

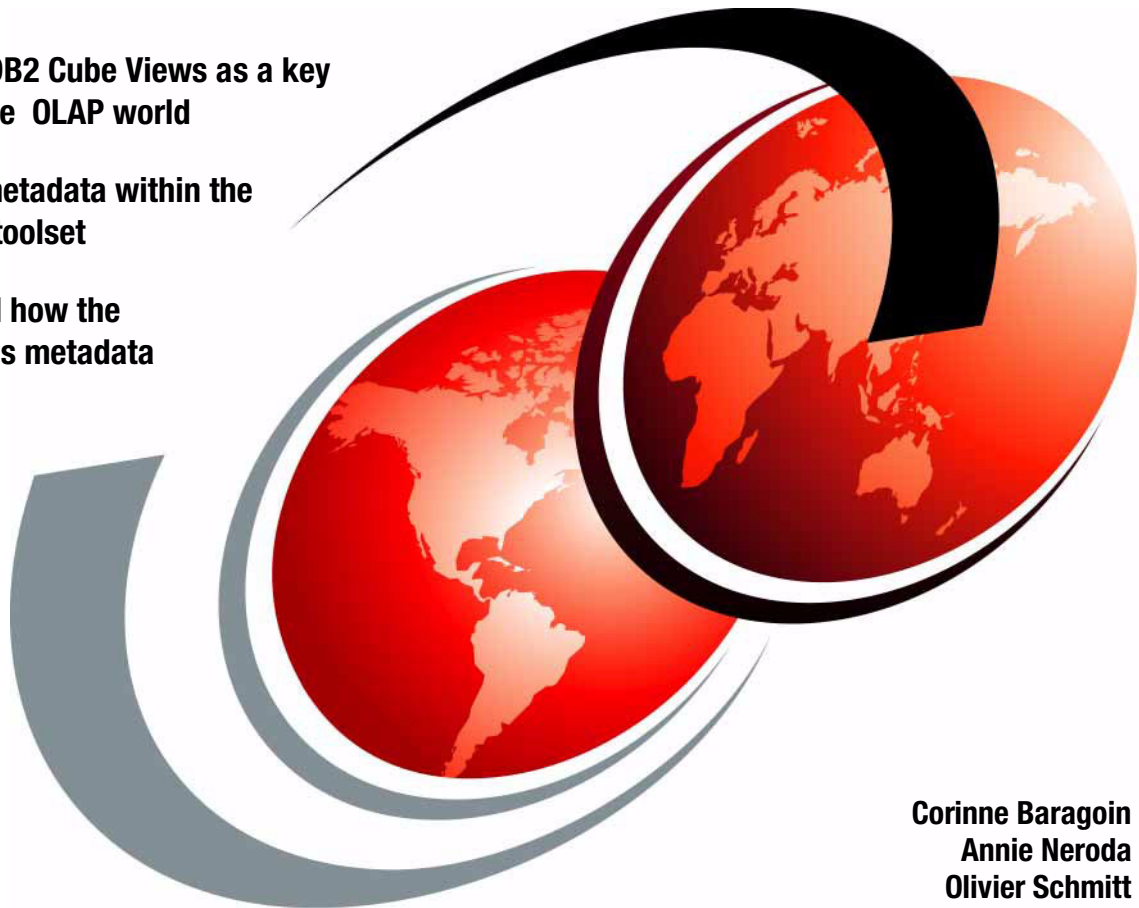
DB2 Cube Views

Getting Started with Meta Integration

Introduce DB2 Cube Views as a key player in the OLAP world

Integrate metadata within the enterprise toolset

Understand how the bridge maps metadata



Corinne Baragoin
Annie Neroda
Olivier Schmitt



International Technical Support Organization

**DB2 Cube Views:
Getting Started with Meta Integration**

September 2003

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

First Edition (September 2003)

This edition applies to IBM DB2 Universal Database Version 8.1 FixPack2, IBM DB2 Cube Views V8.1 and to Meta Integration Model Bridge V3.1.

This document created or updated on September 4, 2003.

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this redbook for more current information.

© Copyright International Business Machines Corporation 2003. All rights reserved.

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

Contents

Notices	v
Trademarks	vi
Preface	vii
The team that wrote this Redpaper	vii
Become a published author	ix
Comments welcome	ix
Chapter 1. DB2 Cube Views: scenarios and benefits	1
1.1 What can DB2 Cube Views do for you?	2
Chapter 2. Meta Integration of DB2 Cube Views within the enterprise toolset	
9	
2.1 Meta Integration Technology products overview	10
2.1.1 Meta Integration Works (MIW)	10
2.1.2 Meta Integration Repository (MIR)	12
2.1.3 Meta Integration Model Bridge (MIMB)	12
2.2 Architecture and components involved	13
2.3 Metadata flow scenarios	14
2.4 Metadata mapping and limitations considerations	17
2.4.1 Forward engineering from a relational model to a cube model	17
2.4.2 Reverse engineering of a cube model into a relational model	18
2.5 Implementation steps scenario by scenario	18
2.5.1 Metadata integration of DB2 Cube Views with ERwin v4.x.	19
2.5.2 Metadata integration of DB2 Cube Views with ERwin v3.x.	32
2.5.3 Metadata integration of DB2 Cube Views with PowerDesigner	45
2.5.4 Metadata integration of DB2 Cube Views with IBM Rational Rose	58
2.5.5 Metadata integration of DB2 Cube Views with CWM and XMI	73
2.5.6 Metadata integration of DB2 Cube Views with DB2 Warehouse Manager	
79	
2.5.7 Metadata integration of DB2 Cube Views with Informatica	88
2.6 Refresh considerations	92
2.7 Conclusion: benefits	94
Related publications	97
IBM Redbooks	97
Other publications	97
Online resources	97
How to get IBM Redbooks	98

Help from IBM	98
Index	99

Notices

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

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

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

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates 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

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

DB2 Universal Database™

DB2®

@server™


@server™

ibm.com®

IBM®

Rational®

Rational Rose®

Redbooks (logo) ™

Redbooks™

SP1®

XDE™

The following terms are trademarks of other companies:

Meta Integration is a registered trademark of Meta Integration Technology, Inc.

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, 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 Sun Microsystems, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

Multidimensionality is the primary requirement for an OLAP system, and the cube always refers to the collections of the data that an OLAP system implements.

Business Intelligence and OLAP systems are no longer limited to the privileged few business analysts: they are being democratized by being shared with the rank and file employee demanding a Relational Database Management System (RDBMS) that is more OLAP-aware.

IBM DB2® Cube Views V8.1 (DB2 Cube Views through the Redpaper) and its cube model provides DB2 Universal Database™ (DB2 through the Redpaper) the ability to address multidimensional analysis and become an actor in the OLAP world, as detailed in the IBM Redbook, *DB2 Cube Views: A Primer*, SG24-7002.

This Redpaper documents the Meta Integration metadata bridges for DB2 Cube Views and will help DB2 database administrators and Business Intelligence architects understand and evaluate its benefits in their own Business Intelligence and OLAP system environment.

The team that wrote this Redpaper

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

Corinne Baragoïn is a Business Intelligence Project Leader at the International Technical Support Organization, San Jose Center. She has over 17 years of experience as an IT specialist on DB2 UDB and related solutions. Before joining the ITSO in 2000, she worked as an IT Specialist for IBM France, supporting Business Intelligence technical presales activities and assisting customers on DB2 UDB, data warehouse and OLAP solutions.

Annie Neroda is a Senior Consulting Software IT Specialist in the USA. She has 35 years of experience in IT field. She holds a degree in Mathematics from Trinity College in Washington, DC. Her areas of expertise include Business Intelligence, especially OLAP, ETL and Data Mining. She has taught extensively on DB2, Business Intelligence and Data Warehousing.

Olivier Schmitt is a software engineer and metadata specialist and has been working with Meta Integration Technology for 3 years. He holds an BS and MS in computer science from the ENSIMAG engineering school (Grenoble, France). His areas of expertise include metadata mapping and management, C++ and Java™ application development, and he has experience with numerous databases, design tools, business intelligence tools, ETL tools and metadata repositories.

Thanks to the IBM residents team for their contributions and help on this project:

Geetha Balasubramaniam

Bhuvaneshwari Chandrasekharan

Landon DelSordo

Jan B Lillelund

Julie Maw

Paulo Pereira

Jo A Ramos

Thanks to the following people for their involvement and contributions all along this project:

Christian Breteau
Simon Dynin
Bruno Lecointre
Elisabeth Leprince
Meta Integration Technology, Inc

Nathan Colossi
John Poelman
Gary Robinson
Graig Tomlyn
IBM Silicon Valley Lab

Thanks to the following people for their reviews on this redbook:

Bruce Baumbush ?
Mike Biere?
Daniel Graham ?
Paul Wilms
IBM WW Marketing, Sales and Technical Sales

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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 papers to be as helpful as possible. Send us your comments about this Redpaper or other Redbooks™ in one of the following ways:

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

ibm.com/redbooks

- ▶ Send your comments in an Internet note to:

redbook@us.ibm.com

- ▶ Mail your comments to:

IBM® Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099



DB2 Cube Views: scenarios and benefits

In this chapter, we will introduce DB2 Cube Views, the new DB2 feature that makes DB2 OLAP aware.

1.1 What can DB2 Cube Views do for you?

Let's say your organization has decided to deliver first rate analytical capabilities to its end users, and after reading all the latest books and articles on Business Intelligence systems, they have decided to build a star schema database like the one in Figure 1-1 as the heart of this new system. They have probably done this because star schemas offer such rich, business oriented analytical options, such as slicing and dicing, trending, comparisons, rollups and drill-downs.

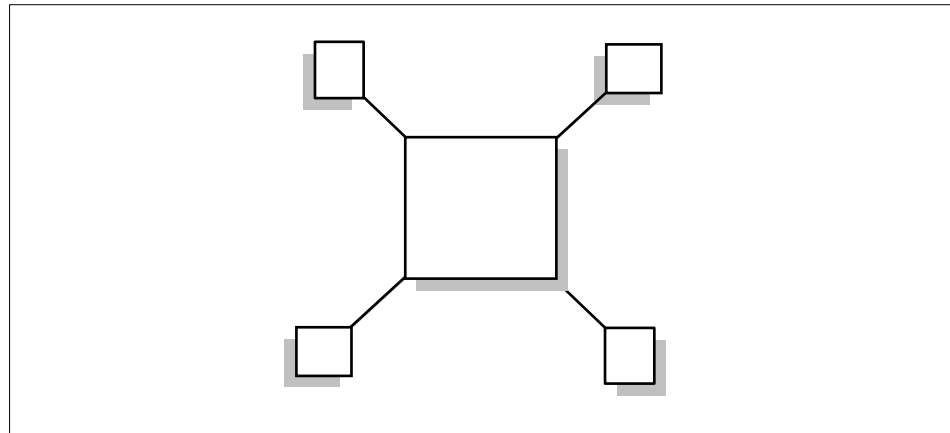


Figure 1-1 Your star schema database

In addition, they will most likely be using one of today's premier data delivery platforms as a front-end for the database because it provides ease of use and because it works so well when coupled with a star schema database. To integrate your front-end tool, the star schema that you have built as tables, columns, primary keys, foreign keys will need to be mapped to the tool as a collection of OLAP objects like measures, derivations, dimensions, hierarchies, attributes and joins. DB2 Cube Views gives you a new GUI called the OLAP Center where you can map these OLAP objects directly to your relational objects and hold these mappings in DB2, as shown in Figure 1-2.

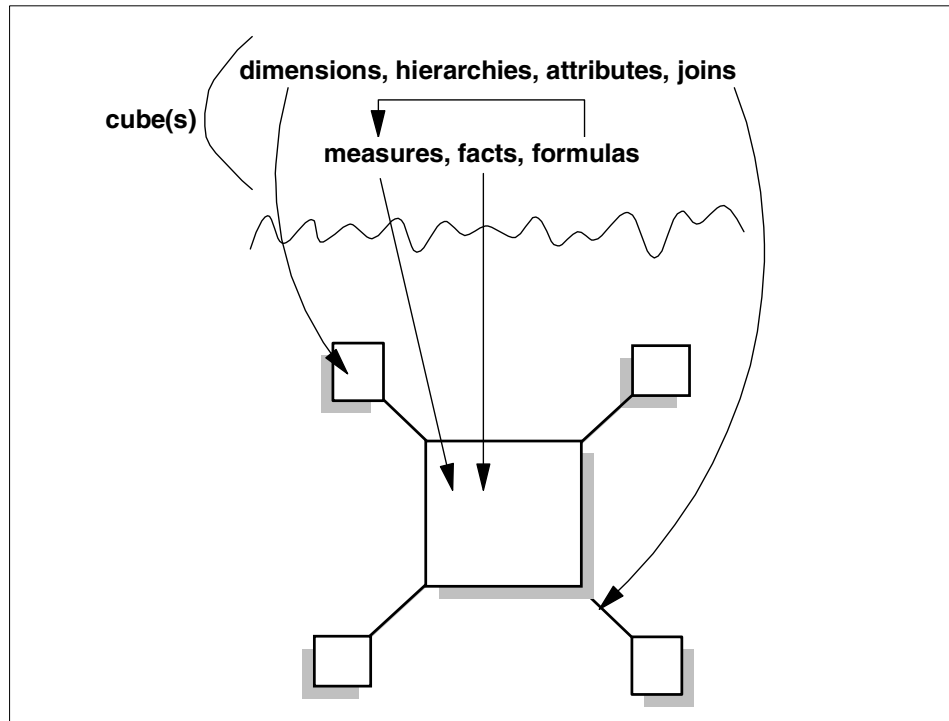


Figure 1-2 Mapping your star schema to OLAP objects

Using the OLAP Center, you can pinpoint the columns in the fact table that actually contain the measures and capture formulas for deriving additional measures that are not physically stored in the star. Further, you can describe the dimensions and their various hierarchies, even multiple hierarchies if that applies. You can also indicate the proper joins to use when accessing the star. Once you have these OLAP objects described, you can group them into cubes, even into multiple cubes, each of which represents a subset of your full cube model based on the star schema. If you have already captured this information in a back-end data modeling or ETL (Extract, Transform, Load) tool, you can skip the data entry and just import the metadata directly via a metadata bridge.

Once the OLAP metadata is stored in DB2 Cube Views, you can use another metadata bridge to send it over to your favorite front-end data delivery tool, automatically, to populate its metadata layer. This way, if a different person is responsible for the database from the one who is responsible for the data delivery tool, then the metadata layer will be consistent. Also, if you will be using multiple tools, the metadata only needs to be captured once, in DB2 Cube Views, and then shared with all the other tools in your solution. Figure 1-3 below illustrates this metadata transfer.

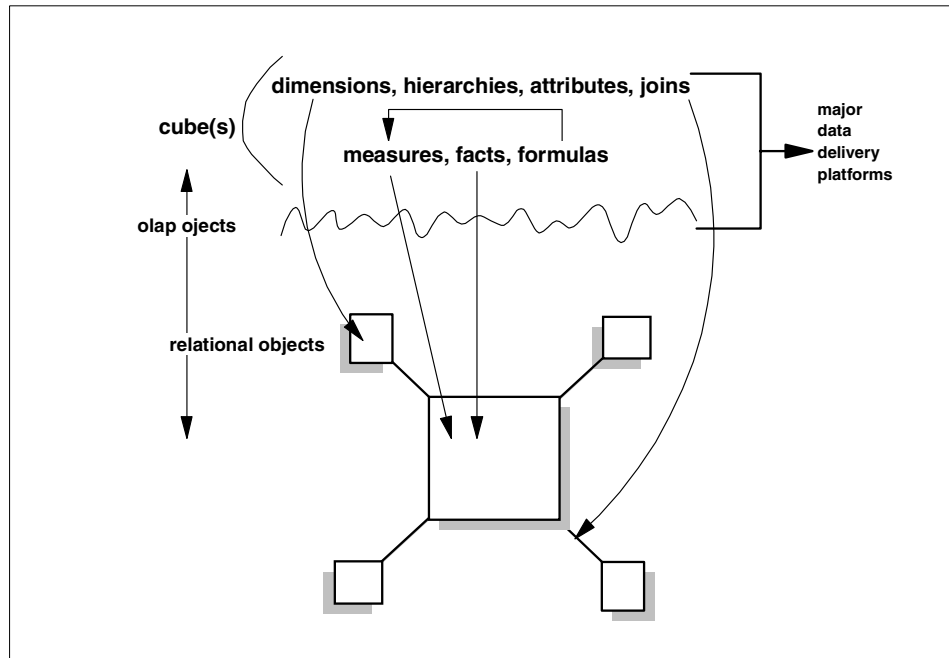


Figure 1-3 Sharing OLAP metadata with reporting tools

Once the metadata layer in your reporting tool has been populated, the tool will soon be sending SQL queries to your star schema. If the SQL requires aggregation and joins, and it probably does, the user's response time could possibly be slow. That is a problem.

But let us say you have a good DBA who knows what to do. He pre-builds an aggregate table and adds it to the database where your star schema is located. The really nice thing about pre-built aggregates in DB2 is that the tool writing the SQL doesn't have to know about them. The DB2 optimizer will automatically use them if the query matches up to them well enough. This makes for very much faster query response times. Figure 1-4 shows a query being satisfied by a pre-built aggregate.

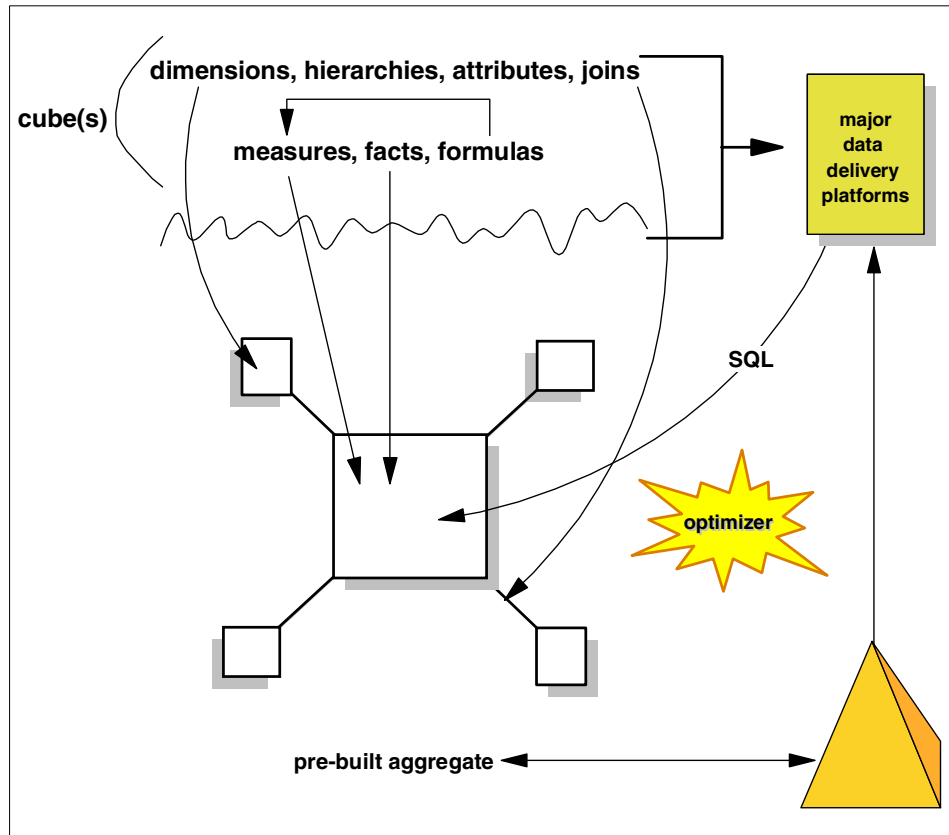


Figure 1-4 Using aggregates

The not-so-nice thing about pre-built aggregates is that the optimizer might not choose to use them every time if the SQL doesn't quite match up. In that case, your DBA may have wasted his time building the wrong aggregates. Perhaps he could solve this problem by building more aggregates, maybe even one for every possible situation. The trouble with that approach is he might end up using as much disk space on aggregates as he did on the star schema itself, not to mention the time he'll have to spend designing the aggregates and refreshing them with data periodically. DB2 Cube Views can help. It can build the ideal set of aggregates or MQTs for him the first time and find out the best compromise between space, time and query performance.

In Figure 1-5, you can see the DB2 Cube Views Optimization Advisor, a very smart expert system on performance that is going to ask your DBA a few questions before it gets to work on building the aggregates. Questions like these:

1. What kinds of queries do you plan to use against this star schema?

- Extracts? For instance, are you going to load multidimensional (*MOLAP*) databases from this star and need a pre-built aggregate that corresponds to the base level or “bottom” of the DB2 Cube Views logical cube?
 - Drill-downs? For instance, are your users going to start-at-the-top (spreadsheet-style), and then drill down from there, typically as ROLAP tools do when they emulate cube-drilling, originating at the top levels of the dimensional hierarchies? If yes, you are going to need aggregations that are at or near the “top” of the logical cube.
 - Drill-through? (also known as Hybrid OLAP or *HOLAP*). For instance, are your users going to drill-down beyond the base level of the MOLAP database, back to the relational database?
 - Reporting? For instance, will your users be making any of the ad-hoc combinations of dimensions and levels, hitting various levels of aggregation through the “center” of the logical cube?
2. How much space are you willing to spend on aggregates?
 - Clearly, if you give the Optimization Advisor lots of space, it will build bigger, more inclusive aggregates.
 - If you give it less space to work with, it will prioritize and build very useful aggregates that will fit.
 3. Next, it will look at your DB2 Cube Model metadata to understand your aggregations and dimensions and hierarchies to improve its decisions.
 4. Next, it is going to look at the DB2 catalog statistics on your star schema tables, just as your DBA would do.
 5. Next, using a data sampling technique, the Optimization Advisor will examine the data in your star schema. This affects the aggregate decisions because while it is sampling, it will actually do the star joins so it can understand the *sparsity* of your data — this gives a very accurate estimate of aggregate size.

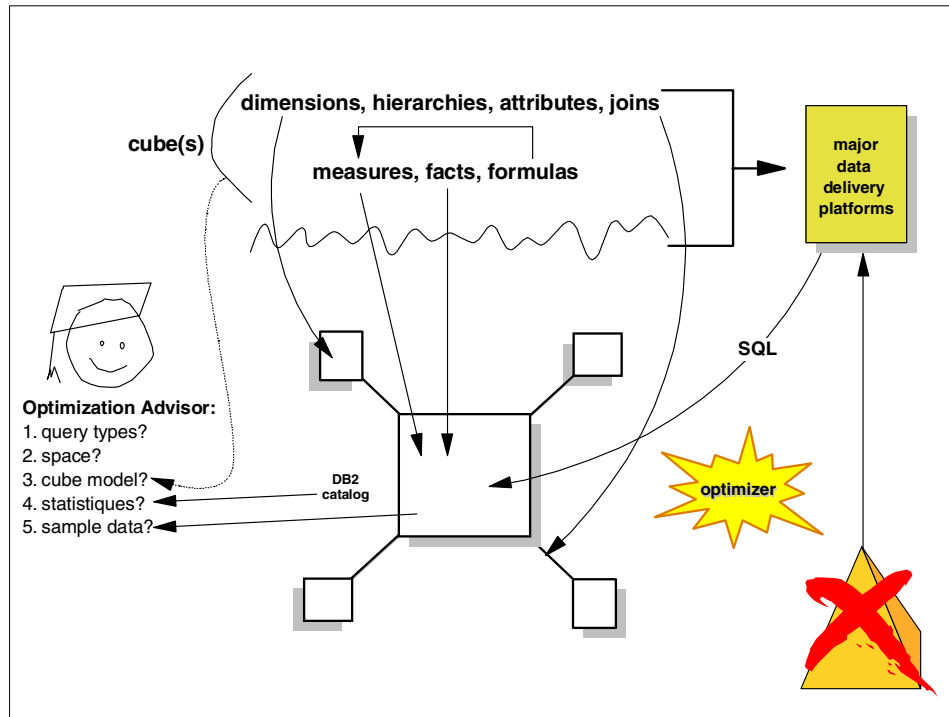


Figure 1-5 The Optimization Advisor gathering its aggregate intelligence

Now, the Optimization Advisor has what it needs to recommend one or more aggregates for your database. In Figure 1-6 you can see that it has generated an aggregate table, in some ways similar to the aggregates your DBA might have built by himself, but it is probably much more than that. By using very sophisticated rules and techniques, the aggregates recommended by the Optimization Advisor will very likely be *super aggregates* with multiple aggregations across multiple combinations of hierarchical levels of multiple dimensions defined within the cube model. In a way, some aggregate tables become a little bit like cubes, but not complete ones because of the space restrictions placed on it by your DBA and by the Optimization Advisor itself. Best of all, the aggregates will be recommended in such a way that they are highly likely to be chosen by the DB2 optimizer at query time.

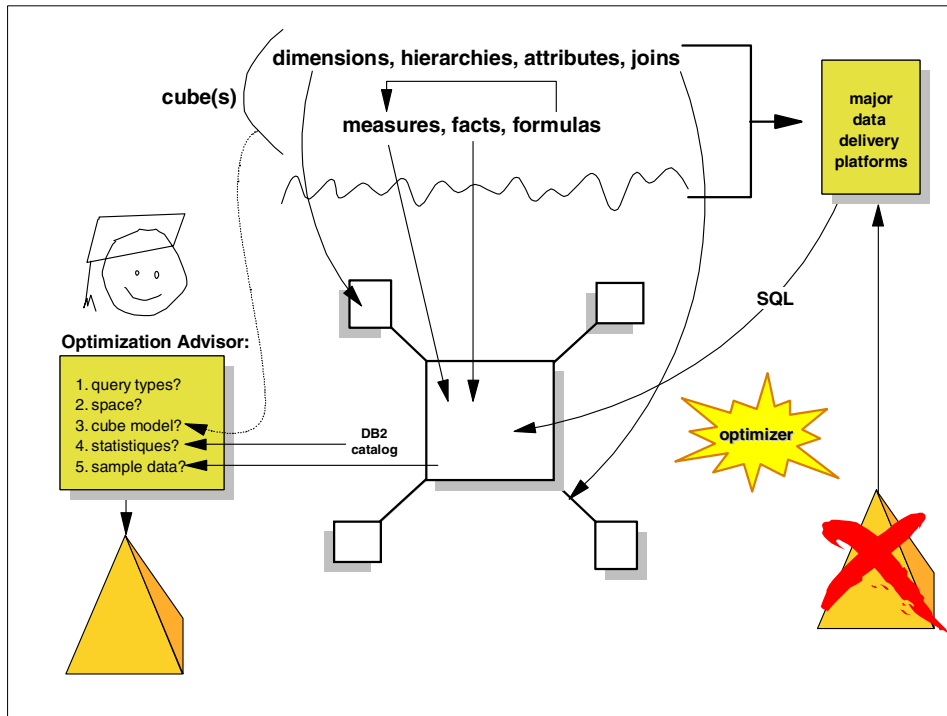


Figure 1-6 The big picture

That's the big picture!



Meta Integration of DB2 Cube Views within the enterprise toolset

This chapter describes certain deployment scenarios for using Meta Integration Technology products. It explains in each scenario how to implement and to use the metadata bridges.

2.1 Meta Integration Technology products overview

Meta Integration Technology, Inc. is a Silicon Valley, California based software vendor specialized in tools for the integration and management of metadata across tools from multiple vendors, and multiple purposes including data and object modeling tools, data Extraction, Transformation, and Load (ETL) tools, Business Intelligence (BI) tools, and so on. The need for data movement and data integration solutions is driven by the fact that data is everywhere underneath business applications. The same applies for metadata: metadata is also everywhere underneath the data and object modeling tools, as well as within the repositories of the ETL, Data Warehouse, Enterprise Application Integration, and Business Intelligence development tools. Meta Integration offers metadata movement solutions for the integration of popular development tools with IBM DB2 Cube Views, as illustrated in Figure 2-1.

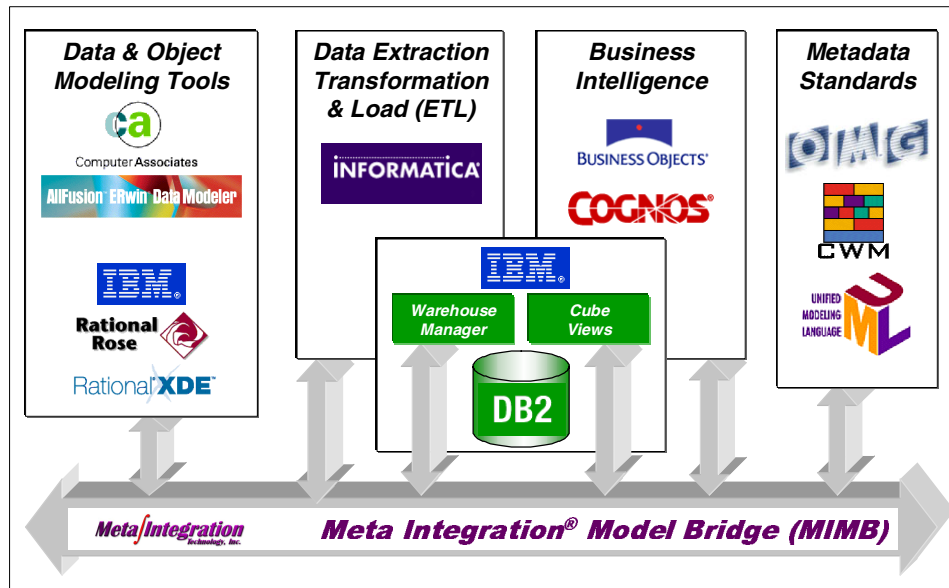


Figure 2-1 A sample of typical metadata movement solutions

2.1.1 Meta Integration Works (MIW)

MIW is a complete metadata management solution with sophisticated functionalities such as the Model Browser, the Model Bridges, the Model Comparator, the Model Integrator, and the Model Mapper all integrated around a powerful metadata version and configuration management as shown in Figure 2-2.

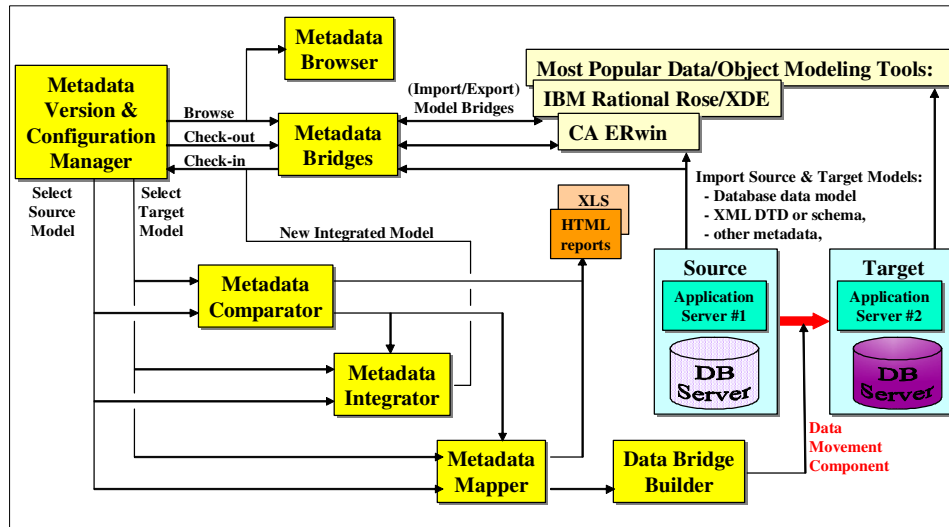


Figure 2-2 Meta Integration functionality

MIW is a powerful metadata management solution, and integrates well with today's best practices in software development, as it provides a unique component based approach to the ETL tool market. Indeed, the MIW development environment generates C++ based data movement components that can be easily integrated (plug and play) with any Windows® or UNIX based business applications. Multiple data movement components can be produced for various purposes such as:

- ▶ Legacy Data Migration (LDM)
- ▶ Enterprise Application Integration (EAI)
- ▶ Data Warehousing (DW) and datamarts.

The code of the produced data movement components can be reviewed through any Quality Assurance (QA) processes, and does not depend on any middleware (free of any run-time cost at deployment time). The Model Mapper provides the mapping migrations required to support the perpetual changes in the source and destination data stores. Indeed, one of the key features of MIW is the built-in support for change management facilitating the maintenance and/or generation of new versions of the data movement components as needed. Data Connectors are available for most popular databases via ODBC (as DB2), as well as for XML data sources (as HL7 for Health Care) to service the expanding needs in the fields of EDI, e-business, and enterprise information portals.

MIW is entirely written in Java, and can be connected to a local or centralized metadata repository.

2.1.2 Meta Integration Repository (MIR)

MIR is based on a modern 3-tier architecture as shown in Figure 2-3 with support for multi-users, security, and concurrency control. The repository metamodel integrates standards like the OMG CWM and UML, and supports XMI compliant metadata interchange. MIR can manage massive amounts of metadata and make it persistent on most popular RDBMS like DB2, Oracle or SQL Server. The underlying repository database is fully open allowing users to build their own metadata Web portals, or use their existing data tools to perform metadata reporting, mining, and even intelligence.

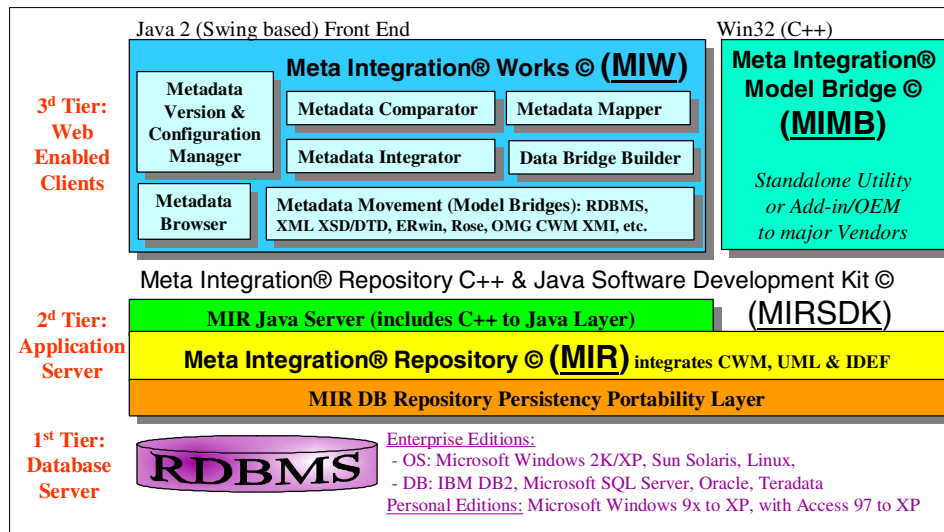


Figure 2-3 Meta Integration architecture

Open database access to the repository for:

- ▶ Web enabled end user Enterprise Metadata Portal
- ▶ Metadata Intelligence and Reporting

2.1.3 Meta Integration Model Bridge (MIMB)

MIMB is a utility for legacy model migration and metadata integration. MIMB also operates as an add-in integrated inside popular modeling, ETL, and BI tools. With over 40 bridges, MIMB is the most complete metadata movement solution on the market. MIMB supports most popular standards and the market leading tool vendors, as illustrated in Figure 2-4.

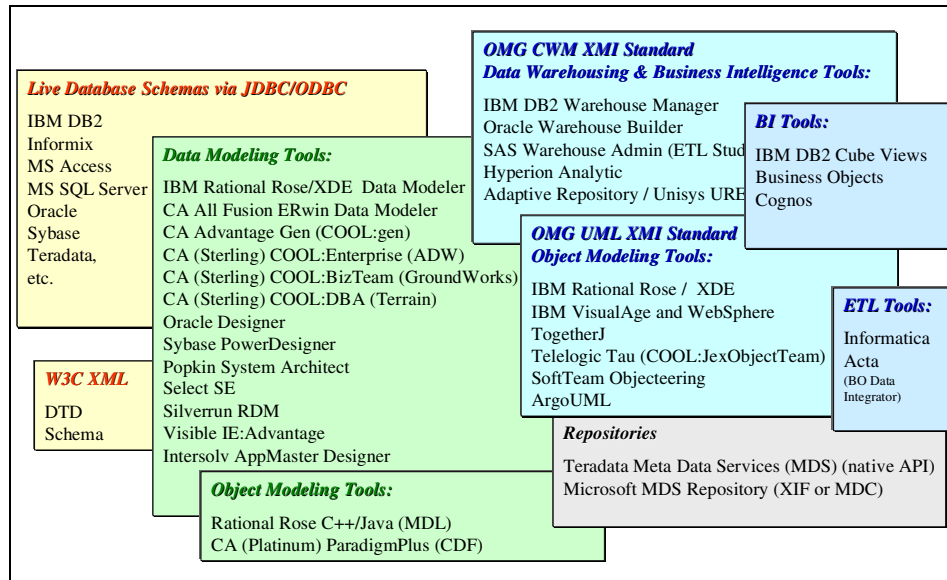


Figure 2-4 Meta Integration supported tools

More details on supported tools and versions are available at:

<http://www.metaintegration.net/Products/MIMB/SupportedTools.html>

<http://www.metaintegration.net/Products/MIMB/AboutVendors.html>

<http://www.metaintegration.net/Products/MIMB/AboutStandards.html>

2.2 Architecture and components involved

Meta Integration Model Bridge (MIMB) as a standalone utility (or add-in metadata movement component to popular ETL/BI tools) is based on the non-persistent version of the Meta Integration Repository (MIR) in memory. Each MIMB Import bridge creates metadata that can be reused by any MIMB export bridge. In other words, Meta Integration does not create point-to-point bridges, as illustrated in Figure 2-5.

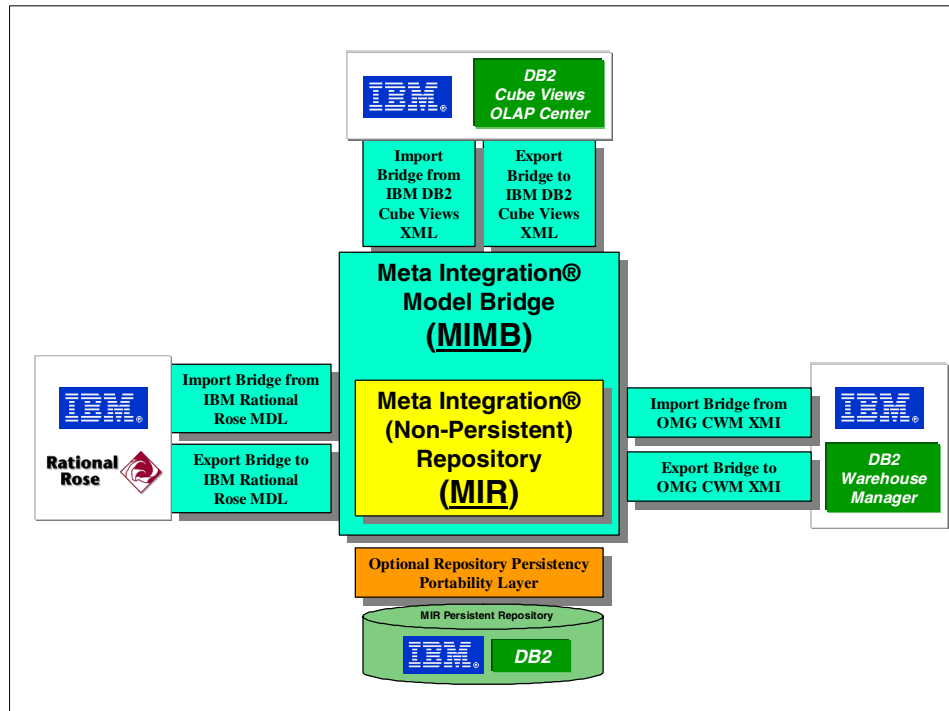


Figure 2-5 A metadata integration solution example

2.3 Metadata flow scenarios

MIMB provides bi-directional metadata movement solutions for the integration of IBM DB2 Cube Views with the development tools of the enterprise.

The exchange of metadata between various tools and DB2 Cube Views using metadata bridges is motivated by several business cases (tools integration in the enterprise, documentation...) and helps data warehouse specialists, database administrators, data modelers and application developers in the following ways:

- ▶ Forward engineering of a data model created in a design tool or an ETL tool to a DB2 Cube Views cube model. This metadata movement capability allows a data modeler to reuse metadata already designed and available in the enterprise to quickly create a cube model in DB2 Cube Views, therefore saving time when creating the OLAP metadata and leveraging the existing metadata, such as business names and descriptions that are not likely to be stored in the database.

- ▶ Reverse engineering of a DB2 Cube Views cube model into a design model. This metadata movement capability allows extracting and reusing the metadata of a cube model created in DB2 Cube Views to quickly create a model in a data modeling tool, an object modeling tool, or an ETL tool in order to document the model, develop a software application or other purposes.

The generic metadata flows can be summarized as in Figure 2-6.

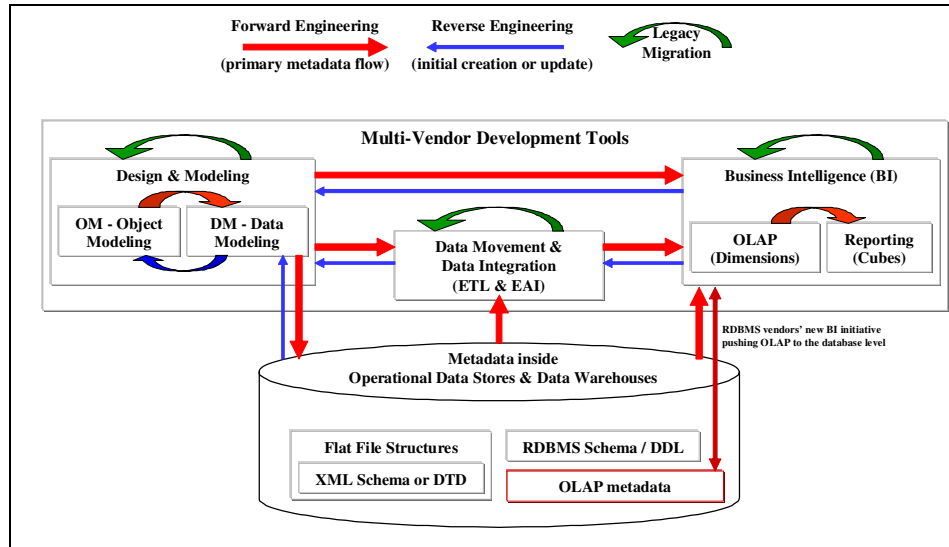


Figure 2-6 Business cases for metadata movement solutions

The tools vendors themselves can provide some of these metadata movements, for example, IBM Rational® Rose® provides bi-directional integration between UML object modeling and physical data modeling. Similarly, BI vendors provide the forward engineering from their OLAP dimension design tool to their OLAP based reporting tool. However, large corporations use best-of-breed tools from many vendors. In such case, MIMB can play a key role implementing all the metadata movement required for the integration of their development tools, as illustrated in Figure 2-7.

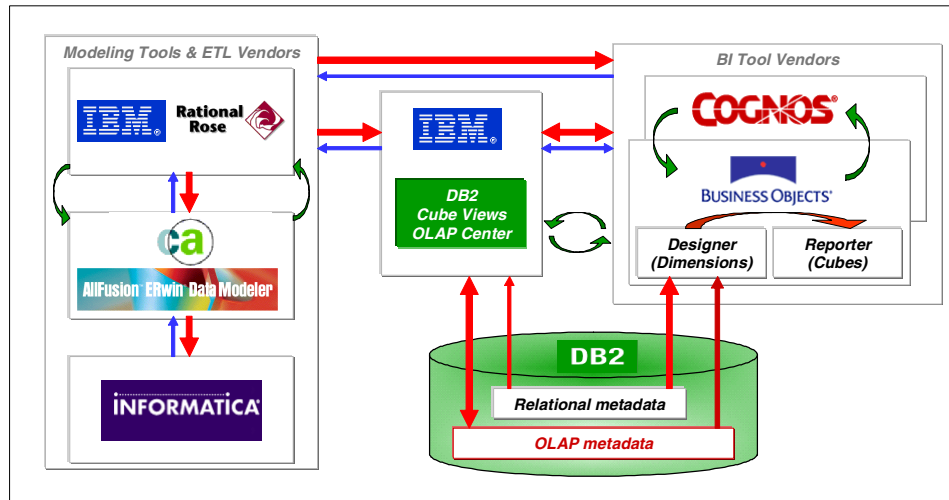


Figure 2-7 Possible metadata movement solutions for DB2 Cube Views

We will demonstrate the implementation of the 7 metadata movement scenarios shown in Figure 2-8, based on popular modeling tools, ETL, and the OMG CWM metadata standard.

As each tool has its own tricks and each MIMB bridge has its own set of import/export options, each scenario has been written as an independent piece and can be read separately based on your interests.

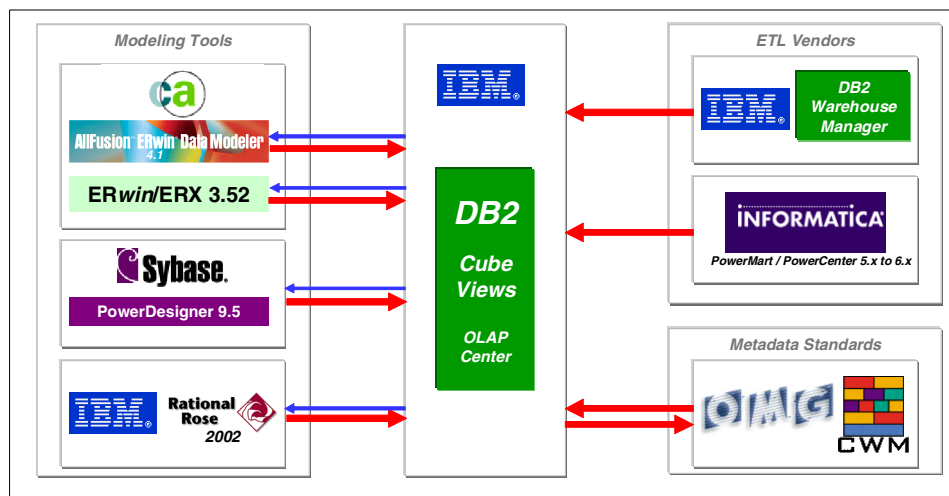


Figure 2-8 Metadata movement scenarios illustrated in this chapter

2.4 Metadata mapping and limitations considerations

These are the four primary scenarios:

1. Forward engineering from popular data modeling tools like IBM Rational Rose or IBM Rational XDE™, and Computer Associates ERwin Data Modeler
2. DB2 Cube Views Integration with ETL tools like Informatica and DB2 Warehouse Manager
3. DB2 Cube Views Integration with BI vendors like BO and Cognos
4. DB2 Cube Views support for metadata standards like OMG CWM XMI.

The current MIMB v3.1 provides IBM DB2 Cube Views import and export bridges for IBM DB2 OLAP Center, and is available for download at:

<http://www.metaintegration.net/Products/Downloads/>

This version 3.1 provides very complete support for the foregoing user cases (1) and (2) of forward engineering:

- ▶ An ERwin star schema sample model is provided with instructions to generate the DB2 Cube Views dimensions, facts, and cube model.
- ▶ However, MIMB v3.1 provides currently incomplete support for the foregoing user cases (3) and (4), due to current BI/OLAP limitations in the Meta Integration Repository (MIR) metamodel of v3.x.

Note: To get the most up-to-date information on new versions and releases, concerning metamodel extensions and support for change management and impact analysis between all the integrated data modeling, ETL, and BI tools, check the following site regularly:

<http://www.metaintegration.net/Products/MIMB>

2.4.1 Forward engineering from a relational model to a cube model

In a forward engineering scenario:

- ▶ The relational tables are used to specify where the tables are located in DB2
- ▶ The fact tables are used to create the cube model facts object
- ▶ The measure columns of the fact tables are transformed into measure objects
- ▶ The dimension and outrigger tables are transformed into dimension objects
- ▶ The dimension and outrigger columns are transformed into dimension attributes
- ▶ The foreign key relationships are used to build joins

- ▶ The business name, description and data type of relational objects are also converted

The produced cube model can then be edited DB2 OLAP Center to enrich it with additional OLAP metadata such as hierarchies, levels, cubes, calculated measures and more.

2.4.2 Reverse engineering of a cube model into a relational model

In a reverse engineering scenario:

- ▶ The relational tables referenced by the cube model are converted to the destination tool.
- ▶ The joins are analyzed in details to create relationships when possible.
- ▶ The OLAP dimensions, facts, attributes and measures business name, description are also converted to the destination tool.

The generated model can be edited in the destination tool to further document it, and add information that was not contained in the source cube model XML file. This missing information can be physical information (such as indexes or tablespaces) that can be retrieved automatically from the database using the destination tool's database synchronization features, or it can be logical information, such as generalizations (super type sub type entities) or UML methods.

For more mapping information, please read the MIMB software documentation, which includes the complete mapping specification of each bridge. This documentation can be consulted online at:

<http://www.metaintegration.net/Products/MIMB/>

2.5 Implementation steps scenario by scenario

This section describes the implementation steps for both forward and reverse engineering for the following metadata exchanges:

- ▶ Metadata integration of DB2 Cube Views with Computer Associates AllFusion ERwin Data Modeler versions 4.0 to 4.1
- ▶ Metadata integration of DB2 Cube Views with Computer Associates ERWin versions 3.0 to 3.5.2
- ▶ Metadata integration of DB2 Cube Views with Sybase PowerDesigner versions 7.5 to 9.5

- ▶ Metadata integration of DB2 Cube Views with IBM Rational Rose versions 2000e to 2002
- ▶ Metadata integration of DB2 Cube Views with the OMG CWM and UML XML standards
- ▶ Metadata integration of DB2 Cube Views with DB2 Warehouse Manager via the OMG CWM XML standard
- ▶ Metadata integration of DB2 Cube Views with Informatica PowerMart

The DB2 Cube Views cube model we used is shown in Figure 2-9.

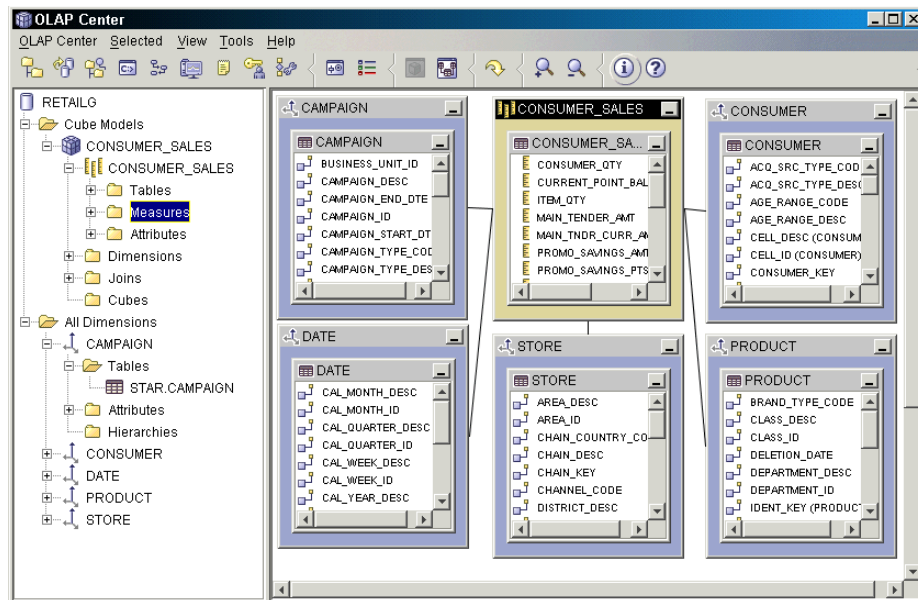


Figure 2-9 The cube model used

2.5.1 Metadata integration of DB2 Cube Views with ERwin v4.x

Computer Associates AllFusion ERwin v4.x Data Modeler is one of the leading database design tools. It supports DB2 as a target database system and allows designing star-schemas databases via its dimensional modeling notation.

Forward engineering from ERwin v4.x to DB2 Cube Views

The goal of this scenario is to demonstrate how an existing ERwin v4.x model can be converted to a DB2 cube model.

The overall process of this metadata conversion is as follows:

1. Using ERwin v4, create the star schema model
2. Using ERwin v4, generate the SQL DDL for this database
3. Using DB2, run this SQL script to create the tables and columns of this schema
4. Using ERwin v4, save the model as XML
5. Using MIMB, convert this ERwin v4 XML file into a DB2 Cube Views XML file
6. Using DB2 Cube Views, import this DB2 Cube Views XML file

Each step of this process is described in the following paragraphs.

1) Using ERwin v4.x, create the star schema model

The ERwin v4 model used in this scenario is shown in Figure 2-10. This database model is in the form of a star-schema and the bridges will use the dimensional information specified in the ERwin v4 model to create a cube model.

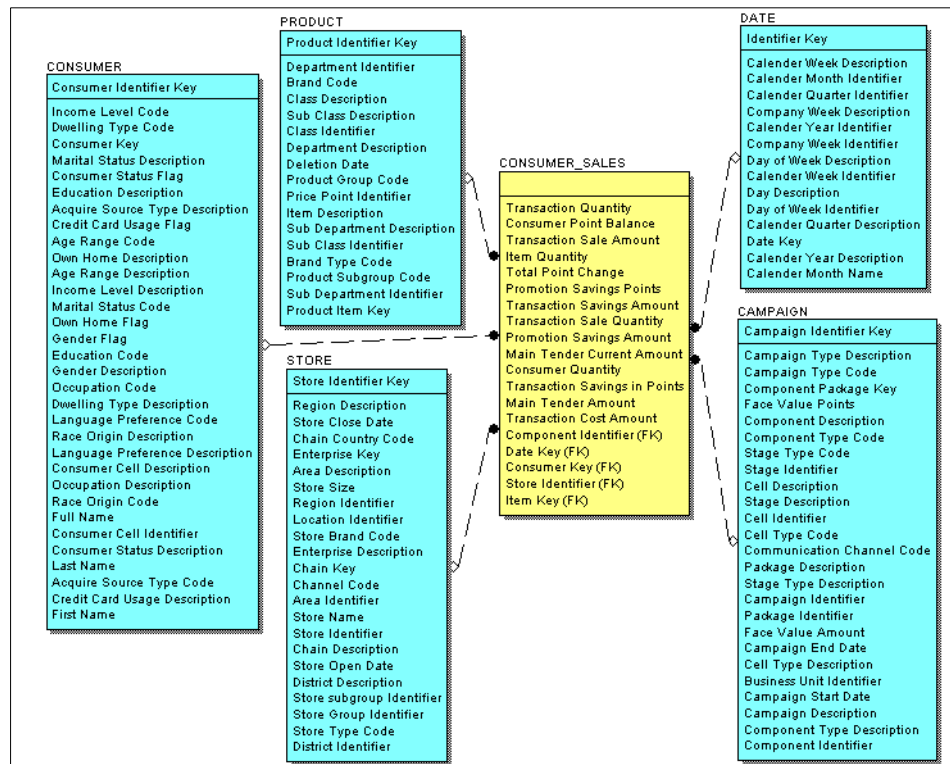


Figure 2-10 Logical view of the ERwin v4 model

During the implementation of this data model in ERwin v4, the dimensional modeling features were enabled, as shown in Figure 2-11. These features can be activated in the menu **Model -> Model Properties** and in the tab **General**.

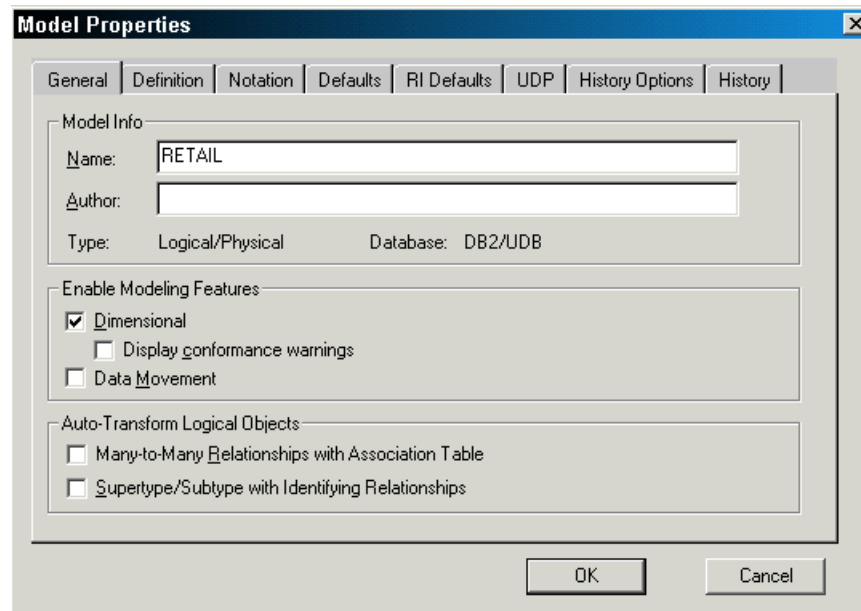


Figure 2-11 Enabling the ERwin v4 dimensional features

This option enables an additional **Dimensional** panel in the **Table** properties window shown in Figure 2-12, so that we can specify the role of each table (**Fact**, **Dimension** or **Outrigger**).

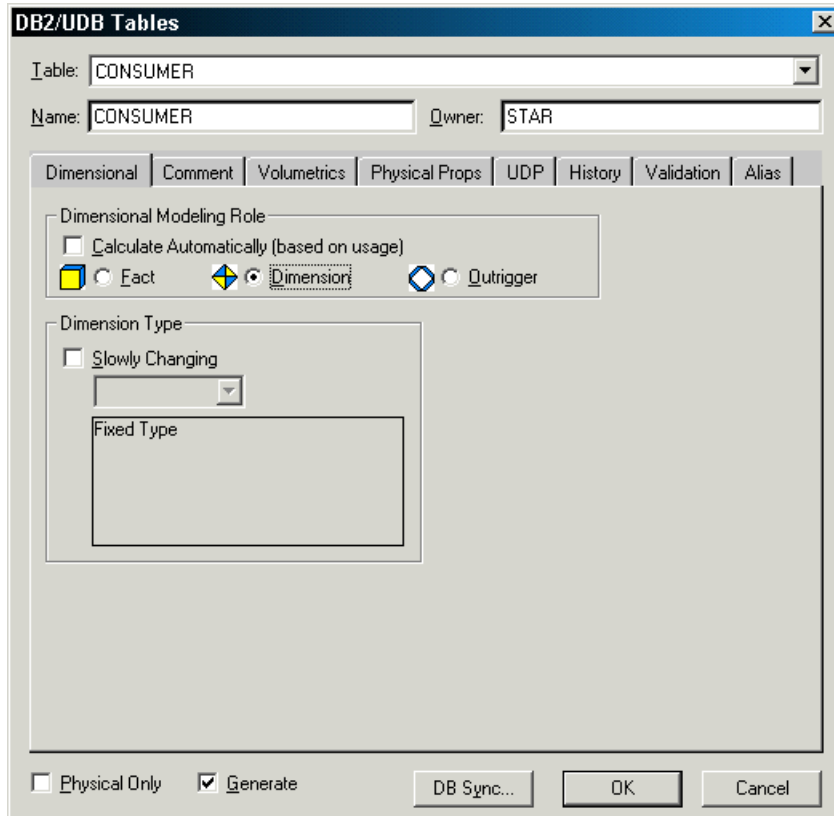


Figure 2-12 Specifying the table dimensional roles

Note: The role of each table should be set explicitly, so that it is saved in the ERwin v4.x XML file format and the bridges can use it. Otherwise, if ERwin v4.x computes the dimensional role of the table automatically, it will not be saved in the ERwin v4.x XML file.

2) Using ERwin v4.x, generate the SQL DDL for this database

Once the model has been designed, the SQL DDL can be generated and the database created in DB2 UDB. In the ERwin v4.x model Physical View, choose the menu **Tools -> Forward Engineer/Schema Generation** to generate the SQL script as shown in Figure 2-13.

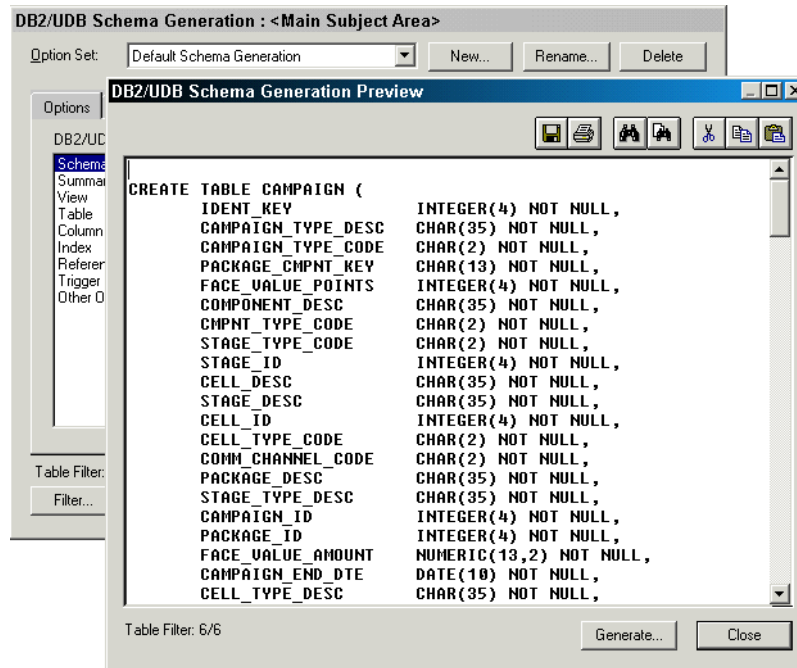


Figure 2-13 DB2 schema generation

3) Using DB2, create the tables and columns of this schema

The SQL script in Figure 2-13 can be executed to create the DB2 Tables. Here is how to execute it using the DB2 Command Window tool:

```

db2 connect to MDSAMPLE
db2 set current schema = STAR
db2 -tvf C:\Temp\star.sql

```

Note: The database schema must be created in DB2 before the cube model is imported into DB2 Cube Views.

At this point, the database has been setup and is ready to receive the cube model metadata.

4) Using ERwin v4.x, save the model as XML

The next step of this process is to save the ERwin v4 model as an XML file. The bridge will use this file as input.

When the model is loaded in ERwin v4.x, choose **Save As** from the **File** menu, select the XML format type in the **Save as type list**, type the file name for the model you are saving in the **File name text** box and click **Save**.

Note: If the ERwin v4 model is logical and physical (business names have been defined in the logical view), the **Save as XML** process described above will not properly save the physical names into the XML file, if ERwin v4.x automatically computed these physical names.

To work around this issue, you can use an alternate **Save as XML** feature of ERwin v4 located in menu **Tools -> Add-Ins' -> Advantage Repository Export**. It produces a slightly different XML file format where the physical names are expanded.

This issue does not occur if the ERwin v4.x model is physical only.

5) Using MIMB, convert ERwin XML into DB2 Cube Views XML file

Start the MIMB tool and select the import bridge labeled **CA ERwin 4.0 SP1® to 4.1**, and import your ERwin v4.x XML file, as shown in Figure 2-14.

The MIMB *validation* feature checks that the model is valid according to the rules of the MIR metamodel. If something is wrong (a key is empty or a column does not belong to any table or a foreign key does not reference a primary key), it will display a warning or error message.

The *subsetting* feature allows you to create a subset of the model so that the model exported to the destination tool only contains the few tables you chose.

Both features are described in the online documentation:

<http://www.metaintegration.net/Products/MIMB/Documentation/>

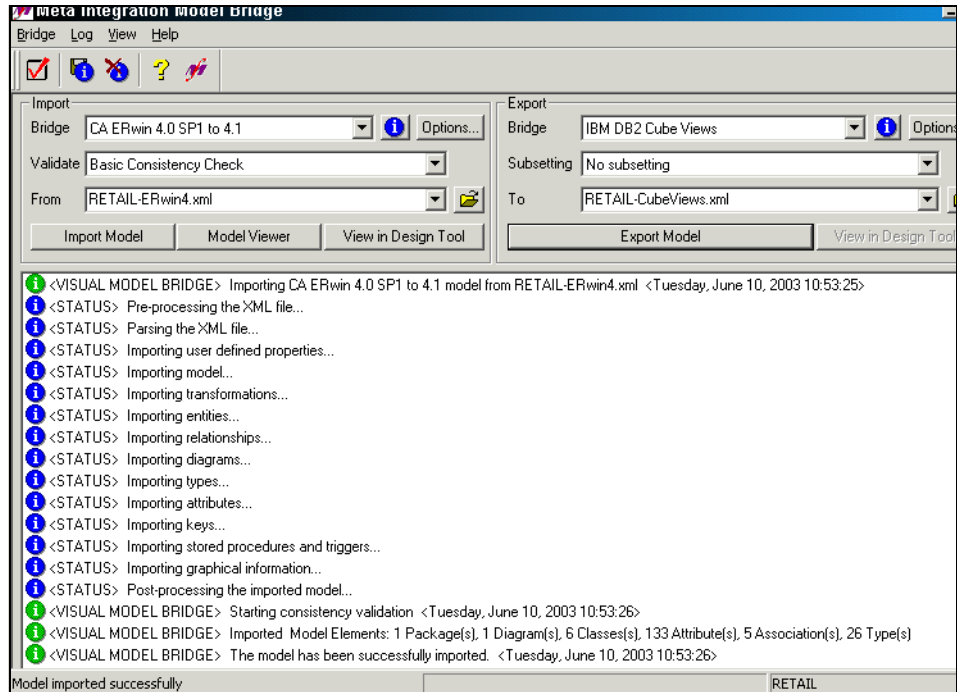


Figure 2-14 Importing the ERwin v4 model into MIMB

Select the export bridge labeled **IBM DB2 Cube Views** and click the **Options** button to specify the export parameters as shown in Figure 2-15.

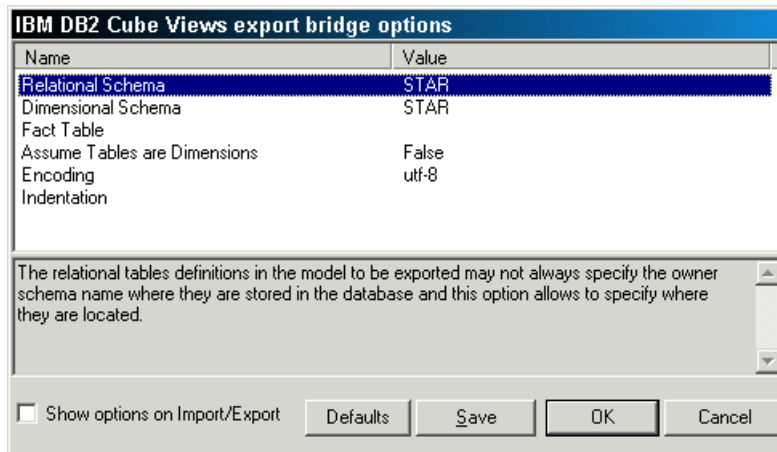


Figure 2-15 Specifying the export bridge parameters

The export parameters used in this scenario are as follows:

- ▶ The DB2 schema for the tables of the model is *STAR*, as the model may not always specify where each table is located.
- ▶ The cube model to be created will be located in the same *STAR* DB2 schema.
- ▶ We specify that the source encoding of the ERwin v4 model is utf-8.
- ▶ The other options are left with their default value.

Close this window, specify the name of the DB2 Cube Views XML file to be created, and click the **Export** button.

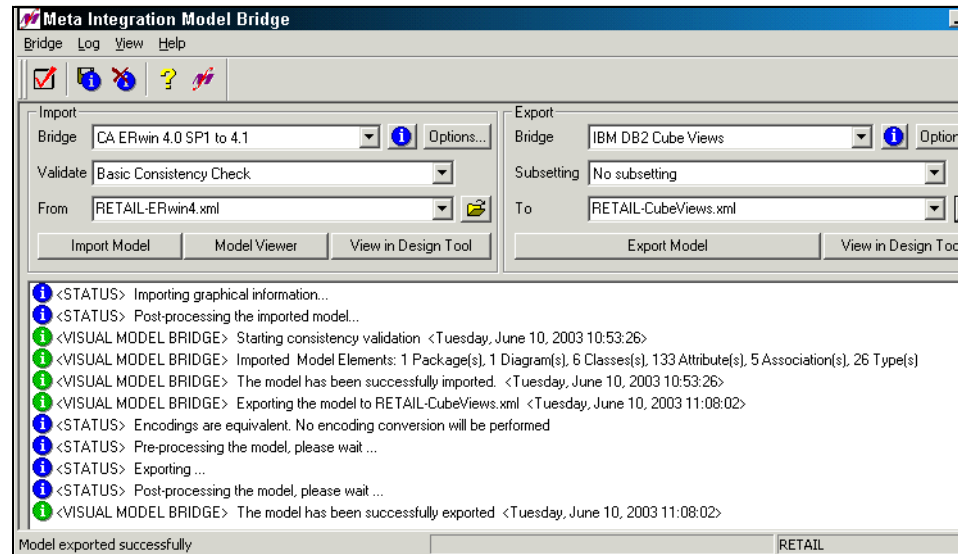


Figure 2-16 Exporting the model to DB2 Cube Views

6) Using DB2 Cube Views, import this DB2 Cube Views XML file

At this point, the cube model XML file has been created and is ready to be opened into the DB2 OLAP Center graphical tool. Just start OLAP Center, connect to your database, and choose **Import** in the OLAP Center menu as shown in Figure 2-17.

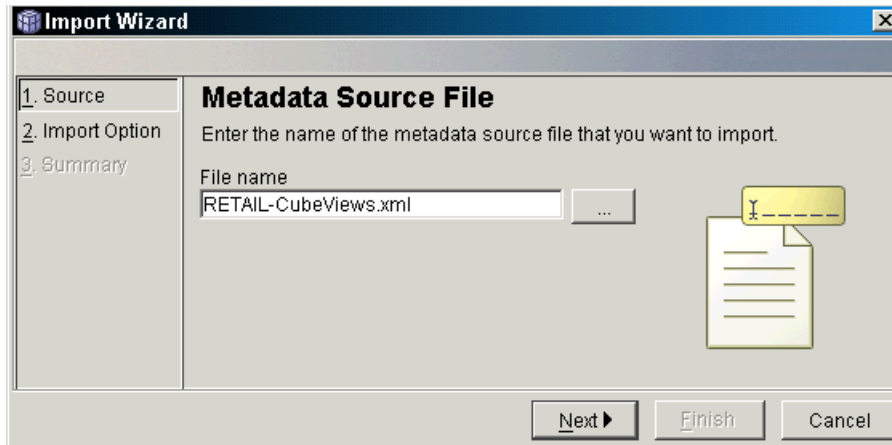


Figure 2-17 Specifying the XML file to import into OLAP Center

The content of the XML file is displayed in Figure 2-18, which allows controlling how this metadata should be imported, in case there is already some metadata in place and object name collision should occur:

- ▶ Either update the existing objects with the new imported version.
- ▶ Or keep the current version of the metadata.

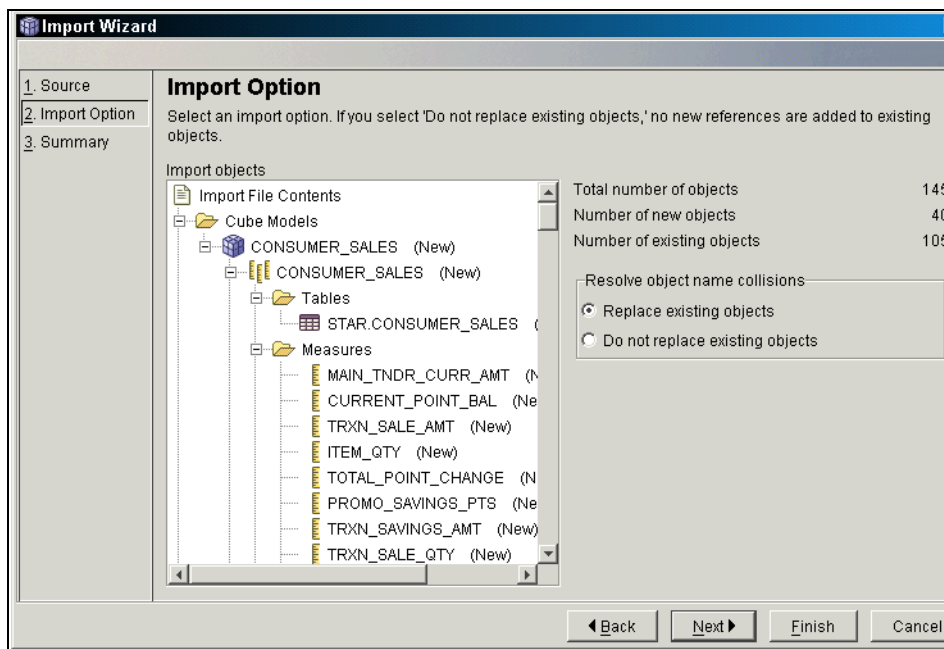


Figure 2-18 Controlling how the metadata is imported into OLAP Center

Finally, the ERwin star schema metadata converted and imported into OLAP Center will provide the DB2 cube model in Figure 2-9 on page 19.

The business names and descriptions defined in ERwin v4.x are also converted to the cube model, as shown in Figure 2-19.

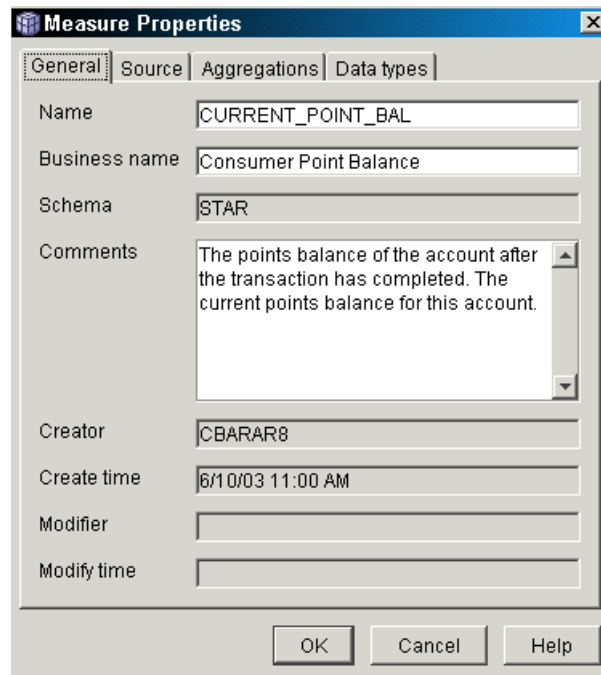


Figure 2-19 The ERwin v4 business names and description are also converted

Congratulations, the ERwin v4.x star schema model was converted to DB2 Cube Views!

Reverse engineering from DB2 Cube Views to ERwin v4.x

The goal of this scenario is to demonstrate how an existing DB2 Cube Views model can be converted into an ERwin v4.x Model.

The overall process of this metadata conversion is as follows:

1. Using DB2 Cube Views, export your cube model as an XML file,
2. Using MIMB, convert this DB2 Cube Views XML file into an ERwin v4.x XML file.
3. Using ERwin v4.x, import this XML file.

Each step of this process is described in the following paragraphs.

1) Using DB2 Cube Views, export your cube model as an XML file

The DB2 Cube Views model used in this scenario is the one shown in Figure 2-9 on page 19.

The first step of the conversion process is to save this cube model into an XML file. Use the OLAP Center menu **OLAP Center -> Export** and select the cube model to be exported as shown in Figure 2-20.

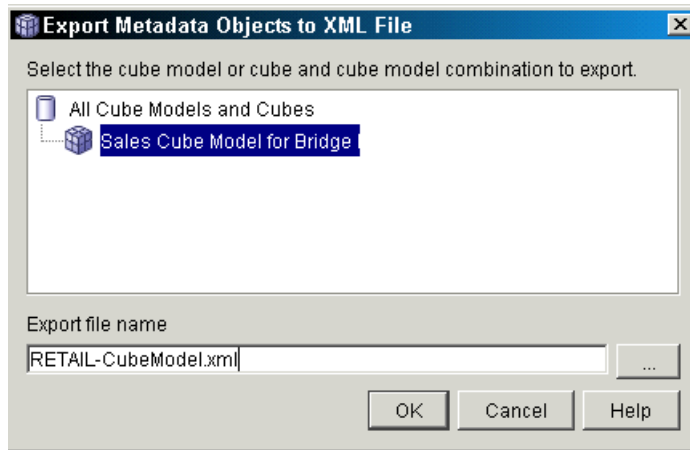


Figure 2-20 Exporting from the DB2 cube model as XML

2) Using MIMB, convert DB2 Cube Views XML into ERwin XML file

Start the MIMB software, select the import bridge labeled **IBM DB2 Cube Views** and import your model. Select the export bridge labeled **CA ERwin 4.0 SP1 to 4.1**, select the name of the export ERwin v4 XML file, and click the **Export Model** button as shown in Figure 2-21.

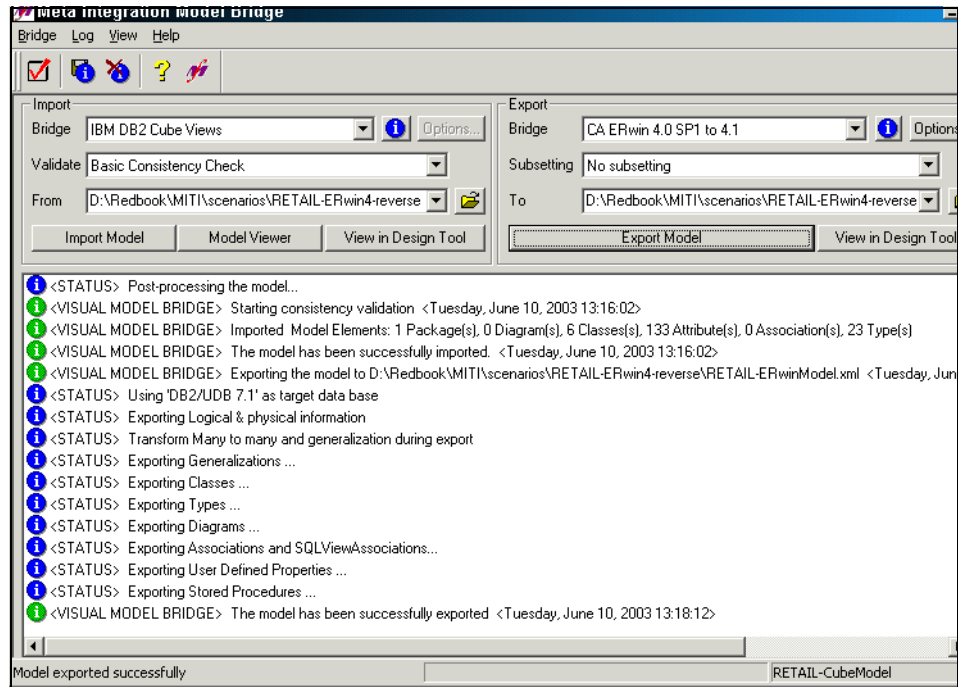


Figure 2-21 Converting the cube model XML file to an ERwin v4 XML file

3) Using ERwin v4.x, import this XML file

At this point, you can open the generated XML file into ERwin v4 using menu **File** -> **Open**. When the file choice window appears, select XML Files (*.xml) in the 'Files of type list box and select the XML file produced by MIMB.

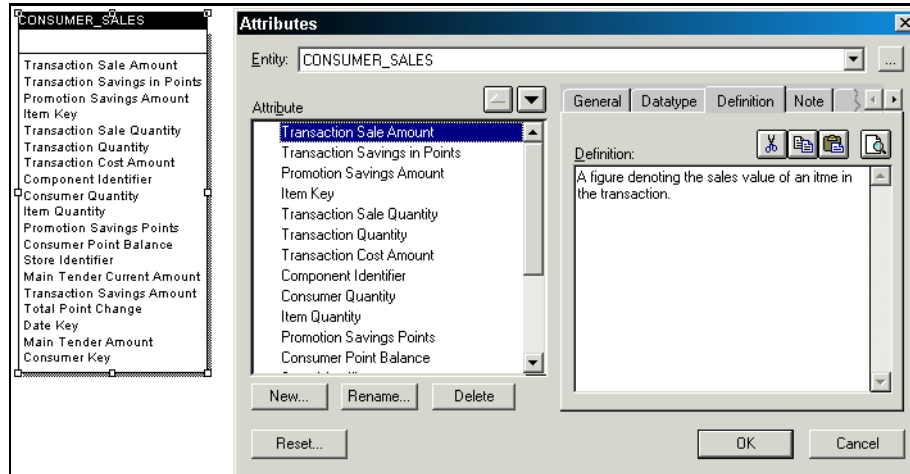


Figure 2-22 Cube model converted to ERwin v4 with business names

The cube model converted to ERwin v4.x contains the business names and descriptions, and the logical view is shown in Figure 2-23.

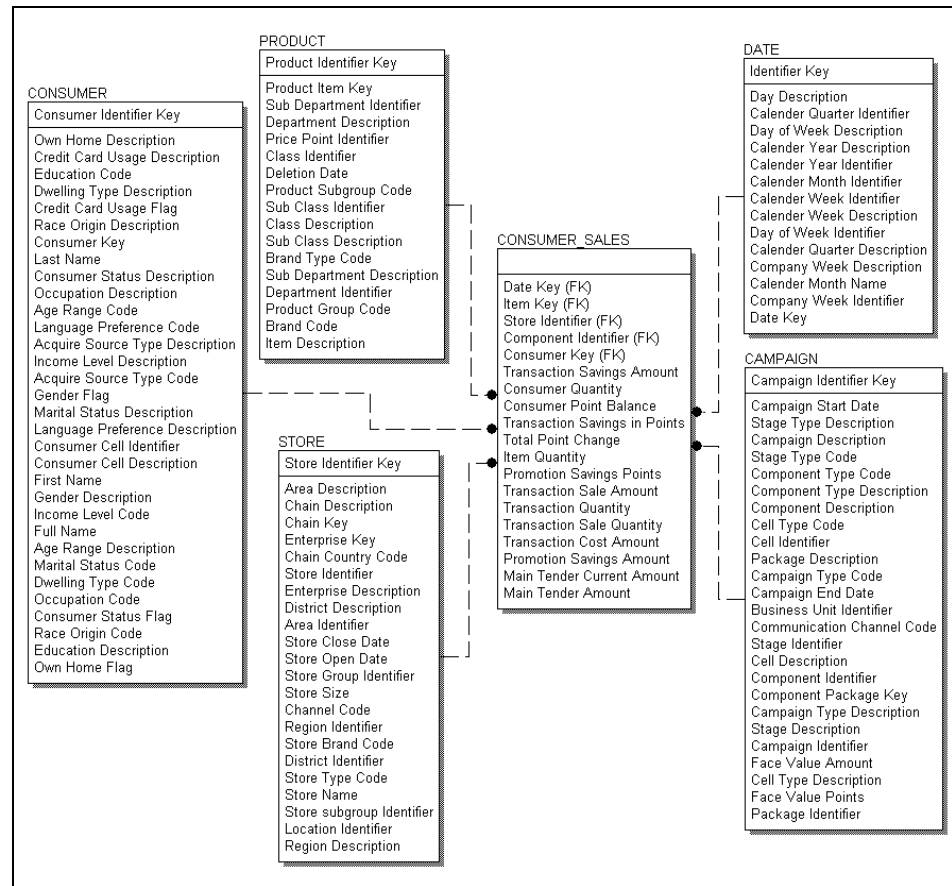


Figure 2-23 Logical view of the ERWin model

Congratulations, the cube model was converted to ERwin v4.x!

2.5.2 Metadata integration of DB2 Cube Views with ERwin v3.x

Computer Associates ERwin 3.x is still one of the leading database design tools. It supports DB2 UDB as target database system and allows designing star-schemas databases via its dimensional modeling notation.

Note: The ERwin v3.x ERX file format is very widely used as a de facto standard means of exchanging relational database metadata. Many design tools support it and therefore this scenario can also be used to interact and exchange metadata with them.

A non-exhaustive list of such tools would include Embarcadero ER/Studio, Microsoft® Visio, Sybase PowerDesigner, Computer Associates Advantage Repository, Casewise, and Informatica PowerCenter.

Forward engineering from ERwin v3.x to DB2 Cube Views

The goal of this scenario is to demonstrate how an existing ERwin model can be converted to a DB2 cube model.

The overall process of this metadata conversion is as follows:

1. Using ERwin, create the star schema model.
2. Using ERwin, generate the SQL DDL for this database.
3. Using DB2, run this SQL script to create the tables and columns of this schema.
4. Using ERwin, save the model as ERX.
5. Using MIMB, convert this ERwin ERX file into a DB2 Cube Views XML file.
6. Using DB2 Cube Views, import this DB2 Cube Views XML file.

Each step of this process is described in the following paragraphs.

1) Using ERwin, create the star schema model

The ERwin model used in this scenario is the one shown in Figure 2-24. This database model is in the form of a star-schema and the bridges will use the dimensional information specified in the model to create a cube model.

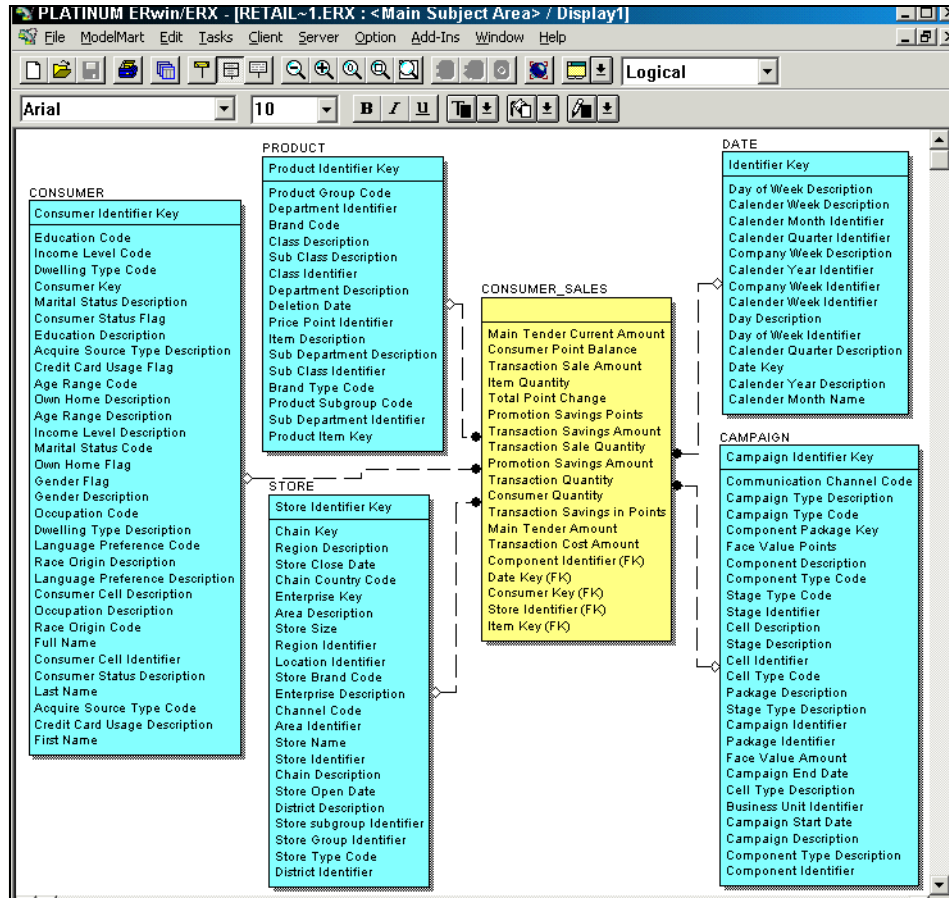


Figure 2-24 Logical view of the ERwin model

During the implementation of this data model in ERwin, the dimensional modeling features were enabled, as shown in Figure 2-25. These features can be activated in the menu **Options -> Preferences** and in the tab **Methodology**.

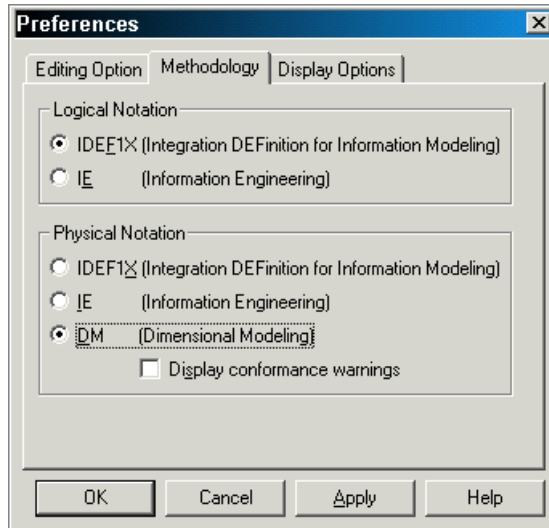


Figure 2-25 Enabling the ERwin dimensional features

This option enables an additional **Dimensional** panel in the tables properties, so that we can specify the role of each table (for example **Fact**, **Dimension** or **Outrigger**), as shown in Figure 2-26.

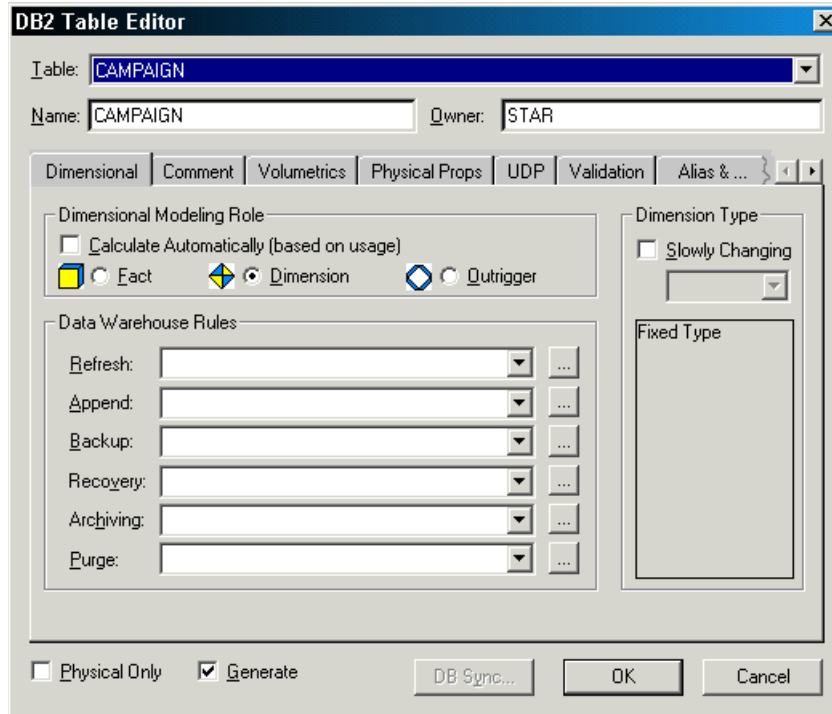


Figure 2-26 Specifying the tables dimensional roles

Note: The role of each table should be set explicitly, so that it is saved in the ERwin ERX file format and the bridges can be used. Otherwise, if ERwin computes the dimensional role of the table automatically, it is not saved in the ERwin ERX file.

2) Using ERwin, generate the SQL DDL for this database

Once the model has been designed, the SQL DDL can be generated and the database created in DB2 UDB. In the ERwin model physical view, choose the menu **Tasks -> Forward Engineer/Schema Generation** to generate the SQL script as in Figure 2-13 on page 23.

3) Using DB2, create the tables and columns of this schema

The SQL script can be executed to create the DB2 Tables through the DB2 Command Window tool (or any other tool).

4) Using ERwin, save the model as ERX

The next step of this process is to save the ERwin model as an ERX file. The bridge will use this file as input.

When the model is loaded in ERwin, choose **Save As** from the File menu, select the ERX format type in the File format area, type the file name for the model you are saving in the **File name** text box and click **OK** as shown in Figure 2-27.

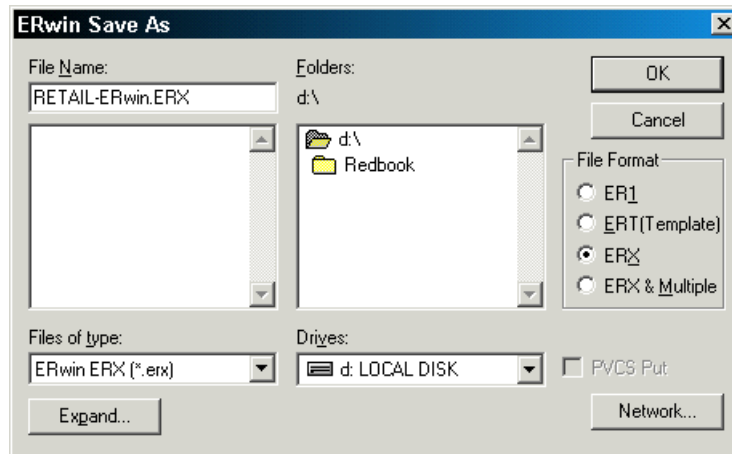


Figure 2-27 Saving the model as ERX

Note: When saving a logical and physical model, the physical names of tables, columns, and keys may not always be saved into the ERX file. Indeed, when ERwin is used to manage the automatic generation of physical names from logical names, only the generation rules are saved.

One solution is to make sure all physical names are explicitly set, therefore not relying on any generation rules from the logical names.

Alternatively, when saving a model as ERX, the dialog box offers a button called **Expand**, which opens another dialog box labeled **Expand Property Values**. Select the **DB2** tab of this window, and check the appropriate names to expand (column name) as shown in Figure 2-28.

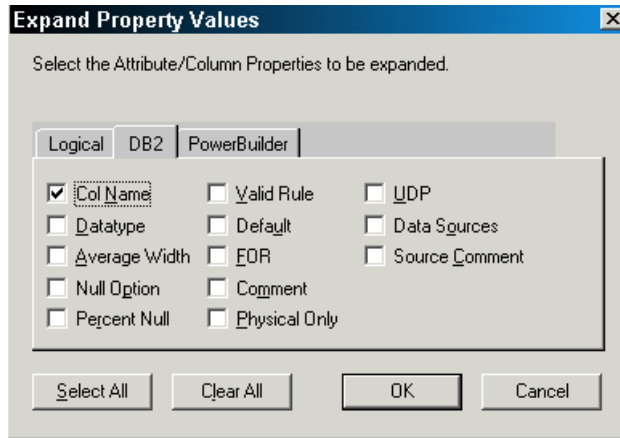


Figure 2-28 ERwin names expansion feature

5) Using MIMB, convert ERwin ERX file into DB2 Cube Views XML

Start the MIMB tool and select the import bridge labeled “CA ERwin 3.0 to 3.5.2”, and import your ERX file, as shown in Figure 2-29.

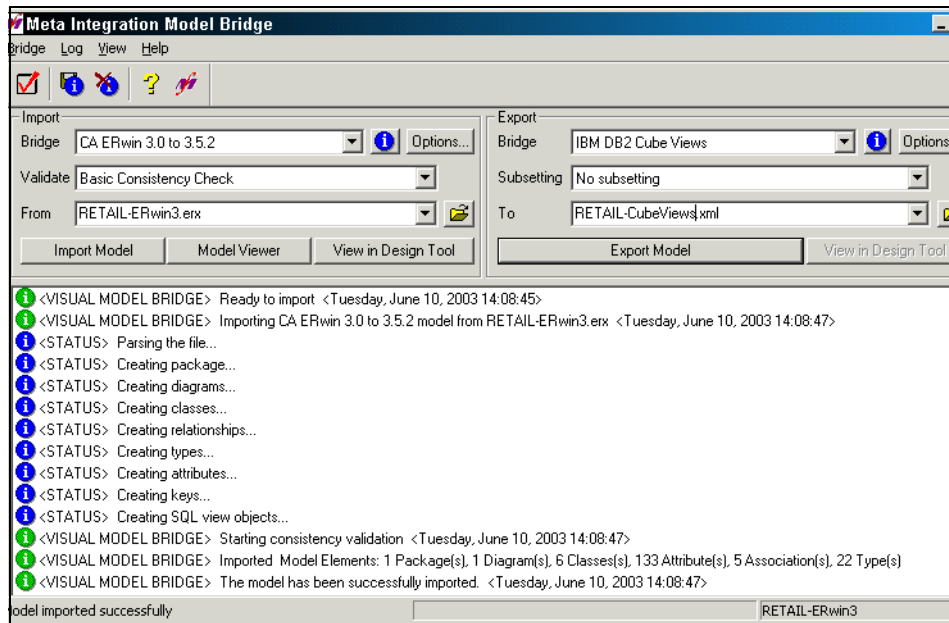


Figure 2-29 Importing the ERwin model into MIMB

Select the export bridge labeled **IBM DB2 Cube Views** and click on the **Options** button to specify the export parameters as in Figure 2-30.

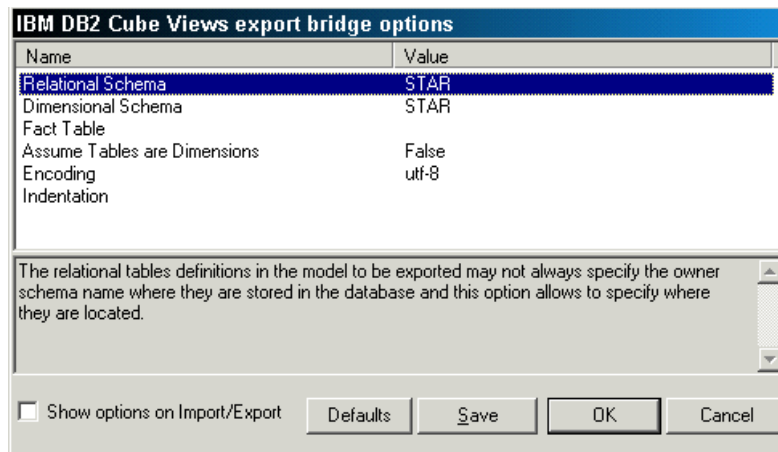


Figure 2-30 Specifying the export bridge parameters

The export parameters used in this scenario are as follows:

- ▶ The DB2 schema for the tables of the model is *STAR*, as the model may not always specify where each table is located.
- ▶ The cube model to be created will be located in the same *STAR* DB2 schema.
- ▶ The other options are left with their default value.

Close this window, specify the name of the DB2 Cube Views XML file to be created, and click the **Export** button (see Figure 2-31).

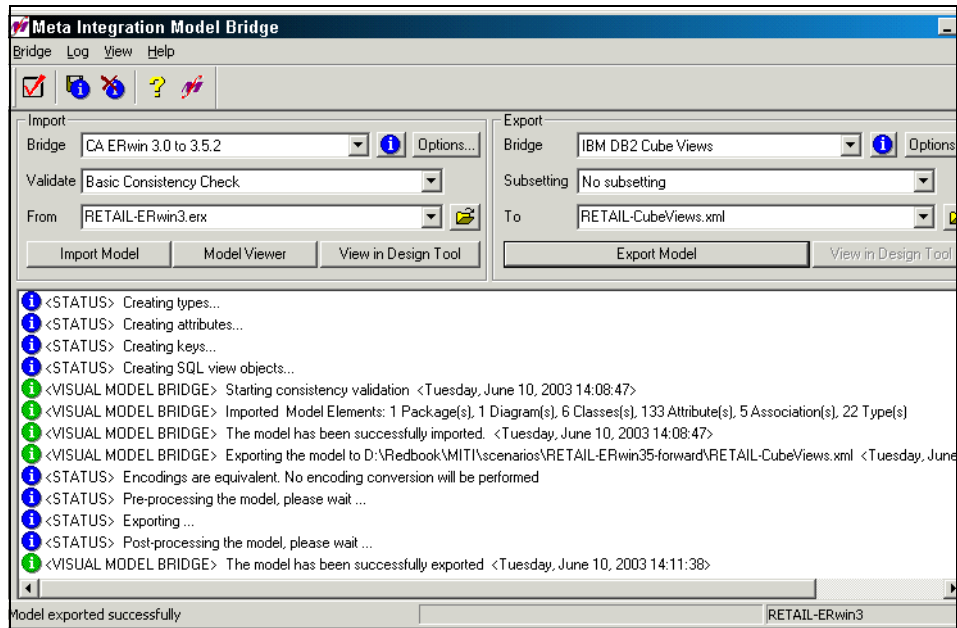


Figure 2-31 Exporting the model to DB2 Cube Views

6) Using DB2 Cube Views, import this DB2 Cube Views XML file

At this point, the cube model XML file has been created and is ready to be opened into the OLAP Center graphical tool. Just start OLAP Center, connect to your database and choose **Import** in the OLAP Center menu to get the display in Figure 2-32.

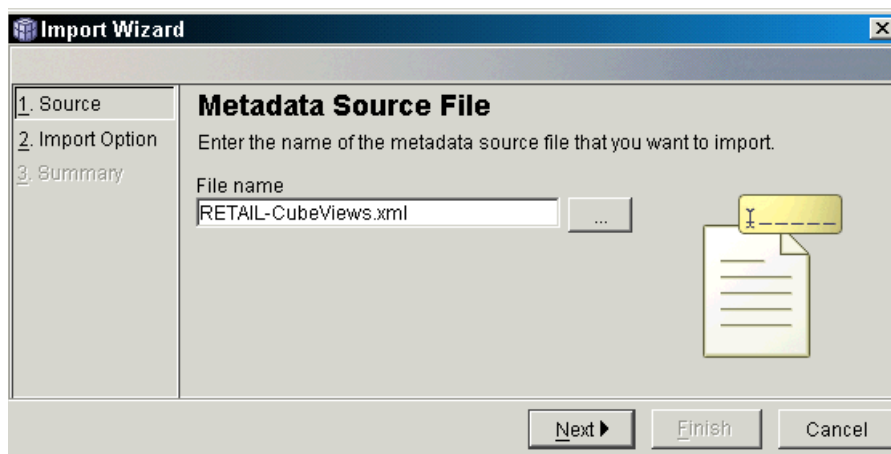


Figure 2-32 Specifying the XML file to import into OLAP Center

The content of the XML file is displayed in Figure 2-33, which allows controlling how this metadata should be imported, in case there is already some metadata in place and object name collision should occur:

- ▶ Either update the existing objects with the new imported version.
- ▶ Or keep the current version of the metadata.

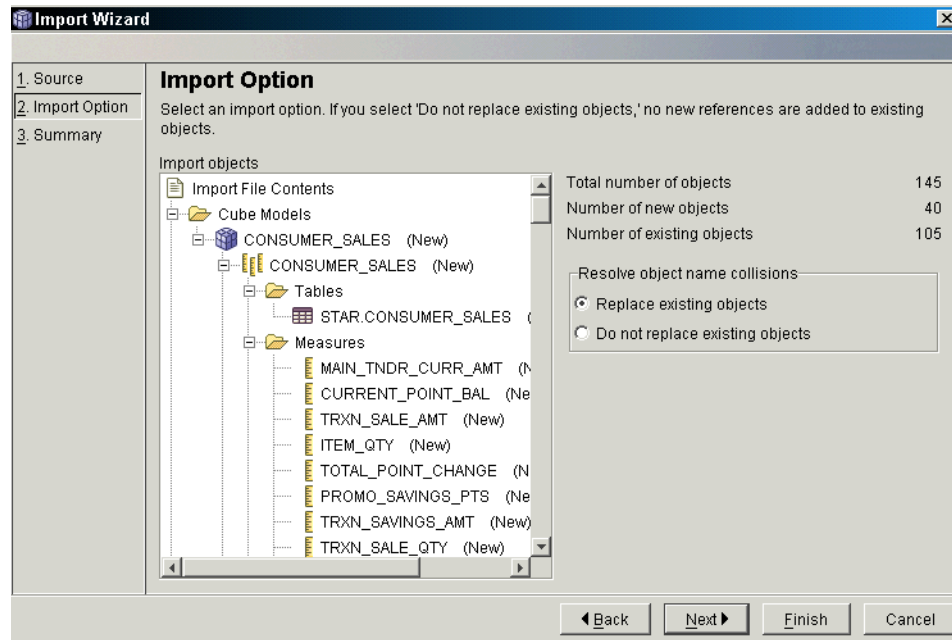


Figure 2-33 Controlling how the metadata is imported into OLAP Center

Finally, the ERwin star schema metadata converted and imported into OLAP Center will provide the DB2 cube model in Figure 2-9 on page 15.

The business names and descriptions defined in ERwin are also converted to the cube model, as shown in Figure 2-34.

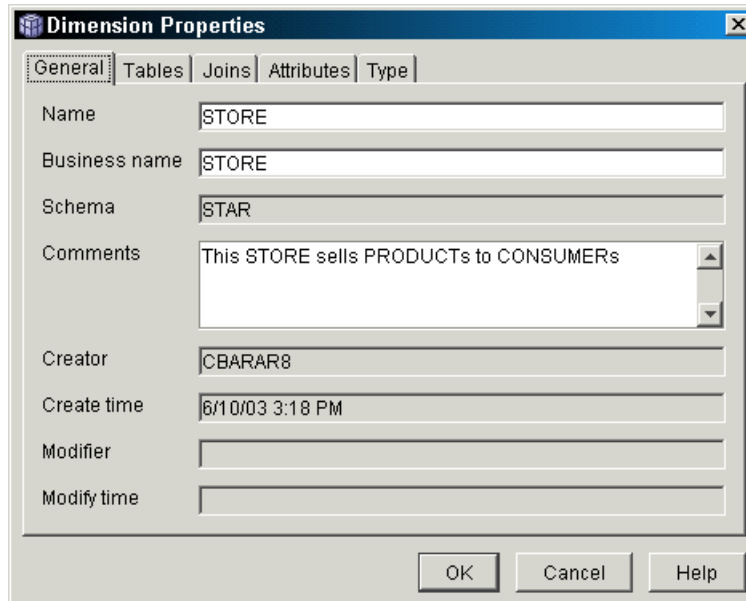


Figure 2-34 The ERwin business names and descriptions are also converted

The ERwin objects' business name and description are also converted.

Congratulations, the ERwin star schema model was converted to DB2 Cube Views!

You can now edit this model in DB2 OLAP Center to enrich it with additional OLAP metadata such as hierarchies, levels, cubes, calculated measures, and more.

Reverse engineering from DB2 Cube Views to ERwin v3.x

The goal of this scenario is to demonstrate how an existing DB2 Cube Views model can be converted into an ERwin model.

The overall process of this metadata conversion is as follows:

1. Using DB2 Cube Views, export your cube model as an XML file.
2. Using MIMB, convert this DB2 Cube Views XML file into an ERwin 3.x ERX file.
3. Using ERwin, import this ERX file.

Each step of this process is described in the following paragraphs.

1) Using DB2 Cube Views, export your cube model as an XML file

The DB2 Cube Views model used in this scenario is the one shown in Figure 2-9 on page 19.

This step has already been detailed in “1) Using DB2 Cube Views, export your cube model as an XML file” on page 29. The DB2 cube model is saved into an XML file using **OLAP Center > Export** in DB2 Cube Views.

2) Using MIMB, convert DB2 Cube Views XML file into ERX file

Start the MIMB software, select the import bridge labeled IBM DB2 Cube Views’ and import your model. Select the export bridge labeled “CA ERwin 3.0 to 3.5.2”, select the name of the export ERwin ERX file, and click the **Export Model** button.

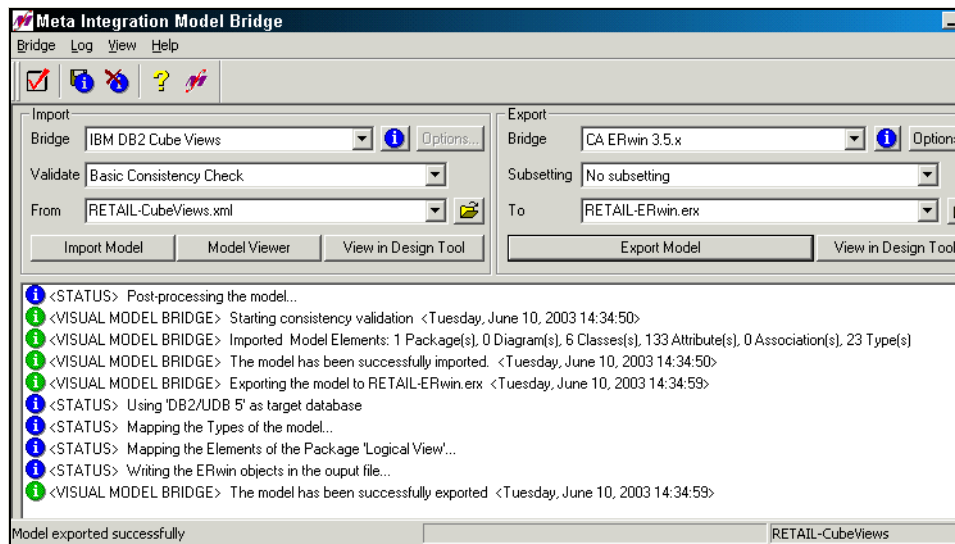


Figure 2-35 Converting the DB2 cube model XML to an ERwin ERX file

3) Using ERwin, import this ERX file

At this point, you can open the generated ERX file into ERwin using menu **File -> Open**. When the file choice window appears, select ERwin ERX (*.erx) in the **List files of type** list box and select the ERX file produced by MIMB as shown in Figure 2-36.

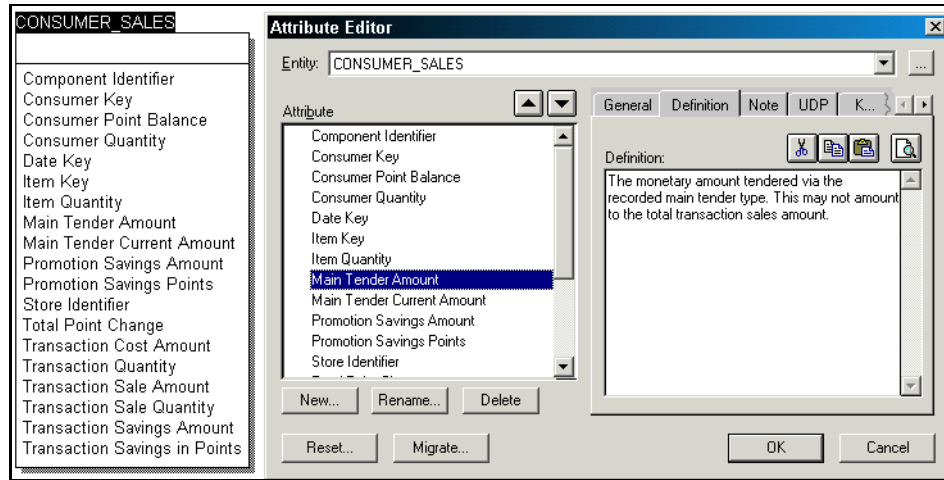


Figure 2-36 DB2 cube model converted to ERwin

The cube model converted to ERwin contains the business names and descriptions, and a logical view is displayed in Figure 2-37.

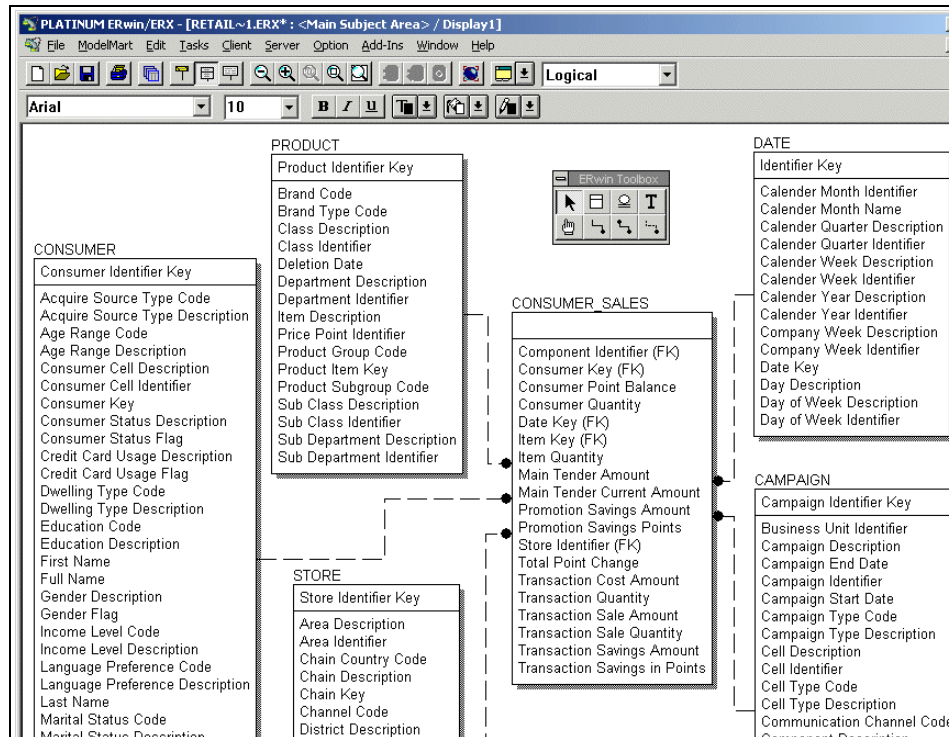


Figure 2-37 The cube model reversed engineered to ERwin 3.x

Congratulations, the cube model was converted to ERwin!

2.5.3 Metadata integration of DB2 Cube Views with PowerDesigner

Sybase PowerDesigner is one of the leading database design tools. PowerDesigner allows designing Conceptual Data Models (CDM) as well as Physical Data Models (PDM). It supports DB2 UDB as target database system of a physical data model, provides star schema database design features in *physical diagrams* and also provides multidimensional/OLAP modeling features in *multidimensional diagrams*. In this scenario, we will demonstrate the forward and reverse engineering of star schema in PowerDesigner PDM *physical diagrams*.

Forward engineering from PowerDesigner to DB2 Cube Views

The goal of this scenario is to demonstrate how an existing PowerDesigner PDM model can be converted to a DB2 cube model.

The overall process of this metadata conversion is as follows:

1. Using PowerDesigner, create the star schema PDM model.
2. Using PowerDesigner, generate the SQL DDL for this database.
3. Using DB2, run this SQL script to create the tables and columns of this schema.
4. Using PowerDesigner, save the model as PDM XML.
5. Using MIMB, convert this PowerDesigner XML file into a DB2 Cube Views XML file.
6. Using DB2 Cube Views, import this Cube Views XML file to import the metadata.

Each step of this process is described in the following paragraphs.

1) Using PowerDesigner, create the star schema PDM model

The PowerDesigner model used in this scenario is the one shown in Figure 2-38. This database model is in the form of a star-schema, and the bridge will use the dimensional information specified in the model to create a cube model.

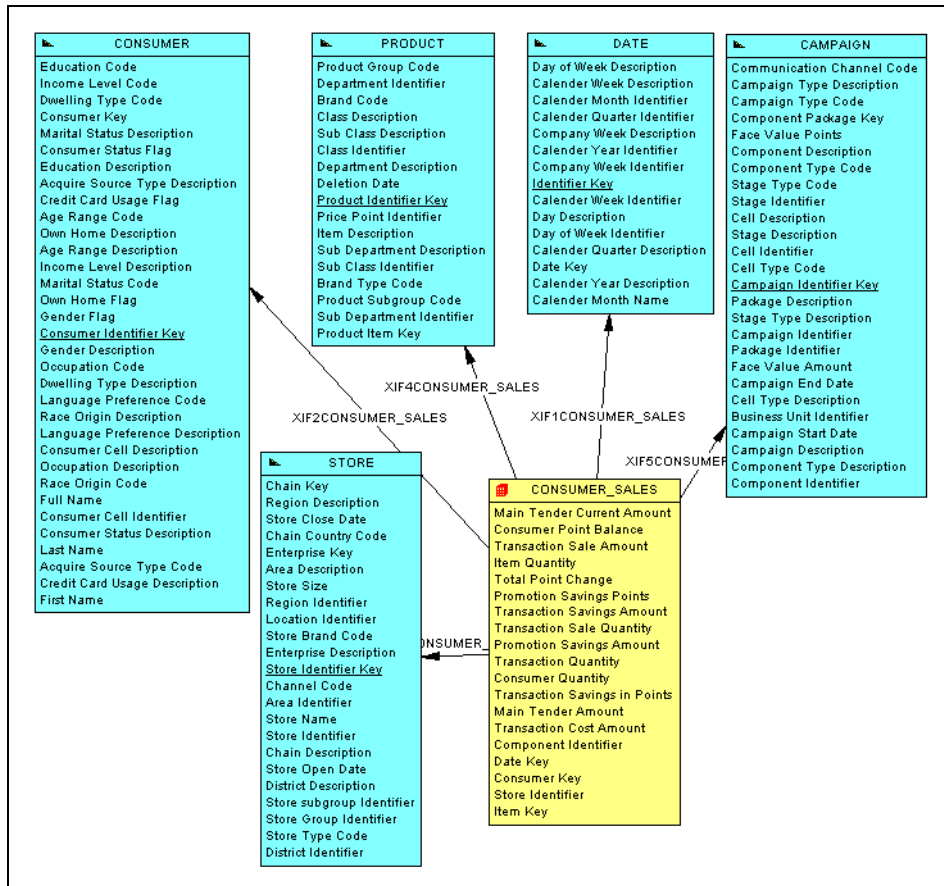


Figure 2-38 Logical view of the PowerDesigner PDM model

During the implementation of this data model in PowerDesigner, the dimensional modeling features of the PDM *physical diagram* were used. A dimensional type was specified on each table (**Fact** or **Dimension**) as shown in Figure 2-39.

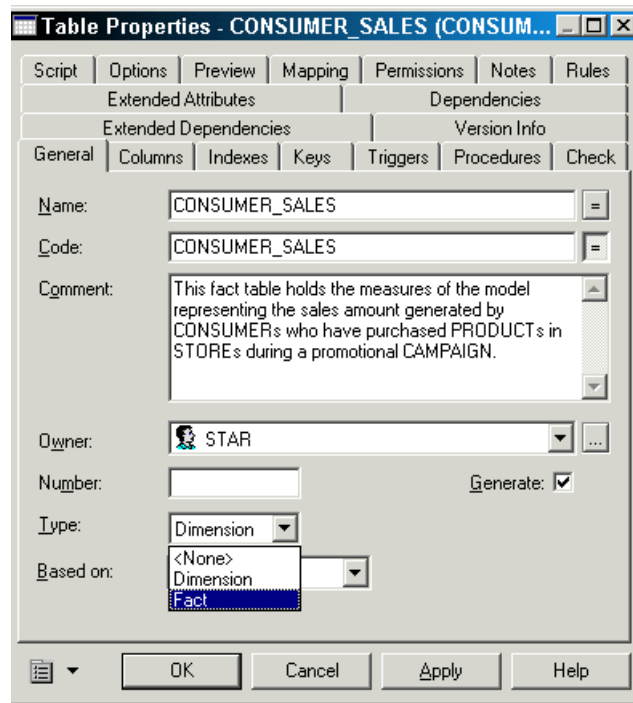


Figure 2-39 Specifying the tables' dimensional type

Documentation was also specified in this model, in the form of a **Comment** field on the objects as shown in Figure 2-40.

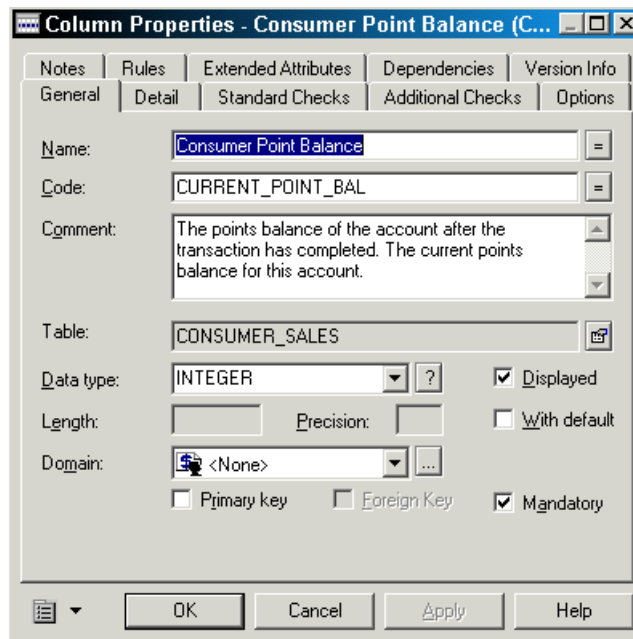


Figure 2-40 Adding documentation to the PowerDesigner model

2) Using PowerDesigner, generate the SQL DDL for this database

Once the model has been designed, the SQL DDL can be generated and the database created in DB2 UDB. In PowerDesigner, choose the menu **Database -> Generate Database** to generate the SQL script as shown in Figure 2-41.

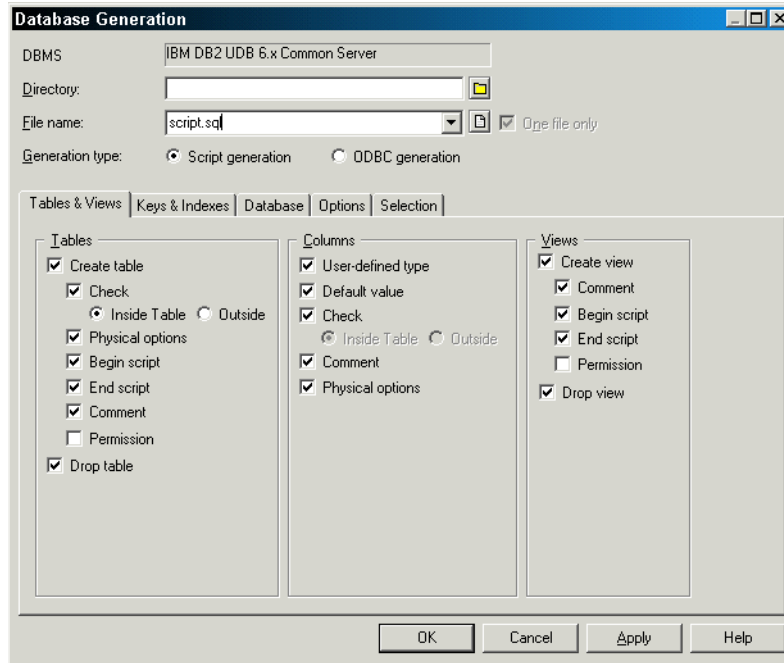


Figure 2-41 DB2 schema generation

3) Using DB2, create the tables and columns of this schema

This SQL script generated can be executed under DB2 Command Tool to create the DB2 tables as already discussed in “3) Using DB2, create the tables and columns of this schema” on page 36.

4) Using PowerDesigner, save the model as PDM XML

The next step of this process is to save the PowerDesigner model as a PDM XML file. The bridge will use this file as the input.

When the model is loaded in PowerDesigner, choose **Save As** from the File menu, select the **Physical Data Model (xml) (*.pdm)** format in the **Save as type list**, type the file name for the model you are saving in the **File name** text box and click **Save**.

Note: PowerDesigner also allows sharing the definition of metadata (tables, views, relationships) across different models via the notion of shortcuts. If your model contains shortcuts to external objects defined in other models, the definition of the referenced objects may not be completely saved in the current PDM file.

We recommend not using such external shortcuts for the purpose of metadata integration with DB2 Cube Views.

5) Using MIMB, convert PowerDesigner to DB2 Cube Views

Start the MIMB tool and select the import bridge labeled **Sybase PowerDesigner PDM 7.5 to 9.5**, and import your PowerDesigner PDM XML file, as shown in Figure 2-42.

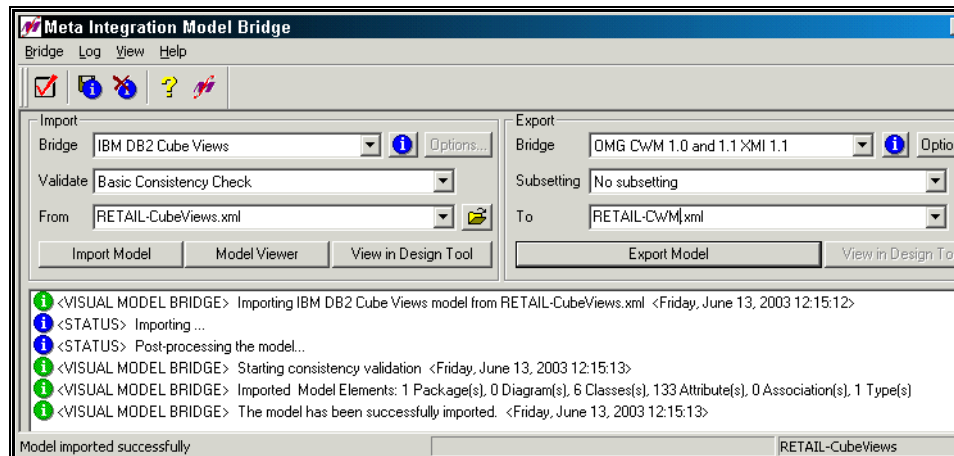


Figure 2-42 Importing the PowerDesigner model into MIMB

Select the export bridge labeled **IBM DB2 Cube Views** and click the **Options** button to specify the export parameters as shown in Figure 2-43.

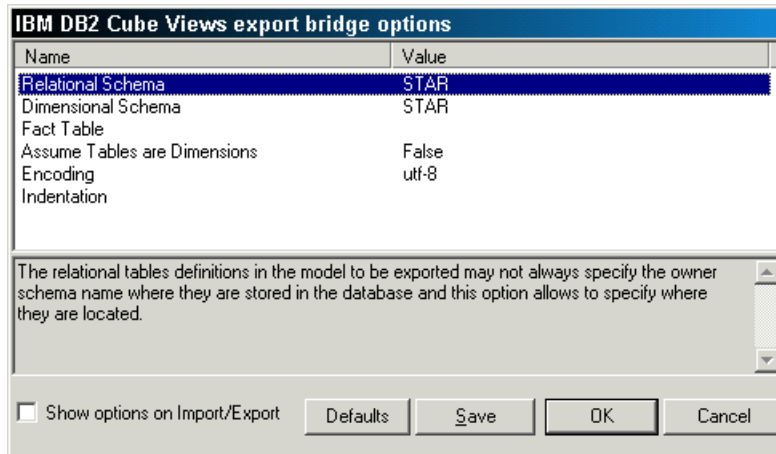


Figure 2-43 Specifying the export bridge parameters

The export parameters used in this scenario are as follows:

- ▶ The DB2 schema for the tables of the model is 'STAR', as the model may not always specify where each table is located.
- ▶ The cube model to be created will be located in the same 'STAR' DB2 schema.
- ▶ We specify that the encoding of the source model is utf-8.
- ▶ The other options are left with their default value.

Close this window, specify the name of the DB2 Cube Views XML file to be created, and click the **Export** button to get the display shown in Figure 2-44.

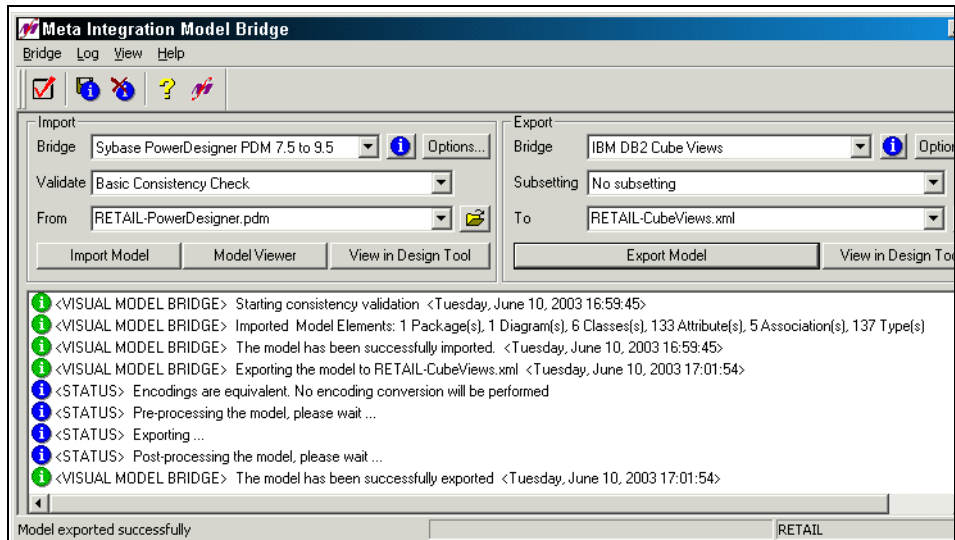


Figure 2-44 Exporting the model to DB2 Cube Views

6) Using DB2 Cube Views, import the DB2 Cube Views XML file

At this point, the cube model XML file has been created and is ready to be opened into the OLAP Center graphical tool. Just start OLAP Center, connect to your database, and choose **Import** in the OLAP Center menu to get the display shown in Figure 2-45.

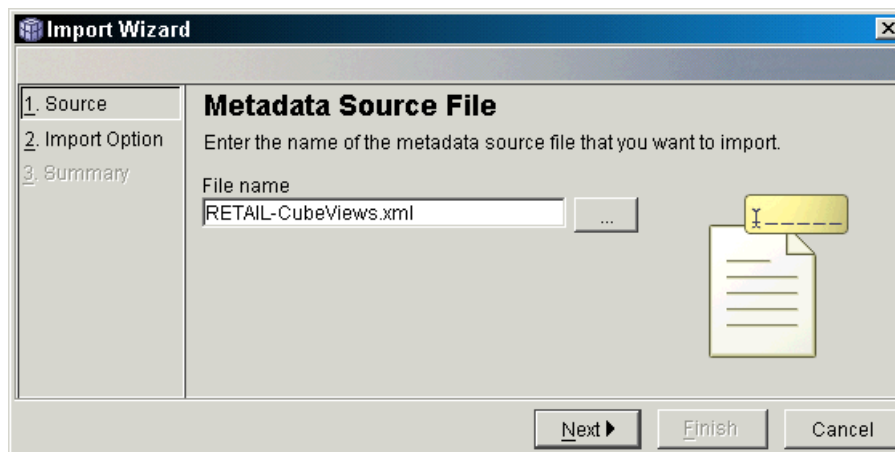


Figure 2-45 Specifying the XML file to import into OLAP Center

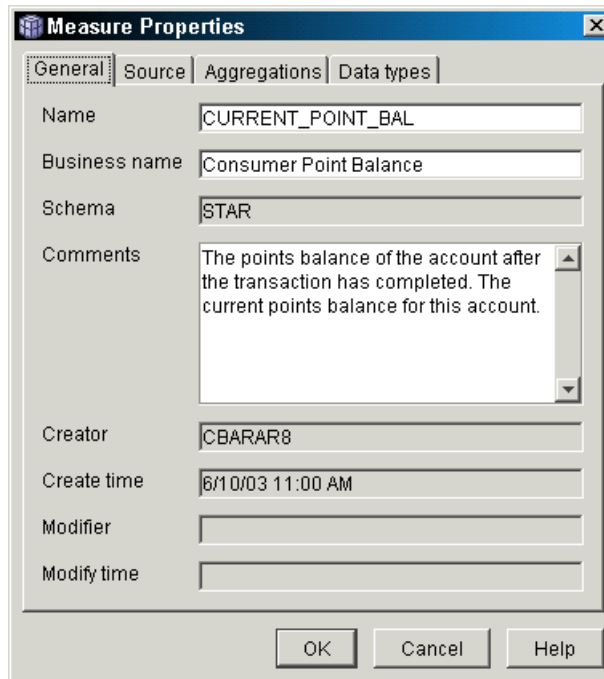


Figure 2-47 The PowerDesigner business names and descriptions are converted

Congratulations, the PowerDesigner star schema model was converted to DB2 Cube Views!

Reverse engineering from DB2 Cube Views to PowerDesigner

The goal of this scenario is to demonstrate how an existing DB2 Cube Views model can be converted into a PowerDesigner PDM physical diagram.

The overall process of this metadata conversion is as follows:

1. Using DB2 Cube Views, export your cube model as an XML file.
2. Using MIMB, convert this DB2 Cube Views XML file into an PowerDesigner XML file.
3. Using PowerDesigner, import this XML file.

Each step is detailed in the following paragraphs.

1) Using DB2 Cube Views, export your cube model as an XML file

The DB2 Cube Views model used in this scenario is the one shown in Figure 2-9 on page 19.

This step has already been detailed in “1) Using DB2 Cube Views, export your cube model as an XML file” on page 24. The DB2 cube model is saved into an XML file using OLAP Center in DB2 Cube Views.

2) Using MIMB, convert DB2 Cube Views into PowerDesigner

Start the MIMB software, select the import bridge labeled **IBM DB2 Cube Views** and import your model. Select the export bridge labeled **Sybase PowerDesigner PDM 7.5 to 9.5**, select the name of the export PowerDesigner PDM file and click the **Export Model** button to get the display in Figure 2-48.

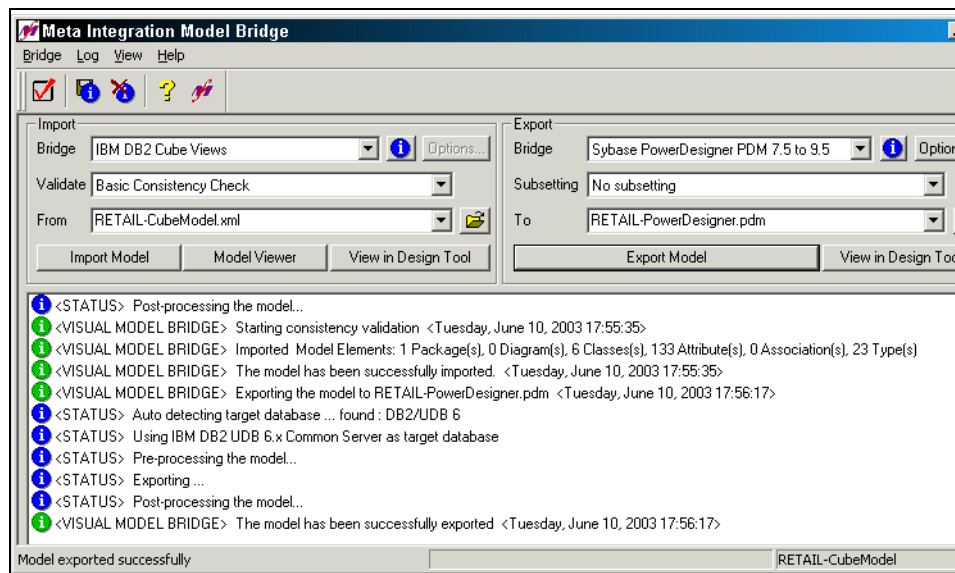


Figure 2-48 Converting the cube model XML file to an PowerDesigner XML file

3) Using PowerDesigner, import this XML file

At this point, you can open the generated PDM file into PowerDesigner using menu **File -> Open** as shown in Figure 2-49.

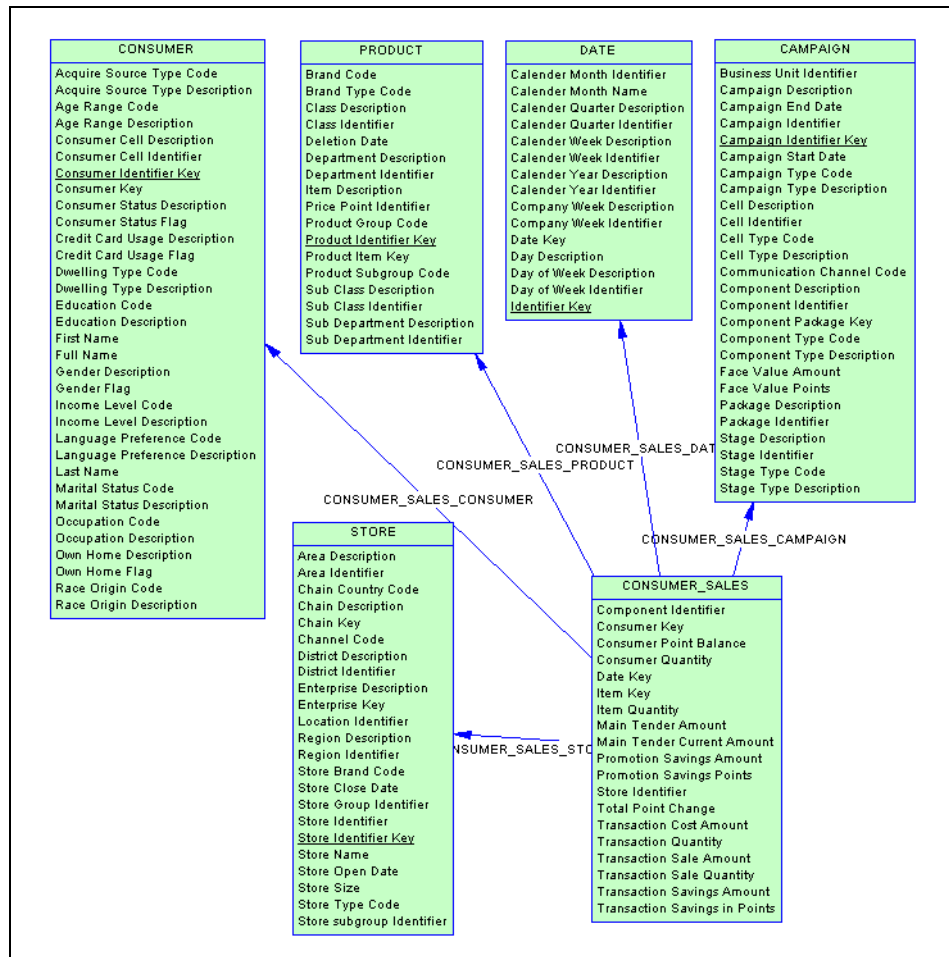


Figure 2-49 The cube model reverse engineered to PowerDesigner

The cube model converted to PowerDesigner contains the business names and descriptions as displayed in Figure 2-50.

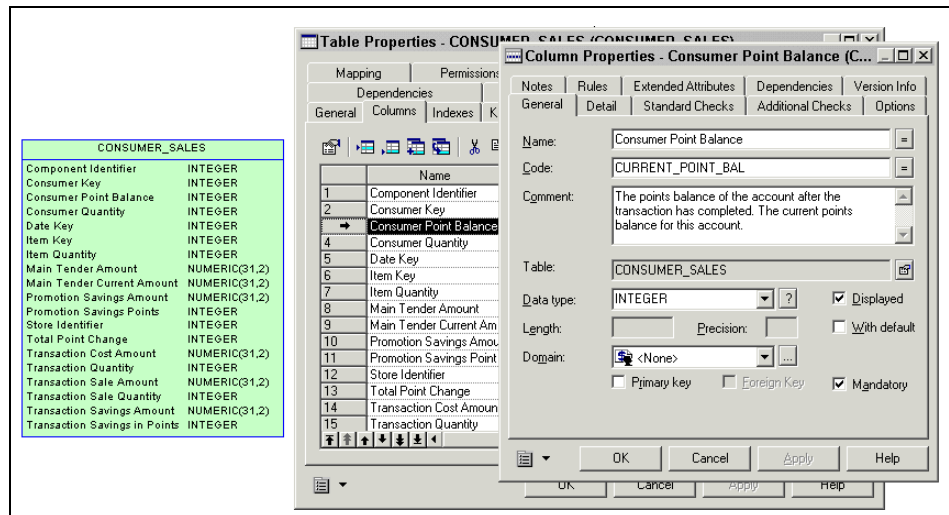


Figure 2-50 PowerDesigner business names and descriptions

Congratulations, the cube model was converted to PowerDesigner!

2.5.4 Metadata integration of DB2 Cube Views with IBM Rational Rose

IBM Rational Rose is one of the leading object modeling and data modeling tools. Rose can be used to design UML models for several target languages (C++, Java) as well as database schemas for DB2 UDB and many other database systems.

Note: The Rose MDL file format is very widely used as a de facto standard means of exchanging UML metadata. Many design tools support it and therefore this scenario can also be used to interact and exchange metadata with them.

A non-exhaustive list of such tools would include IBM Rational XDE, Microsoft Visual Studio 6 (Visual Modeler), Sybase PowerDesigner, Embarcadero Describe, Genteware Poseidon and Casewise.

Forward engineering from Rational Rose to DB2 Cube Views

The goal of this scenario is to demonstrate how an existing Rose model can be converted to a DB2 cube model.

The overall process of this metadata conversion is as follows:

1. Using Rose, create the model.
2. Using Rose, generate the SQL DDL for this database.
3. Using DB2, run this SQL script to create the tables and columns of the database.
4. Using Rose, save the model as an MDL file.
5. Using MIMB, convert this Rose MDL file into a DB2 Cube Views XML file.
6. Using DB2 Cube Views, import this DB2 Cube Views XML file.

Each step of this process is described in the following paragraphs.

1) Using Rose, create the model

The Rose model used in this scenario is composed of an object model (*UML Class Diagram*) and a data model (*Database Schema diagram*). The object model in Figure 2-51 holds the logical definition of the tables (such as business names and descriptions).

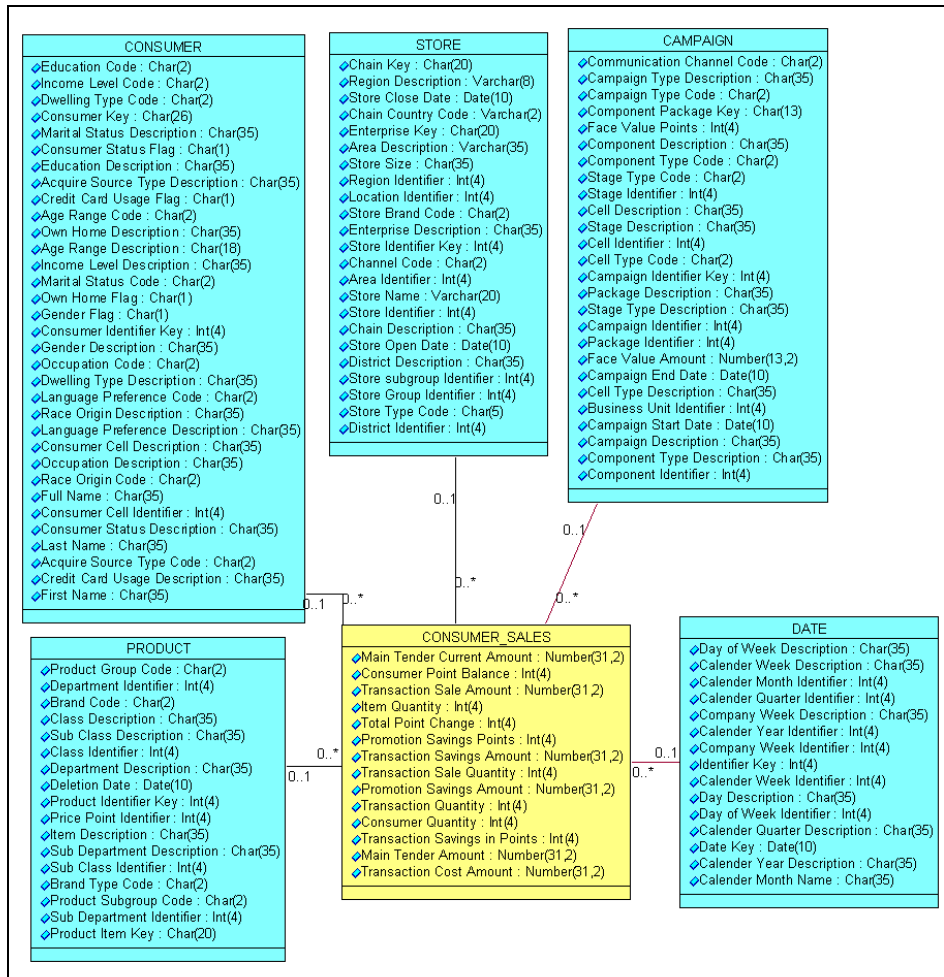


Figure 2-51 The Rose object model

The data model in Figure 2-52 holds the physical definition of the database (such as column names, primary keys, foreign keys)

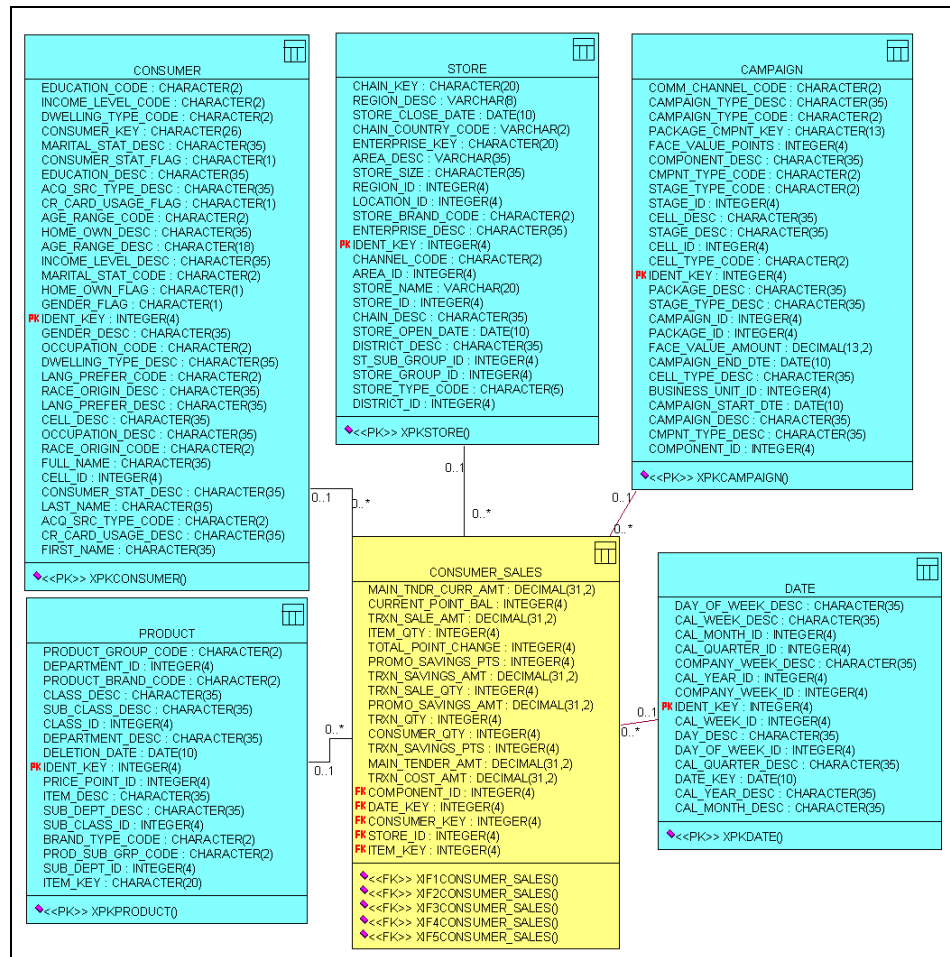


Figure 2-52 The Rose data model

To create this model in Rose, the UML object model was developed first, and was then transformed into a relational database schema, as shown in Figure 2-53, Figure 2-54, Figure 2-55, and Figure 2-56.

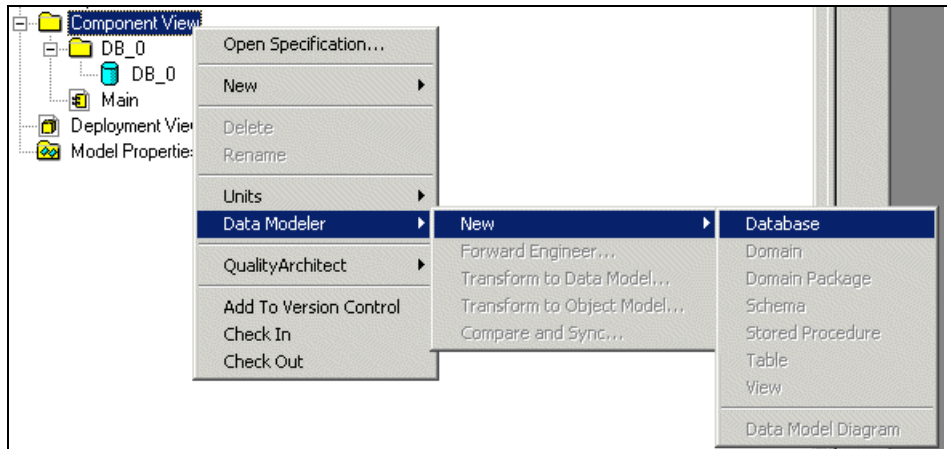


Figure 2-53 Create a new database

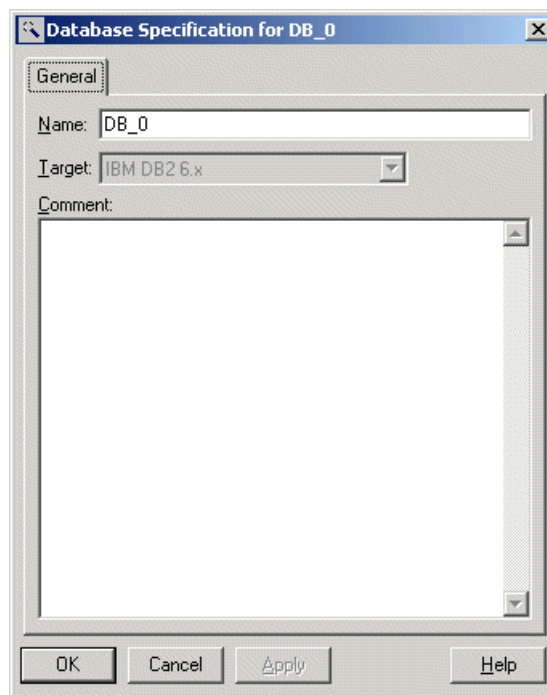


Figure 2-54 Define the database properties

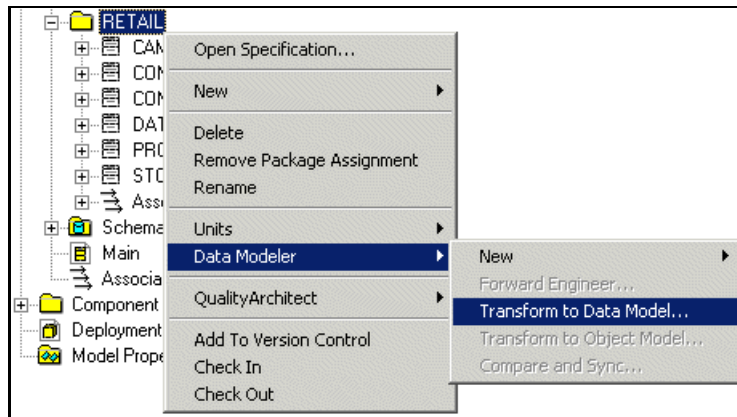


Figure 2-55 Transform to data model option

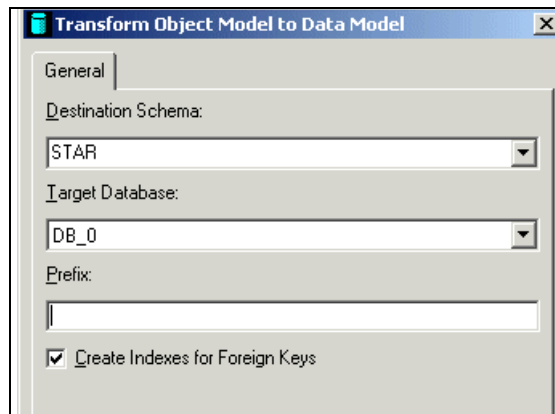


Figure 2-56 Transforming an object model into a data model in Rose

2) Using Rose, generate the SQL DDL

Once the model has been designed, the SQL DDL can be generated and the database created in DB2 UDB as shown in Figure 2-57.

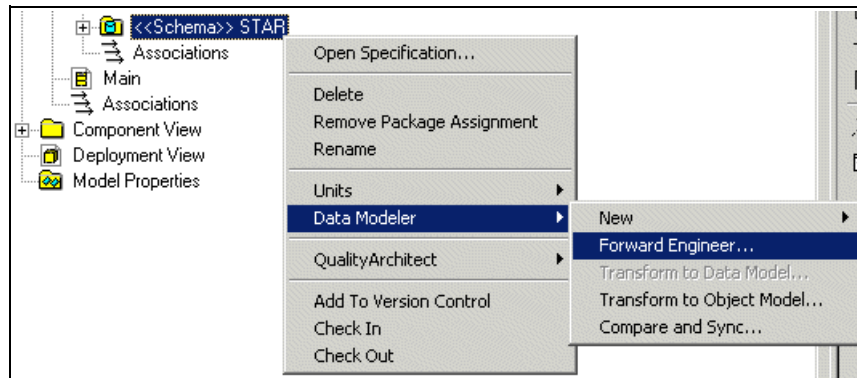


Figure 2-57 Generation of the SQL DDL in Rose

3) Using DB2, create the tables and columns of the database

This script can be executed to create the DB2 Tables using the DB2 Command Window tool (or any other tool) as described in “3) Using DB2, create the tables and columns of this schema” on page 36:

At this point, the database has been set up and it is ready to receive the cube model.

4) Using Rose, save the model as an MDL file

The next step of this process is to save the Rose model as an MDL file, the bridge will use this file as input.

When the model is loaded in Rose, choose **Save** from the **File** menu.

5) Using MIMB, convert Rose MDL into DB2 Cube Views XML

Start the MIMB tool and select the import bridge labeled **IBM Rational Rose 2000e to 2002**, and click on the **Options** button to specify the import parameters as shown in Figure 2-58.

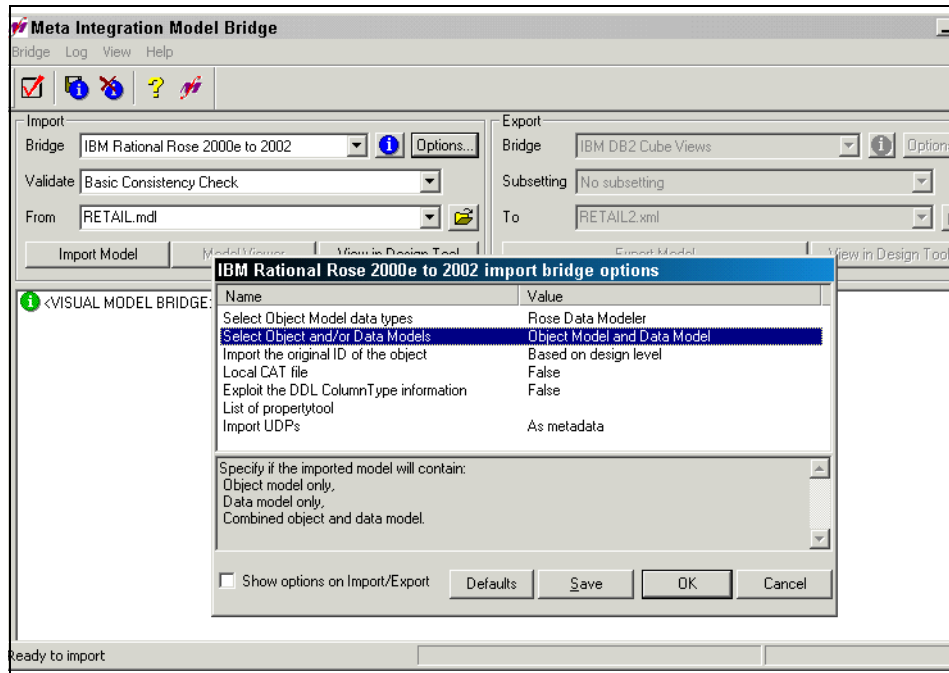


Figure 2-58 Specifying the import parameters

In this scenario, the import parameters are set as follows:

- ▶ The data types are imported as defined in the Rose data model.
- ▶ The Rose data model and its associated UML model are imported and integrated into logical and physical model.
- ▶ The other options are left with their default values.

Then, we can import the Rose MDL file, as shown in Figure 2-59.

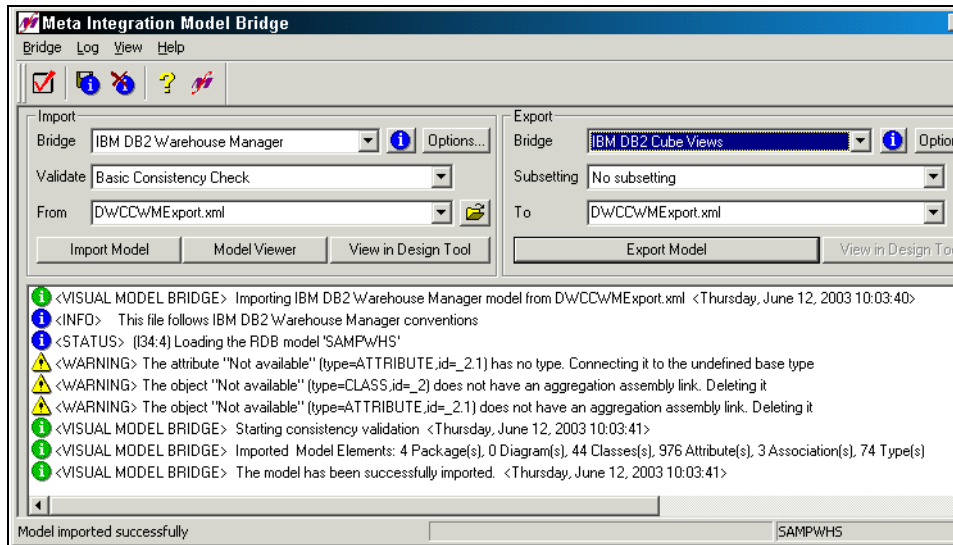


Figure 2-59 Importing the Rose model into MIMB

Select the export bridge labeled **IBM DB2 Cube Views** and click the **Options** button to specify the export parameters as shown in Figure 2-60.

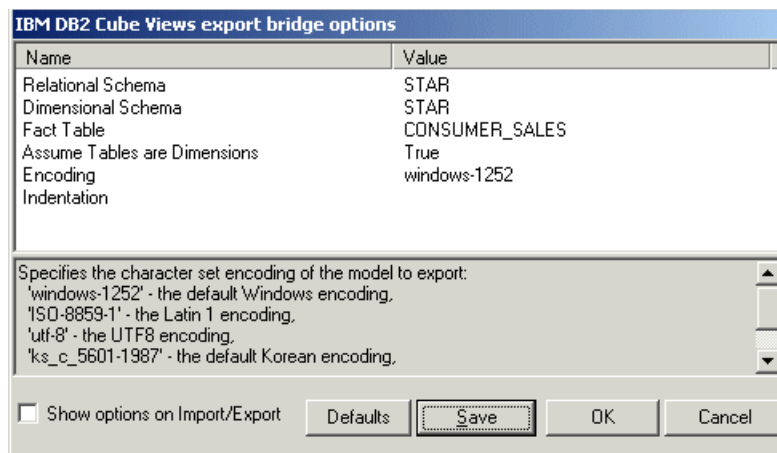


Figure 2-60 Specifying the export bridge parameters

The export parameters used in this scenario are as follows:

- ▶ The DB2 schema for the tables of the model is *STAR*, as the model may not always specify where each table is located.
- ▶ The cube model to be created will be located in the same 'STAR' DB2 schema.
- ▶ Rose Data Modeler 2002 doesn't support the notion of fact table and dimension tables yet. To work around this limitation, we can specify explicitly which table is to be considered as fact (*CONSUMER_SALES* in this case) and force the bridge to consider the other tables as dimensions.
- ▶ We should specify the encoding of the source model (locale encoding of the computer on which the Rose model was created), it is windows-1252 by default on Microsoft Windows.
- ▶ The other options are left with their default value.

Close this window, specify the name of the DB2 Cube Views XML file to be created, and click the **Export** button as shown in Figure 2-61.

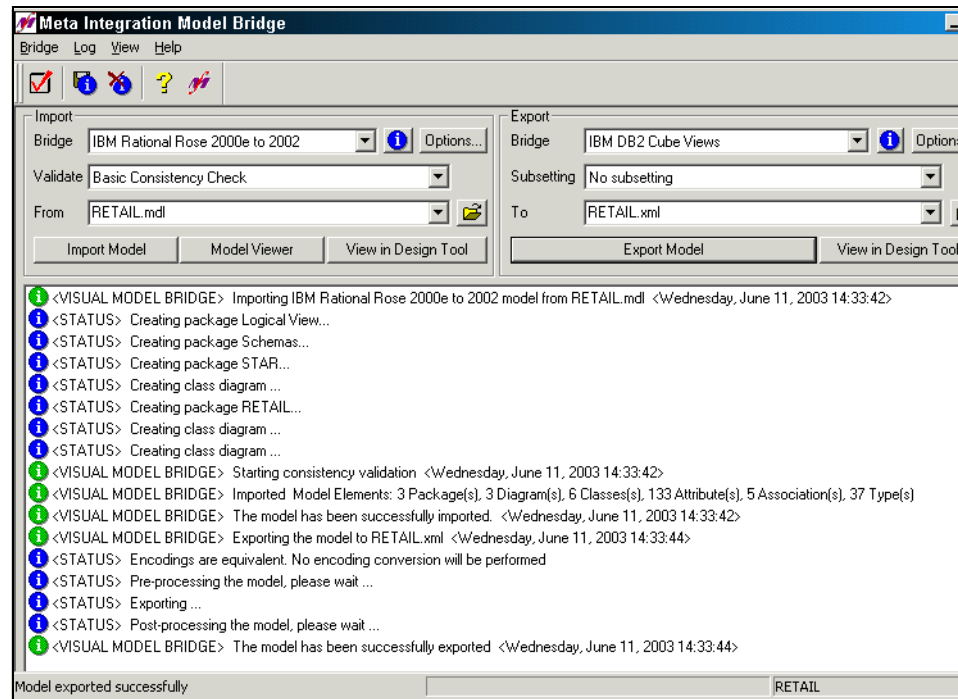


Figure 2-61 Exporting the model to DB2 Cube Views

6) Using DB2 Cube Views, import the DB2 Cube Views XML file

At this point, the cube model XML file has been created and is ready to be opened into the OLAP Center graphical tool. Just start OLAP Center, connect to your database, and choose **Import** in the OLAP Center menu to display Figure 2-62.

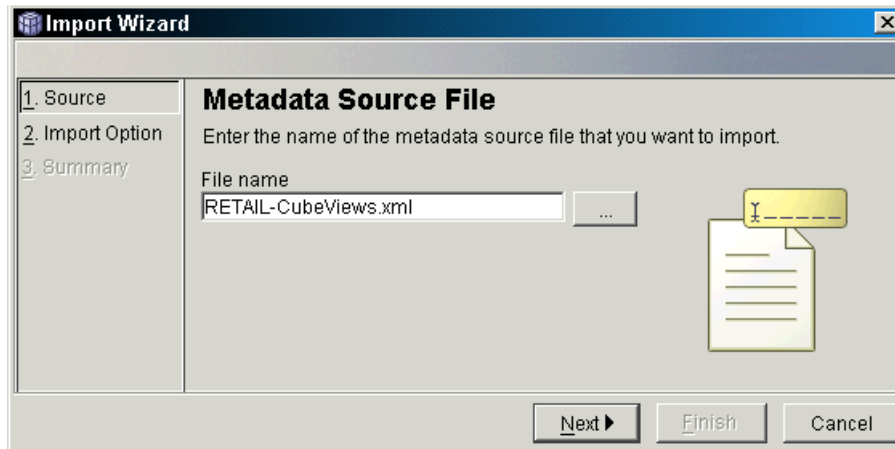


Figure 2-62 Specifying the XML file to import into OLAP Center

The content of the XML file is displayed in Figure 2-63, which allows controlling how this metadata should be imported, in case there is already some metadata in place and object name collision should occur:

- ▶ Either update the existing objects with the new imported version.
- ▶ Or keep the current version of the metadata.

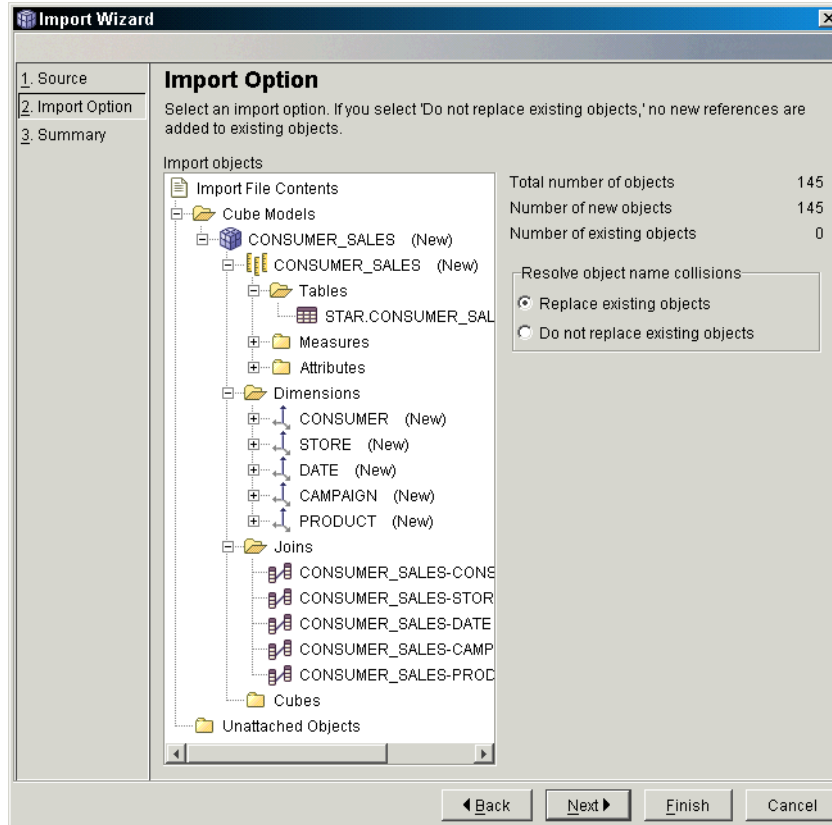


Figure 2-63 Controlling how the metadata is imported into OLAP Center

Finally, the Rose star schema metadata converted and imported into OLAP Center will provide the DB2 cube model in Figure 2-9 on page 19.

The business names and descriptions defined in Rose are also converted to the cube model, as shown in Figure 2-64.

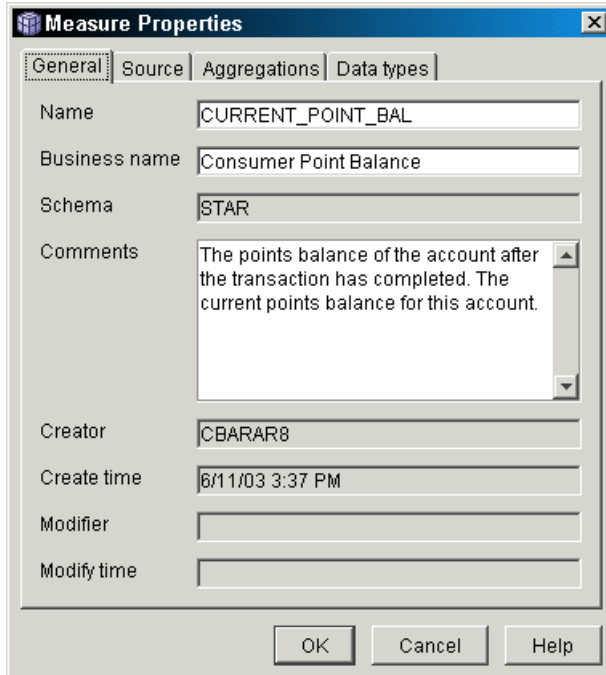


Figure 2-64 The Rose objects' business name and description are also converted

Congratulations, the Rose star schema model was converted to DB2 Cube Views!

Reverse engineering from DB2 Cube Views to Rational Rose

The goal of this scenario is to demonstrate how an existing DB2 Cube Views model can be converted into a Rose Model.

The overall process of this metadata conversion is as follows:

1. Using DB2 Cube Views, export your cube model as an XML file.
2. Using MIMB, convert this DB2 Cube Views XML file into a Rose MDL file.
3. Using Rose, import this MDL file.

Each step of this process is described in the following paragraphs.

1) Using DB2 Cube Views, export your cube model as an XML file

The DB2 Cube Views model used in this scenario is the one shown in Figure 2-9 on page 19.

This step has already been detailed in “1) Using DB2 Cube Views, export your cube model as an XML file” on page 24. The DB2 cube model is saved into an XML file using **OLAP Center > Export** in DB2 Cube Views.

2) Using MIMB, convert DB2 Cube Views XML into Rose MDL

Start the MIMB software, select the import bridge labeled **IBM DB2 Cube Views** and import your model. Select the export bridge labeled **IBM Rational Rose 2002**, select the name of the export Rose MDL file, and click the **Export Model** button as shown in Figure 2-65.

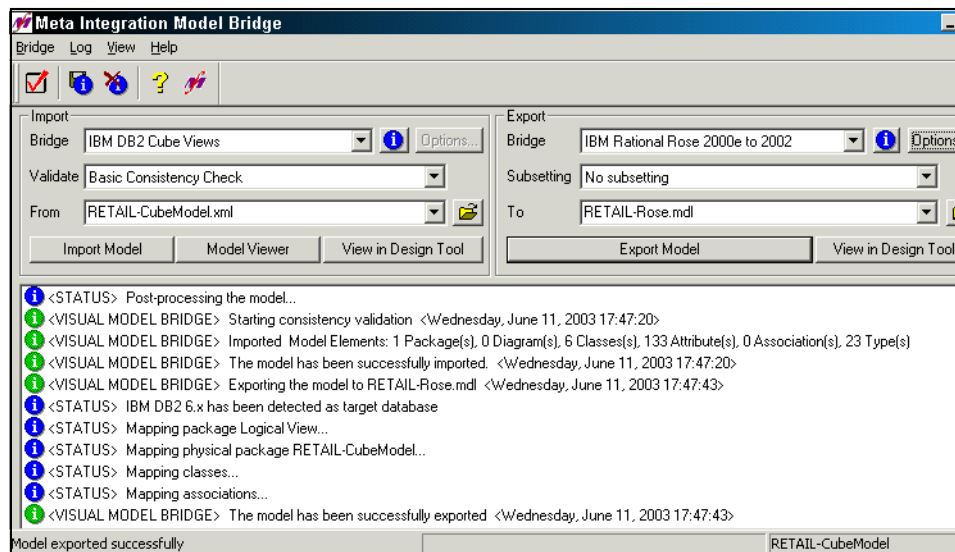


Figure 2-65 Converting the cube model XML file to a Rose MDL file

3) Using Rose, import this MDL file

At this point, you can open the generated MDL file into Rose using menu **File > Open**.

The data model after conversion is shown in Figure 2-66.

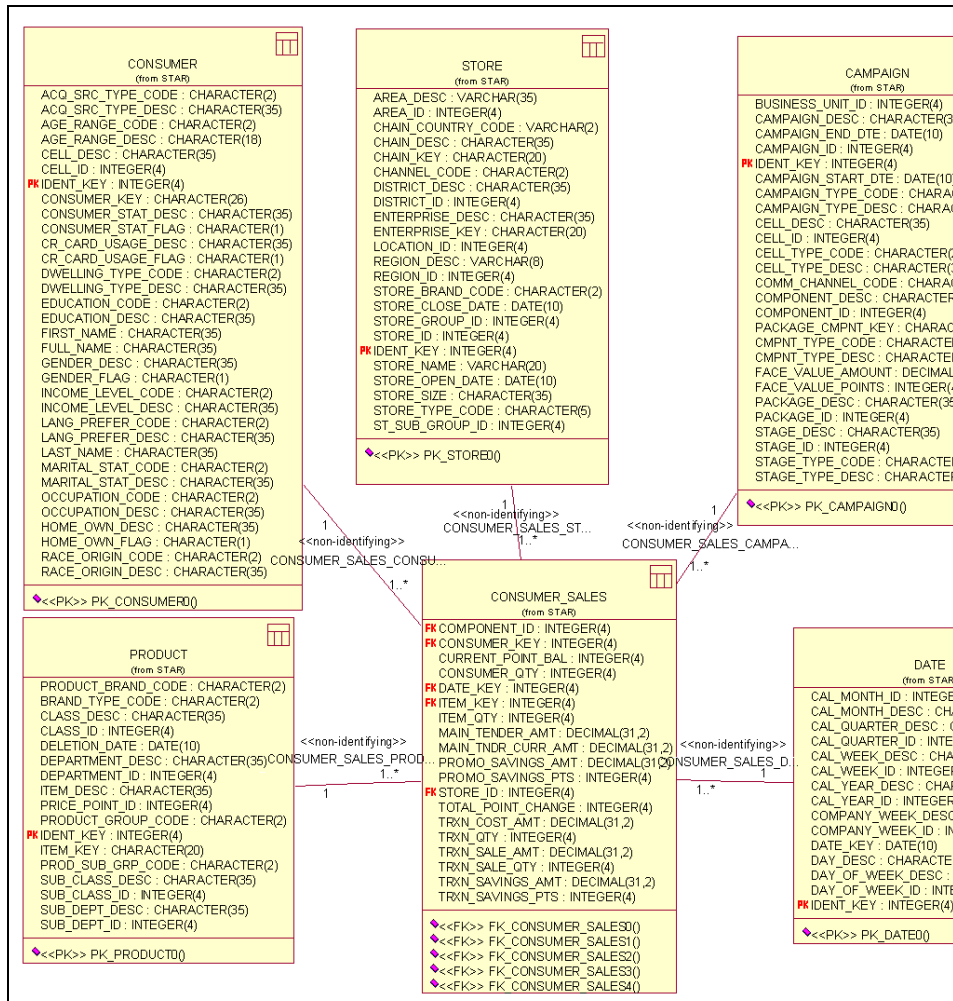


Figure 2-66 The cube model converted to Rose Data Modeler

The object model after conversion is shown in Figure 2-67.

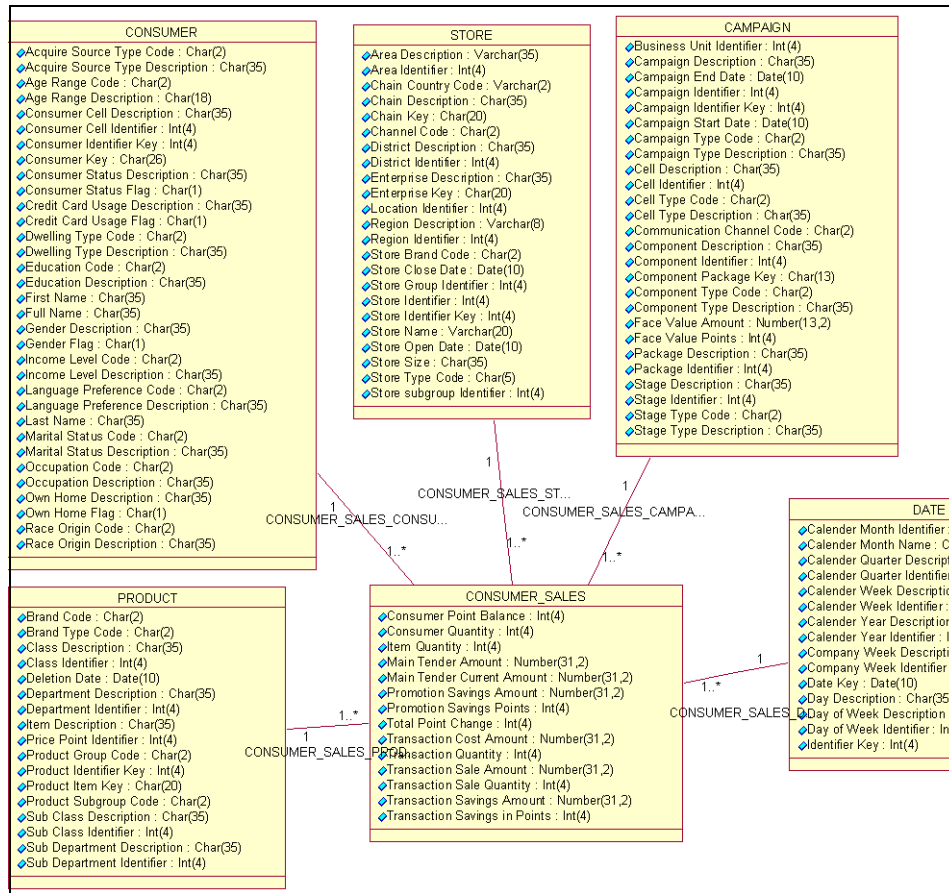


Figure 2-67 The cube model converted to Rose Object Modeler

Congratulations, the cube model was converted to Rose!

2.5.5 Metadata integration of DB2 Cube Views with CWM and XMI

The Object Management Group (OMG) Common Warehouse Metamodel (CWM) is an industry standard metamodel supported by numerous leading data and metadata management tools vendors. The CWM metamodel shown in Figure 2-68 is defined as an instance of the Meta Object Facility (MOF) meta-metamodel and expressed using the OMG Unified Modeling Language (UML) in terms of classes, relationships, diagrams and packages. Any model instance of the UML and CWM metamodel can also be serialized into an XML document using the XML Metadata Interchange (XMI) facility.

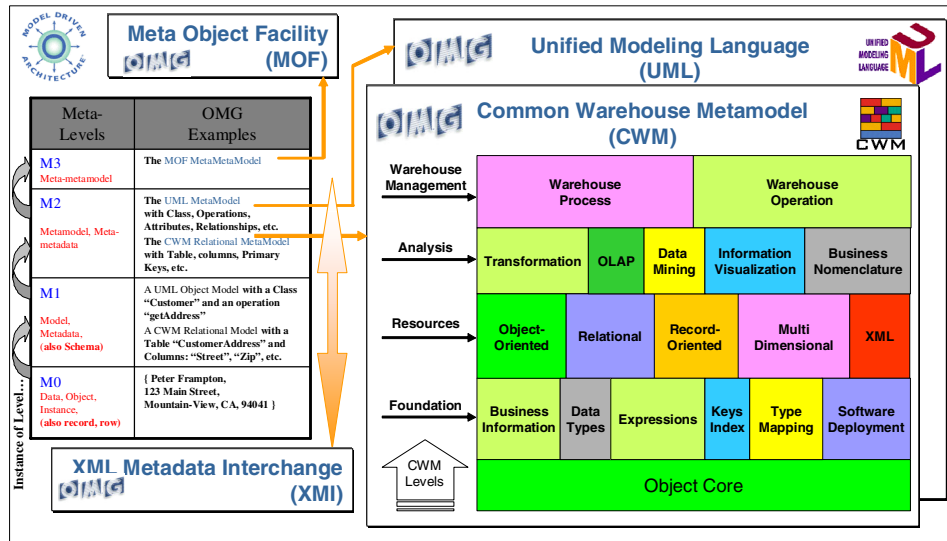


Figure 2-68 The OMG standards

Meta Integration Technology, Inc. (MITI) has been a strong supporter of the OMG CWM standard since 1999 and joined the OMG in 2000 as an influencing member. Since 2001, MITI became a domain member of the OMG focusing on XMI based metadata interchange. MITI is mostly working on the implementation and support of the CWM standard with other key OMG members such as Adaptive, Hyperion, IBM, Oracle, SAS, and Unisys. MITI is also actively participating to all OMG enablement showcases demonstrating bi-directional metadata bridges with many design tools, ETL tools and BI tools.

The April 28 - May 2, 2002 CWM Enablement Showcase at the Annual Meta Data Conference / DAMA Symposium, San Antonio, Texas is shown in Figure 2-69.

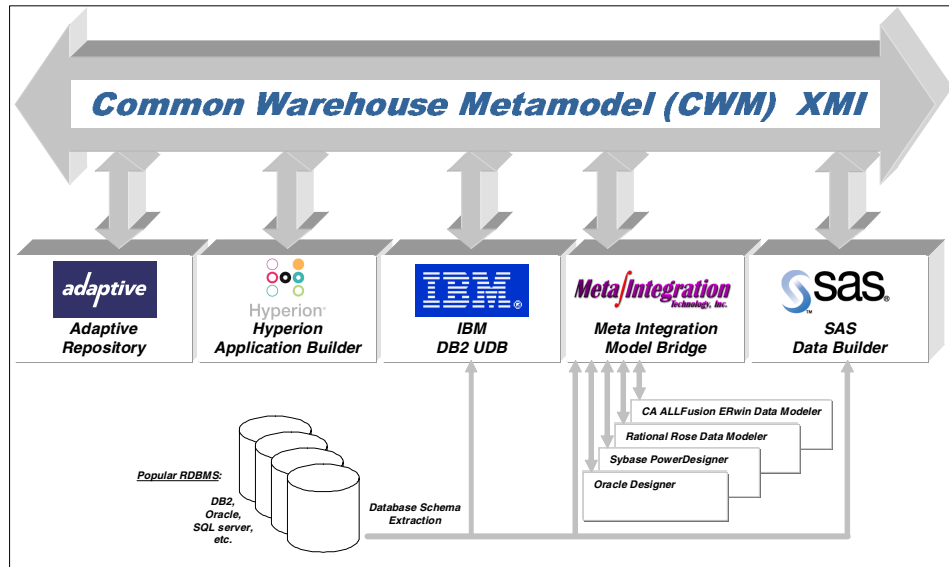


Figure 2-69 CWM Enablement Showcase

The metadata interchange and integration challenges using the CWM and XMI standards are due to multiple factors:

- ▶ The UML, CWM and XMI standards are evolving and each of them has multiple versions. A given instance metadata document is therefore a combination of versions of each of these standards.
- ▶ A testing suite or open source reference implementation is not yet available.
- ▶ Software vendors implementing import/export capabilities often need to extend the standard by using additional properties (TaggedValues), additional metadata resources (the CWMX extension packages) leading to specific CWM dialects.

Nevertheless, CWM XMI is the leading standard in metadata interchange and MITI has implemented a comprehensive support for it, including support for various versions of the metamodel and XMI encoding, and also many specific tool vendors' features. For more information, please read the following pages online:

<http://www.metaintegration.net/Partners/OMG.html>

<http://www.metaintegration.net/Products/MIMB/Documentation/OMGXMI.html>

<http://www.metaintegration.net/Products/MIMB/SupportedTools.html>

This scenario demonstrates how to export DB2 Cube Views metadata in the OMG CWM XMI format.

The next scenario will show how to import OMG CWM XML metadata generated by DB2 Warehouse Manager into DB2 Cube Views.

The overall process of this metadata conversion is as follows:

1. Using DB2 Cube Views, create a cube model and save it in an XML file.
2. Using MIMB, convert this XML file into a CWM XML file.

Each step of this process is described in the following paragraphs.

1) Using DB2 Cube Views, create a cube model and export it in XML

The DB2 Cube Views model used in this scenario is the one shown in Figure 2-9 on page 19.

This step has already been detailed in “1) Using DB2 Cube Views, export your cube model as an XML file” on page 24. The DB2 cube model is saved into an XML file using **OLAP Center > Export** in DB2 Cube Views.

2) Using MIMB, convert this XML file into a CWM XML file

Start the MIMB software, select the import bridge labeled **IBM DB2 Cube Views** and import the cube model XML file as shown in Figure 2-70.

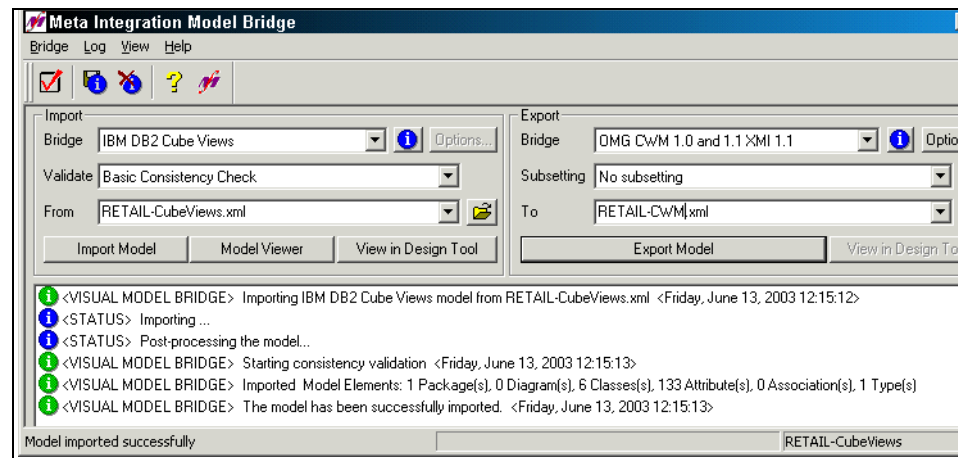


Figure 2-70 MIMB: Importing the cube model XML file

Select the export bridge labeled **OMG CWM 1.0 and 1.1 XMI 1.1** and type the name of the export file in the **To** field. Click on the **Options** button to specify the export options.

The CWM export bridge has many parameters, which allow controlling how the CWM file should be created.

For example, in MIMB version 3.x, the export bridge **Model** option allows you to choose how the cube model's metadata should be mapped:

- ▶ As a relational model instance of the CWM:RDB package to represent the star-schema database information.
- ▶ As a UML software model instance of the CWM:ObjectModel package to allow software developers to import it in a UML-enabled design tool and develop their application, taking advantage of all the business names and description defined in DB2 Cube Views.
- ▶ Or, both of these possibilities.

Note: To follow up on enhancements for the export bridge with OLAP metadata, please check the following site regularly:

<http://www.metintegration.net/Products/MIMB>

The model option is shown in Figure 2-71.

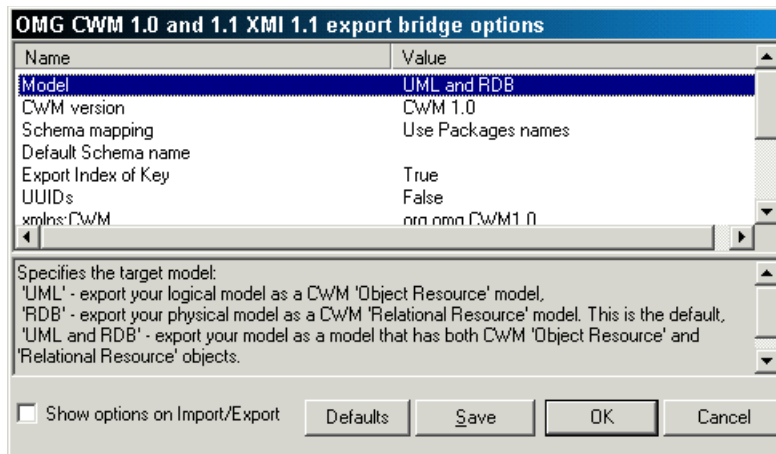


Figure 2-71 Specifying the export options: model

We also specify that the source encoding of the cube model XML file is utf-8 as shown in Figure 2-72.

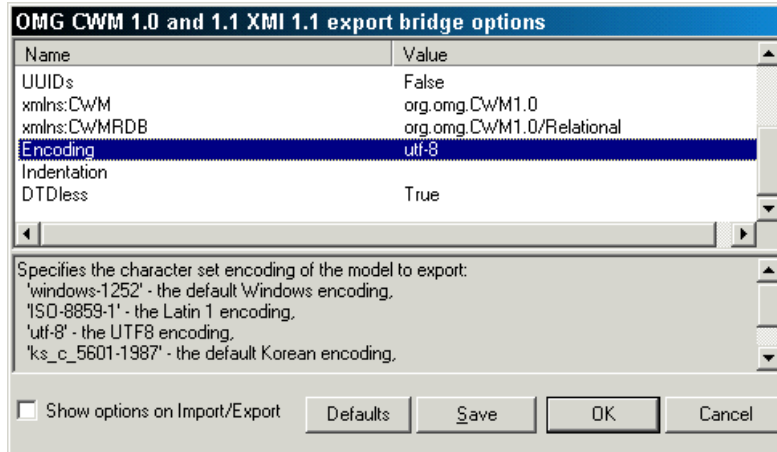


Figure 2-72 Specifying the export options: encoding

Finally, click on the **Export Model** button as shown in Figure 2-73 to create the CWM XMI file.

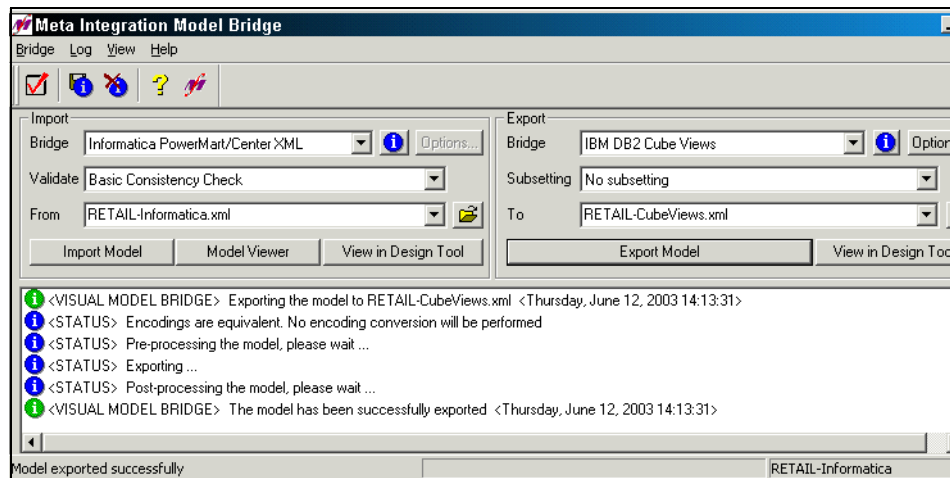


Figure 2-73 MIMB: exporting to CWM

Figure 2-74 is a sample of the exported CWM file.

```

D:\Redbook\MITI\Redpaper\scenarios\RETAIL-CWM-reverse\RETAIL-CWM.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help Links
<?xml version="1.0" encoding="UTF-8" ?>
<!-- <!DOCTYPE XMI SYSTEM "cwm.dtd" -->
-->
- <XMI xmi.version="1.1" timestamp="Jun 13 2003 13:21:26" xmlns:CWM="org.omg.CWM1.0"
  xmlns:CWMRDB="org.omg.CWM1.0/Relational">
- <XMI.header>
- <XMI.documentation>
  <XMI.exporter>Meta Integration Model Bridge</XMI.exporter>
  <XMI.exporterVersion>3.1.1 - May 29 2003 10:47:56</XMI.exporterVersion>
</XMI.documentation>
  <XMI.metamodel xmi.name="CWM" xmi.version="1.0" />
</XMI.header>
- <XMI.content>
- <CWM:Model xmi.id="_1" name="RETAIL-CubeViews" visibility="public">
- <CWM:Namespace.ownedElement>
+ <CWM:Class xmi.id="_2" name="CAMPAIGN" visibility="public" namespace="_1">
+ <CWM:Class xmi.id="_31" name="CONSUMER" visibility="public" namespace="_1">
+ <CWM:Class xmi.id="_66" name="CONSUMER_SALES" visibility="public" namespace="_1">
+ <CWM:Class xmi.id="_100" name="DATE" visibility="public" namespace="_1">
+ <CWM:Class xmi.id="_117" name="PRODUCT" visibility="public" namespace="_1">
+ <CWM:Class xmi.id="_136" name="STORE" visibility="public" namespace="_1">
  <CWM:DataType xmi.id="_5" name="undefined" isAbstract="false" visibility="public" />
</CWM:Namespace.ownedElement>
</CWM:Model>
- <CWMRDB:Catalog xmi.id="_161" name="RETAIL-CubeViews" visibility="public">
- <CWM:Namespace.ownedElement>
- <CWMRDB:Schema xmi.id="_162" name="STAR" visibility="public" namespace="_161">
- <CWM:Namespace.ownedElement>
+ <CWMRDB:Table xmi.id="_163" name="CAMPAIGN" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">
+ <CWMRDB:Table xmi.id="_191" name="CONSUMER" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">
+ <CWMRDB:Table xmi.id="_225" name="CONSUMER_SALES" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">
+ <CWMRDB:Table xmi.id="_258" name="DATE" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">
+ <CWMRDB:Table xmi.id="_274" name="PRODUCT" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">
+ <CWMRDB:Table xmi.id="_292" name="STORE" isSystem="false"
  isTemporary="false" visibility="public" namespace="_162">

```

Figure 2-74 Sample CWM XMI file reverse engineered from a cube model

2.5.6 Metadata integration of DB2 Cube Views with DB2 Warehouse Manager

DB2 Warehouse Manager provides ETL and warehouse management functionalities to the DB2 platform. DB2 Warehouse Manager can be used to design a data warehouse or data mart, manage the different data sources populating it, and design the complex flow of data transformation between the source databases and the target data warehouse in an intuitive, GUI oriented way. The main user interface of this software is the DB2 Data Warehouse Center tool, and it supports the import and export of metadata via the OMG CWM XMI file format.

This scenario focuses on the exchange of metadata between DB2 Warehouse Center and DB2 Cube Views via the OMG CWM XMI file format. We will demonstrate how a datamart designed in DB2 Warehouse Center in the form of a star schema can be saved as a CWM XMI file, then converted to a DB2 Cube Views XML file using the MIMB utility, and finally, open it in DB2 Cube Views as a cube model.

The overall process of this metadata conversion is as follows:

1. Using DB2 Warehouse Center, create the star schema model.
2. Using DB2 Warehouse Center, save the model as a CWM XMI file.
3. Using MIMB, convert this CWM XMI file into a DB2 Cube Views XML file.
4. Using DB2 Cube Views, import this DB2 Cube Views XML file.

Each step of this process is described in the following paragraphs.

1) Using DB2 Data Warehouse Center, create the star schema model

This scenario uses the small “Beverage Company” star schema model shown in Figure 2-75.

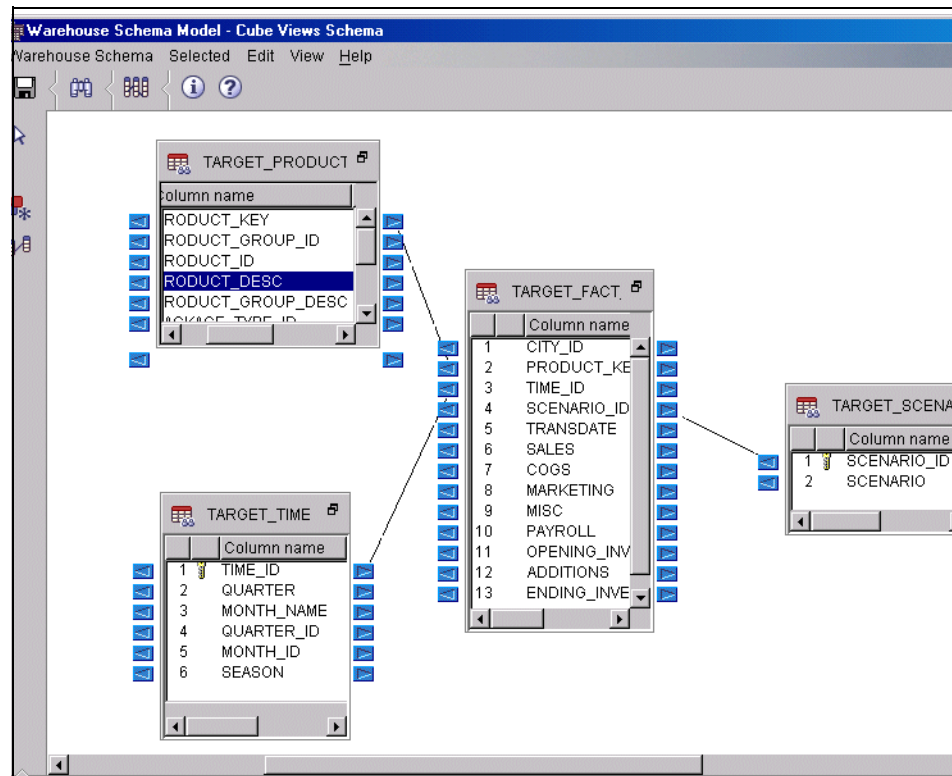


Figure 2-75 The Beverage company model in Data Warehouse Center

During the design of this model, a property has been set on each table to specify its role in the datamart (Fact or Dimension) as shown in Figure 2-76.

The screenshot shows a window titled "Properties - TARGET_FACT_TABLE" with a sub-header "TBC Sample Targets - TARGET_FACT_TABLE". It has four tabs: "Target Table", "Columns", "Warehouse Primary Key", and "Warehouse Foreign Keys". The "Target Table" tab is active. Fields include: "Table schema" (DWCTBC), "Table name" (TARGET_FACT_TABLE), "Table space" (empty), "Index table space" (empty), and "Description" (TBC Fact Table with sales, production costs, and inventory measures.). A section titled "Data Warehouse Center options" contains: "Business name" (empty), "Data Warehouse Center created table" (checked), "Part of an OLAP schema" (checked), "Transient data" (unchecked), "Dimension table" (radio button), "Grant to public" (checked), "Fact table" (radio button, selected), "Number of editions" (0), and "Edition column" (<Select>).

Figure 2-76 This is the fact table of the star schema

2) Using Data Warehouse Center, save the model as a CWM XMI file

From the DB2 Data Warehouse Center, export the metadata as shown in Figure 2-77.

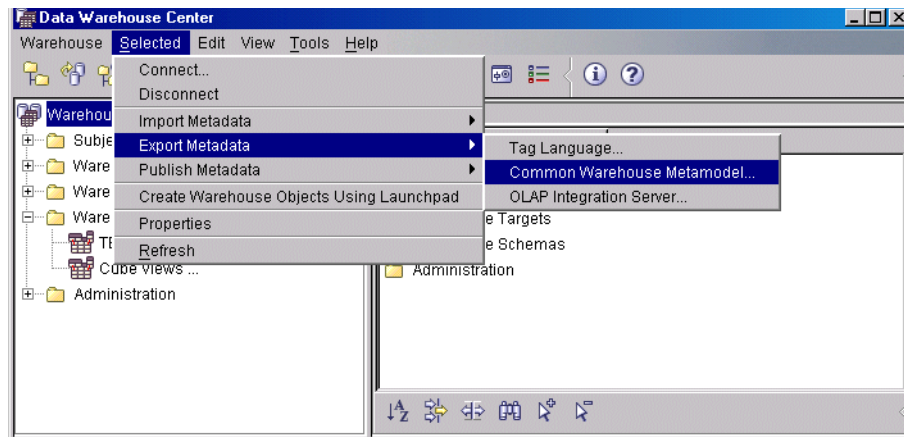


Figure 2-77 Starting the CWM export wizard from DB2 Data Warehouse Center

Then select the database to be exported as shown in Figure 2-78.

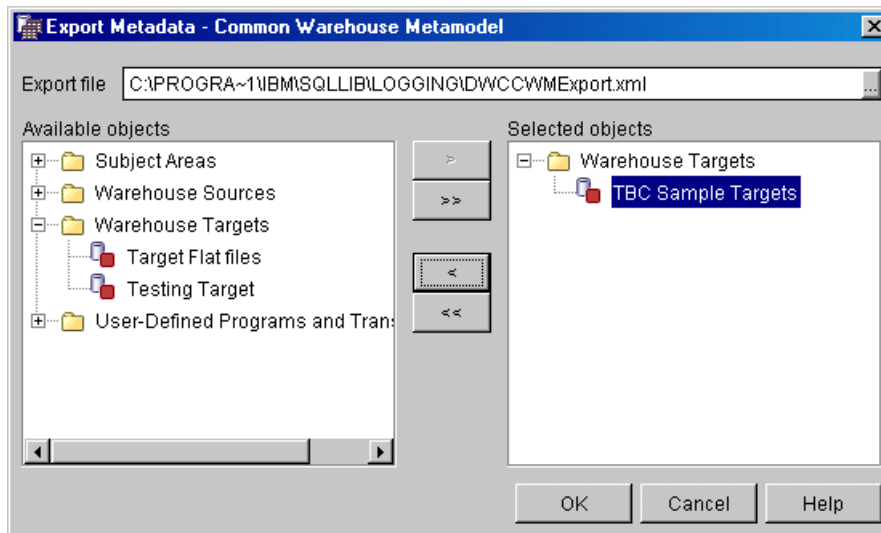


Figure 2-78 Selecting the database to be exported to CWM

Figure 2-79 is a sample of the exported CWM file.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <XMI xmi.version="1.1" timestamp="Wed Jun 11 12:22:17 MST 2003" xmlns:CWM="org.omg.CWM1.0"
  xmlns:CWMRDB="org.omg.CWM1.0/Relational">
- <XMI.header>
  - <XMI.documentation>
    <XMI.exporter>XMI Application Framework</XMI.exporter>
    <XMI.exporterVersion>1.15</XMI.exporterVersion>
  </XMI.documentation>
</XMI.header>
- <XMI.content>
+ <CWM:SoftwareSystem xmi.id="_1" name="DWC">
+ <CWM:Class xmi.id="_2" name="Not available">
  <CWM:DataType xmi.id="_3" name="1" />
  <CWM:DataType xmi.id="_4" name="2" />
  <CWM:DataType xmi.id="_5" name="3" />
- <CWMRDB:Catalog xmi.id="_6" name="SAMPWHS">
+ <CWM:ModelElement.taggedValue>
- <CWM:Namespace.ownedElement>
+ <CWMRDB:Schema xmi.id="_6.10" name="ADMINISTRATOR" namespace="_6">
+ <CWMRDB:Schema xmi.id="_6.11" name="C360" namespace="_6">
- <CWMRDB:Schema xmi.id="_6.12" name="DWCTBC" namespace="_6">
  - <CWM:Namespace.ownedElement>
    - <CWMRDB:Table xmi.id="_6.12.1" name="TARGET_FACT_TABLE" namespace="_6.12">
      + <CWM:ModelElement.taggedValue>
        - <CWM:Classifier.feature>
          + <CWMRDB:Column xmi.id="_6.12.1.16" name="CITY_ID" length="0" scale="0"
            precision="0" isNullable="columnNoNulls" owner="_6.12.1" type="_25">
          + <CWMRDB:Column xmi.id="_6.12.1.17" name="PRODUCT_KEY" length="0"
            scale="0" precision="0" isNullable="columnNoNulls" owner="_6.12.1"
            type="_25">
          + <CWMRDB:Column xmi.id="_6.12.1.18" name="TIME_ID" length="0" scale="0"
            precision="0" isNullable="columnNoNulls" owner="_6.12.1" type="_25">
          - <CWMRDB:Column xmi.id="_6.12.1.19" name="SCENARIO_ID" length="0"
            precision="0" isNullable="columnNoNulls" owner="_6.12.1" type="_25">
```

Figure 2-79 The CWM XMI file rendered in a browser

3) Using MIMB, convert CWM XMI file into DB2 Cube Views XML file

At this point, start MIMB and select the **IBM DB2 Warehouse Manager** import bridge. This bridge is designed to understand the DB2 Warehouse Manager dialect of CWM. Then, import the CWM XMI file as shown in Figure 2-80.

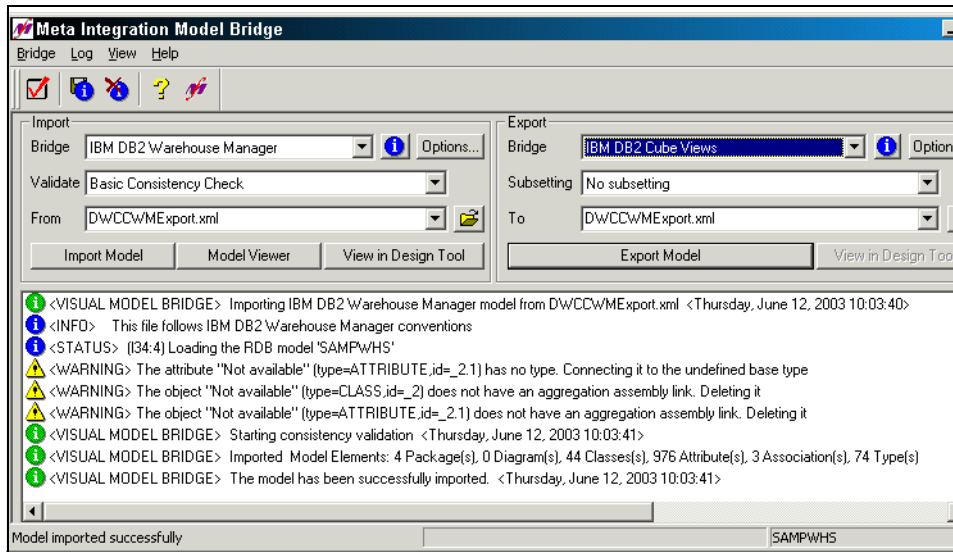


Figure 2-80 MIMB: importing the DB2 Data Warehouse Center CWM XML file

Click the button labeled **Model Viewer** to review the imported metadata in Figure 2-81.

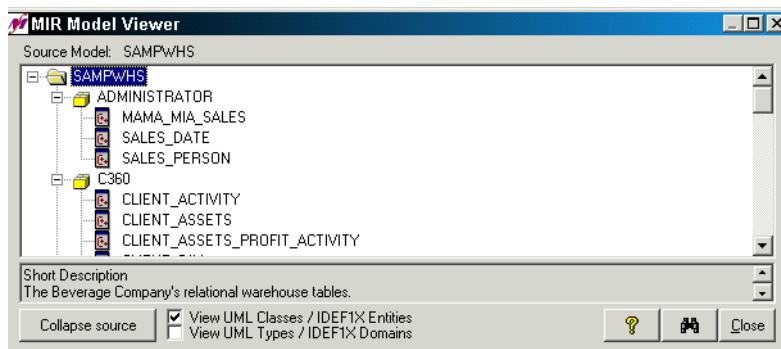


Figure 2-81 The sample warehouse Beverage Company imported from CWM

Select the export bridge labeled **IBM DB2 Cube Views** and click the **Options** button to specify the export parameters as shown in Figure 2-82.

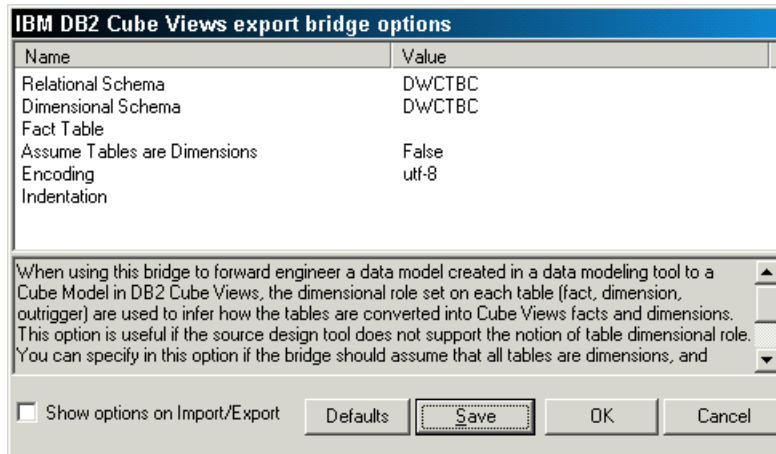


Figure 2-82 Specifying the export parameters

In this scenario, the star schema tables are located in a DB2 schema called *DWCTBC*, which we also use to store the OLAP metadata. We specify that the source encoding of the CWM file is utf-8. The other parameters are left to their default value.

We need to select the 4 tables to be exported as a cube model. As we have seen in the MIMB Model Viewer, there are many more tables in this warehouse, but we only need these 4 tables to be exported to the cube model.

Select the model subsetting option labeled **Specific Classes** as shown in Figure 2-83.

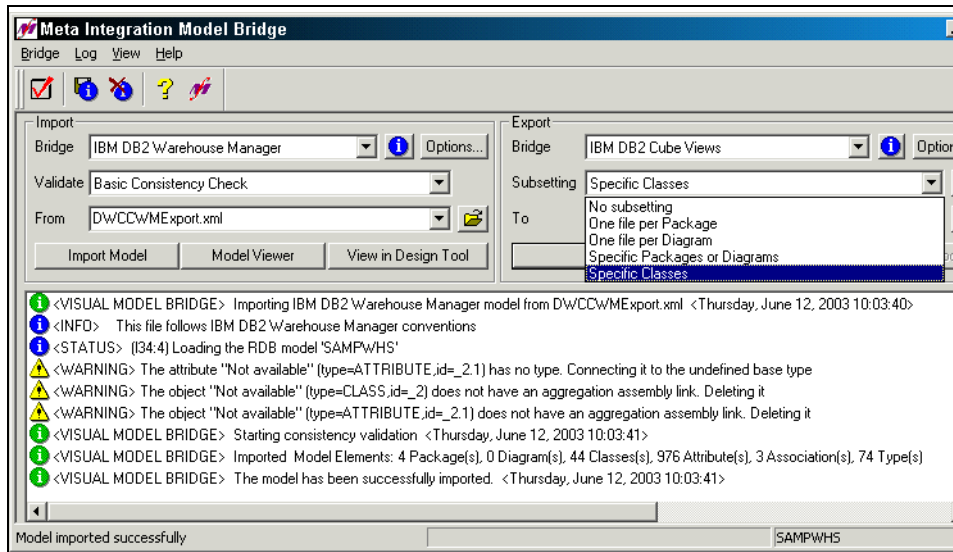


Figure 2-83 Choosing a subsetting mode

Drag and drop the 4 tables to be subsetting and click on button **Subset selected class(es)** as shown in Figure 2-84.

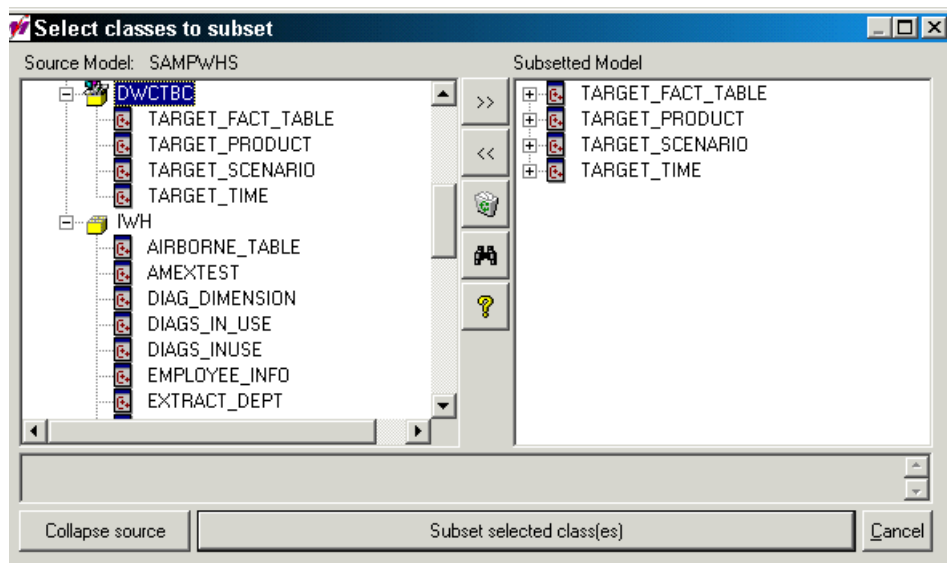


Figure 2-84 Subsetting the star schema model

Set the name of the final cube model XML file to be produced and click on button **Export elements** as shown in Figure 2-85.

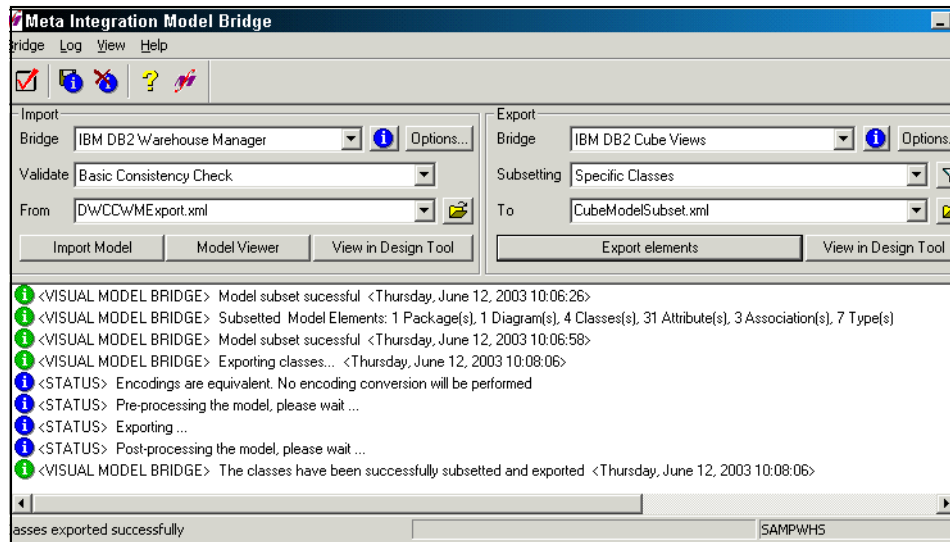


Figure 2-85 Exporting the cube model

The cube model has now been produced and is ready to be imported into DB2 Cube Views.

4) Using DB2 Cube Views, import this Cube Views XML file

Finally, the metadata is imported into OLAP Center as shown in Figure 2-86.

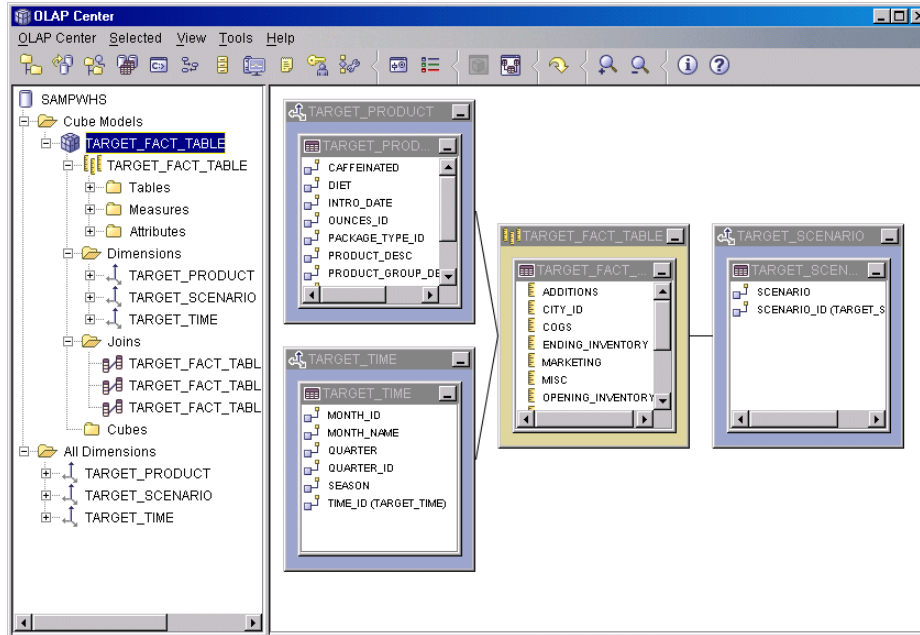


Figure 2-86 The Beverage Company star schema imported into DB2 Cube Views

Congratulations, you have imported into DB2 Cube Views a star schema designed in DB2 Warehouse Manager!

2.5.7 Metadata integration of DB2 Cube Views with Informatica

Informatica is one of the leading ETL tool vendors with tools such as PowerMart and PowerCenter that you can use to populate a DB2 data warehouse. The complex flow of data and transformations can be designed using PowerMart Designer. You can use Informatica PowerMart Designer to import and export of metadata via an XML file format.

This scenario demonstrates how to transform the metadata of a data warehouse designed in PowerMart 5.x and 6.x in the form of a star schema into a DB2 Cube Views cube model.

The overall process of this metadata conversion is as follows:

1. Using Informatica PowerCenter, save the metadata of the target data mart as XML.
2. Using MIMB, convert this Informatica XML file into a DB2 Cube Views XML file.

3. Using DB2 OLAP Center, import this DB2 Cube Views XML file.

Each step of this process is described in the following paragraphs.

1) Using PowerCenter, save target datamart metadata as XML

Using the Informatica PowerMart Designer tool, export definitions of target tables into an XML file as shown in Figure 2-87. Please refer to the Informatica documentation for details.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- This file was generated by Informatica XML Export Model Bridge from Meta Integration
Technology, Inc. (MITI) -->
<POWERMART>
- <REPOSITORY NAME="Generated by MIMB">
- <FOLDER NAME="Generated by MIMB">
- <FOLDERVERSION NAME="Generated by MIMB">
+ <TARGET NAME="CAMPAIGN" BUSINESSNAME="CAMPAIGN" DESCRIPTION="A promotional
CAMPAIGN is a limited period of time when some PRODUCTS are subjects to specific
rebates. The promotional campaign is targetted to CONSUMERS via multiple
communication channels and CONSUMERS purchasing PRODUCTS in our STOREs
during this period can earn savings points." DATABASETTYPE="DB2/UDB 6">
+ <TARGET NAME="CONSUMER" BUSINESSNAME="CONSUMER" DESCRIPTION="Holds the
complete description of a CONSUMER purchasing PRODUCTS in our STOREs."
DATABASETTYPE="DB2/UDB 6">
+ <TARGET NAME="CONSUMER_SALES" BUSINESSNAME="CONSUMER_SALES"
DESCRIPTION="This fact table holds the measures of the model representing the sales
amount generated by CONSUMERS who have purchased PRODUCTS in STOREs during
a promotional CAMPAIGN." DATABASETTYPE="DB2/UDB 6">
+ <TARGET NAME="DATE" BUSINESSNAME="DATE" DESCRIPTION="This is the TIME dimension
of the star schema" DATABASETTYPE="DB2/UDB 6">
+ <TARGET NAME="PRODUCT" BUSINESSNAME="PRODUCT" DESCRIPTION="A PRODUCT can be
purchased by a CONSUMER in a STORE at a particular DATE A PRODUCT can also be
the subject of a time limited promotional CAMPAIGN" DATABASETTYPE="DB2/UDB 6">
+ <TARGET NAME="STORE" BUSINESSNAME="STORE" DESCRIPTION="This STORE sells
PRODUCTs to CONSUMERS" DATABASETTYPE="DB2/UDB 6">
</FOLDERVERSION>
</FOLDER>
</REPOSITORY>
</POWERMART>
```

Figure 2-87 The sample Informatica XML model

Note: A copy of the Informatica software was not available during the writing of this chapter, so the XML file shown here was not directly generated by Informatica, but instead was forward engineered from an ERwin model to Informatica using the MIMB software. Nevertheless, the principles of this scenario are still relevant and the conversion process is the same.

2) Using MIMB, convert Informatica XML into DB2 Cube Views XML

Start the MIMB software, select the import bridge labeled **Informatica PowerMart/Center XML**, select the XML file to be imported, and click on the Import Model to import it as shown in Figure 2-88.

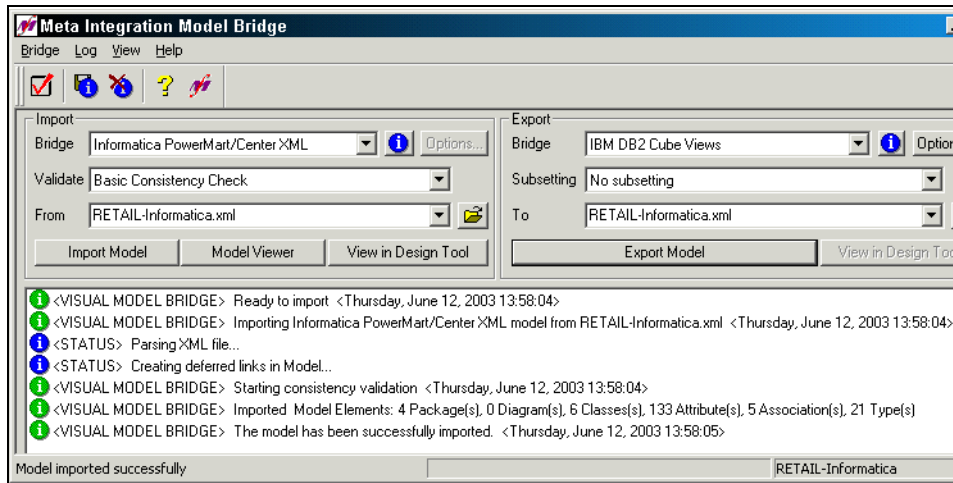


Figure 2-88 Importing the Informatica model

If the Informatica XML file contains the definition of tables that are not part of the target star schema, you can filter them out using the MIMB subsetting feature. Please refer to the MIMB documentation for details.

Then, select the export bridge labeled **IBM DB2 Cube Views** and press on the **Options** button to specify the export parameters as shown in Figure 2-89.

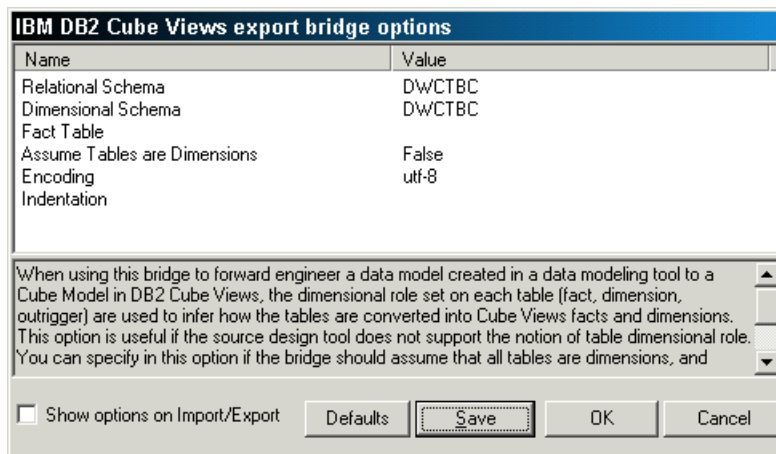


Figure 2-89 Specifying the export parameters

In this scenario, the tables are located in schema *STAR* and we will also create the OLAP objects in this schema. We have also indicated the name of the fact table and set the bridge to consider that the other tables are to be processed as dimensions.

Note: The fact or dimension information on each table may not always be specified in the Informatica XML file. In this case, we can specify it this way.

We also specify that the source encoding of the Informatica model is utf-8.

At this point, you can export the model to the DB2 Cube Views XML file format as shown in Figure 2-90.

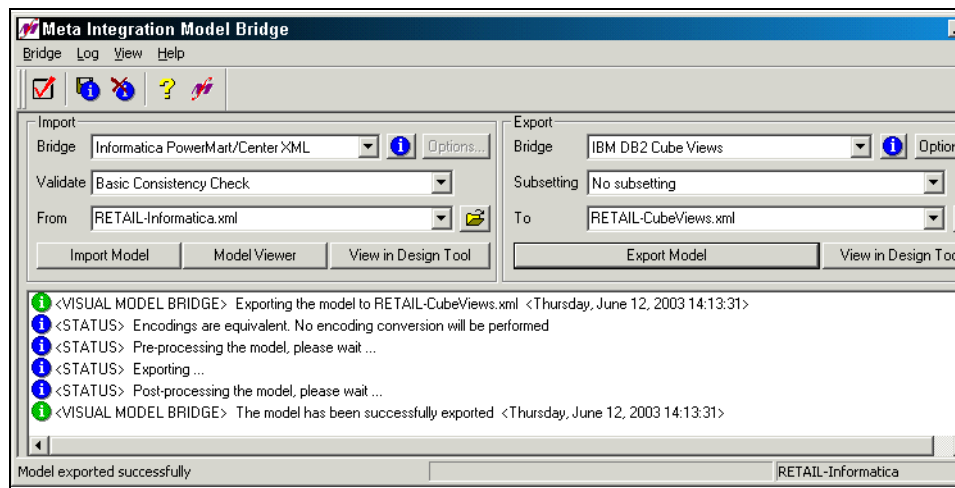


Figure 2-90 Exporting the model to DB2 Cube Views

3) Using DB2 Cube Views, import this DB2 Cube Views XML file

At this point, the cube model file is ready for importing. Select the **Import** item in the OLAP Center menu and use the wizard to import the file. You can see the result in Figure 2-91.

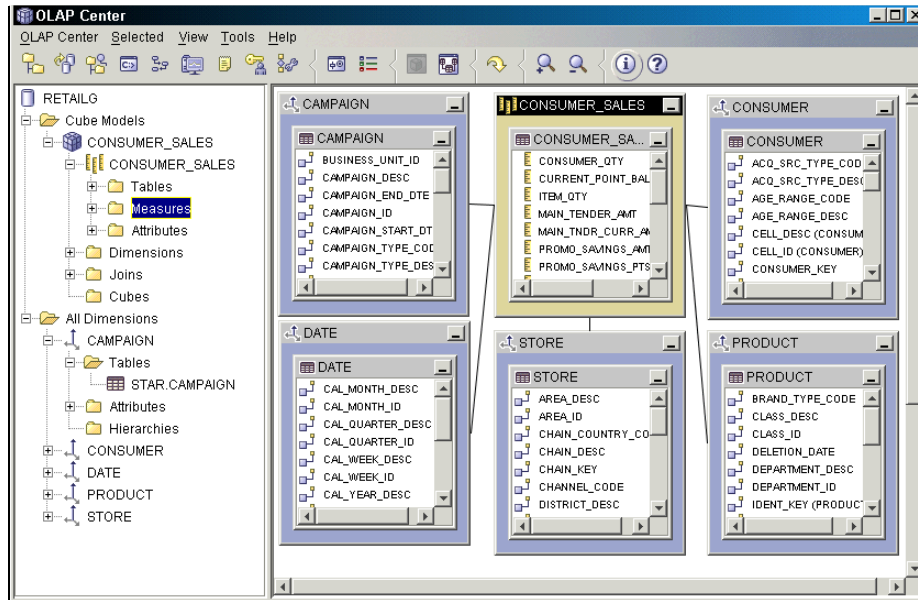


Figure 2-91 The cube model as imported in DB2 OLAP Center

Congratulations, you have imported into DB2 Cube Views a star schema designed in Informatica PowerMart/Center!

2.6 Refresh considerations

The metadata flows described above enable the forward engineering and reverse engineering of metadata between different tools (ERwin, PowerDesigner, Rose, DB2 Cube Views, DB2 Warehouse Center, PowerMart/Center) from different vendors (Computer Associates, Sybase, IBM, Informatica, OMG) using different metadata file formats (XML, ERX, PDM, MDL, CWM XMI) carrying metadata under different methodologies (Relational, OLAP).

The metadata conversion process between these tools, formats, and methodologies is complex and is sometimes more complicated in one direction than the other. For example, the definition of a relational foreign key can be used to create an OLAP join. However, when converting the other way, an OLAP join does not always represent a referential integrity constraint, and can be defined in the context of a specific business query.

Change in the enterprise is a reality and therefore, each of these tools have implemented metadata version and configuration management features to properly capture change and manage the versions of the enterprise metadata.

Most of these tools have also implemented their own metadata repository, which can store, compare and merge different versions of the metadata. For example, two versions of an ERwin model can be stored in the ModelMart repository and they can be compared and merged using the ERwin 'Complete Compare' feature; two versions of a PowerDesigner model can be stored in the PowerDesigner Repository and they can be compared using the 'Compare Models' feature.

Whether change occurs first in the database, or in a tool managing the database, and whether it is a small incremental update of a dramatic new version, it needs to be propagated to the other tools in the enterprise, and these tools also need to understand what has changed and how to handle this new version of the metadata.

The Meta Integration Model Bridge utility can extract the new version of the metadata from the source tool where the change happened, transform this metadata using sophisticated forward engineering and reverse engineering algorithms across vendors tools, formats, and methodologies, and publish the new version of the metadata into the destination tool.

To analyze the new version of the metadata in the destination tool and compare it to the current version of the metadata that may be already in place, it is recommended to use the version management features such as metadata comparator and metadata integrator in the destination tool, such as the ones implemented in most design tools, ETL tools and their underlying metadata repositories.

In case of DB2 Cube Views as a destination of a metadata flow, the version and configuration management features are available in the XML import wizard. They can be used to control how the current version of the metadata stored in the DB2 catalog can be replaced by the new version of the metadata in the XML file.

Using a third party metadata management suite such as a metadata repository equipped with advanced metadata versions comparison and integration tools could provide additional and complementary features in this regard.

For example, the Meta Integration Repository server (MIR) and Works client (MIW) suite of software is fully equipped for metadata version and configuration management, with a metadata repository manager, metadata comparator, metadata integrator, metadata mapper, in addition to all the metadata bridges also available in the Meta Integration Model Bridge (MIMB) utility (more than 40 of them as of summer 2003).

2.7 Conclusion: benefits

DB2 Cube Views simplifies and expedites business data analysis by presenting relational information as multidimensional objects.

To take advantage of these new features, you need to define this multidimensional metadata layer and configure these objects with business names, descriptions and more. You also need to understand the structure of the underlying database, such as the definition of the tables, their dimensionality, and the integrity relationships between them. The database stores a limited amount of this information in a very laconic and cryptic form. However, this structure and business information may be already defined in an ETL tool or a design tool in your company, and being able to share it with DB2 Cube Views would make the job of understanding and creating the multi-dimensional objects much easier.

MIMB does exactly this plus more. MIMB allows you to bring design and modeling information into DB2 Cube Views, and automates the creation process of the Cube Model and its related dimensions.

When you are done with your DB2 Cube Views design, you can also use MIMB to exchange your multidimensional model with your BI and reporting tools.

MIMB allows you to reuse the multidimensional objects you've created in DB2 Cube Views and populate this metadata in your BI and reporting tools.

Because understanding business data starts with a good understanding of the enterprise metadata, MIMB plays a key role in the metadata integration of DB2 Cube Views in the complete enterprise toolset. With MIMB, your design tools, ETL tools and BI tools are now compatible with each other and can exchange metadata with DB2 Cube Views.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this Redpaper.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 98. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *DB2 Cube Views: A Primer*, SG24-7002

Other publications

These publications are also relevant as further information sources:

- ▶ *IBM DB2 Cube Views Setup and User's Guide*, SC18-7298
- ▶ *Bridge for Integration Server User's Guide*, SC18-7300
- ▶ *The Data Warehouse Toolkit* by Ralph Kimball, ISBN 0-471-15337-0

Online resources

These Web sites and URLs are also relevant as further information sources:

- ▶ IBM DB2 Cube Views Homepage:
<http://www.ibm.com/software/data/db2/db2md/>
- ▶ IBM Software Homepage:
<http://www.software.ibm.com/>
- ▶ IBM Information Management Homepage:
<http://www.software.ibm.com/data/>
- ▶ Meta Integration Model Bridge (MIMB) Homepage:
www.metaintegration.net/Products/MIMB
- ▶ MIMB documentation Homepage:
www.metaintegration.net/Products/MIMB/Documentation/

- ▶ MIMB download:
<http://www.metaintegration.net/Products/Downloads/>
- ▶ MIMB related information:
www.metaintegration.net/Products/MIMB/SupportedTools.html
www.metaintegration.net/Products/MIMB/AboutVendors.html
www.metaintegration.net/Products/MIMB/AboutStandards.html

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

A

- ad-hoc reports 6
- aggregates
 - size 6
- AllFusion ERwin v4.x Data Modeler 19
- attributes 2

B

- business names 24, 42, 44, 58, 69, 77

C

- CDM 45
- changes 93
- classes 73
- Common Warehouse Metamodel 73
- Computer Associates 92
- Computer Associates AllFusion ERwin Data Modeler 18
- Computer Associates ERWin 18
- cube
 - bottom 6
 - top levels 6
- cubes 3
- CWM 73
 - ObjectModel package 77
 - RDB package 77
- CWM file 78, 83
- CWM XMI 92
- CWM XMI file 76, 82

D

- data
 - movement 11
- data modeling tool 3
- data transformation 79
- Data Warehouse Center
 - save the model 82
- DB2
 - create tables 23
 - create the tables 50
- DB2 Cube Views 94
 - benefits 1

- create a cube model 76
- cube model export 29, 43
- export 56, 71, 76
- import 26, 40, 53, 68, 87, 91
- DB2 Cube Views with Sybase PowerDesigner 18
- DB2 Data Warehouse Center 79
 - star schema 80
- DB2 optimizer 4, 7
- DB2 Warehouse Center 80
- DB2 Warehouse Manager 79
- derivations 2
- design tools 93
- diagrams 73
- dice 2
- dimensions 2
- drill down 2, 6
- drill through 6

E

- ERwin 92
 - ERX file import 44
 - save the model 37
 - SQL DDL 36
 - star schema 33
- ERwin v4.x
 - import 31
 - save the model 23
 - SQL DDL 22
 - star schema 20
- ERX 92
- ERX file 37, 43
- ETL 10, 79
- ETL tools 3, 93–94
- extract 6

F

- formulas 3
- forward engineering 92–93
- front-end tool 2

H

- hierarchies 2

HOLAP 6

I

IBM 92

IBM Rational Rose 19

Informatica 88, 92

PowerCenter 88

PowerMart 19, 88

PowerMart Designer 88

J

Java 11

joins 2

L

logical model 24

M

MDL 92

MDL file 64, 71

measures 2–3

Meta Integration

bridges

refresh 92

data model

BI vendors 17

DB2 Warehouse Manager 17

ETL tools 17

forward engineering 14, 17

Informatica 17

metadata standards 17

OMG CWM XMI 17

reverse engineering 15, 18

forward engineering

from ERwin v3.x 33

from ERwinv4.x 19

from PowerDesigner 46

from Rational Rose 59

generic metadata flows 15

implementation steps 18

metadata movement

bi-directional 14

Model Bridges 10

Model Browser 10

Model Comparator 10

Model Integrator 10

Model Mapper 10

repository metamodel 12

reverse engineering

to ERwin v3.x 42

to ERwinv4.x 28

to PowerDesigner 55

to Rational Rose 70

Meta Integration Model Bridge 12, 93

Meta Integration Repository 12, 93

Meta Integration Works 93

Meta Object Facility 73

metadata

bridge 3

comparator 93

configuration management 93

exchange 94

import 3

integrator 93

mapper 93

movement 10

repository manager 93

versions 93

metadata repository 93

meta-metamodel 73

metamodel 73

MIMB 12–13, 94

convert 51, 56, 64, 71, 76, 83, 89

convert ERX file 38

convert XML file 24, 43

MIMB utility 80

MIMB, convert XML file 30

MIW 10

model

physical 24

MOF 73

MOLAP 6

O

Object Management Group 73

OLAP Center 2

OLAP metadata 3

OMG 73, 92

OMG CWM 12, 19

OMG CWM XMI 79

OMG UML 12

Optimization Advisor 5

data sampling 6

space 6

P

- packages 73
- PDM 45, 92
- PDM file 51
- PowerCenter
 - save target datamart 89
- PowerDesigner 92
 - Conceptual Data Models 45
 - import 57
 - Physical Data Models 45
 - save the model 50
 - shortcut 51
 - SQL DDL 49
 - star schema PDM 46

Q

- Quality Assurance 11

R

- Redbooks Web site 98
 - Contact us ix
- relationships 73
- reporting 6
- response time 4
- reverse engineering 92–93
- ROLAP 6
- rollup 2
- Rose 92
 - create the model 59
 - import 71
 - MDL file 58
 - save the model 64
 - SQL DDL 64

S

- slice 2
- sparsity 6
- SQL queries
 - aggregation 4
 - joins 4
- standard 75
- star schema 2
- super aggregates 7
- Sybase 92

T

- trends 2

U

- UML 61, 73
- UML XMI 19
- Unified Modeling Language 73
- utf-8 77

X

- XMI 73
- XMI compliant 12
- XML 92
- XML file 24
- XML Metadata Interchange 73



Redpaper

DB2 Cube Views

Getting Started with Meta Integration

Introduce DB2 Cube Views as a key player in the OLAP world

Multidimensionality is the primary requirement for an OLAP system, and the cube always refers to the collections of the data that an OLAP system implements.

Integrate metadata within the enterprise toolset

Business Intelligence and OLAP systems are no longer limited to the privileged few business analysts: they are being democratized by being shared with the rank and file employee demanding a Relational Database Management System (RDBMS) that is more OLAP-aware.

Understand how the bridge maps metadata

IBM DB2 Cube Views V8.1 (DB2 Cube Views through the Redpaper) and its cube model provides DB2 Universal Database (DB2 through the Redpaper) the ability to address multidimensional analysis and become an actor in the OLAP world, as detailed in the IBM Redbook, *DB2 Cube Views: A Primer*, SG24-7002.

This Redpaper documents the Meta Integration metadata bridges for DB2 Cube Views and will help DB2 database administrators and Business Intelligence architects understand and evaluate its benefits in their own Business Intelligence and OLAP system environment.

**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**