OK** two times to close the pop-up windows and save the conversion queue in the BW Conversions window (floppy disc icon in Figure 8-94 on page 355). The pop-up window in Figure 8-99 is shown.

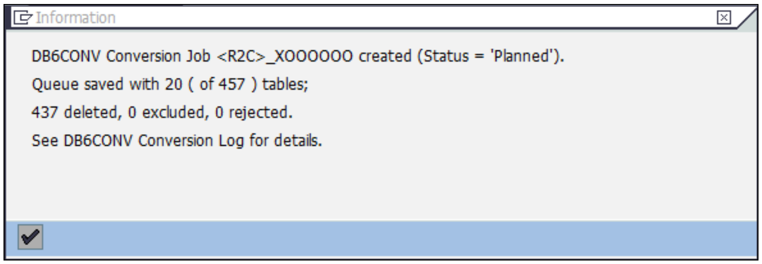



*Figure 8-99   DB6CONV - Information about newly created conversion job*

8. When you confirm the information message, the conversion overview window is shown with the newly created conversion job with status *PLANNED* (see Figure 8-100).

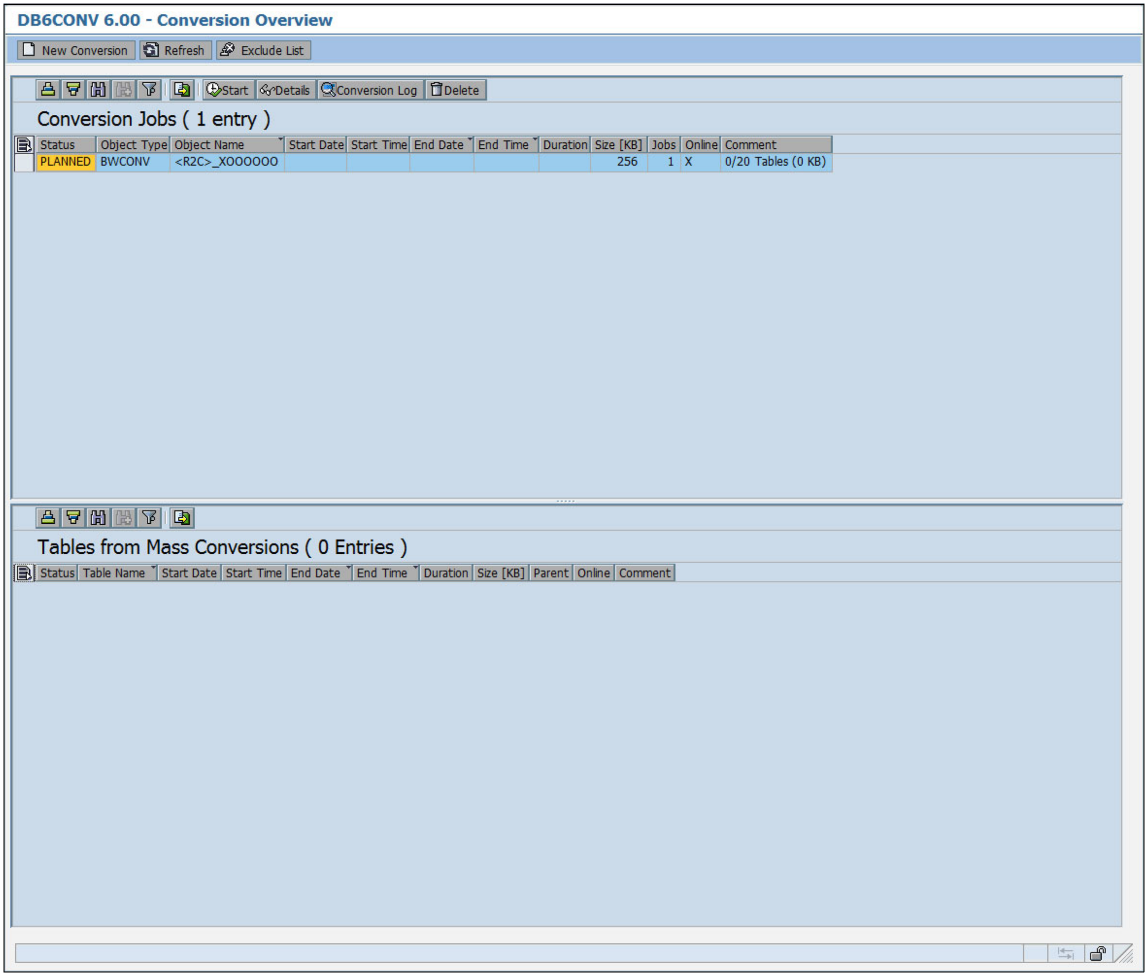



*Figure 8-100 DB6CONV - Conversion jobs overview window*

The object type (BWCONV) of the job indicates that this is a conversion of BW objects. The job name reflects the conversion direction (R2C = row-organized to column-organized) and the conversion options that were selected.

9. Select the job and choose **Start** to schedule it. The *Start Conversion* window is shown (Figure 8-101).

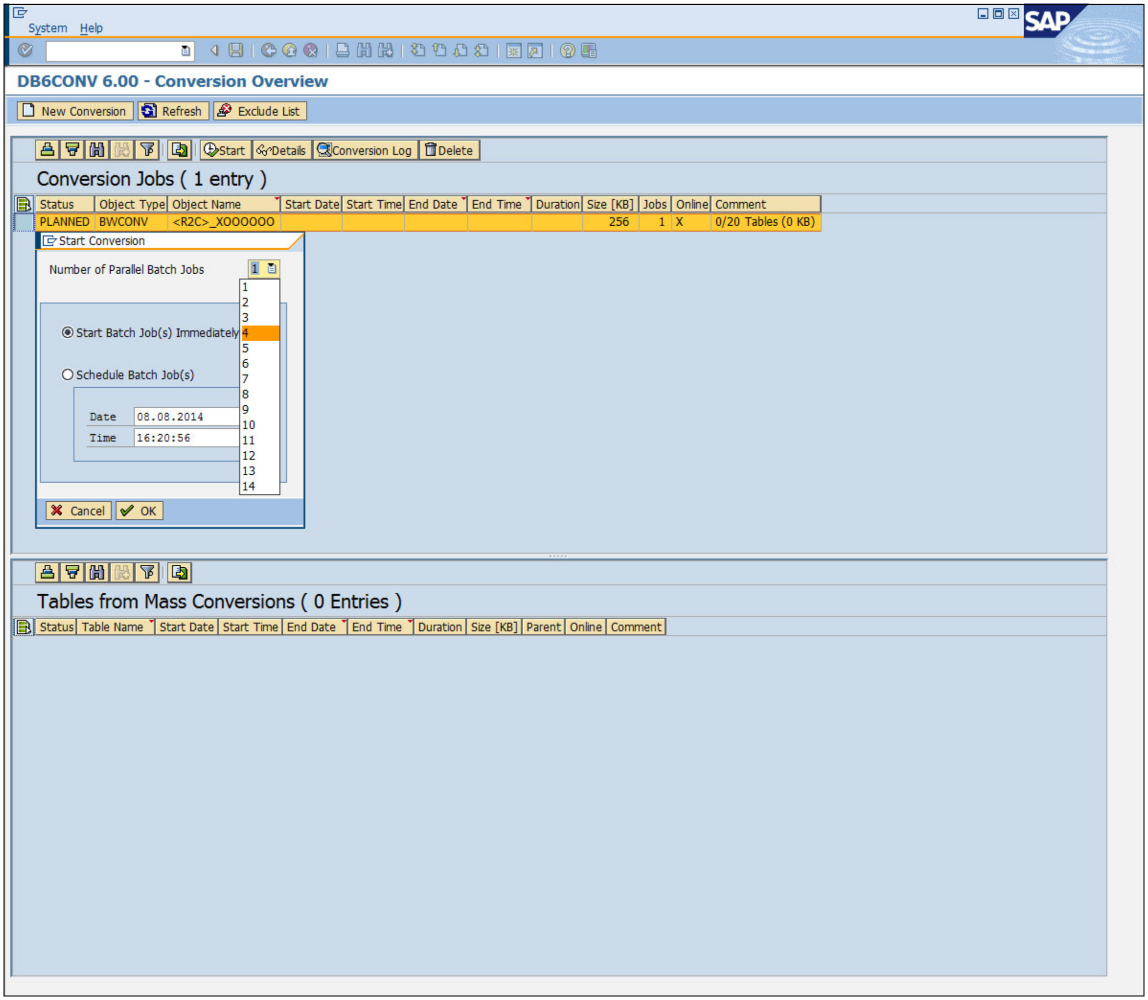



*Figure 8-101 DB6CONV - Schedule a conversion job*

10. We can choose the job start time and the number of parallel processes with which it is executed. Improvements in DB6CONV V6 allow to choose a much higher degree of parallelism than in previous DB6CONV versions.

During the conversion, the job status changes to *RUNNING* and the tables that have already been processed and that are currently being processed are listed on the *Tables from Mass Conversions* area in the *Conversion Overview* window (Figure 8-102).

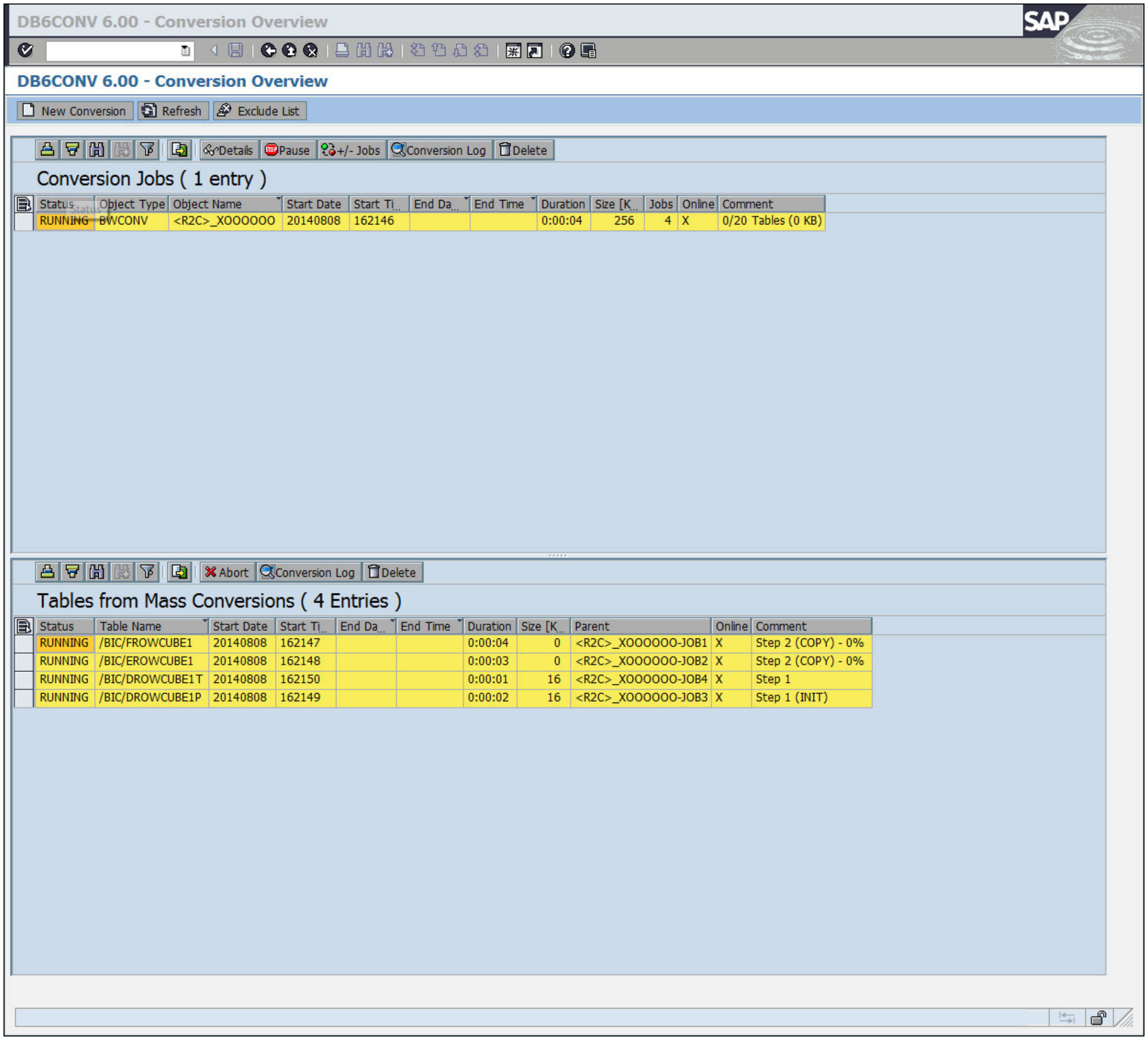


*Figure 8-102 DB6CONV - Running conversion job*

11. Wait until the job is finished (Figure 8-103).

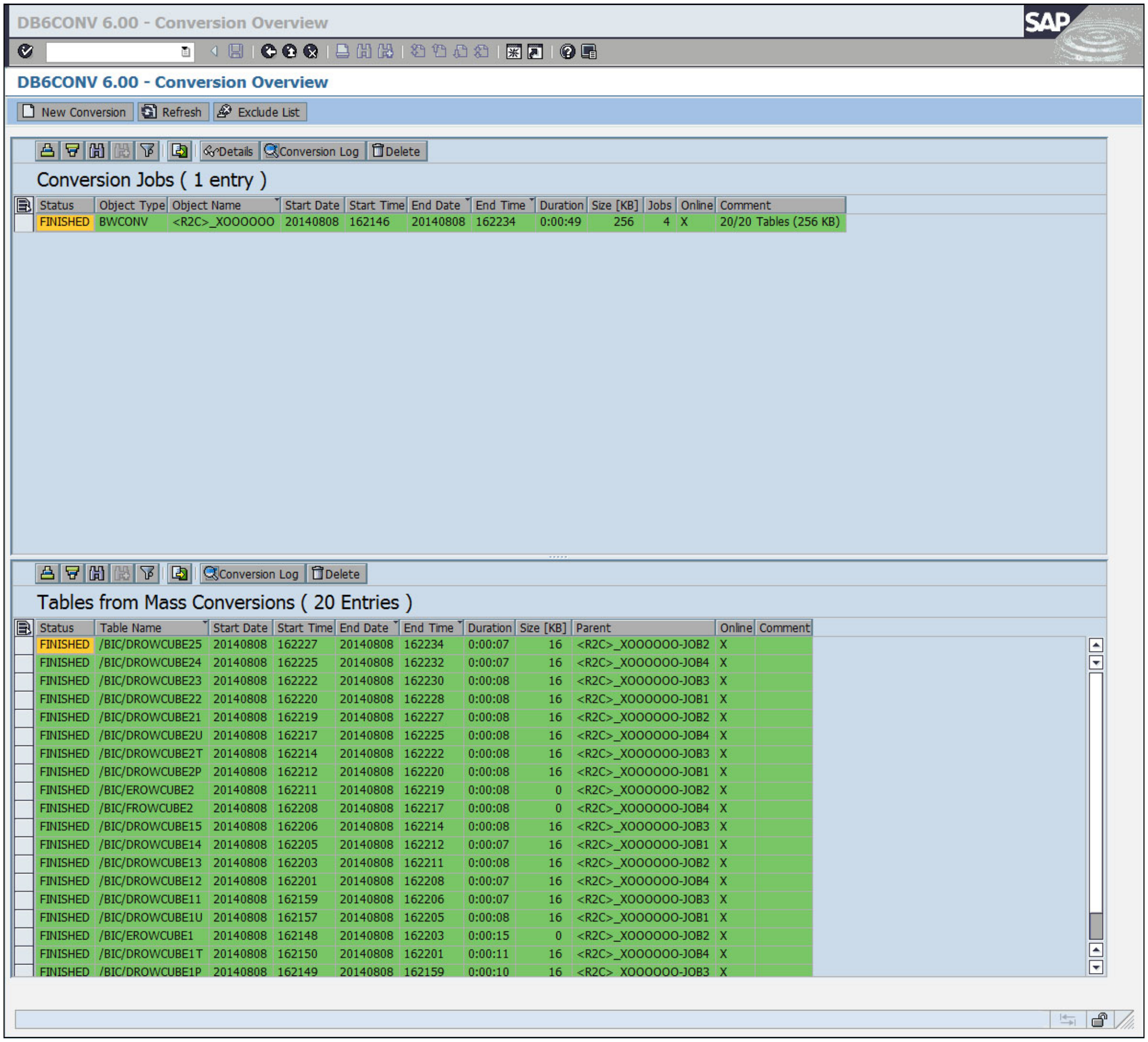



*Figure 8-103 DB6CONV - Completed conversion job*

12. DB6CONV writes a conversion log for the complete conversion job and for each table separately. You can examine the log by double-clicking the conversion job or table or by selecting the job or table and choosing **Conversion Log** in the *Conversion Overview* window.

Figure 8-104 and Figure 8-105 show an excerpt from the conversion logs of the conversion job and of dimension table /BIC/DROWCUBE25. The log for each table log contains the conversion options that were selected in the **BW Conversions** tab, the CREATE TABLE statement for the target table, an overview of the indexes and constraints that are defined on the target table, and information about the ADMIN_MOVE_TABLE steps that were executed. The log of the conversion job contains information about the InfoCubes and tables that were deleted from the original conversion queue, about the parallel child jobs that executed the conversion and of required post processing steps that were executed.

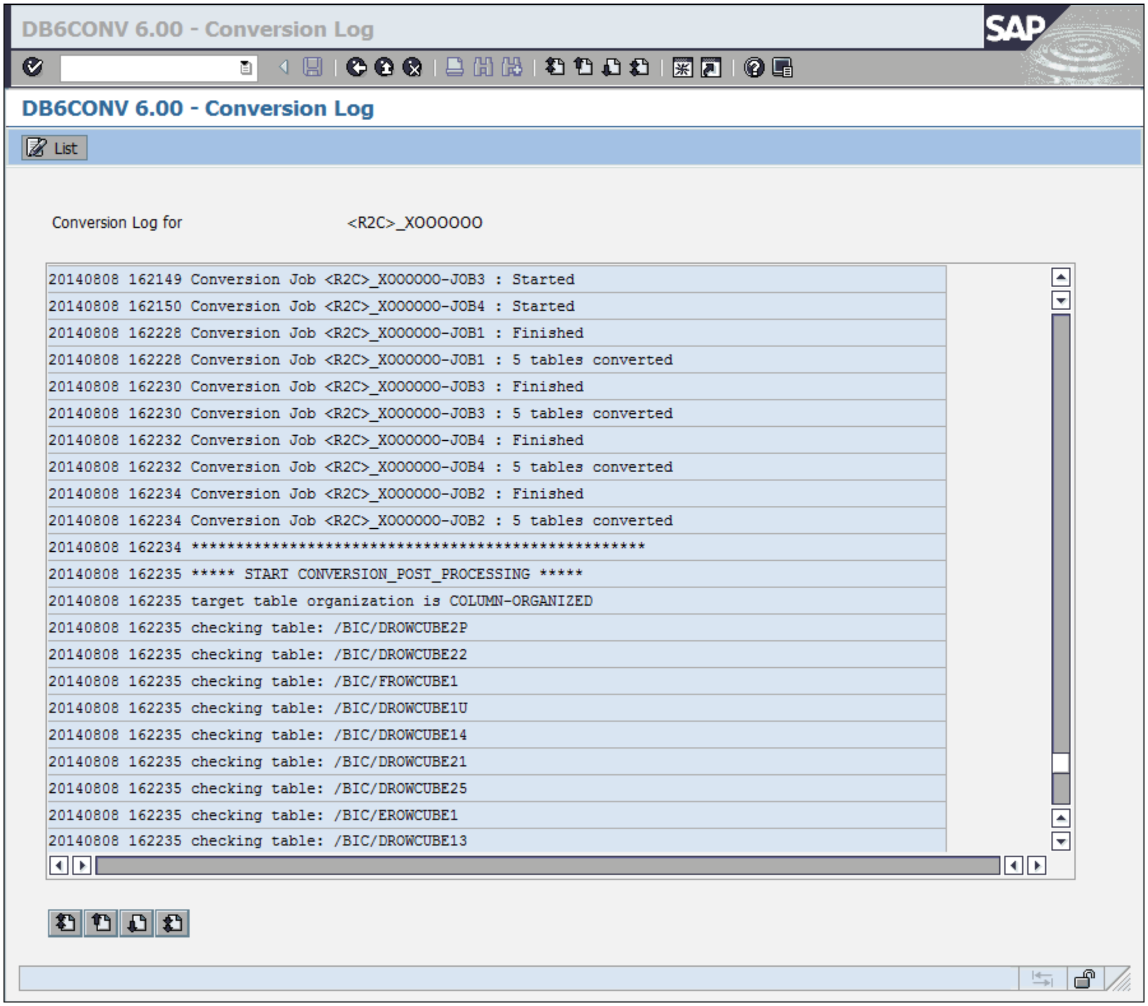



*Figure 8-104 DB6CONV - Sample conversion log file*

13. When a conversion aborts, take the following actions:

   a. For each table that failed during the conversion, examine the conversion log to identify the root cause of the failure. If you cannot solve the issue, open an incident at SAP.

   b. Reset the conversion of each table that failed in the Table from *Mass Conversions* area in the *Conversion Overview* window.

   c. Reset the conversion of the whole job in the *Conversion Jobs* area.

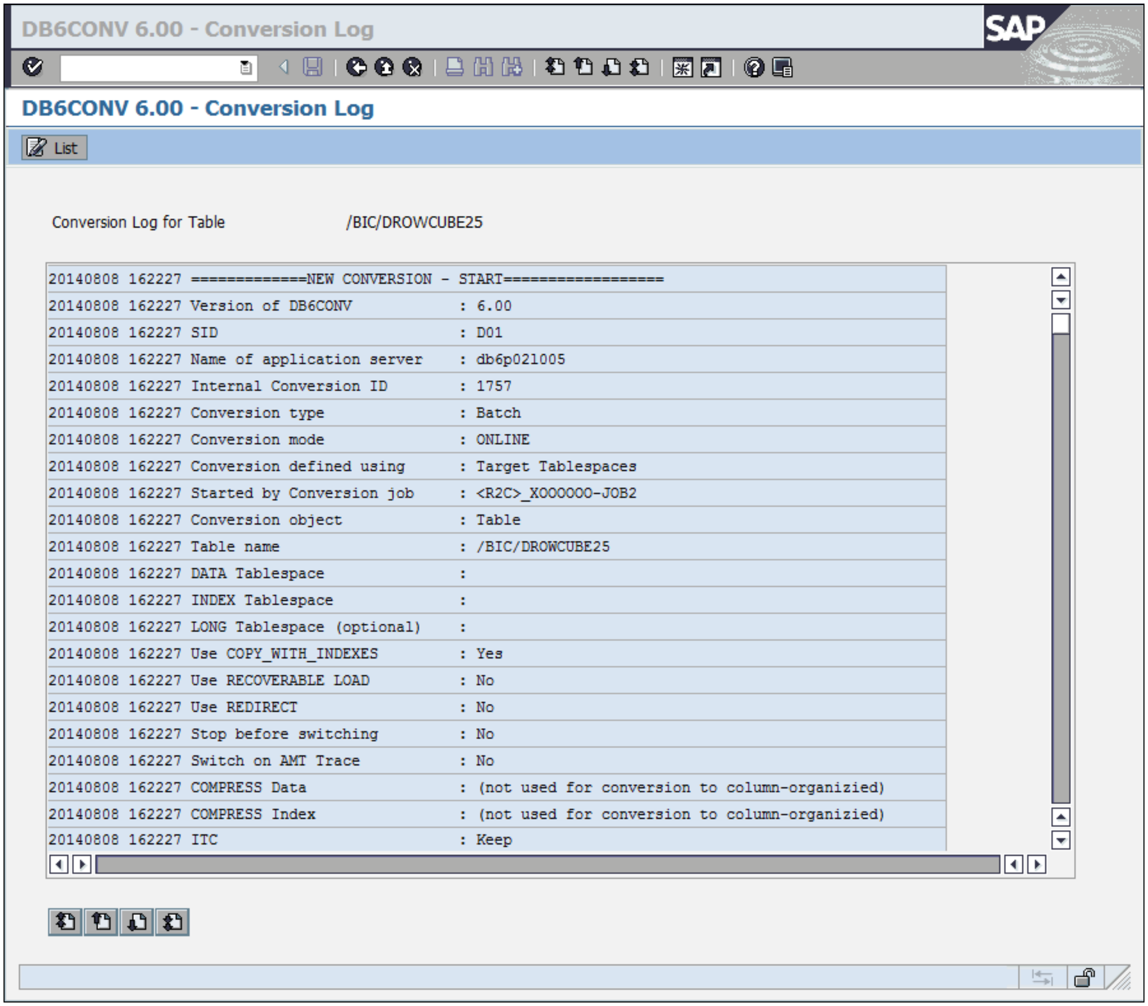



*Figure 8-105 DB6CONV - Sample table conversion log file*

### 8.6.2 Conversion of InfoCubes to column-organized tables before DB2 Cancun Release 10.5.0.4

SAP BW provides the SAP_CDE_CONVERSION_DB6 report for converting existing InfoCubes from row-organized to column-organized tables and vice versa if you have DB2 10.5 FP3aSAP installed. You can convert a single InfoCube or all InfoCubes in the BW system. Only the fact and dimension tables are converted. The InfoObjects that are referenced in the InfoCubes remain row-organized. The report collects the database tables for the selected InfoCube or for all InfoCubes and creates a job for the SAP DB2-specific table conversion report DB6CONV. The conversion job can be scheduled and monitored in the DB6CONV report.

To use SAP_CDE_CONVERSION_DB6, install *SAP Note 1964464*. This SAP Note fixes issues with respect of the handling of InfoCube aggregates.

You need at least DB6CONV V5.30 for column-organized tables to be supported. The DB6CONV report and documentation are in *SAP Note 1513862*. SAP recommends that you always use the latest available DB6CONV version.

It is important that you use the SAP_CDE_CONVERSION_DB6 report for the conversion to BLU Acceleration and not native DB2 tools such as **`db2convert`** or the ADMIN_MOVE_TABLE stored procedure directly. The report ensures that all InfoCube tables are converted and it ensures that the constraints on the converted tables are created as required by SAP BW.

In the following example, we use the SAP_CDE_CONVERSION_DB6 report and the DB6CONV report to convert InfoCube ROWCUBE1 to column-organized tables:

1. We call transaction **SE38** and run the SAP_CDE_CONVERSION_DB6 report, as shown in Figure 8-106.

.

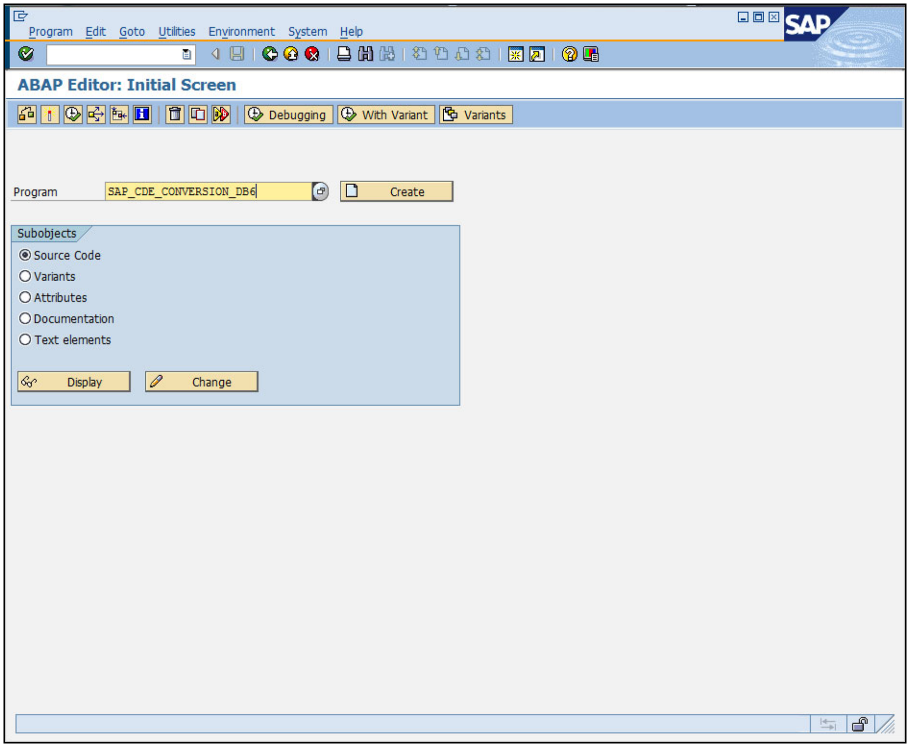



*Figure 8-106 Invocation of report SAP_CDE_CONVERSION_DB6*

2. You can enter the name of an InfoCube in the InfoCube field, get a list for selection of InfoCubes in the system with F4 (help), or select **All InfoCubes** to convert all InfoCubes in the BW system. Enter `ROWCUBE1` in the InfoCube field and click **Get Dependent Tables** (Figure 8-107).

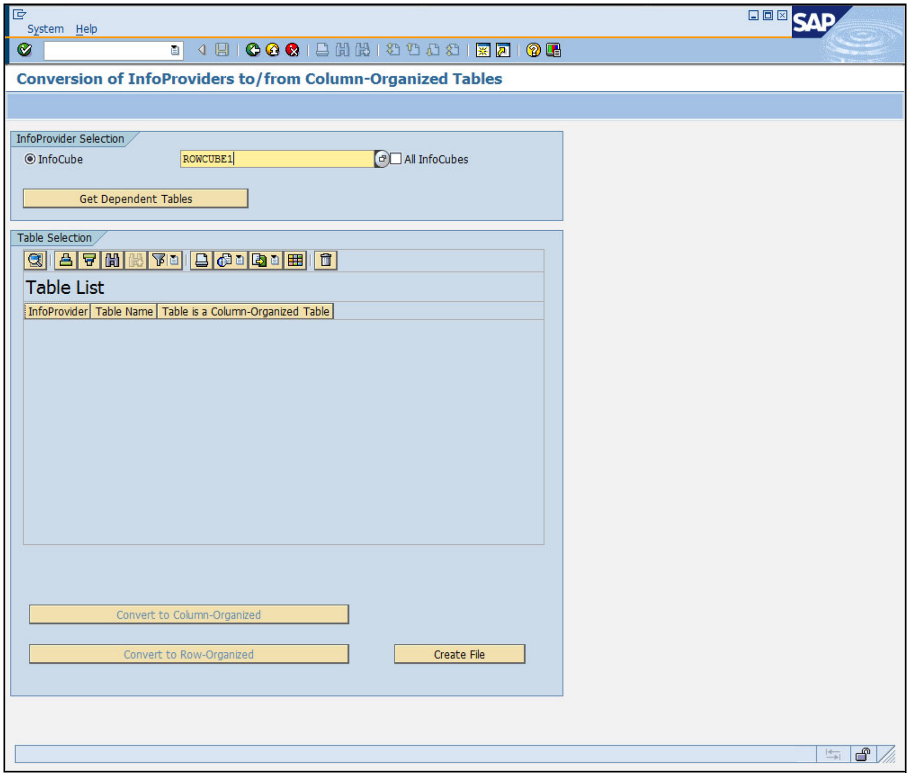



*Figure 8-107 Report SAP_CDE_CONVERSION_DB6: Conversion of an InfoCube*

3. The InfoCube tables are listed in the *Table Selection area* (Figure 8-108). The list indicates whether each table is or is not column-organized.

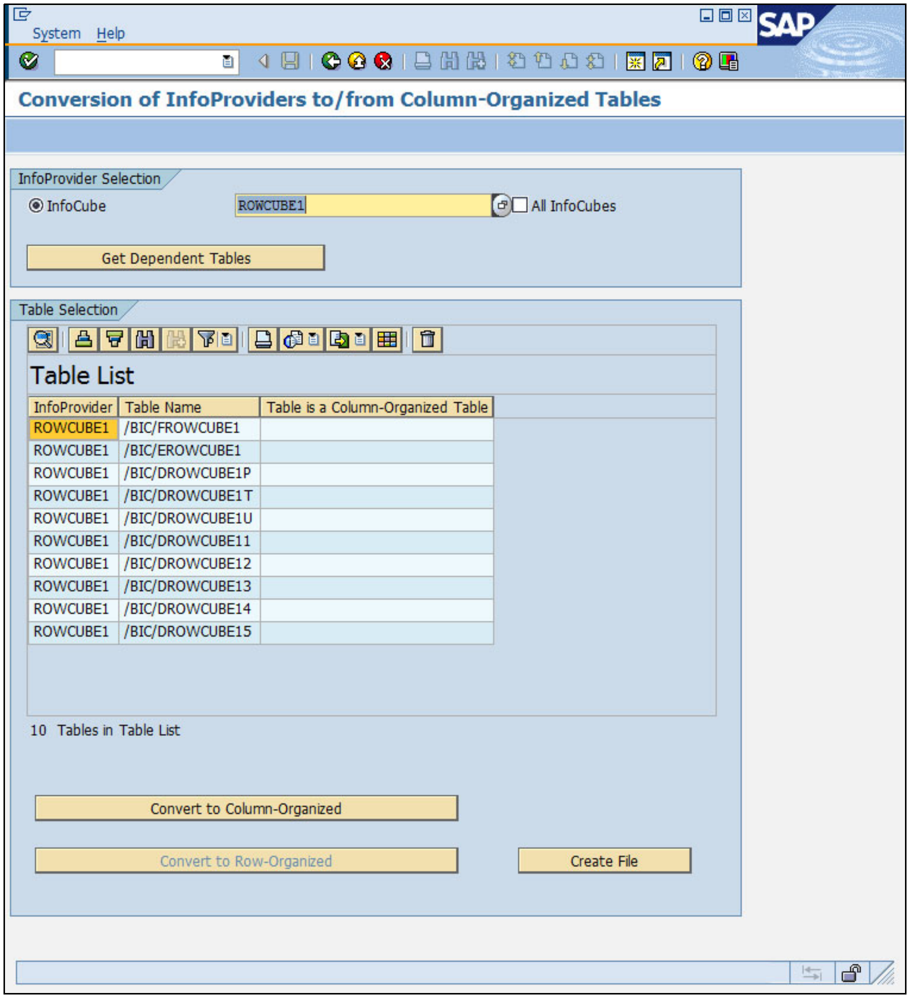



*Figure 8-108 Report SAP_CDE_CONVERSION_DB6 -List of InfoCube tables to be converted*

4. In this example, all tables are row-organized. You can save the table list in a file by clicking **Create File**, and you can create a DB6CONV conversion job by choosing **Convert to Column-Organized**.

   When you click **Convert to Column-Organized**, the message in Figure 8-109 is displayed.

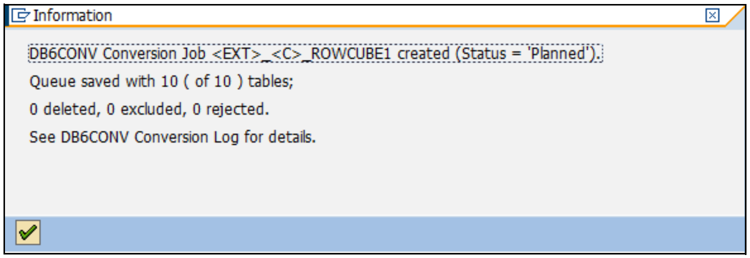



*Figure 8-109 Report SAP_CDE_CONVERSION_DB6: Information about DB6CONV job*

5. A DB6CONV job, <EXT>_<C>_ROWCUBE1, with 10 tables to be converted was created. The <C> in the job name indicates that this is a conversion job to column-organized tables. You schedule, run, and monitor the job using the DB6CONV report.
6. If you choose to convert all InfoCubes, all InfoCube tables are determined. Because a mixture between column-organized and row-organized InfoCubes might exist in your system, both options (Convert to Column-Organized and Convert to Row-Organized) are offered. Depending on which option you choose, a job for either the column-organized or the row-organized tables is created.

   You can convert only basic InfoCubes. If the InfoCube that you want to convert has aggregates, the following options are available:

   – The InfoCube is a row-organized InfoCube that you want to convert to column-organized.

     In this case, you are asked whether you want to keep the aggregates or deactivate them. If you use BLU Acceleration, you do not need aggregates any more in many cases. Therefore, you might choose to deactivate the aggregates. In this case, the aggregate definitions are preserved but the aggregate database tables are dropped.

If you determine later that some of the deactivated aggregates would still be useful, you can reactivate these aggregates in the *Maintenance for Aggregates* window of the Data Warehousing Workbench. However, you must refill them with the data from the InfoCube. If you want to keep the aggregates when you convert an InfoCube to BLU Acceleration, the aggregate tables are included into the conversion job in such a way that they are also converted to BLU Acceleration.

– The InfoCube is a column-organized InfoCube that you want to convert back to row-organized.

  In this case, the aggregate tables are automatically added to the conversion job in a way that the aggregate tables are also converted back to row-organized.

### Running and monitoring InfoCube conversions

You can run and monitor InfoCube conversion to or from BLU Acceleration by using the DB6CONV report as follows:

1. Open a new SAP window, call transaction **SE38**, and run **DB6CONV** (Figure 8-110).

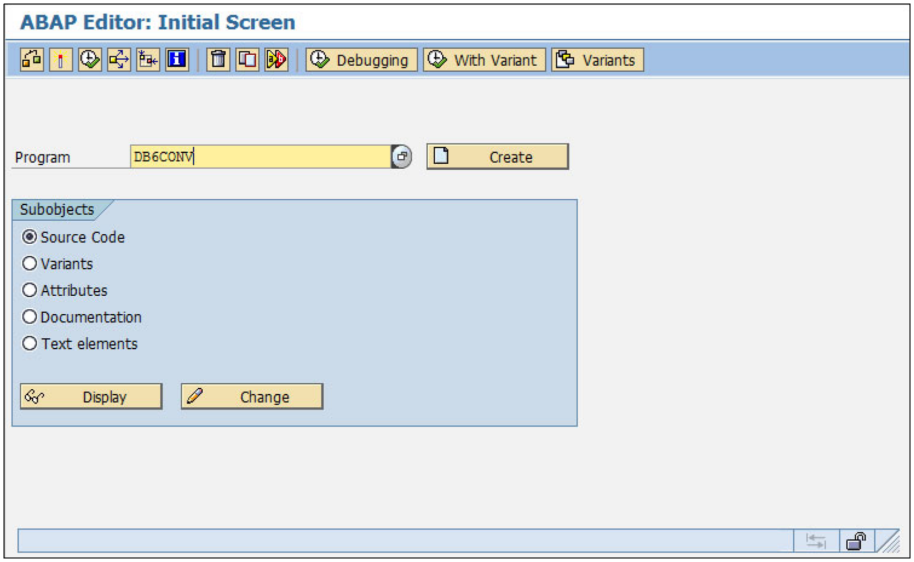



*Figure 8-110 Invocation of report DB6CONV*

2. The DB6CONV Conversion Overview window opens. It contains the job that you created with SAP_CDE_CONVERSION_DB6 with PLANNED status (Figure 8-111). The job has the *Object type* `external`. This means that the job has been created outside of the DB6CONV tool.

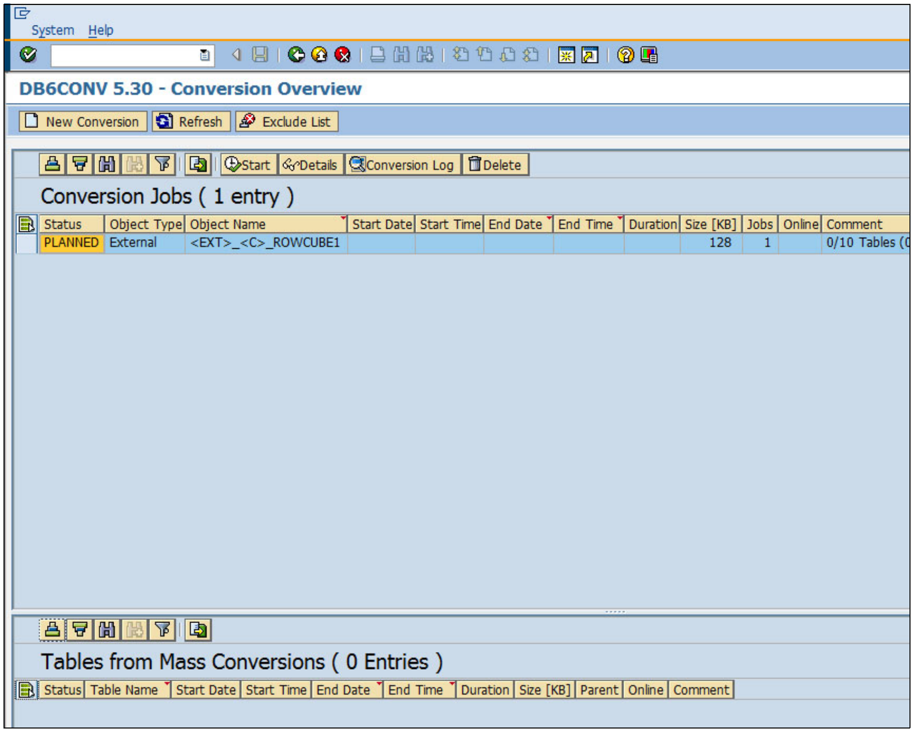



*Figure 8-111 Report DB6CONV: Job overview*

3. Select the job in the Conversion Jobs list and click **Details**.

   The Details window opens where you can inspect or change conversion details (Figure 8-112).

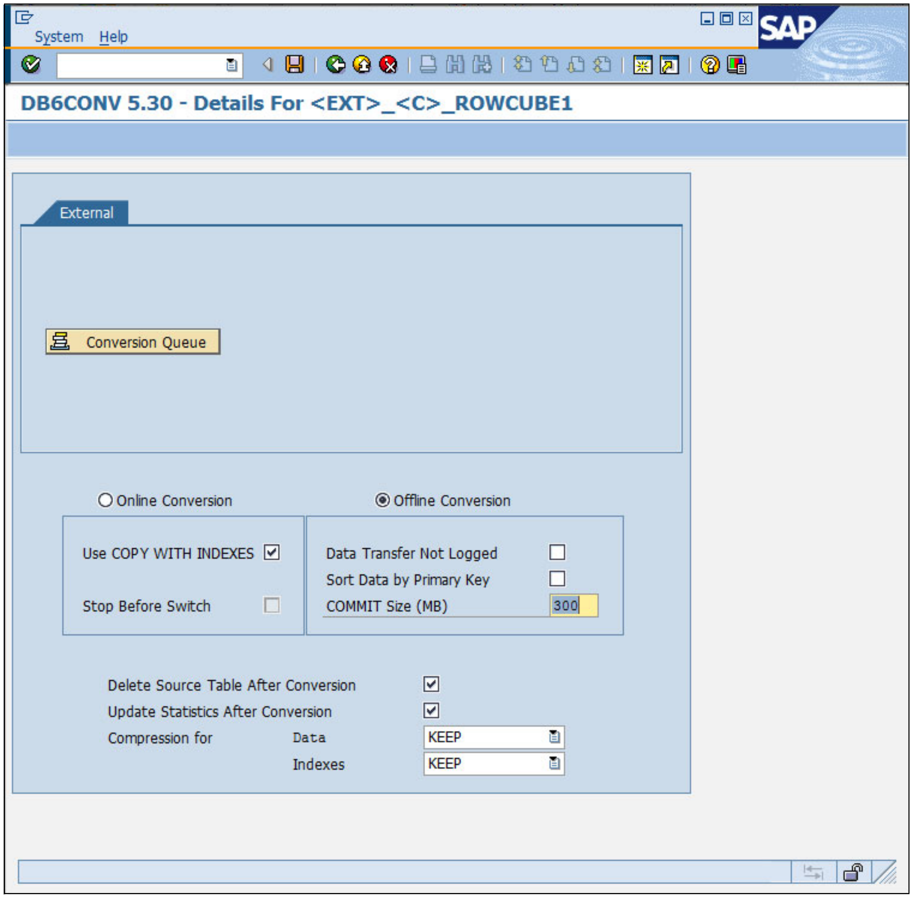



*Figure 8-112 Report DB6CONV: Job details*

The default conversion type is offline. You can convert row-organized tables to column-organized tables online. Conversions of column-organized tables to row-organized tables can be run only in read-only mode.

The `Use COPY WITH INDEXES` option is not evaluated for conversions of InfoCube tables, neither column-organized nor row-organized. Indexes and constraints are always created as required by a callback routine into SAP BW.

Because column-organized tables are always automatically compressed, the `Compression for Data` and `Indexes` options are not evaluated when you convert row-organized tables into column-organized tables.

4. Save the conversion details and return to the Conversion Overview window.
5. Schedule the job to run immediately or at your preferred time (Figure 8-113).

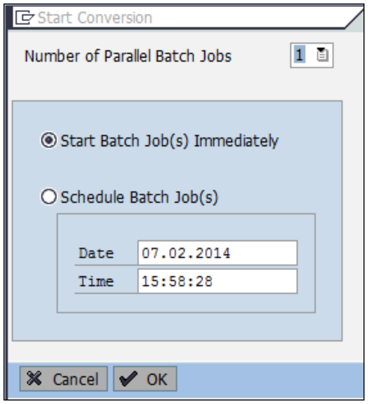



*Figure 8-113   Report DB6CONV: Job scheduling window*

You can split large conversion jobs with many tables into several jobs running in parallel by selecting the wanted number of jobs from the *Number of Parallel Batch Jobs* drop-down list. You can choose a maximum of 14 parallel jobs.

During the conversion, the tables that have already been processed and that are currently processed are listed on the *Tables from Mass Conversions* area in the *Conversion Overview* window (see details in “Considerations for InfoCube conversions” on page 375).

When all tables are processed successfully, the job status is changed to *FINISHED* (Figure 8-114).

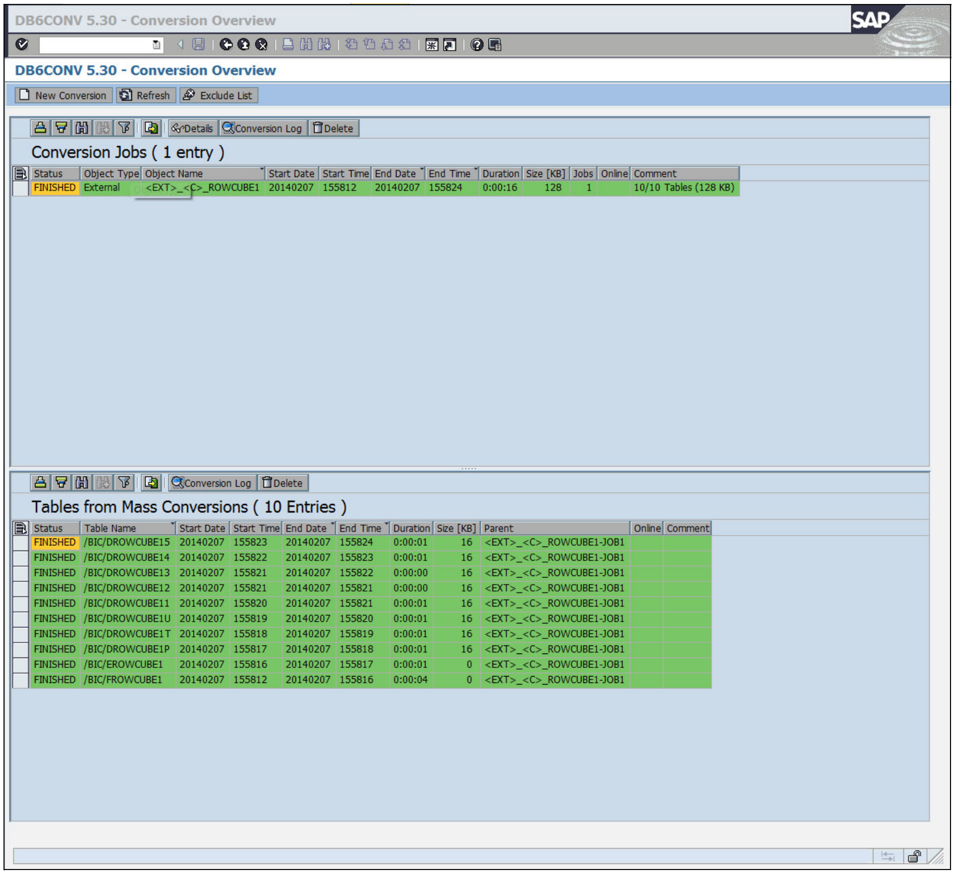



*Figure 8-114 Report DB6CONV: Successfully finished conversion job*

## Considerations for InfoCube conversions

DB6CONV provides options for excluding tables that are contained in conversion jobs from actually being converted. It is best *not* to use the options to exclude tables to make sure that all tables belonging to an InfoCube are converted.

When a conversion aborts, take the following actions:

1. For each table that failed during the conversion, examine the conversion log to identify the root cause of the failure. If the root cause is the issue described in "Known issue with InfoCube conversions" on page 376, apply the fix or the workaround and continue with the next step. If the issue is not listed, open an incident at SAP.
2. Reset the conversion of each table that failed in the *Table from Mass Conversions* area in the *Conversion Overview* window.
3. Reset the conversion of the whole job in the Conversion Jobs area.
4. Go back to the SAP_CDE_CONVERSION_DB6 report, reselect the InfoCube or all InfoCubes and create another conversion job. This new job contains only the tables that failed or have not yet been processed.
5. Schedule and run the new conversion job.

### Known issue with InfoCube conversions

When you convert column-organized tables to row-organized, the row-organized tables are not compressed. The compression options in the job details window cannot be changed to row or adaptive compression and the default option, *KEEP*, is not applied correctly. This issue can be corrected by activating compression using the DB6CONV report. Make sure that you have installed the most recent version of DB6CONV.

## 8.7 Deployment

DB2 Cancun Release 10.5.0.4 brings a lot of improvements and enables additional features. This makes it a preferred release, especially for new deployments. To get your SAP BW system ready for using BLU Acceleration, three options are available:

- If your SAP BW system runs on DB2 but currently uses DB2 10.1 or earlier, use these steps:
  a. Upgrade your DB2 database to DB2 10.5 as described in the SAP database upgrade guide (*Upgrading to Version 10.5 of IBM DB2 for Linux, UNIX, and Windows*[5]).

     This guide is at the SAP Service Marketplace:

     http://service.sap.com/instguides

     Select **Database Upgrades** → **DB2 UDB**.

[5] See http://service.sap.com/~sapidb/011000358700000843422013E

b. Adapt your DB2 parameter settings (as described in *SAP Note 1851832*), including the parameter settings for BLU Acceleration that are described in the “DB2 Column-Organized Tables (BLU Acceleration)” section of *SAP Note 1851832.*

c. Install the required Support Packages and SAP Kernel described in chapter 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255.

d. Install the required code corrections described in chapter 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255.

e. Configure the SAP BW system for DB2 BLU Acceleration by setting the RSADMIN parameters.

- If you want to install a completely new SAP BW system, use these steps:

a. Install the system directly on DB2 10.5 with the SAP Software Logistics (SL) Toolset. Be sure to use the most recent release and Support Package Stack.

b. Adapt the DB2 parameter settings manually as described in the *DB2 Column-Organized Tables (BLU Acceleration)* section of *SAP Note 1851832*.

c. If necessary, upgrade to the required Support Packages and SAP Kernel (see 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255).

d. Install the required code corrections described in chapter 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255.

e. Configure the SAP BW system for DB2 BLU Acceleration by setting the RSADMIN parameters.

- If you have an SAP BW system that runs on DB2 or another database platform supported by SAP and you want to create a new BW system by performing a heterogeneous system copy (migration) of the existing system, proceed as follows:

a. Upgrade your existing system to the required Support Package level and SAP kernel version (see 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255).

b. Install the required code corrections described in chapter 8.2, “Prerequisites and restrictions for using BLU Acceleration in SAP BW” on page 255.

c. Generate the DDL for the SAP BW tables for the target DB2 database with the SMIGR_CREATE_DDL report. In this step, you can either select to directly generate DDL for creating column-organized tables or you can choose row-organized tables and convert selected SAP BW objects to BLU Acceleration later in the target system.

d. Perform the heterogeneous system copy as described in the SAP documentation and install the target system directly on DB2 10.5.

e. Before you start loading the data into the target system, adapt the DB2 parameter settings manually as described in the *DB2 Column-Organized Tables (BLU Acceleration)* section of *SAP Note 1851832*.

f. Configure the SAP BW system for DB2 BLU Acceleration by setting RSADMIN parameters.

g. Perform the required post migration steps including the execution of the RS_BW_POST_MIGRATION report in the target system.

### 8.7.1 Upgrading an SAP BW system to DB2 10.5

The procedure to upgrade an SAP database to DB2 10.5 is described in the SAP database upgrade guide (*Upgrading to Version 10.5 of IBM DB2 for Linux, UNIX, and Windows*). This is mentioned in 8.7, "Deployment" on page 376.

After you have performed the database upgrade and before you start to create BLU Acceleration InfoCubes or convert existing InfoCubes to column-organized tables, you must set the DB2 configuration parameters to the recommended values for BLU Acceleration of *SAP Note 1851832* manually and restart the DB2 instance.

The most important configuration parameter settings for BLU Acceleration that are described in *SAP Note 1851832* are as follows:

- Database Manager configuration parameters
  - INTRA_PARALLEL = YES
  - MAX_QUERYDEGREE = ANY
- Database configuration parameters
  - INSTANCE_MEMORY >= 64 GB
  - DATABASE_MEMORY = AUTOMATIC
  - SHEAPTHRES_SHR = 40% of the instance memory
  - SORTHEAP = 1/20 of SHEAPTHRES_SHR
  - UTIL_HEAP_SZ = AUTOMATIC

- Buffer pool size = 40% of instance memory
- To benefit from the parallel processing capabilities of BLU Acceleration, at least eight processor cores are needed.

The ratio of 8 GB RAM per processor core ensures a high performance on larger systems as well. Development or quality assurance systems with a minor amount of parallel workload are able to use the parallel processing capabilities of BLU Acceleration with at least four processor cores and 32 GB RAM.

If you want to create SAP BW temporary tables, PSA tables, and InfoObject tables as column-organized tables, you must set the RSADMIN parameters shown in Table 8-7 with report SAP_RSADMIN_MAINTAIN.

*Table 8-7 SAP BW RSADMIN parameters for DB2 BLU Acceleration*

| RSADMIN Parameter Name and Value | Comment |
|---|---|
| DB6_IOBJ_USE_CDE=YES | Create SID and time-independent and time-dependent attribute SID tables of new characteristic InfoObjects as column organized tables |
| DB6_PSA_USE_CDE=YES | Create new PSA tables as column-organized tables |
| DB6_TMP_USE_CDE=YES | Create new BW temporary tables as column-organized tables |

Make sure that you have implemented the latest version of *SAP Note 1152411* for the SAP DBA Cockpit. Then create at least one BLU table in your system, for example by creating a new column-organized InfoCube.

After these steps run the latest available version (at least V44) of the `db6_update_db` script. The script checks whether at least one BLU table exists. If this is the case it deletes the SAP specific Workload Manage (WLM) service classes and workloads and sets a WLM threshold is set for the number of long running BW queries on BLU that can be executed in parallel. With this threshold you avoid overloading the CPU resources of the database server. The most recent version of the script is in the attachment of *SAP Note 1365982* in the “DB6: Current db6_update_db/db2_update_client Script (V45)”. The version number changes over time.

## 8.7.2 Installing a new SAP BW system on DB2 10.5

The installation of a new SAP BW system does not involve the creation of InfoCubes or other SAP BW objects. Before you start creating BW objects that use column-organized tables, proceed by setting the DB2 configuration parameters as it is done during an update process, make sure that the SAP DBA Cockpit is up to date, create at least one BLU table and run the `db6_update_db` script. These steps are described in 8.7.1, “Upgrading an SAP BW system to DB2 10.5” on page 378.

If you want to create SAP BW temporary tables, PSA tables, and InfoObject tables as column-organized tables you must set the RSADMIN parameters shown in Table 8-7 with report SAP_RSADMIN_MAINTAIN.

## 8.7.3 Migrating an SAP BW system to DB2 10.5 with BLU Acceleration InfoCubes

The procedure to perform a heterogeneous system copy (also called *migration*) of an SAP NetWeaver system is described in the SAP system copy guide that is available on SAP Service Marketplace:

http://service.sap.com/instguidesnw

Select **<Your SAP NetWeaver Main Release>** → **Installation**.

With *SAP Note 1969500*, you have two options when you perform an SAP heterogeneous system copy. You can choose whether you want to migrate your InfoCubes only or all BW objects for which BLU Acceleration is supported to column-organized tables. This works when your SAP BW system already runs on DB2 or when you migrate your SAP BW system from another database platform to DB2.

Figure 8-115 shows the steps for the source system and the target system that must be completed during a heterogeneous system copy of an SAP BW system.

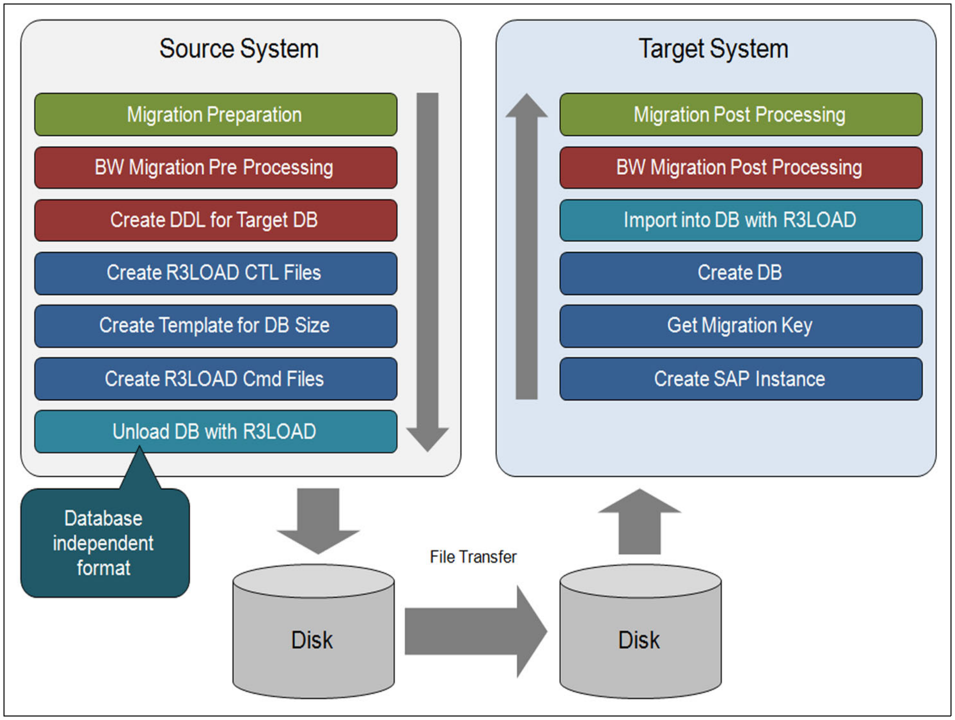



*Figure 8-115 Steps to perform an SAP BW heterogeneous system copy*

The steps in the red boxes are specific to SAP BW systems. The most important step to ensure that BW object tables are created as column-organized tables in the target system is “Create DDL for Target DB.” In this step, you run the SMIGR_CREATE_DDL report on the source system to generate DDL statements for creating tables and indexes for SAP BW objects in the target database.

With *SAP Note 1969500*, two new options for creating DDL for column-organized tables are available in the report.

Complete these steps to create the DDL files needed for the migration:

1. Run the SMIGR_CREATE_DDL report in SAP transaction SE38.
2. Select **DB6** as the target database (Figure 8-116 on page 382).

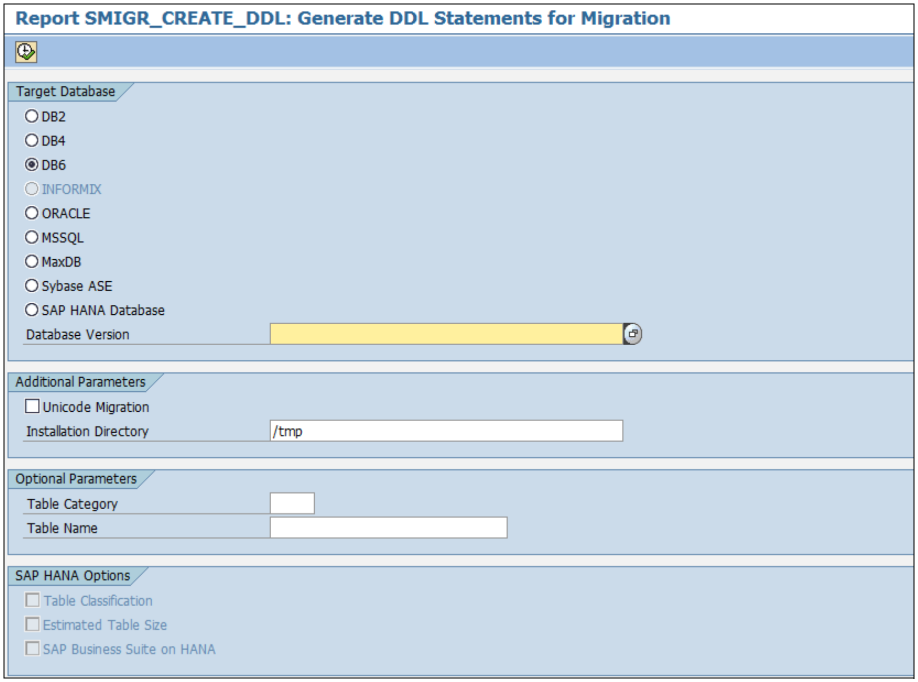



*Figure 8-116 Report SMIGR_CREATE_DDL: main window*

The Database Version field provides the following options:

– When your source system runs on DB2, the options shown in Figure 8-117 are offered.

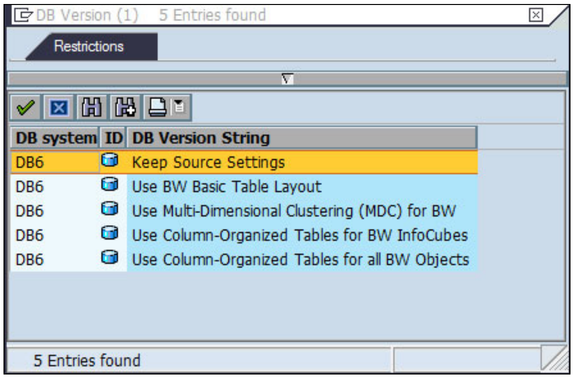



*Figure 8-117 DB2 feature options for a heterogeneous system copy of an SAP BW system that runs on DB2*

The options are as follows:

- *Keep Source Settings*: The report generates the same DDL for the target database that is used in the source database.
- *Use BW Basic Table Layout:* The report generates DDL without MDC and DDL for row-organized tables only.
- *Use Multi-Dimensional Clustering (MDC) for BW*: The report generates DDL with MDC for all PSA tables, DSO change log tables, DSO activation queue tables, F fact tables, E fact tables that already use MDC in the source system, and DSO active tables that already use MDC in the source system.
- *Use Column-Organized Tables for BW InfoCubes*: The report generates DDL for column-organized tables for the InfoCube fact tables and dimension tables. For the other SAP BW object tables, it generates DDL with MDC if possible.
- *Use Column-Organized Tables for all BW Objects*: The report generates DDL for column-organized tables for InfoCube fact tables, InfoCube dimension tables, DSO active tables, PSA tables, and InfoObject SID and attribute SID tables. For the other SAP BW object tables, it generates DDL with MDC if possible.

– If your source system runs on another database platform, the options shown in Figure 8-118 are offered.

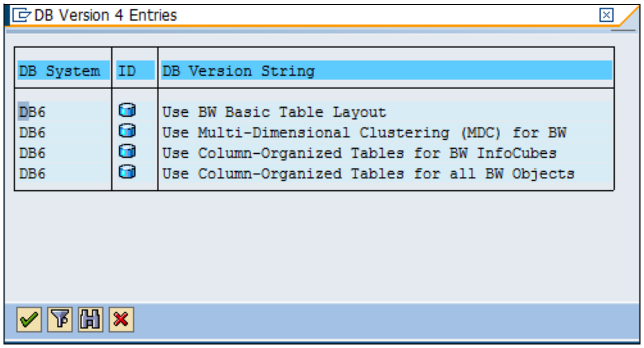



*Figure 8-118 DB2 feature options for a heterogeneous system copy of an SAP BW system that runs on another database platform*

The options are as follows:

- *Use BW Basic Table*: The report generates DDL without MDC and DDL for row-organized tables only.

- *Use Multi-Dimensional Clustering (MDC) for BW*: The report generates DDL with MDC for all PSA tables, DSO change log tables, DSO activation queue tables, F fact tables, E fact tables that use range partitioning in the source system, and DSO active tables that use range partitioning in the source system.
- *Use Column-Organized Tables for BW InfoCubes*: The report generates DDL for column-organized tables for the InfoCube fact tables and dimension tables. For the other SAP BW object tables, it generates DDL with MDC if possible.
- *Use Column-Organized Tables for all BW Objects*: The report generates DDL for column-organized tables for InfoCube fact tables, InfoCube dimension tables, DSO active tables, PSA tables, and InfoObject SID and attribute SID tables. For the other SAP BW object tables, it generates DDL with MDC if possible.

3. Choose **Use Column-Organized Tables for all BW Objects**, select the output directory for the generated DDL in the Installation Directory field, and run the report (Figure 8-119).

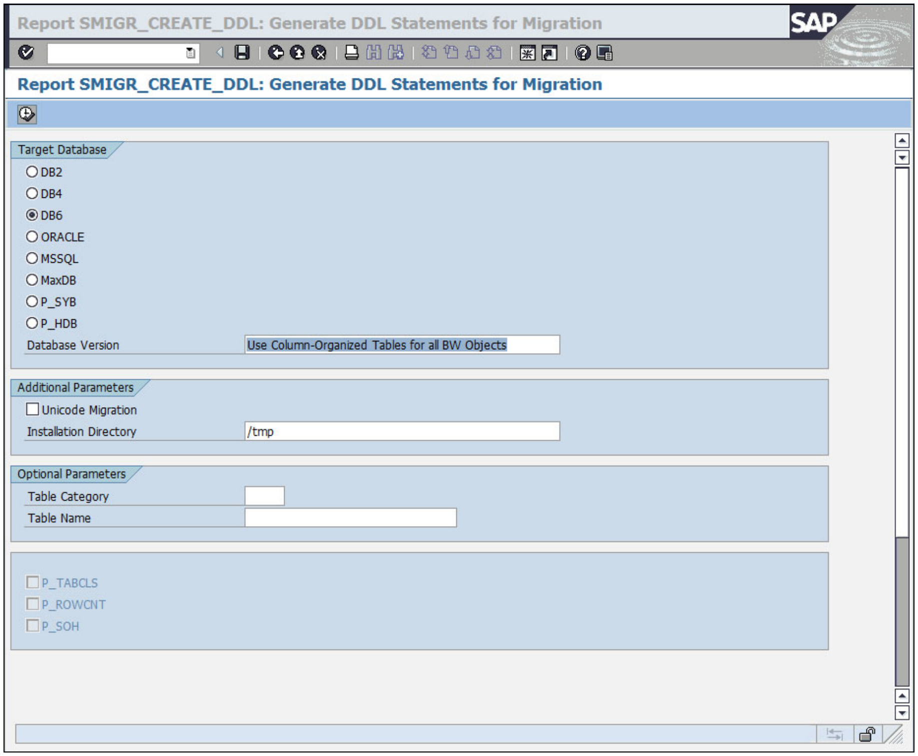



*Figure 8-119 Report SMIGR_CREATE_DDL: Generating DDL for column-organized tables*

When the report is finished, the success message is shown (Figure 8-120).

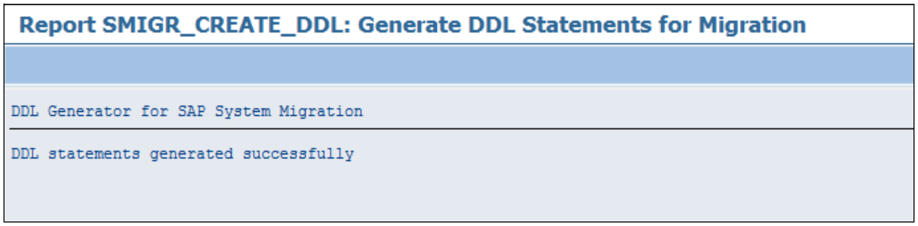



*Figure 8-120 Report SMIGR_CREATE_DDL: Final message when completed*

4. In the directory that you entered in *Installation Directory*, locate these files:
   - Generated DDL
   - The `*.LST` file that contains the list of the files that the report created
   - The `db6_compression_tablist.txt` file (see Figure 8-121)

   The `db6_compression_tablist.txt` file is generated only when the source system runs on DB2. This file contains the compressed row-organized tables and the row compression type (static or adaptive) that is used. You can use this file as input when you generate and load your DB2 target database to compress the tables in exactly the same way as in the source database.

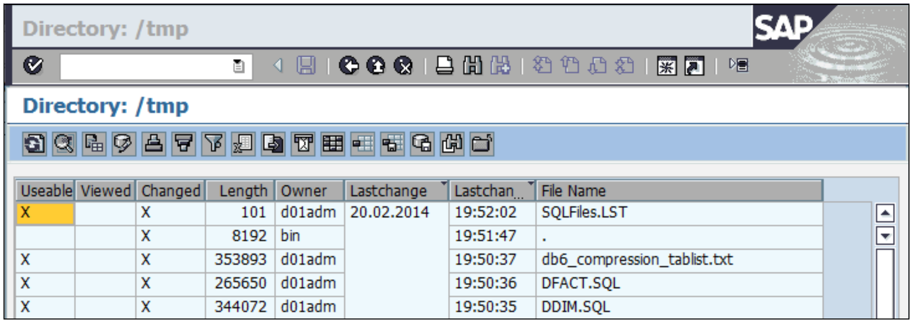



*Figure 8-121 Directory with the sample files generated by report SMIGR_CREATE_DDL*

   The SMIGR_CREATE_DDL report generates a `*.SQL` file for each SAP data class that contains at least one table with DB2-specific features such as MDC, distribution keys, clustered indexes, or column-organized tables.

5. During the installation of the target system, the SAP installation tool asks for the location of the files. Make sure that the files are accessible.

   Example 8-13 shows output of the SMIGR_CREATE_DDL report for the E fact table, `/BIC/EROWCUBE1`, of a row-organized InfoCube that is to be created with column-organized tables in the target system. A `tab` entry is for the table itself, several `ind` entries are for the indexes that exist on the row-organized table in the source system.

The entries for most of the indexes, for example, `/BIC/EROWCUBE1~020`, contain no SQL statement because these indexes are not created in the target database. The entry for index `/BIC/EROWCUBE1~P` contains an `ALTER TABLE ADD CONSTRAINT` statement that must be used for column-organized tables.

*Example 8-13 Sample output of report SMIGR_CREATE_DDL*

```
tab: /BIC/EROWCUBE1
sql:     CREATE TABLE "/BIC/EROWCUBE1"
        ("KEY_ROWCUBE1P" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE1T" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE1U" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE11" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE12" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE13" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE14" INTEGER
        DEFAULT 0 NOT NULL,
        "KEY_ROWCUBE15" INTEGER
        DEFAULT 0 NOT NULL,
        "SID_0CALMONTH" INTEGER
        DEFAULT 0 NOT NULL,
        "/B49/S_CRMEM_CST" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_CRMEM_QTY" DECIMAL(000017,000003)
        DEFAULT 0 NOT NULL,
        "/B49/S_CRMEM_VAL" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_INCORDCST" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_INCORDQTY" DECIMAL(000017,000003)
        DEFAULT 0 NOT NULL,
        "/B49/S_INCORDVAL" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_INVCD_CST" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_INVCD_QTY" DECIMAL(000017,000003)
        DEFAULT 0 NOT NULL,
        "/B49/S_INVCD_VAL" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_OPORDQTYB" DECIMAL(000017,000003)
        DEFAULT 0 NOT NULL,
        "/B49/S_OPORDVALS" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
        "/B49/S_ORD_ITEMS" DECIMAL(000017,000003)
        DEFAULT 0 NOT NULL,
        "/B49/S_RTNSCST" DECIMAL(000017,000002)
        DEFAULT 0 NOT NULL,
```

```
    "/B49/S_RTNSQTY" DECIMAL(000017,000003)
    DEFAULT 0 NOT NULL,
    "/B49/S_RTNSVAL" DECIMAL(000017,000002)
    DEFAULT 0 NOT NULL,
    "/B49/S_RTNS_ITEM" DECIMAL(000017,000003)
    DEFAULT 0 NOT NULL)
    IN "&location&"
    INDEX IN "&locationI&"
    LONG  IN "&locationL&"
  DISTRIBUTE BY HASH (
  "KEY_ROWCUBE11"
  , "KEY_ROWCUBE12"
  , "KEY_ROWCUBE13"
  , "KEY_ROWCUBE14"
  , "KEY_ROWCUBE15"
  , "KEY_ROWCUBE1T"
  , "KEY_ROWCUBE1U"
  )
  ORGANIZE BY COLUMN;

ind: /BIC/EROWCUBE1~0

ind: /BIC/EROWCUBE1~020

ind: /BIC/EROWCUBE1~040

ind: /BIC/EROWCUBE1~050

ind: /BIC/EROWCUBE1~060

ind: /BIC/EROWCUBE1~070

ind: /BIC/EROWCUBE1~080

ind: /BIC/EROWCUBE1~P
sql:     ALTER TABLE "/BIC/EROWCUBE1"
      ADD CONSTRAINT "/BIC/EROWCUBE1~P" unique
         ("KEY_ROWCUBE1T",
          "KEY_ROWCUBE11",
          "KEY_ROWCUBE12",
          "KEY_ROWCUBE13",
          "KEY_ROWCUBE14",
          "KEY_ROWCUBE15",
          "KEY_ROWCUBE1U",
          "KEY_ROWCUBE1P");
```

6. Before you start to load the database, set the DB2 configuration parameters as described in 8.7.1, “Upgrading an SAP BW system to DB2 10.5” on page 378.
7. After you have installed the target database, created the tables, and loaded the data, you will have BLU tables in the database.

8. Make sure that you have implemented the latest version of *SAP Note 1152411* for the SAP DBA Cockpit.
9. Run the latest available version (at least V44) of the **db6_update_db** script. The script checks whether at least one BLU table exists. If this is the case it deletes the SAP specific service classes and workloads and sets a WLM threshold is set for the number of long running BW queries on BLU that can be executed in parallel. With this threshold you avoid overloading the CPU resources of the database server. The most recent version of the script is in the attachment of *SAP Note 1365982* in the "DB6: Current db6_update_db/db2_update_client Script (V45)". The version number changes over time.
10. If you want to create SAP BW temporary tables, PSA tables, and InfoObject tables as column-organized tables you must set the RSADMIN parameters shown in Table 8-7 with report SAP_RSADMIN_MAINTAIN.
11. Run the RS_BW_POST_MIGRATION report to complete the migration of the SAP BW object tables.

# 8.8 Performance of SAP BW with BLU Acceleration

This section describes the performance benefits of BLU Acceleration in an SAP BW environment. An internal lab test is used for illustration purposes. This test is based on a performance test scenario adopted to meet the characteristics observed in customer scenarios as closely as possible. By this example, the most important attributes of the column-store implementation are highlighted, which have a significant effect as to improved performance, reduced resource consumption, and enhanced usability.

## 8.8.1 Example environment

The starting point for these measurements is a basic cumulative InfoCube that is provided by the performance test. This InfoCube references 75 characteristics, in total. Those characteristics are spread on 16 dimensions. Each dimension holds 2 - 11 characteristics. This InfoCube definition is copied three times: once using *MDC* as a clustering method and twice using *Organized in Columns* to create a standard and a flat InfoCube.

### Preparation

The data used to load the InfoCubes for this scenario is generated. The data produced by the generator represents transactional data of five years. This data is stored in plain text files containing $10^8$ rows in total. These files are loaded to a PSA, which in turn represents the source for Data Transfer Processes (DTPs) that load the data into the InfoCubes.

As a result, both InfoCubes are loaded with exactly the same data. The fact tables of each InfoCube contain $10^8$ rows. The cardinality of the resulting dimension tables varies between a few hundred and $11*10^6$ rows, depending on the number of characteristics and their cardinality. The dimensions, grouped by their cardinality, are distributed in the following way:

- Six small dimensions contain less than 10,000 rows.
- Four medium dimensions contain more than 10,000 but less than 100,000 rows.
- Six large dimensions contain more than 100,000 records.

From the application server point of view, the process of creating a column-organized InfoCube using BLU Acceleration is nearly the same as creating one in the row store. See “Table layout of column-organized InfoCubes” on page 279. A flat InfoCube is created as described in 8.5.2, “Flat InfoCubes in SAP BW 7.40” on page 297. The load process is exactly the same. A common runtime layer for both data stores enables DB2 to provide a transparent interface to the application. After a table is created using BLU Acceleration, it is accessed in the same way as a conventional table. This brings two fundamental benefits compared to less integrated architectures:

- Existing programs and processes need no, or only minimal, adoptions to use BLU Acceleration.
- Column- and row-organized InfoObjects can coexist without any special treatment within the application allowing an easy migration to BLU Acceleration.

## Configuration

The hardware used for these query measurements is an IBM Power 770. A logical partition (LPAR) is configured with eight Power7 cores at 3.1 GHz each and 64 GB of RAM running AIX 7.1. This LPAR runs both, the SAP BW application server and the DB2 Cancun Release 10.5.0.4database server. An IBM XIV is used as storage system.

These measurements aim for a direct comparison of row and column store within the same database instance. For that reason, a common configuration is used, which fits to columnar stored objects and to objects in the row store. Because this lab configuration is a two-tier setup, the database cannot dispose of the RAM completely. Instead, 9 GB of RAM are reserved for the operating system and the application server. The remaining 55 GB of RAM are assigned to the database server.

The database memory is mainly consumed by two memory areas: the buffer pool and the sort heap threshold for shared sorts. Each is set to 22 GB. The utility heap size and the sort heap size for private sorts are set to 2 GB. Intrapartition parallelism is enabled with unlimited maximum degree (according to the recommendations in 8.7.1, “Upgrading an SAP BW system to DB2 10.5” on page 378).

Additional configuration for BLU Acceleration is not required. Because BLU Acceleration is a memory-optimized, but not an in-memory column store technology, it uses data pages to organize and buffer data, just as the conventional row store does. This architectural approach has several advantages, including these:

- Neither a table, nor even column has to fit completely into RAM. Frequently used pages reside in buffer pools and less frequently used ones can be dynamically read from disk. Thus table and column sizes are not limited by the available amount of RAM.
- DB2 is able to share buffers between both data stores. This simplifies the configuration because no additional parameters have to be maintained. Furthermore, it reduces the over provisioning needed to cope with peak high load phases.

## 8.8.2 Storage consumption and compression rates

To make a statement on relative storage consumptions or compression rates, a reference size is required. Both the row store and column store use compression techniques, but differ from each other in terms of indexes and constraints. As a consequence, a direct comparison is not meaningful.

Therefore, both storage saving mechanisms and the impact of the denormalization of the flat InfoCube must be compared against a neutral element. However, in each case, the storage actually used by tables and indexes must be considered separately. For that purpose, a fourth, uncompressed instance of the InfoCube is created.

Regarding compression and storage type, there are now four differently configured cubes:

**ROW uncompressed** No compression technique is used.

**ROW compressed** Page level compression, adaptive compression, and index compression are used.

**BLU Acceleration** On column-organized tables, a new, order-preserving compression algorithm is activated and maintained automatically.

**Flat InfoCube** The same attributes as for the BLU accelerated standard InfoCube apply, because this InfoProvider is only available with BLU Acceleration.

Table 8-8 lists the actual storage consumptions of the cubes. For each cube, the total consumption is horizontally split into the size of fact and dimension tables. Vertically, the consumption is split into storage consumed by actual table data and storage consumed by indexes and similar structures.

*Table 8-8 Storage consumption of InfoCube objects split up into their components*

| | Fact table | Dimension tables | Total |
|---|---|---|---|
| **ROW uncompressed** | **42.23 GB** | **3.19 GB** | **46.42 GB** |
| Tables | 32.47 GB | 0.93 GB | 33.40 GB |
| Indexes | 10.77 GB | 2.26 GB | 13.03 GB |
| **ROW compressed** | **17.50 GB** | **1.99 GB** | **19.34 GB** |
| Tables | 11.02 GB | 0.49 GB | 11.50 GB |
| Indexes | 6.33 GB | 1.51 GB | 7.83 GB |
| **BLU Acceleration** | **4.99 GB** | **1.16 GB** | **6.15 GB** |
| Tables | 4.97 GB | 0.15 GB | 5.12 GB |
| Indexes | 0.01 GB | 1.01 GB | 1.02 GB |
| Synopsis | 0.01 GB | 0.00 GB | 0.01 GB |
| **Flat InfoCube** | **5.59 GB** | **0.00 GB** | **5.59 GB** |
| Tables | 5.57 GB | 0.00 GB | 5.57 GB |
| Indexes | 0.01 GB | 0.00 GB | 0.01 GB |
| Synopsis | 0.01 GB | 0.00 GB | 0.01 GB |

In total, conventional compression methods on the row store reduce the total storage consumption by a factor of 2.4x. Using BLU Acceleration, this ratio is increased to a factor of 7.5x. In case of the flat InfoCube, the data volume is reduced by even factor 8.3.

Comparing the compression rates of fact and dimension tables, the compression rate on the row store is constant on both table types. The column-organized tables, however, in total vary in their compression rate depending on their type. The fact table reaches a compression rate of 8.5x but the dimension tables are compressed by factor 2.8x.

On closer consideration of the dimension tables, actual table and index sizes, their ratio attracts attention. The indexes are by factors larger than the actual data. This also applies to the uncompressed InfoCube. Thus, this fact does not result from compression. The reason is that for dimension tables in the row store, an index is defined on each column. The data is actually stored twice, in the table and in the index as well. In addition, the index tree needs some storage, too, which makes this ratio reasonable. In the case of column-organized dimension tables, only two constraints are defined: The primary key constrained on the dimension identifier column, and a unique key constrained on all columns. The second one represents a unique index. According to that, a ratio of index and table size larger than one in the column store is also reasonable.

Although the compressed indexes on the columnar dimension tables are already 1.2 times smaller than those of the row-based tables, the actual data in the columnar dimension tables is still 6.7 times smaller than their own indexes. This is because indexes, in general, are more difficult to compress than data. In this particular case, the index compression rate on column-organized tables is already high. Because of the advantages of a column store regarding compression capabilities and the new compression algorithm used by BLU Acceleration, the data compression is even better.

The effect of the increased redundancy of the flat InfoCube data, caused by the denormalization of the enhanced star schema, is far smaller than the benefit gained by abandoning the indexes on dimension tables and dropping the SID columns required for modeling the foreign key relationship. This makes the flat InfoCube even more space efficient than the standard InfoCube with BLU Acceleration.

In summary, the following statements can be made:

- With the use of BLU Acceleration, a compression rate of 7.5x is reached in this test scenario.
- The new flat InfoCube with BLU Acceleration even reaches a compression rate of 8.3x.
- Even compared to the most sophisticated row-based compression features, the storage consumption can be reduced by 3.2-3.5x because of the advanced possibilities of the new columnar compression algorithm.
- Assuming that dimension tables in the field have a cardinality of less than 10% of the number of fact table entries, even higher compression rates of InfoCubes are possible.

### 8.8.3 Query performance

To make a general assumption about the query performance of SAP BW using DB2 BLU Acceleration, a set of queries is used that satisfy several attributes.

The queries differ from each other in the following terms:

- The queries select either 20% or 40% of the transactional data.
- The predicate is either a range predicate on the time dimension, an access using a hierarchy, a predicate on a user defined dimension.
- Three different user defined dimensions are used: A small one with 10^6 rows, a medium one with 6*10^6 rows and a large one with 11*10^6 rows.
- There are three kinds of predicates defined on these user defined dimensions: Range predicates, lists, or they are accessed using temporary tables which are created for predicates including a large list.

This results in a set of 22 queries. For simplicity, the queries are labeled from 1 to 22, based on the order of the attributes. The SAP Business Explorer (BEx) Query Designer is used to define these queries as BEx queries, which form the basic instrument to query SAP BW data.

These BEx queries are designed to compare database performance only. Therefore, the OLAP cache is deactivated during the measurements. So the query processing almost exclusively occurs on the database server. This also ensures that no precalculated results are used.

By shifting the workload from the application server to the database, the effect on the query performances by running the application server on the same LPAR as the database server is minimized also, because the application server is nearly idle while the queries are run.

The database is configured to provide enough memory for the buffer pool so that each InfoCubes and temporary results completely fit into memory. All queries are run twice, recording only the second result to make sure that no I/O effects are distorting the result. This approach ensures reproducible results by eliminating I/O, a hardly predictable factor, which has a great influence on database performance. Furthermore, it assures that query processing on both technologies, the conventional row store, and BLU Acceleration, are compared under the same conditions.

However, two important advantages of BLU Acceleration are implicitly left out by these measurements:

- BLU Acceleration provides far better compression capabilities than the row store. As a result, I/O costs and the amount of buffer pool memory that is required to buffer data are reduced. To make the row-based tables fit into the buffer pool, more memory must be provided than the tables with BLU Acceleration require.
- By using BLU Acceleration, only the data that is actually used by the query must be read. Columns that are not required to execute the query are never read.

As a consequence, these measurements compare pure processing time only. The results of these measurements are outlined in Figure 8-122.

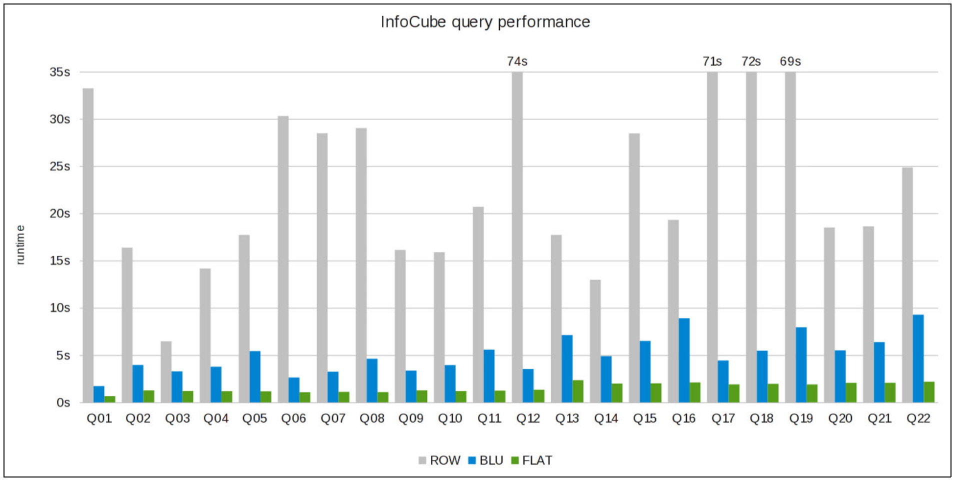



*Figure 8-122 Query performance results comparing conventional row store performance with queries using BLU Acceleration*

On average, BLU Acceleration on a standard InfoCube outperforms the row-based sample queries by factor 6.9x. On a flat InfoCube, BLU Acceleration is even 20.6x faster on average than a standard row-based InfoCube. The ratio varies depending on the query attributes from 2x to 55x. Customers report an even larger performance gain by using BLU Acceleration on real life data.

## Scalability and throughput

Besides the static analysis of single query performance, scalability of BLU Acceleration is also investigated. For that purpose, scalability is analyzed in two dimensions:

- A growing amount of analyzed data
- An increasing number of parallel running queries

Starting with the performance behavior caused by a growing amount of data, a single query is chosen as a starting point. The query is modified to fetch data starting from 0% up to 100% of the InfoCube. The resulting 21 queries are used to make a prediction on the engine's scalability.

The run times that are achieved in this test are shown in Figure 8-123. Besides the actual measured points, this chart shows linear approximations of both measurement series. The row store and BLU Acceleration show an almost linear scaling behavior. The obvious performance advantage through BLU Acceleration makes it hard to show this linearity in the chart. A detailed analysis of the measured run times shows a similarly low deviation from the approximation as the measurements on the row store do.

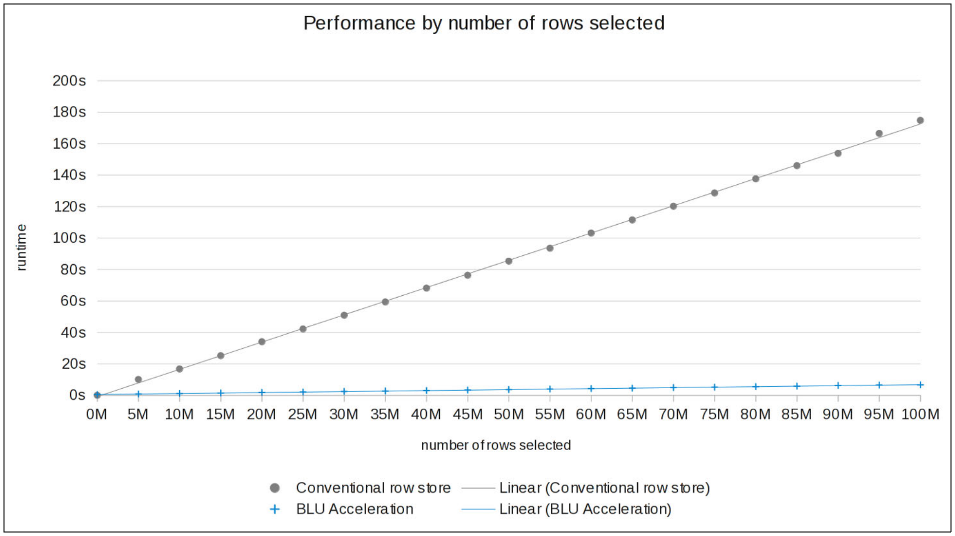



*Figure 8-123 Performance scalability regarding the amount of selected data*

To measure parallel query throughput, some additional aspects must be considered. Especially synergy effects between queries should be minimized. For that purpose, the following setup is used:

- Eight InfoCubes are set up as data source for the queries.
- Up to 24 query sets are run against each cube in parallel.
- Each of those sets contains the same queries but in different order.

In an environment where multiple queries are run in parallel, not all data is expected to be in the buffer pool. Thus the buffer pool is configured to provide only enough memory for one-third of the actual size all InfoCubes would need in total.

The first measurement with only one query set running at a time forms a reference. In further measurements up to 192 query sets are run against all eight cubes in parallel.

All query sets are run several times in a loop. The first and the last iterations are dropped in each case. By taking the run times of the remaining iterations, an average throughput can be computed, which describes how many queries can be run per hour. These results are outlined in Figure 8-124.

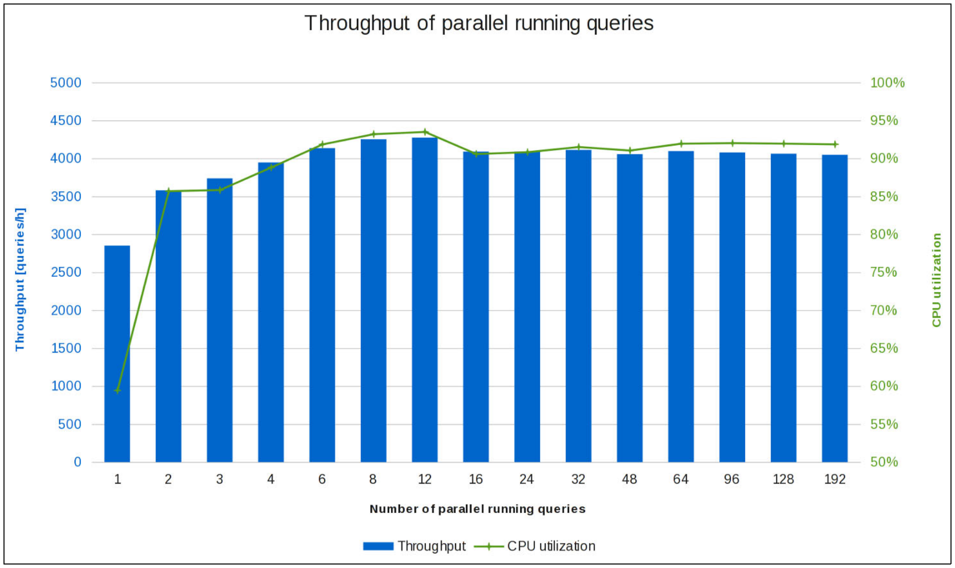



*Figure 8-124 Average query throughput of parallel running queries*

The results show that the actual throughput is slightly increasing. Monitoring the operating system when the tests are running shows a CPU utilization around 90% in all cases. So, at least three interpretations can be derived from these observations:

- The parallel processing of SAP BW queries using BLU Acceleration is capable of utilizing almost all available processors.
- Even if multiple queries run in parallel and the available resources must be shared, the throughput remains constant.
- The system remains responsive, even under a high, parallel workload.

In summary, both tests show that DB2 with BLU Acceleration scales well regarding a growing amount of data and with an increasing degree of parallelism. The linear scaling behavior with an increasing data volume results from the optimized parallel query processing, which allows a high utilization of available resources. Furthermore, from the use of SIMD instructions, the parallel processing within a process is supported, additionally. At the same time, a high cache utilization is reached, which enables the processor to keep fully utilized and reduces the number of context switches. This makes BLU Acceleration stable in terms of parallel workload.

## Access plans

Performance reliability and stability depend significantly on access plans that are generated by the query optimizer. During query compilation time, the optimizer must make decisions on join order, join types and which indexes are to be used.

The waiver of explicit indexes and several join types using BLU Acceleration reduces the degree of freedom for possible decisions of the optimizer and thus increases the reliability of access plans. Accessing dimension tables using a full table scan instead of an index scan is accelerated in several ways.

Synopsis tables hold additional information about sets of tuples, describing per column which value ranges are included. This supports skipping complete sets of tuples if the searched value is not included. In contrast to indexes, synopsis tables are created automatically. In addition, actionable compression allows scanning the table without having to decompress it first. This does not only save memory but also computing time and enables DB2 to compare more values immediately by using SIMD instructions.

Example 8-14 shows two excerpts of access plans joining a dimension table to a fact table. The left is an example using the conventional row store; the right is an example using BLU Acceleration. As this example shows, column-organized tables are labeled as `CO-TABLE`. Data skipping is applied implicitly by the table scan operator. Therefore, it is not mentioned separately.

*Example 8-14 Excerpts of access plans joining a dimension table to the fact table compare conventional row store to BLU Acceleration*

```
[upper part of the access plan skipped in the intrests of clarity]
Conventional row store:                     BLU Acceleration:


           9.48521e+07                                  9.47975e+07
             ^HSJOIN                                      ^HSJOIN
             (  12)                                       (  10)
           2.61552e+06                                  1.23391e+06
             721997                                        31872
        /------+-------\                              /-------+-------\
     1e+08          1731                          1e+08              1731
    TBSCAN          FETCH                         TBSCAN             TBSCAN
    (  13)          (  14)                        (  11)             (  12)
  2.60943e+06       36.5425                     1.23279e+06          59.1637
    721992            5                           31863.5            8.52632
      |            /---+----\                        |                  |
     1e+08      1826        1826                  1e+08               1826
TABLE: SAPBL2    IXSCAN   TABLE: SAPBL2     CO-TABLE: SAPBL2    CO-TABLE: SAPBL2
 /BIC/FICROW02   (  15)   /BIC/DICROW02T       /BIC/FICBLU02      /BIC/DICBLU02T
      Q4        14.7311        Q6                  Q2                 Q4
                   2
                  |
```

```
        1826
INDEX: SAPBL2
/BIC/DICROW02T~0
         Q6
```

## 8.8.4 Discussion

As outlined by the tests in the previous sections, SAP BW InfoCubes on DB2 with BLU Acceleration outperform conventional row-based InfoCubes regarding storage consumption and query performance.

Additional internal lab tests (not outlined in this document) show a comparable ETL performance. The compression rate of 8.3x is by factor of 3.5x better than on the row store. Even running completely in memory, the query performance is up to 22x better. With I/O involved, even higher ratios are expected because of the storage model.

BLU Acceleration is a true column store. That means the complete internal representation and the engine internal processing is columnar. Other column-oriented storage models, which do not use an in-memory architecture, such as Decomposition Storage Model (DSM)[6] and Partition Attributes Across (PAX)[6] are easy to implement into an existing engine, but their architectures have some major drawbacks (see Table 8-9).

*Table 8-9 Storage model comparison*

| | NSM[a] | DSM | PAX | BLU Acceleration |
|---|---|---|---|---|
| Spatial locality | | X | X | X |
| Cheap record construction | X | | X | X |
| I/O Avoidance | | X[b] | | X |

a. N-ary Storage Model (NSM)/row store
b. Additional I/O to fetch row key

DSM is basically a row store with an extended implementation of vertical partitioning. PAX organizes data in column-oriented style at page level only. In this way, a cache pollution can be admittedly reduced, SIMD instructions can be applied and records cheaply reconstructed, but nevertheless all data must be read to process a query referencing the table.

[6] Anastassia Ailamaki, David J. DeWitt, Mark D. Hill, Marios Skounakis, *Weaving Relations for Cache Performance,* Morgan Kaufmann Publishers Inc., 2001, ISBN:1-55860-804-4

BLU Acceleration stores columns on separate pages. This allows you to read only the columns that are actually needed. Regarding InfoCubes, that data store design implies that adding key figures to an InfoCube does not affect the performance of existing queries. Furthermore, spatial locality of data and subsequently cache utilization and SIMD exploitation are maximized. Both play an important role in calculating aggregates and filtering selected dimension entries. Furthermore this enables DB2 Cancun Release 10.5.0.4 to support flat InfoCubes, because the large number of columns in the fact table has no negative impact on the I/O performance.

# 8.9 BLU Acceleration for SAP near-line storage solution on DB2 (NLS)

The SAP near-line storage solution on DB2 (NLS) has been enhanced so that tables for the NLS storage objects can be created as column-organized tables. Table 8-10 lists the supported SAP NetWeaver releases and SAP BW Support Packages. See *SAP Note 1834310* before you use this feature with NLS. For DB2 Cancun Release 10.5.0.4 specific enhancements read *SAP Note 2030925*.

*Table 8-10 Supported SAP BW releases and support packages*

| SAP NetWeaver release | Minimum BW Support Packages[a] | Recommended BW Support Package |
|---|---|---|
| 7.01 | 6 | 15 |
| 7.02 | 3 | 15 |
| 7.30 | 1 | 11 |
| 7.31 | 1 | 10 |
| 7.40 | GA | 5 |

a. These BW Support packages additionally require the installation of *SAP Note 1834310*

## 8.9.1 Overview of NLS

NLS is a category of data persistency that is similar to archiving. Using an NLS database enables you to transfer historical read-only data of the InfoProviders from an SAP BW system to a separate NLS database. The data in the NLS database can be accessed transparently from an SAP BW system.

Figure 8-125 shows a basic setup of the NLS solution with DB2 for Linux, UNIX, and Windows as an NLS database. This is an excerpt from *SAP Guide: Enabling SAP NetWeaver Business Warehouse Systems to Use IBM DB2 for Linux, UNIX, and Windows as Near-Line Storage (NLS)*:

http://service.sap.com/instguidesnw

Select **<Your SAP NetWeaver Main Release>** → **Operations** → **Database-Specific Guides**.

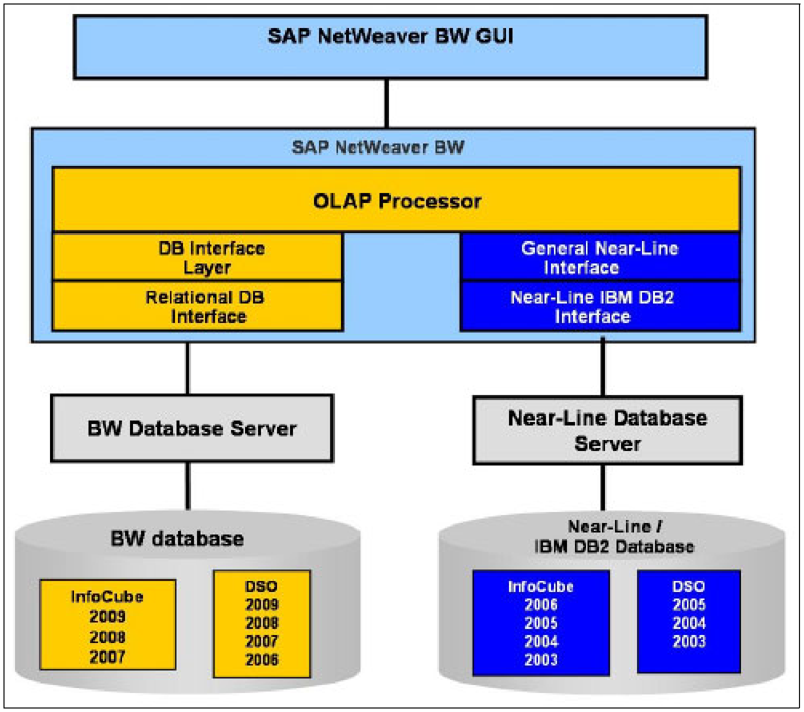



*Figure 8-125 Basic Setup of NLS solution*

Mainly, two data models are implemented on the NLS database on DB2 to store historical read-only data; decisions about the data models are based on whether the InfoProvider is similar to an InfoCube object or a DataStore object.

InfoCube objects that are created in the NLS database contain the following tables:

- Archiving request table to store the archiving request information
- Fact tables that store the key figures of the original InfoCube
- A set of dimension tables to store the characteristics of the original InfoCube

The DataStore object created in the NLS database contains the following items:

- NLS archiving request table to store the archiving request information
- NLS active data table to store the business-related information of the original DataStore object

These tables are created on the NLS database when you create a data archiving process for an InfoProvider on your SAP BW system.

### 8.9.2 BLU Acceleration for NLS storage objects

By default, the data on NLS databases is stored in row-organized tables. If your NLS database runs on DB2 10.5, you can choose to create these tables as column-organized tables. DB2 column-organized tables support the objects of both the NLS InfoCubes and DataStore data models.

To use column-organized tables, you must set the RSADMIN parameter DB6_NLS_USE_CDE to `YES` in your SAP BW system. This parameter is disabled by default. If you create a new data archiving process for an InfoProvider with this parameter enabled, the corresponding new tables in the NLS database are created as column-organized tables. This parameter has no impact on existing row-organized tables in the NLS database.

The following procedure outlines how to set the DB6_NLS_USE_CDE parameter and how to create column-organized tables for NLS InfoCubes and DataStore objects:

1. In your SAP BW system, call transaction **SE38** and run the SAP_RSADMIN_MAINTAIN report.
2. In the *Object* field, enter `DB6_NLS_USE_CDE`. In the *Value* field, enter `YES`. Select the **INSERT** option and click the **Execute** icon (Figure 8-126).

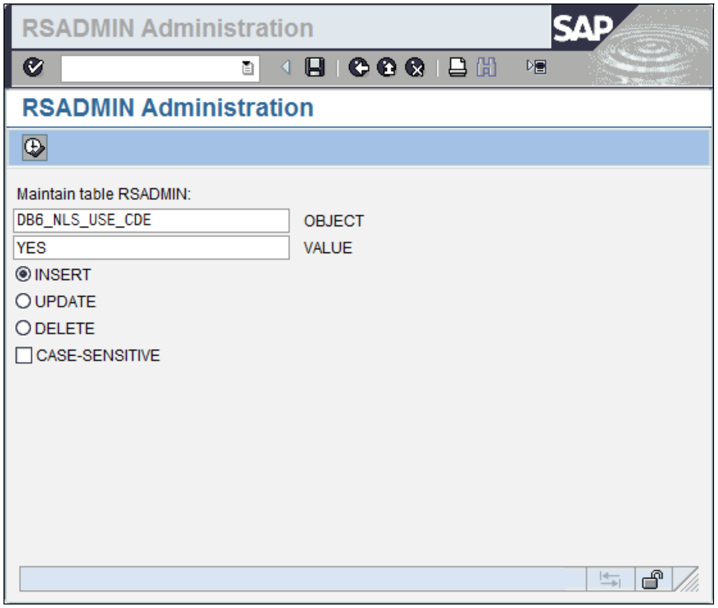



*Figure 8-126 Report SAP_RSADMIN_MAINTAIN: Set DB6_NLS_USE_CDE parameter*

3. Select an InfoProvider and then select **Create Data Archiving Process** (Figure 8-127). In this example, InfoCube `/BIB/Bench02` was selected as InfoProvider.

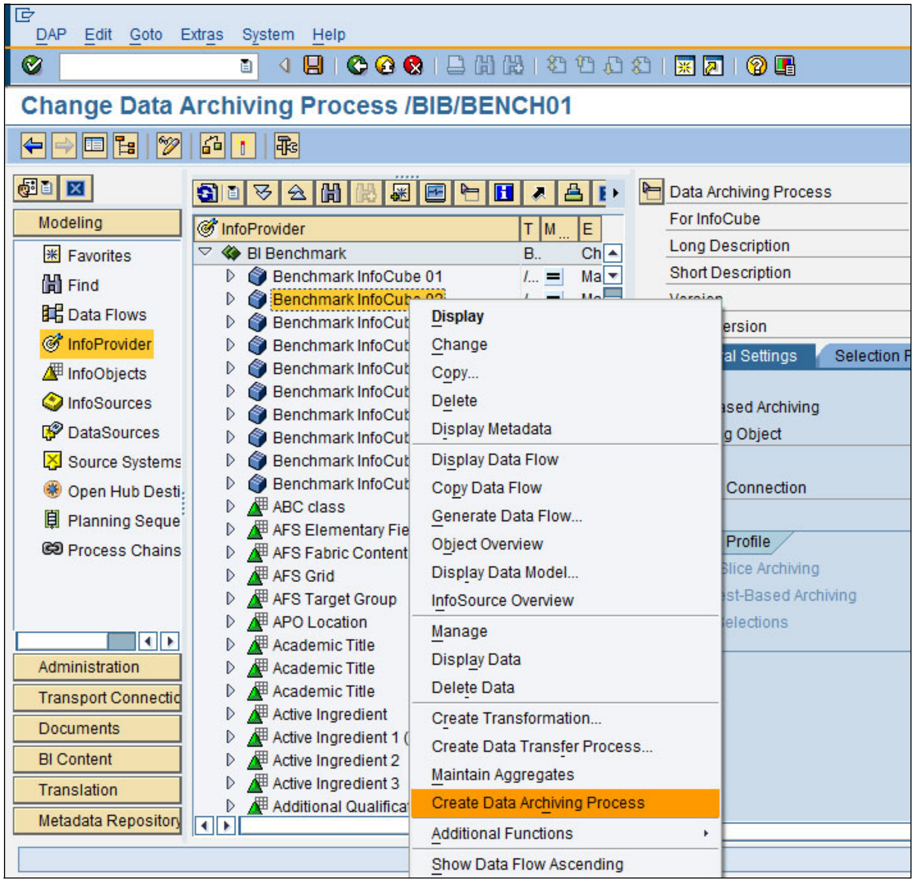


*Figure 8-127 SAP Data Warehousing Workbench: creating data archiving process*

4. On the General Settings tab (Figure 8-128), proceed as follows:
   a. For the Data Archiving Process, enter descriptions in the *Long Description* and *Short Description* fields.
   b. Deselect **ADK-Based Archiving** if you did not install the ADK software.
   c. Enter the name of the connection to your NLS database in the *Near-Line Connection* field.
   d. Save and activate the data archiving process.

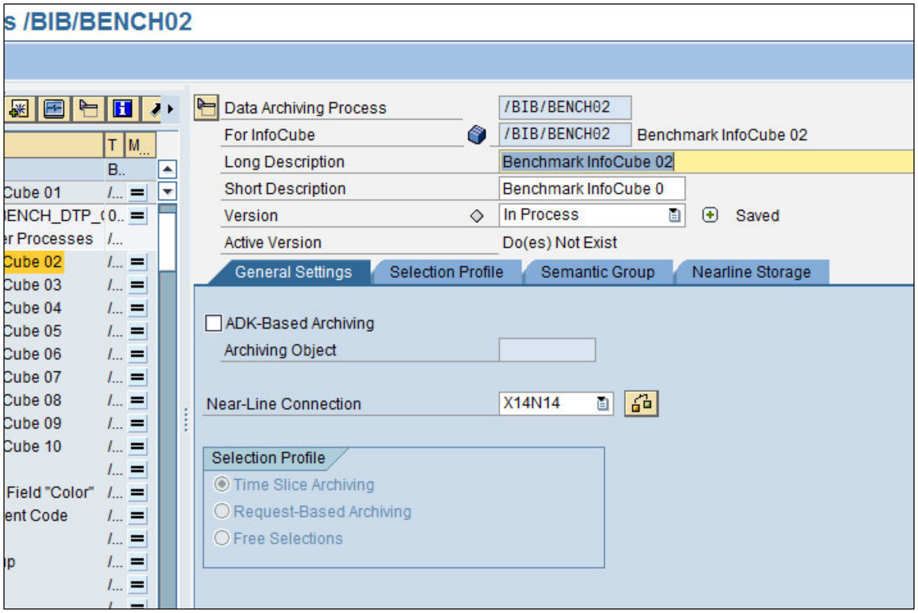



*Figure 8-128 SAP Data Warehousing Workbench: Changing data archiving process*

The *Log Display* window (Figure 8-129) opens. It indicates that all related tables have been created in the NLS database.

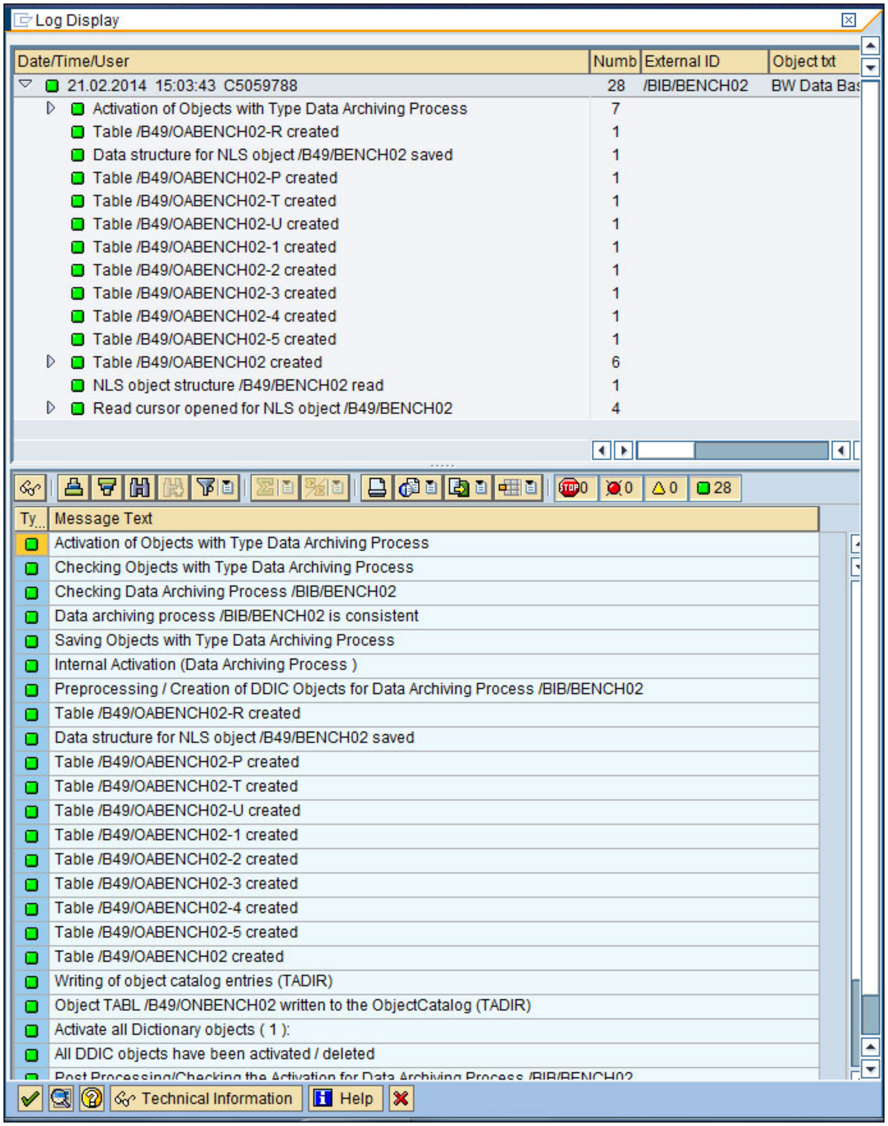



Figure 8-129 SAP Data Warehousing Workbench: Log Display

5. To check whether these newly created tables are column-organized, use the DBACOCKPIT (Figure 8-130).

   a. From your SAP BW system, call transaction **DBACOCKPIT**.

   b. Select your NLS system and then choose **Space** → **Single Table Analysis** in the navigation frame.

   c. In the *Table and Indexes* Details area, enter the table name in the *Name* field and choose **Enter**. You can see if the table is column-organized in the *Table Organization* field on the *System Catalog* tab page.

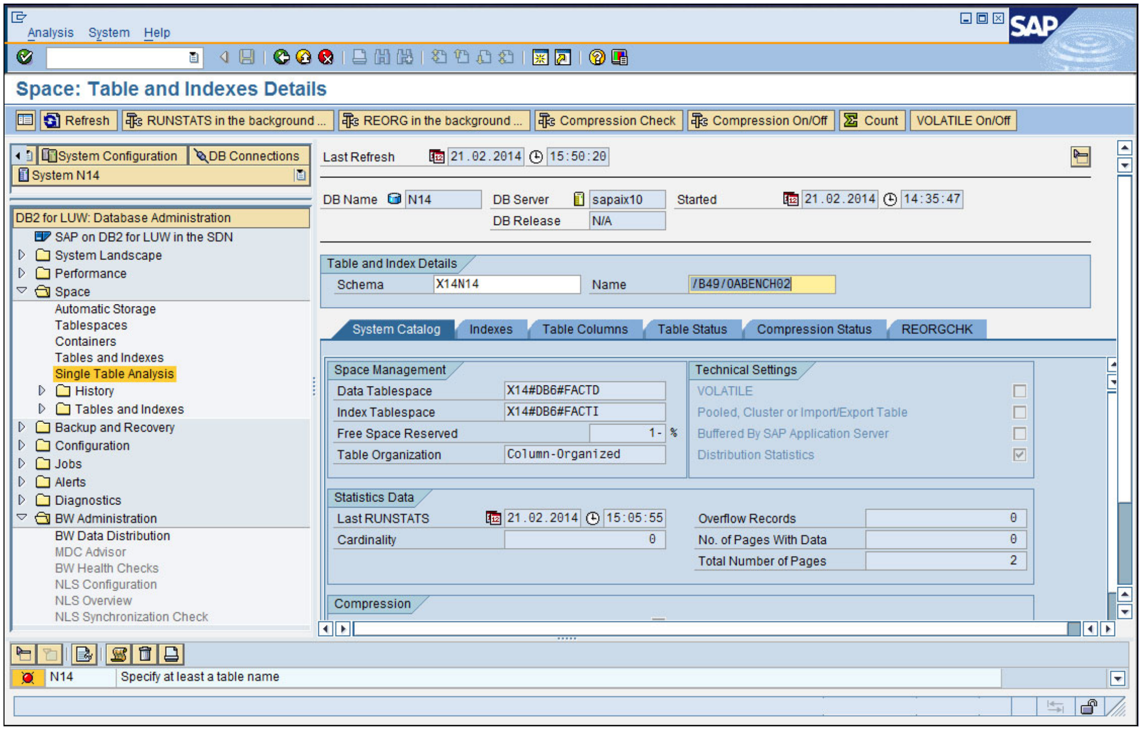



*Figure 8-130 SAP DBA Cockpit: Properties of a single table*

6. Alternatively, you can check the newly created table using **`db2look`**. To run **`db2look`**, as instance owner, log on to the server that is hosting the NLS database and run the command as follows:

```
db2look -d <DBNAME> -a -e -t <TABLENAME>
```

Example 8-15 shows the DDL of `OABENCH02-P` table.

*Example 8-15 The DDL for OABENCH02-P table generated by db2look*

```
------------------------------------------------
-- DDL Statements for Table "X14N14  "."/B49/OABENCH02-P"
------------------------------------------------


CREATE TABLE "X14N14  "."/B49/OABENCH02-P"  (
                  "PK_DBENCH02P" INTEGER NOT NULL ,
                  "CHNGID" VARCHAR(42 OCTETS) NOT NULL ,
                  "RECORDTP" VARCHAR(3 OCTETS) NOT NULL ,
                  "REQUID" VARCHAR(90 OCTETS) NOT NULL )
                 IN "X14#DB6#DIMD" INDEX IN "X14#DB6#DIMI"
                 ORGANIZE BY COLUMN;


-- DDL Statements for Primary Key on Table "X14N14  "."/B49/OABENCH02-P"

ALTER TABLE "X14N14  "."/B49/OABENCH02-P"
        ADD CONSTRAINT "/B49/OABENCH02-P~0" PRIMARY KEY
                ("PK_DBENCH02P");


-- DDL Statements for Unique Constraints on Table "X14N14  "."/B49/OABENCH02-P"

ALTER TABLE "X14N14  "."/B49/OABENCH02-P"
        ADD CONSTRAINT "/B49/OABENCH02-P~1" UNIQUE
                ("CHNGID",
                 "RECORDTP",
                 "REQUID");
```

After the NLS tables have been created, regardless whether they are row-organized or column-organized, the rest of the NLS operations are the same. For more information, see the SAP documentation *Enabling SAP NetWeaver Business Warehouse Systems to Use IBM DB2 for Linux, UNIX, and Windows as Near-Line Storage (NLS)*[7]:

http://service.sap.com/instguidesnw

Select ***<Your SAP NetWeaver Main Release>*** → **Operations** → **Database-Specific Guides**.

[7] See http://service.sap.com/~sapidb/011000358700000815822010E

### 8.9.3 Configuration for BLU Acceleration on NLS

If you are using column-organized tables on the NLS database, you must increase the database configuration parameters SHEAPTHRES_SHR and SORTHEAP to meet the special memory requirements for column-organized tables. Also, INTRA_PARALLEL and MAX_QUERYDEGREE must be set to `ANY`. See *SAP Note 1851832* for more information about SAP standard parameter settings.

### 8.9.4 NLS specific limitations

When you use column-organized tables for NLS InfoCubes and DataStore objects, the following restrictions might apply:

- Starting with DB2 Cancun Release 10.5.0.4, the conversion of existing row-organized NLS tables, using DB6CONV, to column-organized tables is supported. This does not apply on DB2 10.5 FP3aSAP or earlier fix packs.
- As of DB2 Cancun Release 10.5.0.4, adding columns is supported, if the corresponding NLS objects are created as column-organized tables. Other table layout changes are not directly supported in SAP NLS. If you are running on DB2 10.5 FP3aSAP or earlier, or you want to perform any other layout changes than adding columns in the SAP BW system, manual steps are required.

### 8.9.5 Benefits of BLU Acceleration for SAP NLS on DB2

SAP NLS on DB2 profits from BLU Acceleration in the same aspects as SAP BW does:

- Query performance is improved due to the column-oriented data store.
- Less tuning is necessary and no secondary indexes have to be maintained.
- The total storage consumption is reduced due to the advanced compression capabilities of DB2 with BLU Acceleration.

A

# BLU Acceleration monitor elements

Table A-1 lists the monitor elements, related to BLU Acceleration, that are added to DB2 10.5.

*Table A-1 BLU Acceleration monitor elements for DB2 10.5*

| Area | Monitor element | Purpose |
|---|---|---|
| **Disk Data** | | |
| Logical Pages | COL_OBJECT_L_PAGES | Column-organized logical pages monitor element: The number of logical pages used on disk by column-organized data. |
| Logical Size | COL_OBJECT_L_SIZE | Column-organized data object logical size monitor element: Amount of disk space logically allocated for the column-organized data in the table, reported in kilobytes. |
| Physical Size | COL_OBJECT_P_SIZE | Column-organized data object physical size monitor element: Amount of disk space physically allocated for the column-organized data in the table, reported in kilobytes. |

| Area | Monitor element | Purpose |
|---|---|---|
| **BUFFERPOOL** | | |
| Page Reads | POOL_COL_L_READS | Buffer Pool column-organized logical reads monitor element: This indicates the number of column-organized pages read in from the table space containers (physical) for temporary table spaces. |
| | POOL_COL_P_READS | Buffer Pool column-organized physical reads monitor element: This indicates the number of column-organized pages read in from the table space containers (physical) for regular and large table spaces. |
| | POOL_COL_LBP_PAGES_FOUND | Buffer pool column-organized Local Buffer Pool pages found monitor element: This indicates the number of times that a column-organized page was present in the local buffer pool. |
| Page Writes | POOL_COL_WRITES | Buffer pool column-organized writes monitor element: This is the number of times a buffer pool column-organized page was physically written to disk |
| Asynch Reads / Writes | POOL_ASYNC_COL_READS | Buffer pool asynchronous column-organized reads monitor element: Indicates the number of column-organized pages read in from the table space containers (physical) by a prefetcher for all types of table spaces. |
| | POOL_ASYNC_COL_READ_REQS | Buffer pool asynchronous column-organized read requests monitor element: This is the number of asynchronous column-organized read requests by the prefetcher to the operating system. |
| | POOL_ASYNC_COL_WRITES | Buffer pool asynchronous column-organized writes monitor element: This is the number of times a buffer pool data page was physically written to disk by either an asynchronous page cleaner, or a prefetcher. |
| | POOL_ASYNC_COL_LBP_PAGES_FOUND | Asynchronous local buffer pool column-organized pages found monitor element: This is the number of times a data page was present in the local buffer pool when a prefetcher attempted to access it. |

| Area | Monitor element | Purpose |
|---|---|---|
| Prefetch Requests | POOL_QUEUED_ASYNC_COL_REQS | Column-organized prefetch requests monitor element: This is the number of column-organized prefetch requests successfully added to the prefetch queue. |
| | POOL_QUEUED_ASYNC_COL_PAGES | Column-organized page prefetch requests monitor element: This is the number of column-organized pages successfully requested for prefetching. |
| | POOL_FAILED_ASYNC_COL_REQS | Failed column-organized prefetch requests monitor element: This is the number of times an attempt to queue a column-organized prefetch request was made but failed. One possible reason is the prefetch queue is full. |
| | SKIPPED_PREFETCH_COL_P_READS | Skipped prefetch column-organized physical reads monitor element: This is the number of column-organized pages that an I/O server (prefetcher) skipped because the pages were already loaded into the buffer pool. |
| | SKIPPED_PREFETCH_UOW_COL_P_READS | Skipped prefetch unit of work column-organized physical reads monitor element: This is the number of column-organized pages that an I/O server (prefetcher) skipped because the pages were already loaded into the buffer pool by an agent in the same unit of work. |
| Page Reads Table/Statement | OBJECT_COL_L_READS | Column-organized logical reads monitor element: This is the number of column-organized pages that are logically read from the buffer pool for a table. |
| | OBJECT_COL_P_READS | Column-organized physical reads monitor element: This is the number of column-organized pages that are physically read for a table. |
| | OBJECT_COL_LBP_PAGES_FOUND | Local buffer pool, column-organized pages found, monitor element: This is the number of times that a column-organized page for a table is present in the local buffer pool (LBP). |
| **SQL** | | |
| Statement | NUM_COLUMNS_REFERENCED | Number of columns referenced monitor element: This element counts the number of columns referenced during the execution of a section for an SQL statement. |
| Execution Times | TOTAL_COL_TIME | Total column-organized time monitor element: This is the total elapsed time spent accessing columnar data in a query on column-organized tables. The value is given in milliseconds. |
| | TOTAL_COL_PROC_TIME | Total column-organized processing time monitor element: This is the total non-wait processing time spent accessing columnar data in a query on column-organized tables. The value is given in milliseconds |
| | TOTAL_COL_EXECUTIONS | Total column-organized executions monitor element: This is the total number of times that data in column-organized tables was accessed. |

## A.1 Sample Monreport output

Example A-1 shows output of the Monreport module.

*Example: A-1 Monreport output*

```
Result set 1
  --------------

  TEXT
  --------------------------------------------------------------------------------
  --------------------------------------------------------------------------------
  Monitoring report - database summary
  --------------------------------------------------------------------------------
  Database:                                 COLTEST
  Generated:                                03/19/2014 08:25:25
  Interval monitored:                       10

  ================================================================================
  Part 1 - System performance

  Work volume and throughput
  --------------------------------------------------------------------------------
                                    Per second             Total
                                    ---------------------  -----------------------
  TOTAL_APP_COMMITS                 0                      7
  ACT_COMPLETED_TOTAL               2                      27
  APP_RQSTS_COMPLETED_TOTAL         3                      30

  TOTAL_CPU_TIME                    = 3548373
  TOTAL_CPU_TIME per request        = 118279

  Row processing
    ROWS_READ/ROWS_RETURNED         = 123971 (7686237/62)
    ROWS_MODIFIED                   = 10015

  Wait times
  --------------------------------------------------------------------------------

  -- Wait time as a percentage of elapsed time --

                                           %    Wait time/Total time
                                           ---  ----------------------------------
  For requests                             43   15889/36753
  For activities                           42   15694/36530

  -- Time waiting for next client request --
```

```
CLIENT_IDLE_WAIT_TIME              = 3946
CLIENT_IDLE_WAIT_TIME per second   = 394

-- Detailed breakdown of TOTAL_WAIT_TIME --

                              %    Total
                              ---  ---------------------------------------------
TOTAL_WAIT_TIME               100  15889

I/O wait time
  POOL_READ_TIME              227  36142
  POOL_WRITE_TIME             0    0
  DIRECT_READ_TIME            0    144
  DIRECT_WRITE_TIME           0    32
  LOG_DISK_WAIT_TIME          0    72
LOCK_WAIT_TIME                0    0
AGENT_WAIT_TIME               0    0
Network and FCM
  TCPIP_SEND_WAIT_TIME        0    0
  TCPIP_RECV_WAIT_TIME        0    0
  IPC_SEND_WAIT_TIME          0    11
  IPC_RECV_WAIT_TIME          0    0
  FCM_SEND_WAIT_TIME          0    0
  FCM_RECV_WAIT_TIME          74   11902
WLM_QUEUE_TIME_TOTAL          0    0
CF_WAIT_TIME                  0    0
RECLAIM_WAIT_TIME             0    0
SMP_RECLAIM_WAIT_TIME         0    0

Component times
--------------------------------------------------------------------------------
-- Detailed breakdown of processing time --

                                    %                 Total
                                    ----------------  --------------------------
Total processing                    100               20864

Section execution
  TOTAL_SECTION_PROC_TIME           51                10686
    TOTAL_SECTION_SORT_PROC_TIME    0                 0
Compile
  TOTAL_COMPILE_PROC_TIME           0                 34
  TOTAL_IMPLICIT_COMPILE_PROC_TIME  0                 0
Transaction end processing
  TOTAL_COMMIT_PROC_TIME            0                 0
  TOTAL_ROLLBACK_PROC_TIME          0                 0
Utilities
  TOTAL_RUNSTATS_PROC_TIME          0                 0
  TOTAL_REORGS_PROC_TIME            0                 0
```

```
  TOTAL_LOAD_PROC_TIME              0                  0

Buffer pool
--------------------------------------------------------------------------------
Buffer pool hit ratios

Type             Ratio            Reads (Logical/Physical)
---------------  ---------------  ----------------------------------------------
Data             77               328/73
Index            87               1030/130
XDA              0                0/0
COL              67               29268/9532
Temp data        100              310/0
Temp index       0                0/0
Temp XDA         0                0/0
Temp COL         0                0/0
GBP Data         0                (0 - 0)/0
GBP Index        0                (0 - 0)/0
GBP XDA          0                (0 - 0)/0
GBP COL          0                (0 - 0)/0
LBP Data         88               (642 - 77)/(328 + 310)
LBP Index        87               (900 - 0)/(1030 + 0)
LBP XDA          0                (0 - 0)/(0 + 0)
LBP COL          99               (29581 - 328)/(29268 + 0)

I/O
--------------------------------------------------------------------------------
Buffer pool writes
  POOL_DATA_WRITES      = 0
  POOL_XDA_WRITES       = 0
  POOL_INDEX_WRITES     = 0
  POOL_COL_WRITES       = 0
Direct I/O
  DIRECT_READS          = 326
  DIRECT_READ_REQS      = 26
  DIRECT_WRITES         = 8
  DIRECT_WRITE_REQS     = 3
Log I/O
  LOG_DISK_WAITS_TOTAL  = 6

Locking
--------------------------------------------------------------------------------
                        Per activity                    Total
                        ------------------------------  ----------------------
LOCK_WAIT_TIME          0                               0
LOCK_WAITS              0                               0
LOCK_TIMEOUTS           0                               0
DEADLOCKS               0                               0
LOCK_ESCALS             0                               0
```

```
Routines
--------------------------------------------------------------------------------
                              Per activity             Total
                              ------------------------ ------------------------
TOTAL_ROUTINE_INVOCATIONS     0                        4
TOTAL_ROUTINE_TIME            380                      10260

TOTAL_ROUTINE_TIME per invocation   = 2565

Sort
--------------------------------------------------------------------------------
TOTAL_SORTS                         = 9
SORT_OVERFLOWS                      = 0
POST_THRESHOLD_SORTS                = 0
POST_SHRTHRESHOLD_SORTS             = 0

Network
--------------------------------------------------------------------------------
Communications with remote clients
TCPIP_SEND_VOLUME per send          = 0          (0/0)
TCPIP_RECV_VOLUME per receive       = 0          (0/0)

Communications with local clients
IPC_SEND_VOLUME per send            = 159        (3674/23)
IPC_RECV_VOLUME per receive         = 155        (3741/24)

Fast communications manager
FCM_SEND_VOLUME per send            = 40736      (6436340/158)
FCM_RECV_VOLUME per receive         = 39730      (6436340/162)

Other
--------------------------------------------------------------------------------
Compilation
  TOTAL_COMPILATIONS                = 10
  PKG_CACHE_INSERTS                 = 24
  PKG_CACHE_LOOKUPS                 = 33
Catalog cache
  CAT_CACHE_INSERTS                 = 6
  CAT_CACHE_LOOKUPS                 = 62
Transaction processing
  TOTAL_APP_COMMITS                 = 7
  INT_COMMITS                       = 0
  TOTAL_APP_ROLLBACKS               = 0
  INT_ROLLBACKS                     = 0
Log buffer
  NUM_LOG_BUFFER_FULL               = 0
Activities aborted/rejected
  ACT_ABORTED_TOTAL                 = 0
```

```
  ACT_REJECTED_TOTAL                 = 0
Workload management controls
  WLM_QUEUE_ASSIGNMENTS_TOTAL        = 0
  WLM_QUEUE_TIME_TOTAL               = 0

DB2 utility operations
--------------------------------------------------------------------------------
  TOTAL_RUNSTATS                     = 0
  TOTAL_REORGS                       = 0
  TOTAL_LOADS                        = 0

================================================================================
Part 2 - Application performance drill down

Application performance database-wide
--------------------------------------------------------------------------------
TOTAL_CPU_TIME          TOTAL_       TOTAL_APP_     ROWS_READ +
per request             WAIT_TIME %  COMMITS        ROWS_MODIFIED
----------------------  -----------  -------------  ----------------------------
118279                  43           7              7696252

Application performance by connection
--------------------------------------------------------------------------------
APPLICATION_   TOTAL_CPU_TIME        TOTAL_         TOTAL_APP_     ROWS_READ +
HANDLE         per request           WAIT_TIME %    COMMITS        ROWS_MODIFIED
-------------  -------------------   -----------    -------------  -------------
7              131343                59             7              7696252
16             9093                  0              0              0

Application performance by service class
--------------------------------------------------------------------------------
SERVICE_       TOTAL_CPU_TIME        TOTAL_         TOTAL_APP_     ROWS_READ +
CLASS_ID       per request           WAIT_TIME %    COMMITS        ROWS_MODIFIED
--------       -------------------   -----------    -------------  -------------
4              0                     0              0              0
11             0                     0              0              0
12             0                     0              0              0
13             118303                43             7              7696252

Application performance by workload
--------------------------------------------------------------------------------
WORKLOAD_      TOTAL_CPU_TIME          TOTAL_       TOTAL_APP_     ROWS_READ +
NAME           per request             WAIT_TIME %  COMMITS        ROWS_MODIFIED
-------------  ----------------------  -----------  -------------  -------------
SYSDEFAULTADM  0                       0            0              0
SYSDEFAULTUSE  119118                  43           7              7696252

================================================================================
```

```
Part 3 - Member level information

- I/O wait time is
  (POOL_READ_TIME + POOL_WRITE_TIME + DIRECT_READ_TIME + DIRECT_WRITE_TIME).

        TOTAL_CPU_TIME          TOTAL_       RQSTS_COMPLETED_  I/O
MEMBER  per request             WAIT_TIME %  TOTAL             wait time
------  ----------------------  -----------  ----------------  -----------------
0       93420                   43           38                1411


242 record(s) selected.

Return Status = 0
```

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks publications

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.


- *DB2 Workload Manager for Linux, UNIX, and Windows*, SG24-7524
- *IBM Cognos Dynamic Cubes*, SG24-8064
- *InfoSphere Warehouse: A Robust Infrastructure for Business Intelligence*, SG24-7813
- *Leveraging DB2 10 for High Performance of Your Data Warehouse*, SG24-8157


You can search for, view, download or order these documents and other Redbooks publications, Redpaper publications, Web Docs, draft and additional materials, at the following website:

**ibm.com**/redbooks

## Other publications

These publications are also relevant as further information sources:


- *DB2 10.5 with BLU Acceleration: New Dynamic In-Memory Analytics for the Era of Big Data,* by Paul Zikopoulos, Matthew Huras, George Baklarz, Sam Lightstone, and Aamer Sachedina; McGraw Hill, October 2013

## Online resources

These websites are also relevant as further information sources:

- IBM DB2 Version 10.5:

  http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.welcome.doc/doc/welcome.html
- IBM Cognos Business Intelligence 10.2.0 documentation:

  http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp
- IBM Cognos Dynamic Cubes FAQ:

  http://www-01.ibm.com/support/docview.wss?uid=swg27036155
- IBM Big Data:

  http://www-01.ibm.com/software/data/bigdata
- IBM PureSystems:
- http://www.ibm.com/ibm/puresystems/us/en/pf_pureapplication.html

## Help from IBM

IBM Support and downloads:

**ibm.com**/support

IBM Global Services:

**ibm.com**/services

Architecting and Deploying DB2 with BLU Acceleration

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

# Architecting and Deploying DB2 with BLU Acceleration

**DB2 with BLU Acceleration deployment details**

**Integration with SAP Business Warehouse information**

**Examples with Cognos Business Intelligence**

IBM DB2 with BLU Acceleration is a revolutionary technology that is delivered in DB2 for Linux, UNIX, and Windows Release 10.5. BLU Acceleration delivers breakthrough performance improvements for analytic queries by using dynamic in-memory columnar technologies. Different from other vendor solutions, BLU Acceleration allows the unified computing of OLTP and analytics data inside a single database, therefore, removing barriers and accelerating results for users. With observed hundredfold improvement in query response time, BLU Acceleration provides a simple, fast, and easy-to-use solution for the needs of today's organizations; quick access to business answers can be used to gain a competitive edge, lower costs, and more.

This IBM Redbooks publication introduces the concepts of DB2 with BLU Acceleration. It discusses the steps to move from a relational database to using BLU Acceleration, optimizing BLU usage, and deploying BLU into existing analytic solutions today, with an example of IBM Cognos. This book also describes integration of DB2 with BLU Acceleration into SAP Business Warehouse (SAP BW) and SAP's near-line storage solution on DB2. This publication is intended to be helpful to a wide-ranging audience, including those readers who want to understand the technologies and those who have planning, deployment, and support responsibilities.

**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

SG24-8212-01 ISBN 0738440124