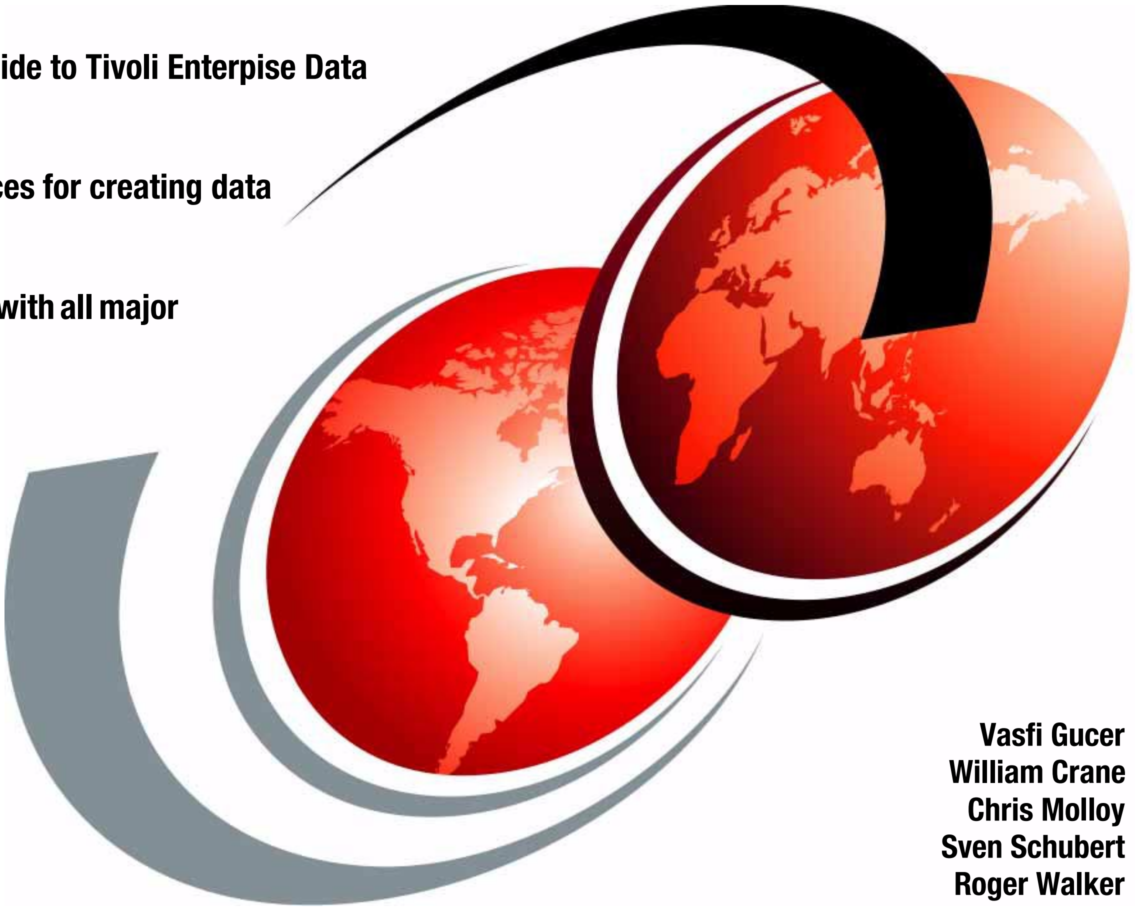
Tivoli. software

IBM

Introduction to Tivoli Enterprise Data Warehouse



Insider's guide to Tivoli Enterprise Data Warehouse

Best practices for creating data marts

Integration with all major OLAP tools

Vasfi Gucer
William Crane
Chris Molloy
Sven Schubert
Roger Walker

Redbooks

ibm.com/redbooks

International Technical Support Organization

Introduction to Tivoli Enterprise Data Warehouse

May 2002

SG24-6607-00

Take Note! Before using this information and the product it supports, be sure to read the general information in “Notices” on page xvii.

First Edition (May 2002)

This edition applies to Tivoli Enterprise Data Warehouse Version 1.1.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. JN9B Building 003 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

Contents

Figures

Tables

Notices

Trademarks

Preface

Tivoli Enterprise^{TM} Data Warehouse is a brand new technology from Tivoli, which allows customers to get cross application reports from various Tivoli and customer applications. The infrastructure enables a set of extract, transform, and load (ETL) utilities to extract and move data from Tivoli application data stores to a central data warehouse database. Tivoli Enterprise Data Warehouse's open architecture also allows data from non-Tivoli applications to be integrated into its central data repository.

This redbook gives a broad understanding of the Tivoli Enterprise Data Warehouse. Some of the topics that are covered in this redbook are:

- Concepts behind the Tivoli Enterprise Data Warehouse
- Architecture and installation
- Tips for using the Report Interface that comes with the product
- Writing your own ETLs for putting your data into Tivoli Enterprise Data Warehouse's central data repository
- Best practices in creating data marts
- Integrating Tivoli Enterprise Data Warehouse with major online analytical processing (OLAP) tools such as Brio, Business Objects, and Cognos PowerPlay
- Implementing a multi-customer (Service Provider) environment with Tivoli Enterprise Data Warehouse
- Operational considerations and troubleshooting

Most of the topics are explained using real customer implementations. We think that this redbook will be a major reference for Tivoli specialists and customers who are responsible for implementing Tivoli Enterprise Data Warehouse.

The team that wrote this redbook

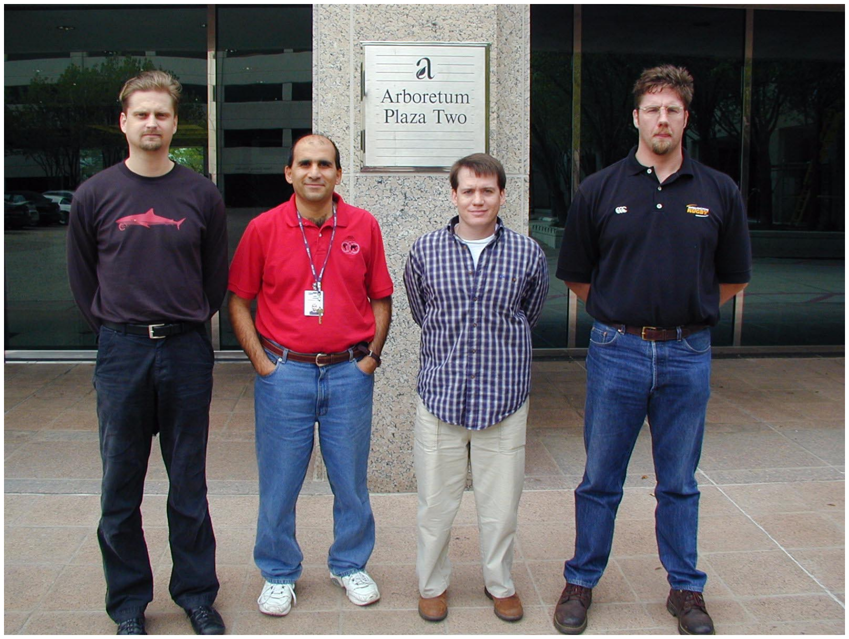


Figure 0-1 From left to right: Sven, Vasfi, William, and Roger

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

Vasfi Gucer is a Project Leader at the International Technical Support Organization, Austin Center. He worked for IBM Turkey for 10 years and has been with the ITSO since January 1999. He has more than nine years of experience in the areas of systems management, and networking hardware and software on mainframe and distributed platforms. He has worked on various Tivoli customer projects as a Systems Architect in Turkey and the U.S. Vasfi is also an IBM Certified Senior IT Specialist.

Chris Molloy is a Senior Technical Staff Member working for IBM Global Services. He has over 20 years of experience in the IT industry, and currently focuses on strategic direction for IBM Service Delivery Centers. He holds a bachelors degree in Computer Science and a masters degree in Business Administration, as well as a professional certification of a DB2 DBA. He holds a patent for some of his work in performance management, and has published several papers on performance management and capacity planning.

William Crane is a Senior Project Team Lead for global retailer in Ahold. He graduated from the University South Carolina, Columbia, South Carolina, with a B.S. in Computer Science in 1991. His duties include development and project management of Data Warehousing, E-Business and Enterprise Systems Management projects. He is a Certified Java Programmer and has programming experience in Java, C, C++, CGI, Perl, HTML, XML, AIX Shell, and PL/SQL. He also has five years of Oracle Data Warehouse development experience.

Dr. Sven Schubert is a Senior System Administrator working for SZM Studios in Germany. Sven worked as a Certified Tivoli Consultant in many Tivoli projects before he joined SZM Studios. In his company he is responsible for system management software and is also involved in project management.

Roger Walker is a Tivoli Implementation Specialist working for IBM New Zealand. He has worked in various customer projects, implementing Tivoli products such as Storage Manager, Distributed Monitoring, NetView, TEC, Remote Control, and Decision Support. He also has experience on DB2 UDB implementation and customization.

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

International Technical Support Organization, Austin Center
Budi Darmawan and Julie Czubik

International Technical Support Organization, San Jose Center
Corinne Baragoin

IBM USA
Becky Brenner, Terri Buchanan, Jonathan Cook, Catherine Cook, Vladislavas S Judys, Kathy Henley, Robin Hernandez, Kevin Kinsbury, Doug Ledden, Paige Menke, Steve Pauli, David Robinson, Brad Stern, Don Thomas, Lorraine Vassberg

Notice

This publication is intended to help Tivoli specialists who use Tivoli Enterprise Data Warehouse to implement a reporting infrastructure in their environment. The information in this publication is not intended as the specification of any programming interfaces that are provided by Tivoli Enterprise Data Warehouse Version 1.1. See the PUBLICATIONS section of the IBM Programming Announcement for Tivoli Enterprise Data Warehouse Version 1.1 for more information about what publications are considered to be product documentation.

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this 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 the address on page ii.

1

Introducing building blocks

This chapter provides an introduction to the concepts, technologies, and products behind the Tivoli Enterprise Data Warehouse. It is an ideal place to start for a reader who is coming from a Tivoli background and is not familiar with the Business Intelligence and data warehouse concepts.

This chapter has the following sections:

- Business Intelligence
- Business driving forces for Business Intelligence
- Main Business Intelligence terms
- Different Business Intelligence implementations
- Data warehouse architecture and processes
- IBM Data Warehouse Manager
- Tivoli Enterprise Data Warehouse
- Benefits of using Tivoli Enterprise Data Warehouse

1.1 Business Intelligence

Business intelligence is not business as usual. It is about making better decisions more quickly and easily.

Businesses collect enormous amounts of data every day: Information about orders, inventory, accounts payable, point-of-sale transactions, and, of course, customers. Businesses also acquire data, such as demographics and mailing lists, from outside sources. Unfortunately, based on a recent survey, over 93 percent of corporate data is not usable in the business decision-making process today.

Consolidating and organizing data for better business decisions can lead to a competitive advantage, and learning to uncover and leverage those advantages is what business intelligence is all about.

The amount of business data is increasing exponentially. In fact, it doubles every two to three years. More information means more competition. In the age of the information explosion, executives, managers, professionals, and workers all need to be able to make better decisions faster. Because now, more than ever, time is money.

Much more than a combination of data and technology, BI helps you to create knowledge from a world of information. Get the right data, discover its power, and share the value, BI *transforms information into knowledge*. Business Intelligence is the application of putting the *right information* into the hands of the *right user* at the *right time* to support the decision-making process.

1.2 Business driving forces

It can be noted that there are some business driving forces behind business intelligence, one being the need to improve ease-of-use and reduce the resources required to implement and use new information technologies. There are additional driving forces behind business intelligence, for example:

- The need to increase revenues, reduce costs, and compete more effectively.

  Gone are the days when end users could manage and plan business operations using monthly batch reports, and IT organizations had months to implement new applications. Today companies need to deploy informational applications rapidly, and provide business users with easy and fast access to business information that reflects the rapidly changing business environment.

Business intelligence systems are focused towards end user information access and delivery, and provide packaged business solutions in addition to supporting the sophisticated information technologies required for the processing of today’s business information.

- The need to manage and model the complexity of today’s business environment.

  Corporate mergers and deregulation means that companies today are providing and supporting a wider range of products and services to a broader and more diverse audience than ever before. Understanding and managing such a complex business environment and maximizing business investment is becoming increasingly more difficult. Business intelligence systems provide more than just basic query and reporting mechanisms, they also offer sophisticated information analysis and information discovery tools that are designed to handle and process the complex business information associated with today’s business environment.

- The need to reduce IT costs and leverage existing corporate business information.

  The investment in IT systems today is usually a significant percentage of corporate expenses, and there is a need not only to reduce this overhead, but also to gain the maximum business benefits from the information managed by IT systems. New information technologies like corporate intranets, thin-client computing, and subscription-driven information delivery help reduce the cost of deploying business intelligence systems to a wider user audience, especially information consumers like executives and business managers. Business intelligence systems also broaden the scope of the information that can be processed to include not only operational and warehouse data, but also information managed by office systems and corporate Web servers.

1.3 Main Business Intelligence terms

Now let us explain some of the terms (Figure 1-1 on page 4) related to Business Intelligence, which are fundamental for understanding the concepts behind the Tivoli Enterprise Data Warehouse.

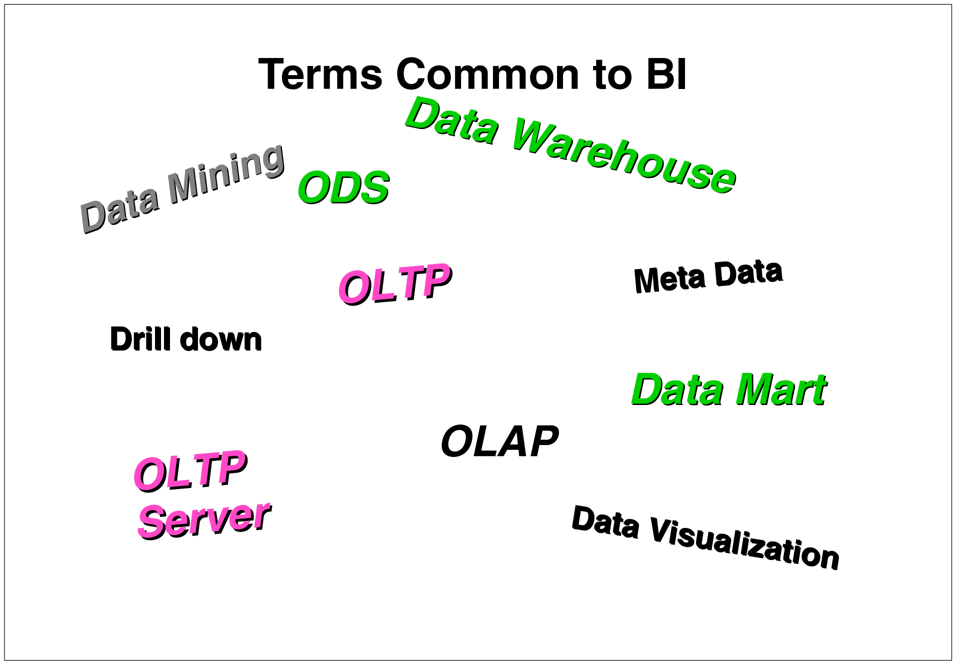


Figure 1-1 Common Business Intelligence terms

1.3.1 Operational databases

Operational databases are detail-oriented databases defined to meet the needs of, at times, very complex processes in a company. This detailed view is reflected in the data arrangement in the database. The data is highly normalized to avoid data redundancy and "double-maintenance."

1.3.2 Online transaction processing (OLTP)

OLTP describes the way data is processed by an end user or a computer system. It is detail-oriented, and highly repetitive with massive amounts of updates and changes of the data by the end user. It is also very often described as the use of computers to run the on-going operation of a business.

1.3.3 Data warehouse

A data warehouse is a database where data is collected for the purpose of being analyzed. The defining characteristic of a data warehouse is its purpose. Most data is collected to handle a company's on-going business. This type of data can be called *operational data*. The systems used to collect operational data are referred to as OLTP.

A data warehouse collects, organizes, and makes data available for the purpose of analysis in order to give management the ability to access and analyze information about its business. This type of data can be called *informational data*. The systems used to work with informational data are referred to as online analytical processing (OLAP).

Bill Inmon coined the term *data warehouse* in 1990. His definition is:

"A (data) warehouse is a subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management's decision-making process."

- Subject-oriented: Data that gives information about a particular subject instead of about a company's on-going operations.
- Integrated: Data that is gathered into the data warehouse from a variety of sources and merged into a coherent whole.
- Time-variant: All data in the data warehouse is identified with a particular time period.

1.3.4 Data mart

A data mart contains a subset of corporate data that is of value to a specific business unit, department, or set of users. This subset consists of historical, summarized, and possibly detailed data captured from transaction processing systems, or from an enterprise data warehouse. It is important to realize that a data mart is defined by the functional scope of its users, and not by the size of the data mart database. Most data marts today involve less than 100 GB of data; some are larger, however, and it is expected that as data mart usage increases they will rapidly increase in size.

1.3.5 External data source

External data is data that cannot be found in the OLTP systems but is required to enhance the information quality in the data warehouse.

1.3.6 Online analytical processing (OLAP)

OLAP is a category of software technology that enables analysts, managers, and executives to gain insight into data through fast, consistent, interactive access to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user.

OLAP functionality is characterized by dynamic multi-dimensional analysis of consolidated enterprise data supporting end user analytical and navigational activities including:

- Calculations and modeling applied across dimensions, through hierarchies, and/or across members
- Trend analysis over sequential time periods
- Slicing subsets for on-screen viewing
- Drill-down to deeper levels of consolidation
- Reach-through to underlying detail data
- Rotation to new dimensional comparisons in the viewing area

OLAP is implemented in a multi-user client/server mode and offers consistently rapid responses to queries, regardless of database size and complexity. OLAP helps the user synthesize enterprise information through comparative, personalized viewing, as well as through analysis of historical and projected data in various "what-if" data model scenarios. This is achieved through use of an OLAP Server.

1.3.7 OLAP server

An OLAP server is a high-capacity, multi-user data manipulation engine specifically designed to support and operate on multi-dimensional data structures. A multi-dimensional structure is arranged so that every data item is located and accessed based on the intersection of the dimension members that define that item. The design of the server and the structure of the data are optimized for rapid ad hoc information retrieval in any orientation, as well as for fast, flexible calculation and transformation of raw data based on formulaic relationships. The OLAP Server may either physically stage the processed multi-dimensional information to deliver consistent and rapid response times to end users, or it may populate its data structures in real-time from relational or other databases, or offer a choice of both. Given the current state of technology and the end user requirement for consistent and rapid response times, staging the multi-dimensional data in the OLAP Server is often the preferred method.

1.3.8 Metadata: A definition

Metadata is the kind of information that describes the data stored in a database and includes such information as:

- A description of tables and fields in the data warehouse, including data types and the range of acceptable values
- A similar description of tables and fields in the source databases, with a mapping of fields from the source to the warehouse
- A description of how the data has been transformed, including formulae, formatting, currency conversion, and time aggregation
- Any other information that is needed to support and manage the operation of the data warehouse

1.3.9 Drill-down

Drill-down can be defined as the capability to browse through information following a hierarchical structure. A small sample is shown in Figure 1-2.

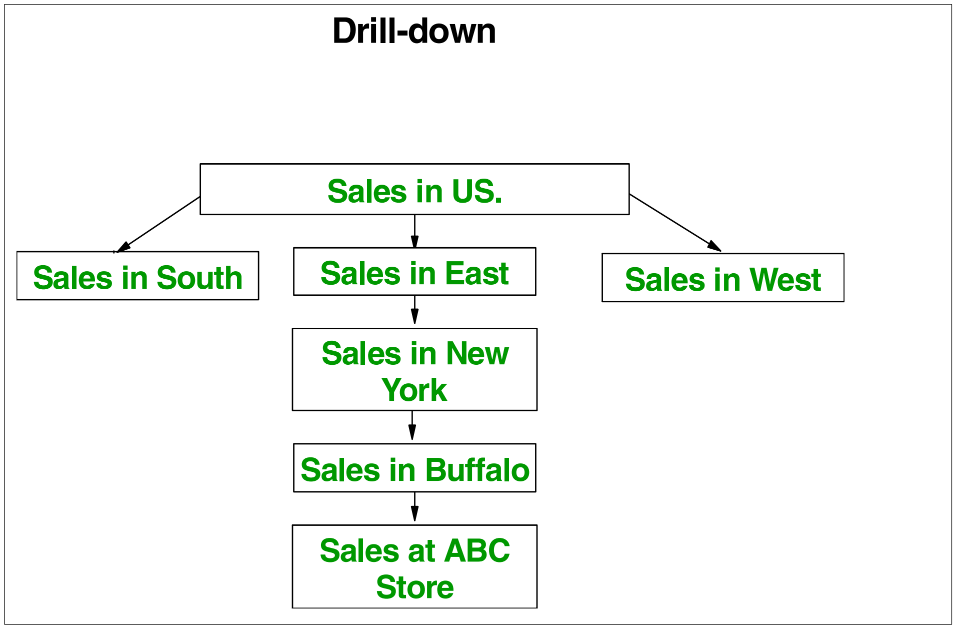


Figure 1-2 Drill-down

1.3.10 Operational versus informational databases

The major difference between operational and informational databases is the update frequency.

- On operational databases a high number of transactions take place every hour. The database is always up-to-date, and it represents a snapshot of the current business situation or, as more commonly referred to, point-in-time.
- Informational databases are usually stable over a period of time and represent a situation at a specific point in time in the past, which can be noted as historical data. For example, a data warehouse load is usually done overnight. This load process extracts all changes and new records from the operational database into the informational database. This process can be seen as one single transaction that starts when the first record gets extracted from the operational database and ends when the last data mart in the data warehouse is refreshed.

Figure 1-3 shows some of the main differences between these two database types.

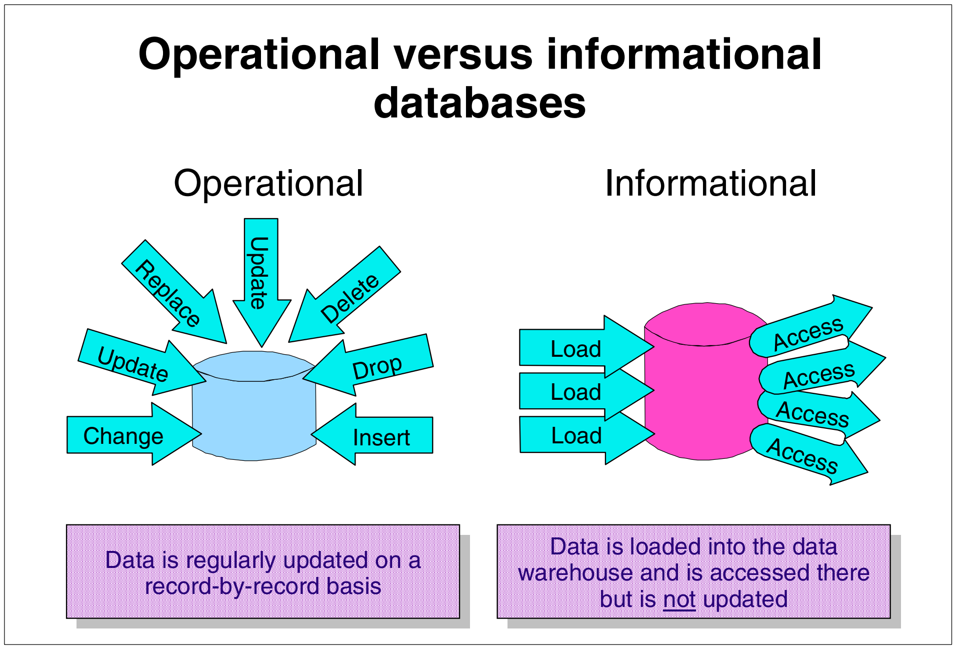

Operational versus informational databases
Operational
Informational
Replace
Update
Delete
Update
Drop
Change
Insert
Load
Load
Load
Access
Access
Access
Access
Data is regularly updated on a record-by-record basis
Data is loaded into the data warehouse and is accessed there but is not updated

Figure 1-3 Operational versus informational databases

1.3.11 Data mining

Data mining is the process of extracting *valid, useful, previously unknown,* and *comprehensible* information from data and using it to make business decisions.

After getting familiar with the terminology, now let us look at different approaches that can be taken to implement a Business Intelligence solution and show some basic concepts of a data warehouse.

1.4 Different Business Intelligence implementations

Different approaches have been made in the past to find a suitable way to meet the requirements for online analytical processing.

Figure 1-4 gives an overview of four major models for implementing a decision support system.

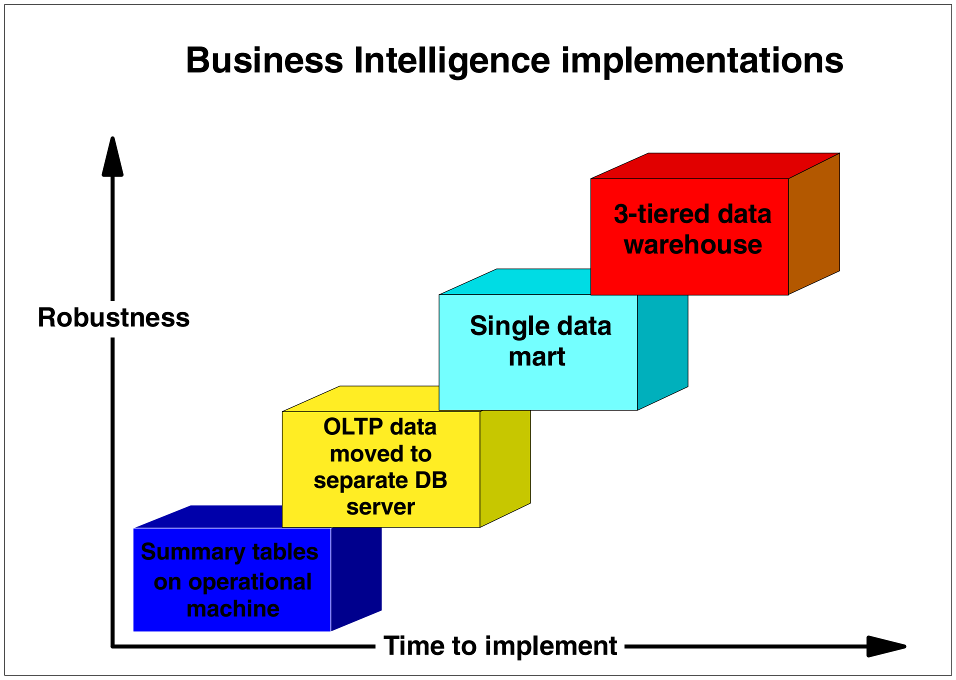
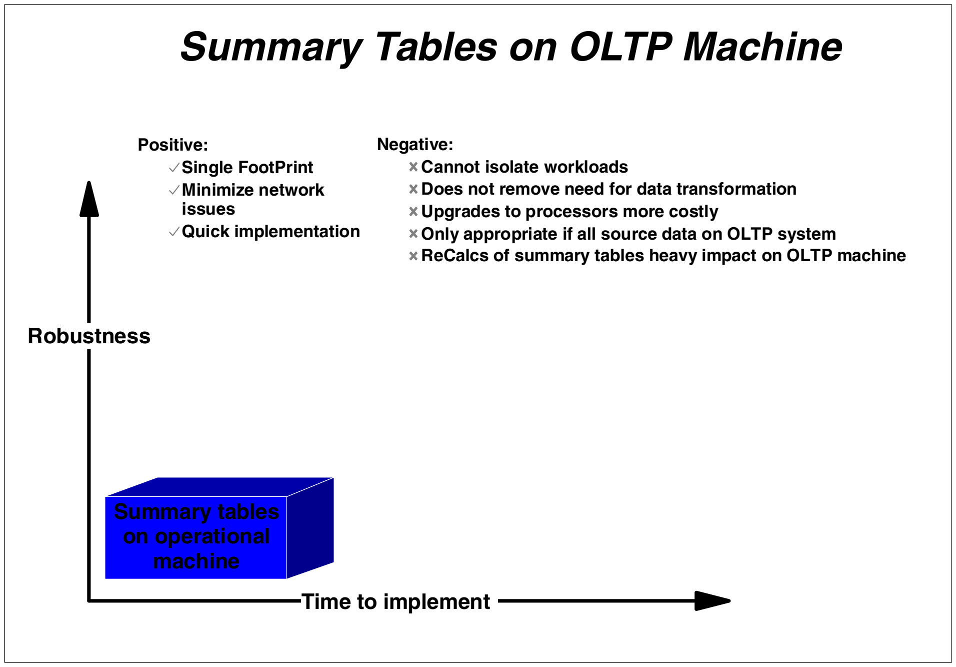

Business Intelligence implementations
Robustness
Time to implement
Summary tables on operational machine
OLTP data moved to separate DB server
Single data mart
3-tiered data warehouse

Figure 1-4 Business Intelligence implementations

The approaches shown are described in the following sections.

1.4.1 Summary table

A summary table on an OLTP system is the most common implementation that is already included in many standard software packages. Usually these summary tables cover only a certain set of requirements from business analysts. Figure 1-5 shows the advantages and disadvantages of this approach.



Summary Tables on OLTP Machine
Positive:
✓Single FootPrint
✓Minimize network issues
✓Quick implementation
Negative:
×Cannot isolate workloads
×Does not remove need for data transformation
×Upgrades to processors more costly
×Only appropriate if all source data on OLTP system
×ReCalcs of summary tables heavy impact on OLTP machine
Robustness
Summary tables on operational machine
Time to implement

Figure 1-5 Summary tables on OLTP

1.4.2 OLTP data in separate server

OLTP data is moved to a separate server—no changes in the database structure are made. This mirroring is a first step to offload the workload from the OLTP system to a separate dedicated OLAP machine. As long as no restructuring of the database takes place, this solution will not be able to track changes over time. Changes in the past cannot be reflected in the database because the fields for versioning of slowly changing dimensions are missing. Figure 1-6 on page 11 shows this approach, sometimes called “a poor man’s data warehouse.”

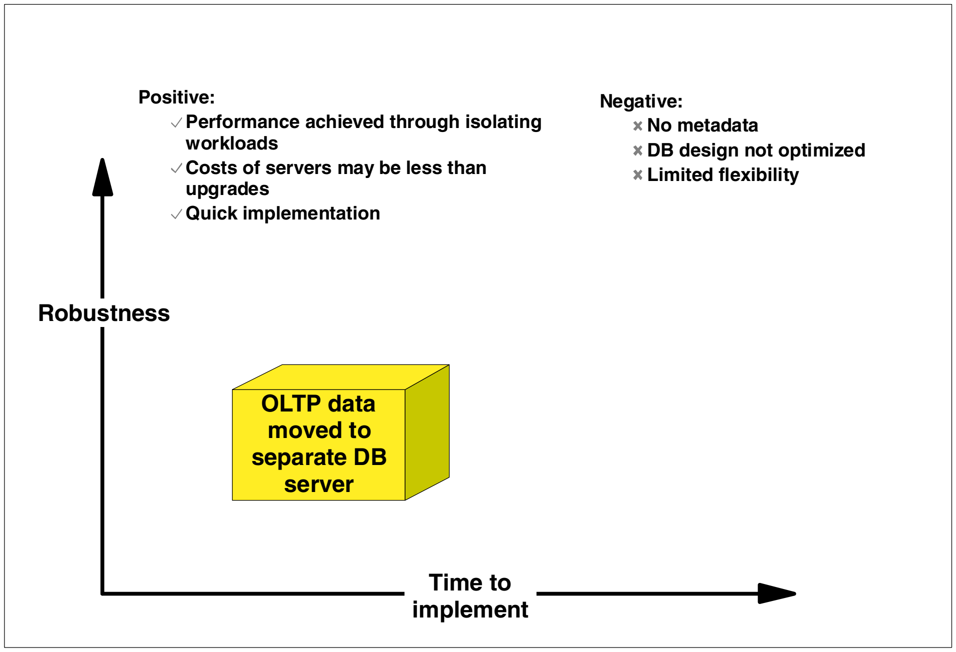

Positive:
✓ Performance achieved through isolating workloads
✓ Costs of servers may be less than upgrades
✓ Quick implementation
Negative:
✗ No metadata
✗ DB design not optimized
✗ Limited flexibility
Robustness
OLTP data moved to separate DB server
Time to implement

Figure 1-6 Poor man's data warehouse

The technique of moving the original OLTP data regularly to a dedicated system for reporting purposes is a step that can be made to avoid the impact of long-running queries on the operational system. In addition to the advantages in performance, security issues can be handled very easily in this architecture.

Totally isolated machines eliminate any interdependence between analysis and operational workload. The major problem that will still persist in this architecture is the fact that the database architecture has not changed or been optimized for query performance—the most detailed level of information is copied over to the dedicated analysis server.

The lack of summary tables or aggregations will result in long-running queries with a high number of files and joins in every request. To build an architecture like this, file transfer or FTP can be sufficient for some situations.

1.4.3 Single data mart

A growing number of customers are implementing single data marts now to get the experiences with data warehousing. These single data marts are usually implemented as a proof of concept and keep growing over time. "A data warehouse has to be built—you cannot buy it!" This first brick in the data warehouse has to be kept under control—too many single data marts would create an administration nightmare.

The *two-tiered* model of creating a single data mart on a dedicated machine includes more preparation, planning and investment. Figure 1-7 shows this approach.

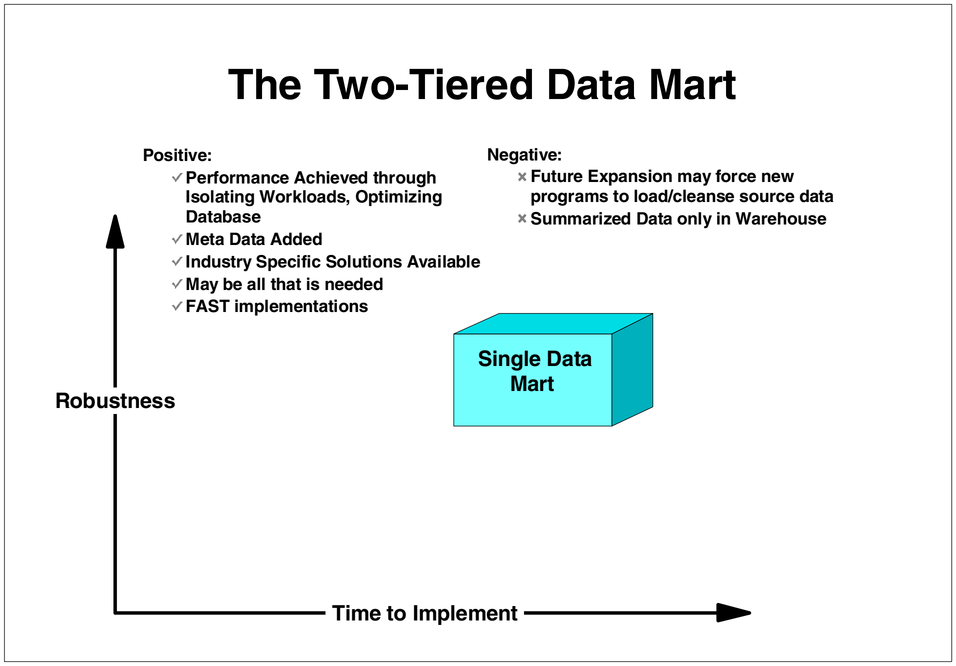

The Two-Tiered Data Mart
Positive:
✓ Performance Achieved through Isolating Workloads, Optimizing Database
✓ Meta Data Added
✓ Industry Specific Solutions Available
✓ May be all that is needed
✓ FAST implementations
Negative:
× Future Expansion may force new programs to load/cleanse source data
× Summarized Data only in Warehouse
Single Data Mart
Robustness
Time to Implement

Figure 1-7 Two-tiered data mart

The major benefits of this solution compared to the other models are in performance, precalculated and aggregated values, higher flexibility to add additional data from multiple systems and OLTP applications, and better capabilities to store historical data.

Metadata can be added to the data mart to increase the ease-of-use and the navigation through the information in the informational database.

The implementation of a stand-alone data mart can be done very quickly as long as the scope of the information to be included in the data mart is precisely limited to an adequate number of data elements.

The *three-tiered* data warehouse model consists of three stages of data stored on the system(s) (shown in Figure 1-8):

- OLTP data in operational databases
- Extracted, detailed, denormalized data organized in a Star-Join Schema to optimize query performance
- Multiple aggregated and precalculated data marts to present the data to the end user

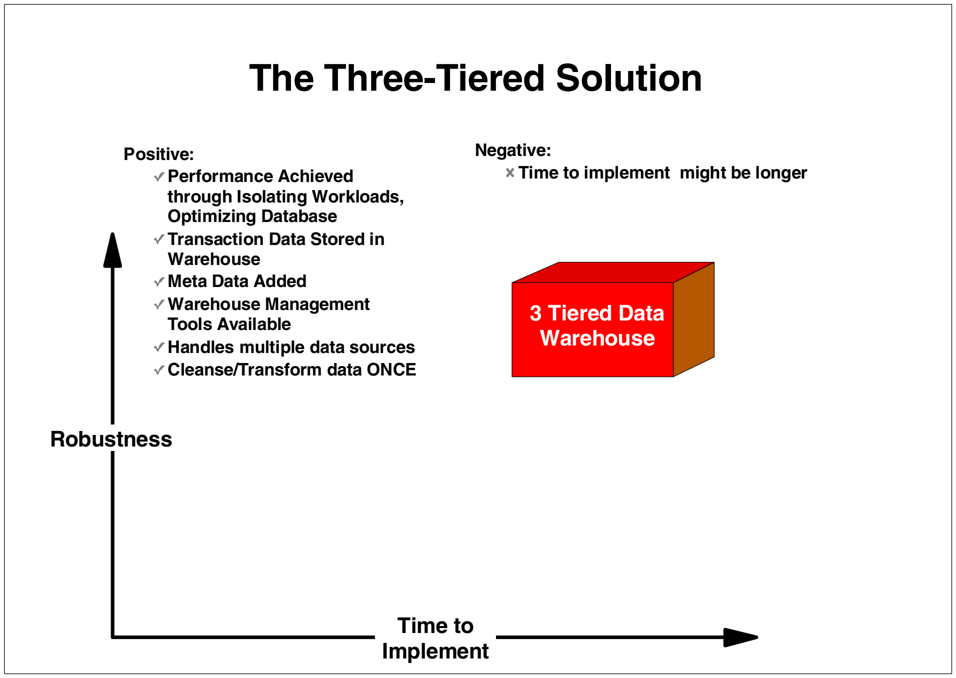

The Three-Tiered Solution
Positive:
✓ Performance Achieved through Isolating Workloads, Optimizing Database
✓ Transaction Data Stored in Warehouse
✓ Meta Data Added
✓ Warehouse Management Tools Available
✓ Handles multiple data sources
✓ Cleanse/Transform data ONCE
Negative:
× Time to implement might be longer
3 Tiered Data Warehouse
Robustness
Time to Implement

Figure 1-8 Three-tiered data mart

The characteristics of this model are:

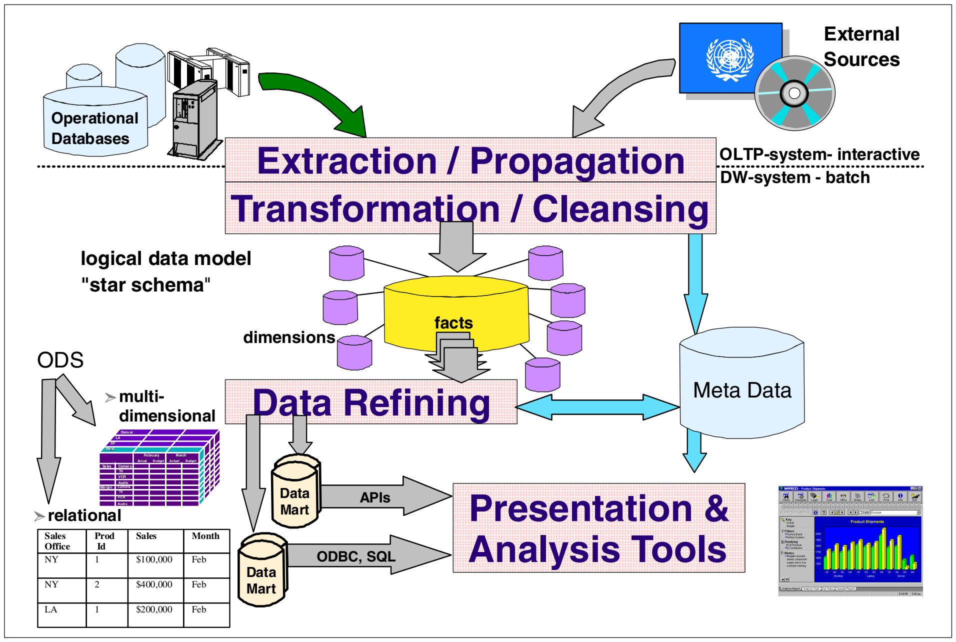
- Departmental data marts to hold data in an organizational form that is optimized for specific requests. New requirements usually require the creation of a new data mart, but have no further influence on already existing components of the data warehouse.
- Historical changes over time can be kept in the data warehouse.

- Metadata is the major component to guarantee success of this architecture—ease-of-use and navigation support for end users.
- Cleansing and transformation of data is implemented at a single point in the architecture.
- The three different stages in aggregating/transforming data offer the capability to perform data mining tasks in the extracted, detailed data without creating workload on the operational system.
- Workload created by analysis requests is totally offloaded from the OLTP system.

Note: Tivoli Enterprise Data Warehouse uses the three-tiered approach.

1.5 Data warehouse architecture and processes

Figure 1-9 shows the entire data warehouse architecture in a single view. The following sections will concentrate on single parts of this architecture and explain them in detail.



Sales Office	Prod Id	Sales	Month
NY	1	$100,000	Feb
NY	2	$400,000	Feb
LA	1	$200,000	Feb

Figure 1-9 Data warehouse components

This figure shows the following ideas:

- The processes required to keep the data warehouse up-to-date as marked are extraction/propagation, transformation/cleansing, data refining, presentation, and analysis tools.
- The different stages of aggregation in the data are OLTP data, ODS Star-Join Schema, and data marts.
- Metadata and how it is involved in each process is shown with solid connectors.

The horizontal dotted line in the figure separates the different tasks into two groups:

- Tasks to be performed on the dedicated OLTP system are optimized for interactive performance and to handle the transaction-oriented tasks in the day-to-day-business.
- Tasks to be performed on the dedicated data warehouse machine require high batch performance to handle the numerous aggregation, precalculation, and query tasks.

1.5.1 Data sources

Data sources can be operational databases, historical data (usually archived on tapes), external data (for example, from market research companies or from the Internet), or information from the already existing data warehouse environment. The data sources can be relational databases from the line of business applications. They also can reside on many different platforms and can contain structured information, such as tables or spreadsheets, or unstructured information, such as plain text files or pictures and other multimedia information.

1.5.2 Extraction/propagation

Data extraction/data propagation is the process of collecting data from various sources and different platforms to move it into the data warehouse. Data extraction in a data warehouse environment is a selective process to import decision-relevant information into the data warehouse.

Data extraction/data propagation is much more than mirroring or copying data from one database system to another. Depending on the technique, this process is either:

- Pulling (extraction) or
- Pushing (propagation)

1.5.3 Transformation/cleansing

Transformation of data usually involves code resolution with mapping tables (for example, changing *0* to *female* and *1* to *male* in the gender field) and the resolution of hidden business rules in data fields, such as account numbers. Also, the structure and relationships of the data are adjusted to the analysis domain. Transformations occur throughout the population process, usually in more than one step. In the early stages of the process, the transformations are used more to consolidate the data from different sources, whereas, in the later stages the data is transformed to suit a specific analysis problem and/or tool.

Data warehousing turns data into information; on the other hand, *cleansing* ensures that the data warehouse will have valid, useful, and meaningful information. Data cleansing can also be described as standardization of data. Through careful review of the data contents, the following criteria are matched:

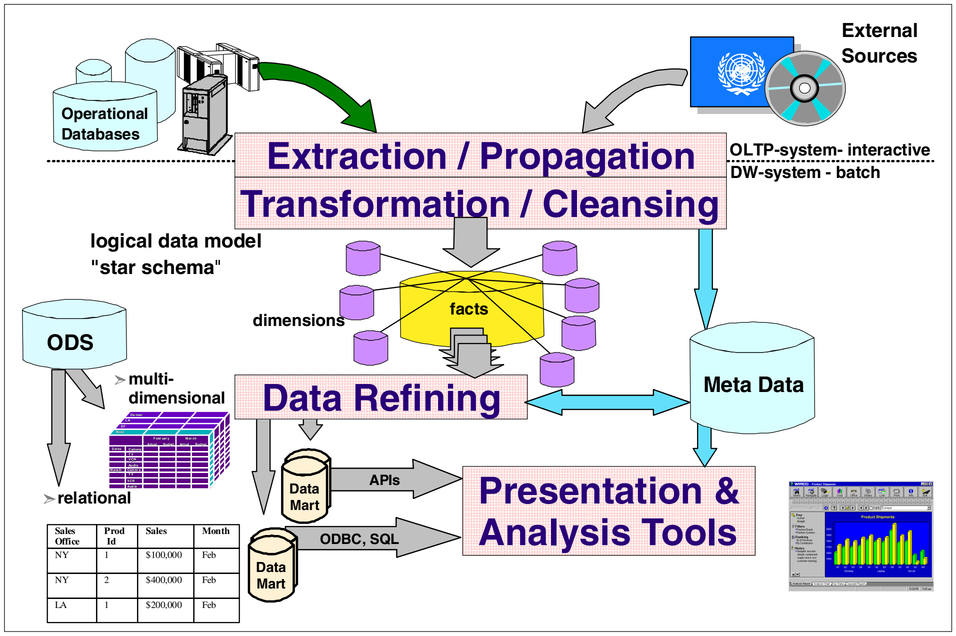
- Correct business and customer names
- Correct and valid addresses
- Usable phone numbers and contact information
- Valid data codes and abbreviations
- Consistent and standard representation of the data
- Domestic and international addresses
- Data consolidation

1.5.4 Data refining

Data refining is creating subsets of the enterprise data warehouse, which have either a multidimensional or a relational organization format for optimized OLAP performance. Figure 1-10 on page 17 shows where this process is located within the entire BI architecture.

The atomic level of information from the star schema needs to be aggregated, summarized, and modified for specific requirements. This data-refining process generates data marts that:

- Create a subset of the data in the star schema.
- Create calculated fields/virtual fields.
- Summarize the information.
- Aggregate the information.



Sales Office	Prod Id	Sales	Month
NY	1	$100,000	Feb
NY	2	$400,000	Feb
LA	1	$200,000	Feb

Figure 1-10 Data refining

This layer in the data warehouse architecture is needed to increase the query performance and minimize the amount of data that is transmitted over the network to the end user query or analysis tool.

When talking about data transformation/cleansing, there are basically two different ways the result is achieved. These are:

- Data aggregation: Change the level of granularity in the information.

  Example: The original data is stored on a daily basis—the data mart contains only weekly values. Therefore, data aggregation results in less records.

- Data summarization: Add up values in a certain group of information.

  Example: The data-refining process generates records that contain the revenue of a specific product group, resulting in more records.

1.5.5 Physical database model

In BI, talking about the physical data model is talking about relational or multidimensional data models. Figure 1-11 on page 18 shows the difference between those two physical database models.

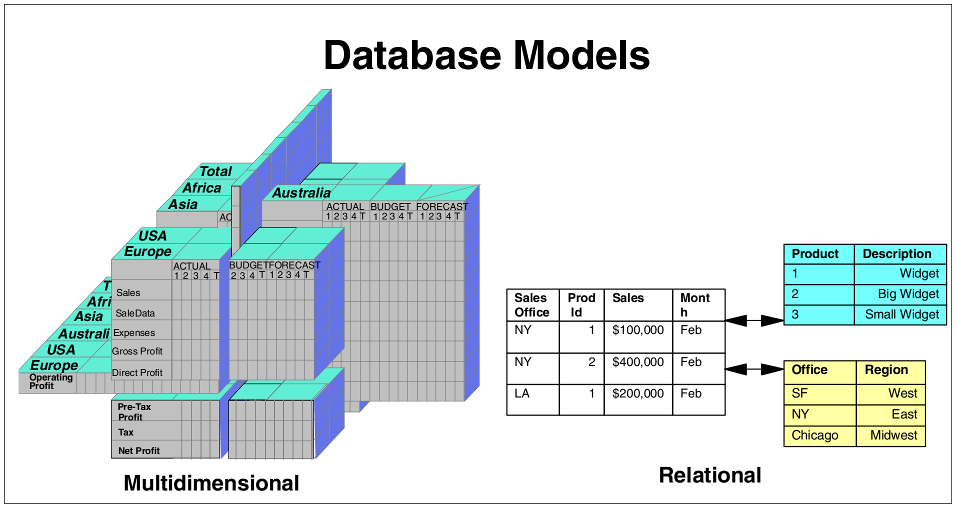


Figure 1-11 Physical database models

Both database architectures can be selected to create departmental data marts, but the way to access the data in the databases is different:

- To access data from a *relational database*, common access methods like SQL or middleware products like ODBC can be used.
- *Multidimensional databases* require specialized APIs to access the usually proprietary database architecture.

1.5.6 Logical database model

In addition to the previously mentioned physical database model, there also is a certain logical database model. When talking about BI, the most commonly used logical database model is the *Star-Join Schema*. The Star-Join Schema consists of two components, as shown in Figure 1-12 on page 19:

- Fact tables
- Dimension tables

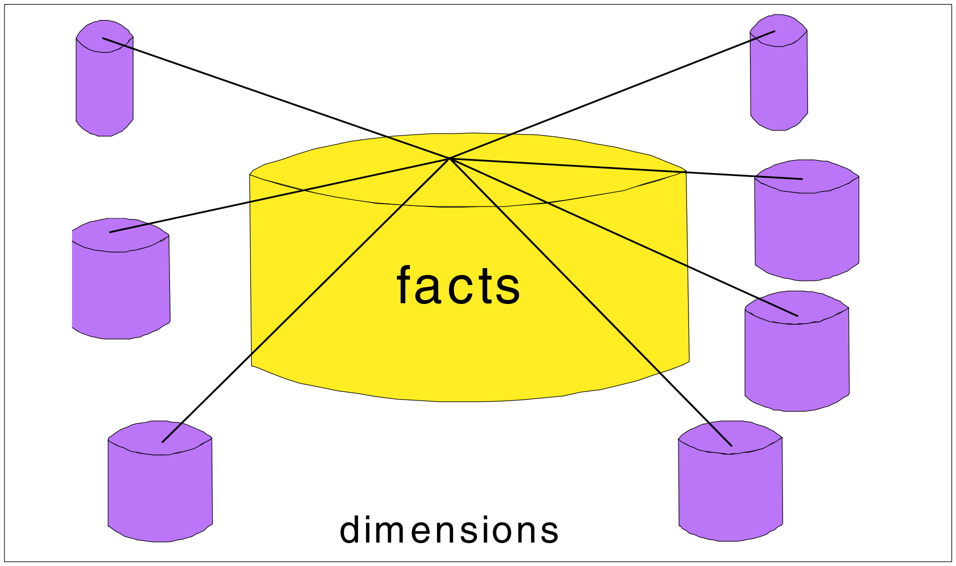


Figure 1-12 Logical data model

The following are definitions for those two components of the Star-Join Schema:

- Fact tables: What are we measuring?

  Fact tables contain the basic transaction-level information of the business that is of interest to a particular application. In marketing analysis, for example, this is the basic sales transaction data. Fact tables are large, often holding millions of rows, and mainly numerical.

- Dimension tables: By what are we measuring?

  Dimension tables contain descriptive information and are small in comparison to the fact tables. In a marketing analysis application, for example, typical dimension tables include time period, marketing region, product type, etc.

1.5.7 Metadata information

Metadata structures the information in the data warehouse in categories, topics, groups, hierarchies, and so on. It is used to provide information about the data within a data warehouse, as given in the following list and shown in Figure 1-13 on page 20:

- Subject-oriented, based on abstractions of real-world entities like (project, customer, organization, etc.).
- Defines the way in which the transformed data is to be interpreted (5/9/99 = 5th September 1999 or 9th May 1999—British or US?).

- Gives information about related data in the data warehouse.
- Estimates response time by showing the number of records to be processed in a query.
- Holds calculated fields and precalculated formulas to avoid misinterpretation, and contains historical changes of a view.

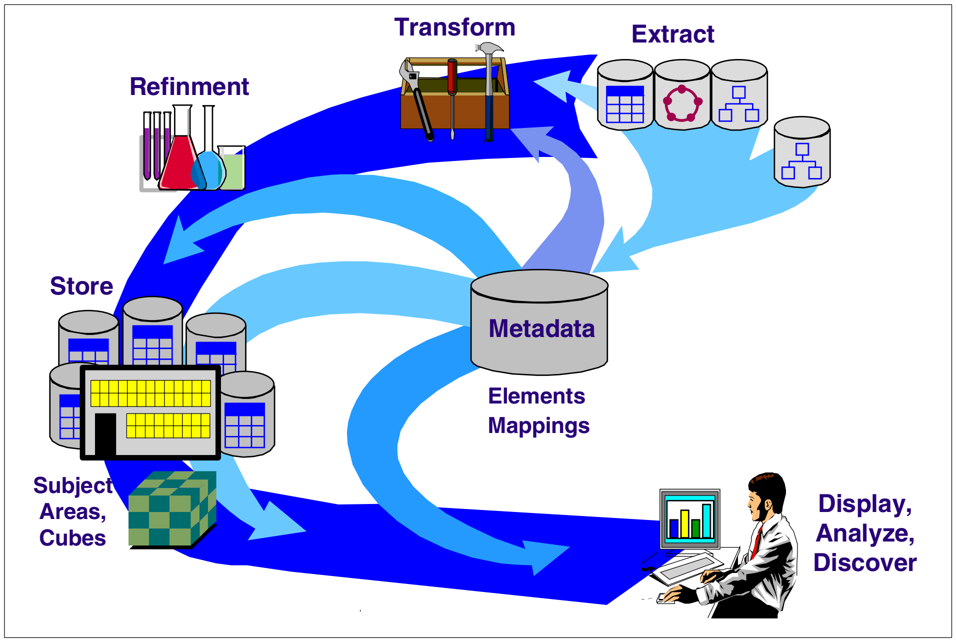
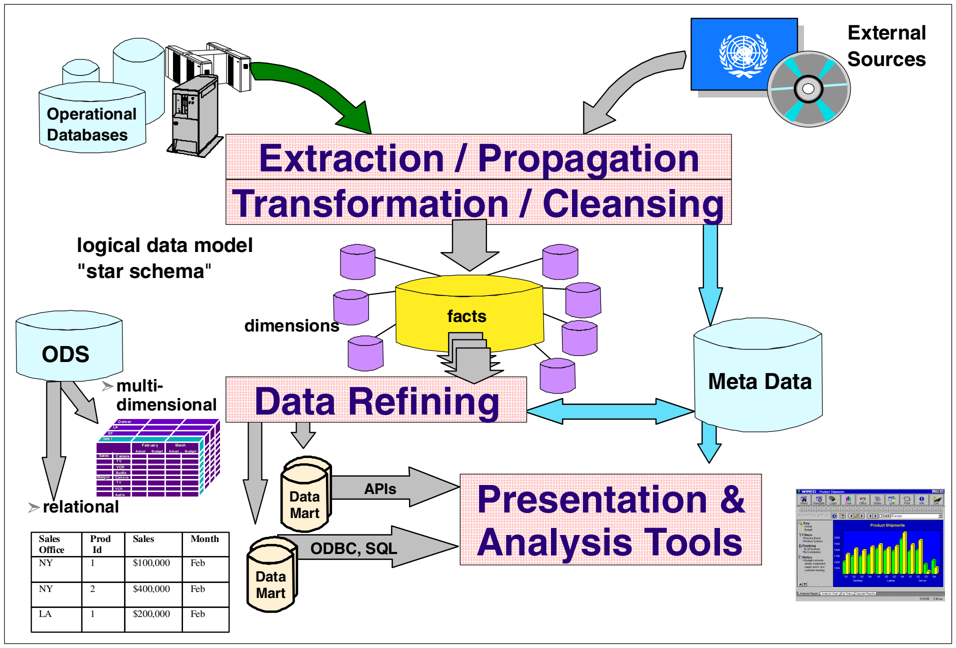


Figure 1-13 Metadata

The data warehouse administrator perspective of metadata is a full *repository* and documentation of all contents and all processes in the data warehouse, whereas, from an end user perspective, metadata is the *roadmap* through the information in the data warehouse.

1.5.8 Operational data source (ODS)

The operational data source (see Figure 1-14 on page 21) can be defined as an updatable set of integrated data used for enterprise-wide tactical decision making. It contains live data, not snapshots, and has minimal history that is retained.



Sales Office	Prod Id	Sales	Month
NY	1	$100,000	Feb
NY	2	$400,000	Feb
LA	1	$200,000	Feb

Figure 1-14 Operational data store

Here are some features of an Operational Data Store (ODS):

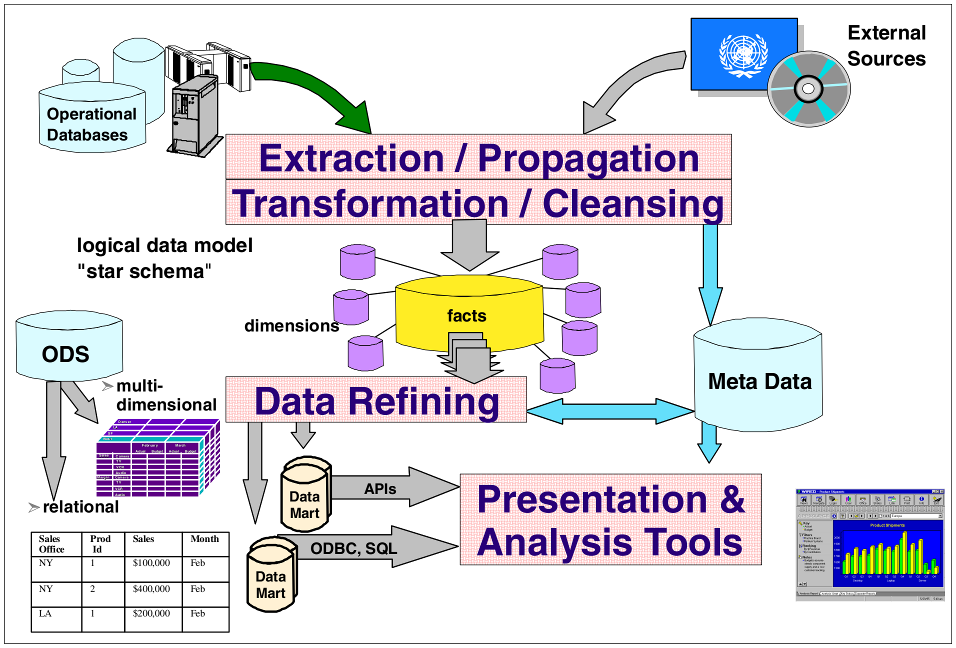
- An ODS is subject oriented: It is designed and organized around the major data subjects of a corporation, such as customer or product. They are not organized around specific applications or functions, such as order entry or accounts receivable.
- An ODS is integrated: It represents a collectively integrated image of subject-oriented data, which is pulled in from potentially any operational system. If the customer subject is included, then all of the customer information in the enterprise is considered as part of the ODS.
- An ODS is current valued: It reflects the current content of its legacy source systems. Current may be defined in different ways for different ODSs depending on the requirements of the implementation. An ODS should not contain multiple snapshots of whatever current is defined to be. That is, if current means one accounting period, then the ODS does not include more that one accounting period's data. The history is either archived or brought into the data warehouse for analysis.

- An ODS is volatile: Since an ODS is current valued, it is subject to change on a frequency that supports the definition of current. That is, it is updated to reflect the systems that feed it in the true OLTP sense. Therefore, identical queries made at different times will likely yield different results because the data has changed.
- An ODS is detailed: The definition of detailed also depends on the business problem that is being solved by the ODS. The granularity of data in the ODS may or may not be the same as that of its source operational systems.

1.5.9 Data mart

Figure 1-15 on page 23 shows where data marts are located logically within the BI architecture. The main purpose of a data mart can be defined as follows:

- Store pre-aggregated information.
- Control end user access to the information.
- Provide fast access to information for specific analytical needs or user groups.
- Represent the end user's view and data interface of the data warehouse.
- Create the multidimensional/relational view of the data.
- Offer multiple "slice-and-dice" capabilities.
- The database format can either be multidimensional or relational.



Sales Office	Prod Id	Sales	Month
NY	1	$100,000	Feb
NY	2	$400,000	Feb
LA	1	$200,000	Feb

Figure 1-15 Data mart

1.5.10 Presentation and analysis tools

From the end user's perspective, the presentation layer is the most important component in the BI architecture shown in Figure 1-16 on page 24.

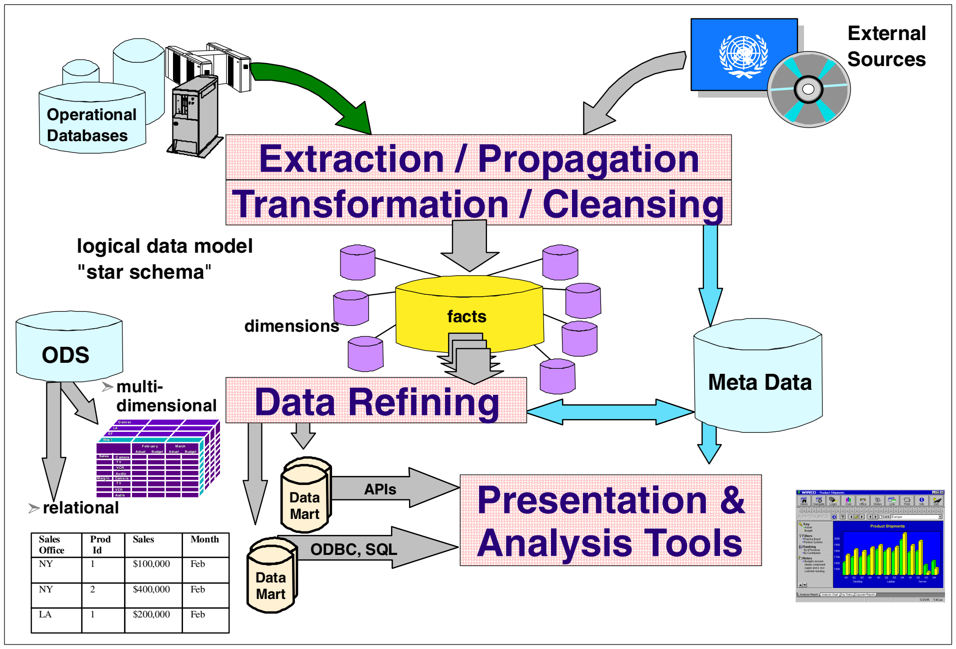
To find the adequate tools for the end users with information requirements, the assumption can be made that there are at least four user categories and the possibility of any combination of these categories.

- The power user

  Users that are willing and able to handle a more or less complex analysis tool to create their own reports and analysis. These users have an understanding of the data warehouse structure and interdependencies of the organization form of the data in the data warehouse.

- The non-frequent user

  This user group consists of people that are not interested in the details of the data warehouse but have a requirement to get access to the information from time to time. These users are usually involved in the day-to-day business and do not have the time or the requirement to work extensively with the information in the data warehouse. Their virtuosity in handling reporting and analysis tools is limited.



Sales Office	Prod Id	Sales	Month
NY	1	$100,000	Feb
NY	2	$400,000	Feb
LA	1	$200,000	Feb

Figure 1-16 Presentation and analysis tools

- Users requiring static information

  This user group has a specific interest in retrieving precisely defined numbers in a given time interval, such as: "I have to get this quality-summary report every Friday at 10:00 a.m. as preparation for our weekly meeting and for documentation purposes."

- Users requiring dynamic or ad hoc query and analysis capabilities

  Typically, this is a business analyst. All the information in the data warehouse might be of importance to those users, at some point in time. Their focus is related to availability, performance, and drill-down capabilities to slice-and-dice through the data from different perspectives at any time.

Different user-types need different front-end tools, but all can access the same data warehouse architecture. Also, the different skill levels require different visualization of the result, such as graphics for a high-level presentation or tables for further analysis.

This concludes our discussion of general BI concepts. Next, we will discuss DB2 DataWarehouse Manager, IBM's solution for creating and managing a data warehouse, which is also used by the Tivoli Enterprise Data Warehouse product.

Note: Tivoli Enterprise Data Warehouse is shipped withthe DB2 UDB and DB2 Warehouse Manager feature.

1.6 DB2 DataWarehouse Manager

DB2 warehouse management is built on the DB2 UDB and DB2 DataWarehouse Manager feature. It provides an integrated, distributed, heterogeneous warehouse management infrastructure for designing, building, maintaining, governing, and accessing highly scalable, robust data warehouses, operational data stores, and datamarts stored in DB2 UDB databases.

DB2 UDB and DB2 DataWarehouse Manager help warehouse administrators:

- To manage data volumes, to move data directly from source-to-target (also allowing packaged and simplified access to popular partner products such as SAP R/3), and to control the servers on which transformations take place with distributed warehouse agents.
- To speed warehouse and data mart deployment with commonly used, pre-built data cleansing and statistical transformations.
- To build and manage from a central point of control, integrated in DB2 UDB, utilizing the Data Warehouse Center graphical user interface.

DB2 warehouse management consists of:

- An administrative client to define and manage data warehousing tasks and objects, and warehouse or datamart operations: The *Data Warehouse Center*
- A manager to manage and control the flow of data: The *warehouse server*
- Agents residing on DB2 UDB server platforms (that could be also SUN, HP, and so on) to perform requests from the manager or warehouse server: The *local or remote warehouse agent*
- A *warehouse control database* storing the warehouse management metadata on a DB2 UDB database on a UNIX or Intel server

- A metadata administrative and publishing tool with its own administration graphical user interface (GUI): *Information Catalog Manager* to manage and present both technical and business metadata.

The different components of the DB2 DataWarehouse Manager are shown in Figure 1-17.

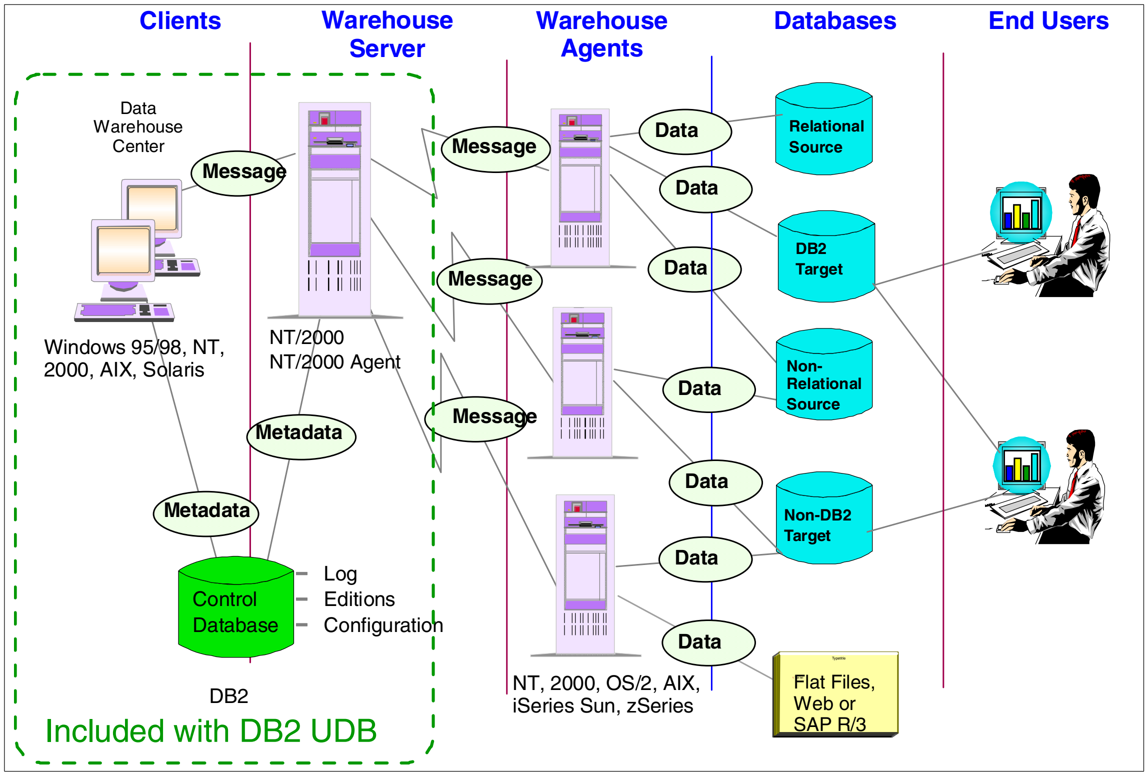


Figure 1-17 DB2 DataWarehouse Manager

The following Redbooks are excellent sources of information on DB2 DataWarehouse Manager and Business Intelligence concepts. Please refer to them for additional information on these topics.

- *Business Intelligence Certification Guide*, SG24-5747
- *DB2 Warehouse Management: High Availability and Problem Determination Guide,* SG24-6544

1.7 Tivoli Enterprise Data Warehouse

Now after having set the stage, we can start our discussion about Tivoli Enterprise Data Warehouse, which is the main subject of this redbook.

We will begin the discussion by reviewing the need for the Tivoli Enterprise Data Warehouse.

1.7.1 The problem

There has been an inherent problem with most management applications. Management applications collect a lot of enterprise data and store them in databases or files. Each application uses its own database, spreadsheet data, or flat file with their own different formats. For each application an individual report has to be set up with a reporting tool. This gives information about the data collected by the individual application, but it only shows part of the story. The analysis and correlation of data from different applications in one report is impossible, or at least hard to achieve.

This argument has been also partly true for Tivoli applications, until the availability of Tivoli Data Warehouse.

This problem is shown in Figure 1-18 on page 28.

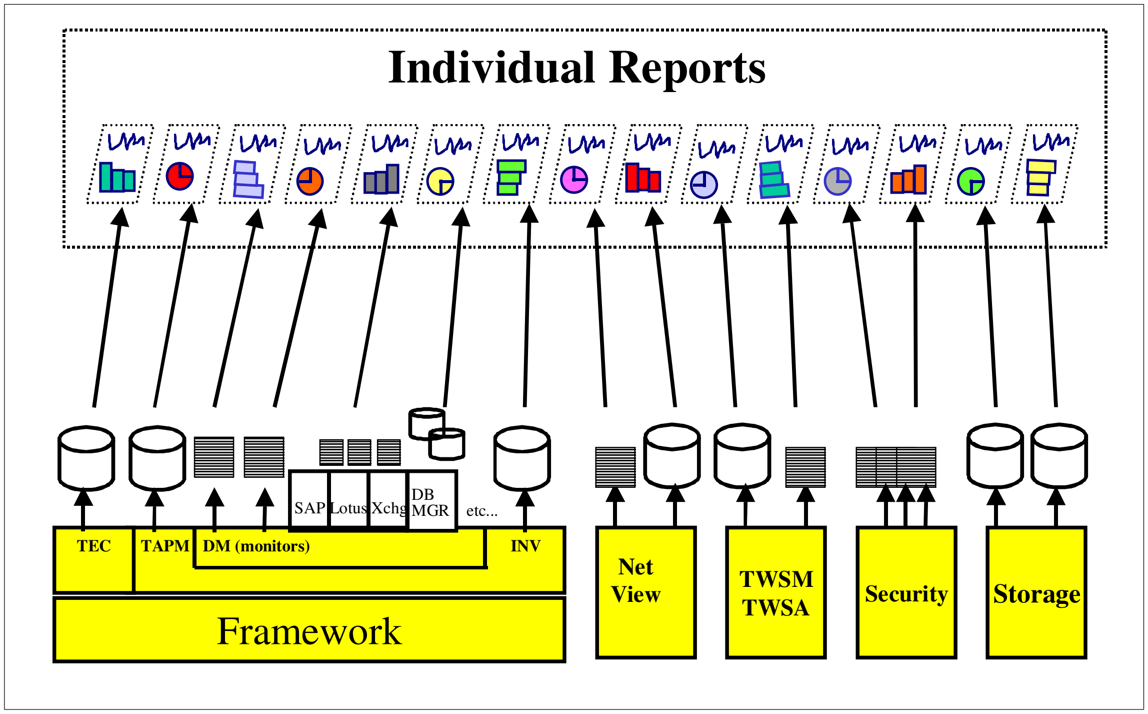


Figure 1-18 Reporting without Tivoli Enterprise Data Warehouse

Tivoli first tried to solve this problem by introducing Tivoli Decision Support a few years ago. Tivoli Decision Support has been successful to a great extend and provides great value in the distributed management environment. However, it does have some limitations in the following areas:

- Flexibility: Flexibility in the types of reports and the look and feel of reports is a key requirement. The data should be “pluggable” to any reporting application (including all OLAP tools).
- Customization: The ability to customize is also a critical factor. The current Tivoli Decision Support (TDS) product brings great value out of the box, but many customers want to customize the data and the kind of reports that can be generated. This customization is a difficult process with TDS and there is a need to provide an environment where customization can be done more easily. This includes the use of data from multiple management applications, both Tivoli and non-Tivoli.
- Security-scalability: The current TDS environment also has some limitations in the area of scalability (number of users able to look at and modify reports at one time). It is also important to allow customers to provide security related to who can view what reports and what data related to what portions of the

enterprise. For instance, in a Service Provider environment, individual customers should only be able to see reports related to the services they receive. This also applies to large enterprises where reports must be secure on a division or business unit basis.

- Web enablement: The current TDS interface is Windows-based. Though it does provide some support for publishing reports to HTML and is accessible via a Web browser, this facility is not all that it could be. Customers would like to have access to the reports via a standard browser.
- Globalization: There are some limitations on the National Language Support capability of TDS.
- Management application basis: TDS guides typically are built on a management application-by-management application basis. To be able to do high-value reporting applications, such as service level management or capacity planning, it is critical to be able to correlate data from across many management applications.

1.7.2 The solution

Tivoli Enterprise Data Warehouse is made available by Tivoli to solve exactly this problem and also overcome the limitations of TDS. The key point of Tivoli Enterprise Data Warehouse is that all historical data from different management applications is collected in one centralized database, the Tivoli Data Warehouse. The schemas of this database are open and published. Systems management data from third party applications can also be easily integrated.

This new architecture is shown in Figure 1-19 on page 30.

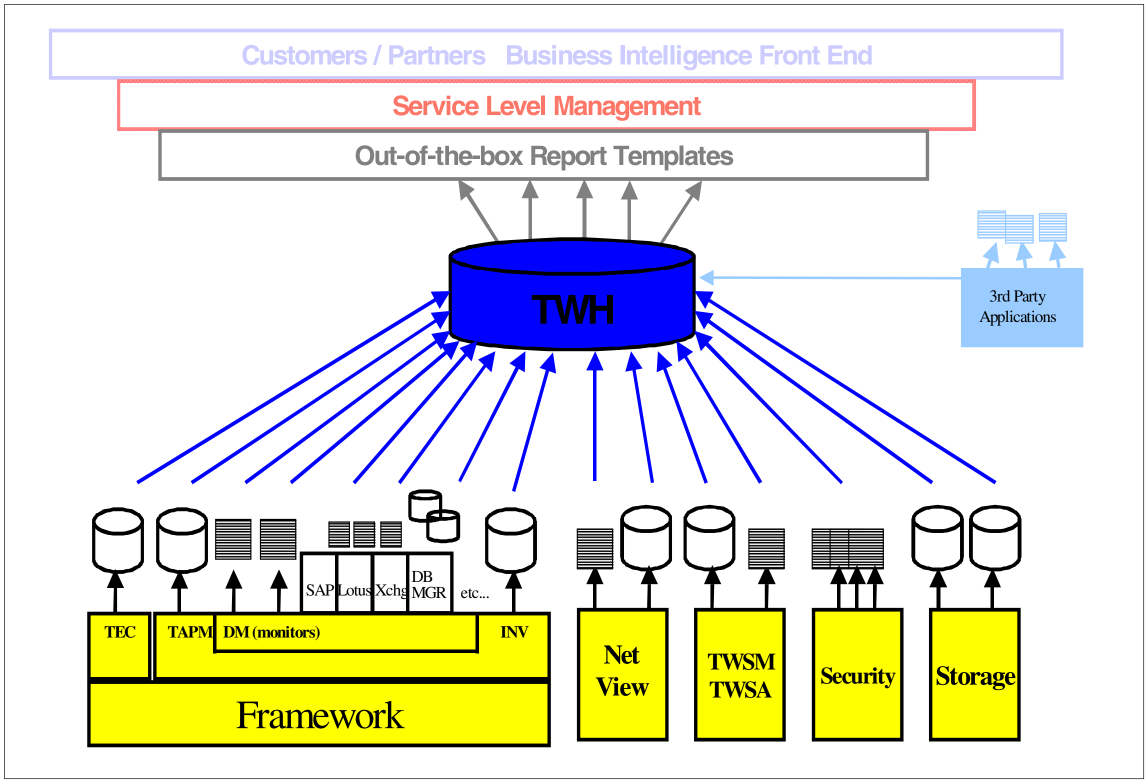


Figure 1-19 Reporting with Tivoli Enterprise Data Warehouse

In the Tivoli Enterprise Data Warehouse, all data is aggregated and correlated for use by reporting and third party OLAP tools and also by planning, trending, analysis, accounting, and data mining tools.

Tivoli Enterprise Data Warehouse applications also provide static standard reports using a Web console reporting tool. In Release 1 of Tivoli Enterprise Data Warehouse, three classes of reports are supported:

- Two-dimensional representation of measurements versus components/groups of components
  - Graphical report
  - Tabular report
- Measurements versus time

We will discuss the architecture of Tivoli Enterprise Data Warehouse in more detail in Section 2.1, “Tivoli Enterprise Warehouse components” on page 34.

Important: Tivoli Enterprise Data Warehouse is not an independent product. It is delivered for free with all Tivoli Enterprise Data Warehouse-enabled applications. All enabled Tivoli source applications will be shipped with the necessary Tivoli Enterprise Data Warehouse components to import their data into the central data warehouse.

1.7.3 Benefits of using Tivoli Enterprise Data Warehouse

Customers can benefit from using Tivoli Enterprise Data Warehouse in various ways such as:

- Tivoli Enterprise Data Warehouse collects historical data from many Tivoli applications into one central place.

  Tivoli Enterprise Data Warehouse collects the underlying data about customers' network devices/connections, desktops/servers, applications/software, and the problems and activities that have gone on to manage the infrastructure. This allows the customers to construct an end-to-end view of their enterprise and view the components independent of specific applications used to monitor and control resources.

- Tivoli Enterprise Data Warehouse adds value to raw data.

  Tivoli Enterprise Data Warehouse performs data aggregation (such as daily or weekly) and allows the user to restrict the amount of data stored in the central data repository. The data is also cleaned and consolidated in order to allow the data model of the central repository to share common dimensions. For example, Tivoli Enterprise Data Warehouse ensures that time, hostname, and IP address are the same dimensions across all the applications.

- Tivoli Enterprise Data Warehouse allows the correlation of information from many Tivoli applications.

  Tivoli Enterprise Data Warehouse can also be used to derive added value by correlating data from many Tivoli applications. It allows *killer reports* to be written, which correlate cross application data. The first *killer application* that utilizes Tivoli Enterprise Data Warehouse is the Tivoli Service Level Advisor (TSLA), which has become available recently. It uses and correlates the data from the following applications to check conformance with predefined service levels.

  – Tivoli Enterprise Console
  – Tivoli Distributed Monitoring
  – Tivoli Web Services Manager
  – Tivoli Application Performance Management
  – Tivoli Business Systems Manager

- Tivoli Enterprise Data Warehouse uses open, proven interfaces for extracting, storing, and sharing the data.

  Tivoli Enterprise Data Warehouse can extract data from any application (Tivoli and non-Tivoli) and store them in a common central database. The Tivoli Enterprise Data Warehouse application also provides transparent access for third party BI solutions (CWM standard), such as IBM DB2 OLAP, Crystal Decisions, Cognos, Business Objects, Brio Technology, and Microsoft OLAP Server. CWM stands for Common Warehouse Metadata, an industry standard specification for metadata interchange defined by the Object Management Group (see `http://www.omg.org`). Tivoli Enterprise Data Warehouse provides a Web-based reporting front end, called the Report Interface, but the open architecture provided by the Tivoli Enterprise Data Warehouse allows other BI front ends to be used to access the data in the central warehouse. The value here is *flexibility*. Customers can use the reporting application of their choice, and are not limited to any application.

- All Tivoli applications will provide standard out-of-the-box reports.

  All Tivoli applications will provide standard out-of-the-box reports and report templates, utilizing the Tivoli Enterprise Data Warehouse's common central warehouse. These reports will provide similar information to those provided by many of the TDS guides today. As mentioned earlier, Tivoli will also develop and provide (as separate products) high value, cross-product reporting applications or killer applications such as Tivoli Service Level Advisor.

- Tivoli Enterprise Data Warehouse provides robust security mechanism.

  Tivoli Enterprise Data Warehouse provides a robust security mechanism by allowing data marts to be built with data from subsets of managed resources. By providing database level authorization to access those data marts, Tivoli Enterprise Data Warehouse can address most of the security requirements related to limiting access to specific data to those customers/business units with a need-to-know.

- Tivoli Enterprise Data Warehouse provides a scalable architecture.

  Since Tivoli Enterprise Data Warehouse depends on the proven and industry standard relational database management system (RDBMS) technology, it provides a scalable architecture for storing and retrieving data.

2

Tivoli Enterprise Data Warehouse architecture

This chapter discusses the architecture of Tivoli Enterprise Data Warehouse and how it fits in the general data warehouse model.

This chapter has the following sections:

- Tivoli Enterprise Data Warehouse components
- Tivoli Enterprise Data Warehouse architecture
- How applications use Tivoli Enterprise Data Warehouse

2.1 Tivoli Enterprise Warehouse components

In this section we introduce the basic components of the Tivoli Enterprise Data Warehouse and briefly describe their functionality. We explain how the Tivoli Enterprise Data Warehouse application is packaged.

2.1.1 Basic components

The Tivoli Enterprise Data Warehouse is an application used to collect and manage data from various Tivoli and non-Tivoli system management applications. The data is imported from the source applications, stored centrally, and further processed to fit the needs of the end users. Here we describe the basic components of the Tivoli Enterprise Data Warehouse in the logical order of the data flow.

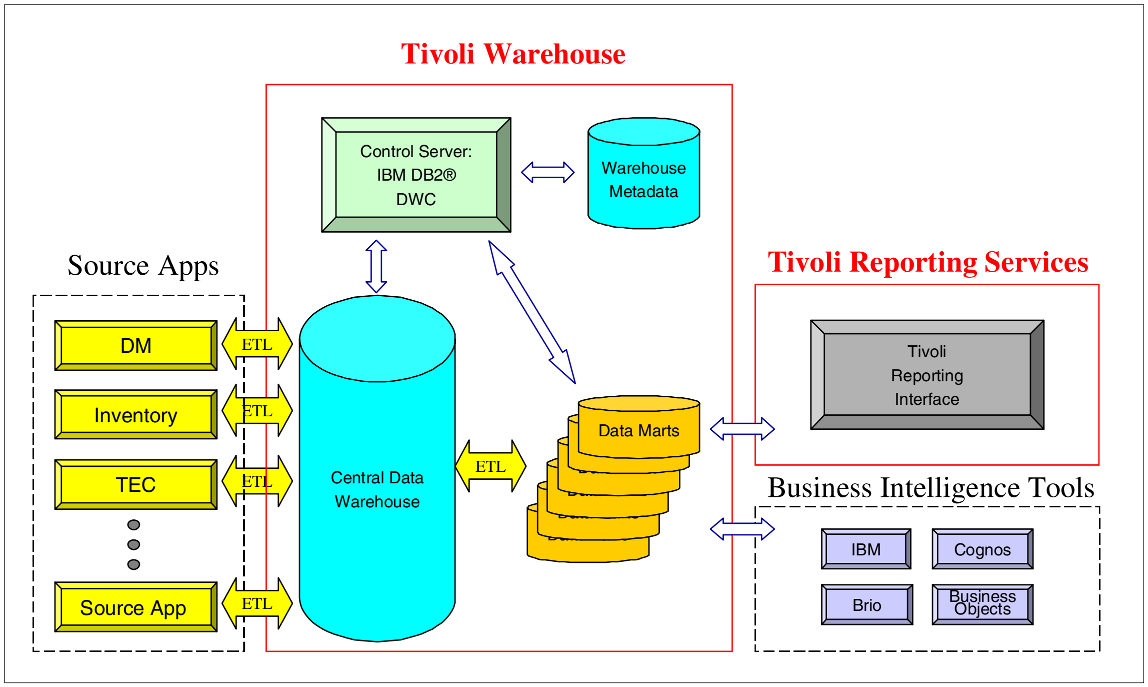


Figure 2-1 Components of the Tivoli Enterprise Data Warehouse

The first step to introduce Tivoli Enterprise Data Warehouse is to enable the source applications. This means providing all tools and customizations necessary to import the source operational data into the central data warehouse. All components needed for that task are collected in so-called *warehouse packs* for each source application.

Future releases of all Tivoli applications will be Tivoli Enterprise Data Warehouse-ready and shipped with their warehouse packs. How to enable your third party applications for Tivoli Enterprise Data Warehouse is covered in Chapter 5, “Integration of application data to central data repository” on page 127 and Chapter 6, “How to create data marts” on page 179. In these chapters we will implement two case studies.

One important part of the warehouse packs are the *ETL programs*. The abbreviation ETL is for extract, transform and load data. In principle, ETL programs process data in three steps. First they extract the data from a data source. Then the data is validated, transformed, aggregated, and/or cleansed so that it fits the format and needs of the data target. Finally the data is loaded into the target database.

In Tivoli Enterprise Data Warehouse there are two types of ETLs. The *central data warehouse ETL* pulls the data from the source applications and loads it into the central data warehouse (see Figure 2-1 on page 34). The central data warehouse ETL is also known as *source ETL* or *ETL1.* The second type of ETL is the *data mart ETL,* which is discussed later.

The *central data warehouse (CDW)* is the database that contains all enterprise-wide historical data (with hour as the lowest granularity). This data store is optimized for the efficient storage of large amounts of data and has a documented format that makes the data accessible to many analysis solutions. The database is organized in a very flexible way, which lets you store data from new applications without adding or changing tables. Section 2.2.4, “Central data warehouse data model” on page 42 covers the CDW model in detail.

The *data mart ETL* extracts a subset of historical data from the central data warehouse that contains data tailored to and optimized for a specific reporting or analysis task. This subset of data is used to create data marts. The data mart ETL is also known as *target ETL* or *ETL2*.

A *data mart* is a subset of the historical data that satisfies the needs of a specific department, team, or customer. A data mart is optimized for interactive reporting and data analysis. The format of a data mart is specific to the reporting or analysis tool you plan to use. Each application that provides a data mart ETL creates its data marts in the appropriate format. Section 2.2.5, “Data mart’s star schema” on page 43 gives more information on typical Tivoli Enterprise Data Warehouse data mart schemas.

Tivoli Enterprise Data Warehouse provides a *Report Interface (RI)* that creates static two-dimensional reports of your data using the data marts. The RI is a role-based Web interface that can be accessed with a simple Web browser without any additional software installed on the client. You can also use other tools to perform OLAP analysis, business intelligence reporting, or data mining.

The *Control server* is the system that contains the control database that contains metadata for Tivoli Enterprise Data Warehouse and from which you manage your data warehouse. The Control server controls communication between the Control server, the central data warehouse, the data marts, and the Report Interface.

The Control server uses the Data Warehouse Center to define the ETL processes and the star schemas used by the data marts. You use the Data Warehouse Center to schedule, maintain, and monitor these processes.

2.1.2 How Tivoli Enterprise Data Warehouse is packaged

When installing Tivoli Enterprise Data Warehouse support for Tivoli software, you receive and install two logical parts:

- The Tivoli Enterprise Data Warehouse core application, which provides the warehouse infrastructure.
- One or more warehouse packs, which are applications that make use of the infrastructure.

Tivoli Enterprise Data Warehouse

The Tivoli Enterprise Data Warehouse core application is packaged as a collection of CDs that are provided with each Tivoli software product that uses its infrastructure. You receive a different set of CDs depending on whether you order support for single byte character set (SBCS) languages or double byte character set (DBCS) languages. The Tivoli Enterprise Data Warehouse CD set consists of the following CDs:

- Tivoli Enterprise Data Warehouse: The installation media for the Tivoli Enterprise Data Warehouse application.
- Tivoli Enterprise Data Warehouse Language Support: The files necessary to use Tivoli Enterprise Data Warehouse in non-English languages. This CD contains both SBCS and DBCS language support.
- Tivoli Enterprise Data Warehouse Documentation: The Tivoli Enterprise Data Warehouse documentation library.
- A collection of DB2 CDs: These vary depending on whether you order the SBCS or DBCS version of Tivoli Enterprise Data Warehouse.

Warehouse packs

A warehouse pack is the part of a Tivoli software product that provides warehouse functionality. It can be provided on the installation media for the product, on a separate CD, or in a collection of warehouse packs. When not provided on a CD containing only one or more warehouse packs, a warehouse pack is located in a subdirectory named tedw_apps_etl.

2.2 Tivoli Enterprise Data Warehouse architecture

The basic components of the Tivoli Enterprise Data Warehouse application have been introduced in the previous sections. We now cover the different architectures of a Tivoli Enterprise Data Warehouse installation, for example, how the components can be reasonably placed on many machines and how they work together.

For architecture considerations there are four components of the Tivoli Enterprise Data Warehouse:

- Control server
- Central data warehouse
- Data marts
- Report Interface

These components can be installed on one system for a single system installation, or distributed across as many as four systems in your IT enterprise. The Tivoli Enterprise Data Warehouse components can be, but do not need to be, installed on the same systems as other Tivoli software or on the systems where the operational data stores reside.

2.2.1 Single machine installation

You can install all components of Tivoli Enterprise Data Warehouse on one single machine. This configuration is easy to set up and to maintain. However, we recommend this configuration only for test or demonstration environments.

The Control server must be on a Windows 2000 or Windows NT machine. Thus the single machine configuration cannot be installed on a UNIX server. You can install all Tivoli Enterprise Data Warehouse components in one step. Before you install Tivoli Enterprise Data Warehouse you have to install IBM DB2 Universal Database Enterprise Edition on this machine first.

2.2.2 Distributed installation

A distributed installation is recommended for most production systems and for customers who already run their database servers on UNIX systems. Each of the above mentioned components of Tivoli Enterprise Data Warehouse can be on a separate machine. Such a configuration is illustrated in Figure 2-2.

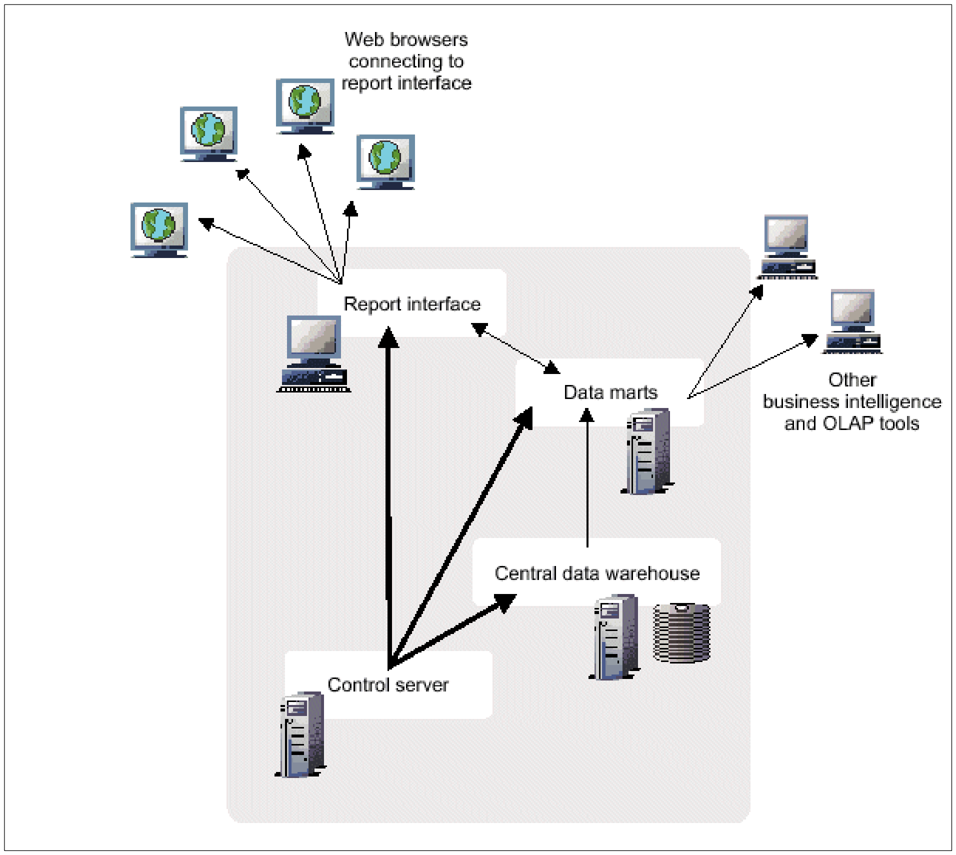

Web browsers connecting to report interface
Report interface
Data marts
Other business intelligence and OLAP tools
Central data warehouse
Control server

Figure 2-2 A distributed Tivoli Data Warehouse configuration

We will provide further information about the four components, including prerequisites like DB2 installations and supported operational systems. However, always first review *Tivoli Enterprise Data Warehouse Release Notes,* GI11-0857, thoroughly before planning your installation.

The Control server is the system that contains the control database for Tivoli Enterprise Data Warehouse and from which you manage your data warehouse. Supported operating systems are Windows NT and Windows 2000.

Before you install the Tivoli Enterprise Data Warehouse component on the Control server you have to install IBM DB2 Universal Database Enterprise Edition on this machine first. The Control server uses the DB2 Server, the Data Warehouse Center, and the warehouse agent.

The Data Warehouse Center on your Control server automates the data warehouse processing. You can use it to define the ETL processes that move and transform data into the central data warehouse and the star schemas used by the data marts. Then you can use the Data Warehouse Center to schedule, maintain, and monitor these processes. The warehouse agent is a part of the DB2 Warehouse Manager. In this configuration, the warehouse agent runs only on the Control server.

The system on which you install the Control server must connect to the operational data stores of your enterprise, which potentially reside on other systems and in relational databases other than DB2. To enable the Control server to access these data sources, you must install the appropriate database client for each data source on the Control server system.

The *central data warehouse server* contains the DB2 databases only. In this configuration no pieces of the Tivoli Enterprise Data Warehouse software or DB2 Warehouse components are needed on this server. Supported operating systems are Windows NT, Windows 2000, AIX, and Solaris.

The same applies to the Data mart server. For this reason, in a typical configuration, the central data warehouse and the data marts will be on one database server.

The *Report Interface server* (or Report server) provides tools and a graphical user interface to create and display reports that help you analyze the data in your warehouse to answer questions that are important to your business. The Report Interface uses Tivoli Presentation Services. If it is already installed in your enterprise, you must install the Report Interface component on the system that hosts the server for IBM Console.

The Report Interface requires a DB2 run-time client to access data in the DB2 instances on the central data warehouse, data mart, and Control servers. You must manually install the IBM DB2 product before installing the Report Interface component. When installing the IBM DB2 product from the CDs provided with Tivoli Enterprise Data Warehouse, any one of the following components is sufficient:

- DB2 Enterprise Edition
- DB2 Application Development Client
- DB2 Administration Client (this component has the smallest footprint)

Supported operating systems for the Report server are Windows NT, Windows 2000, AIX, Solaris, and Linux. It performs best on Windows NT and Windows 2000 systems.

2.2.3 Distributed installation with remote warehouse agents

IBM DB2 Warehouse Manager provides warehouse agents that manage the flow of data between warehouse sources and targets. When you install the Tivoli Enterprise Data Warehouse Control server, a warehouse agent is installed for you. In some environments, it is sufficient to use a warehouse agent only on the Control server. The Control server can use its local warehouse agent to manage data flow for the central data warehouse and the data marts, if these databases are not on the same system is the Control server.

In other environments, you might improve the performance of Tivoli Enterprise Data Warehouse if you place a warehouse agent on the central data warehouse server and Data mart server. The Control server can use these remote warehouse agents to manage data flow. This is an advanced configuration.

Figure 2-3 on page 41 shows an example configuration with remote warehouse agents. In this example the central data warehouse and the data marts are on one server.

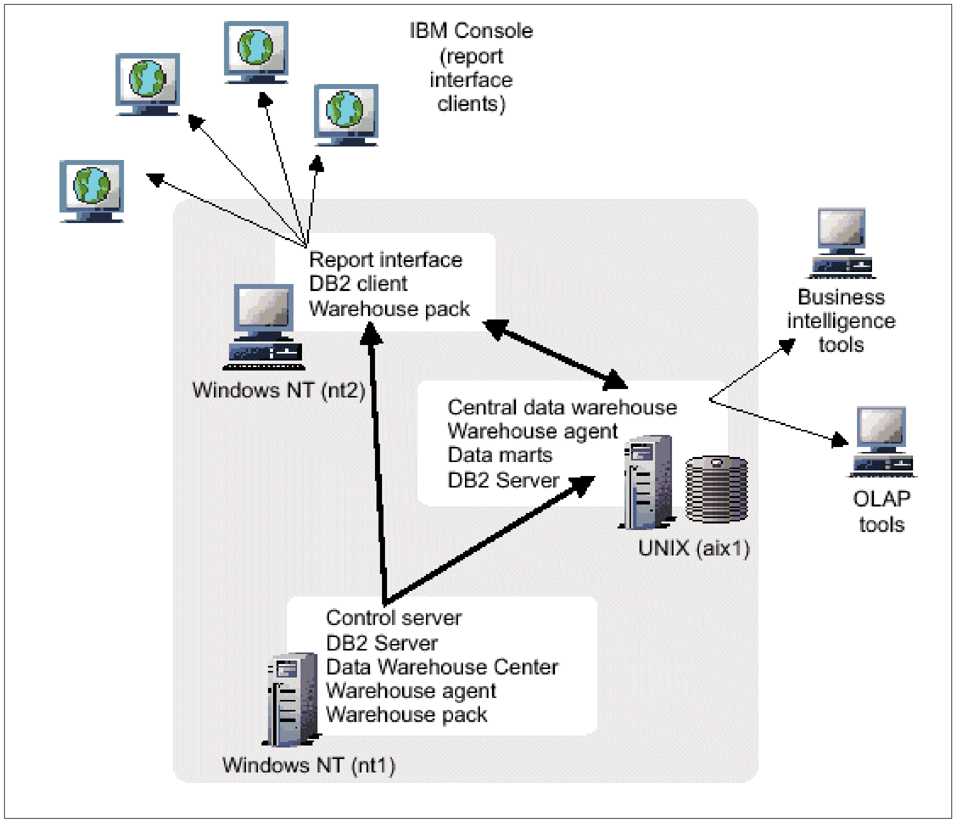


Figure 2-3 Advanced configuration with remote warehouse agents

Creating this advanced configuration requires the following tasks in addition to those you perform for the basic configuration:

- Installing DB2 Warehouse Manager on the systems that become the central data warehouse and Data mart servers
- Installing the central data warehouse and data mart components of Tivoli Enterprise Data Warehouse
- Configuring the warehouse agent daemon on the central data warehouse and Data mart server

An example of the installation of such a configuration is given in Appendix D of the *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744, manual.

2.2.4 Central data warehouse data model

Figure 2-4 shows the central data warehouse data model used by Tivoli Enterprise Data Warehouse.

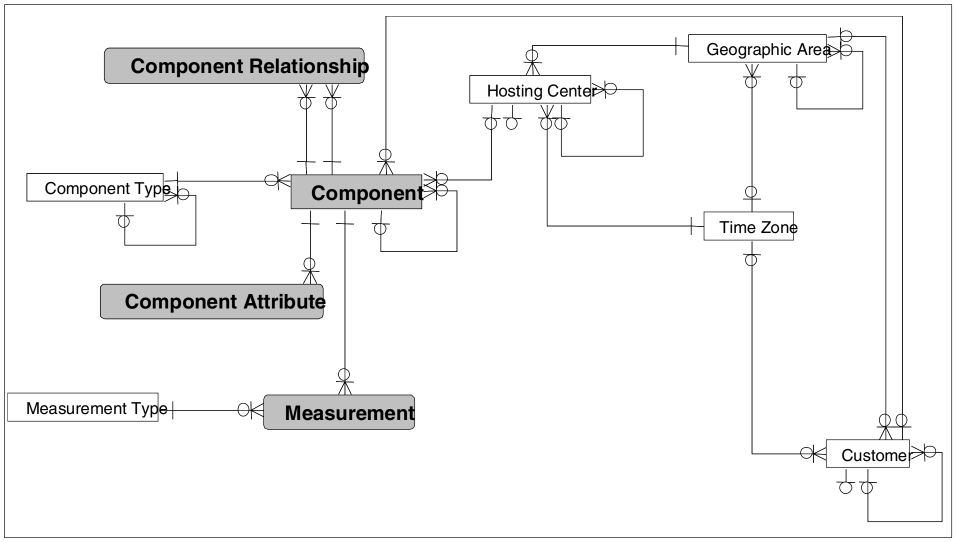


Figure 2-4 Central data warehouse data model

According to this model, each hosting center provides hosting services for a geographic area. To provide these services to a set of customers, the hosting center manages many components. A component may be a shared infrastructure component such as a switch or firewall, or it may be a private component associated with a single customer, such as a Web site or Web server. Each component is classified by a component type. Depending on the component type, a component may contain sub-components. For example, a server contains CPUs, memory, and disks. These sub-components are also components, and so are similarly classified by a component type and may contain further sub-components. For each component type, there is a set of appropriate measurement types (for example, CPU utilization, CPU load, or Web server requests). The mapping between component types and measurement types is enforced as a component measurement rule. Monitoring tools collect raw measurements at specified intervals for each component. These raw measurements are retrieved and summarized as measurements in the cdw according to the component measurement rules. Each summarized measurement identifies:

- The component (for example, a Web server)
- The measurement type (for example, CPU utilization)

- The date and time interval (for example, 12th hour of September 16, 2000)
- The minimum, maximum, average, and total value for that hour (for example, 1, 13, 4.5, 248)
- The sample count, or number of raw measurements that were summarized (for example, 55)

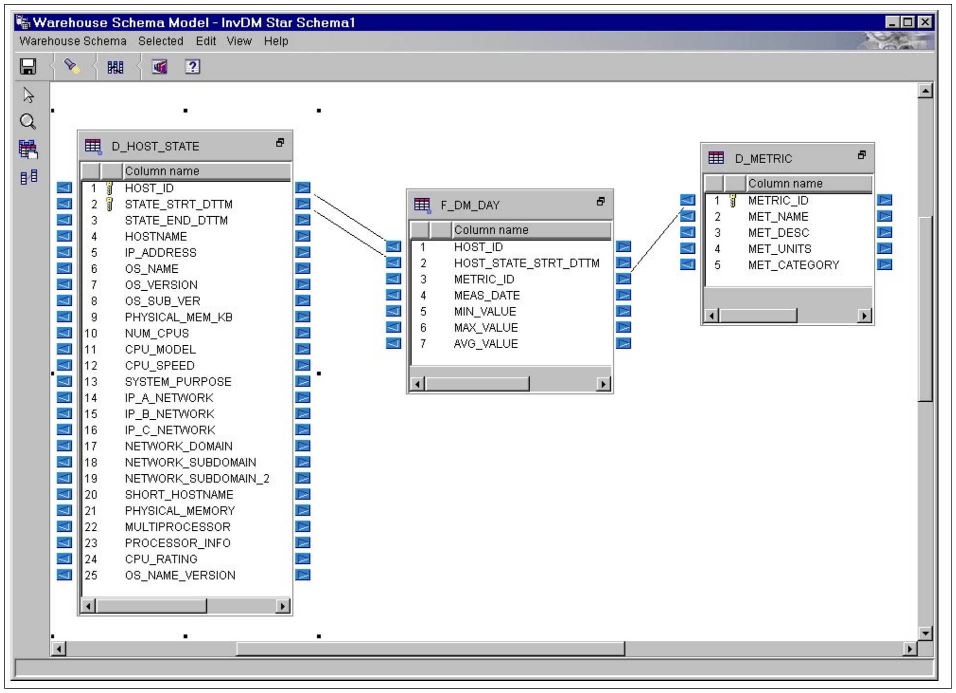
These measurements are actual readings. This generic approach was chosen to ensure flexibility and extensibility for the CDW. It is independent of the underlying monitoring tools being used as external data sources. This allows tools to be added or replaced without structural changes to the CDW. This is accomplished by making component types and rules configurable, so that new components, attributes, relationships and hierarchies may be introduced. Measurement types and rules are also configurable, so that new types may be loaded, again without structural changes.

The value of the central repository is that it contains an integrated view of the raw materials for the CDW and provides a safe data store over time. It can be tuned to manage the specific problems of loading large volumes of source data, and can defer the problems of changing end-user requirements and unpredictable access workloads to data mart design.

2.2.5 Data mart's star schema

The inherent flexibility of the CDW makes it cumbersome to build queries spanning different components and types of measurements. The purpose of the specialized data marts is to transform raw business data into business information that can be easily reported and analyzed. Each data mart targets a specific business audience and a particular data analysis problem domain.

Figure 2-5 on page 44 shows an example Tivoli Enterprise Data Warehouse data mart.


Warehouse Schema Model - InvDM Star Schema1
Warehouse Schema Selected Edit View Help

D_HOST_STATE

	Column name
1	HOST_ID
2	STATE_STRT_DTTM
3	STATE_END_DTTM
4	HOSTNAME
5	IP_ADDRESS
6	OS_NAME
7	OS_VERSION
8	OS_SUB_VER
9	PHYSICAL_MEM_KB
10	NUM_CPUS
11	CPU_MODEL
12	CPU_SPEED
13	SYSTEM_PURPOSE
14	IP_A_NETWORK
15	IP_B_NETWORK
16	IP_C_NETWORK
17	NETWORK_DOMAIN
18	NETWORK_SUBDOMAIN
19	NETWORK_SUBDOMAIN_2
20	SHORT_HOSTNAME
21	PHYSICAL_MEMORY
22	MULTIPROCESSOR
23	PROCESSOR_INFO
24	CPU_RATING
25	OS_NAME_VERSION

F_DM_DAY

	Column name
1	HOST_ID
2	HOST_STATE_STRT_DTTM
3	METRIC_ID
4	MEAS_DATE
5	MIN_VALUE
6	MAX_VALUE
7	AVG_VALUE

D_METRIC

	Column name
1	METRIC_ID
2	MET_NAME
3	MET_DESC
4	MET_UNITS
5	MET_CATEGORY

Figure 2-5 Data mart's star schema

Some examples are:

- Single customer analysis for performance engineers
- Infrastructure analysis for network analysts
- Summarized, overall customer health for server farm management.

Data marts are designed to be customizable by the customer. Tivoli Enterprise Data Warehouse supplies templates for both data marts and the ETL to build them as examples of how a typical data mart could be built. These templates can be used as a starting place for applications developing for Tivoli Enterprise Data Warehouse.

Access is protected in the following ways through the Tivoli Warehouse Report Interface (RI):

- Customer-dedicated data marts
- Multi-customer data marts with controlled access through database views
- User and group access mechanisms proprietary in the Report Interface

2.3 Applications with Tivoli Enterprise Data Warehouse

The Tivoli Enterprise Data Warehouse core installation provides a framework for enterprise-wide data warehousing. Many system management applications, including forthcoming releases of Tivoli products, will provide their own warehouse packs, which implement the integration of the product into the Tivoli Enterprise Data Warehouse.

The central data warehouse and other parts of Tivoli Enterprise Data Warehouse, for example, the ETLs, are open and documented, as well as the way they are managed in the Data Warehouse Center. This enables you to change a product's warehouse integration according to your needs or to implement the Tivoli Enterprise Data Warehouse integration of your own product.

We discuss examples of the application integration in Chapter 5, "Integration of application data to central data repository" on page 127 and Chapter 6, "How to create data marts" on page 179. In the manual *Enabling an Application for Tivoli Enterprise Data Warehouse*, GC32-0745*,* you find an explanation of the infrastructure of all Tivoli Enterprise Data Warehouse components and how to integrate your own components to Tivoli Enterprise Data Warehouse. Furthermore, naming conventions and useful tips are given.

In principle, there are two levels of Tivoli Enterprise Data Warehouse integration. Level one is to load your operational data from the applications point of storage to the central data warehouse. You can access your data directly from the central data warehouse or you can change the data marts and reports of other applications to use the data of your new application, as well.

A second level of integration is to create separate data marts for the application, provide mechanisms to populate the data mart from the central data warehouse (possibly also using data from other applications) and to create reports for the applications using the Tivoli Enterprise Data Warehouse Report Interface or other OLAP tools. You can also provide rollup mechanisms in the data marts, which populate daily and weekly star schemas from your hourly data.

Finally you can pack all these components to warehouse packs. You can ship the warehouse packs together with your application to your customers or colleagues. They can install the warehouse pack simply using the Tivoli Enterprise Data Warehouse installation wizard.

3

Installation and configuration

This chapter provides different types of Tivoli Enterprise Data Warehouse installations (stand-alone and distributed) and give recommendations for each configuration.

This chapter has the following sections:

- Hardware and software requirements
- Planning for Tivoli Enterprise Data Warehouse
- Different types of installations
- Basic customizing

3.1 Planning for Tivoli Enterprise Data Warehouse

When installing Tivoli Enterprise Data Warehouse support for Tivoli software, you receive and install two logical parts:

- The Tivoli Enterprise Data Warehouse core application, which provides the warehouse infrastructure
- One or more warehouse packs, which are applications that make use of the infrastructure

We will give you the two scenario's installations of the Tivoli Enterprise Data Warehouse core application in Section 3.3, "Stand-alone Tivoli Enterprise Data Warehouse" on page 59 and Section 3.4, "Distributed Tivoli Enterprise Data Warehouse" on page 67. The warehouse packs will be provided with the Tivoli application and each warehouse pack will have its own installation instructions.

3.1.1 Selecting port numbers

You must allocate port numbers for Tivoli Enterprise Data Warehouse for the following purposes:

- Communication between the Control server and remote DB2 installations
- Communications for the IBM Console

Note: You must specify unused port numbers when you install Tivoli Enterprise Data Warehouse. Specifying port numbers that are already in use by other programs causes the installation to hang. In particular, if there is already a Web server on the system where you plan to install the Report server, you must un-install or disable that Web server, or specify a different port number for the HTTP Server Port for Tivoli Presentation Services.

To determine which port numbers are in use on a particular computer, type either of the following commands from a command prompt:

- **`netstat -a`**
- **`netstat -an`**

Check the results of these commands to see if their are any conflicts with the default Tivoli Enterprise Data Warehouse port (see Table 3-1 on page 49) and correct the problem.

If there are multiple DB2 instances in your DB2 installation, make sure that the user name you specify is the owner of the instance whose DB2 port you specify. A mismatch between user name and port number results in installation errors that are difficult to identify.

To determine the port number for your instance, use the following technique: On the DB2 server edit the services file (/etc/services on UNIX systems and C:\WINNT\system32\drivers\etc\services on Windows 2000 systems) and locate the line that defines the connection port for the instance name. For example, if the instance name is db2, look for a line similar to this in the services file:

```
db2cDB2        50000/tcp           #connection port for the DB2 instance DB2
```

Tip: You can use the following DB2 command to check the instance configuration to verify the port number:

```
db2 get dbm cfg :grep SV
```

Which should give:

```
TCP/IP Service name <SVCENAME> = 50000
```

This should match the port number in the services file.

If the port is different from the default Tivoli Enterprise Data Warehouse port (see Table 3-1) then the default port needs to be changed.

Table 3-1 Default port numbers used by Tivoli Enterprise Data Warehouse

For the Control server component			
Default port number	**Name of port in InstallShield Wizard**	**Description**	**Can this default be changed?**
50000	Database port (Control server)	In a distributed installation used by the data mart server, central data warehouse server, and report server to communicate with the Control server. (The same port number can be used for the Control server, Data mart server, and central data warehouse server.)	Yes, you can change it when you install or configure DB2. Refer to the DB2 documentation for details.

For the Control server component			
Default port number	**Name of port in InstallShield Wizard**	**Description**	**Can this default be changed?**
50000	Database port (central data warehouse)	In a distributed installation used by the Control server, Data mart server, and report server to communicate with the central data warehouse server. (The same port number can be used by the Control server, Data mart server, and central data warehouse server.)	Yes, you can change it when you install or configure DB2. Refer to the DB2 documentation for details.
50000	Database port (data mart)	In a distributed installation used by the Control server, central data warehouse server, and report server to communicate with the Data mart server. (The same port number can be used by the Control server, Data mart server, and central data warehouse).	Yes, you can change it when you install or configure DB2. Refer to the DB2 documentation for details.
For the IBM Console and Tivoli Presentation Services			
80	IBM HTTP Server Port	Used by Tivoli Presentation Services HTTP Server for HTTP communications.	Yes, you can change it during the installation of the Report Interface.
8008	IBM HTTP Administration Port	Used by Tivoli Presentation Services HTTP Administration.	Yes, you can change it during the installation of the Report Interface.
8010	Server For IBM Console IPC Port	Used by Server for IBM Console (part of Tivoli Presentation Services).	Yes, you can change it during the installation of the Report Interface.
8007	Web Console Port	Used by Web Services for the IBM Console (part of Tivoli Presentation Services).	Yes, you can change it during the installation of the Report Interface.
8040	Web Console IPC Port	Used by Web Services for the IBM Console (part of Tivoli Presentation Services).	Yes, you can change it during the installation of the Report Interface.
8050	Not applicable	Used by Server for IBM Console the IBM Console.	No.

3.1.2 Other network checks

We recommend that you use a static IP address for each server. You might have problems if you use DHCP addressing for Tivoli Enterprise Data Warehouse servers.

Your operating system must be configured to provide Tivoli Enterprise Data Warehouse and Tivoli Presentation Services with a fully qualified computer name rather than a short name. This is especially important in environments with many different operating systems. To ensure that a system is configured to provide a fully qualified computer name, follow the instructions in the following sections.

On AIX systems

The default domain name search order is as follows:

1. Domain name system (DNS) server
2. Network Information Service (NIS)
3. Local /etc/hosts file

If the /etc/resolv.conf file does not exist, the /etc/hosts file is used. If only the/etc/hosts file is used, the fully qualified computer name must be the first one that is listed after the IP address. Verify that the /etc/resolv.conf file exists and contains the appropriate information, such as:

```
domain mydivision.mycompany.com
nameserver 123.123.123.123
```

If NIS is installed, the /etc/irs.conf file overrides the system default. It contains the following information:

```
hosts =bind,local
```

The /etc/netsvc.conf file, if it exists, overrides the /etc/irs.conf file and the system default. It contains the following information:

```
hosts =bind,local
```

If the NSORDER environment variable is set, it overrides all of the preceding files. It contains the following information:

```
export NSORDER=bind,local
```

On Linux systems

Verify that the /etc/resolv.conf file exists and contains the appropriate information, such as:

```
domain mydivision.mycompany.com
nameserver 123.123.123.123
```

A short name is used if the /etc/nsswitch.conf file contains a line that begins as follows and if the /etc/hosts file contains the short name for the computer:

```
hosts:files
```

To correct this, follow these steps:

1. Change the line in the /etc/nsswitch.conf file to:

   ```
   hosts:dns nis files
   ```

2. Stop the network service.
3. Restart the network service.

On Solaris systems

Verify that the /etc/resolv.conf file exists and contains the appropriate information, such as:

```
domain mydivision.mycompany.com
nameserver 123.123.123.123
```

A short name is used if the /etc/nsswitch.conf file contains a line that begins as follows and if the /etc/hosts file contains the short name for the computer:

```
hosts:files
```

To correct this, follow these steps:

1. Change the line in the /etc/nsswitch.conf file to:

   ```
   hosts:dns nis files
   ```

2. Enter the following command to stop the inet service:

   ```
   /etc/init.d/inetsvc stop
   ```

3. Enter the following command to restart the inet service:

   ```
   /etc/init.d/inetsvc start
   ```

On Windows 2000 systems

To verify that a primary domain name system suffix is set, follow these steps:

1. On the desktop, right-click **My Computer**.
2. Click the **Network Identification** tab.
3. Ensure that the field Full Computer Name contains a fully qualified domain name. If it does not, follow these steps:
   a. Click **Properties**.
   b. Click **More**.

c. In the field Primary DNS suffix for this computer, type the primary DNS suffix, and restart the computer when prompted.

On Microsoft Windows NT systems

To verify that a primary domain name system suffix is set, follow these steps:

1. From the Windows taskbar, click **Start -> Settings -> Control Panel**.
2. In the Control Panel window, double-click **Network.**
3. Click the **Protocols** tab.
4. Select the TCP/IP protocol and then click **Properties**.
5. Click the **DNS** tab.
6. Ensure that the field Domain contains a domain suffix. If it does not, type the suffix, click **OK**, and restart the computer when prompted.

3.2 Hardware and software requirements

You can deploy Tivoli Enterprise Data Warehouse one of these ways:

- As a single system installation, with all the components installed on a single Microsoft Windows NT or Microsoft Windows 2000 system. This is convenient for demonstrations, as an educational or test platform, and for companies that do not plan to have many users concurrently analyzing data and that do not need to capture and analyze large amounts of data in the warehouse.
- As a distributed installation, with the components installed on multiple systems in your enterprise, including UNIX servers. See "Software requirements" on page 54 to determine the operating systems on which you can place each component of Tivoli Enterprise.

3.2.1 Hardware requirements

This section provides information about the hardware requirements for installing Tivoli Enterprise Data Warehouse. As the warehouse enablement pack for each Tivoli software product is added to the Tivoli Enterprise Data Warehouse installation, additional hard disk space is required. See the documentation for each warehouse pack for application planning information and hard disk space requirements.

Table 3-2 Hardware requirements

Installation configuration	**Tivoli Enterprise Data Warehouse core components**	**Minimum requirements**	**Recommended size**	**Temp hard disk space**
Stand-alone	All	522 MB RAM 933 MHz processor 1 GB hard disk space	1 GB RAM 3 GB hard disk space	1 GB
Distributed	Control server	512 MB RAM 800 MB hard disk space 933 MHz processor	1 GB RAM 1 GB hard disk space	1 GB
	Report server	512 MB RAM 800 MB hard disk space 933 MHz processor	1 GB RAM 1.5 GB hard disk space	1 GB
	Central data warehouse	512 MB RAM 800 MB hard disk space 933 MHz processor	1 GB RAM 20 GB hard disk space	1 GB
	Data mart	512 MB RAM 800 MB hard disk space 933 MHz processor	1 GB RAM 3 GB hard disk space	1 GB

3.2.2 Software requirements

This section provides information about the software requirements for the Tivoli Enterprise Data Warehouse.

Note: You might receive confusing error messages if your systems do not meet the software requirements listed in this section.

Table 3-3 Software requirements

	Tivoli Enterprise Data Warehouse core components					
Operating system	**Data source**	**Warehouse agents**	**Control server**	**Central data warehouse**	**Data mart database**	**Report server**
Microsoft Windows NT® Service Pack 6 or higher, Windows 2000 Server, Windows 2000 Advanced Server	Yes	Yes	Yes	Yes	Yes	Yes
IBM AIX Versions 5.1 or 4.3.3 with FixPak 2 or higher	Yes	Yes	No	Yes	Yes	Yes
Sun Solaris Versions 2.7 and 2.8	Yes	Yes	No	Yes	Yes	Yes
RedHat Linux Version 7.1	Yes	No	No	No	No	Yes
SuSE Linux Version 7.2	Yes	No	No	No	No	Yes

3.2.3 Database requirements

Tivoli Enterprise Data Warehouse requires DB2 Version 7 Release 2 with FixPak 5. If you are currently running a DB2 version prior to 7.2, you must upgrade to Version 7.2 with FixPak 5 using the DB2 CDs provided with Tivoli Enterprise Data Warehouse. On the Control server, you must additionally apply the emergency fixes (e-fixes) for APARs JR16650 and JR16766. If you are an application developer creating a warehouse pack, you must also apply these APARs on the system from which you export the tag files for your application. The e-fixes are available on the following Tivoli Enterprise Data Warehouse support web site:

http://www.ibm.com/software/sysmgmt/products/support

The recommended hard disk space in Table 3-2 on page 54 is large enough to accommodate some data growth as transactions are added to the database. However, when you plan for your database requirements, you must consider the following:

- Future data growth
- Addition of warehouse packs
- Customizing reports
- Saving report output

It is recommended that you install your central data warehouse on an expandable system with a minimum of 20 GB of data space. Refer to the DB2 library for database recommendations.

The Tivoli Enterprise Data Warehouse components IBM DB2 Warehouse Manager, central data warehouse and data marts are currently only supported on DB2 Version 7.2 with FixPak 5. For the DB2 installation on the Control server, you have to apply e-fixes as well.

The data source support is determined by Open Database Connectivity (ODBC) support in DB2 releases. The following is a list of source databases that are supported:

- DB2 Version 6, DB2 Version 7
- Oracle Version 8.1.7 on Windows NT, Windows 2000, AIX, and Solaris
- MS SQL Server Version 7.0 on Windows NT
- MS SQL Server Version 2000 on Windows 2000
- Sybase Version 11.5 on Windows NT
- Sybase Version 11.9.2 on AIX and Solaris
- Informix Version 7.2.2 - V9.0 on Windows NT
- Informix Version 7.2.4 - V9.2.0 on AIX and Solaris

Note: The actual RDBMSs supported as data sources for a given warehouse pack are documented in the readme file for that warehouse pack.

3.2.4 Web browser requirements

You can configure Tivoli Presentation Services to use the Secure Sockets Layer (SSL).

Note: Using SSL requires a Web browser with 128-bit support.

Each user is responsible for installing the correct version of the browser on their workstation. The Tivoli Enterprise Data Warehouse installation program does not install the Web browser.

The supported Web browsers are listed below:

- Internet Explorer Version 5.5 and Version 6
- Netscape Navigator Version 4.62 and Version 4.71

Tips:

- JavaScript and style sheets must be enabled in these browsers.
- When using the Report Interface, we found the performance of Internet Explorer slightly better than Netscape.

3.2.5 Report Interface requirements

The Report Interface uses the IBM Console, which is implemented using Tivoli Presentation Services. The following operating systems are supported:

- Windows systems
  - Windows NT 4.0 Server with Service Pack 6
  - Windows 2000 Server
  - Windows 2000 Advanced Server

  The Java version of the IBM Console can also run on Windows 2000 Professional, but the server and Web Services for the IBM Console cannot.
- UNIX-based systems
  - AIX 4.3.3.10 and 5.1 with the required operating system patches for Java Runtime Environment 1.3. See Section 3.2.6, “AIX system requirements” on page 57 for more information.
  - Red Hat Linux 7.1
  - Solaris 7 and 8 with the required operating system patches for Java Runtime Environment 1.3. See Section 3.2.7, “Solaris system requirements” on page 58 for more information.
  - SuSE Linux 7.1
  - UNIX-based GUIs for X-Window environment
    - For AIX and Solaris systems, the Common Desktop Environment (CDE)
    - For Red Hat Linux and SuSE Linux systems, the K Desktop Environment (KDE), and the GNU Network Object Model Environment (GNOME)

3.2.6 AIX system requirements

Tivoli Presentation Services includes the IBM AIX Java Runtime Environment Version 1.3. This JRE requires AIX 4.3.3.10 or AIX 5.1.

3.2.7 Solaris system requirements

Tivoli Presentation Services includes the Java 2 Platform, Standard Edition (J2SE) and the Java Runtime Environment (JRE). This JRE runs on Solaris 7 or 8 with the required and recommended patches listed in Table 3-4 and Table 3-5 on page 59.

To determine which patches are already installed on your system, use the following shell command:

```
showrev -p
```

In addition to verifying that your system includes the correct patches, you might want to install the latest patch cluster for your version of the Solaris system. Patch clusters include additional recommended and security patches. You can obtain the patches and patch clusters from your service provider, or you can download them individually from the SunSolve Web site at:

http://sunsolve.sun.com

Use the search function on the SunSolve site to search for the patch number. The J2SE download site includes download tar bundles that contain the patches and includes the latest information about recommended and required patches for the JRE. For more information, refer to the following Web site:

http://java.sun.com/j2se/1.3/install-solaris-patches.html

In Table 3-4 and Table 3-5 on page 59, the two-digit number following the hyphen in each patch ID is the revision number for that patch. Although the tables list the revisions with which this release of J2SE was tested, later patch revisions should work as well.

Table 3-4 Patches for Solaris 8

Solaris-SPARC patch ID	Description	Required
108940-12 or later	Motif 2.1 patch	Yes
108921-07 or later	For CDE window manager	Recommended for CDE users

Table 3-5 Patches for Solaris 7

Solaris-SPARC patch ID	Description	Required
107226-12 or later	For CDE window manager	Recommended for CDE users
106980-12 or later	Libthreads patch	Yes
107153-01 or later	Required for zh.GBK Chinese locale	Recommended
107636-06 or later	Implementation of composition enabling/disabling API for X input methods	Recommended
107544-03 or later	SunOS 2.7 Kernel update	Yes
106541-12 or later	SunOS 2.7 Kernel update	Yes
109104-04 or later	SunOS 2.7 Kernel update	Yes
108376-16 or later	OpenWindows 3.6.1 Xsun patch. Note: Some later versions of this patch, including 108377-10, cause an X server crash	Yes
106950-13 or later	Linker patch	Yes
107081-25 or later	Motif 1.2, Motif 2.1, and runtime library patch	Yes
106300-09 or later, and 106327-08 or later	Shared library patch for C++	Yes

Note: For the latest hardware and software requirements refer to the *Tivoli Enterprise Data Warehouse Release Notes,* GI11-0857.

3.3 Stand-alone Tivoli Enterprise Data Warehouse

In this section we set up all the Tivoli Enterprise Data Warehouse core components on one machine.

Our environment consists of a single PC server.

- Hardware
  - 1 GB RAM
  - 933 MHz 4 processors
  - 4 GB hard disk space
- Software
  - Windows 2000 server with Service Pack 2
  - Internet Explorer Version 5.5

Tip: If you are planning to install all Tivoli Enterprise Data Warehouse core components on one machine, we recommend that you use at least a 512 MB RAM machine for decent performance.

3.3.1 Windows DB2 Universal Database installation

Before installing DB2 Universal Database for Tivoli Enterprise Data Warehouse, you need to do the following things:

- Read the DB2 Quick Beginnings document for the operating system on which you are installing (for example, *DB2 UDB Quick Beginnings for Windows Version 7*, GC09-2971, or *DB2 UDB Quick Beginnings for UNIX,* GC09-2970).
- Use the DB2 installation media provided with Tivoli Enterprise Data Warehouse. This ensures that you get the correct version and FixPaks of the DB2 Server.

We used the installation wizard as an easy way to install IBM DB2 Universal Database Enterprise Edition. See the steps below:

1. Load the Tivoli Enterprise Data Warehouse DB2 installation media.
2. We used the options shown in Table 3-6 in the wizard windows.

Table 3-6 DB2 Wizard options

Wizard window	Option selected
Select the product(s) you would like to install.	DB2 Enterprise Edition
Select the installation type you prefer.	Typical
Destination folder.	C:\SQLLIB
DB2 user and password.	db2admin

Wizard window	Option selected
Install OLAP starter kit	Do not install OLAP start kit

   This will install DB2 and FixPak 5. In addition to this, we need to install the e-fix. The e-fix consists of two files: iwh2exp2.exe and iwh2imp2.exe. We need to copy these files over the ones installed by the DB2 installation as follows:

   a. Rename D:\sqllib\bin\iwh2exp2.exe to D:\sqllib\bin\iwh2exp2.exe.old.
   b. Rename D:\sqllib\bin\iwh2imp2.exe to D:\sqllib\bin\iwh2imp2.exe.old.
   c. xcopy iwh2exp2 D:\sqllib\bin.
   d. xcopy iwh2imp2 D:\sqllib\bin.

3. Reboot the machine.

3.3.2 Tivoli Enterprise Data Warehouse installation

When the Tivoli Enterprise Data Warehouse installation wizard runs the installation it will install the three databases in the default directory. If you want these databases to be installed in another directory, then create these databases before you start the Tivoli Enterprise Data Warehouse installation wizard by doing the following:

Open a DB2 command window and issue these commands:

```
db2 create db twh_md on d
db2 create db twh_mart on d
db2 create db twh_cwd on d
```

Where d is the drive you would like the database installed on.

After installing DB2, perform a connection test by completing these steps:

1. Enter the following command in a DB2 command window to list the local databases:

   ```
   db2 list database directory
   ```

   In most cases, the command lists at least one database, even in new installations.

2. Enter the following command to test a local connection:

   ```
   db2 connect to database_name user user_name using password
   ```

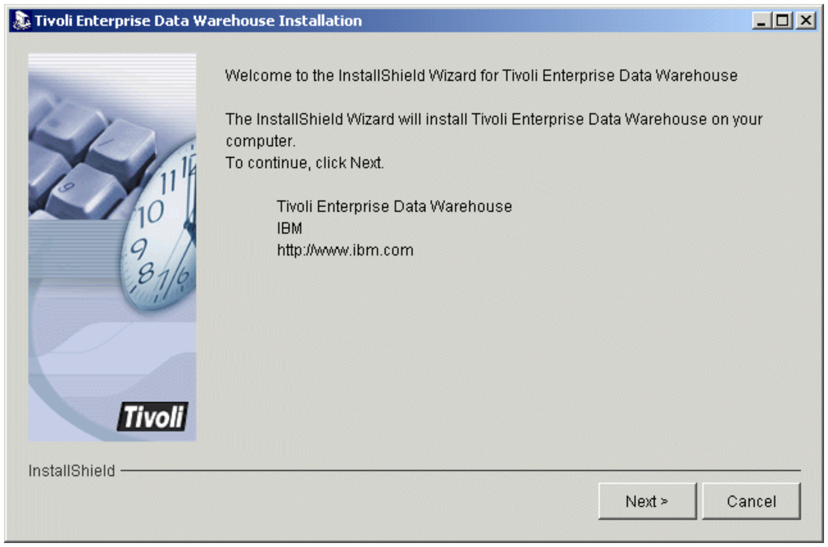
Where database_name is a database name returned in the first step, user_name and password are the user name and password you specified when installing DB2. If this command is successful, DB2 is installed and ready for remote connections. If you experience difficulty, refer to the DB2 documentation for troubleshooting information.

To install the Tivoli Enterprise Data Warehouse components complete the following:

1. Insert the Tivoli Enterprise Data Warehouse installation media into the machine’s CD-ROM drive and run the following command:

   ```
   setup.exe
   ```

   The Tivoli Enterprise Data Warehouse installation wizard will be started (Figure 3-1).


Tivoli Enterprise Data Warehouse Installation

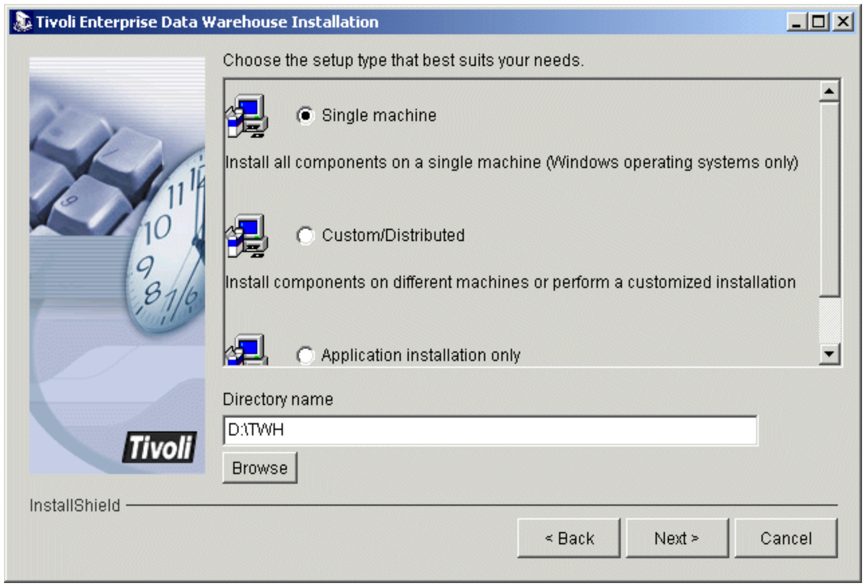
Welcome to the InstallShield Wizard for Tivoli Enterprise Data Warehouse

The InstallShield Wizard will install Tivoli Enterprise Data Warehouse on your computer.
To continue, click Next.

Tivoli Enterprise Data Warehouse
IBM
http://www.ibm.com

Figure 3-1 Welcome dialog window

2. Choose **Next** and the dialog window for installation type will appear (Figure 3-2 on page 63).


Tivoli Enterprise Data Warehouse Installation

Choose the setup type that best suits your needs.

(● Single machine)

Install all components on a single machine (Windows operating systems only)

○ Custom/Distributed

Install components on different machines or perform a customized installation

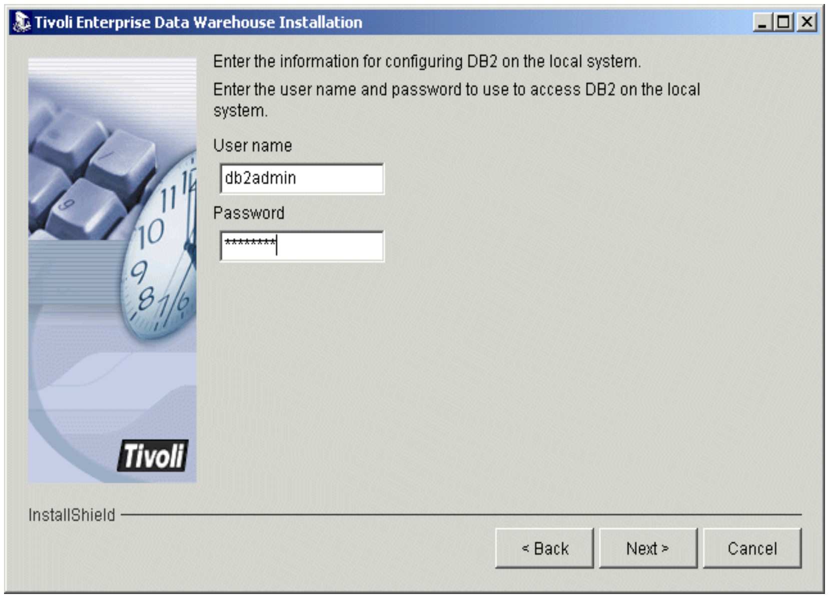
○ Application installation only

Directory name

D:\TWH

Figure 3-2 Install type dialog window

3. Select **Single machine** and insert your install directory name, then choose **Next**. The verify hostname dialog window will appear.
4. Confirm that this is the correct host name and choose **Next**. The DB2 configuration dialog window will appear (Figure 3-3 on page 64).


Tivoli Enterprise Data Warehouse Installation

Enter the information for configuring DB2 on the local system.

Enter the user name and password to use to access DB2 on the local system.

User name

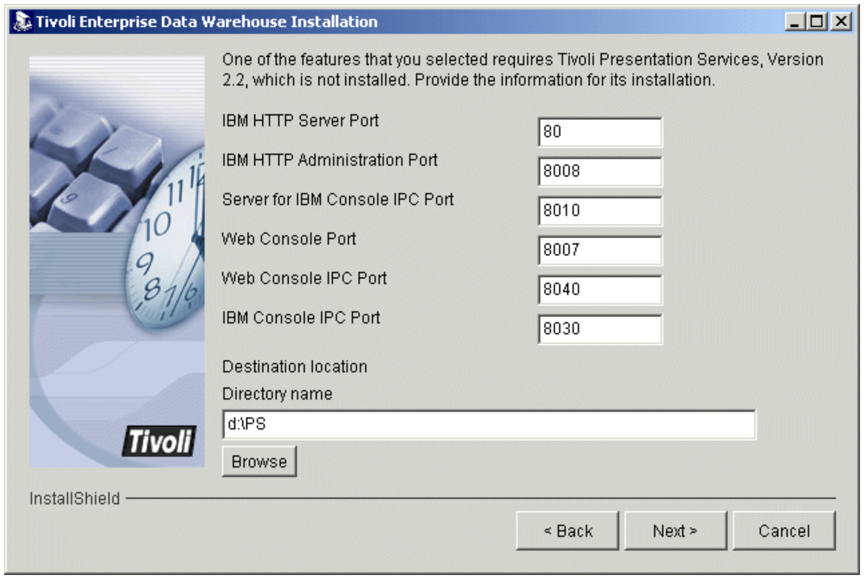
db2admin

Password

****

Figure 3-3 DB2 configuration dialog window

5. Enter the DB2 user ID and password created during the DB2 installation and choose **Next**. The Tivoli Presentation Services dialog will appear (Figure 3-4 on page 65).


Tivoli Enterprise Data Warehouse Installation

One of the features that you selected requires Tivoli Presentation Services, Version 2.2, which is not installed. Provide the information for its installation.

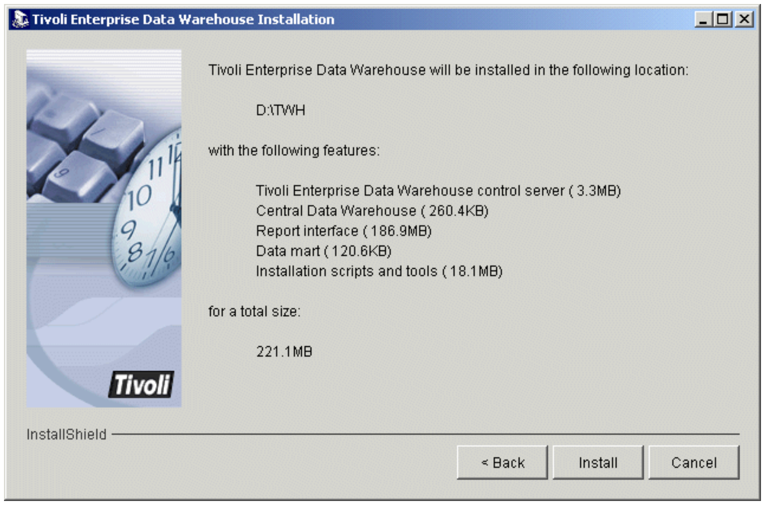
IBM HTTP Server Port	80
IBM HTTP Administration Port	8008
Server for IBM Console IPC Port	8010
Web Console Port	8007
Web Console IPC Port	8040
IBM Console IPC Port	8030

Destination location

Directory name: d:\PS

Figure 3-4 Default port setting dialog window

6. If any of these ports are being used by another application, change the default ports now (see Section 3.1.1, “Selecting port numbers” on page 48) and choose **Next**. The Install languages dialog window will appear.
7. Leave this box unchecked and choose **Next**. The install applications dialog window will appear.
8. Leave this box unchecked and choose **Next**. The Overview of selected options dialog window will appear (Figure 3-5 on page 66).



Tivoli Enterprise Data Warehouse will be installed in the following location:

D:\TWH

with the following features:

Tivoli Enterprise Data Warehouse control server (3.3MB)
Central Data Warehouse (260.4KB)
Report interface (186.9MB)
Data mart (120.6KB)
Installation scripts and tools (18.1MB)

for a total size:

221.1MB

Figure 3-5 Selected options dialog window

9. Choose **Install** to start the installation.
10. Verify that the installation has been successful by doing the following:
    - Make sure the installation wizard's completion window does not list any errors. See the "Messages" section in *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744, for more information.
    - If the completion window lists warnings, check the TWH.log to ensure that the warnings can safely be ignored.
11. Wait for the help set to be rebuilt. The help set contains the user assistance for the IBM Console. This process happens asynchronously and might not complete by the time the wizard has completed the installation of the remaining components of Tivoli Enterprise Data Warehouse. Do not restart the system until the help set is complete. If you do, follow the procedure documented in the *Tivoli Enterprise Data Warehouse Release Notes,* GI11-0857, to rebuild the help set. To determine whether the help set rebuild is complete, look for the completion message in the Tivoli Presentation Services installation log. This log is in directory PS_directory/log/fwp_mcr, where PS_directory is the target directory you specified for Tivoli Presentation Services. The logs are named stdoutn. Look for the message shown in Example 3-1 on page 67 in the most recent stdoutn file. In some cases, the message can be in the second most recent stdoutn file.

Example 3-1 Successful completion message

```
FWP1734I The utility that was started by the Management Component Repository to
build the help set has completed successfully.
```

This process can take up to 30 or 40 minutes.

12. Reboot the machine.

The Tivoli Enterprise Data Warehouse core installation is now complete, but further steps need to be taken to make it ready for use (see "Tivoli Enterprise Data Warehouse configuration" on page 76). If errors were detected during installation or an un-install is required, see "Troubleshooting and maintenance" on page 283.

3.4 Distributed Tivoli Enterprise Data Warehouse

In this section we install all the Tivoli Enterprise Data Warehouse components on two machines. Our environment consists of a PC server and RS6000.

- PC server: (itsotiv-a)
  - Hardware
    - 512 RAM
    - 933 MHz 1processor
    - 5 GB hard disk space
  - Software
    - Windows 2000 server with Service Pack 1
    - Internet Explorer Version 5.5
- RS6000: (itsotiv-b)
  - Hardware
    - Model 7043-150
    - 256 MB RAM
  - Software
    - AIX Version 4.3.3 with FixPak 9

The Control server and Report server components will be installed on the PC server, and the central data warehouse and data mart will be installed on the RS6000.

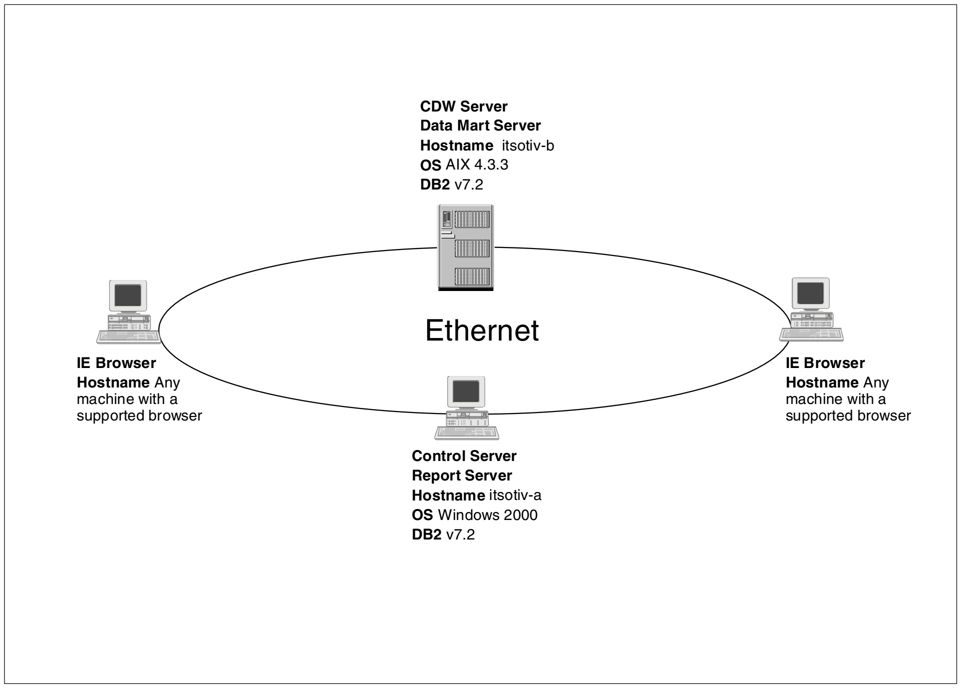


Figure 3-6 Our environment

3.4.1 DB2 Universal Database installation

We install the DB2 Server on the systems that are to host the Control server, central data warehouse server, and the Data mart server. We install a DB2 Server on the system that is to be the Report server only because it is on the same server as the Control server. Do this before installing the Tivoli Enterprise Data Warehouse Control server because the Control server must be able to connect to, create, and save data in DB2 databases on those systems during its installation.

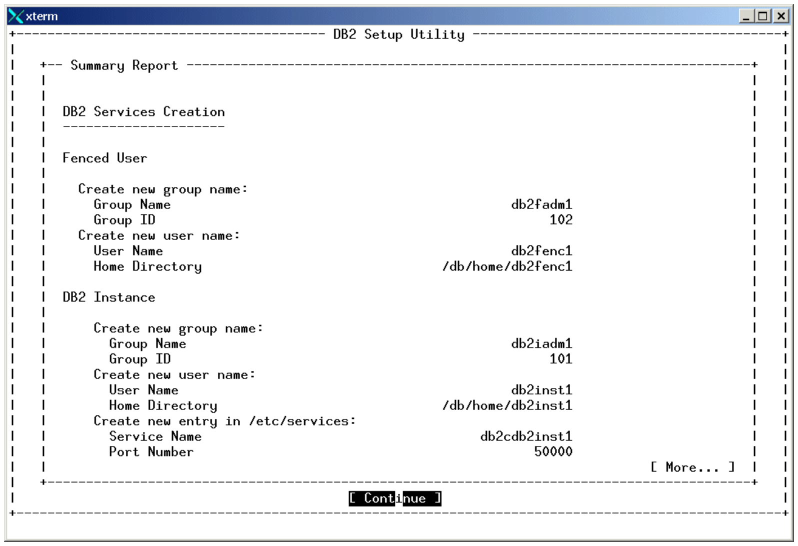
Note: If the Report server is on its own server, then only the DB2 client will need to be setup.

To install DB2 on the PC server see Section 3.3.1, “Windows DB2 Universal Database installation” on page 60. To install DB2 on the AIX box we did the following:

1. Start the DB2 setup utility for AIX run command:

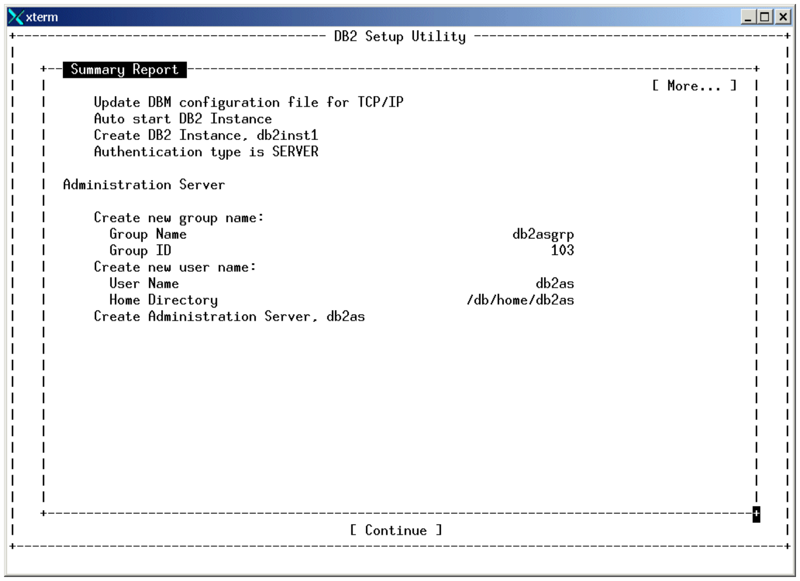
```
./db2setup
```

2. Create a new DB2 instance. See Figure 3-7 and Figure 3-8 for the install options we selected.



```
xterm
+---------------------------------------- DB2 Setup Utility ----------------------------------------+
|                                                                                                   |
|   +-- Summary Report ---------------------------------------------------------------------+      |
|   |                                                                                        |     |
|   |                                                                                        |     |
|   |  DB2 Services Creation                                                                 |     |
|   |  ---------------------                                                                 |     |
|   |                                                                                        |     |
|   |  Fenced User                                                                           |     |
|   |                                                                                        |     |
|   |    Create new group name:                                                              |     |
|   |      Group Name                                       db2fadm1                         |     |
|   |      Group ID                                              102                         |     |
|   |    Create new user name:                                                               |     |
|   |      User Name                                        db2fenc1                         |     |
|   |      Home Directory                          /db/home/db2fenc1                         |     |
|   |                                                                                        |     |
|   |  DB2 Instance                                                                          |     |
|   |                                                                                        |     |
|   |      Create new group name:                                                            |     |
|   |        Group Name                                     db2iadm1                         |     |
|   |        Group ID                                            101                         |     |
|   |      Create new user name:                                                             |     |
|   |        User Name                                      db2inst1                         |     |
|   |        Home Directory                        /db/home/db2inst1                         |     |
|   |      Create new entry in /etc/services:                                                |     |
|   |        Service Name                               db2cdb2inst1                         |     |
|   |        Port Number                                       50000                         |     |
|   |                                                                           [ More... ]  |     |
|   +----------------------------------------------------------------------------------------+     |
|                                          [ Continue ]                                             |
+---------------------------------------------------------------------------------------------------+
```

Figure 3-7   AIX DB2 install summary



```
xterm
+---------------------------------------- DB2 Setup Utility ----------------------------------------+
|                                                                                                   |
|   +-- Summary Report ---------------------------------------------------------------------+      |
|   |                                                                           [ More... ]  |     |
|   |      Update DBM configuration file for TCP/IP                                          |     |
|   |      Auto start DB2 Instance                                                           |     |
|   |      Create DB2 Instance, db2inst1                                                     |     |
|   |      Authentication type is SERVER                                                     |     |
|   |                                                                                        |     |
|   |  Administration Server                                                                 |     |
|   |                                                                                        |     |
|   |      Create new group name:                                                            |     |
|   |        Group Name                                     db2asgrp                         |     |
|   |        Group ID                                            103                         |     |
|   |      Create new user name:                                                             |     |
|   |        User Name                                         db2as                         |     |
|   |        Home Directory                           /db/home/db2as                         |     |
|   |      Create Administration Server, db2as                                               |     |
|   |                                                                                        |     |
|   +----------------------------------------------------------------------------------------+     |
|                                          [ Continue ]                                             |
+---------------------------------------------------------------------------------------------------+
```

Figure 3-8   AIX DB2 install summary

3.4.2 Tivoli Enterprise Data Warehouse installation

When the Tivoli Enterprise Data Warehouse installation wizard runs the installation, it will install the three databases in the default directory. If you want these databases installed in another directory, create these databases before you start the Tivoli Enterprise Data Warehouse installation wizard by doing the following:

1. Open a DB2 command window and issue these command on the Control server:

   ```
   db2 create db twh_md on d
   ```

   Where d is the drive you would like the database installed on or, if it is a UNIX machine, then /db/data is the file system.

2. Issue these commands on the Data mart server:

   ```
   db2 create db twh_mart on /db/data
   ```

3. Issue these commands on the CDW server:

   ```
   db2 create db twh_cwd on /db/data
   ```

After installing DB2, perform a connection test by completing these steps:

1. Enter the following command in a DB2 command window to list the local databases:

   ```
   db2 list database directory
   ```

   In most cases, the command lists at least one database, even in new installations.

2. Enter the following command to test a local connection:

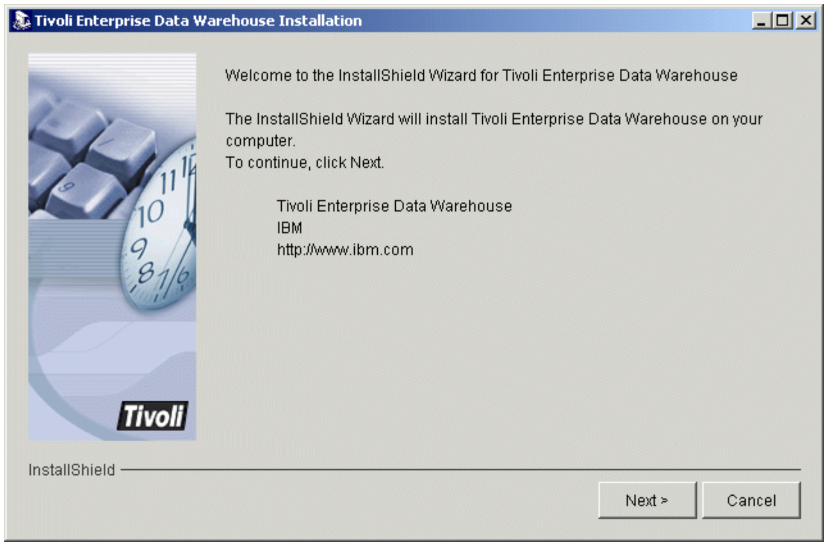
   ```
   db2 connect to database_name user user_name using password
   ```

   Where database_name is the database name returned in the first step, user_name and password are the user name and password you specified when installing DB2. If this command is successful, DB2 is installed and ready for remote connections. If you experience difficulty, refer to the DB2 documentation for troubleshooting information.

To install the Tivoli Enterprise Data Warehouse components complete the following:

1. Insert the Tivoli Enterprise Data Warehouse installation media into the CD-ROM drive of the machine that will be the Control server and run **`setup.exe`**.

   This will start the Tivoli Enterprise Data Warehouse installation wizard (Figure 3-9 on page 71).


Tivoli Enterprise Data Warehouse Installation

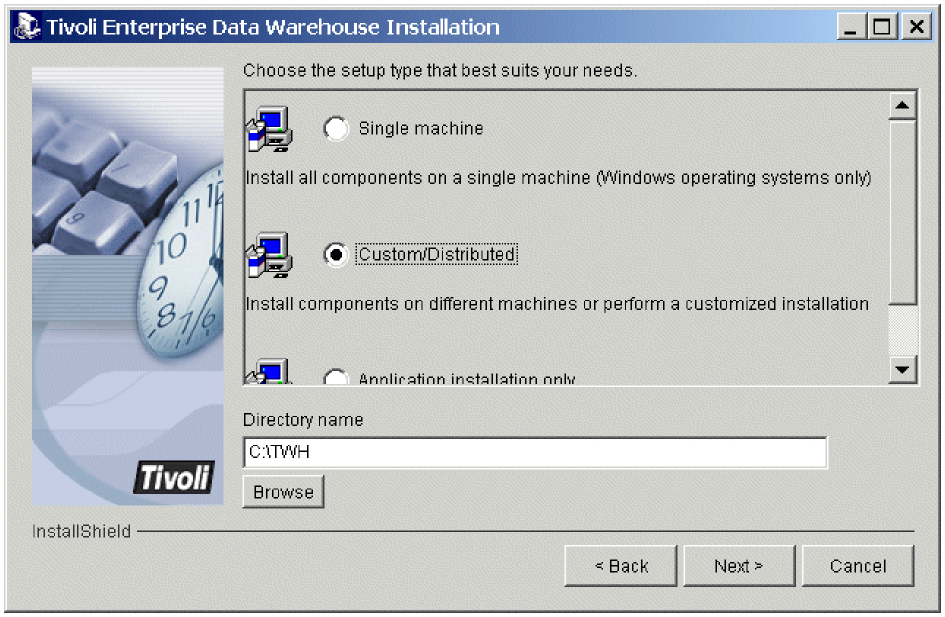
Welcome to the InstallShield Wizard for Tivoli Enterprise Data Warehouse

The InstallShield Wizard will install Tivoli Enterprise Data Warehouse on your computer.
To continue, click Next.

Tivoli Enterprise Data Warehouse
IBM
http://www.ibm.com

Figure 3-9 Welcome dialog window

2. Choose **Next** and the dialog window for installation type will appear (Figure 3-10).


Tivoli Enterprise Data Warehouse Installation

Choose the setup type that best suits your needs.

( ) Single machine

Install all components on a single machine (Windows operating systems only)

(●) Custom/Distributed

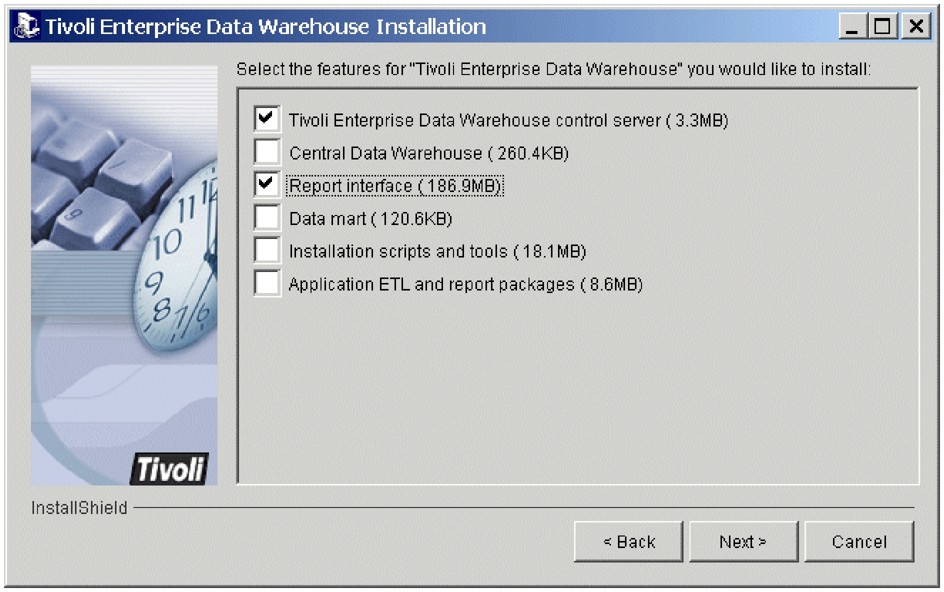
Install components on different machines or perform a customized installation

( ) Application installation only

Directory name
C:\TWH

Figure 3-10 Install type dialog window

3. Select **Custom/Distributed** and insert your install directory name, then choose **Next**. The select features dialog window will appear (Figure 3-11).



Select the features for "Tivoli Enterprise Data Warehouse" you would like to install:

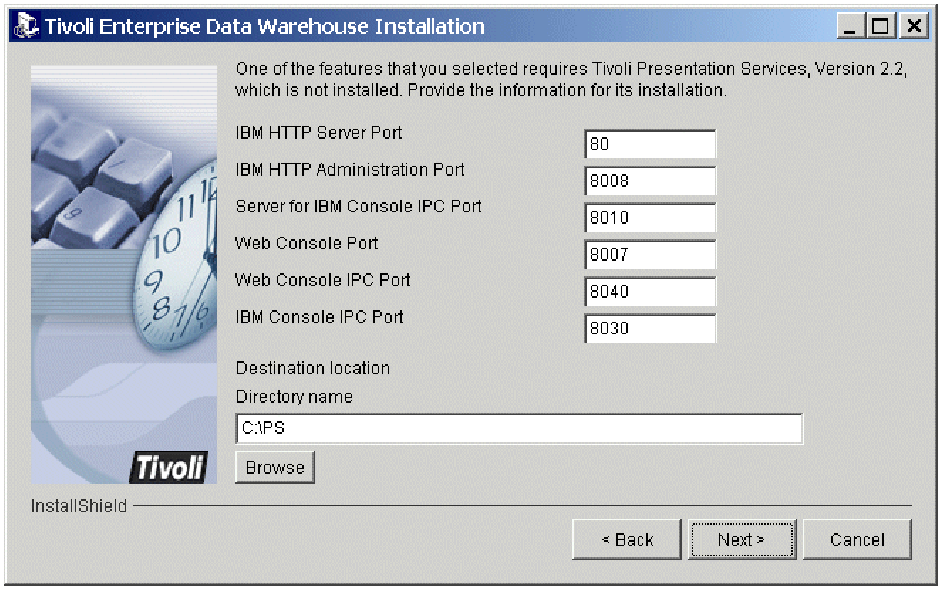
- [✓] Tivoli Enterprise Data Warehouse control server (3.3MB)
- [ ] Central Data Warehouse (260.4KB)
- [✓] Report interface (186.9MB)
- [ ] Data mart (120.6KB)
- [ ] Installation scripts and tools (18.1MB)
- [ ] Application ETL and report packages (8.6MB)

Figure 3-11 Select features dialog window

4. Select **Tivoli Enterprise Data Warehouse Control server** and **Report Interface** (because this machine will be our Report Server as well) and choose **Next**. The verify hostname dialog window will appear.

    Note: The installation wizard displays an additional feature, Installation tools and scripts. This feature is required for each of the Tivoli Enterprise Data Warehouse components, but you do not have to select it manually. It is automatically included when it is required. It is not needed when installing warehouse packs (listed in the installation wizard as Application ETL and Report packages).

5. Confirm that this is the correct hostname and choose **Next**. The local DB2 configuration dialog window will appear.
6. Enter the DB2 user ID and password created during the local DB2 installation on the Control server and choose **Next**. The Tivoli Presentation Services dialog will appear (Figure 3-12 on page 73).


Tivoli Enterprise Data Warehouse Installation

One of the features that you selected requires Tivoli Presentation Services, Version 2.2, which is not installed. Provide the information for its installation.

IBM HTTP Server Port	80
IBM HTTP Administration Port	8008
Server for IBM Console IPC Port	8010
Web Console Port	8007
Web Console IPC Port	8040
IBM Console IPC Port	8030

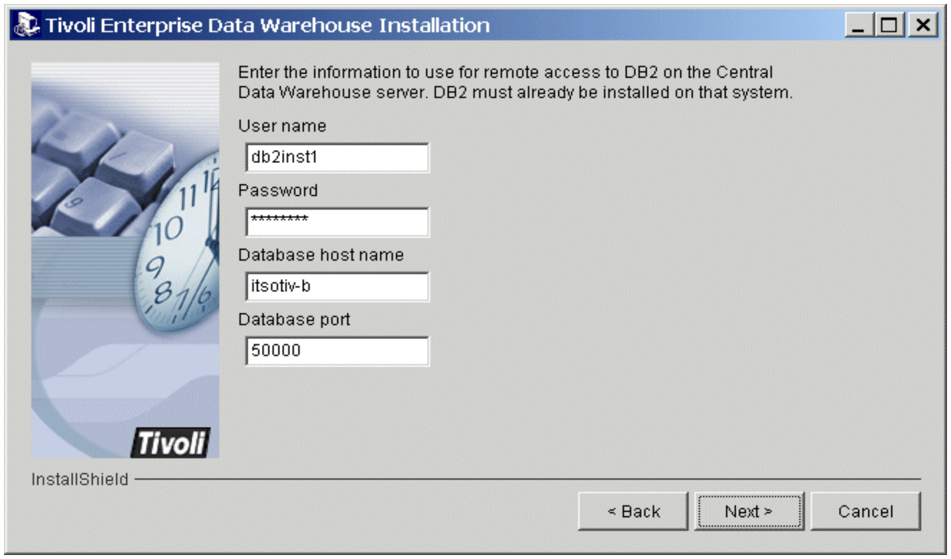
Destination location

Directory name

C:\PS

Figure 3-12 Default ports dialog window

7. If any of these ports are being used by another application, change the default ports now and choose **Next**. The install languages dialog window will appear.
8. Leave this box unchecked and choose **Next**. The remote DB2 configuration for the Central Data Warehouse server dialog window will appear (Figure 3-13 on page 74).


Tivoli Enterprise Data Warehouse Installation

Enter the information to use for remote access to DB2 on the Central Data Warehouse server. DB2 must already be installed on that system.

User name: db2inst1

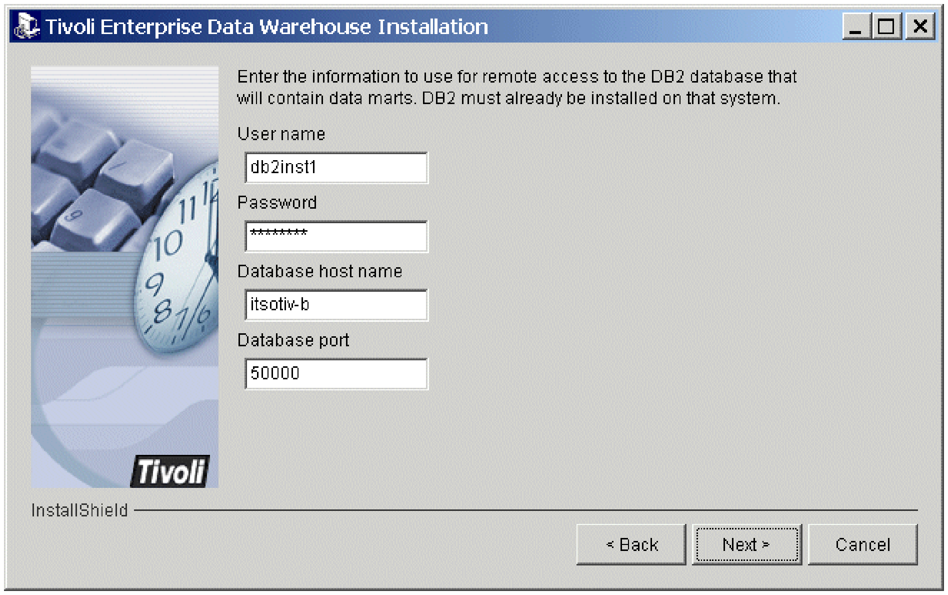
Password: *********

Database host name: itsotiv-b

Database port: 50000

Figure 3-13 Remote DB2 configuration dialog window

9. Enter the DB2 user ID and password created during the DB2 installation on the central data warehouse server and choose **Next**. The remote DB2 configuration for the Data mart server dialog will appear (Figure 3-14 on page 75).


Tivoli Enterprise Data Warehouse Installation

Enter the information to use for remote access to the DB2 database that will contain data marts. DB2 must already be installed on that system.

User name: db2inst1

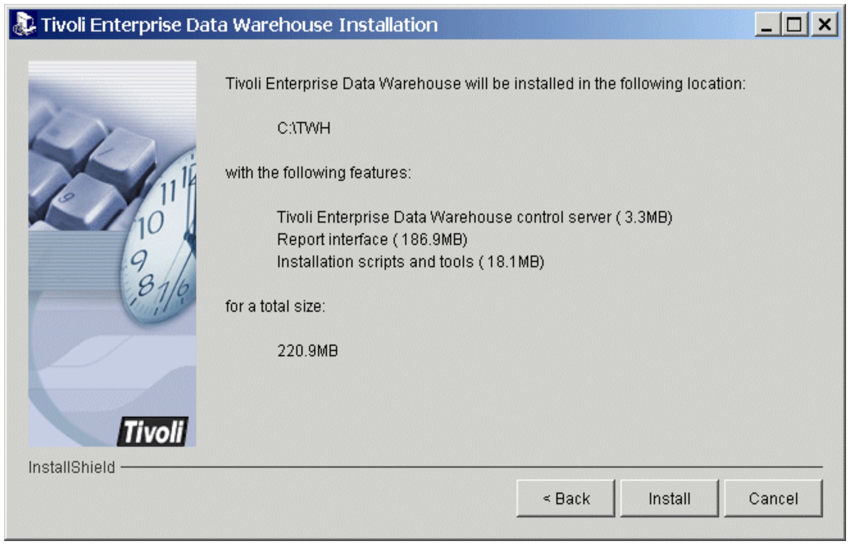
Password: ********

Database host name: itsotiv-b

Database port: 50000

Figure 3-14 Remote DB2 configuration dialog window

10. Enter the DB2 user ID and password created during the DB2 installation on the Data mart server and choose **Next**. The dialog window overview of chosen options will appear (Figure 3-15).


Tivoli Enterprise Data Warehouse Installation

Tivoli Enterprise Data Warehouse will be installed in the following location:

C:\TWH

with the following features:

Tivoli Enterprise Data Warehouse control server (3.3MB)
Report interface (186.9MB)
Installation scripts and tools (18.1MB)

for a total size:

220.9MB

Figure 3-15 Selected options dialog window

11. Choose **Install** to start the installation.
12. Verify that the installation was successful by doing the following:
    - Make sure the installation wizard's completion window does not list any errors. See "Messages" in *Installing and Configuring Tivoli Enterprise Data Warehouse*, GC32-0744, for more information.
    - If the completion window lists warnings, check the TWH.log to ensure that the warnings can safely be ignored.
13. Wait for the help set to be rebuilt. The help set contains the user assistance for the IBM Console. This process happens asynchronously and might not complete by the time the wizard has completed the installation of the remaining components of Tivoli Enterprise Data Warehouse. Do not restart the system until the help set is complete. If you do, follow the procedure documented in the Tivoli Enterprise Data Warehouse Release Notes to rebuild the help set. To determine whether the help set rebuild is complete, look for the completion message in the Tivoli Presentation Services installation log. This log is in directory PS_directory/log/fwp_mcr, where PS_directory is the target directory you specified for Tivoli Presentation Services. The logs are named stdoutn. Look for the message shown in Example 3-2 in the most recent stdoutn file. In some cases, the message can be in the second most recent stdoutn file.

Example 3-2 Successful completion message

```
FWP1734I The utility that was started by the Management Component Repository to
build the help set has completed
successfully.
```

    This process can take up to 30 to 40 minutes.

14. Reboot the machine.

To create the basic installation using a local warehouse agent, you do not need to install any Tivoli Enterprise Data Warehouse components on the central data warehouse server or the Data mart server. Installing the Control server creates the databases that are needed on those systems. If errors were detected during installation or an un-install is required, see "Troubleshooting and maintenance" on page 283.

3.5 Tivoli Enterprise Data Warehouse configuration

This section gives instructions on the basic configuration setup it applies to both stand-alone Tivoli Enterprise Data Warehouse installation and distributed Tivoli Enterprise Data Warehouse installation.

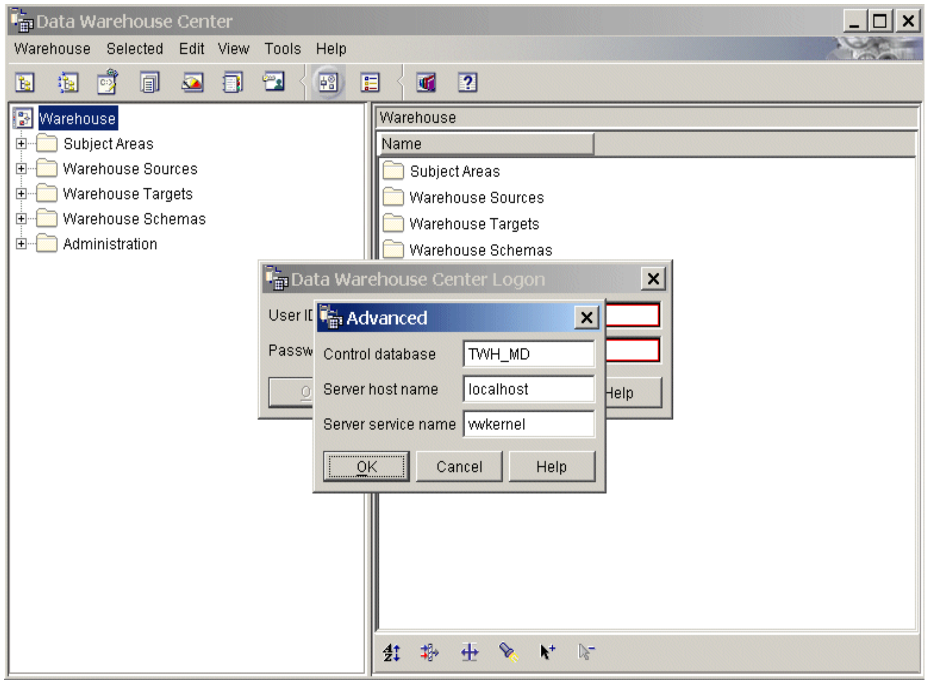
To configure the Tivoli Enterprise Data Warehouse installation you need to follow these steps:

1. Configure the Data Warehouse Center for Tivoli Enterprise Data Warehouse by following the instructions in Section 3.5.1, "Specifying the control database for the Data Warehouse Center" on page 77.
2. Test the sources and targets in the Data Warehouse Center. For instructions, refer to Section 3.5.2, "Test sources and targets in the Data Warehouse Center" on page 79.
3. Back up the Tivoli Enterprise Data Warehouse installation as described in *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744, in the section "Backup and Restore recommendations."
4. Install the warehouse packs that allow your system management software to work with the Tivoli Enterprise Data Warehouse. For instructions, refer to Section 3.5.3, "Installing warehouse packs" on page 81.
5. Configure the reporting server users and create reports (see Chapter 4, "Implementation of the Report Interface" on page 89).

3.5.1 Specifying the control database for the Data Warehouse Center

The first time you open the Data Warehouse Center, and if you are using the Data Warehouse Center with applications other than Tivoli Data Enterprise Warehouse, you might need to set the control database in the IBM DB2 Data Warehouse Center as follows:

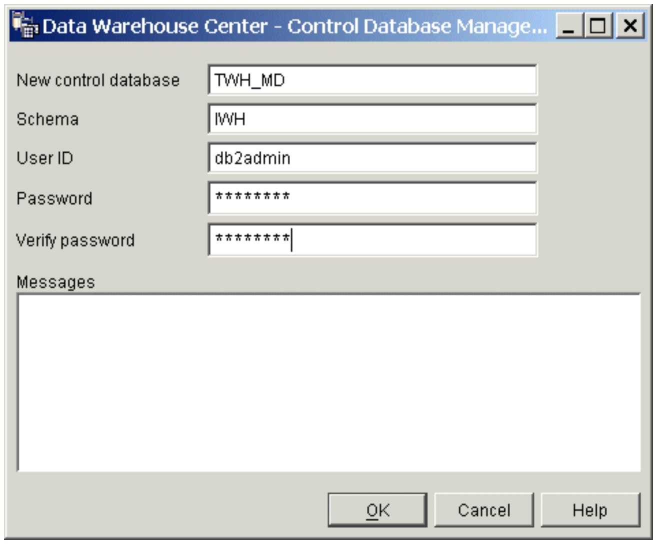
1. On the Windows taskbar, click **Start Programs -> IBM DB2 -> Control Center**. The Control Center window is displayed.
2. From the DB2 Control Center, start the DB2 Data Warehouse Center by clicking **Tools -> Data Warehouse Center**. The Data Warehouse Center Logon window is displayed.
3. In the Data Warehouse Center Logon window, click **Advanced**.
4. Type TWH_MD for the control database and click **OK** (Figure 3-16 on page 78).



Control database	TWH_MD
Server host name	localhost
Server service name	vwkernel

Figure 3-16 Configure Data Warehouse Center window

5. Click **Cancel** to close the logon panel.
6. Open the Control Database Management window. On the Windows taskbar, click **Start -> Programs -> IBM DB2 -> Warehouse Control Database Management**.
7. Type `TWH_MD` in the New control database field and type the DB2 user name and password, then click **OK** (Figure 3-17 on page 79).



New control database: TWH_MD
Schema: IWH
User ID: db2admin
Password: ********
Verify password: ********
Messages

Figure 3-17 Configure Warehouse Control Database Management window

8. When the `Processing has completed` message appears, click **Cancel**.

The set up and connection is now complete.

3.5.2 Test sources and targets in the Data Warehouse Center

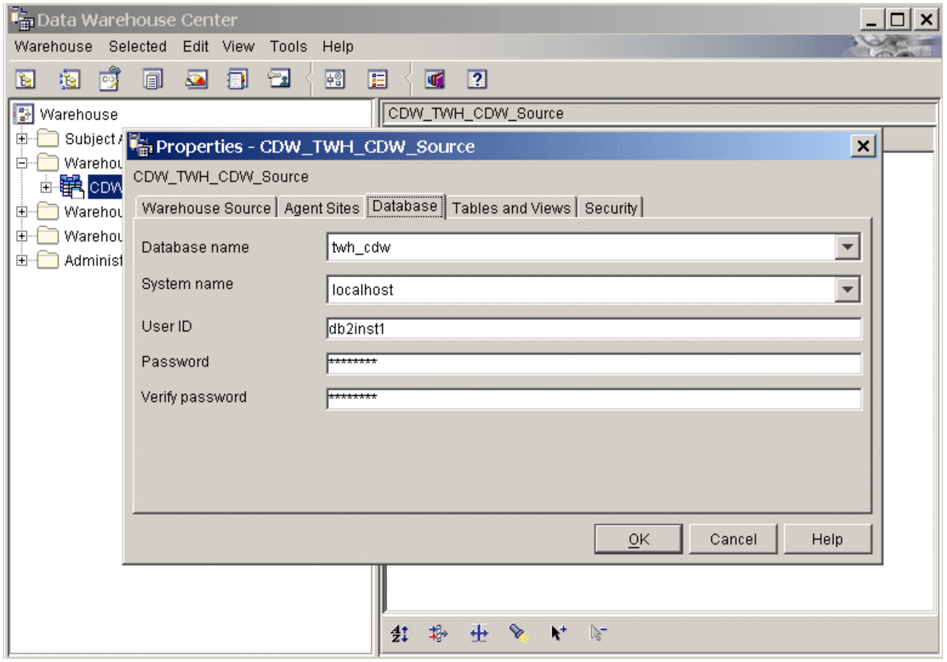
The installation program will automatically create the ODBC data sources required to connect to DB2 databases.

Tip: The only time you would need to create the ODBC data sources for the source application databases manually is when you are using is remote warehouse agents on UNIX systems. Please refer to *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744, Appendix D “Using remote warehouse agents.” For all other cases, ODBC data sources are created automatically by the Tivoli Enterprise Data Warehouse installation program.

You can test the ODBC connection before attempting to run an ETL process as follows:

1. On the Windows taskbar, click **Start Programs -> IBM DB2 -> Control Center**. The Control Center window is displayed.

2. From the DB2 Control Center, start the DB2 Data Warehouse Center by clicking **Tools -> Data Warehouse Center**. The Data Warehouse Center Logon window is displayed
3. Enter the user name and password and click **OK**. The Data Warehouse Center is displayed.
4. Expand the tree **Warehouse Sources**. Right click the source and select **Properties**, then **Database** from this window.
5. Fill in the User ID and Password fields and click **OK** (Figure 3-18).



Database name: twh_cdw
System name: localhost
User ID: db2inst1
Password: *********
Verify password: *********

Figure 3-18 Configure target and sources window

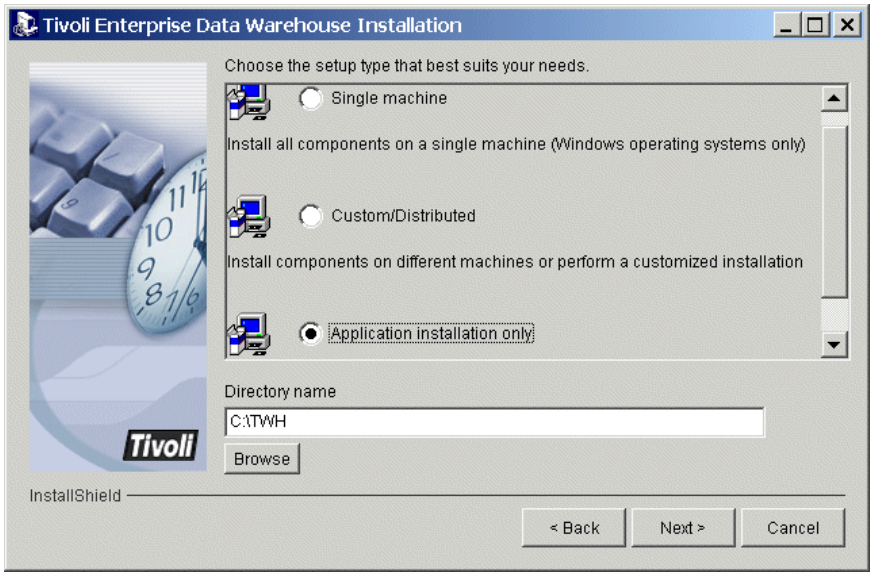
For each of the source and target databases repeat the same steps.

3.5.3 Installing warehouse packs

Each application will come with its own warehouse pack. Installation and configuration processes will differ slightly for each warehouse pack. We recommend that the installation instructions for each warehouse pack are followed. Also see *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744, section “Configuring and scheduling warehouse pack ETL processes”. Typically the installation has the following steps:

1. Use the Tivoli Enterprise Data Warehouse installation program to install the package on each machine where it is needed. In general, this is the Control server and the Report server. The warehouse pack’s documentation describes whether it needs to be installed on additional or fewer systems. After starting the Tivoli Enterprise Data Warehouse installation wizard, specify the installation of one or more warehouse packs using the following techniques:
   a. Select **Application installation only** in the setup type window (Figure 3-19 on page 82). This is the recommended way to install warehouse packs. You can also install warehouse packs during the installation of the Tivoli Enterprise Data Warehouse core application, but this is not recommended.

Note: The documentation is on the installation media for the warehouse pack, in a PDF file in the subdirectory tedw_apps_etl/product_code/pkg/version/doc. product_code and version specify, respectively, the product that is being enabled to use Tivoli Enterprise Data Warehouse and the version of the warehouse pack.



Choose the setup type that best suits your needs.

( ) Single machine

Install all components on a single machine (Windows operating systems only)

( ) Custom/Distributed

Install components on different machines or perform a customized installation

(●) Application installation only

Directory name

C:\TWH

Figure 3-19   Install type dialog window

b. Click **Next** and verify that the correct host name is entered. Click **Next** to enter the local DB2 database user name and password. Click **Next**.

c. This step is only for a distributed Tivoli Enterprise Data Warehouse installation. Enter the remote DB2 central data warehouse server database user name, password, and database host name, and click **Next**.

d. This step is only for a distributed Tivoli Enterprise Data Warehouse installation. Enter the remote DB2 data mart database user name, password, and database host name, and click **Next**.

e. Enter the path of the installation media for the warehouse pack, and click **Next**.

f. Click **Next** on the install additional application packages and then **Install** to start the installation.

g. Once the installation is finished, check the log files for any errors. See Chapter 10, “Troubleshooting and maintenance” on page 283 if errors are detected.

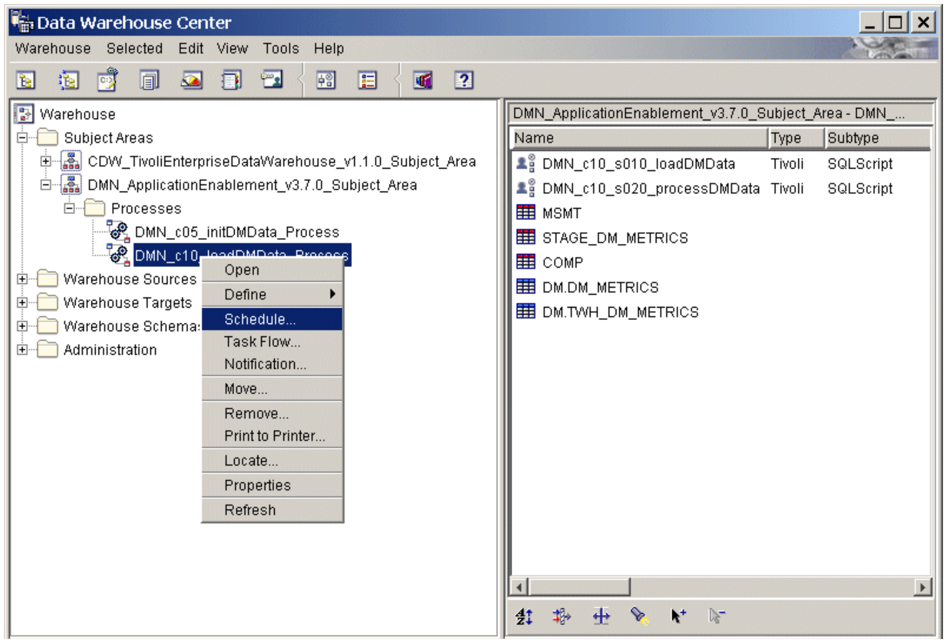
2. Install any product patches specified by the warehouse pack documentation.

3. Perform any pre-installation configuration steps specified by the warehouse pack documentation. For example, this might include tasks such as:
   – Creating additional tables in an existing database.
   – Establishing an ODBC connection.
   – Configuring the Source database user name and password. See Section 3.5.2, “Test sources and targets in the Data Warehouse Center” on page 79.
4. Each warehouse pack will come with its own set of ETL processes that must be scheduled to run so the Tivoli Enterprise Data Warehouse can be populated with data. Check the warehouse pack documentation for the order that the ETL processes must run in.

   Note: A process’ start could depend on another’s completion.

   To schedule a process follow these tasks:

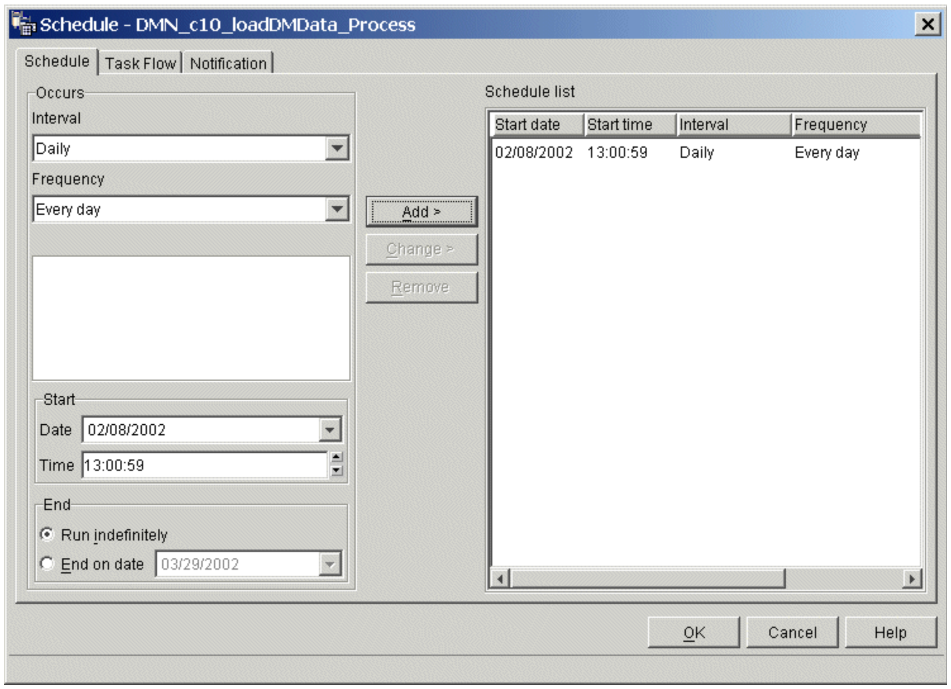
   a. On the Windows task bar, click **Start -> Programs -> IBM DB2 -> Control Center**. The Control Center window is displayed.
   b. From the DB2 Control Center, start the DB2 Data Warehouse Center by clicking **Tools -> Data Warehouse Center**. The Data Warehouse Center Logon window is displayed
   c. Enter the user name and password and click **OK.** The Data Warehouse Center is displayed.
   d. Expand the tree **Subject Areas**, there will be a list of applications. Expand the application you just installed and expand the **Processes** tree (Figure 3-20 on page 84).


Data Warehouse Center
Warehouse Selected Edit View Tools Help
Warehouse
Subject Areas
CDW_TivoliEnterpriseDataWarehouse_v1.1.0_Subject_Area
DMN_ApplicationEnablement_v3.7.0_Subject_Area
Processes
DMN_c05_initDMData_Process
DMN_c10_loadDMData_Process
Warehouse Sources
Warehouse Targets
Warehouse Schemas
Administration
Open
Define
Schedule...
Task Flow...
Notification...
Move...
Remove...
Print to Printer...
Locate...
Properties
Refresh
DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_...

Name	Type	Subtype
DMN_c10_s010_loadDMData	Tivoli	SQLScript
DMN_c10_s020_processDMData	Tivoli	SQLScript
MSMT		
STAGE_DM_METRICS		
COMP		
DM.DM_METRICS		
DM.TWH_DM_METRICS		

Figure 3-20 Data Warehouse Center

e. Right click the process you want to schedule and select **Schedule**. The schedule dialog window will open. Select the Interval, Frequency, Start Time and click **Add** (Figure 3-21 on page 85).


Schedule - DMN_c10_loadDMData_Process

Occurs
Interval: Daily
Frequency: Every day

Start
Date: 02/08/2002
Time: 13:00:59

End
(●) Run indefinitely
( ) End on date 03/29/2002

Schedule list

Start date	Start time	Interval	Frequency
02/08/2002	13:00:59	Daily	Every day

Figure 3-21 Schedule Process window

f. Click **OK**.

g. To enable the schedule the run, the mode of the tasks for the process need to be moved from development production mode. In the right pane of the Data Warehouse Center window select all the tasks, right click, select **Mode** and then **Production** (Figure 3-22 on page 86).

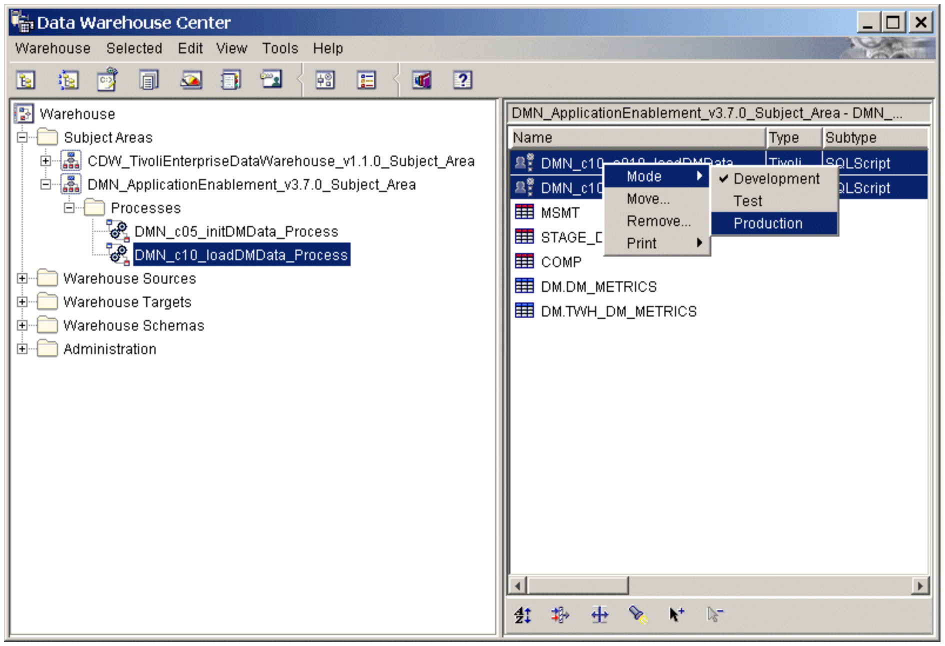
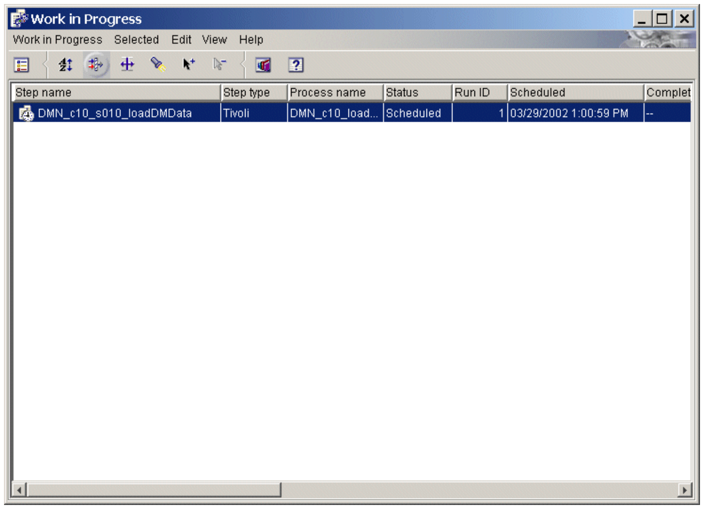


Figure 3-22 Change task mode window

h. The process and all its tasks are now scheduled to run. To verify this from the Data Warehouse Center window, select **Warehouse -> Work in Progress** (Figure 3-23).



Step name	Step type	Process name	Status	Run ID	Scheduled	Complet
DMN_c10_s010_loadDMData	Tivoli	DMN_c10_load...	Scheduled	1	03/29/2002 1:00:59 PM	--

Figure 3-23 Work in Progress window

5. After you install one or more warehouse packs on the Report server, manually stop and then restart the following services for Tivoli Presentation Services:
   – Server for IBM Console
   – Web Services for the IBM Console

Once each process has completed successfully, you are now ready to view your performance data. See Chapter 4, “Implementation of the Report Interface” on page 89.

4

Implementation of the Report Interface

In this chapter we introduce the Report Interface (RI) of Tivoli Enterprise Data Warehouse. We show how to best customize and use this interface to get performance reports from various application data. Some tips for troubleshooting are also provided.

This chapter has the following sections:

- Tivoli Enterprise Data Warehouse Report Interface
- Basic customization
- Types of reports

4.1 Tivoli Enterprise Data Warehouse Report Interface

Using the Tivoli Enterprise Data Warehouse Report Interface, you can create and run some basic reports against your data marts and publish them on your intranet or the Internet. The Report Interface is not meant to replace OLAP or Business Intelligence tools. If you have *multidimensional reporting* requirements or need to create a more sophisticated analysis of your data, Tivoli Enterprise Data Warehouse's open structure provides you with an easy interface to plug into OLAP or BI-tools. How to use those tools is explained in Chapter 7, "OLAP integration" on page 203. Nevertheless, for *two-dimensional reporting requirements,* Tivoli Enterprise Data Warehouse Report Interface provides you with a powerful tool.

The RI is a role-based Web interface that allows you to create reports from your aggregated enterprise-wide data that is stored in various data marts. The RI uses the Tivoli Presentation Service.

The GUI can be customized for each user. Different roles can be assigned to the users according to the tasks they have to fulfill and the reports they may look at. The users see in their GUI only those menus, which they can use according to their roles.

The RI can be accessed with a normal Web browser from everywhere in the network. We recommend using Internet Explorer. Other Web browsers, like Netscape, will also work, but might be slower.

Tip: JavaScript and style sheets must be enabled in your browser.

To connect to your Report Interface start your Web browser with the following:

```
http://your_ri_server/IBMConsole
```

Where you have to use the host name (fully qualified) of your Report server. The server port is 80 by default. If you chose another port during installation of the Tivoli Presentation Service (see Figure 3-4 on page 65, entry `IBM HTTP Server Port`), use the following syntax to start the Report Interface through a different port:

```
http://your_ri_server:your_port/IBMConsole
```

When you log in for the first time use the login `superadmin` and password `password` (you should change this password immediately). After the log in you see the Welcome page. On the left-hand side you will find the pane My Work, with all tasks that you may perform.

4.2 Basic customization

In this section we describe the concept of users, user groups, roles, and data marts, and how they are customized. We will also give a short description of the three report types supported by the Report Interface. You will find more sophisticated examples for working with the Report Interface in Chapter 8, “Real-life scenarios” on page 253.

4.2.1 Roles

Roles are used to grant rights to users according to the tasks they have to perform and data they may access. A customer may, for instance, see his own data, but not the data of other customers. A sales representative may create and manage reports for the customer he is responsible for, but only view the reports of all other customers.

The granted roles will also change the user’s GUI, so that he only sees those menus that he is allowed to use. The user roles listed in Table 4-1 are predefined for you.

Table 4-1 Predefined roles

Predefined roles	Description
com.tivoli.twh.rpi.role.impl/1.1.0/RepReaderRole[a]	Report reader. Enables you to run, display, and save output of public reports.
com.tivoli.twh.rpi.role.impl/1.1.0/RepAuthRole [a]	Report author. Combines the report reader role and the ability to create and manage personal reports.
MACImpl/2.2.0/MACLoggingUser [a]	Access to all tasks within the Administer Logging task group. Can perform logging administration functions.
com.tivoli.pf.admin.impl/2.2.0/Administration [b]	Portfolio Administration portfolio entries.
com.tivoli.pf.pfconsole.impl/2.2.0/ConsoleUsers [b]	IBM Console user.
com.tivoli.twh.rpi.role.impl/1.1.0/AdvRepAuthRole [b]	Advanced report author. Includes the report author role plus the ability to create and manage public reports.
com.tivoli.twh.rpi.role.impl/1.1.0/SecAdminRole [a]	Warehouse security administrator. Controls access by managing data marts and user groups to Tivoli Enterprise Data Warehouse.
com.tivoli.pf.admin.impl/2.2.0/Administration [b]	Authorizations Administration authorized tasks.

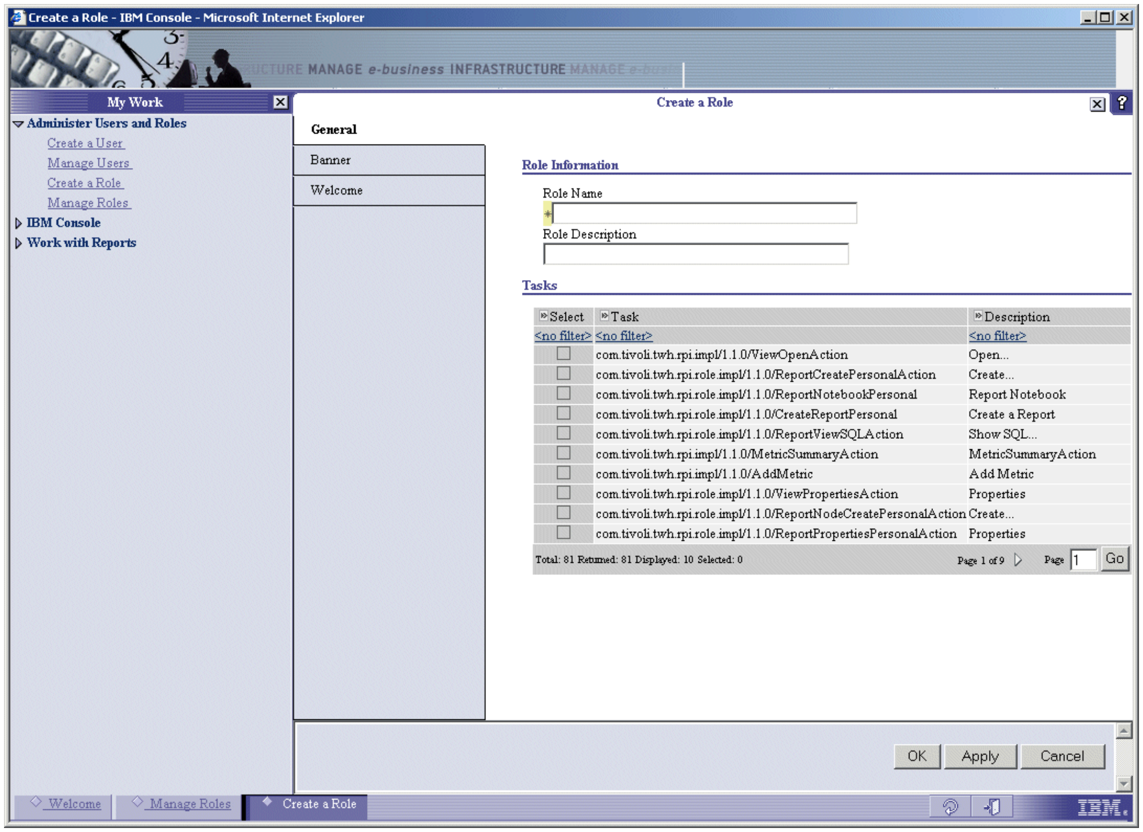
a Tivoli Enterprise Data Warehouse specific roles

b IBM Console’s users and administrators roles

The roles given in Table 4-1 on page 91 should be sufficient for most purposes. However, we will also demonstrate how to create new roles and how to customize existing roles.

Important: You have to be careful and know exactly what you are doing when creating new roles or modifying existing ones. If performed badly, this action could make the whole system unstable.

1. Log in as superadmin or as a user with sufficient rights to manage roles.
2. Click **Administer Users and Roles** and then **Create Roles**. The GUI shown in Figure 4-1 will appear.



Role Information

Role Name

Role Description

Tasks

Select	Task	Description
<no filter>	<no filter>	<no filter>
☐	com.tivoli.twh.rpi.impl/1.1.0/ViewOpenAction	Open...
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ReportCreatePersonalAction	Create...
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ReportNotebookPersonal	Report Notebook
☐	com.tivoli.twh.rpi.role.impl/1.1.0/CreateReportPersonal	Create a Report
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ReportViewSQLAction	Show SQL...
☐	com.tivoli.twh.rpi.impl/1.1.0/MetricSummaryAction	MetricSummaryAction
☐	com.tivoli.twh.rpi.impl/1.1.0/AddMetric	Add Metric
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ViewPropertiesAction	Properties
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ReportNodeCreatePersonalAction	Create...
☐	com.tivoli.twh.rpi.role.impl/1.1.0/ReportPropertiesPersonalAction	Properties

Total: 81 Returned: 81 Displayed: 10 Selected: 0   Page 1 of 9   Page 1 Go

Figure 4-1 Create a Role dialog - General

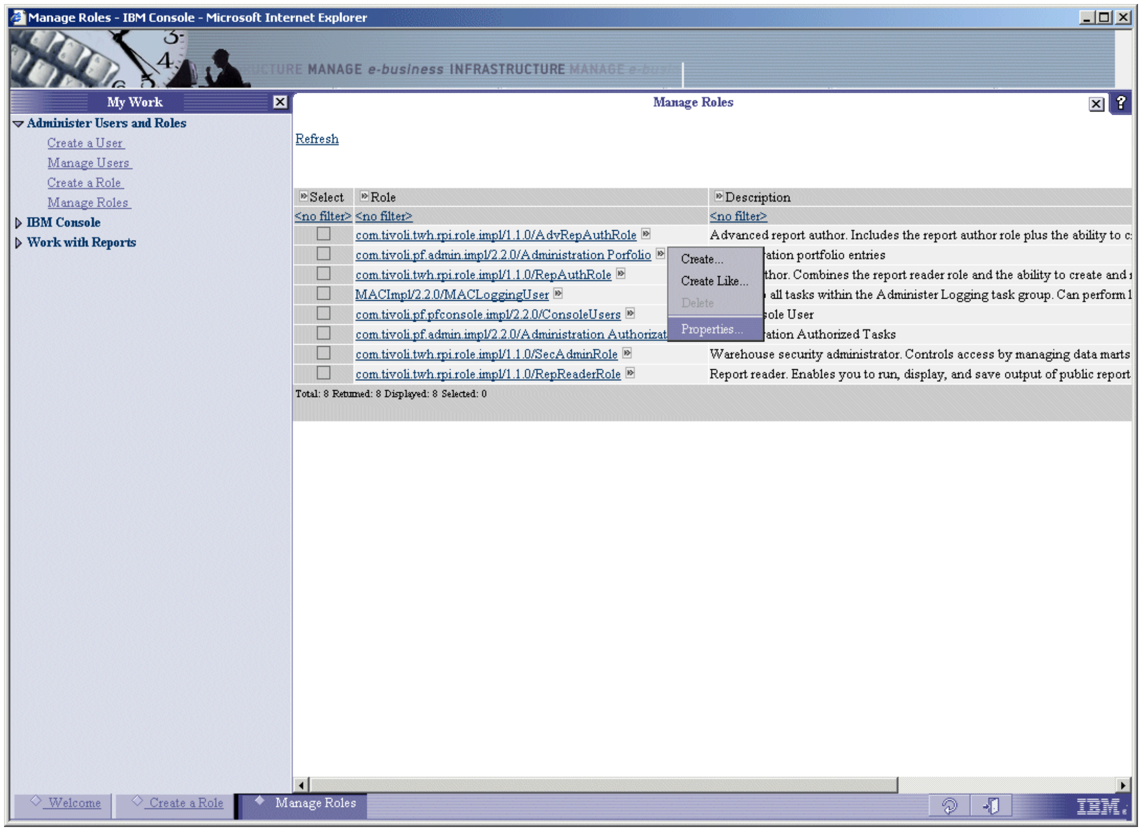
3. First you have to put in a name for the new role and give it a description. Then choose the tasks needed to execute from a list of 81 possible tasks in the Tivoli Enterprise Data Warehouse Report Interface.

Note: Not all of these tasks are Tivoli Enterprise Data Warehouse-specific. Some of them are related to IBM Console administration.

4. On the panels named Banner and Welcome you can define URLs, which will be shown on the Welcome page and the Banner area (the upper panel of the work space), respectively. These settings become active for users that have the new role as their primary role.
5. When finished click **OK** to create the new role.

To change existing roles:

1. Click **Manage Roles** in the panel My Work. You will see a list of all existing roles.
2. Choose **Properties** from the role you want to change (Figure 4-2).



Select	Role	Description
<no filter>	<no filter>	<no filter>
☐	com.tivoli.twh.rpi.role.impl/1.1.0/AdvRepAuthRole	Advanced report author. Includes the report author role plus the ability to c
☐	com.tivoli.pf.admin.impl/2.2.0/Administration Porfolio	ation portfolio entries
☐	com.tivoli.twh.rpi.role.impl/1.1.0/RepAuthRole	thor. Combines the report reader role and the ability to create and
☐	MACImpl/2.2.0/MACLoggingUser	all tasks within the Administer Logging task group. Can perform
☐	com.tivoli.pf.pfconsole.impl/2.2.0/ConsoleUsers	sole User
☐	com.tivoli.pf.admin.impl/2.2.0/Administration Authorizat	ation Authorized Tasks
☐	com.tivoli.twh.rpi.role.impl/1.1.0/SecAdminRole	Warehouse security administrator. Controls access by managing data marts
☐	com.tivoli.twh.rpi.role.impl/1.1.0/RepReaderRole	Report reader. Enables you to run, display, and save output of public report

Total: 8 Returned: 8 Displayed: 8 Selected: 0

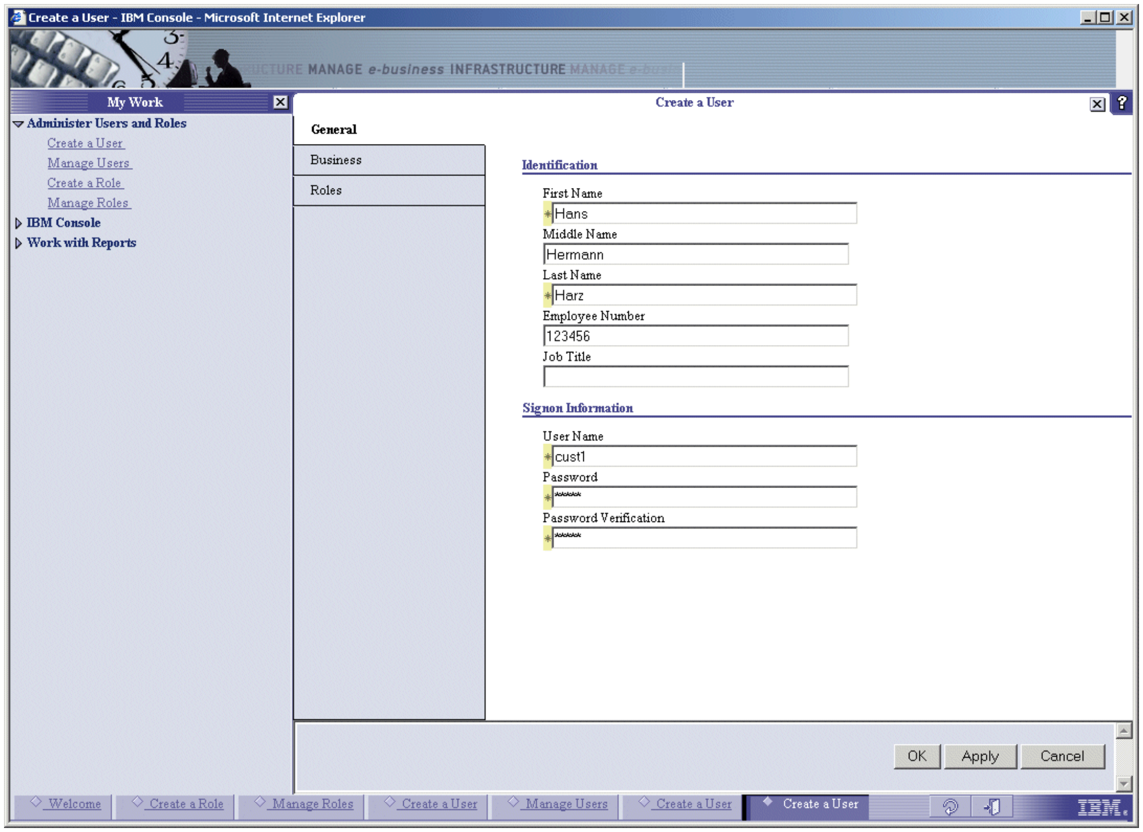
Figure 4-2 Manage Roles dialog

3. You will see the Manage Roles dialog, which is very similar to the Create Roles dialog. Note that you cannot change the default roles of the Tivoli Enterprise Data Warehouse Report Interface.

4.2.2 Users

As mentioned above, the Tivoli Enterprise Data Warehouse Report Interface supports many users with their own user ID and passwords. To create new users do the following:

1. Click **Create User** in the panel My Work (Figure 4-3). You need to have sufficient authorization to do this.



Identification

First Name: Hans
Middle Name: Hermann
Last Name: Harz
Employee Number: 123456
Job Title:

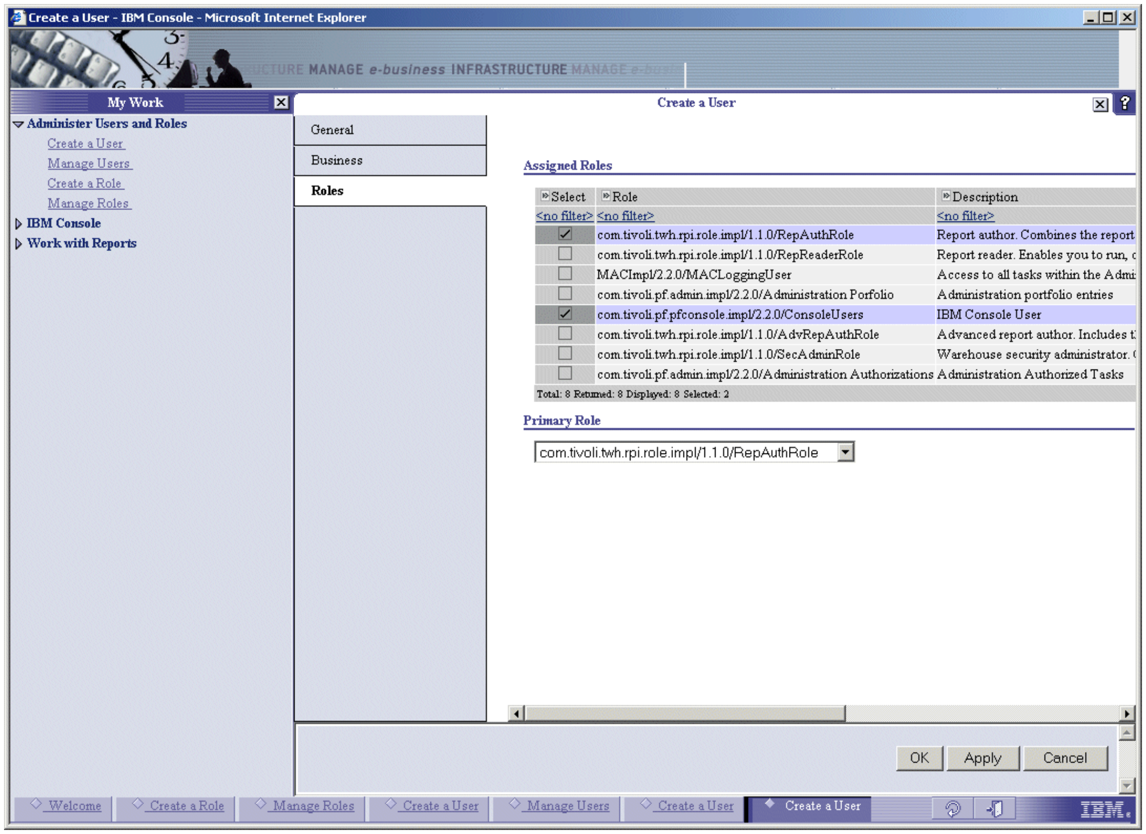
Signon Information

User Name: cust1
Password: *****
Password Verification: *****

Figure 4-3 Create a User dialog - General

2. In the General panel you can provide information about the user's identification and you give him a user ID and a password. Note that the entries with the yellow sign are required entries.

3. In the Business panel you can store contact information like e-mail address, phone number, and mailing address.
4. Click **Roles** to open the Roles panel (Figure 4-4).



Assigned Roles

Select	Role	Description
<no filter>	<no filter>	<no filter>
✓	com.tivoli.twh.rpi.role.impl/1.1.0/RepAuthRole	Report author. Combines the report
	com.tivoli.twh.rpi.role.impl/1.1.0/RepReaderRole	Report reader. Enables you to run, c
	MACImpl/2.2.0/MACLoggingUser	Access to all tasks within the Admi
	com.tivoli.pf.admin.impl/2.2.0/Administration Porfolio	Administration portfolio entries
✓	com.tivoli.pf.pfconsole.impl/2.2.0/ConsoleUsers	IBM Console User
	com.tivoli.twh.rpi.role.impl/1.1.0/AdvRepAuthRole	Advanced report author. Includes t
	com.tivoli.twh.rpi.role.impl/1.1.0/SecAdminRole	Warehouse security administrator. (
	com.tivoli.pf.admin.impl/2.2.0/Administration Authorizations	Administration Authorized Tasks

Total: 8 Returned: 8 Displayed: 8 Selected: 2

Primary Role

com.tivoli.twh.rpi.role.impl/1.1.0/RepAuthRole

Figure 4-4 Create a Users dialog - Roles

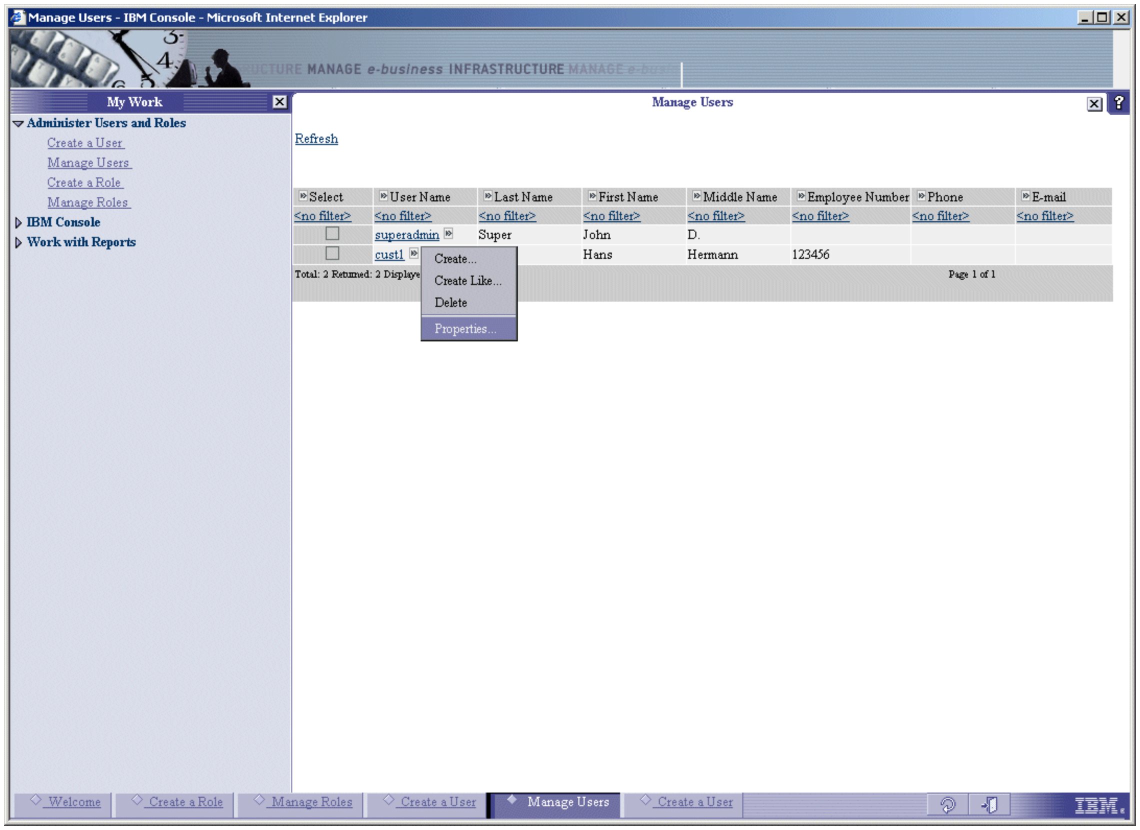
5. In this dialog you can grant one or more roles to the user. One of the granted roles has to be the primary role of the user. This setting has no effect on the rights of the user, but defines the appearance of the Welcome panel and the banner area. Remember, these values are defined with a role (see Section 4.2.1, “Roles” on page 91).

Tip: We recommend the following selection of roles when creating a new Report Interface user:

- You have to select the IBM Console User. It is a must.
- If you want this user to only work with the reports (not to assign data marts to user groups) select one role from Report reader, Report author, or Advanced report author. You should *not* select more than one role (such as Report reader and Report author together), otherwise you will see one Create, and one Properties menu in the reports pop-up menu for each role you have selected, which might be confusing. This is the case when using the superadmin role, which combines all the roles. If you run the reports with the superadmin role, you will see two Create and three Properties menus in the report pop-up menu. For an example of this see Figure 4-22 on page 116.
- If you want this user to only work with assigning data marts to user groups, not to work with reports, select the Warehouse security administrator role.
- If you want this user to both work with assigning data marts to user groups and managing reports, select the Warehouse security administrator role and one role from the Report reader, Report author, or Advanced Report author role.

If you only select the IBM Console User, but not any additional role(s), the user will only be able to see the Welcome page, but do nothing else.

6. When all settings are done, click **OK** to create the user. The new user should appear in the Manage User dialog. If not, click **Refresh**.



Select	User Name	Last Name	First Name	Middle Name	Employee Number	Phone	E-mail
<no filter>	<no filter>	<no filter>	<no filter>	<no filter>	<no filter>	<no filter>	<no filter>
☐	superadmin	Super	John	D.			
☐	cust1		Hans	Hermann	123456		

Total: 2 Returned: 2 Displaye

Page 1 of 1

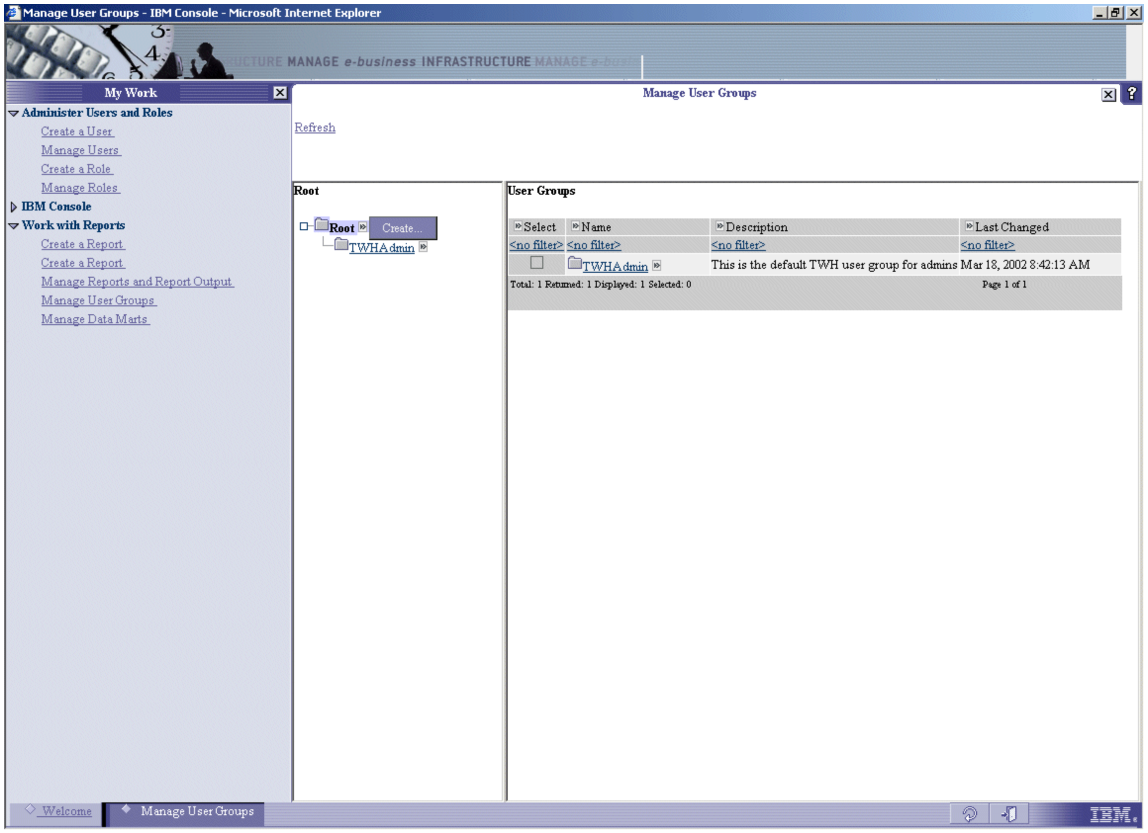
Figure 4-5 Manage Users dialog

7. To change the settings of an existing user, choose **Properties** from the users pop-up menu in the Manage Users dialog. The dialogs that will appear are analogous to the Create Users dialogs.

4.2.3 User groups

User groups are a means of organizing users. Users from different companies or departments can be organized in different groups, each assigned the data mart with the data of their company or department. To create a user group do the following:

1. Choose **Work with reports** from the My Work panel.
2. Click **Manage User Groups**. You will see all existing user groups.
3. To add a new one choose **Create** from the pop-up menu of root (Figure 4-5).

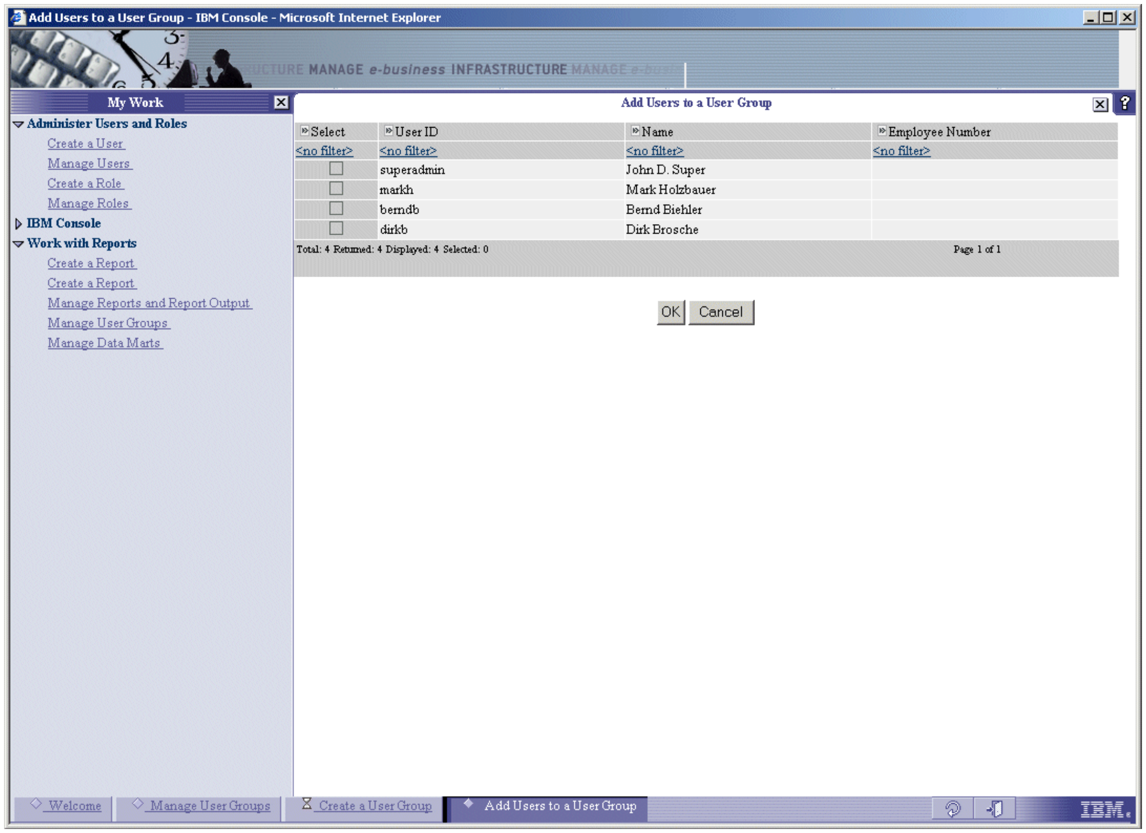


Select	Name	Description	Last Changed
<no filter>	<no filter>	<no filter>	<no filter>
☐	TWHAdmin	This is the default TWH user group for admins	Mar 18, 2002 8:42:13 AM

Total: 1 Returned: 1 Displayed: 1 Selected: 0 Page 1 of 1

Figure 4-6 Create a new user group

4. You will see the Create User Group dialog. You can now provide a name and description for the new user group. Then you have to add users to the user group. Click **Users** and **Add** in the users panel. You will see all users, except those who have no eutrophication (or role) to create or view reports (Figure 4-7 on page 99).


Add Users to a User Group - IBM Console - Microsoft Internet Explorer

My Work

- Administer Users and Roles
  - Create a User
  - Manage Users
  - Create a Role
  - Manage Roles
- IBM Console
- Work with Reports
  - Create a Report
  - Create a Report
  - Manage Reports and Report Output
  - Manage User Groups
  - Manage Data Marts

Add Users to a User Group

Select	User ID	Name	Employee Number
<no filter>	<no filter>	<no filter>	<no filter>
☐	superadmin	John D. Super	
☐	markh	Mark Holzbauer	
☐	berndb	Bernd Biehler	
☐	dirkb	Dirk Brosche	

Total: 4 Returned: 4 Displayed: 4 Selected: 0 Page 1 of 1

Figure 4-7 Add users to user Group dialog

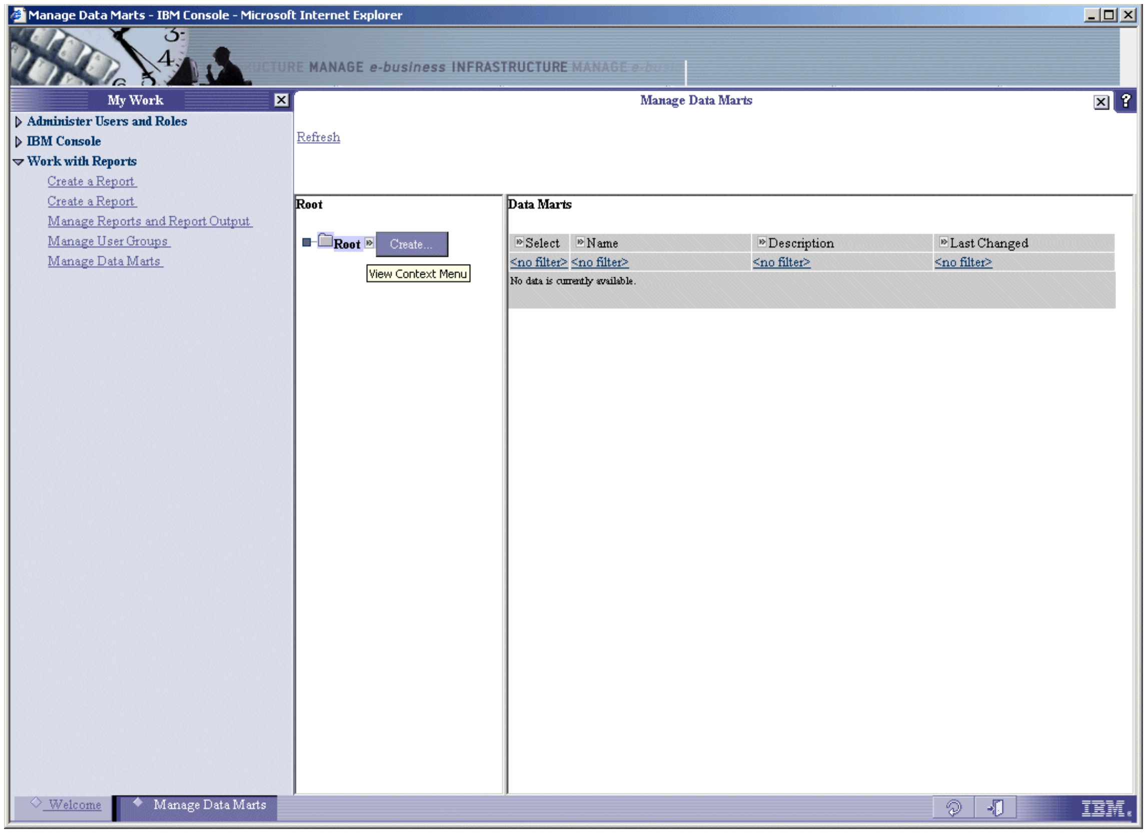
5. Select the users you want to have in your user group and click **OK**.
6. The last step is to select all data marts you want the users of this user group have access to. Click **Data Marts** and **Add** in the panel that appears. You will see a list of all data marts in your Report Interface (see Section 4.2.4, “Data marts” on page 100 for how to create a data mart in the RI).
7. Select the appropriate data marts and click **OK**. You now find all chosen data marts in the list.
8. Click **OK** to create the user group.

4.2.4 Data marts

Before you can run reports you have to set up the connection to one or more data marts. A data mart is a grouping of one or more star schemas, where all measurements, all possible types of measurements, and all components are stored. Using the data marts, the access of user groups to the physical data can also be controlled.

We now demonstrate how to create a data mart in the Tivoli Enterprise Data Warehouse Report Interface. In our example the star schemas are defined in a database named TWH_MART.

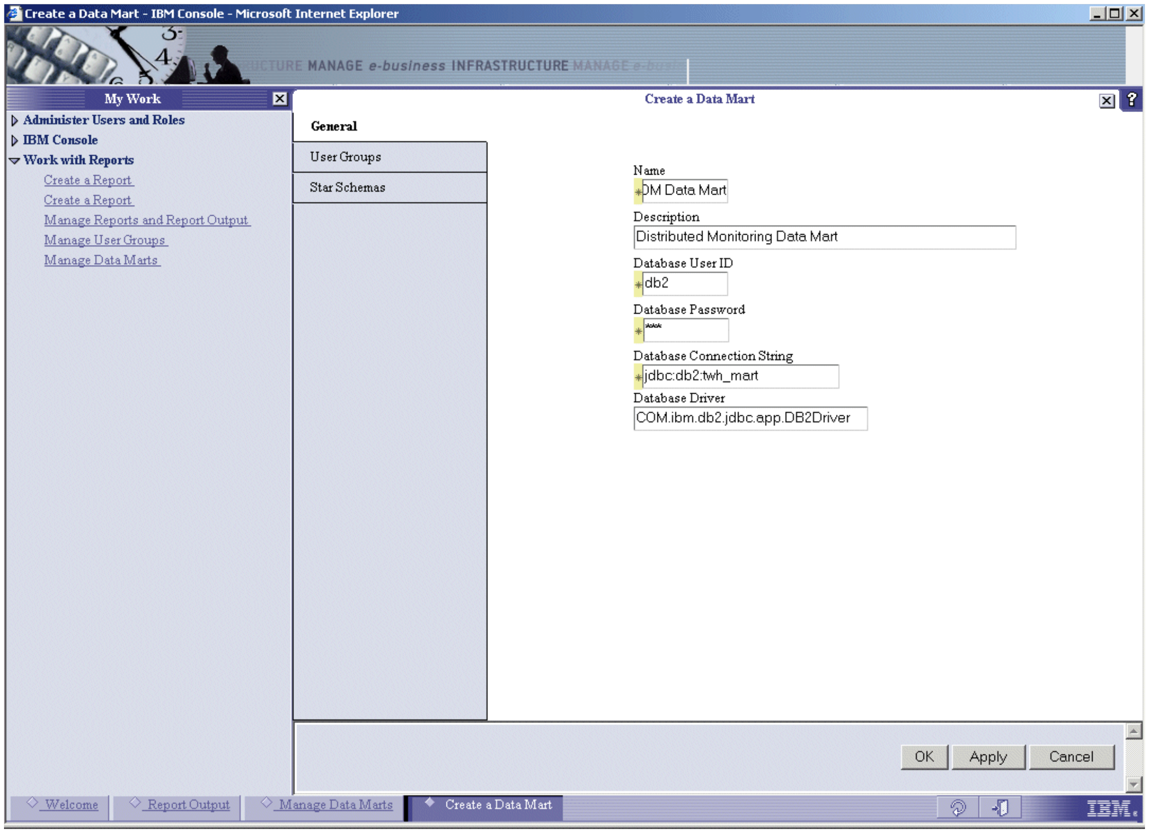
1. Choose **Manage Data Marts** from the menu, and **Work with Reports** in the left-hand side pane (Figure 4-9 on page 101).



Select	Name	Description	Last Changed
<no filter>	<no filter>	<no filter>	<no filter>
No data is currently available.			

Figure 4-8 Manage Data Marts dialog - Create a new data mart

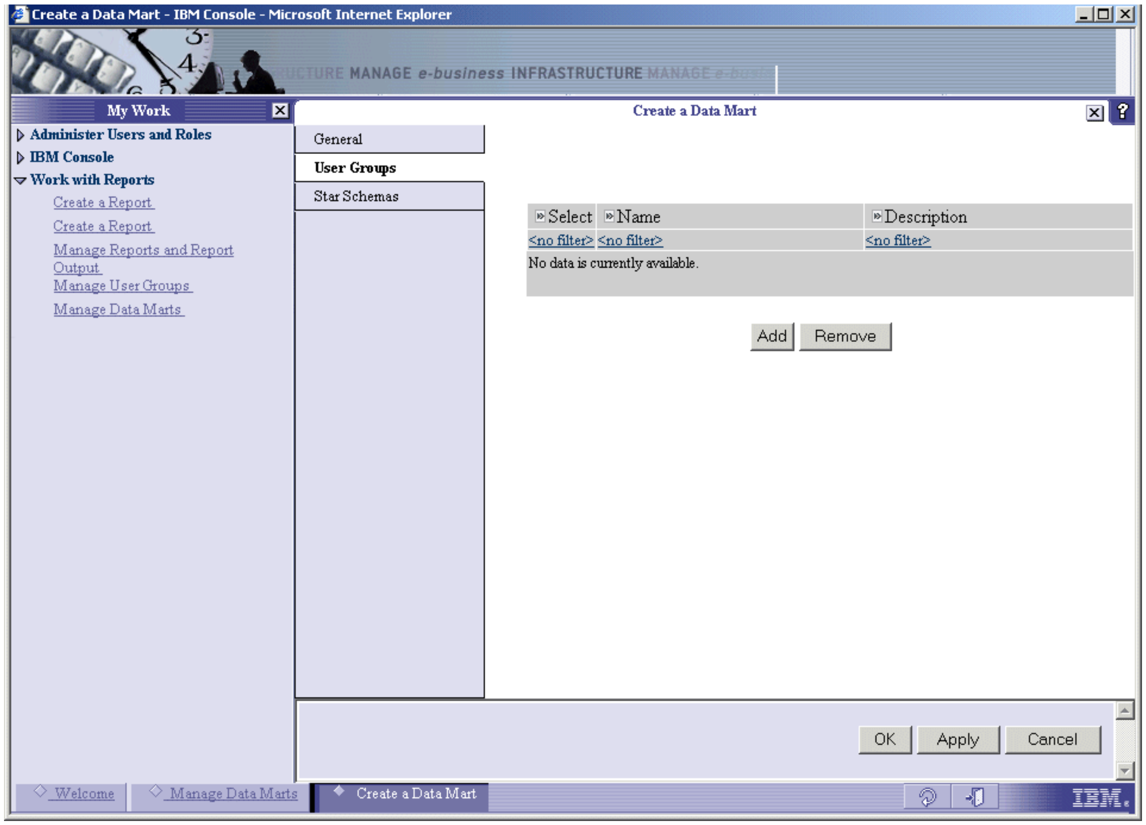
2. Select **Create** from the pop-up menu of Root. The Create a Data Mart dialog will appear.



Name: DM Data Mart
Description: Distributed Monitoring Data Mart
Database User ID: db2
Database Password: ***
Database Connection String: jdbc:db2:twh_mart
Database Driver: COM.ibm.db2.jdbc.app.DB2Driver

Figure 4-9 Create a Data Mart dialog - General information

3. In the dialog Create a Data Mart you have to define a unique name for the data mart and a useful description. The name of the data mart should follow the convention <Name of Data Mart> Data Mart. Furthermore, you have to type in the database user and password for the database where the star schemas are stored (Figure 4-10 on page 102).
4. The database connection string is specific to the database driver. For DB2, the connection string is jdbc:db2:database alias. For example, if your data mart database alias is TWH_MART, the database connection string should be jdbc:db2:twh_mart.
5. The entry Database Driver is for future use, when databases other than DB2 are supported. Leave this field unchanged.



No data is currently available.

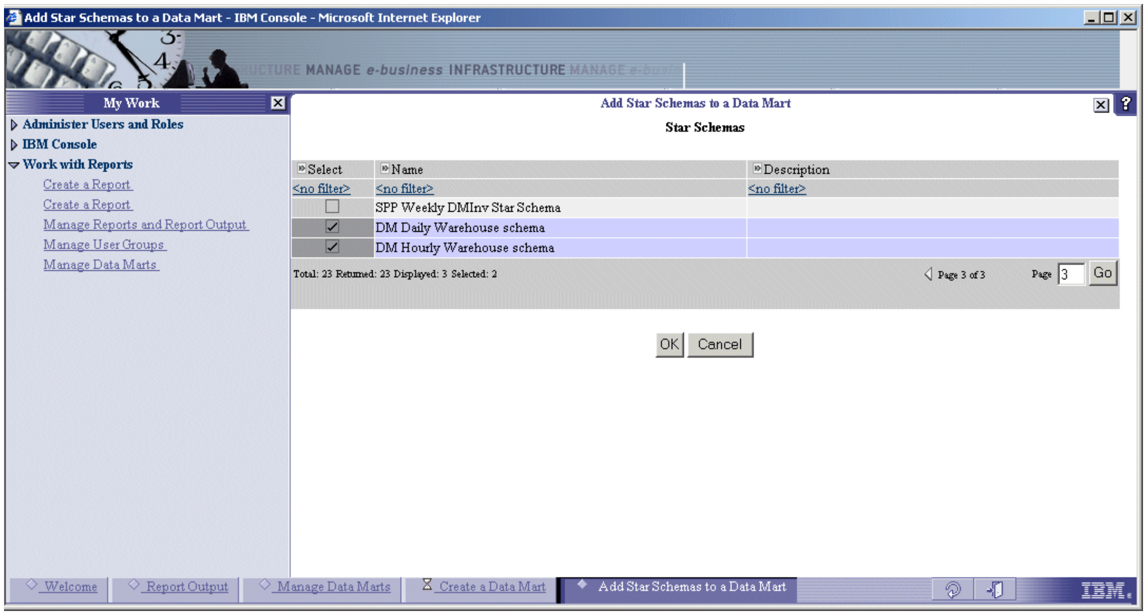
Figure 4-10 Create a Data Mart dialog - User Groups

In the panel User Groups you have to assign one or more user groups to the data mart. The users belonging to the assigned groups will be able to choose star schemas from this data mart to create reports.

6. To add more user groups press **Add** and a choice of all available user groups appears.
7. Select the user groups you want to have access to the data mart and click **OK**. All chosen user groups now appear in the User Groups panel.

   The next step is to add star schemas from the database.
8. Click on **Star Schemas** and then **Add** in the appearing dialog.

   You will see a list of all star schemas in the data mart (Figure 4-11 on page 103).


Add Star Schemas to a Data Mart

Star Schemas

Select	Name	Description
<no filter>	<no filter>	<no filter>
☐	SPP Weekly DMInv Star Schema	
✓	DM Daily Warehouse schema	
✓	DM Hourly Warehouse schema	

Total: 23 Returned: 23 Displayed: 3 Selected: 2

Figure 4-11 Add Star Schemas to a Data Mart dialog

The star schemas used with Tivoli Enterprise Data Warehouse have to follow the naming convention <product code> <time granularity> <description> star schema.

In the above example we see star schemas belonging to the Server Performance Prediction (SPP) warehouse pack and to the Distributed Monitoring (DM) warehouse pack. The second word describes the aggregation of data, for example, the hourly schema contains all measurements (hourly is, by default, the lowest granularity held in Tivoli Enterprise Data Warehouse). The daily schema contains data that is rolled up to daily values and so on.

Note: The Server Performance Prediction warehouse pack is a used in our redbook as an example Tivoli Enterprise Data Warehouse warehouse pack implementation. It is not an official product.

The star schemas in a data mart database have to be defined and can be seen in the Data Warehouse Center. As the structure of the star schemas is important for the understanding of the reports, it will be explained in Section 4.3.1, “How star schemas are used to create reports” on page 104.

9. In our example, we chose the DM star schemas that contain hourly data and daily rolled up data.

10. Click **OK** after you made your selection of star schemas.

    You will now see the selected star schemas in the Create a Data Mart dialog.

11. To finish the creation of the data mart now click **OK** in this dialog. The new data mart now appears in the list of data marts. If not, click **Refresh** in the upper panel. If the new data mart still does not appear in the list, there may be problems with the database connection. Refer to Chapter 10, “Troubleshooting and maintenance” on page 283 if this is the case.

4.3 Types of reports

The Report Interface of Tivoli Enterprise Data Warehouse provides three types of reports:

- Summary
- Extreme case
- Health check

In this section we briefly introduce the three types of reports of the RI and demonstrate how to create reports. However, we focus on star schemas first, since the understanding of the basic structure of a star schema will help you to understand how to create a meaningful report from your data.

4.3.1 How star schemas are used to create reports

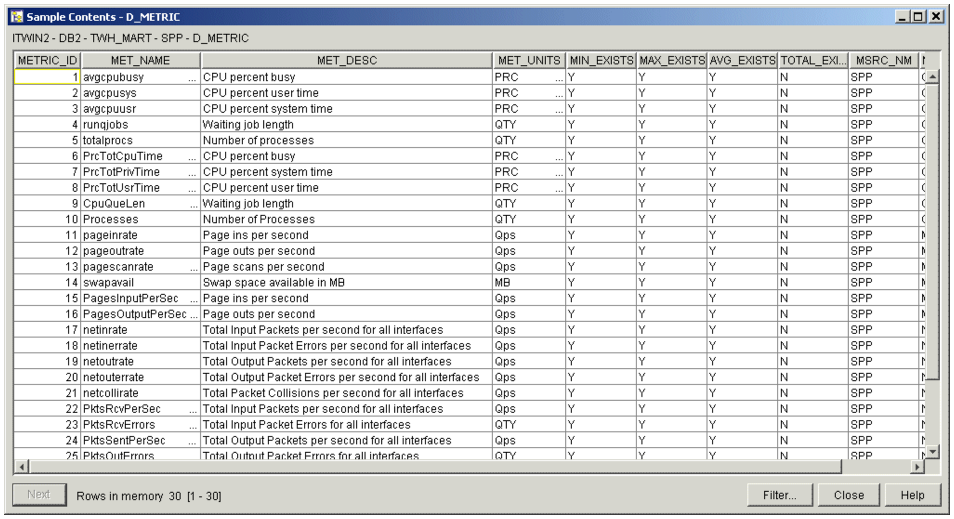
The reports generated in the RI are translated to SQL queries, which run against the star schemas. Thus we want to shed light on star schemas before we go into detail about reports.

A star schema consists of:

- One metric dimension table
- One or more component dimension tables
- One fact table

The metric dimension table

The metric dimension table (the name must end with METRIC) contains information about metrics; for example, the measured items like CPU usage or available swap space. Figure 4-12 on page 105 shows an example of a metric dimension table.



ITWIN2 - DB2 - TWH_MART - SPP - D_METRIC

METRIC_ID	MET_NAME	MET_DESC	MET_UNITS	MIN_EXISTS	MAX_EXISTS	AVG_EXISTS	TOTAL_EXI...	MSRC_NM
1	avgcpubusy ...	CPU percent busy	PRC ...	Y	Y	Y	N	SPP
2	avgcpusys	CPU percent user time	PRC ...	Y	Y	Y	N	SPP
3	avgcpuusr	CPU percent system time	PRC ...	Y	Y	Y	N	SPP
4	runqjobs	Waiting job length	QTY	Y	Y	Y	N	SPP
5	totalprocs	Number of processes	QTY	Y	Y	Y	N	SPP
6	PrcTotCpuTime ...	CPU percent busy	PRC ...	Y	Y	Y	N	SPP
7	PrcTotPrivTime ...	CPU percent system time	PRC ...	Y	Y	Y	N	SPP
8	PrcTotUsrTime ...	CPU percent user time	PRC ...	Y	Y	Y	N	SPP
9	CpuQueLen ...	Waiting job length	QTY	Y	Y	Y	N	SPP
10	Processes	Number of Processes	QTY	Y	Y	Y	N	SPP
11	pageinrate	Page ins per second	Qps	Y	Y	Y	N	SPP
12	pageoutrate	Page outs per second	Qps	Y	Y	Y	N	SPP
13	pagescanrate ...	Page scans per second	Qps	Y	Y	Y	N	SPP
14	swapavail	Swap space available in MB	MB	Y	Y	Y	N	SPP
15	PagesInputPerSec ...	Page ins per second	Qps	Y	Y	Y	N	SPP
16	PagesOutputPerSec ...	Page outs per second	Qps	Y	Y	Y	N	SPP
17	netinrate	Total Input Packets per second for all interfaces	Qps	Y	Y	Y	N	SPP
18	netinerrate	Total Input Packet Errors per second for all interfaces	Qps	Y	Y	Y	N	SPP
19	netoutrate	Total Output Packets per second for all interfaces	Qps	Y	Y	Y	N	SPP
20	netouterrate	Total Output Packet Errors per second for all interfaces	Qps	Y	Y	Y	N	SPP
21	netcollirate	Total Packet Collisions per second for all interfaces	Qps	Y	Y	Y	N	SPP
22	PktsRcvPerSec ...	Total Input Packets per second for all interfaces	Qps	Y	Y	Y	N	SPP
23	PktsRcvErrors ...	Total Input Packet Errors for all interfaces	QTY	Y	Y	Y	N	SPP
24	PktsSentPerSec ...	Total Output Packets per second for all interfaces	Qps	Y	Y	Y	N	SPP
25	PktsOutErrors	Total Output Packet Errors for all interfaces	QTY	Y	Y	Y	N	SPP

Rows in memory 30 [1 - 30]

Figure 4-12 Example metric dimension table

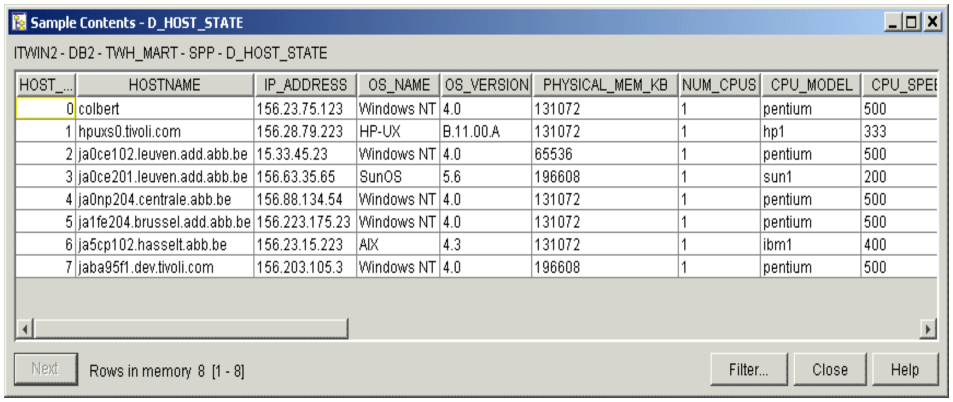
Each metric has a unique name (column MET_NAME), which can be chosen when a report is created. The unique metric_id is referred to in the facts table; for example, each measurement in the facts table has a metric_id. The rest of the columns describe the properties of each metric.

The component dimension table

The component dimension table(s) contain information about the components against which the measurements run, e.g., servers, databases, or trouble tickets. The name starts with D_ to signify a dimension table. Component dimension tables that are not the metric dimension table must not end in _METRIC.

Component dimension tables contain component attributes (column name=component attribute). These attributes are displayed in the Report Interface so that you can filter on them. If there are more than 27 values for a particular component attribute, they are not displayed in the Report Interface for filtering.

Figure 4-13 on page 106 displays an example of a component dimension table. The components in this example are hosts.


Sample Contents - D_HOST_STATE

ITWIN2 - DB2 - TWH_MART - SPP - D_HOST_STATE

HOST_...	HOSTNAME	IP_ADDRESS	OS_NAME	OS_VERSION	PHYSICAL_MEM_KB	NUM_CPUS	CPU_MODEL	CPU_SPEI
0	colbert	156.23.75.123	Windows NT	4.0	131072	1	pentium	500
1	hpuxs0.tivoli.com	156.28.79.223	HP-UX	B.11.00.A	131072	1	hp1	333
2	ja0ce102.leuven.add.abb.be	15.33.45.23	Windows NT	4.0	65536	1	pentium	500
3	ja0ce201.leuven.add.abb.be	156.63.35.65	SunOS	5.6	196608	1	sun1	200
4	ja0np204.centrale.abb.be	156.88.134.54	Windows NT	4.0	131072	1	pentium	500
5	ja1fe204.brussel.add.abb.be	156.223.175.23	Windows NT	4.0	131072	1	pentium	500
6	ja5cp102.hasselt.abb.be	156.23.15.223	AIX	4.3	131072	1	ibm1	400
7	jaba95f1.dev.tivoli.com	156.203.105.3	Windows NT	4.0	196608	1	pentium	500

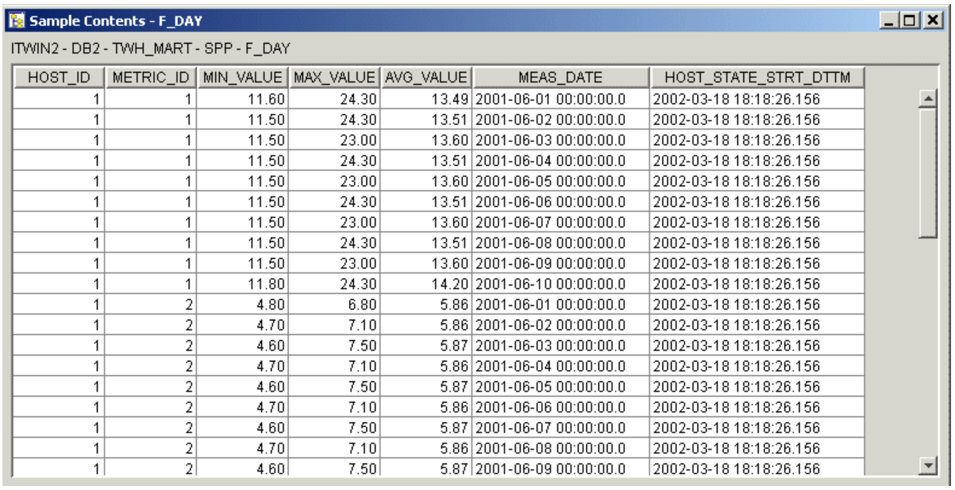
Next Rows in memory 8 [1 - 8] Filter... Close Help

Figure 4-13 An example of a component dimension table

The fact table

The fact table contains the values of the measurements. The name starts with an F_ and ends with the aggregation period of the data. The fact table contains foreign keys to the metric dimension table (METRIC_ID) and the component dimension tables (in this example the HOST_ID), which built the star structure and allow the analysis of the measured values according to the different dimensions.

Figure 4-14 displays an example of a component dimension table.


Sample Contents - F_DAY

ITWIN2 - DB2 - TWH_MART - SPP - F_DAY

HOST_ID	METRIC_ID	MIN_VALUE	MAX_VALUE	AVG_VALUE	MEAS_DATE	HOST_STATE_STRT_DTTM
1	1	11.60	24.30	13.49	2001-06-01 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	24.30	13.51	2001-06-02 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	23.00	13.60	2001-06-03 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	24.30	13.51	2001-06-04 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	23.00	13.60	2001-06-05 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	24.30	13.51	2001-06-06 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	23.00	13.60	2001-06-07 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	24.30	13.51	2001-06-08 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.50	23.00	13.60	2001-06-09 00:00:00.0	2002-03-18 18:18:26.156
1	1	11.80	24.30	14.20	2001-06-10 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.80	6.80	5.86	2001-06-01 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.70	7.10	5.86	2001-06-02 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.60	7.50	5.87	2001-06-03 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.70	7.10	5.86	2001-06-04 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.60	7.50	5.87	2001-06-05 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.70	7.10	5.86	2001-06-06 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.60	7.50	5.87	2001-06-07 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.70	7.10	5.86	2001-06-08 00:00:00.0	2002-03-18 18:18:26.156
1	2	4.60	7.50	5.87	2001-06-09 00:00:00.0	2002-03-18 18:18:26.156

Figure 4-14 Example of a fact table

Reports as SQL statements

When you create reports using the Report Interface, an SQL query is created, which is used to extract the data needed for the report from the star schema. In principle, such an SQL statement has the following structure (SQL statements differ from one type of report to another, so Example 4-1 is just an example).

Example 4-1 SQL statement of a report

```
1  select <aggregation>(<value>),..
2  from D_HOST_STATE, D_METRIC, F_HOUR
3  where
4  D_HOST_STATE.HOST_ID = F_HOUR.HOST_ID AND
5  F_HOUR.METRIC_ID = D_METRIC.METRIC_ID AND
6  D_HOST_STATE.STATE_STRT_DTTM = F_HOUR.HOST_STATE_STRT_DTTM AND
7  D_METRIC.met_name = '<metric name>' AND
8  F_HOUR.meas_hour between '<start time>' and '<end time>'
9  D_HOST_STATE.HOSTNAME = '<hostname filter>’
10 group by <value>
11 order by <value>
```

The parameters embraced by <> are completed by the choices made in the Create Report dialogs. The values selected in line 1 are, in general, the values displayed on the x-axis and y-axis of the reports. The tables are connected due to the foreign keys, as shown in Example 4-1 (lines 4, 5, and 6). We will refer to this example when we examine the reports in more detail.

Tip:

- Extreme case reports allow you to have one metric from one star schema. Once you have added a metric in the Metrics page in the Create Report notebook, you cannot add any more.
- Summary reports allow you to select up to five metrics from one star schema. All these metrics must be chosen at once, because you will not be allowed to add more metrics from the Metrics page if you have at least one metric already added.
- Health check reports allow you to select five metrics from multiple star schemas. Metrics can be added either individually or several at once (if they belong to the same star schema.

4.3.2 Summary reports

The summary report is typically used to display a many measurements versus many components relationship. The result is a table where the rows show components or groups of components and the columns typically show the measurements. Additionally, summary values for all components and all groups of components are shown. This kind of report can be used if, for example, you want to create an overview of the workload of servers or server groups.

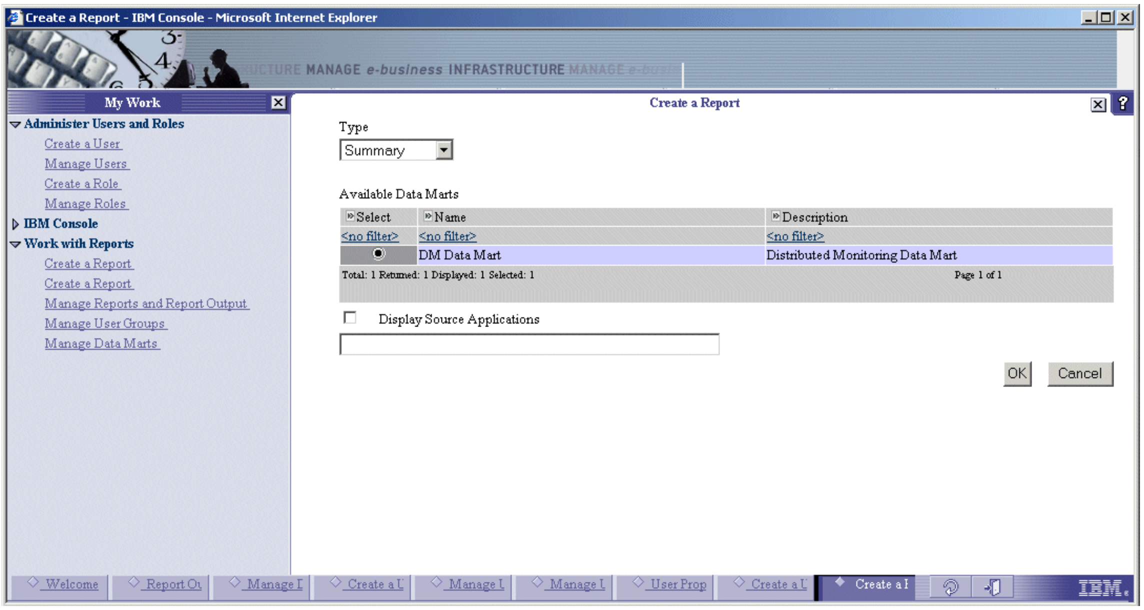
We now demonstrate how to create a new summary report.

1. Log in to your IBM Console as a user who has enough roles to create reports.
2. Expand **Work with Reports** and click **Create Report**.

> **Tip:** For users logged as superadmin, the portfolio and the context menus for reports contain multiple menus. For example, the context menus contain two Create a Report entries and three entries for Properties. Use the first entry in the list of duplicate entries, which represents the highest level role authorized to perform that function.

You will see the dialog shown in Figure 4-15.

.



Type: Summary

Available Data Marts

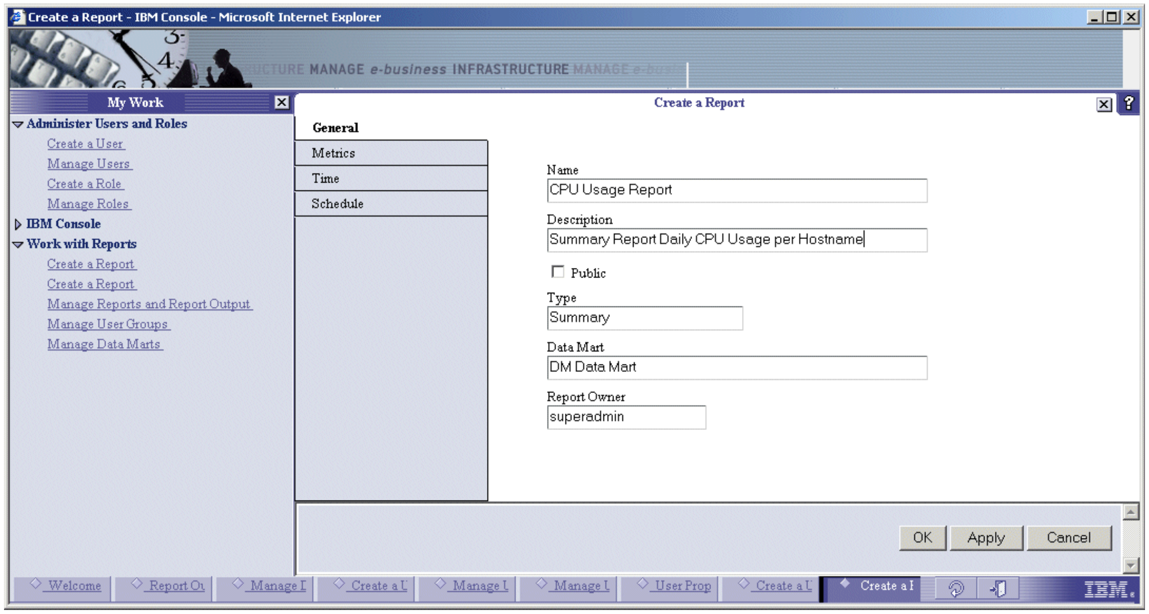
Select	Name	Description
<no filter>	<no filter>	<no filter>
●	DM Data Mart	Distributed Monitoring Data Mart

Total: 1 Returned: 1 Displayed: 1 Selected: 1   Page 1 of 1

☐ Display Source Applications

Figure 4-15 Create a Report dialog - Summary reports

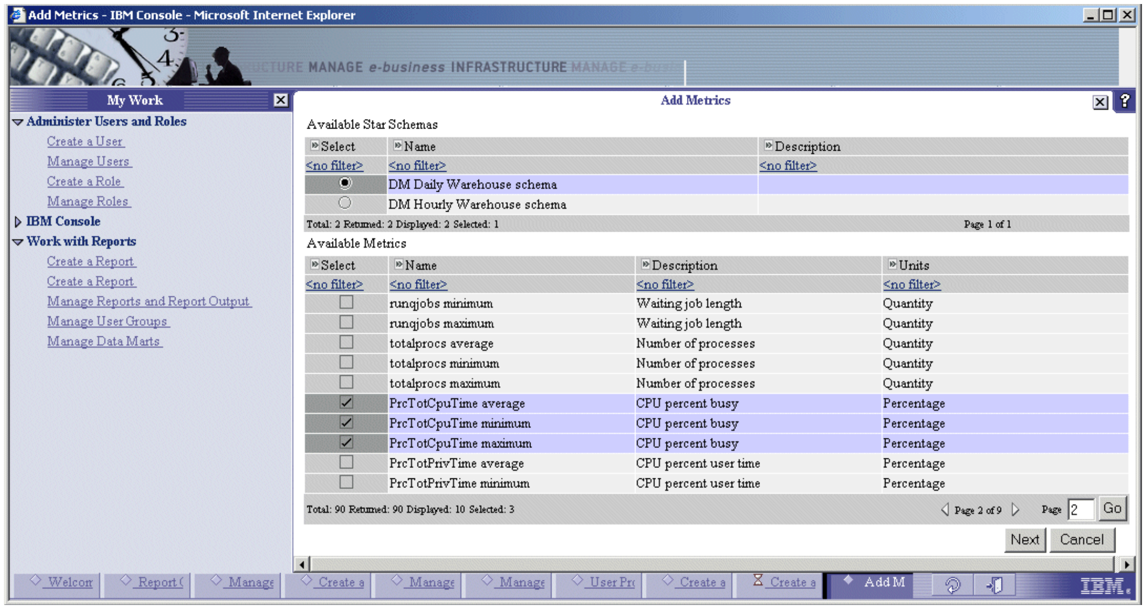
3. Choose **Summary** as report type and select the data mart that contains the star schema with the data for your report. You will later choose the metrics for your reports from the star schemas of this data mart.
4. In our example there is only the DM data mart. Select the data marts you need in your environment and click **OK** when ready. You will see the Create a Report notebook with its four panels (Figure 4-16).



Name: CPU Usage Report
Description: Summary Report Daily CPU Usage per Hostname
Type: Summary
Data Mart: DM Data Mart
Report Owner: superadmin

Figure 4-16 Create summary report - General dialog

5. First provide a meaningful name and description for the new report.
6. Check the **Public** button when you want to create a public report that can be seen and used by other users. You see the public entry only when you have sufficient roles to create public reports.
7. Click the **Metrics** tab. You will see the list of chosen metrics, which is still empty. In a summary report there are typically many metrics.
8. Click **Add** to choose metrics from the star schema. You will see the list of all star schemas of the chosen data mart.
9. Select one of them and you will see all available metrics of this star schema (Figure 4-17 on page 110).



Add Metrics

Available Star Schemas

Select	Name	Description
<no filter>	<no filter>	<no filter>
●	DM Daily Warehouse schema	
○	DM Hourly Warehouse schema	

Total: 2 Returned: 2 Displayed: 2 Selected: 1 Page 1 of 1

Available Metrics

Select	Name	Description	Units
<no filter>	<no filter>	<no filter>	<no filter>
☐	runqjobs minimum	Waiting job length	Quantity
☐	runqjobs maximum	Waiting job length	Quantity
☐	totalprocs average	Number of processes	Quantity
☐	totalprocs minimum	Number of processes	Quantity
☐	totalprocs maximum	Number of processes	Quantity
✓	PrcTotCpuTime average	CPU percent busy	Percentage
✓	PrcTotCpuTime minimum	CPU percent busy	Percentage
✓	PrcTotCpuTime maximum	CPU percent busy	Percentage
☐	PrcTotPrivTime average	CPU percent user time	Percentage
☐	PrcTotPrivTime minimum	CPU percent user time	Percentage

Total: 90 Returned: 90 Displayed: 10 Selected: 3 Page 2 of 9 Page 2 Go

Next Cancel

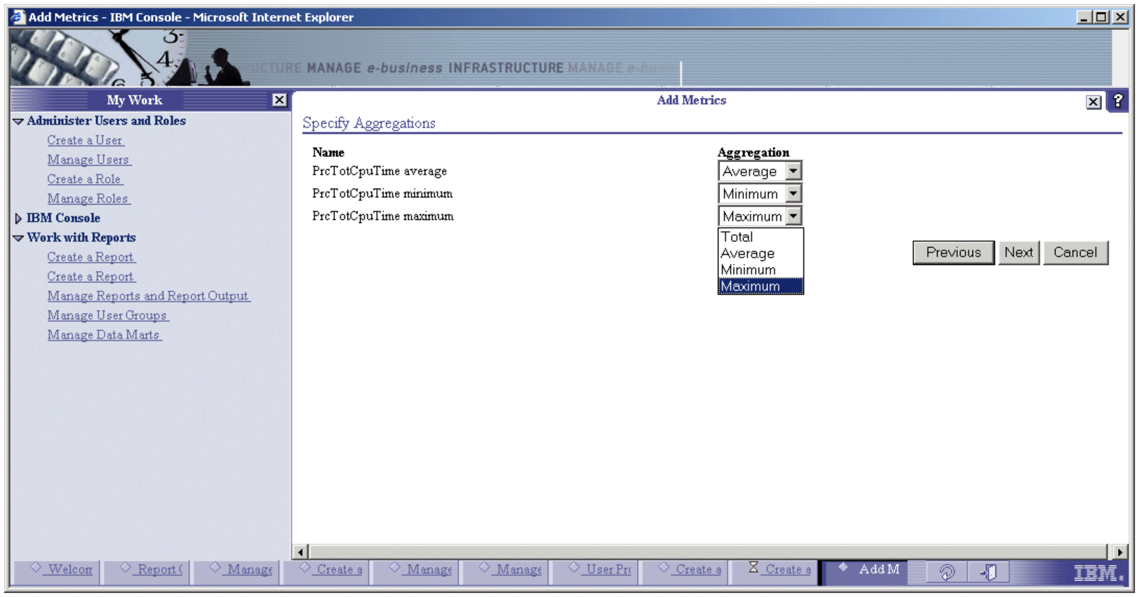
Figure 4-17 Create summary report - Add Metrics dialog

In Figure 4-17 you see that there is a minimum, maximum, and average type of each metric. These values are generated when the aggregation of the source data to hourly and daily data is done. *Each aggregation level has its own star schema with its own fact table.*

In a fact table each measurement can have an minimum, maximum, average, and total value. Which values are used depends on the application and can be defined in the D_METRIC table (see Figure 4-12 on page 105). When a value is used, a corresponding entry will appear in the available metrics list in the Report Interface.

10. Choose the metrics you need in your report and click **Next**.

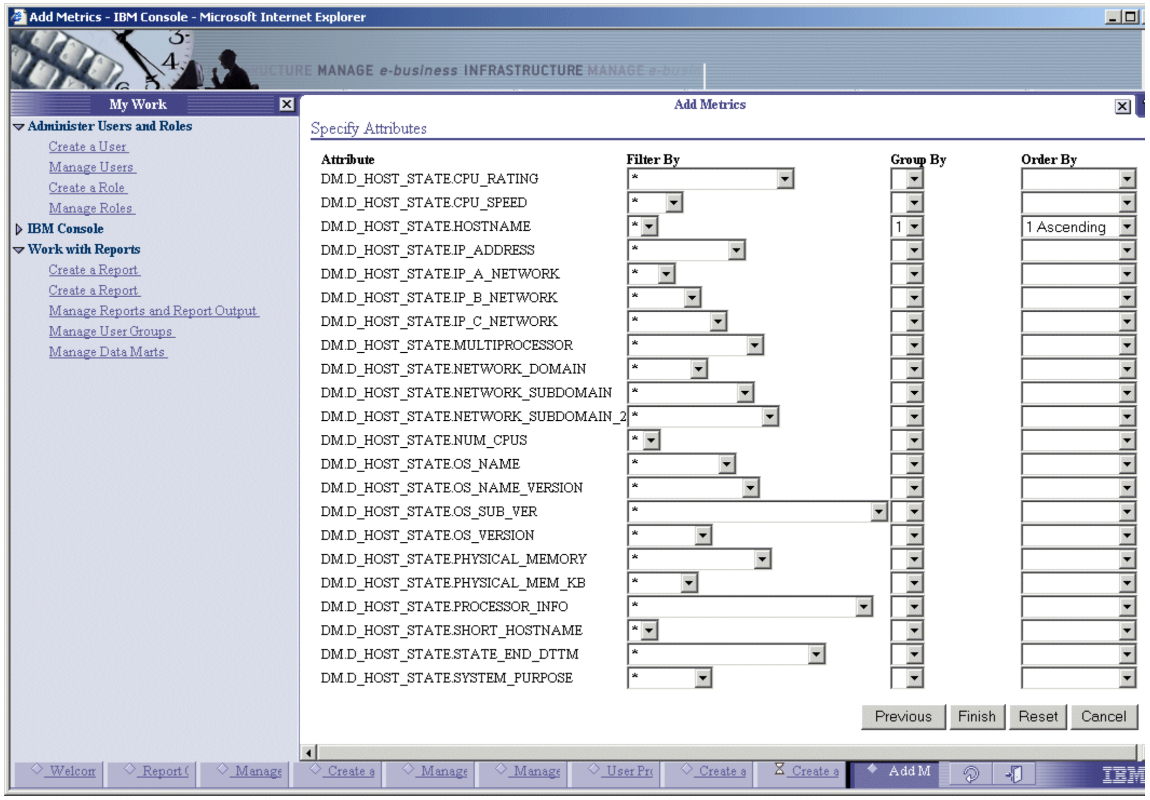
You will see the Specify Aggregations dialog (Figure 4-18 on page 111).



Name	Aggregation
PrcTotCpuTime average	Average
PrcTotCpuTime minimum	Minimum
PrcTotCpuTime maximum	Maximum

Figure 4-18 Create summary reports - Specify Aggregations dialog

In this dialog you have to choose an aggregation type for each chosen metric. A summary report covers a certain time window (defined later in this section). All measurements are aggregated over that time window. The aggregation type is defined here. Furthermore, there might be an aggregation over many components if the report is made for component groups instead of single components. Let us first consider these settings (Figure 4-19 on page 112).



Attribute	Filter By	Group By	Order By
DM.D_HOST_STATE.CPU_RATING	*		
DM.D_HOST_STATE.CPU_SPEED	*		
DM.D_HOST_STATE.HOSTNAME	*	1	1 Ascending
DM.D_HOST_STATE.IP_ADDRESS	*		
DM.D_HOST_STATE.IP_A_NETWORK	*		
DM.D_HOST_STATE.IP_B_NETWORK	*		
DM.D_HOST_STATE.IP_C_NETWORK	*		
DM.D_HOST_STATE.MULTIPROCESSOR	*		
DM.D_HOST_STATE.NETWORK_DOMAIN	*		
DM.D_HOST_STATE.NETWORK_SUBDOMAIN	*		
DM.D_HOST_STATE.NETWORK_SUBDOMAIN_2	*		
DM.D_HOST_STATE.NUM_CPUS	*		
DM.D_HOST_STATE.OS_NAME	*		
DM.D_HOST_STATE.OS_NAME_VERSION	*		
DM.D_HOST_STATE.OS_SUB_VER	*		
DM.D_HOST_STATE.OS_VERSION	*		
DM.D_HOST_STATE.PHYSICAL_MEMORY	*		
DM.D_HOST_STATE.PHYSICAL_MEM_KB	*		
DM.D_HOST_STATE.PROCESSOR_INFO	*		
DM.D_HOST_STATE.SHORT_HOSTNAME	*		
DM.D_HOST_STATE.STATE_END_DTTM	*		
DM.D_HOST_STATE.SYSTEM_PURPOSE	*		

Figure 4-19 Create summary report - Group by and filter dialog

Remember, all measurements are records in the fact table with a link to a metric and a certain set of component attributes. In the Specify Attributes dialog you will be able to filter component attributes and to group measurements to component groups.

With *Filter By*, you select only those records that match the values given in this field. In the resulting SQL statement, each chosen filter will result in a *where* clause (see line 9 in Example 4-1 on page 107).

The *Group By* function works as follows: If you choose one attribute in the Group By field, then all records with the same value for this attribute are taken together and aggregated according to the type chosen in the previous dialog. The result is one aggregated measurement for each different value of the chosen attribute.

Each entry in the Group By column will result in a group by clause in the SQL statement (see line 10 in Example 4-1 on page 107). The aggregation type will show up in the select part (line 1) where total is translated to sum.

11. In our example, we choose the host name as the sole Group By field, which will result in a table with one row for each host (Figure 4-19 on page 112). The Order By field (with the two possible values ascending or descending) defines the order in which the host names appear in the resulting report.

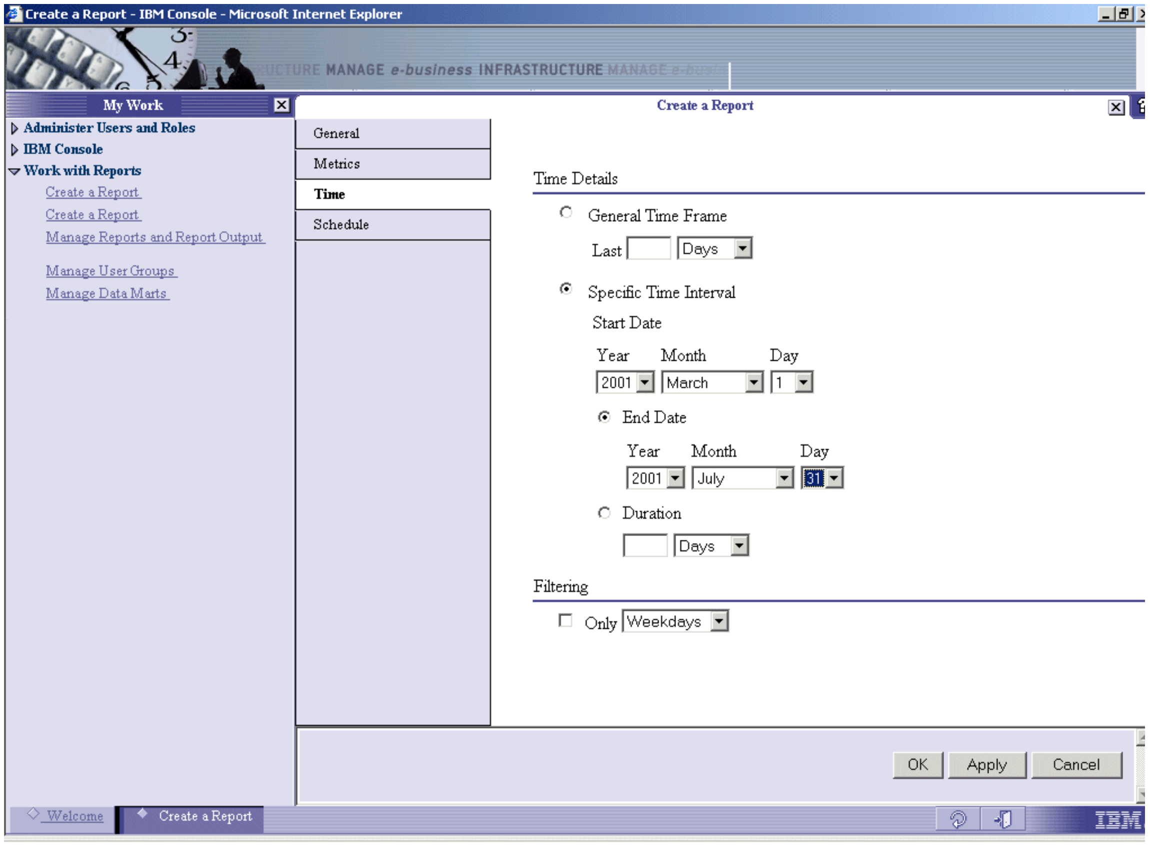
Tip: When creating a summary report, ensure that the Group By and Order By selections match. For example, if the user's selection for Order By is CPU_Speed = 1 (either ascending or descending), and there is no Group By selection for CPU_Speed, an error is displayed. But it is possible to have the following combination, for example:

	Group By	Order By
CPU_Speed	1	2 ascending
Memory	2	1 descending

12. We choose no filter in our example, thus all hosts in our database will appear in the report.

    We could filter, for example, on the domain name or IP sub-net to constrain possible values for host name. The possible choices of the filters are automatically populated from all values in the star schemas. If more than 27 distinct values exist you cannot filter on these attributes.

13. Click **Finish** to set up your metrics and click the **Time** pad (Figure 4-20 on page 114).



Time Details

General Time Frame

Last [ ] Days

[✓ Specific Time Interval]

Start Date

Year	Month	Day
2001	March	1

[✓ End Date]

Year	Month	Day
2001	July	31

Duration

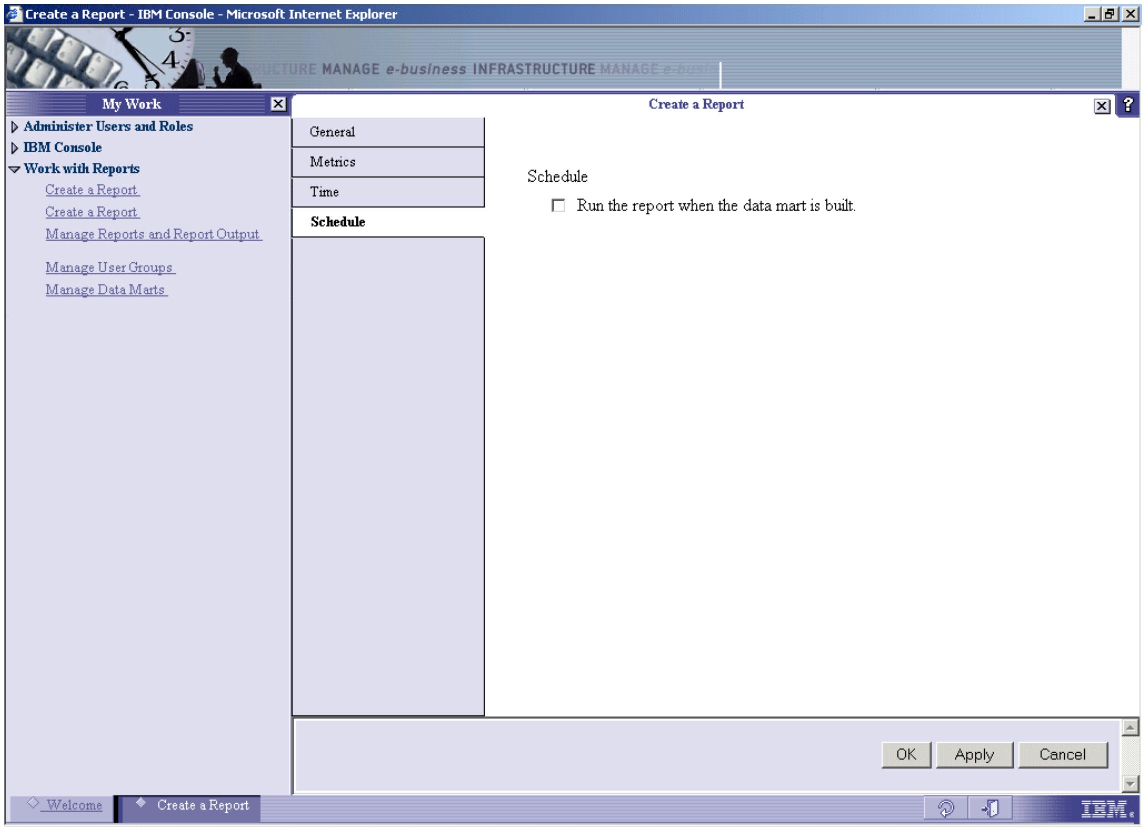
[ ] Days

Filtering

☐ Only Weekdays

Figure 4-20 Create summary report - Time dialog

14. In this dialog you have to choose the time interval for the report. In summary reports all measurements of the chosen time interval will be aggregated for all groups.



Schedule

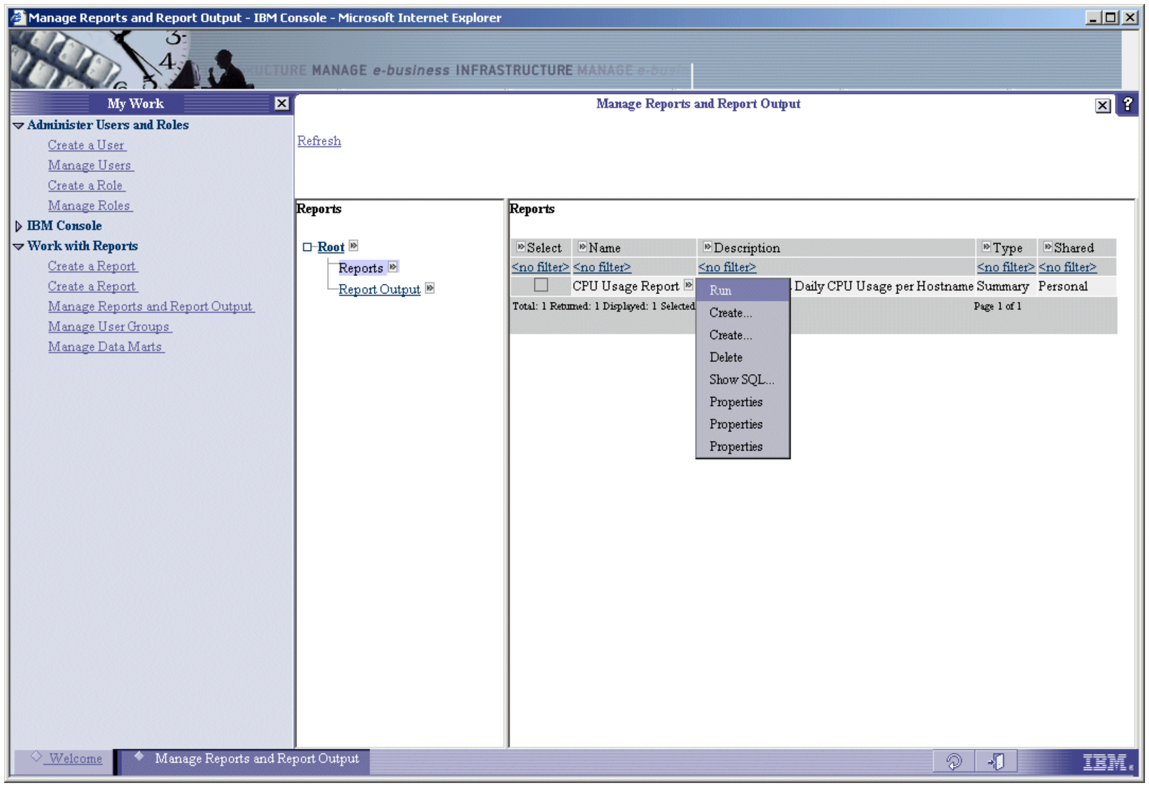
☐ Run the report when the data mart is built.

Figure 4-21 Create summary report - Schedule dialog

15. In the Schedule pad you can select **Run the report when the data mart is built**. A record is inserted into the RPI. The SSUpdated table in the TWH_MD database informs the report execution engine when a star schema has been updated, and in that case the report execution engine runs all scheduled reports that have been created from that star schema.
16. When all settings are done click **OK** to create the report.

    You should see a window displaying `Report created successfully`.
17. To see the report in the report list click **Refresh** and expand root in the Reports panel and click **Reports**.
18. Usually the reports are scheduled and run automatically when the data mart is built. However, you can run the report manually at any time by choosing **Run** from the reports pop-up menu (Figure 4-22 on page 116).

Tip: You might wonder why you see two Create, and three Properties menus in the reports pop-up menu. The reason for this is that in this exercise we are using the superadmin role, which combines all roles available. But you will not see this in a production environment, where you need create and use specific Report Interface users, instead of the superadmin.



Select	Name	Description	Type	Shared
<no filter>	<no filter>	<no filter>	<no filter>	<no filter>
☐	CPU Usage Report	Daily CPU Usage per Hostname	Summary	Personal

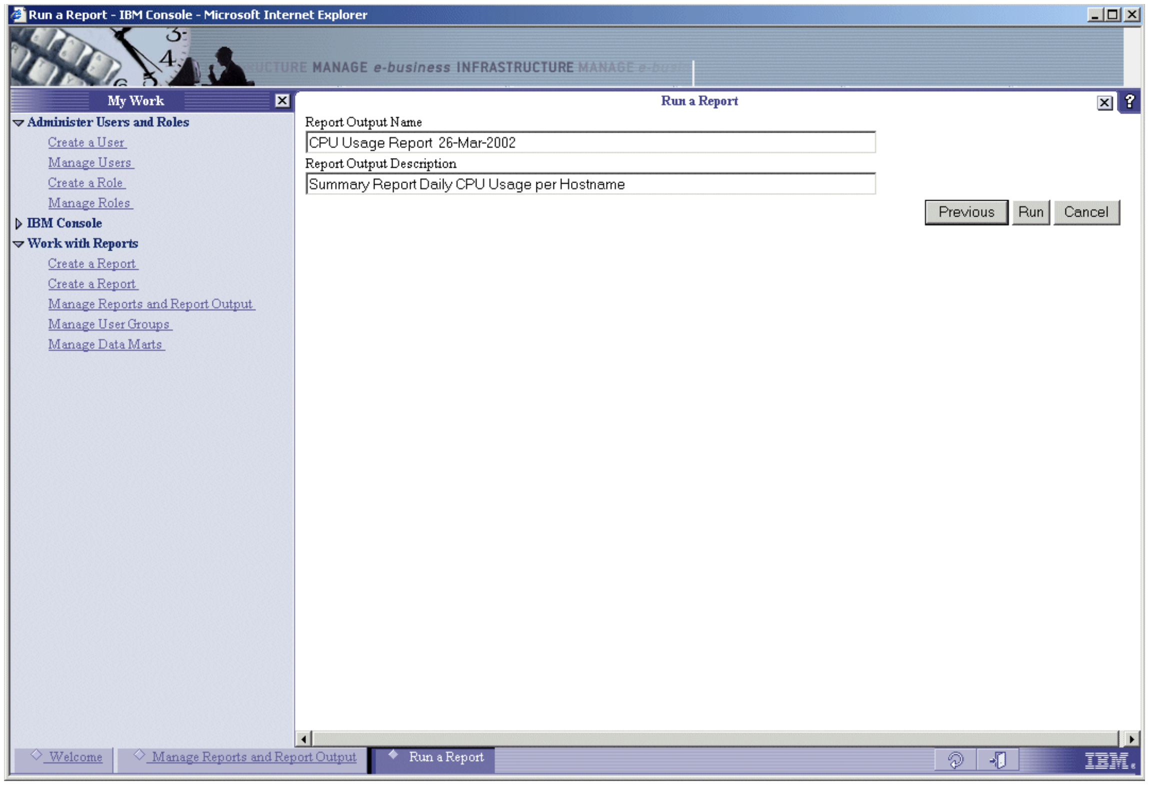
Total: 1 Returned: 1 Displayed: 1 Selected Page 1 of 1

Figure 4-22 Run a report

You will see the time dialog as shown in Figure 4-20 on page 114 again.

19. You can make changes to the time window of the report. Click **Next** to continue.

20. You can now provide a name and a description for the particular report output produced by this run. Click **Run** to continue (Figure 4-23 on page 117).

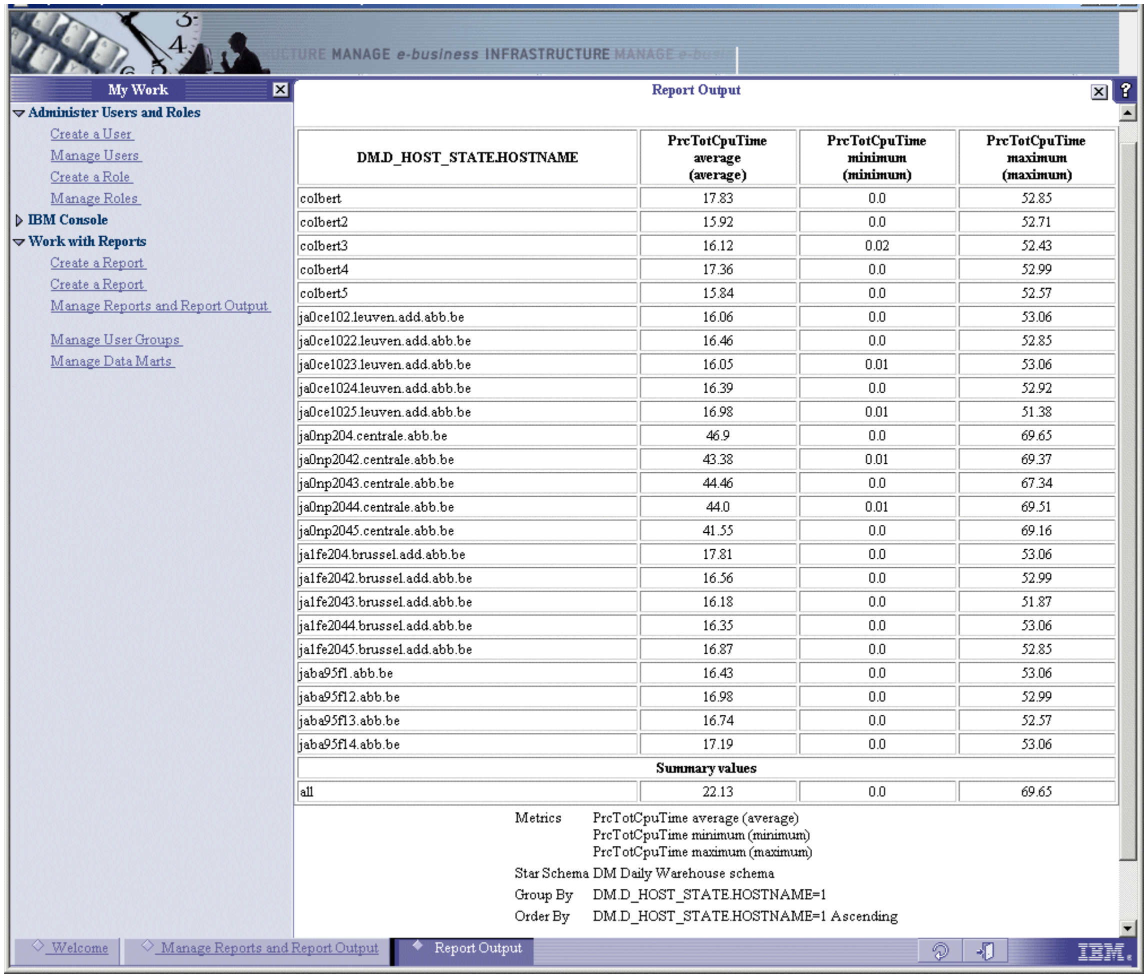


Report Output Name: CPU Usage Report 26-Mar-2002
Report Output Description: Summary Report Daily CPU Usage per Hostname

Figure 4-23 Run a report - Report output name and description

In Figure 4-24 on page 118 you now see the summary report with the three metrics, which were chosen in the Add Metrics dialog (Figure 4-17 on page 110), in the columns.

In the rows you find the host names that were chosen in the Group By entry in the Specify Attributes dialog (Figure 4-19 on page 112). All hosts of our example data mart are displayed as no filter was chosen.



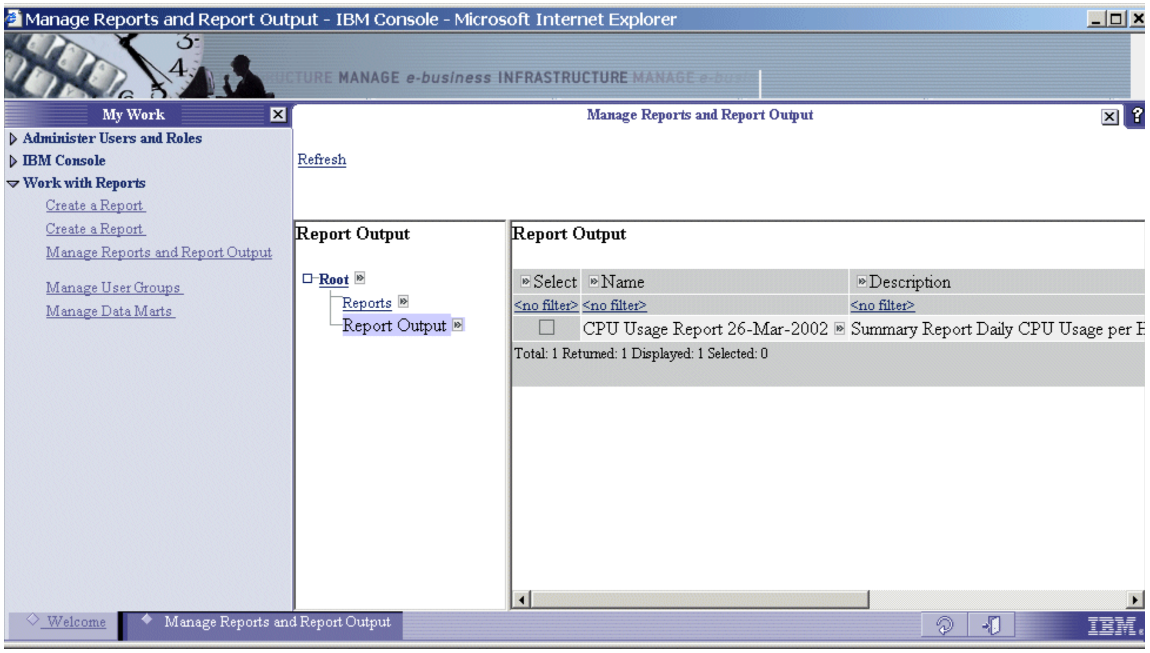
DM.D_HOST_STATE.HOSTNAME	PrcTotCpuTime average (average)	PrcTotCpuTime minimum (minimum)	PrcTotCpuTime maximum (maximum)
colbert	17.83	0.0	52.85
colbert2	15.92	0.0	52.71
colbert3	16.12	0.02	52.43
colbert4	17.36	0.0	52.99
colbert5	15.84	0.0	52.57
ja0ce102.leuven.add.abb.be	16.06	0.0	53.06
ja0ce1022.leuven.add.abb.be	16.46	0.0	52.85
ja0ce1023.leuven.add.abb.be	16.05	0.01	53.06
ja0ce1024.leuven.add.abb.be	16.39	0.0	52.92
ja0ce1025.leuven.add.abb.be	16.98	0.01	51.38
ja0np204.centrale.abb.be	46.9	0.0	69.65
ja0np2042.centrale.abb.be	43.38	0.01	69.37
ja0np2043.centrale.abb.be	44.46	0.0	67.34
ja0np2044.centrale.abb.be	44.0	0.01	69.51
ja0np2045.centrale.abb.be	41.55	0.0	69.16
ja1fe204.brussel.add.abb.be	17.81	0.0	53.06
ja1fe2042.brussel.add.abb.be	16.56	0.0	52.99
ja1fe2043.brussel.add.abb.be	16.18	0.0	51.87
ja1fe2044.brussel.add.abb.be	16.35	0.0	53.06
ja1fe2045.brussel.add.abb.be	16.87	0.0	52.85
jaba95f1.abb.be	16.43	0.0	53.06
jaba95f12.abb.be	16.98	0.0	52.99
jaba95f13.abb.be	16.74	0.0	52.57
jaba95f14.abb.be	17.19	0.0	53.06
Summary values			
all	22.13	0.0	69.65

Metrics	PrcTotCpuTime average (average) PrcTotCpuTime minimum (minimum) PrcTotCpuTime maximum (maximum)
Star Schema	DM Daily Warehouse schema
Group By	DM.D_HOST_STATE.HOSTNAME=1
Order By	DM.D_HOST_STATE.HOSTNAME=1 Ascending

Figure 4-24 Result of the summary report example

In the lowest row you find a summary value for all hosts. If more then one Group By attributes are chosen, then there will be summary values for all component groups.

21. You can now save this report output. You will find it in the folder Report Output (see Figure 4-25 on page 119).


Manage Reports and Report Output - IBM Console - Microsoft Internet Explorer

MANAGE e-business INFRASTRUCTURE
My Work
Administer Users and Roles
IBM Console
Work with Reports
Create a Report
Create a Report
Manage Reports and Report Output
Manage User Groups
Manage Data Marts
Manage Reports and Report Output
Refresh
Report Output
Root
Reports
Report Output
Welcome
Manage Reports and Report Output

Report Output

Select	Name	Description
<no filter>	<no filter>	<no filter>
☐	CPU Usage Report 26-Mar-2002	Summary Report Daily CPU Usage per H

Total: 1 Returned: 1 Displayed: 1 Selected: 0

Figure 4-25 The Report Output dialog

4.3.3 Extreme case reports

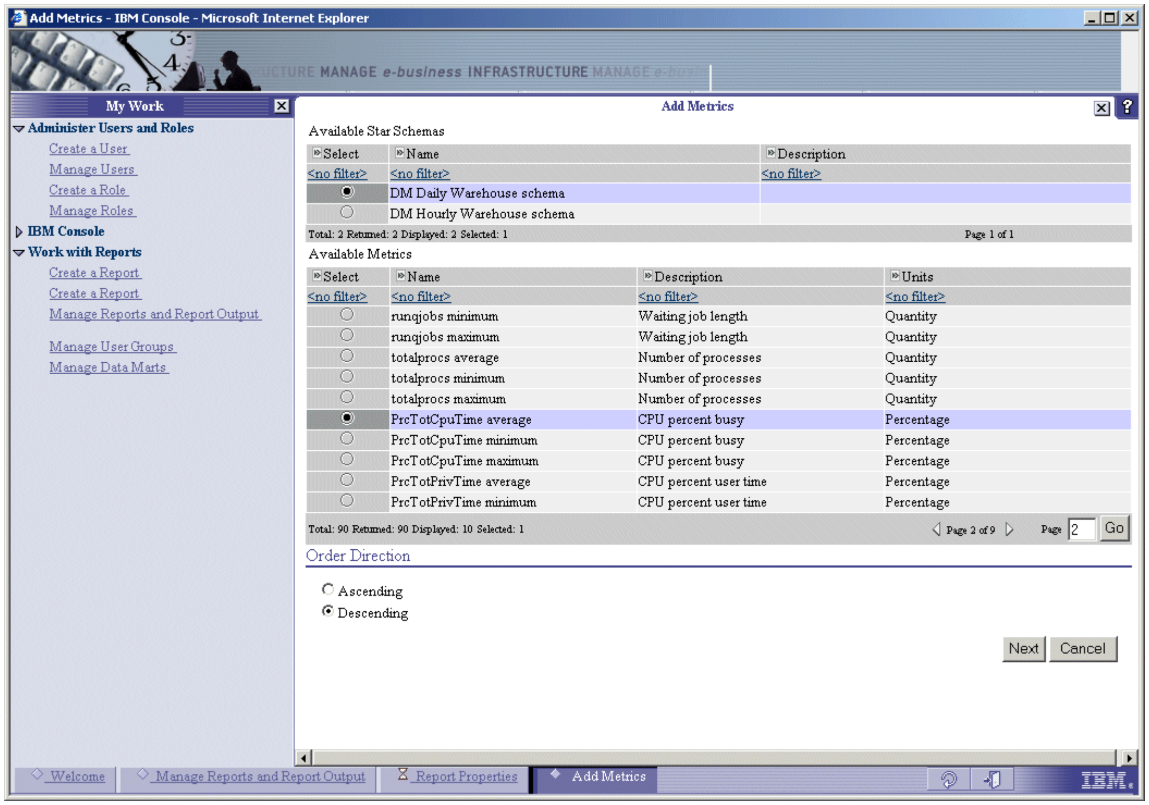
The extreme case report is a one-measurement versus a many-components report. With this type of report you can find the components or component groups with the highest or lowest values of a certain metric. The result will be a graph with the worst or best components in the x-axis and the corresponding metric values in the y-axis.

As an example, a customer might decide to spend some money to upgrade the CPU and memory of his weakest servers. With an extreme case report you can find the servers with the highest CPU usage or the least available swap space in the sub-domain belonging to the customer.

Let us show you how to create a new extreme case report:

1. Open your IBM Console, expand **Work with Reports** and click **Create Report**.
2. Choose **Extreme Case** from the type selection and proceed analogously to the previous section (see Figure 4-26 on page 120 to Figure 4-28 on page 122).

The first difference to the summary report is in the Add Metrics dialog (Figure 4-26).



Add Metrics

Available Star Schemas

Select	Name	Description
<no filter>	<no filter>	<no filter>
●	DM Daily Warehouse schema	
○	DM Hourly Warehouse schema	

Total: 2 Returned: 2 Displayed: 2 Selected: 1 Page 1 of 1

Available Metrics

Select	Name	Description	Units
<no filter>	<no filter>	<no filter>	<no filter>
○	runqjobs minimum	Waiting job length	Quantity
○	runqjobs maximum	Waiting job length	Quantity
○	totalprocs average	Number of processes	Quantity
○	totalprocs minimum	Number of processes	Quantity
○	totalprocs maximum	Number of processes	Quantity
●	PrcTotCpuTime average	CPU percent busy	Percentage
○	PrcTotCpuTime minimum	CPU percent busy	Percentage
○	PrcTotCpuTime maximum	CPU percent busy	Percentage
○	PrcTotPrivTime average	CPU percent user time	Percentage
○	PrcTotPrivTime minimum	CPU percent user time	Percentage

Total: 90 Returned: 90 Displayed: 10 Selected: 1 Page 2 of 9 Page 2 Go

Order Direction

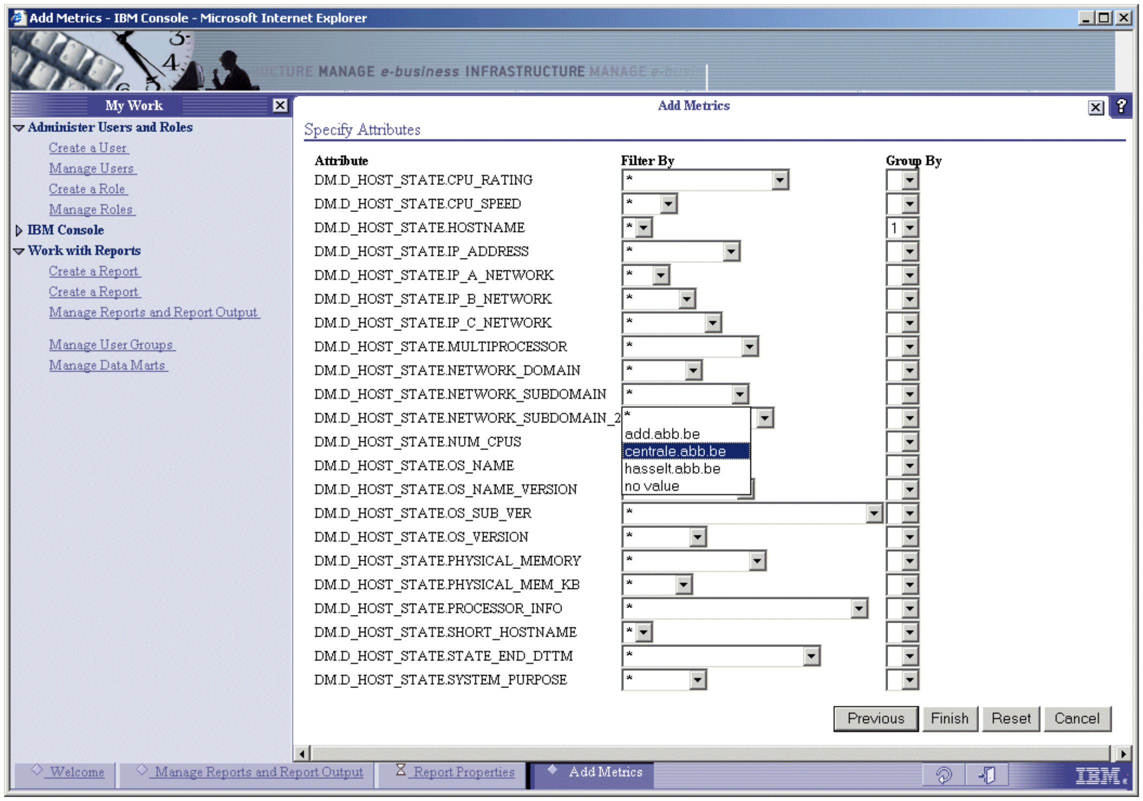
Ascending [✓ Descending]

Next Cancel

Figure 4-26 Create extreme case reports - Add Metrics dialog

In an extreme case report you can choose one metric only, for example, the metric with the extreme value.

There is one additional field compared to the summary report below the metric list. Here you can change the order direction. If you choose ascending order, the graph will start with the lowest value of the metrics. Conversely, you can use descending order to find the largest values. As you already chose the order of the graph in this dialog, the Order By choice will be missing in the Specify Attributes dialog (Figure 4-27 on page 121).


Add Metrics - IBM Console - Microsoft Internet Explorer

My Work

- Administer Users and Roles
  - Create a User
  - Manage Users
  - Create a Role
  - Manage Roles
- IBM Console
- Work with Reports
  - Create a Report
  - Create a Report
  - Manage Reports and Report Output
  - Manage User Groups
  - Manage Data Marts

Add Metrics

Specify Attributes

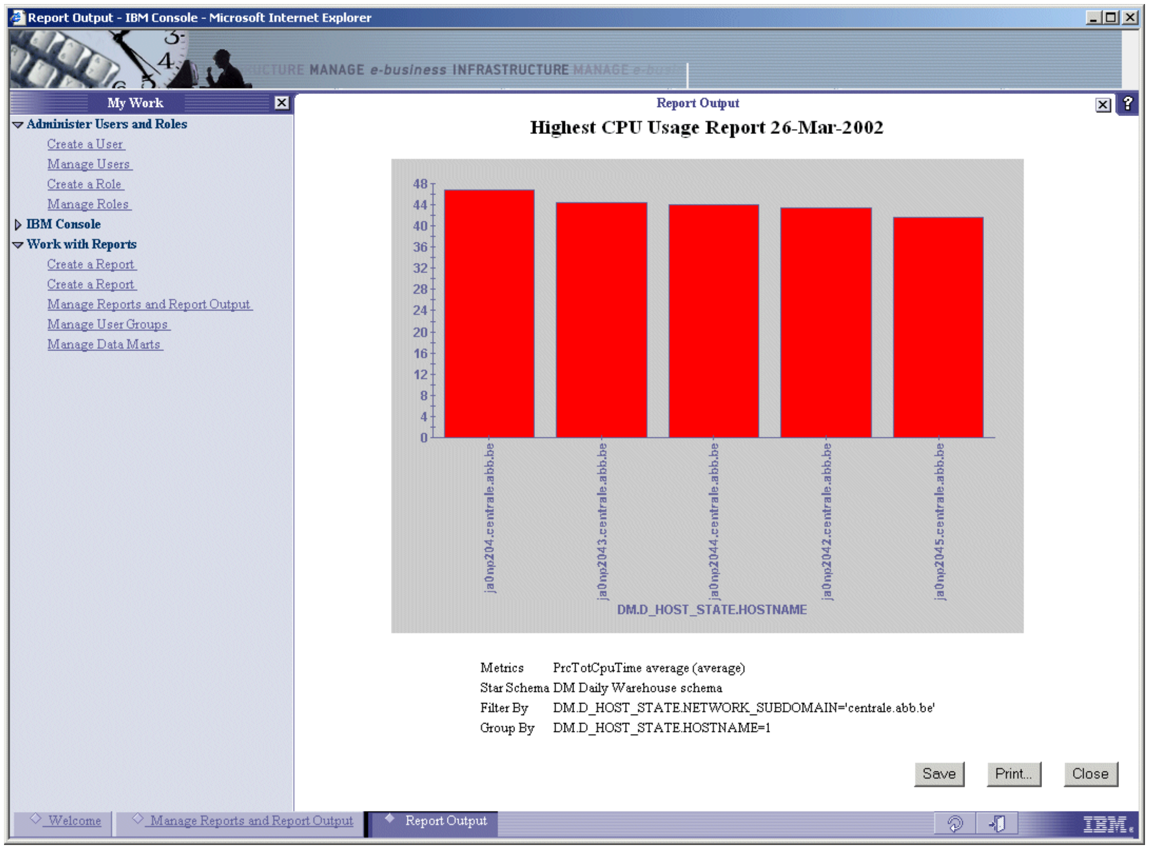
Attribute	Filter By	Group By
DM.D_HOST_STATE.CPU_RATING	*	
DM.D_HOST_STATE.CPU_SPEED	*	
DM.D_HOST_STATE.HOSTNAME	*	1
DM.D_HOST_STATE.IP_ADDRESS	*	
DM.D_HOST_STATE.IP_A_NETWORK	*	
DM.D_HOST_STATE.IP_B_NETWORK	*	
DM.D_HOST_STATE.IP_C_NETWORK	*	
DM.D_HOST_STATE.MULTIPROCESSOR	*	
DM.D_HOST_STATE.NETWORK_DOMAIN	*	
DM.D_HOST_STATE.NETWORK_SUBDOMAIN	*	
DM.D_HOST_STATE.NETWORK_SUBDOMAIN_2	* (add.abb.be, centrale.abb.be, hasselt.abb.be, no value)	
DM.D_HOST_STATE.NUM_CPUS		
DM.D_HOST_STATE.OS_NAME		
DM.D_HOST_STATE.OS_NAME_VERSION		
DM.D_HOST_STATE.OS_SUB_VER	*	
DM.D_HOST_STATE.OS_VERSION	*	
DM.D_HOST_STATE.PHYSICAL_MEMORY	*	
DM.D_HOST_STATE.PHYSICAL_MEM_KB	*	
DM.D_HOST_STATE.PROCESSOR_INFO	*	
DM.D_HOST_STATE.SHORT_HOSTNAME	*	
DM.D_HOST_STATE.STATE_END_DTTM	*	
DM.D_HOST_STATE.SYSTEM_PURPOSE	*	

Figure 4-27 Create extreme case report - Specify Attributes dialog

According to the above example, we want to find the host names of the servers with the highest average CPU usage in a domain.

3. We chose the host name as the Group by entry and the interesting sub-domain in the Filter by entry.
4. The procedure to finish and run this report is again the same as the summary report example (see Figure 4-20 on page 114 to Figure 4-23 on page 117).

   The result of the extreme case report displays the servers with the highest average CPU usage in the chosen time interval (Figure 4-28 on page 122). With the help of this report you can now decide which servers should be upgraded first.


Report Output - IBM Console - Microsoft Internet Explorer
My Work
Administer Users and Roles
Create a User
Manage Users
Create a Role
Manage Roles
IBM Console
Work with Reports
Create a Report
Create a Report
Manage Reports and Report Output
Manage User Groups
Manage Data Marts
Report Output
Highest CPU Usage Report 26-Mar-2002
48
44
40
36
32
28
24
20
16
12
8
4
0
ja0np204.centrale.abb.be
ja0np2043.centrale.abb.be
ja0np2044.centrale.abb.be
ja0np2042.centrale.abb.be
ja0np2045.centrale.abb.be
DM.D_HOST_STATE.HOSTNAME
Save
Print...
Close
Welcome
Manage Reports and Report Output
Report Output

Metrics	PrcTotCpuTime average (average)
Star Schema	DM Daily Warehouse schema
Filter By	DM.D_HOST_STATE.NETWORK_SUBDOMAIN='centrale.abb.be'
Group By	DM.D_HOST_STATE.HOSTNAME=1

Figure 4-28 Result of the extreme case report example

4.3.4 Health reports

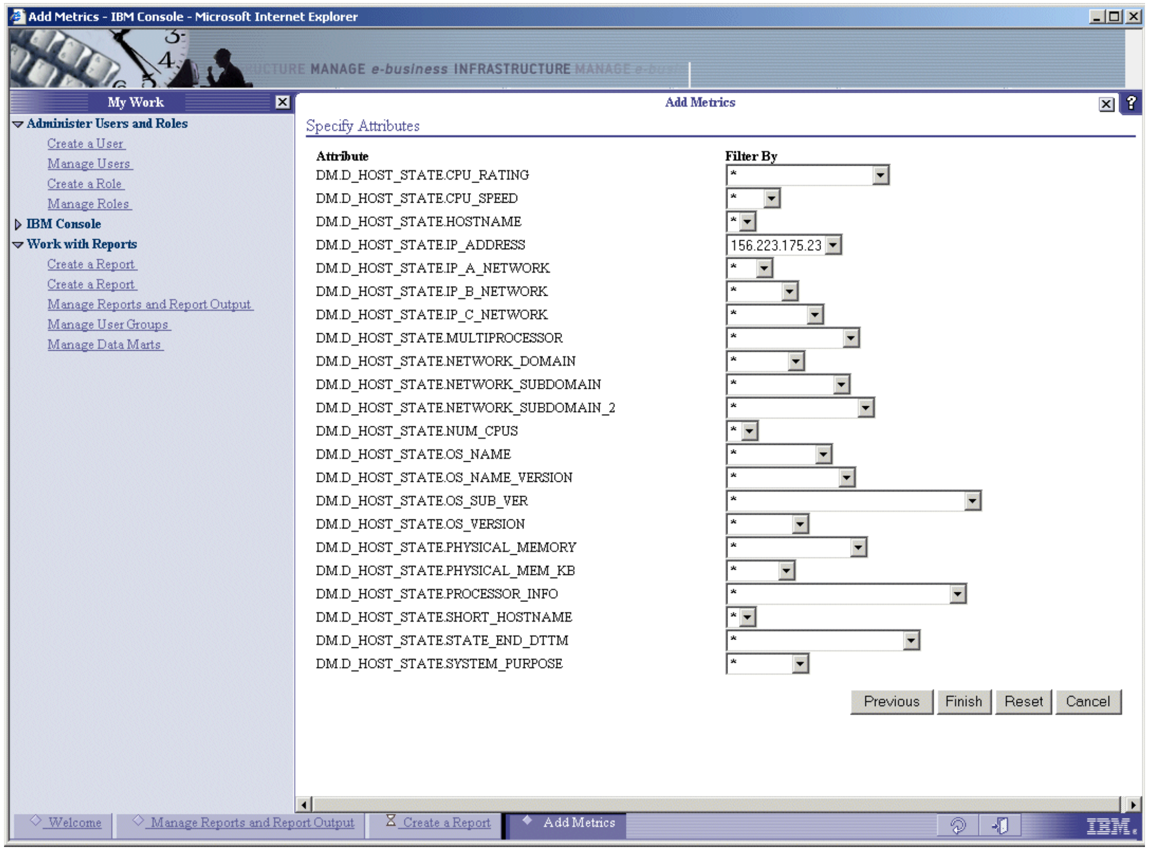
The health report (or health check report) is typically used to display many measurements against many components versus time relation. The result is a graph where the x-axis shows the time and the y-axis shows the measurements. This kind of report is used to show the time-development of a metric. This allows you to recognize a trend and to predict possible future health problems of a component.

As an example let us look at the CPU usage of a single server over a time period over three weeks.

1. To create a new health report open your IBM Console, expand **Work with Reports** and click **Create Report**.

2. Choose **Health Report** from the type selection and proceed analogous to Section 4.3.2, “Summary reports” on page 108 (see Figure 4-15 on page 108 to Figure 4-18 on page 111).
3. For our example we chose the same settings as in the create summary report example. Note that you can choose only five metrics a maximum for a health report.

   The first difference from the creation of a summary report appears in the Specify Attributes dialog (Figure 4-29).


Add Metrics - IBM Console - Microsoft Internet Explorer

MANAGE *e-business* INFRASTRUCTURE

My Work

- Administer Users and Roles
  - Create a User
  - Manage Users
  - Create a Role
  - Manage Roles
- IBM Console
- Work with Reports
  - Create a Report
  - Create a Report
  - Manage Reports and Report Output
  - Manage User Groups
  - Manage Data Marts

Add Metrics

Specify Attributes

Attribute	Filter By
DM.D_HOST_STATE.CPU_RATING	*
DM.D_HOST_STATE.CPU_SPEED	*
DM.D_HOST_STATE.HOSTNAME	*
DM.D_HOST_STATE.IP_ADDRESS	156.223.175.23
DM.D_HOST_STATE.IP_A_NETWORK	*
DM.D_HOST_STATE.IP_B_NETWORK	*
DM.D_HOST_STATE.IP_C_NETWORK	*
DM.D_HOST_STATE.MULTIPROCESSOR	*
DM.D_HOST_STATE.NETWORK_DOMAIN	*
DM.D_HOST_STATE.NETWORK_SUBDOMAIN	*
DM.D_HOST_STATE.NETWORK_SUBDOMAIN_2	*
DM.D_HOST_STATE.NUM_CPUS	*
DM.D_HOST_STATE.OS_NAME	*
DM.D_HOST_STATE.OS_NAME_VERSION	*
DM.D_HOST_STATE.OS_SUB_VER	*
DM.D_HOST_STATE.OS_VERSION	*
DM.D_HOST_STATE.PHYSICAL_MEMORY	*
DM.D_HOST_STATE.PHYSICAL_MEM_KB	*
DM.D_HOST_STATE.PROCESSOR_INFO	*
DM.D_HOST_STATE.SHORT_HOSTNAME	*
DM.D_HOST_STATE.STATE_END_DTTM	*
DM.D_HOST_STATE.SYSTEM_PURPOSE	*

Figure 4-29 Create health report - Specify Attributes dialog

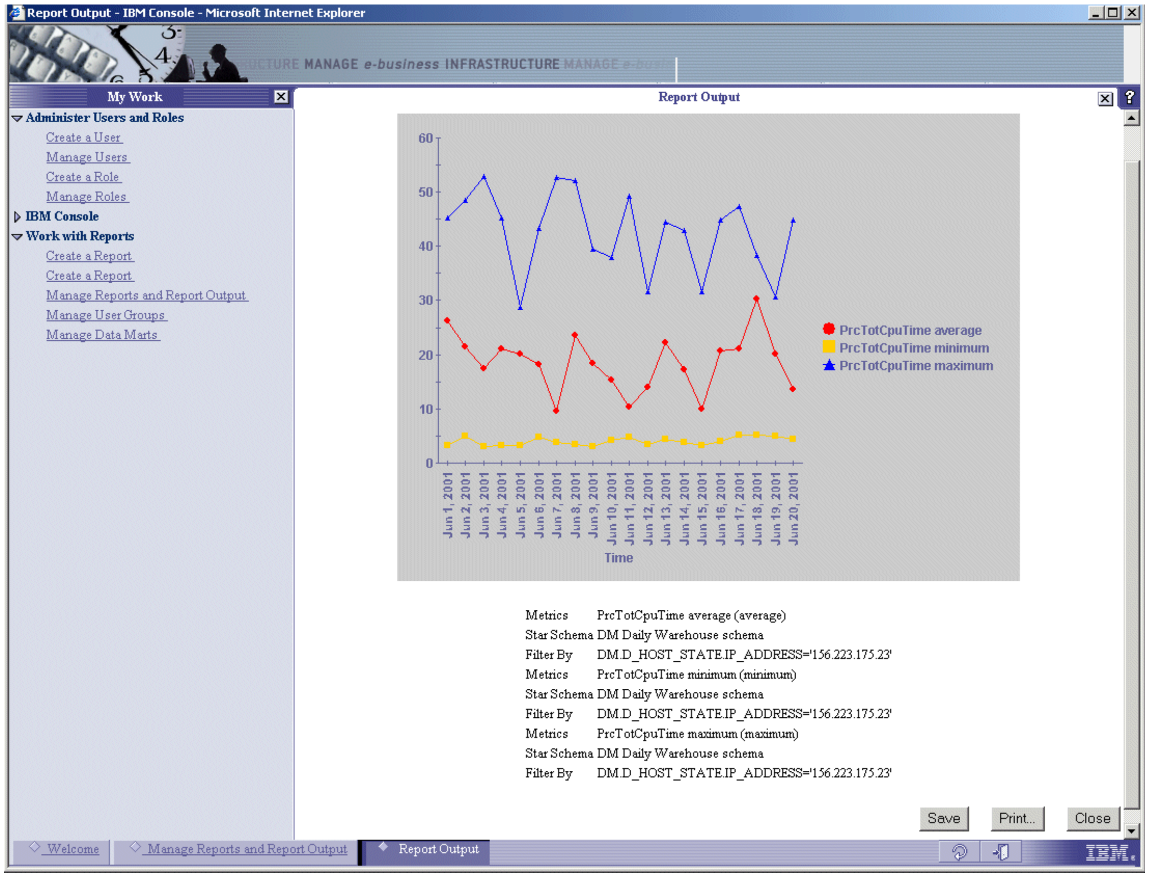
Health reports are always graphs of the type measurements versus time. For this reason no Group By or Order By is to be selected because it is always grouped by time and ordered by ascending time.

If no filter is selected the measurement is aggregated (minimum, maximum, average, or sum, according to the choice in the Specify Aggregations dialog) over all components. There can be many metrics displayed in one graph, but they all apply to the same component (host name or IP address) or group of components; thus it is not possible to compare different instances of the same component in one graph.

Tip: In health reports, it is possible to compare different components in one graph because the metrics on one graph can be selected from many star schemas, not only one.

If there are more than 27 different values for an attribute, no choices can be made for this particular attribute in the Specify Attributes dialog.

For example, Figure 4-30 on page 125 shows the development in time for the CPU usage of a single server over an interval of three weeks.



Metrics	PrcTotCpuTime average (average)
Star Schema	DM Daily Warehouse schema
Filter By	DM.D_HOST_STATE.IP_ADDRESS='156.223.175.23'
Metrics	PrcTotCpuTime minimum (minimum)
Star Schema	DM Daily Warehouse schema
Filter By	DM.D_HOST_STATE.IP_ADDRESS='156.223.175.23'
Metrics	PrcTotCpuTime maximum (maximum)
Star Schema	DM Daily Warehouse schema
Filter By	DM.D_HOST_STATE.IP_ADDRESS='156.223.175.23'

Figure 4-30 Result of the health report example

Tip: For health report choose your time window according to the aggregation level of your data. If there are too many data points on the x-axis the report becomes unreadable.

5

Integration of application data to central data repository

In this chapter we show how to integrate your application data to the central data repository. We will use two different data sources as sample applications, with each data source demonstrating different integration techniques. This chapter has the following sections:

- Why you might need to integrate your own data
- Methodology
- Case study 1- IBM Server Resource Management (SRM)
- Case study 2 - ABC Configuration Database
- Best practices

5.1 Why you might need to integrate your own data

There are two major reasons why you might need to integrate your own data into the Tivoli Data Warehouse. The first reason is to map an existing data source onto the central data repository schema. The second reason is to supplement the central data repository schema with additional data that does not exist in the current schema.

5.1.1 Mapping data to the existing schema

In a heterogeneous distributed server environment, it is not uncommon to find different tools used to support those servers. Each of these tools may have a data repository, where historical information is kept. These repositories have different data schemas, since they were designed by different vendors, and they may have metrics specific to that platform. While platform-specific reports are usually required, it is often useful to create enterprise level reports, which mix information from the various platforms. An example of this is to report performance utilization statistics of servers in a single report, regardless of the platform architecture.

By having a common schema in the Tivoli Data Warehouse central data repository, one is able to map the existing data sources into the repository. With the data in a single repository, common reports can be generated, creating a common look and feel between the reports, making the understanding of different platforms easier. This also reduces the resource requirements for platform-specific reporting.

In addition to having different data repositories caused by different platforms, there are frequently different tools deployed on the same platforms, but on different servers. This is usually the case when these servers are supported by different personnel and then consolidated at some future time (for example, companies that have grown through acquisition, where the servers have come from different companies, or where different support organizations have been allowed to select their own system management tools). By mapping these tools to the central data repository, common reports can be generated, masking the different tools used to create the data.

Lastly, one may want to convert existing servers to Tivoli based products. By using Tivoli Data Warehouse, historical data from existing servers can be loaded into the central data repository, so there is no data loss. This will allow one to convert over to a Tivoli product that uses the central data repository and not lose any data. While servers are in the process of converting, both the new and old data sources can be used to generate reports in the new common format.

The following section is an example of mapping data to the existing schema.

Case study 1 - Server Resource Management (SRM)

In this section we suggest a Tivoli Data Warehouse implementation solution for the IBM Global Services, Service Delivery Center (SDC) environment. The SDCs provide strategic outsourcing (including content hosting Web sites) to thousands of IBM and commercial customers all over the world.

Each SDC delivers a broad scope of solutions including server management (from IBM OS/390 mainframe servers to Microsoft Windows NT servers), desk-side support and customer care services. There are nine geographic SDCs world-wide, with four SDCs in North America. The Server Resource Management service is performed by the South SDC for all SDCs world-wide. SRM is the service used to report performance management and capacity planning statistics on distributed servers.

The purpose of the case study is to detail a Tivoli Warehouse solution, which will be integrated into the IBM SDC South architecture with the intention of providing a simple flowing migration from the reporting tools currently implemented.

Using the existing SRM data

The SRM data is stored on an AIX system, using DB2 UDB as its database management system. An ODBC connection was defined from the test central data repository to the SRM repository. Both repositories resided on the same TCP/IP network, so there were no firewall or additional security issues to address during the case study. The user ID and password for the SRM database was needed by the Tivoli Data Warehouse. In order to extract the data from the SRM database, select level authority to the tables as granted.

Implementing the methodology - Prototype schema

The purpose of this case study was to understand the implications of mapping a non-Tivoli data source into the central data repository. Since the Tivoli Data Warehouse was not available to the general public at the time of the case study, there were no official production schemas defined for the Warehouse. In order to resolve this issue, a prototype schema was provided. This prototype schema was capable of storing the data from the Tivoli Decision Support, Server Performance Prediction Guide, which uses Tivoli Distributed Monitoring and Tivoli Inventory as its data source. The prototype schema was enhanced from the metrics that the Server Performance Prediction Guide kept, providing for 66 metrics instead of the 17 that were part of the guide.

For purposes of the case study, existing fields in the SRM schema were mapped to the corresponding fields in the central data repository. Only fields that existed in both schemas were mapped. We found that all data fields were of the same data type, requiring no translation of data types (there was a single character data field that needed to be translated, and it is described in “Case study 1 - Server Resource Management (SRM)” on page 129.

5.1.2 Supplementing the existing schema

In addition to mapping existing data sources into the central data repository, one may want to supplement the current schema with additional information. The central data repository provides information on several different system management disciplines (for example, performance management, inventory, and software distribution). One may have the requirement of reporting additional metrics that the current schema does not provide with the existing discipline (for example, process level server performance detail in addition to server level performance information) or one may want to provide an additional discipline that is not yet defined.

The generic schema of the central data warehouse, based on the Common Information Model (CIM) data model, makes it easy to define additional different metrics in an existing discipline. The simplest case of this is to add a metric to an existing data object. In the central data warehouse, the data table is keyed by the metric ID. It is easy to add a new metric to the metric ID table, and then start using that metric in the data table.

One may want to create additional disciplines into the central data repository. For example, one may want to add financial data to the inventory data already residing in the repository. By creating additional tables in the repository with similar keys, one would be able to report on both the inventory and finance information, with easy-to-define database queries. This also enables one to take advantage of data marts, which are described in Chapter 6, “How to create data marts” on page 179.

The following section is an example of adding tables and fields to the existing schema.

Case study 2 - AIS

The point of the second case study is to go through all the steps to give the reader an experience in loading data from a source that is outside what Tivoli will provide. This data can consist of any data the user feels should be reported on through the Tivoli Enterprise Data Warehouse.

Note: This case study is actually taken from a real customer implementation. But honoring the customer’s request, we keep the name confidential.

The data for the ABC Information Services (AIS) case study consist of TEC events that are summarized by host by hour. Therefore, each row in the data source will consist of the host name, the hour start, the hour end, the severity, and an event count.

This data can be used similarly to how the TEC event TDS Guide was used and to generate reports similar to the following, plus many more:

- Top ten report showing the ten hosts with the highest event counts. This report can be generated for each time period.
- Worst weekday report where the data can be summarized by day to show the heaviest days measured by events.
- Severity breakdown report that breaks down the count of events by severity over the time frame reported on.

The data is used as a source, and how the data is collected into that source is beyond the scope of this study. How the data moves from the source through the CDW and into the data mart is the scope of this study.

5.2 Methodology

The methodology used in both of the case studies was based on the *Enabling an Application for Tivoli Enterprise Data Warehouse*, GC32-0745, (referred as the Enablement Guide hereafter). Chapter 2 of the manual describes the process of planning for the extract, transform, and load (ETL) process. The process has 14 steps. The first ten steps will be described in this chapter. The last four steps, in which the creation of the data marts and are discussed, are in Chapter 6, “How to create data marts” on page 179.

The intended audience for this manual is application programmers who want to use Tivoli Enterprise Data Warehouse to store and report on their application’s data. This covers both groups of people who want to map existing data sources to the current schema, and those who want to add to the schema.

There are two case studies that follow. The first case study takes advantage of an existing data schema in the central data repository. This example is useful for those who may have several different respositories of the same general type of information. By mapping all these repositories to the central data repository, common reporting of the data can take place. Case study one contains server performance (utilization) data from a non-Tivoli data source, and maps the data onto a prototype common repository. The current repository is called Server Resource Management (SRM).

The second case study extends the current schema by adding additional database tables of related information to existing data tables. This example is useful for those who want to move all their system management data into the central data repository, but where there is currently no schema defining that specific data. Case study two contains server event data that comes from the Tivoli Event Console (TEC). The current repository is called ABC Information Systems (AIS). AIS is further described in Chapter 6, “How to create data marts” on page 179.

The following sections discuss the first ten ETL steps, and specific details as they relates to SRM and AIS. For consistency, this manual will show actual screen images and examples of only the AIS installation. One can assume that the SRM installation is similar, with significant exceptions noted.

5.3 Following the Enablement Guide steps to ETL

In the Enablement Guide Chapter 2, “Planning for the extract, transform, and load process”, there are recommended steps for enabling your application for the Tivoli Enterprise Data Warehouse. These steps were followed to enable a new or existing data source, with the only real exception being step 5, installing the Tivoli Enterprise Data Warehouse and at least one application, which was performed before any other steps. This provided a better understanding of the product and what was to be expected from the process. It also provided the needed directory structure and allows the study team to reference the existing application for guidance on such things as naming conventions and ETL coding.

Note: The scripts used in these case studies can be found in Appendix B, “Scripts” on page 303. They can also be downloaded from the Redbooks Web site. For downloading instructions, please refer to Appendix C, “Additional material” on page 365.

5.3.1 Step 1: Define the data to be extracted

The AIS data consists of counts of TEC events that are summarized by the host by hour. Therefore, each row in the AIS data source will consist of the host name, the hour start, the hour end, the severity, and an event count. The SRM data consists of performance data that are summarized by the host by hour. The AIS and SRM data was created outside the scope of this study. What you gain from this study is how to use any source to feed data into the Tivoli Enterprise Data Warehouse. The AIS data can be used for reports such as the hours with the most events across an organization, top ten hosts with the most events, and also the data can be rolled up to show these numbers over different time frames. The SRM data can be used for reports such as the hours where the CPU, memory, or disk values are over a certain threshold, regardless of platform. This data is relatively simplistic but it allows the user to not be bothered with understanding the data. Instead the user can concentrate on the method for integrating the data into the Tivoli Enterprise Data Warehouse.

5.3.2 Step 2: Familiarize yourself with the schema

The second step is to familiarize yourself with the Tivoli Enterprise Data Warehouse generic schema and how the static, or dimensional, data is common to all applications. This is accomplished by referencing the Enablement Guide appendices for the data model diagrams of the Tivoli Enterprise Data Warehouse combined with working with tables once the product is installed. This will take some time and should be addressed thoroughly. The better one understands this schema the better the decisions one will be able to make for this integration.

5.3.3 Step 3: Complete data enablement template

This step is where the new source data gets mapped into Tivoli Enterprise Data Warehouse. This planning step is crucial to the success of the entire process. Once complete the implementation process picture starts to come together. This template will allow you to know the static data requirements the new source will have as well as provide a view of the common data to be used.

Given how the data for this case study is at a very basic grain, the data template was relatively easy. However, this still took time to ensure there was a clean mapping of the data. It also provided the one-time static data that would need to be loaded before the hourly data could be applied. The importance of this step will be seen in greater proportions as the process continues.

5.3.4 Step 4: Review naming conventions of the Enablement Guide

As stated at the beginning of this case study, the Enablement Guide needs to be completely read and understood. The naming conventions are considered at this point so that fewer modifications will need to be done later in the process. By knowing the naming convention now, the names should not need to be adjusted later. Please review the Enablement Guide Chapter 5, as it discusses the conventions is greater detail, and Section 6.3.1, “Data warehouse terminology” on page 183 of this document for a brief overview of the naming convention.

For illustrative purposes of this manual, AIS and SRM were chosen as the qualifiers for the data.

5.3.5 Step 5: Install at least one application additional application

Performing an installation of the Tivoli Enterprise Data Warehouse and one Tivoli application (Distributed Monitoring) that is shipped with ETL was done prior to beginning this process. This provided a better understanding of the product and what was to be expected from the process. It also provided the needed directory structure and allows the study team to reference the existing application for guidance on such things as naming conventions and ETL coding. If the installation was not completed earlier in this process, then it should be completed at this time. Please refer to Chapter 3, “Installation and configuration” on page 47 for installation instructions for Tivoli Enterprise Data Warehouse, and relevant application documentation for the Tivoli application that you install, besides Tivoli Enterprise Data Warehouse.

5.3.6 Step 6: Insert the one-time static data into the CDW tables

As mentioned in Section 5.2, “Methodology” on page 131 and in the Enablement Guide, the Tivoli Enterprise Data Warehouse utilizes dimensional data to provide descriptive information about the factual information. The dimensional data must be loaded initially.

Some of this data will need to be loaded before the standard ETL process can complete successfully. The data template provides the framework for this data.

Tivoli provides an example script to assist with this initial load; however, as a variation, we loaded the data through direct inserts, as in Example 5-1 and Example 5-2 on page 135, and text file imports as in Example 5-3 on page 135. By loading the data without the use of the Tivoli script, we demonstrate the flexibility of the Tivoli Enterprise Data Warehouse design.

Example 5-1 Sample static loading commands

```
db2 => insert into twg.attrtyp values ('AIS_EVT_SUM_HRLY','AIS Event Summary by
Hour')
DB20000I  The SQL command completed successfully.
db2 => insert into twg.mgrptyp values ('AISEVT','AIS Event Data')
DB20000I  The SQL command completed successfully.
db2 => insert into twg.mgrp values('EVTCNT','AISEVT',null,'AIS_Event_Hrly_Data'
)
DB20000I  The SQL command completed successfully.
db2 => insert into twg.msrc values ('TEC',null,'AIS-Tivoli Enterprise Console')
DB20000I  The SQL command completed successfully.
b2 => insert into twg.msmttyp values (68,'QTY','TEC','AIS_Events','AIS-Number o
f TEC events')
DB20000I  The SQL command completed successfully.
db2 => insert into twg.msmtrul values ('IP_HOST',49)
DB20000I  The SQL command completed successfully.
```

Note: In Example 5-1, the number 68 was inserted into the table; however, there is a database trigger that converts any number inserted into the table to the next available sequence number. This ensures the ID will be unique. The unique number generated for this row was 31. This number will be referenced later in this process. In contrast, the number 49 inserted into the twg.msmtrul table did flow into the table. Therefore, work had to be done to ensure that this was a unique ID.

Example 5-2 Sample static loading commands

```
db2 => insert into twg.mgrpmbr values ('EVTCNT','AISEVT',31)
DB20000I  The SQL command completed successfully.
db2 => insert into twg.centr values ('AIS',null,'US',22,'AIS Default Center')
DB20000I  The SQL command completed successfully.
db2 => insert into twg.cust values (61,null,'US',22,'AIS','AIS-ABC Information
Services')
DB20000I  The SQL command completed successfully.
```

Note: In example 5.2, the number 31 was inserted into the table, however, there is a database trigger that converts any number inserted into the table to the next available sequence number. This ensures the ID will be unique. The unique number generated for this row was 73. This number will be referenced later in this process.

To gather the data from the source for loading the host name and customer name into the customer tables, we first selected the distinct host name combined with the text that we want to be the customer ID, and stored this data into a comma separated file (CSV), as shown in Example 5-3. The query to build the data was executed on the source, and the data was moved to where it could be loaded into the CDW.

Example 5-3 Sample select to gather data to import into static tables

```
D:\SQLLIB\BIN>db2 "select distinct 'AIS'||','||hostname from
inv.stage_ais_sp_hr
ly_evt_sev" > f:\tmp\new_cust_ais_data.csv
```

The data was then imported into the DB in an append mode so that the data would be added to the table and allow the existing data to remain. The procedure for doing this may depend on the RDBMS platform being used; therefore, the syntax is not shown here.

The center tables were populated using the same method.

Note: Depending on options set in the SQL session, the text file may need to be cleaned to have header and/or footer lines removed.

Once the static data has been loaded, the data should be reviewed to make sure that it is in the tables and format expected. Note that character strings stored in the table are case sensitive.

5.3.7 Step 7: Determine the incremental extract columns

To reduce the time and effort for loading the data, care should be taken to only load new data. A method will need to be developed to prevent the overhead of loading duplicate data. There are two main methods of doing this: One is storing date information and only loading data beyond the date of the last loaded date. The second is using an incrementing unique ID, and only loading data where the ID is greater than the largest loaded ID. The date method leaves the possibility of getting late data that would not load properly. Therefore, we decided on the incremental ID method.

This case study source data needed to be altered to contain a column that would store a sequence ID for each row of data that was input. This was accomplished by adding a column to the source data with a trigger that would populate the column with a value selected from a sequence on inserting new data into the table.

Taking into consideration that the already existing data would need to be pulled into the CDW, we simply update all the existing rows in the data source with the ID of 100. This number is arbitrary, as long as the sequence will start greater than this number, meaning our sequence will start at 101. This satisfies our needs because we will load the CDW with all data having an ID greater that 0 on the initial load. Every row that is inserted into the source data will get a number greater than 100 and this number will increment allowing us to store the greatest ID we loaded this run and then we will know where to load from the next run.

It is important to understand that the source data was altered, but in a way that should not affect any existing applications utilizing the data. Also, it is important to understand that this application ETL will store the largest ID that has been loaded, and will only load data with an ID greater than the stored ID.

5.3.8 Step 8: Review timestamps for all source data

The CDW requires that all date and time information be stored in universal time code (UTC). Therefore, if source data spans multiple time zones, some adjustments may need to be made.

- These adjustments could be made in the source, perhaps by adding another date and time column and converting the existing date and time and storing them in the new column. They could also be made by simply altering the source data in the existing columns; of course this may force changes to all the existing applications using the data.
- These adjustments could be made by the source ETL as it moves the data into the CDW, where the changes are made can be decided by the development team, based on the environment.
- These adjustments could also take place inside the CDW. Simply by utilizing the Tivoli Enterprise Data Warehouse's ability to use center and/or customers tables in conjunction with the time zone table (see the data enablement template). This will allow the data to be stored in original form and the adjustments to be made by either the target ETLs, data mart, or Report Interface.

The source data in the AIS study is stored as a UTC timestamp, and the time zone is readily available. The time zone information was input in Example 5-2 on page 135; therefore, this step was effectively completed by utilizing the data enablement template. The source data in the SRM study is not stored as a UTC timestamp. An assumption was made that all the SRM servers were in the same time zone, and identical time zone information was put into the CDW. One of the previously mentioned resolutions would have to be implemented into SRM in order to remove this time zone assumption.

5.3.9 Step 9: Review and apply common task

A task for this step is to make sure the metadata is complete for the new source. This was accomplished by filling out the data template and inserting the static data in previous steps; however, this is an excellent time to review the data and ensure its quality and relationships.

Once the static data is complete the implementation can begin.

Begin implementation

In the study this is where we chose to transition from concept to reality, and begin implementation of the new application. This is to prepare for coding the source ETLs in Section 5.5.11. Remember, DB2, Data Warehouse Center, Tivoli Enterprise Data Warehouse, and at least one application have already been successfully installed.

To begin, open the DB2 control center, choose **Tools** from the menu and then select **Data Warehouse Center**. This will open the Data Warehouse Center where the following steps can be carried out.

Create subject areas

First the DB2 Data Warehouse Center subject areas must be created for the new application. See Figure 5-1 for an example of the screen displayed for the subject areas. The AIS_Protorype_v.1.1_Subject _Area is the newly created subject area for this study.

Note: When creating new items, expand other areas that makes it easier to follow naming conventions in place.

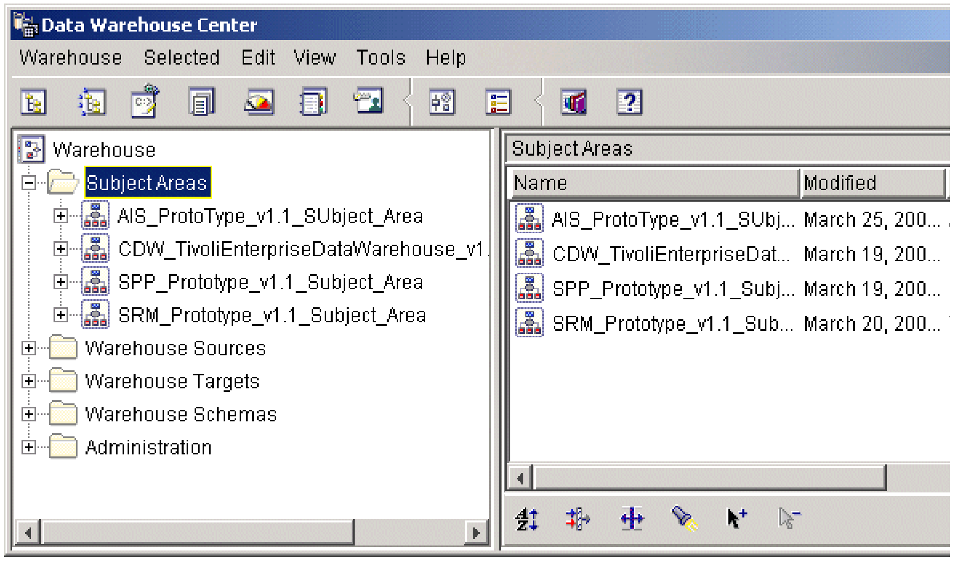
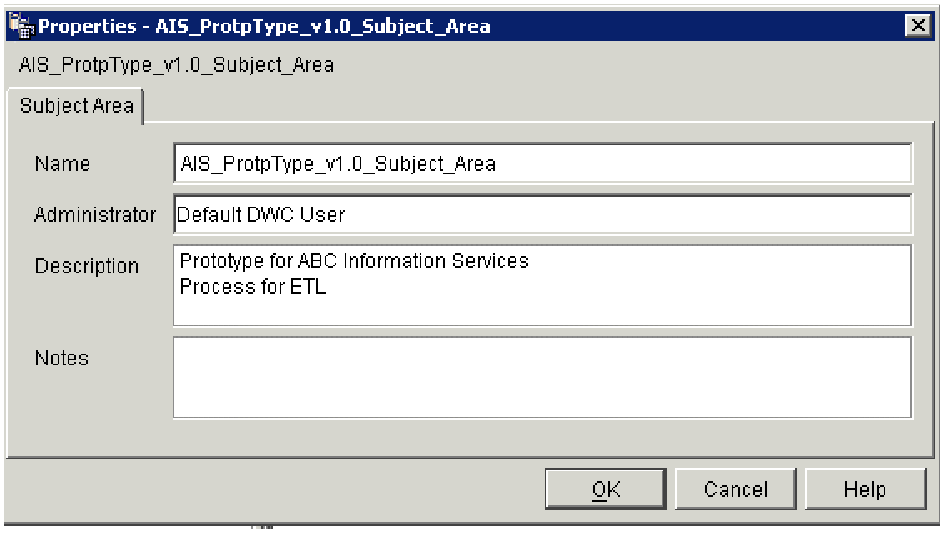

Data Warehouse Center
Warehouse
Selected
Edit
View
Tools
Help
Warehouse
Subject Areas
AIS_ProtoType_v1.1_SUbject_Area
CDW_TivoliEnterpriseDataWarehouse_v1
SPP_Prototype_v1.1_Subject_Area
SRM_Prototype_v1.1_Subject_Area
Warehouse Sources
Warehouse Targets
Warehouse Schemas
Administration
Subject Areas
Name
Modified
AIS_ProtoType_v1.1_SUbj... March 25, 200...
CDW_TivoliEnterpriseDat... March 19, 200...
SPP_Prototype_v1.1_Subj... March 19, 200...
SRM_Prototype_v1.1_Sub... March 20, 200...

Figure 5-1 Data Warehouse Center subject areas

To create a new subject area do the following:

1. Open the Data Warehouse Center and expand subject areas as seen in Figure 5-1.
2. Right click **Subject Areas**, which will produce a sub-menu.
3. Choose **Define** from the sub-menu, and a dialog box appears for the new subject area providing fields for name, administrator, description, and notes.
4. Fill in the subject area name, description, and notes as appropriate, remembering to follow any naming conventions in place (see Figure 5-2 on page 139). For this study the administrator of default DWC user was appropriate; therefore, no changes were made.


Properties - AIS_ProtpType_v1.0_Subject_Area

AIS_ProtpType_v1.0_Subject_Area

Name	AIS_ProtpType_v1.0_Subject_Area
Administrator	Default DWC User
Description	Prototype for ABC Information Services Process for ETL
Notes	

Figure 5-2 Defining a new subject area

Create a warehouse source

Once the subject area is complete, a warehouse source must be added. The process to create a warehouse source is similar to creating a subject area.

1. Open Data Warehouse Center and expand **Warehouse Sources**, then highlight and right click **Warehouse Source**, which produces a sub-menu.
2. Follow the menu path to **Define -> DB2 Family -> DB2 UDP for Windows NT**, as in Figure 5-3 on page 140.

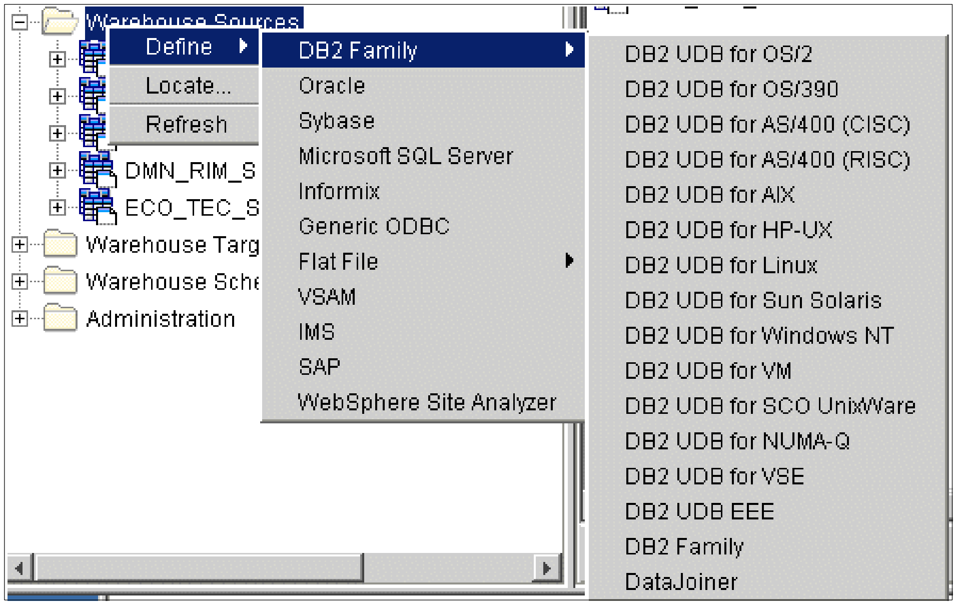
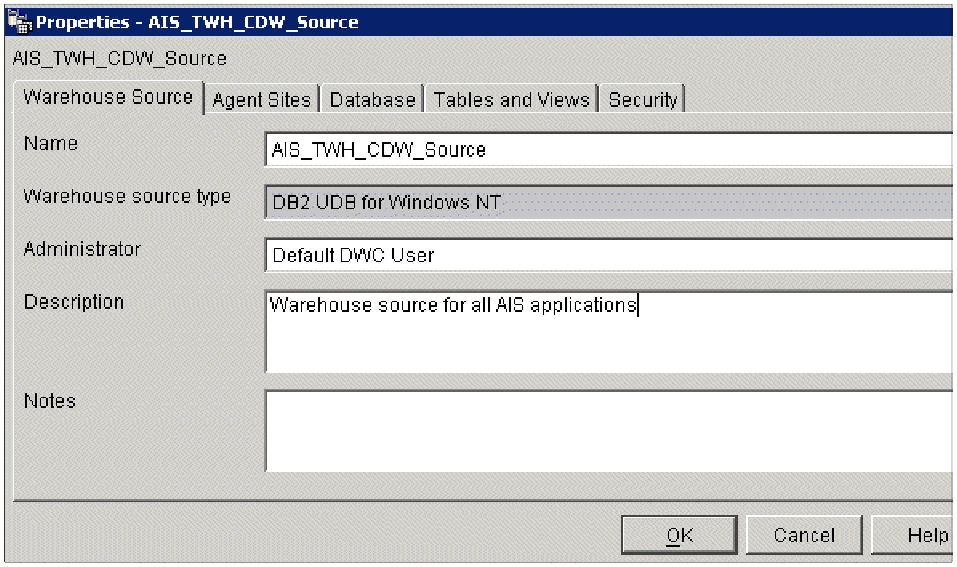


Figure 5-3 Sub-menus for defining a new source

3. This will result in a dialog with five tabs, as in Figure 5-4.



Name: AIS_TWH_CDW_Source

Warehouse source type: DB2 UDB for Windows NT

Administrator: Default DWC User

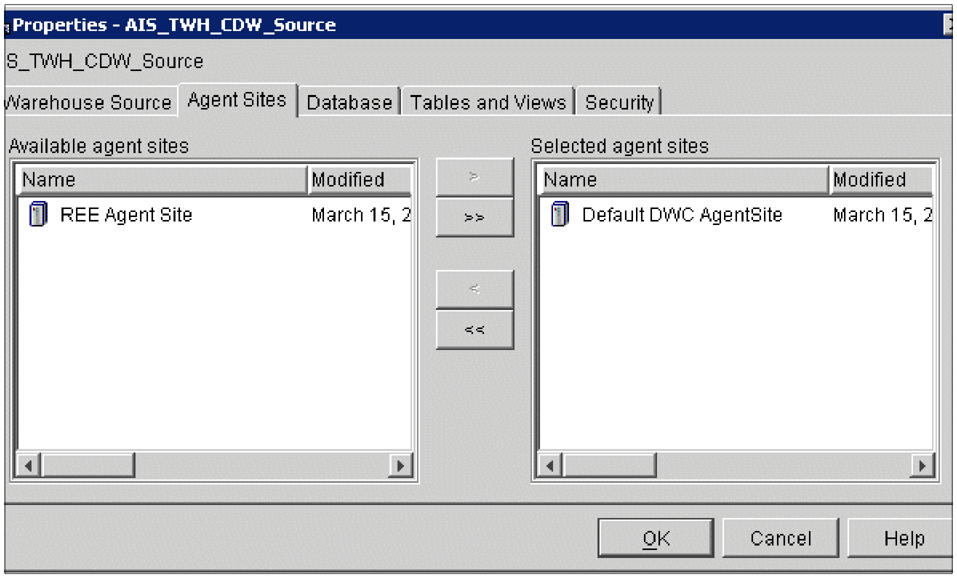
Description: Warehouse source for all AIS applications

Notes:

Figure 5-4 Warehouse source define and properties dialog

4. On the initial tab (Warehouse Source) fill in the name, description, and notes as needed. For this study the administrator of Default DWC user was appropriate; therefore, it was not changed.
5. The Agent Sites tab may need to be configured to match the environment. Note that for the agent site to be populated with the desired agent site, DB2 must be able to reach the agent. ODBC will need to be set up for this tab to show the agent site.

   For this study the data was on the server; therefore, the default DWC agent site served as the agent, as in Figure 5-5.



Available agent sites

Name	Modified
REE Agent Site	March 15, 2

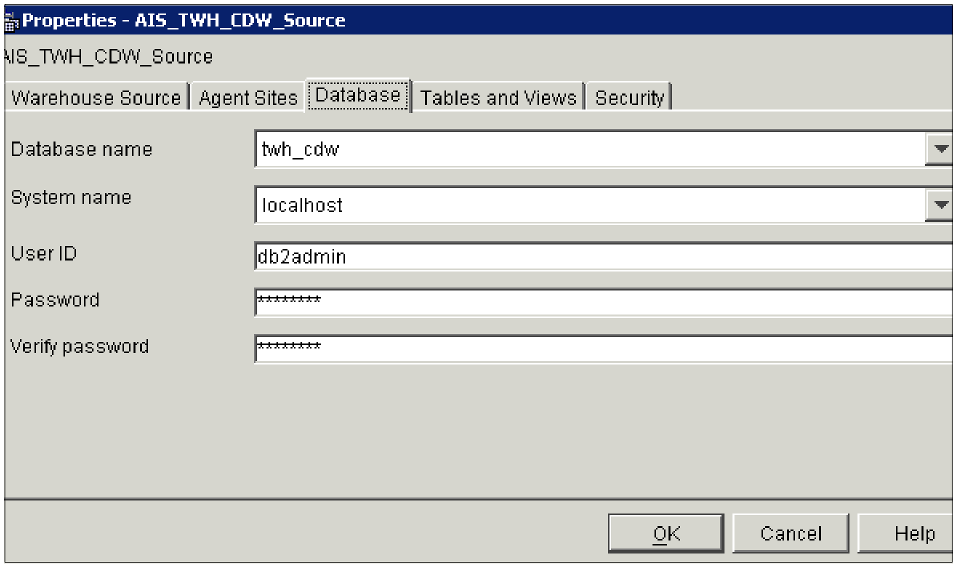
Selected agent sites

Name	Modified
Default DWC AgentSite	March 15, 2

Figure 5-5 Warehouse source definition - Agent Sites tab

6. The Database tab will need to be configured. Please see Figure 5-6 on page 142.

   The first option is to choose the database schema name from a list of previously entered database schemas or type in the schema name if it does not exist in the list. We found the drop-down list slightly misleading in that it only contains previous entries, and it is not obvious that any schema not in the list can simply be typed in.


Properties - AIS_TWH_CDW_Source

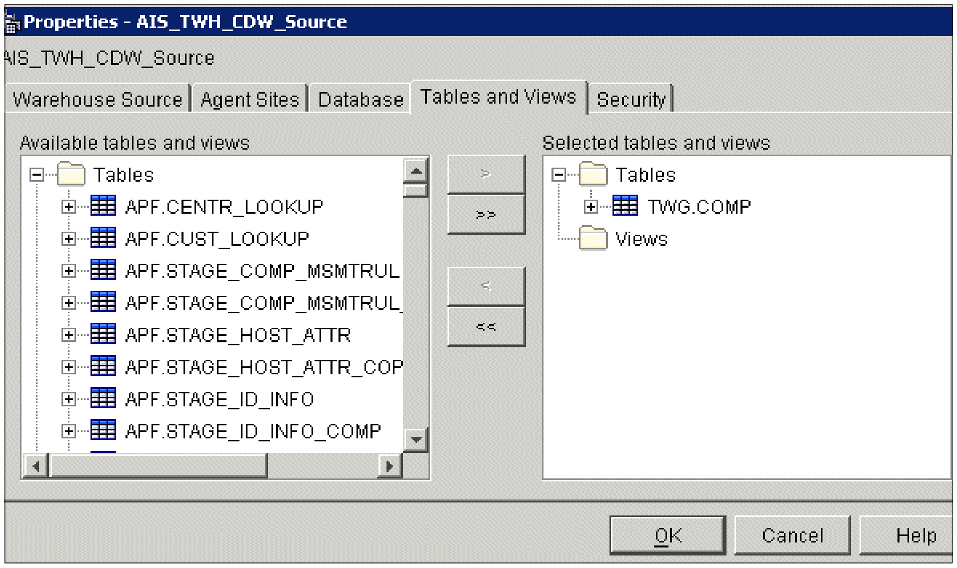
AIS_TWH_CDW_Source

Database name	twh_cdw
System name	localhost
User ID	db2admin
Password	*********
Verify password	*********

Figure 5-6 Warehouse source definition - Database tab

7. The next option is the system/host name of where the database is located. This will need to be configured to the host where the source data can be reached. Again this is through ODBC, therefore the ODBC connection should already exist.
8. The last two options are for the database user name and password to access the source data. These will need to be filled in appropriately.
9. The Tables or Views tab is used to assign the table name(s) or view name(s) to this source. Please see Figure 5-7 on page 143.

   Expand the tables or views by clicking the plus (**+**) sign beside the appropriate folder in the Available tables and views area.


Properties - AIS_TWH_CDW_Source

AIS_TWH_CDW_Source

Available tables and views

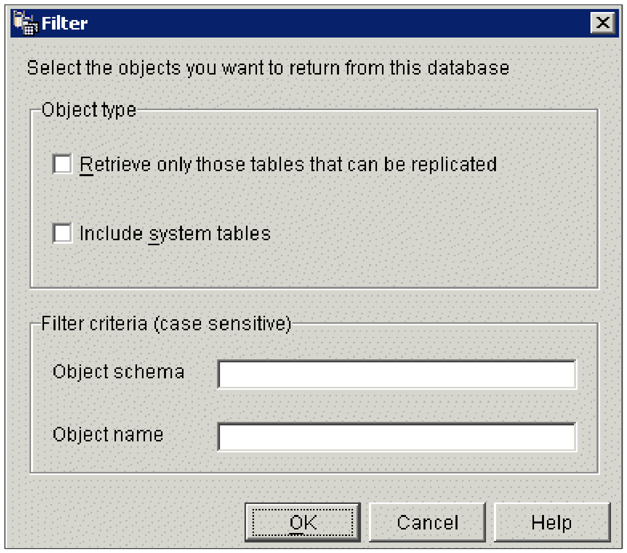
- Tables
  - APF.CENTR_LOOKUP
  - APF.CUST_LOOKUP
  - APF.STAGE_COMP_MSMTRUL
  - APF.STAGE_COMP_MSMTRUL_
  - APF.STAGE_HOST_ATTR
  - APF.STAGE_HOST_ATTR_COP
  - APF.STAGE_ID_INFO
  - APF.STAGE_ID_INFO_COMP

Selected tables and views

- Tables
  - TWG.COMP
- Views

Figure 5-7 Warehouse source definition - Tables and Views tab

10. When clicked, a new dialog appears to provide specifics on retrieving tables and views, as in Figure 5-8.


Filter

Select the objects you want to return from this database

Object type

- [ ] Retrieve only those tables that can be replicated
- [ ] Include system tables

Filter criteria (case sensitive)

Object schema	
Object name	

Figure 5-8 Warehouse source definition - Filter to retrieve tables or views

Important: If the database user name provided has system or administrator rights, the user can click **OK** and all the tables views can be reached.

However, if the database user does not have rights to the database, this will fail. Attempts were made to only provide schema names and limits to tables the supplied user name had rights on, but all attempts failed.

What will work is if both the schema name and specific table are named and can be reached. This causes the user to add one table at a time if the user does not have administrator rights.

11. After the schema and table are provided the user can click **OK**.
12. Once the tables are displayed, as in Figure 5-9, the user will need to highlight the table name(s) in the left window and then click the **Add** button, which is denoted with the greater than (>) sign. This will move the table or view to the right side of the screen into the **Selected Tables and Views** area.

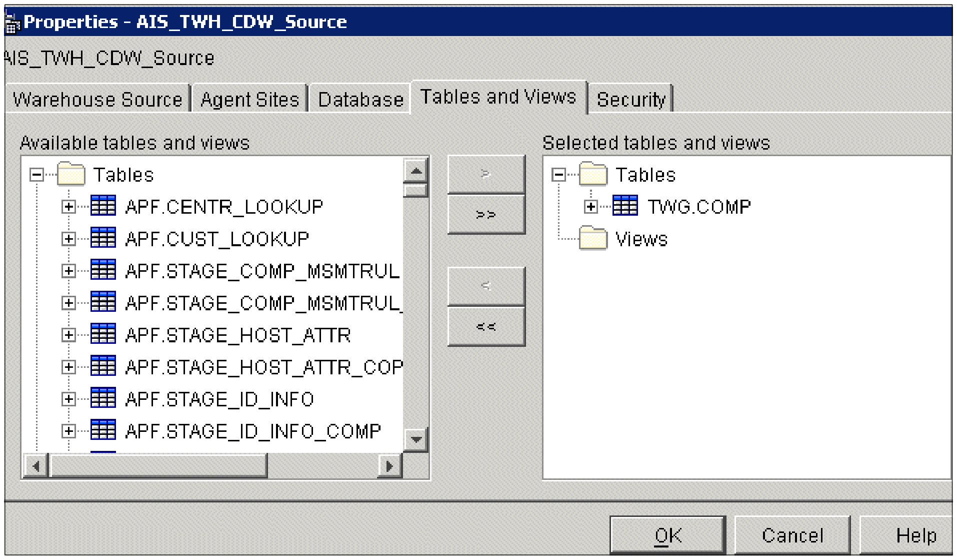
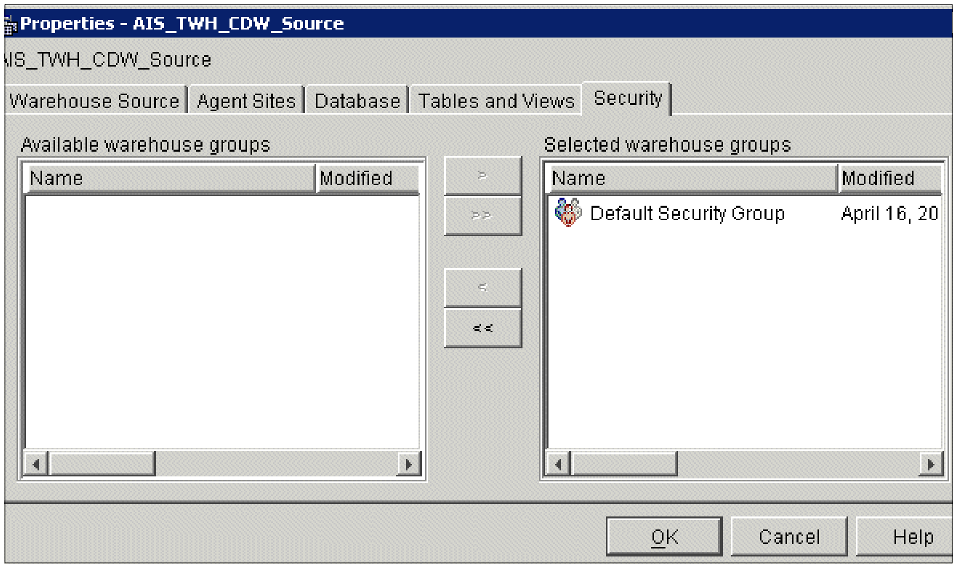


Figure 5-9 Warehouse source definition - Tables and Views tab

13. The last tab, the Security tab, did not need to be adjusted for the study, therefore, no adjustments were made. Please see Figure 5-10 on page 145.


Properties - AIS_TWH_CDW_Source

AIS_TWH_CDW_Source

Warehouse Source | Agent Sites | Database | Tables and Views | Security

Available warehouse groups

Name	Modified

Selected warehouse groups

Name	Modified
Default Security Group	April 16, 20

Figure 5-10 Warehouse source definition - Security tab

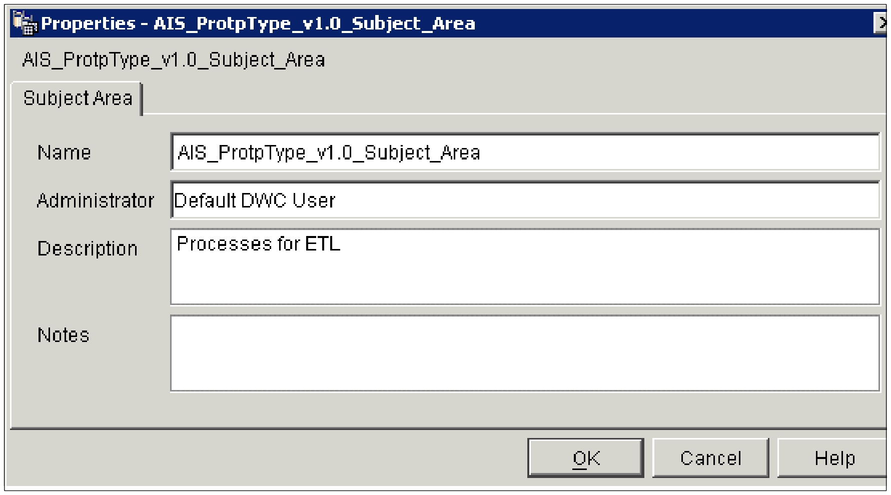
Create a warehouse target

Once the warehouse source is created a warehouse target will need to be created. The steps for creating a warehouse source can be used to create a target, with the obvious change of right clicking warehouse targets instead of sources.

Create a process

Now that the subject area, source(s), and target(s) exist for the new subject, the actual process for this subject will need to be created. This can be done by expanding one of the already existing subject areas to view the names of the processes. This can be used to assist in following the naming standards.

1. Right click **Process** in the left window of the Data Warehouse Center, then choose **Define** from the resulting pop-up menu. This will produce a dialog as in Figure 5-11 on page 146.


Properties - AIS_ProtpType_v1.0_Subject_Area

AIS_ProtpType_v1.0_Subject_Area

Subject Area

Name: AIS_ProtpType_v1.0_Subject_Area

Administrator: Default DWC User

Description: Processes for ETL

Notes:

Figure 5-11 Defining a process

2. The Administrator field was left alone for the case study and will need to be changed to match the user's environment.
3. The Description and Notes fields can be filled in as needed; then click **OK**.

Create a step inside the process

Now to add content to the process:

1. Right click the new process in the left window and follow the menu path to **Define -> User defined programs** and **Transformers -> Tivoli -> SQL Script**, similar to Figure 5-12 on page 147.

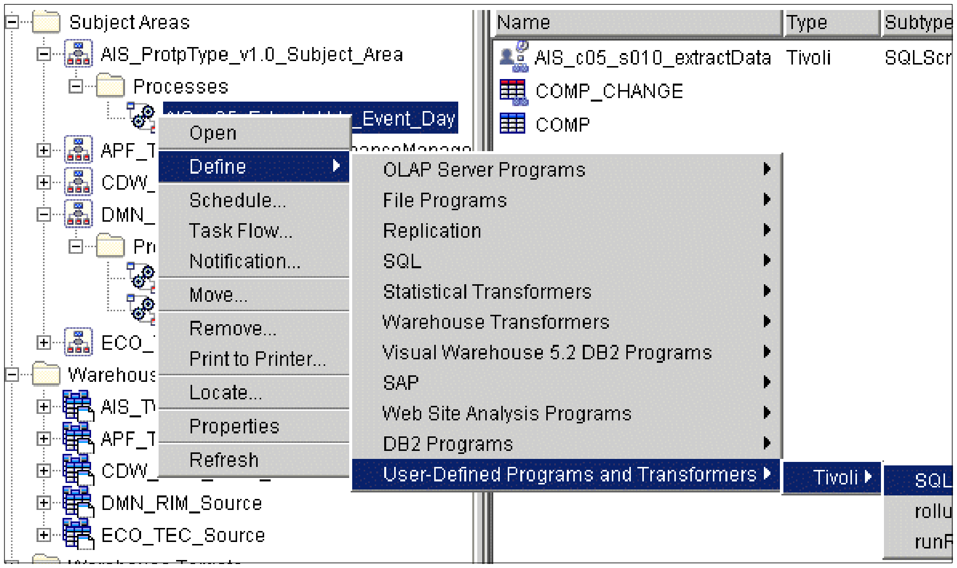
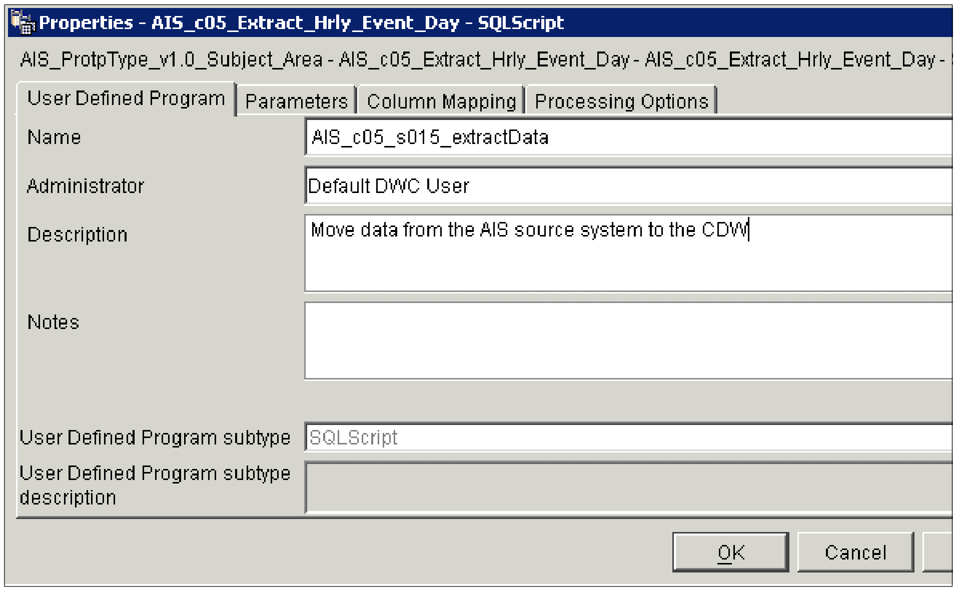


Figure 5-12   Sub-menu to define a sqlscript step

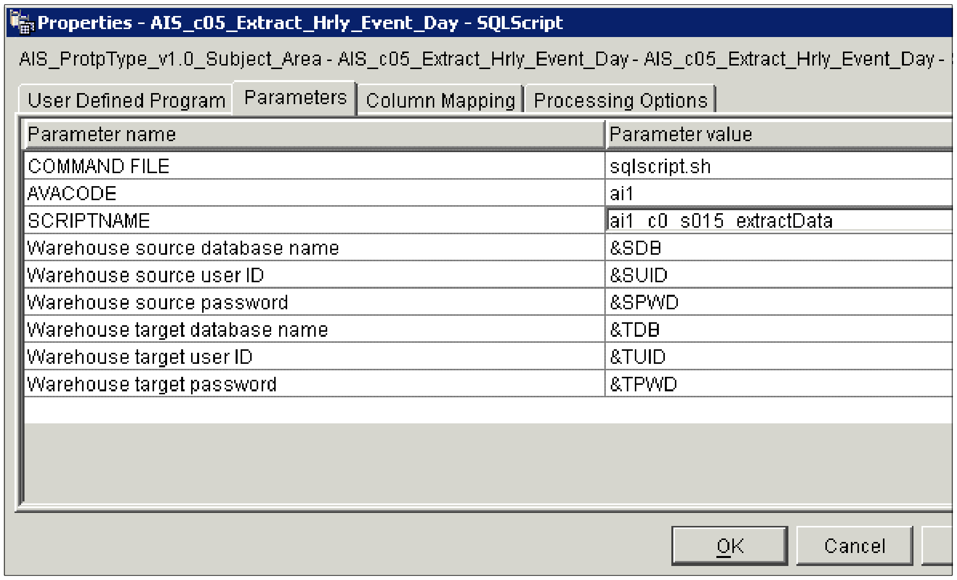
2. This produces a dialog to configure the process, which has four tabs as in Figure 5-13.


Properties - AIS_c05_Extract_Hrly_Event_Day - SQLScript
AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day -
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel

Name: AIS_c05_s015_extractData
Administrator: Default DWC User
Description: Move data from the AIS source system to the CDW
Notes:
User Defined Program subtype: SQLScript
User Defined Program subtype description:

Figure 5-13   Definition of process step - User Defined Program tab

3. The first tab, as in Figure 5-13 on page 147, contains the name, description, and notes and should be filled in appropriately. The naming of this process step should be chosen carefully, again referencing the Enablement Guide to review how the process steps should be named.
4. The second tab, as in Figure 5-14, is the Parameters tab, which contains the details about the process step. The first field is the command file. This will be pre-filled with the sqlscript.sh, which was filled when the user selected SQL script from the pop-up menu. This script is covered in-depth in the Enablement Guide and is the application programmers interface (API) for the ETLs.


Properties - AIS_c05_Extract_Hrly_Event_Day - SQLScript
AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day -
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel

Parameter name	Parameter value
COMMAND FILE	sqlscript.sh
AVACODE	ai1
SCRIPTNAME	ai1 c0 s015 extractData
Warehouse source database name	&SDB
Warehouse source user ID	&SUID
Warehouse source password	&SPWD
Warehouse target database name	&TDB
Warehouse target user ID	&TUID
Warehouse target password	&TPWD

Figure 5-14 Definition of process step - Parameters tab

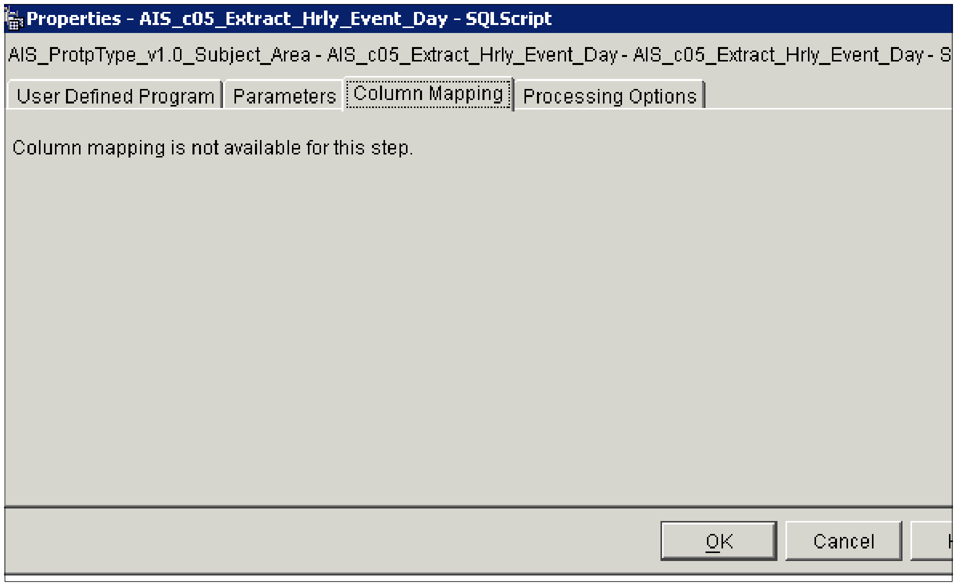
5. The AVACode will be the product code or AVA code for this product, which for IBM products will be the three-digit IBM AVA code. For non-IBM product integrations an AVA/product code will need to be generated. It should contain two alphabetic digits and one numeric digit that will form a unique identifier for this product.
6. The script name will be the actual script used by the sqlscript.sh. This script does not have to exist at this time in the process. This script is the ETL that the user will need to create or copy and edit in Section 5.3.10, “Step 10: Code the source ETL” on page 151. This script must be created in the appropriate directory structure as described in the Enablement Guide and will have an extension of .db2. However, in the dialog the extension is left off as in Figure 5-14. For example, if the script name was

ai1_c10_s015_extractData.db2 on the operating system, in the Windows dialog the script name will be labeled ai1_c10_s015_extractData.

7. The remaining fields are the variables used for the sqlscript.sh to determine the target and source, which do not need to be adjusted.

Tip: The variables passed to sqlscript.sh from the Parameters tab determine which database to use as target and source, not individual tables. Notice that sqlscript.sh does not refer to the Warehouse Sources or Warehouse Targets to decide which tables to update; instead this is controlled by ETL script written by the user. Therefore, the warehouse source and target only need to be in the same database as the tables that will be affected. This is important when debugging, because results may not reflect exactly what the warehouse center is displaying. Although for consistency, the correct tables should be named as sources and targets.

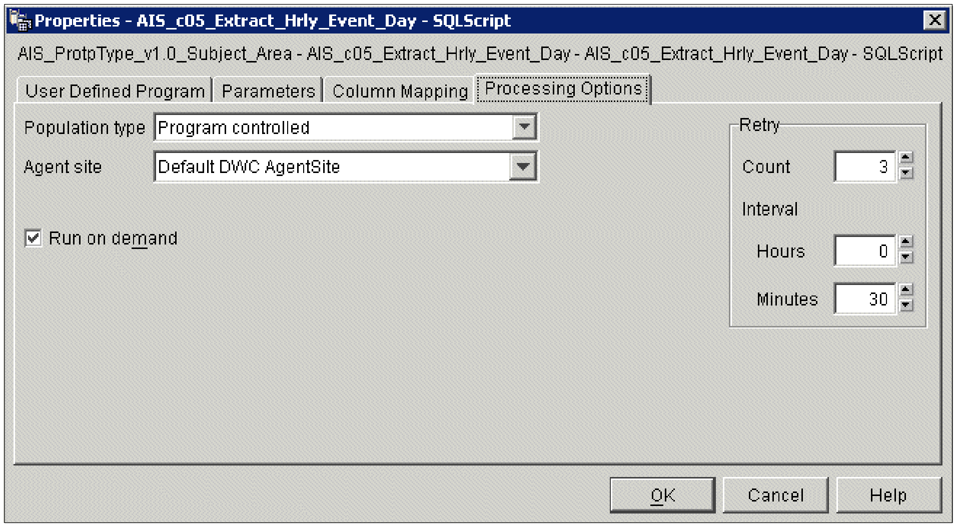
8. The next tab is Column Mapping, which is not available for this type of process, as seen in Figure 5-15.


Properties - AIS_c05_Extract_Hrly_Event_Day - SQLScript
AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day - S
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel

Column mapping is not available for this step.

Figure 5-15 Definition of process step - Column Mapping tab

9. The last tab is Processing Options and can be used for debugging options. Click **OK** to complete the process step creation.


Properties - AIS_c05_Extract_Hrly_Event_Day - SQLScript

AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day - SQLScript

User Defined Program | Parameters | Column Mapping | Processing Options

Population type: Program controlled

Agent site: Default DWC AgentSite

[✓] Run on demand

Retry

Count: 3

Interval

Hours: 0

Minutes: 30

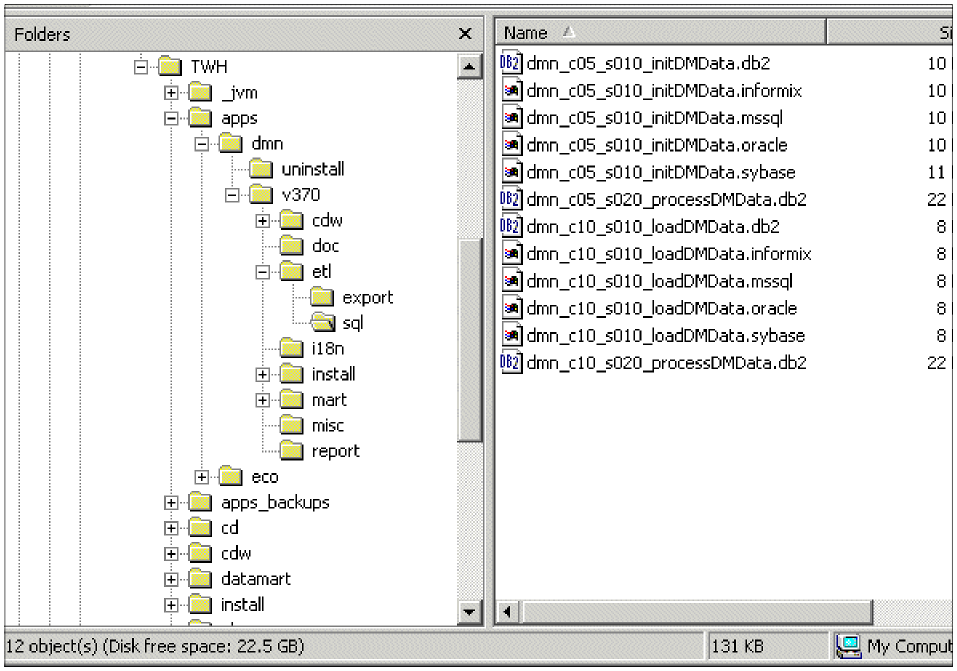
Figure 5-16 Definition of process step - Processing Options tab

Creating the directory structure

The directory structure will need to be in place for the new application. When the Tivoli application (Distributed Monitoring) was installed at the beginning of this process, a directory was created to contain the application's ETLs. This was chosen by the user at time of install, and could be similar to C:\TEDW\TWH\apps. This structure's top directory will be referred to as %TOPDIR%. This can be different depending on the configuration of the machine and/or where it is installed.

In this study the %TOPDIR% is simply c:\twh\apps. Underneath this directory there will be a directory for each product installed, using the product code. For the new product a directory will need to be created named as the product code.

Below the product code directory a version directory will need to exist and this directory must start with the letter v. After the version directory, the structure will need to be created exactly as the existing application. This can be done by using Windows Explorer to copy and paste the existing structure from the installed application to the new version directory. See Figure 5-17 on page 151 for a sample of the directory structure for the DM application.



```
Name                                    Si
dmn_c05_s010_initDMData.db2             10
dmn_c05_s010_initDMData.informix        10
dmn_c05_s010_initDMData.mssql           10
dmn_c05_s010_initDMData.oracle          10
dmn_c05_s010_initDMData.sybase          11
dmn_c05_s020_processDMData.db2          22
dmn_c10_s010_loadDMData.db2              8
dmn_c10_s010_loadDMData.informix         8
dmn_c10_s010_loadDMData.mssql            8
dmn_c10_s010_loadDMData.oracle           8
dmn_c10_s010_loadDMData.sybase           8
dmn_c10_s020_processDMData.db2          22

12 object(s) (Disk free space: 22.5 GB)     131 KB     My Comput
```

Figure 5-17 Example of directory structure for DM application

Once this is created on the operating system, it is time to code the ETLs.

5.3.10 Step 10: Code the source ETL

The ETLs are by far the most involved steps in this process and should be viewed as a development process. Therefore, proper development methodologies should be utilized, such as working in a lab environment before attaching to production data, using proper testing procedures, and possibly utilizing a version control system.

Generally the best way to start is to copy an existing ETL that is the closest match to the new ETL and adapting it to meet the user's needs.

The sqlscript.sh ETLs use a binary provided by Tivoli Enterprise Data Warehouse to parse and process the ETL script. This binary is the SQL execution engine. The Enablement Guide contains a detailed description of the use of the SQL execution engine, which is included below.

Using the SQL execution engine

An extract, transform, and load process that reads data from a source application and places the data into the central data warehouse database is called a *central data warehouse ETL*. A central data warehouse ETL is developed within, and scheduled and run by, the IBM DB2 Data Warehouse Center.

Due to the distributed nature of Tivoli software applications and the Tivoli Enterprise Data Warehouse, some features required for central data warehouse ETL processes are not supported by the Data Warehouse Center. Specifically, a central data warehouse ETL must update the source database (application database) with information on which records are already loaded into the warehouse. SQL steps in the Data Warehouse Center are not designed to offer the facility to update the source database during the ETL process.

Tivoli Enterprise Data Warehouse provides a tool called the SQL execution engine, which augments the DB2 Data Warehouse Center functionality. It consists primarily of a C program (platform-dependent) that allows the caller to pass in a series of SQL statements and have them run on any ODBC data source. A wrapper script is provided with the C program to make it easily callable from a Data Warehouse Center ETL process step.

The SQL execution engine provides the following benefits to the ETL developer:

- Allows updates to source databases
- Significantly improves the run-time performance of SQL statements
- Reduces the time required to install an application’s ETL scripts on the Data Warehouse Center machine
- Makes ETL processes defined in the Data Warehouse Center simpler

You can also use the SQL execution engine when working on data mart ETL processes (moving data from the central data warehouse to a data mart). This section discusses central data warehouse ETL exclusively because a majority of issues encountered during data mart ETL development are similar to the central data warehouse ETL process. ETL developers should run as much central data warehouse ETL (and data mart ETL) SQL as possible using the SQL execution engine instead of using the DB2 Data Warehouse Center SQL steps. The rest of this section describes how to set up the SQL execution engine in DB2 Data Warehouse Center and how to use it in a central data warehouse ETL process.

SQL execution engine overview

The SQL execution engine and its wrapper script are installed by the Tivoli Enterprise Data Warehouse installation software on the machine that hosts the Tivoli Enterprise Data Warehouse control server. If the Tivoli Enterprise Data Warehouse central data warehouse or Tivoli Enterprise Data Warehouse data marts are installed on other systems, then the SQL execution engine and its wrapper script are installed on those systems also.

The execsql.exe and sqlscript.sh files are placed in the TWH_TOPDIR\tools\bin directory along with bash and other executable programs required by Tivoli Enterprise Data Warehouse, where TWH_TOPDIR is the environment variable that represents the base installation directory for Tivoli Enterprise Data Warehouse. The version of the bash shell provided with Tivoli Enterprise Data Warehouse is assumed to be the only version of bash to be used with ETL scripts. The installation program extends the PATH environment variable to include the TWH_TOPDIR\tools\bin directory.

During a Tivoli Enterprise Data Warehouse installation, the DB2 Data Warehouse Center is automatically modified to include SQLScript as a user-defined program or transformer. This is accomplished by importing a user-defined program description into the Data Warehouse Center. To view the user-defined program:

1. From the Data Warehouse Center, expand the **Administration** folder.
2. Expand the **Programs and Transformers** folder.
3. Expand the **User-Defined Program and Transformers** folder.
4. Expand the **Tivoli** folder. The SQLScript program is listed along with rollup and runReport. For information on rollup and runReport, see “Aggregation and rollup” on page 44 and “Report scheduling” on page 57 of the Enablement Guide.

The SQL execution engine handles any type of SQL statement. The SQL can be run in one of three places:

- On the source data source (application database)
- On the target data source (central data warehouse database)
- On both the source and target data sources using the cross-data source INSERT INTO SQL construct

To run SQL statements, the SQL execution engine makes ODBC calls on ODBC drivers. ODBC data sources that correspond to source and target databases must be defined on the machine where the SQL execution engine runs.

SQL execution engine setup

The execsql.exe and sqlscript.sh files that compose the SQL execution engine must be placed in a path that can be located by the DB2 Data Warehouse Center. In a typical Tivoli Enterprise Data Warehouse installation, the sqlscript.sh, execsql.exe, and bash programs are located in the TWH_TOPDIR\tools\bin directory. After the installation of Tivoli Enterprise Data Warehouse, this directory is placed in the PATH environment variable.

The ETL developer must write one or more custom scripts that contain one or more SQL statements to be run by the SQL execution engine. The developer can use a text editor of choice to develop the custom script, or scripts. The SQL to be run must fit one of the three types described in “SQL execution engine overview” on page 23 of the Enablement Guide.

> **Note:** The SQL execution engine does not work with network mounted drives. SQL scripts run by the SQL execution engine must be on a local drive.

After the custom scripts are developed, they must be put in a particular directory path to work with the DB2 Data Warehouse Center and the SQL execution engine. The following directory path simulates the path to an application’s ETL installation. The application ETL scripts are installed by the Tivoli Enterprise Data Warehouse installation program using application-supplied media that contains the relevant scripts. The scripts are installed in the following location:

```
TWH_TOPDIR\apps\product_code\vversion\etl\sql\yourCustomScript
```

Where:

TWH_TOPDIR is the base installation directory for Tivoli Enterprise Data Warehouse.

product_code is the code that uniquely identifies an application. IBM and Tivoli software applications must use the three-character code assigned by IBM to a product. Third-party applications must also use a three-character code that includes at least one number. This code is used in log messages and other places where an application must be uniquely identified.

version is an integer value representing the version of the ETL scripts (for example, 110).

yourCustomScript is the script name. The script name must use the following naming convention, which is also defined in Chapter 5, “Naming conventions” on page 65 of the Enablement Guide.

```
part1_part2_part3_part4.part5
```

Where:

part1 is the product code for the application.

part2 is the process ID, where cnn indicates a process where the target is the central data warehouse and mnn indicates a process where the target is a data mart. The two-digit number represented by nn is a unique process number assigned by the application for its ETL processes. The first process should be numbered starting with 05. Subsequent processes should add 5 to the initial number (05, 10, 15, and so on). Lower to higher numbers in part 2 of the script name indicates the order in which the script runs.

part 3 is the step name where snnn represents a particular step in the process. The first step in a process should start with 010 and subsequent steps should increment the step count by 10.

part4 is the step description. It is recommended that the table name being populated be supplied as the description. Component tables are represented by a name like X _Comp. Measurement tables are X_Msmt. Dimension and fact tables are Dim_tablename and Fact_tablename, respectively. With the execsql.exe program you can populate many tables at the same time, so an extension name based on a target table might not always make sense.

part5 is a suffix that describes the database vendor type to be used. The possible suffix values are:

- .db2: DB2
- .oracle: Oracle
- .sybase: Sybase
- .informix: Informix
- .mssql: Microsoft SQL Server
- .tsm: Tivoli Storage Manager ODBC driver

When the SQL execution engine runs, part5 of the name (including the period) is excluded from the custom script name passed as a parameter to the program. The execsql.exe program automatically generates a suffix based on the ODBC data source definition and appends this value to the custom script name at run time.

If no matches to the suffix list are found, a log message is generated by the execsql.exe program and the execsql.exe program attempts to use the custom file without any extension.

Note: When setting up warehouse sources in the Data Warehouse Center, specify Generic ODBC in the Warehouse source type list. This enables you to have a single step that can read from any ODBC data source; for example, DB2, Oracle, SQL Server, Sybase, or Informix.

The execsql.exe program checks the vendor type of the ODBC data source and looks for a script named as follows:

- *scriptname*.db2 for DB2
- *scriptname*.oracle for Oracle
- *scriptname*.mssql for Microsoft SQL Server
- *scriptname*.sybase for Sybase
- *scriptname*.informix for Informix

By default, the SQL statements are run against the source data source. The following control statements can precede any SQL statement in the custom script that you want run:

- --#INSERT_INTO_TARGET

  Requires the words `INSERT INTO` *table* (followed by a SELECT statement) in the SQL statement, where table is the name of a table that exists on the target data source. Otherwise, this command is ignored and the SQL statement is run entirely on the source data source.

  If the table name is not qualified with the schema name (for example, eco.mytable), the value for user will determine the schema name used in Oracle, DB2, and Informix. For Sybase and MSSQL, the SQL execution engine uses the default table accessible by the user.

- --#INSERT_INTO_SOURCE

  Evaluates the following `INSERT INTO` *table* SQL statement, pulling data from the TARGET data source and inserting it into the specified table in the SOURCE data source.

- --#EXECUTE_AT_TARGET

  Runs the following SQL statement at the target data source instead of at the source data source (default).

- --#IGNORE_ERROR

  Ignores any database error messages returned by the following SQL statement. The error is still logged but it does not prevent subsequent SQL statements in your custom script from running. See Example 5-4 on page 157.

Example 5-4 IGNORE_ERROR example

```
--#IGNORE_ERROR
drop table cto.stage_example
;
create table cto.stage_example like cto.template_stage_example
;
```

Note that system errors such as running out of memory cannot be ignored.

- --#EDIT_USING perl convert.pl

  Used in conjunction with the --#INSERT_INTO_* directives. Allows data from the SQL SELECT to be written into a temporary file in CSV format. The Perl script provided after the directive is called. A base temporary file name is provided in the following format:

  ```
  VWS_LOGGING/execsqlTMP.data_source.nn
  ```

  Where:

  VWS_LOGGING is the DB2 logging directory.

  data_source is the source ODBC data source name.

  nn is an integer used to separate multiple --#EDIT_USING directives in a script (starts at 0 and increments per directive).

  The script is then required to append (preedit) to the temporary file name to determine the actual file name to read from. The script is also required to append (postedit). to the base temporary file name it outputs the altered data to. The *.postedit file must also be in CSV format. To provide for handling of double quotes, an escape character (\) is inserted immediately prior to the embedded double quote on both *.preedit and *.postedit files. Failure to account for this in the script could result in damaged data. Note that double quotation marks are used to contain SQL_CHAR and SQL_VARCHAR column data. Only SQL_CHAR and SQL_VARCHAR column types are supported by --#EDIT_USING.

- --#OVERRIDE_TERMINATOR

  Changes the default SQL statement terminator from a semicolon (;) to whatever the first non-white space character is following the directive. The change is only applied to the current statement. The statement following the altered terminator is again ended with a semicolon. Example 5-5 sets the terminator to a $ instead of a semicolon on the current statement only.

Example 5-5 OVERRIDE_TERMINATOR example

```
--#OVERRIDE_TERMINATOR $
CREATE TRIGGER DMN.CompTyp_Cd_Trig NO CASCADE
BEFORE INSERT ON DMN.stage_comp
REFERENCING NEW AS N
```

```
FOR EACH ROW
MODE DB2SQL
BEGIN ATOMIC
IF LOCATE(.,N.Comp_Nm)>0 THEN
IF LOCATE(.,N.Comp_Nm,LOCATE(.,N.Comp_Nm)+1)>0 THEN
SET N.CompTyp_Cd = IP_HOST;
ELSE
SET N.CompTyp_Cd = DMN_HOST;
END IF;
ELSE
SET N.CompTyp_Cd = DMN_HOST;
END IF;
END
```

Notes:

- Comments are signified by lines that start with two dashes and a character other than the # symbol.
- You can have two comments for a single SQL statement. For example:

  ```
  --#IGNORE_ERROR
  --#EXECUTE_AT_TARGET
  drop table DM.stage_dm_metrics;
  ```

- SQL statements in the script must be ended by a semicolon (;). For example:

  ```
  INSERT INTO TSM.ADMINS (SELECT ADMIN_NAME FROM ADSM.ADMINS);
  ```

- If there is an imbalanced quote condition (failure to close the quote on a string literal), the SQL execution engine logs an error and stops.
- If there are non-white space characters (excluding comments) following the final semicolon (for example, an SQL command at the end that is not ended by a semicolon), a warning message is logged and the SQL statement is ignored.
- The following directives fail if they are used with parenthetical lists. Do not use parenthetical lists in the INSERT INTO statement when using these directives.

  ```
  --#INSERT_INTO_SOURCE
  --#INSERT_INTO_TARGET
  --#EDIT_USING
  ```

Manually testing the custom script

Before attempting to run your custom script from the Data Warehouse Center, run the script manually to validate the script.

To run the script manually, start bash from a Microsoft Windows command prompt. The bash program is installed on your system when the Tivoli Enterprise Data Warehouse software is installed. Be sure you are using this copy of bash for your testing.

Enter the following information, where script_name is the name of your custom script; for example, SPP_c05_s010_extractInvData. (For details on how to name your scripts, see Chapter 5, “Naming conventions” on page 65 of the Enablement Guide.)

```
sqlscript.sh product_code script_name source_db source_uid source_pwd target_db
target_uid target_pw
```

The SQL in your custom script should run and you should see the results in your source or target database.

An example ETL script

Example 5-6 shows a partial ETL script taken from the AIS application. This script was developed for the case study.

Example 5-6 Example of ETL code taken from AIS application

```
-------------------------------------------------------------
-- Date/Author 3/25/02 CC
-- Description  ETL for moving data from the CDW stage tables
--              to the CDW tables.
--
-- Code Sources Copied from spp_c05_s010_extractInvData.db2
--              Adapted to suit needs
-- Parameters   (See Data Warehouse Center)
-- Inputs       twh_cdw.inv.stage_ais_sp_hrly_evt_sev
-- Outputs      twh_cdw.inv.invalid_data ( bad data )
--              twh_cdw.twg.comp(good components)
-- Notes        This is not complete, but used for case study
-- Project
-- Mod log
-------------------------------------------------------------


----------------------------------
-- Move data that does not have a
-- cust or center ids into invalid
-- data table.  Filled with text
-- since we do not have all the values.
----------------------------------
--#EXECUTE_AT_TARGET
--sample check for invalid data
INSERT INTO inv.invalid_data
SELECT n.hostname,'AIS_NULL', 'AIS_NULL','AIS_NULL','AIS_NULL',
       'AIS_NULL',0,0,'AIS_NULL',0,'AIS_NULL'
```

```
  FROM inv.stage_ais_sp_hrly_evt_sev n
 WHERE NOT EXISTS(
        SELECT 1
           FROM inv.cust_lookup
          WHERE inv.cust_lookup.value = n.hostname
             OR inv.cust_lookup.value = '@')
    OR
    NOT EXISTS(
        SELECT 1
          FROM inv.centr_lookup
       WHERE inv.centr_lookup.value = n.hostname
          OR inv.centr_lookup.value = '@')
;
------------------------------------------------------------------------
-- Need to get the distinct hostname to put into comp for some reason
-- a record in the stage table produces a record in the comp table,
-- which produces 100+ comp of the same hostname.
-- Therefore moving them into another stage of just hostnames
------------------------------------------------------------------------
--#EXECUTE_AT_TARGET
DROP TABLE inv.stage_ais_sp_host;

--#EXECUTE_AT_TARGET
CREATE TABLE inv.stage_ais_sp_host(hostname VARCHAR(32));

--#EXECUTE_AT_TARGET
INSERT INTO inv.stage_ais_sp_host
SELECT DISTINCT hostname
  FROM inv.stage_ais_sp_hrly_evt_sev;

------------------------------------------------------------------------
-- Insert data into the twg.comp(component table).  This is the real work
-- of the entire script
--
-- This will not duplicate rows because it does a "not exist" in comp table
--
--Note that the insert of IP_HOST may be changed to SPP_HOST by a trigger
--  if the hostname is not fully qualified.
-------------------------------------------------------------------------
--#EXECUTE_AT_TARGET
INSERT INTO TWG.COMP(COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
SELECT 0 AS Comp_Id,
   'IP_HOST' AS CompTyp_Cd ,
   centr.Centr_Cd AS Centr_Cd,
   c.Cust_ID AS Cust_ID,
   n.hostname AS Comp_Nm,
   current timestamp - current timezone AS Comp_Strt_DtTm ,
   '9999-01-01-00.00.00.000000' AS Comp_End_DtTm,
   '' AS Comp_Ds
FROM inv.stage_ais_sp_host n,
   inv.cust_lookup cust,
   inv.centr_lookup centr,
   twg.cust c
WHERE n.hostname != ''
  AND   0 = (
      SELECT COUNT(1)
        FROM twg.comp c
       WHERE c.comp_nm=n.hostname
         AND c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
         AND current timestamp - current timezone >= (c.comp_strt_dttm - 1
day)
         AND current timestamp - current timezone <  c.comp_end_dttm )
  AND ((cust.value = n.hostname AND cust.cust_acct_cd = c.cust_acct_cd)
   OR cust.value = '@')
  AND (centr.value = n.hostname OR centr.value = '@')
;
```

Potential hurdles in coding ETLs

Following are potential hurdles in coding ETLs:

- Any staging table that you drop in the script must be created manually before the script will complete. The drop will fail and stop the script from executing unless the IGNORE_ERROR directive is used.
- Prepare for incremental extracts. The Enablement Guide discusses this in detail in Chapter 3, “Installation and configuration” on page 47. Not doing incremental extracts can cause several problems, such as no data being loaded, because it senses no change in source data and/or duplicate data being loaded into the target.
- When errors occur the Tivoli Enterprise Data Warehouse is configured to collect data that will not load into the target. It does this using tables named similar to INVALID_DATA. The user needs to make sure all INVALID_DATA table columns match the source data that may be inserted. This can cause the script to stop processing on data that the script has already determined is invalid.

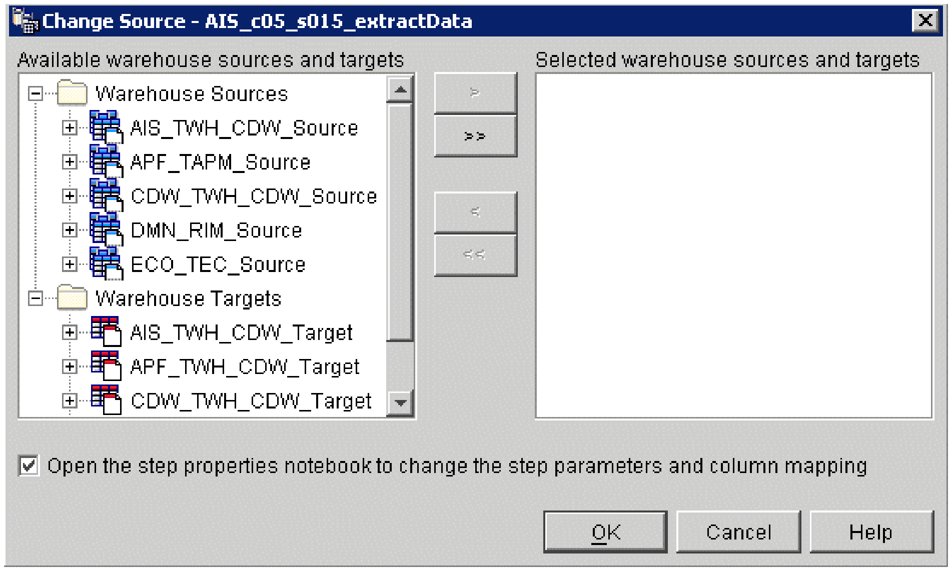
Testing the ETL

Once the ETL script is ready to be tested through the Data Warehouse Center then the GUI needs to be set up to test.

To accomplish this, do the following:

1. Open the Data Warehouse Center, and bring up the process in the left window and the step in the right window.

2. Right click the step, then choose **Change Source** from the sub-menu. Then expand the warehouse sources or targets using the plus sign (**+**), as in Figure 5-18.


Change Source - AIS_c05_s015_extractData

Available warehouse sources and targets

- Warehouse Sources
  - AIS_TWH_CDW_Source
  - APF_TAPM_Source
  - CDW_TWH_CDW_Source
  - DMN_RIM_Source
  - ECO_TEC_Source
- Warehouse Targets
  - AIS_TWH_CDW_Target
  - APF_TWH_CDW_Target
  - CDW_TWH_CDW_Target

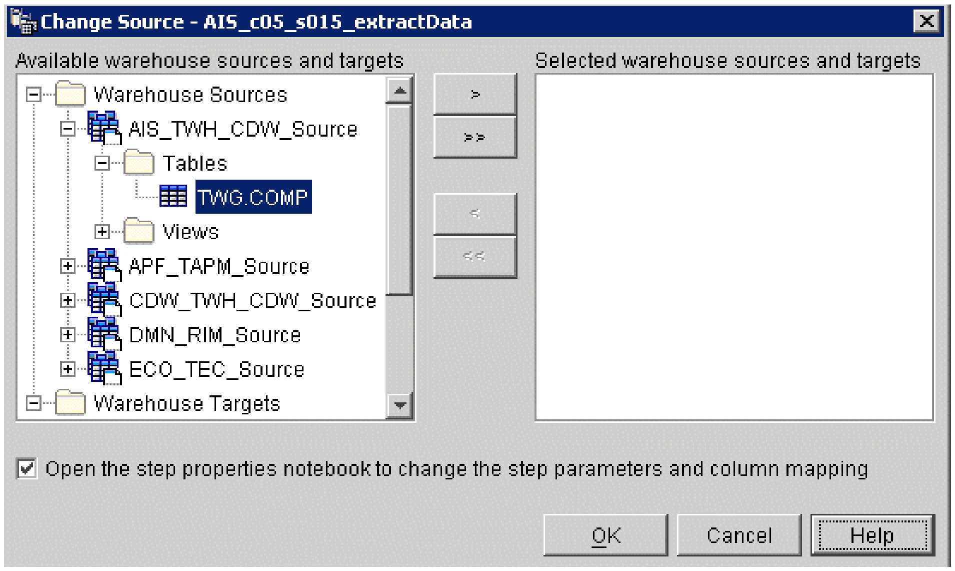
Selected warehouse sources and targets

[✓] Open the step properties notebook to change the step parameters and column mapping

Figure 5-18 Adding the source

Notice that both targets and sources appear as possible sources. This is because there may be a need to get data from a target and use the target as a source.

No matter whether the source is chosen from the list of warehouse targets or warehouse sources, it is considered a source in this context. Choose the appropriate source(s) via the product code and expand the tables or views. These are the sources located in the database where the data will be retrieved.



☑ Open the step properties notebook to change the step parameters and column mapping

Figure 5-19 Adding the source with expanded tables

3. Highlight the table or view to be used as the source, as in Figure 5-19, and use the **>** button located in the middle of the dialog to move the source from the left pane (available sources) of the dialog to the right pane (selected sources). See Figure 5-20 on page 164.

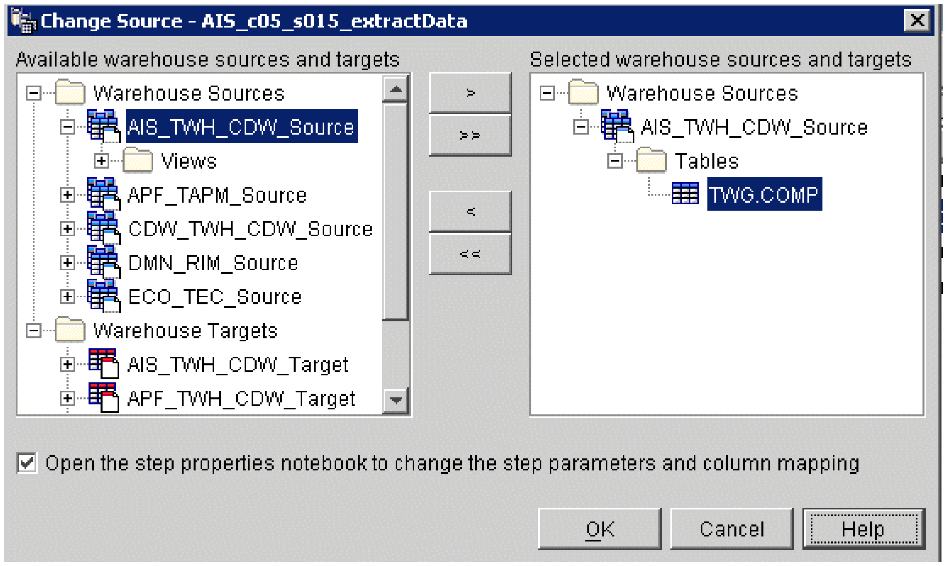


Figure 5-20 Adding the source to the selected warehouse sources and targets

4. Once all the appropriate sources are added click **OK**. This will produce the dialog for the step again. It can be dismissed by clicking **OK**.
5. Once the source has been added the target must be added, which is very similar to the adding the sources. Right click the step and from the sub-menu choose **Change Target**.
6. Then expand the warehouse targets or sources using the plus sign (**+**) as in Figure 5-21 on page 165.

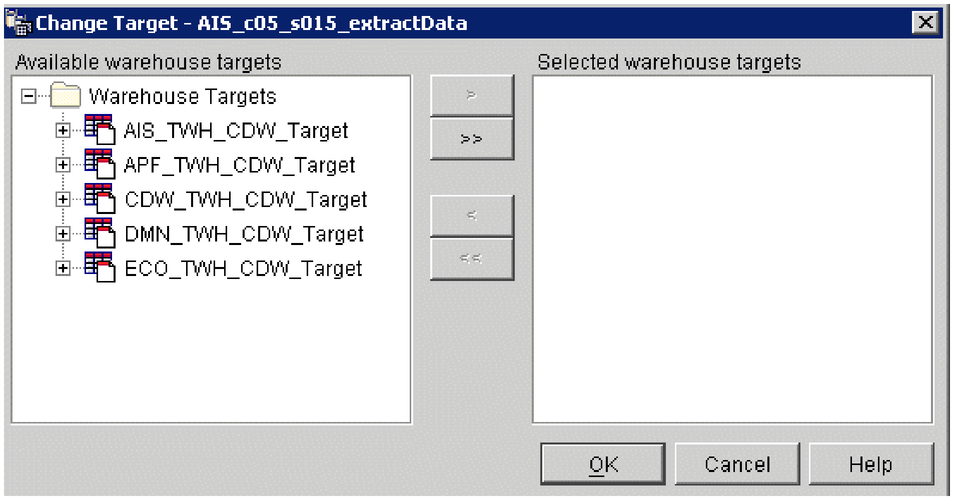


Figure 5-21 Adding a target to a step

7. Choose the appropriate target(s) via the product code and expand the tables or views, as in Figure 5-22. These are the targets located in the database where the data will be stored.

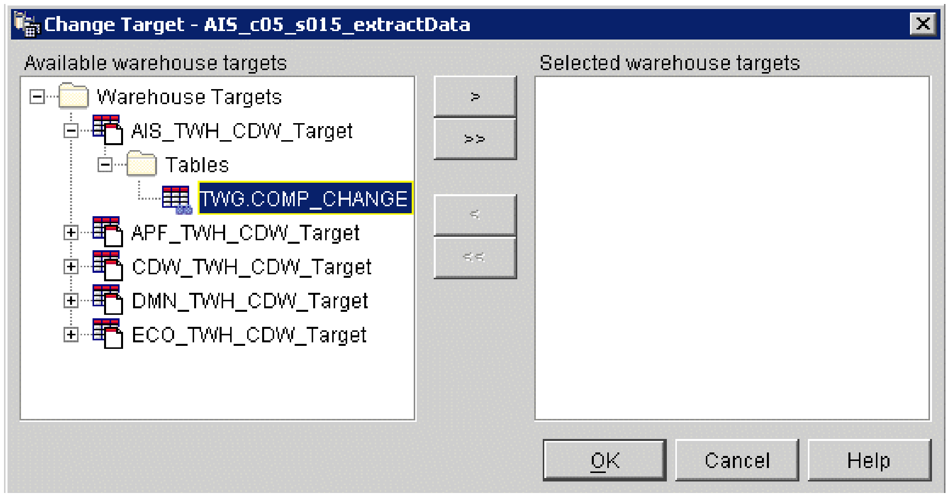


Figure 5-22 Adding a target to a step with expanded tables

Highlight the table or view to be used as the target and use the > button located in the middle of the dialog to move the target from the left pane (available targets) of the dialog to the right pane (selected targets). Please see Figure 5-30 on page 172.

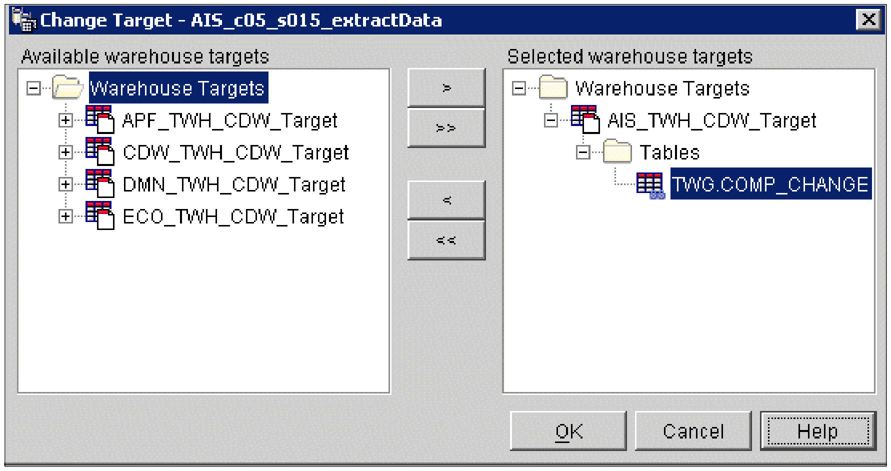


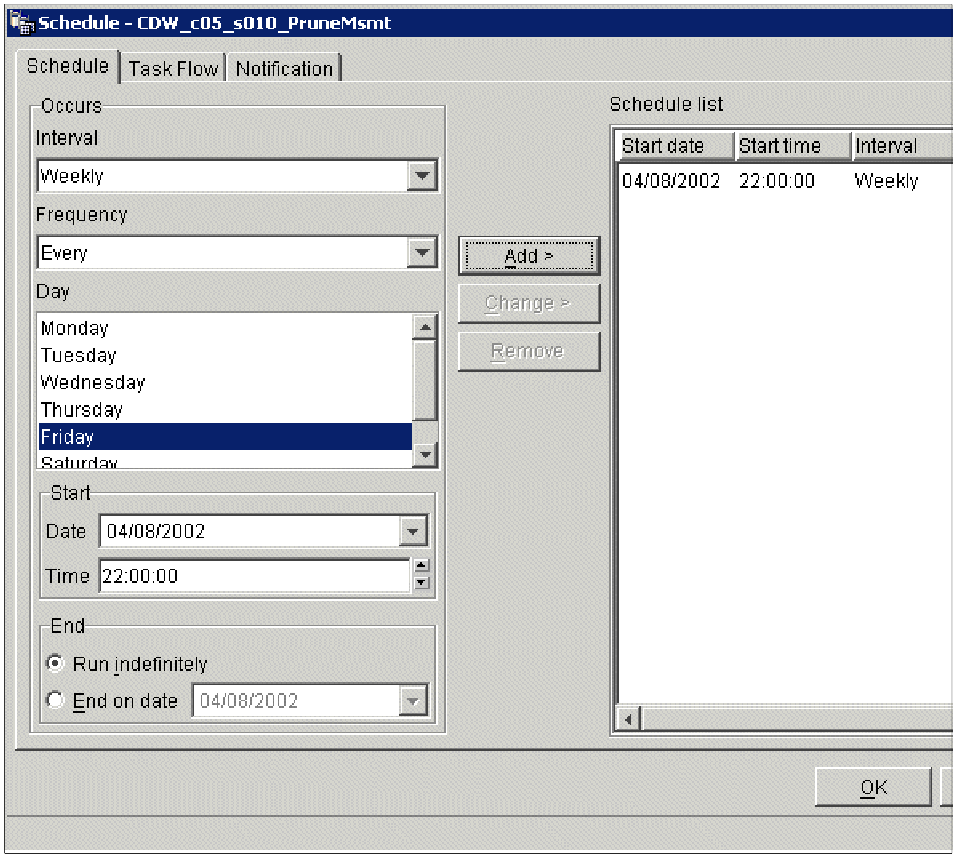
Figure 5-23 Adding target to step with target in selected warehouse targets

8. Once all the appropriate targets are added click **OK**. This will produce the dialog for the step again. It can be dismissed by clicking **OK**.

> **Tip:** At this point the Data Warehouse Center screen may not reflect changes. To refresh the screen right click **Process** in the left tree view of the processes and choose **Refresh**.

   Now the process step must be scheduled to let the Data Warehouse Center know when, and how often, the step should run.

9. Right click the step then choose **Schedule** from the sub-menu. This displays a dialog, as in Figure 5-24 on page 167, for scheduling the step.


Schedule - CDW_c05_s010_PruneMsmt

Schedule | Task Flow | Notification

Occurs

Interval: Weekly

Frequency: Every

Day: Monday, Tuesday, Wednesday, Thursday, Friday (selected), Saturday

Start

Date: 04/08/2002

Time: 22:00:00

End

(●) Run indefinitely

( ) End on date 04/08/2002

Schedule list

Start date	Start time	Interval
04/08/2002	22:00:00	Weekly

Figure 5-24 Example of schedule dialog

10. Choose the interval for running this step, the frequency, day, start date, and time, as well as end date. The options are shown through drop-down boxes or radio buttons and are self explanatory.

11. Once the options are set click the **Add>** button to add this step to the schedule. Click **OK** to complete the setup of the schedule.

    The step must now be put into production mode. Once the step is in production mode changes are not allowed until the step is put back into development or test mode. This is to prevent an inadvertent change to production jobs.

12. To put the step into production mode, right click the step and follow the sub-menu path **Mode -> Production**, as in Figure 5-25 on page 168.

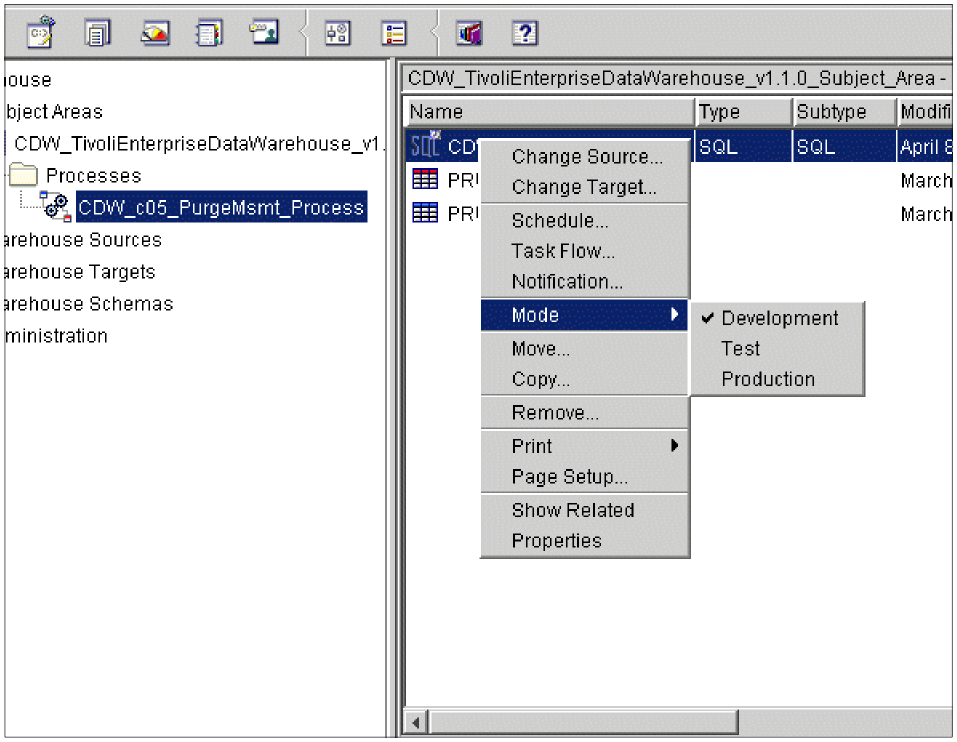
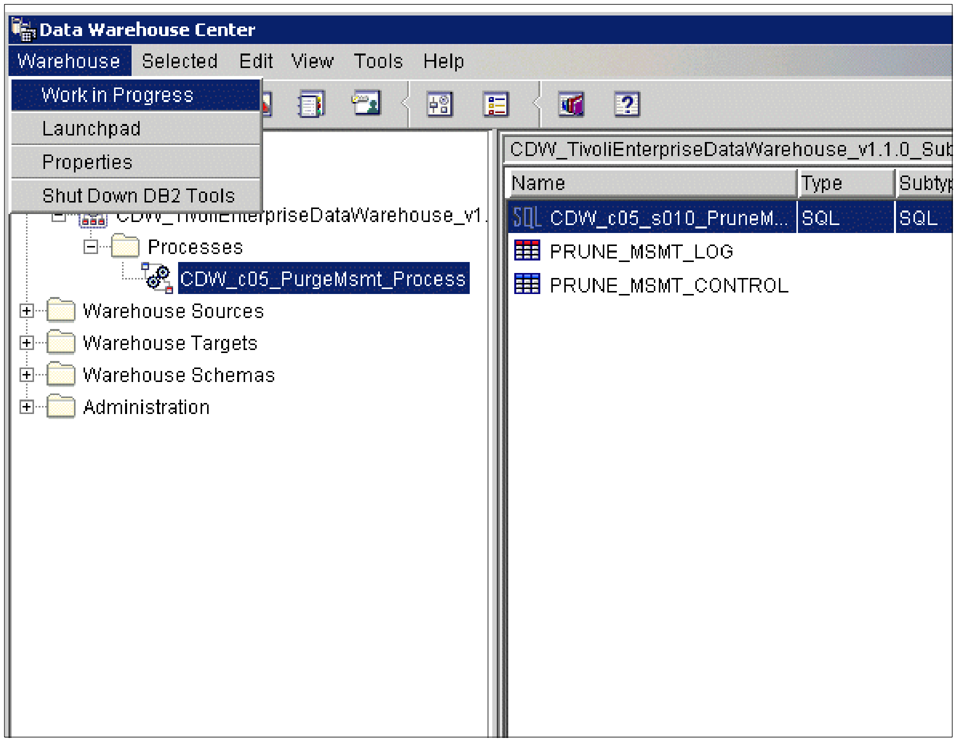


Figure 5-25 Changing modes on steps

Once the mode has been changed, the user can go to the work-in-progress screen to see the results of the executed steps and run the scheduled steps.

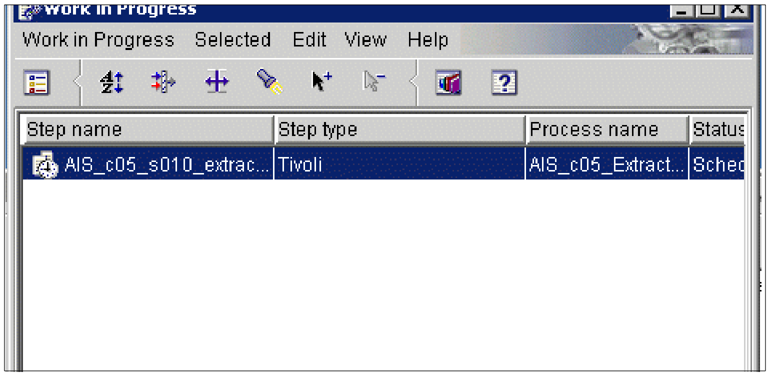
13. At the Data Warehouse Center main window, choose the **Warehouse** option from the menu. Then choose **Work in Progress** from the sub-menu, as in Figure 5-26 on page 169.



Name	Type	Subtyp
CDW_c05_s010_PruneM...	SQL	SQL
PRUNE_MSMT_LOG		
PRUNE_MSMT_CONTROL		

Figure 5-26 Data Warehouse Center main menu option of Work in Progress

Figure 5-27 on page 170 shows an example of the Work in Progress main screen. In the example only one step has been scheduled. The icon beside the step name indicates the state of the scheduled step. The clock, as shown, means that the step is scheduled, while a red X indicates the last time the step executed it failed. Finally a green check mark would show the step ran successfully.

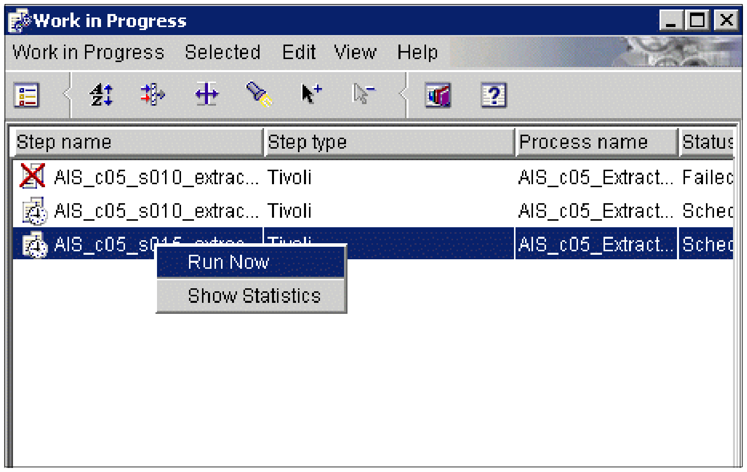


Step name	Step type	Process name	Status
AIS_c05_s010_extrac...	Tivoli	AIS_c05_Extract...	Sched

Figure 5-27 Example of Work In Progress window

Tip: A very common problem is looking in the Work in Progress screen to find a step and the step not being there. This is generally one of two errors. The step can be in test or development mode or, if the step has been scheduled for a time and that time has passed, the Work in Progress screen may not show it.

14. To test the step, run the scheduled step by right clicking the step to produce the sub-menu and choose **Run Now**, as in Figure 5-28.



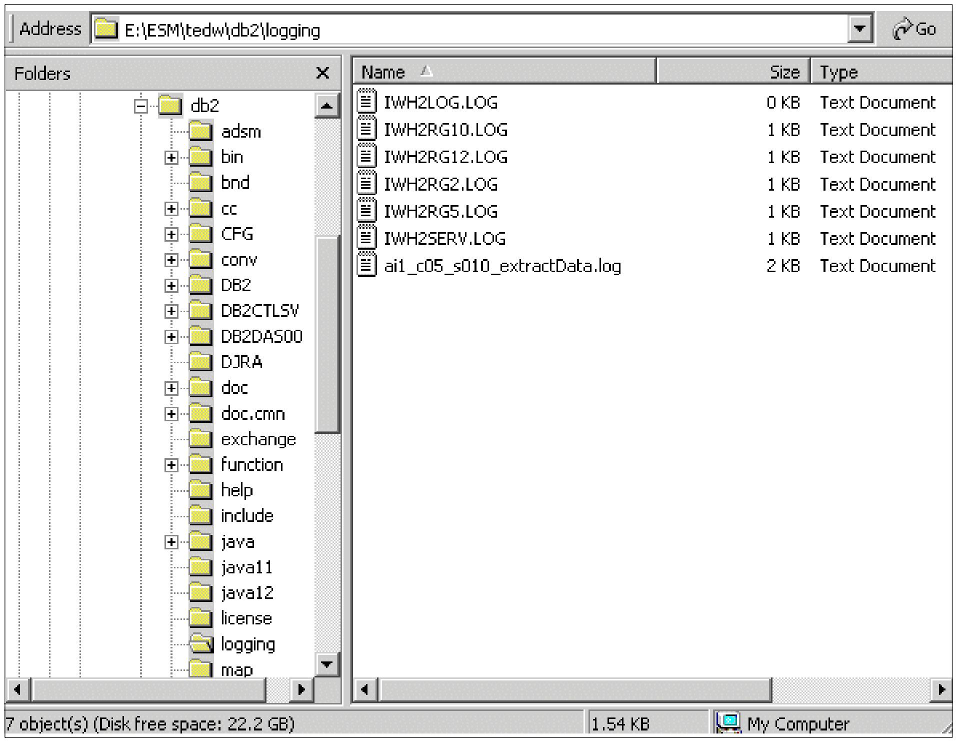
Step name	Step type	Process name	Status
AIS_c05_s010_extrac...	Tivoli	AIS_c05_Extract...	Failec
AIS_c05_s010_extrac...	Tivoli	AIS_c05_Extract...	Schec
AIS_c05_s015_extrac...	Tivoli	AIS_c05_Extract...	Schec

Figure 5-28 Work in Progress sub-menu

If the step is successful the step will then be displayed with the described green check mark. However, if the step is unsuccessful then an error dialog will be displayed. This dialog will provide error codes and some text; however, the text is very limited. A much more descriptive log can be found on the DB2 install drive under the logging directory in a file with the step name and extension of .log. For the case study the logging directory is d:\SQLLIB\logging.

Figure 5-29 displays another example of the directory containing the log file. Note that this was done on another box and does not match the case study directory structure. Again, the directory structure is dependant on the DB2 install.

This log is very useful in determining where the step script failed. It will display the successfully completed steps and a detail of the error message where the step failed. The DB2 documentation can then be used to determine more information about the error.



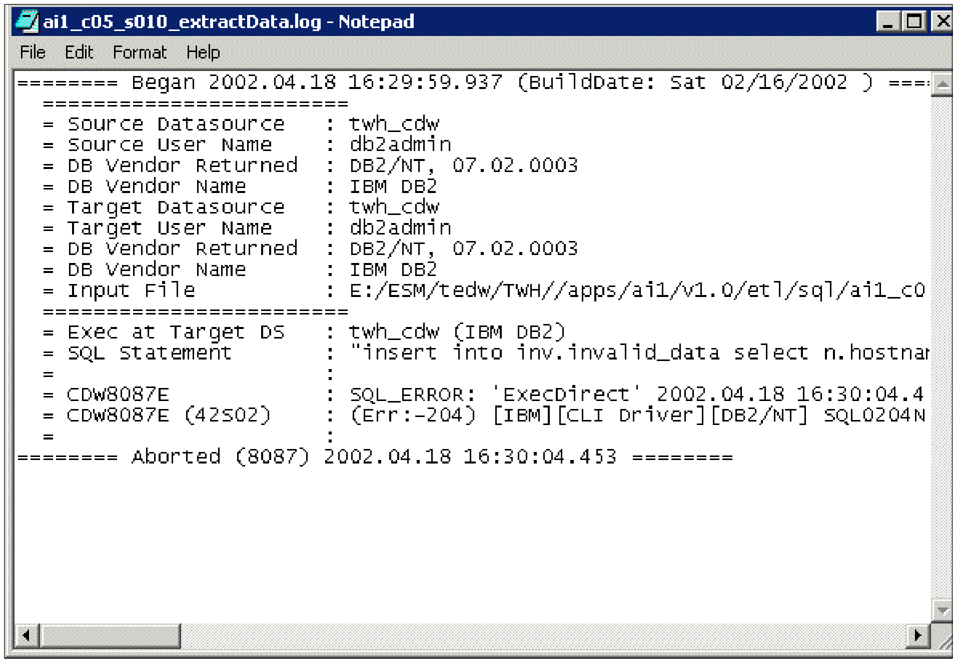
Address: E:\ESM\tedw\db2\logging

Name	Size	Type
IWH2LOG.LOG	0 KB	Text Document
IWH2RG10.LOG	1 KB	Text Document
IWH2RG12.LOG	1 KB	Text Document
IWH2RG2.LOG	1 KB	Text Document
IWH2RG5.LOG	1 KB	Text Document
IWH2SERV.LOG	1 KB	Text Document
ai1_c05_s010_extractData.log	2 KB	Text Document

7 object(s) (Disk free space: 22.2 GB) 1.54 KB

Figure 5-29 Directory with ETL error log

Also in Figure 5-30 on page 172, you can see an example of the log output.


ai1_c05_s010_extractData.log - Notepad

```
======== Began 2002.04.18 16:29:59.937 (BuildDate: Sat 02/16/2002 ) ===
  ========================
  = Source Datasource    : twh_cdw
  = Source User Name     : db2admin
  = DB Vendor Returned   : DB2/NT, 07.02.0003
  = DB Vendor Name       : IBM DB2
  = Target Datasource    : twh_cdw
  = Target User Name     : db2admin
  = DB Vendor Returned   : DB2/NT, 07.02.0003
  = DB Vendor Name       : IBM DB2
  = Input File           : E:/ESM/tedw/TWH//apps/ai1/v1.0/etl/sql/ai1_c0
  ========================
  = Exec at Target DS    : twh_cdw (IBM DB2)
  = SQL Statement        : "insert into inv.invalid_data select n.hostna
  =                      :
  = CDW8087E             : SQL_ERROR: 'ExecDirect' 2002.04.18 16:30:04.4
  = CDW8087E (42S02)     : (Err:-204) [IBM][CLI Driver][DB2/NT] SQL0204N
  =                      :
======== Aborted (8087) 2002.04.18 16:30:04.453 ========
```

Figure 5-30 ETL error log

Once the step completes successfully, this only means that the syntax of the script is correct; the logic of the step must still be tested.

The data should be reviewed to ensure that the expected results are achieved. This can be done through the DB2 interface by right clicking the table and choosing sample contents or by running SQL statements from the database SQL prompt. The template that was completed in step three should be used to verify the results.

Typically, as with any development project, this is an iterative process and will need to be completed several times before getting the desired results.

5.4 Lessons learned from the case studies

In this section we will document our experiences with the case studies.

5.4.1 Case study 1 results

We were able to take existing SRM data and, using the ETL process, were able to map the SRM data into the existing central data repository schema. The use of the enabling manual was critical to that success. The following issues were discovered and resolved as follows:

- Case of the host names of the servers: The servers in SRM were in both uppercase for the CISC servers, and lowercase for the RISC (Unix) servers. The prototype schema we used required lowercase server names. In order to resolve this issue, the names of the servers were translated to lowercase using the DB2 *make lower* capability on the initial query of data from the SRM repository.
- Length of the host names of the servers: The length of the field that kept the server names in SRM was 64 characters. The length of the field that kept the server names in the prototype schema was 32 characters. In order to resolve this issue, the names of the servers were truncated at 32 characters. Fortunately, none of the existing servers had more than 32 characters, and the uniqueness of the server names was maintained.
- DB2 levels: The DB2 levels between the SRM database and the central data repository were different. This caused any query of the SRM database to fail when issued from the Tivoli Data Warehouse. In order to resolve this issue, the DB2 utilities were bound to the SRM database from the Tivoli Data Warehouse database.
- Incremental extract: The prototype schema required either a sequence number in the data or a timestamp in the data when the data was added to SRM. This would be used to extract only the data since the previous extract. The SRM data did not have either method of being able to determine what data should be incrementally extracted. In order to resolve this issue, we used the timestamp that the data was specified for. The ETL was set up to incrementally extract the previous day’s data. In most cases this will be sufficient. Additional exception coding would be needed in order to add data to the central data repository that was not present in SRM when the data was extracted (missing servers that needed to be rerun).
- Log space: The DB2 log space filled up during the initial load of historical data from SRM into the repository. In order to resolve this issue, we incrementally loaded the historical data a day at a time. This is the same amount of data that would be expected during a normal incremental extract.
- Color coding: In order to exactly duplicate some of the SRM reports, a color coding field was needed. These fields did not exist in the prototype schema. In order to resolve this issue, the color coding fields were not loaded into the repository. It would be possible to add the metrics that held the color coding

information. Since the purpose of this case study was to map to existing fields, this was left to be added some time in the future.

- 15 minute time period data: The prototype schema had provisions for hour, day, week, and month data. There was no place to store the 15 minute data. For purposes of the case study, the storing of the 15 minute data was considered out of scope, to be added some time in the future.
- National Language Support: The repository has the provisions for multiple languages. The names of the metrics were defined in English. For purposes of the case study, the creation of the translation strings were considered out of scope, to be added some time in the future.
- Week-to-date and month-to-date values: The rollup function of the Tivoli Data Warehouse provides the capability to automatically calculate the weekly and monthly data, based on having complete weeks and months of information. The SRM schema provided data for partial months and weeks. For purposes of the case study, the reporting of partial weeks and months are to be added some time in the future.
- UTC timestamp: The data in the prototype schema is stored in UTC timestamp format. The SRM data was stored in timestamp format, based on the time in the time zone that the data was collected. In order to resolve this issue, we assumed that all servers resided in the same time zone. Additional coding would need to be done to SRM to store the data in UTC timestamp format, and eliminate the assumption that was made. This required us to predefine the servers in both the customer and center tables of the prototype schema, where the timestamp offset resided.
- Field translation: The data in the SRM tables were keyed by server id, with columns that represented the particular metrics. The prototype schema tables were keyed by server ID, with each metric residing in a different row. In order to resolve this issue, a query for each metric was made and placed into the intermediate tables during the ETL processing.

In summary, all of the items that were identified during the case study were either resolved at the time of the study, or were left as future development work. All of the items left for future development are capable of being implemented without significant resources.

5.4.2 Case study 2 results

The following lessons were learned from the AIS case study:

- Adjusting the extract control data: Remember that when testing and using the same data over again the user may want to adjust the extract control data to allow the data to reload, as in Example 5-7 on page 175.

Example 5-7 Adjusting the extract control data

```
db2 => update twg.extract_control set extctl_from_intseq=45169 where
extctl_target='SPP.STAGE_F_AISEVT_HOUR'
DB20000I  The SQL command completed successfully.
db2 =>
```

- Timestamps: The Tivoli Enterprise Data Warehouse expects timestamps in UTC. This is something that may have to be addressed in the source data. It can also be adjusted through customer and center lookup tables, but plan for this step carefully.
- Hourly data: Any time frame that data can be collected for can be used in the Tivoli Enterprise Data Warehouse. However, as it is out of the box, the Tivoli Enterprise Data Warehouse will only handle one hour data. Any other time frame can be included, but it will need extra code on the ETL.
- Product design: The design of the Tivoli Enterprise Data Warehouse and the CDW in general works very well and the case study provided actual user reports.
- Incremental extracts: Incremental extracts are critical to the long-term success of the project. This may consume large amounts of time during the initial development cycle. However, it may determine the success or failure of the project.
- Source data changes: Although the preferred method for Tivoli Enterprise Data Warehouse is not to change the source data, in some cases, it might be very difficult to accomplish the integration with the repository without changing anything in the source data. At first this might not seem natural, and seems to be problem; however, as the process goes along, it becomes obvious that this will be necessary no matter what system collects the data.
- Field conversion: Some fields were relatively small for what the case study team had in production on other systems, for example, field conversion for host names had to be changed from 64 characters to 32. This was left as a concern, but it caused no direct issues.
- Scalability: Scale will need to be considered. The case study was unable to properly test scalability. This could potentially be a problem, especially on the initial one-time feed from legacy systems.
- Adding sources and targets: When adding sources and targets there are two ways to accomplish this. The source or target folders can be right clicked and then added to, but the best way is to bring up the Properties dialog of the source/target then go to the Tables tab. This prevents the user from having to type in all the table names one-by-one.
- Reviewing data during the process: During the case study, we frequently reviewed the data in the tables to verify that the results of each step of the

work. This is especially important for troubleshooting the ETL scripts. In case of a problem, by making sure which steps produced good data, we were able to know exactly which step the problem was in.

- Table changes: If a table that has been created as a source or target changes, the changes do not automatically flow into the system. The source/target must be removed and added again as well as everywhere this source/target exists.

5.5 Best practices

The following sections review best practices as one integrates application data into the Tivoli Enterprise Data Warehouse central data warehouse.

5.5.1 Follow the Enablement Guide

The Enablement Guide is well thought out and crafted. By using it the user can leverage the experience of others, and smooth out some future hurdles before they are even encountered. The guide gives the user a solid direction as well as laying out some rules to follow. By following the rules the user can prevent wasting development effort on a system that may not be extensible or complete.

5.5.2 Fill out the data template

In any data warehouse project up to eighty five percent of the initial effort can be to load the initial dimensional data into the warehouse. Understanding how this dimensional data relates to the fact data and other dimensional data is key to minimizing this effort. The template forces the user to consider all the areas of the dimensional data before beginning. The step may seem somewhat tedious; however, once fully completed and given the due consideration, it makes implementation much easier. This is because by filling out the data template most of the hard decisions about the data are answered before development begins and because the development course is somewhat laid out.

5.5.3 Install one of the Tivoli-provided sets of ETLs

Installing the Tivoli Enterprise Data Warehouse is obviously a prerequisite, but installing an application set of ETLs may not be so obvious. *However, with another packaged application installed, there is a reference area for all other development.* It was found most useful to be able to view the naming conventions, the dialog box choices, and the ETL code of the existing application. It also provides some debug information, by running the existing steps from another application it proved all was working well except the ETL script being developed.

5.5.4 Adapt existing ETL scripts to create your own

Installing an existing application, which has ETLs ready, provides many benefits. One is the use of existing ETL code. Especially for the first attempts at developing ETLs, the user should consider copying the existing application's ETL code and altering it to meet the needs of the user. This helps establish good coding practices, and gives the user an idea of what to do and how to do it.

In our case studies, we made use of the Tivoli Distributed Monitoring ETLs as sample application ETLs.

6

How to create data marts

This chapter provides the techniques and best practices in creating data marts by using a case study.

This chapter has the following sections:

- Reasons to create data marts
- Benefits of data marts
- Methodology for creating data marts in Tivoli Enterprise Data Warehouse
- Tivoli Enterprise Data Warehouse data mart best practices

6.1 Reasons to create data marts

The value of data is highly recognized as one of business' greatest commodities and this is seen in the wide-spread construction of data warehouses throughout the IT industry. However, without the ability to view that data in a manner that the business can use, the data itself becomes useless. This has set the stage for the data mart and its ability to facilitate reporting.

To create an analogy, if data warehouses were to be viewed as actual retail warehouses, then data marts can be viewed as the showroom. Just as in a retail warehouse, the warehouse is packed full of items with the main consideration being well-organized storage with the ability to get items in and out. In contrast, the show room's consideration is presentation, showing the value of the items, and taking care of overall customer satisfaction. The warehouse has to be suited to store items of all different uses, while the showroom will focus on items of a particular use.

The analogy works well for the data warehouse and data mart. The warehouse's job is to store large amounts of data with little or no concern on exactly how the customer will view the data. While the data mart's concern is precisely the opposite. It focuses on ease of use for the customer, isolation of sensitive data, speed of reporting, and overall presentation of the data.

The data mart is where the customer gets their value and sees their return on investment. Therefore the data mart plays a crucial role in the success of the data warehouse.

A data mart is simply defined as a logically related subset of data from the compete data warehouse, normally meaning that the subset of data is related to a single business process or a group of related business processes. Data marts can be seen as the data from the data warehouse that meets a certain criteria, such as all data relating to purchase orders that falls within the date range of the last three months, or all the data relating to shipping over the last two years. Therefore, sometimes data marts are considered to be subject areas of the data warehouse. This allows the data warehouse customer to only have to work with their business area data, and not be overwhelmed with the entire business' data. This implies that there can be many data marts that get data from one central data warehouse.

6.2 Benefits of data marts

Technically, reporting could take place at the CDW; however, listed in the following section are some of the many benefits from utilizing the data mart.

6.2.1 Incremental development

By adhering to the overall architecture of the data warehouse, data marts can be designed and built separately. The process can fit into an incremental development strategy where only one data mart at a time will be delivered, which will provide the customer with benefits from the warehouse before the entire warehouse is complete. Also, separate teams can build different data marts asynchronously. As long as each data mart conforms to the data warehouse architecture, the marts can be used in conjunction with each other.

Both scenarios will provide the data warehouse development team the ability to get the data warehouse customer a product more rapidly and, hence, start the return on investment sooner. This has great advantages over attempting to complete the entire data warehouse as one single project: A project too large for even the most experienced development team.

6.2.2 Customer understandability of data

By the data mart only suppling data that matters to a specific area of the business, there is less confusion for the customer. The customer does not need to sift through data that they are not interested in. The data mart ETL process does the filtering at the data mart level.

Since the customer only has to work with data that pertains to their area of the business, they are already familiar with it. This familiarity allows them to focus more on how to use the data and less on understanding the data. Therefore, the customer's requirements are easier to gather and development time is cut. Also, less training on the report generation is needed.

By eliminating data that does not relate to a particular business area, meaningless data comparisons can be avoided. For example, if a customer were to have access to the hardware lease information and hardware performance data, the customer may be tempted to compare hardware vendors to hardware performance. When in reality, without comparing the different applications residing on the hardware, the report is basically meaningless, as the number and size of the applications will greatly affect the results of the report. This style of reporting is eliminated by only loading related data into the mart.

6.2.3 Manageable pieces

Data marts break down the complicated data design into small manageable pieces as already shown, this is helpful to the customer, but this is also helpful to the development teams. By the mart being simplistic in design it is easy to communicate across teams and design customer applications, as well as maintain them.

6.2.4 Manipulation of data in the mart

Inside the data mart the data can be aggregated, summarized, averaged, etc., to meet the specific needs of the business area. Since the mart is separate from the data warehouse as a whole, there is freedom to work with the numbers as the business chooses, without having to consider impact of the entire warehouse. As long as the lowest grain of data still exists in its original form in the data mart and the overall data warehouse architecture is complied with, the numbers can be used in any manner the business deems necessary, while the same base data could be used in other marts for other areas of the business.

6.2.5 Better reporting performance

As mentioned earlier, reporting could be done directly off the central data warehouse. However, this would require that each run of the report would have to work through all the data stored for all the different business areas. This will provide very poor performance.

By having only a subset of the data in the mart, the database system can manage the data faster and easier. Also, since the filtering has already been applied at the mart ETL, the reporting queries become much smaller. Smaller queries performing on a small subset of data is, of course, easier to tune. Therefore the end customer will experience better reporting performance.

6.2.6 Use of distributed technology

Since the data marts are smaller they can be placed on smaller distributed machines to allow data warehouse users to break away from massively powered machines and still handle processing of the reports.

6.2.7 Tool ready

Due to the standard and transparent design, several third party tools have become available for report writing. These tools can utilize the standard design and naming to support OLAP, MOLAP, and/or ROLAP technology. The data mart design and implementation is conducive to these tools and technology. This allows the user to create extremely robust reports in a very sophisticated manner.

6.3 Data mart methodology

The Tivoli Enterprise Data Warehouse is built with a full understanding of the industry data warehousing methodology. This section discusses the remaining four steps of the Enablement Guide that describe the creation of the data mart.

6.3.1 Data warehouse terminology

The following terminology is used in this chapter:

- Facts: Measurable data, and usually rapidly changing data, such as the average CPU percent busy, disk IO rates, number of TEC events, etc. It is most useful when the fact data is additive and can therefore be accumulated in some fashion that makes sense.
- Dimensions or metadata: Data about factual data, which is usually slowly changing data, such as host description, physical location, measurement description, etc. Dimensional data is also sometimes referred to as static data.
- Conformed data: Data that is conformed across all data marts, meaning that the same set of data can be used in the exact same manner in each separate data mart. The name, description, and meaning of the same piece of data across all data marts should be consistent.
- Data granularity: Level of detail of the data. This will be relative to the fact data, with some examples being hour, day, and week. Also in a retail environment could be store, district, region, and division.
- Star schema: Involves one factual table with several dimensional tables joined to it. Conceptually the fact table is the center of the star while the dimension tables create the points of the star.
- Fact tables: Tables that contain facts and keys to join to dimension tables that, however, contain no description dimensional data. Fact data usually represents one segment of time or one segment of measurement. The individual facts are stored over long periods of time to show larger cumulative segments of time, meaning that many facts will exist for the same dimension. Since fact data is stored historically, trending can be done to reveal information not readily visible in smaller segments of time.
- Dimension tables: Tables that contain actual description data about the facts and of course the keys that the fact tables used to join. Dimensions are usually stored one at a time. An example would be that a host has one description made up of physical location, hardware components, size, and purchase date. This data will remain almost constant. Therefore, this is only stored once in the data with the key that can be replicated many times over, requiring much less space.

- Aggregation: Any mathematical functions performed on data that form a summarization of data and therefore produce new rows. This usually pertains to summarizing, averaging, taking the minimum or maximum, etc. Aggregations produce new rows of data at a different granularity.

6.3.2 Methodology for data marts and star schemas

The concept of the star schema involves one factual table with several dimensional tables joined to it. Conceptually the fact table is the center of the star while the dimension tables create the points of the star. See Figure 6-1.

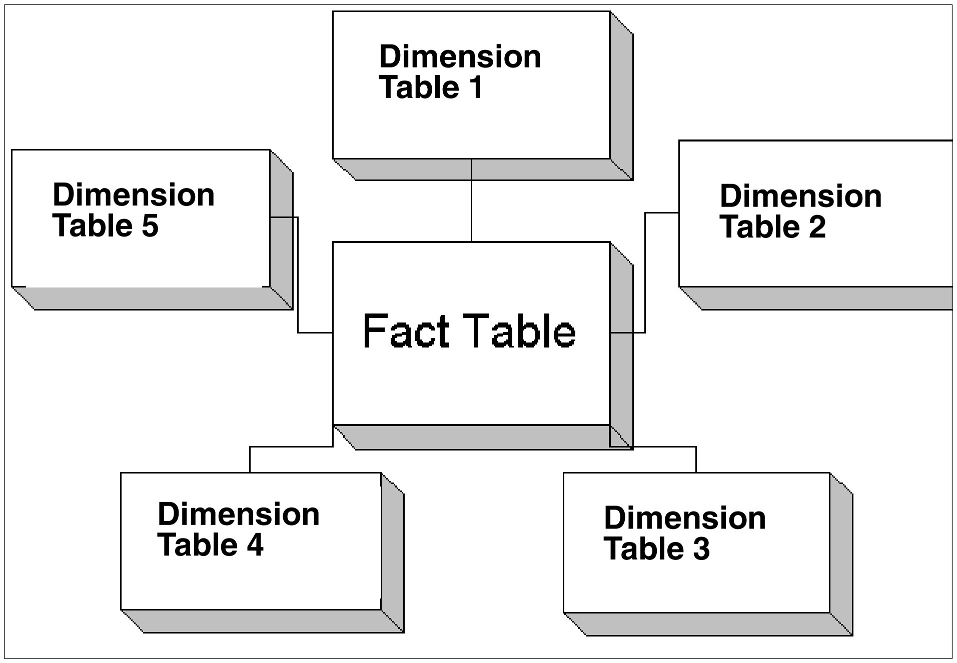


Figure 6-1 Star schema

The fact exists in the fact table with keys linking to the dimensional data that should be joined with this fact. The fact will be for exactly one type of metric, such as event count or page scan rate. The fact will not contain data for more than one metric type, meaning that page scan rate and event count will not be in the same row in the fact table.

Each row can then be joined to a dimension table containing the descriptive text about the metric. An example of the text being `Number of TEC events received`. This twenty-nine character string will only need to be stored once in the dimension table, and it will be stored with a unique key using much less space. That key is then used to associate the facts with the text description. There can be millions of facts, and without the join, the twenty nine characters would need to be stored millions of times.

This component's attributes will also contain textual strings that will be stored once in a dimensional table and joined to the fact table. The descriptive strings could be for physical country location, such as United States, Germany, or New Zealand.

This process will continue with any data that can be stored in dimension tables. Therefore, the fact table rows contain a fact combined with, as many as needed, keys to join the dimensional data for that fact.

The dimensional data then becomes the possible search criteria for reporting. A report could be generated for all hosts that are physically in the United States. The physical location is stored in the dimensional tables. The report could be further constrained by all hosts in the United States and were purchased in the last two years. Again, purchase date is stored in a dimensional table.

Each Tivoli Enterprise Data Warehouse data mart will be made up of one or more star schemas. In this manner the quality of the dimensional data translates to the quality of the data mart because the dimensions are driving the ability to search the fact data.

The Tivoli Enterprise Data Warehouse expects there to be exactly one metric dimensional table containing descriptions of the metrics, one or many component dimensional tables containing descriptions of the components, and one fact table containing the numeric data as is related to the dimensions. This implies that exactly one measurement is associated with one fact in each star schema, and that as much information about the component as needed is available.

This leverages the use of the data mart as described above, in the manner that the star schema will be used for precisely one set of reporting.

6.3.3 Tivoli Enterprise Data Warehouse naming conventions

Chapter five of the Enablement Guide for the Tivoli Enterprise Data Warehouse describes the naming convention in even further detail. Significant conventions are as follows:

- Product code: IBM applications must use the IBM internal product/AVA code. This consists of three alphabetic characters. Non-IBM applications must use a code that will be unique and is made up of two alphabetic characters and one numeric character.

- Database: Some databases used in the Tivoli Enterprise Data Warehouse are TWH_CDW as the central data warehouse, TWH_MD as the database for metadata, and TWH_MART as the database for star schemas.
- Subject areas: <Product_code>_<full_application_name>_v<full_version>_Subject_Area. Example: CTO_Tivoli_Manager_for_Oracle_v2.1.0_Subject_Area.
- Sources: <Product code>_<Source DB alias>_Source. Note that the source DB alias should be uppercase only. Example: CTO_TWH_CDW_Source.
- Targets: <Product code>_<Target DB alias>_Target. Note that target DB alias should be uppercase only. Example: CTO_TWH_MART_Target.
- Processes: <Product_code>_<Process_ID>_<Process_Description>_Process. Note that Process_ID consists of c<nn> or m<nn>, where c indicates the target is the CDW and m indicates the target is the data mart. Example: CTO_c05_Exceptions_Process.
- Steps: <Product_code>_<Process_id>_s<nnn>_<Step_description>, where s indicates a step and <nnn> is the step number. Example: CTO_c05_s010_Extract.
- Schemas: <Product_code><by_time_granularity><Warehouse_schema_description>< Star_schema>. Example: CTO Hourly Oracle Database Star Schema.
- Component dimension tables: <Product_code>.D_<dimension_table_description>. Only A through Z, 0 through 9, and underscore(_) can be used in table names in the data mart databases. Also singular (not plural) is used. For example, DATABASE not DATABASES. Example: CTO.D_ORA_DB_COMP.
- Measurement dimension tables: <Product_code>.D_<Metric_group_description>_METRIC. Component dimension table rules apply. Example: CTO.D_ORA_METRIC.
- Fact tables: <Product_code>.F_<fact_desciption>_<time_granularity>, where time granularity is HOUR, DAY, WEEK, or MONTH. Component dimension table rules apply. Example: CTO.F_ORA_DB_HOUR.
- Staging tables: <Product_code>.STAGE_<description_of_staging_table>. Component dimension table rules apply. Example: CTO.STAGE_ORA.

6.4 Moving on with case study 2 - AIS data

After setting the stage for data marts, we can move on with our case study. In this section we will cover the implementation of the data marts for the AIS data, which was explained in “Case study 2 - AIS” on page 130. In Chapter 5, “Integration of application data to central data repository” on page 127, we have shown you the implementation of the first 10 steps covered in the Enablement Guide. These steps were related to integrating your data with the central data warehouse repository. In this chapter we will cover the last four steps (step 11 through step 14), which deal with creating of the data mart, or ETL2.

Note: We do not cover the data mart implementation of SRM data (case study 1) in the chapter because the implementation steps are very similar.

6.4.1 Step 11: Define star schemas

The Tivoli Enterprise Data Warehouse concept of a star schema is that there is exactly one fact table that contains measurable fact data that contains keys to join to dimension data in two or more tables. The dimension data represents the slow-changing dimensional data, such as host name or measurement description: Data that can change, but will not change often.

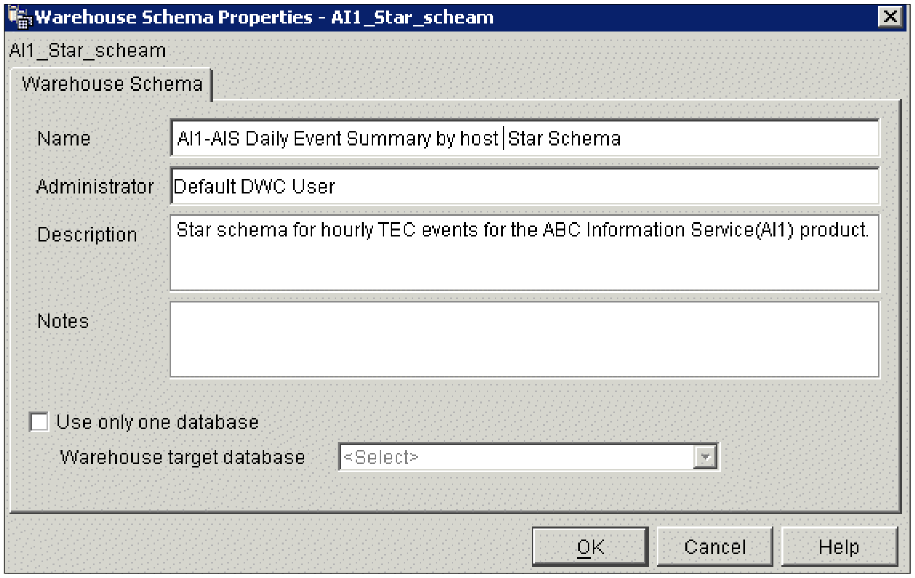
The fact table is made up of rapidly changing data, such as the performance data or the number of TEC events for the latest collection of source data. This data is appended to as the source ETL is run.

The Tivoli Enterprise Data Warehouse expects the table names to follow a naming convention where the dimension tables are named D_<Demsion Description> and the fact tables are named F_<measurement>_HOUR. Review the Enablement Guide for the latest revision of the rules for the data mart star schemas. The current set of rules are discussed in Chapter 3 of the Enablement Guide, “Creating a start schema or multiple dimensional cube.”

Creating the AIS case study star schema

To create a star schema do the following:

1. Open the Date Warehouse Center and expand **Warehouse Schemas**.
2. Right click **Warehouse Schema** in the left pane of the Date Warehouse Center and choose **Define** from the sub-menu. This produces a dialog as in Figure 6-2 on page 188.


Warehouse Schema Properties - AI1_Star_scheam

AI1_Star_scheam

Warehouse Schema

Name: AI1-AIS Daily Event Summary by host Star Schema

Administrator: Default DWC User

Description: Star schema for hourly TEC events for the ABC Information Service(AI1) product.

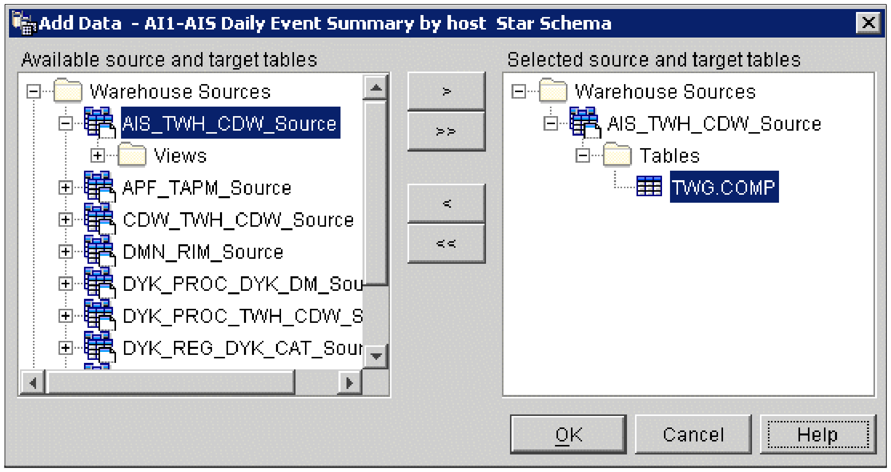
Notes:

☐ Use only one database

Warehouse target database: <Select>

Figure 6-2 Defining a star schema

3. Tables need to be added to the star schema by right clicking the **New Schema** in the left pane of the Date Warehouse Center and choosing **Add Table** from the sub-menu. This produces a dialog similar to Figure 6-3.


Add Data - AI1-AIS Daily Event Summary by host Star Schema

Available source and target tables
- Warehouse Sources
  - AIS_TWH_CDW_Source
    - Views
  - APF_TAPM_Source
  - CDW_TWH_CDW_Source
  - DMN_RIM_Source
  - DYK_PROC_DYK_DM_Sou
  - DYK_PROC_TWH_CDW_S
  - DYK_REG_DYK_CAT_Sour

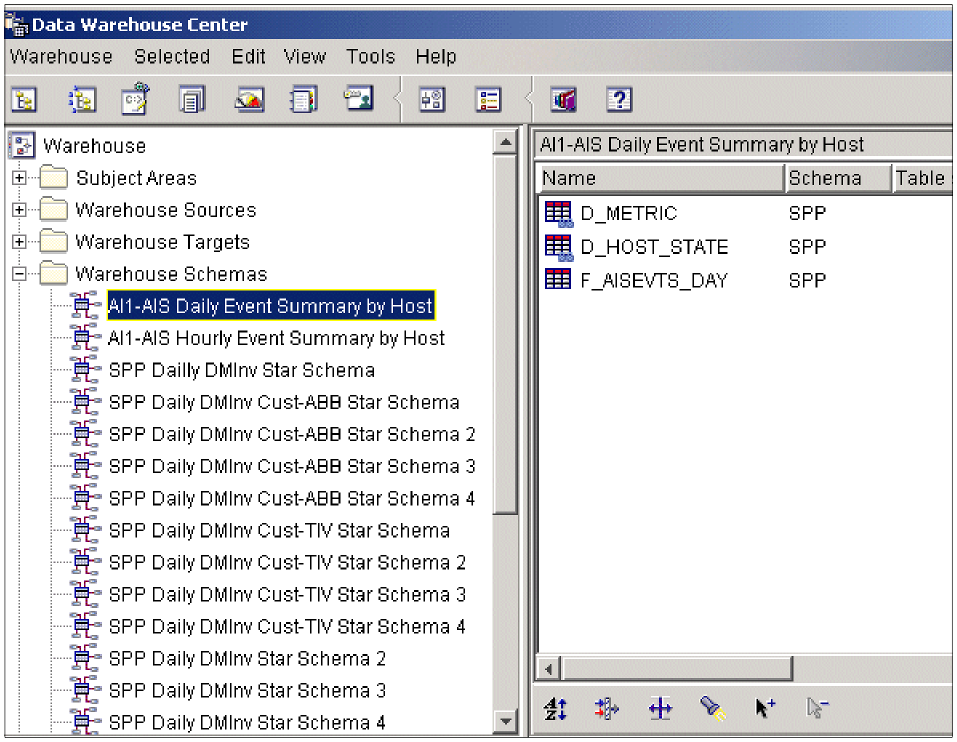
Selected source and target tables
- Warehouse Sources
  - AIS_TWH_CDW_Source
    - Tables
      - TWG.COMP

Figure 6-3 Adding tables to a star schema

4. Using the plus sign (**+**), expand the warehouse sources or targets as needed, expand the product code, then expand the tables or views folder to reveal the table or view desired.

5. Highlight the table or view and click the **Add** button denoted by the greater than sign (**>**). Once all the tables and/or views are moved to the right pane click **OK**.
6. Traditionally these tables or views should be chosen from the data warehouse targets and should be in the data mart.

   Now the Date Warehouse Center screen should look similar to Figure 6-4.



Warehouse
- Subject Areas
- Warehouse Sources
- Warehouse Targets
- Warehouse Schemas
  - AI1-AIS Daily Event Summary by Host
  - AI1-AIS Hourly Event Summary by Host
  - SPP Dailly DMInv Star Schema
  - SPP Daily DMInv Cust-ABB Star Schema
  - SPP Daily DMInv Cust-ABB Star Schema 2
  - SPP Daily DMInv Cust-ABB Star Schema 3
  - SPP Daily DMInv Cust-ABB Star Schema 4
  - SPP Daily DMInv Cust-TIV Star Schema
  - SPP Daily DMInv Cust-TIV Star Schema 2
  - SPP Daily DMInv Cust-TIV Star Schema 3
  - SPP Daily DMInv Cust-TIV Star Schema 4
  - SPP Daily DMInv Star Schema 2
  - SPP Daily DMInv Star Schema 3
  - SPP Daily DMInv Star Schema 4

AI1-AIS Daily Event Summary by Host

Name	Schema	Table
D_METRIC	SPP	
D_HOST_STATE	SPP	
F_AISEVTS_DAY	SPP	

Figure 6-4 Star schema with tables added

7. For reporting purposes these tables now need to be linked. This is done through a canvas style GUI, built into the Date Warehouse Center. To get to the canvas double click the new schema. This will produce the canvas with the tables that were added through the previous steps.

Note: The canvas will appear to have only one table. However, the tables are stacked on top of each other. Drag and drop each table to a separate area of the canvas.

8. Drag and drop each table to a separate area of the canvas so that there are no overlapping tables. The tables that relate should be positioned close together.

   As to how neatly the tables are arranged on the canvas, this matters for documentation and only takes a few minutes to clean up.

9. A left side vertical toolbar will have an icon indicating a link between two objects. This is pictured as the bottom icon in Figure 6-5. Click this icon to change modes of the canvas from a drag and drop to a linking mode.

10. Beside each column is an arrow. Click and hold on the arrow next to the column that will be used to join to another table. A line will begin at that arrow and will extend with the movement of the mouse. Position the line to end at the joining column of the second table in the join.

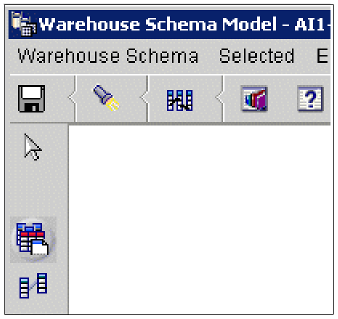
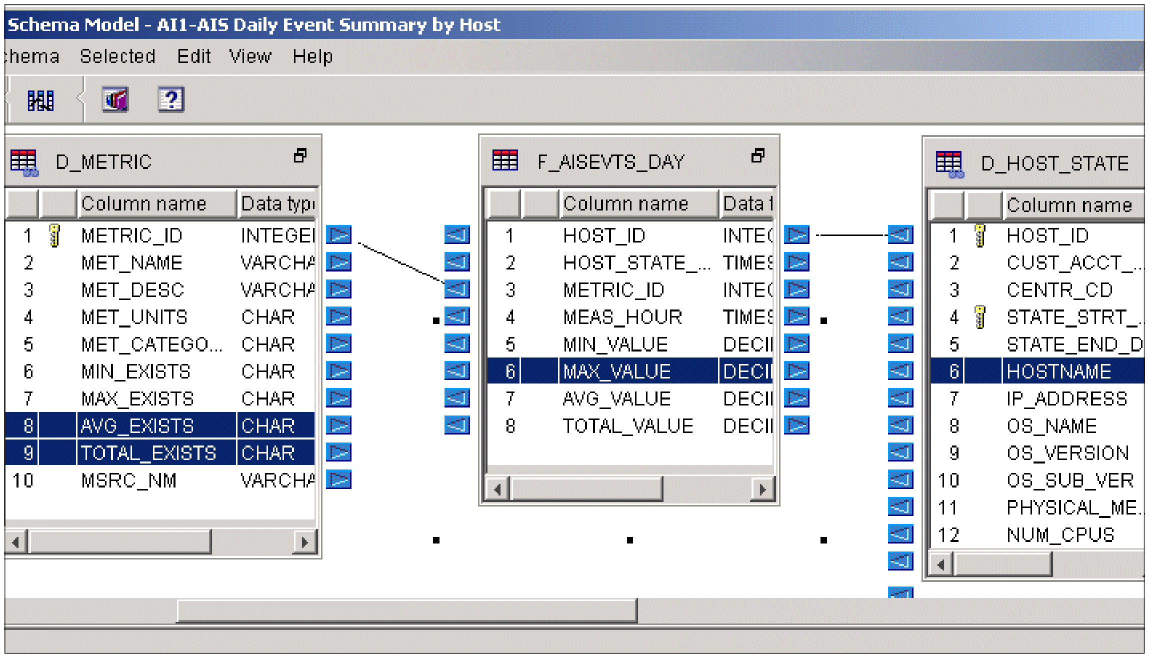


Figure 6-5   Example of model vertical toolbar

11. When complete, save this diagram using the disk icon in the upper left of the warehouse schema model screen, shown as the top left icon in Figure 6-5. Figure 6-6 on page 191 shows a completed diagram.


Schema Model - AI1-AIS Daily Event Summary by Host

D_METRIC

		Column name	Data typ
1		METRIC_ID	INTEGEI
2		MET_NAME	VARCHA
3		MET_DESC	VARCHA
4		MET_UNITS	CHAR
5		MET_CATEGO...	CHAR
6		MIN_EXISTS	CHAR
7		MAX_EXISTS	CHAR
8		AVG_EXISTS	CHAR
9		TOTAL_EXISTS	CHAR
10		MSRC_NM	VARCHA

F_AISEVTS_DAY

		Column name	Data t
1		HOST_ID	INTE(
2		HOST_STATE_...	TIMES
3		METRIC_ID	INTE(
4		MEAS_HOUR	TIMES
5		MIN_VALUE	DECII
6		MAX_VALUE	DECII
7		AVG_VALUE	DECII
8		TOTAL_VALUE	DECII

D_HOST_STATE

		Column name
1		HOST_ID
2		CUST_ACCT_..
3		CENTR_CD
4		STATE_STRT_..
5		STATE_END_D
6		HOSTNAME
7		IP_ADDRESS
8		OS_NAME
9		OS_VERSION
10		OS_SUB_VER
11		PHYSICAL_ME..
12		NUM_CPUS

Figure 6-6 AIS daily schema model

12. After saving this model diagram, the process is complete and the RPI tools should be able to access this model for reporting.

> **Note:** If a table that has been created as a source or target changes, the changes do not automatically flow into the system. The source/target must be removed and added again, as well as everywhere this source/target exists.

6.4.2 Step 12: Code the data mart ETL

Coding the data mart ETL is broader than just developing the scripts, because the user has the responsibility to design the data mart as well. Where the CDW was designed by Tivoli prior to development of the source ETLs, this step is in the hands of the user. This is one of the features that allows for such great flexibility in the design. By allowing the user to design the data mart, the reporting tool, its style, its features, and its flexibility can be changed as the user needs change.

Design considerations

The logic of the source ETLs is to gather data from one or more sources, conform this data to the CDW schema, and to store this with other data in the CDW. Whereas the logic of the mart ETLs, also called the target ETLS, is somewhat different.

The data mart will use the CDW as a source, and will join the dimensional attributes of all the facts with the facts themselves and pull them into the mart. The mart is built to allow all the data associated with a business process or group of processes together in one reporting area. All of the benefits mentioned in Section 6.2, “Benefits of data marts” on page 180 will need to be considered.

The absolute first step in designing the data mart should be to understand what the customer’s requirements are. This is a design step that is often not given the dedication that is needed. By truly understanding what the customer needs to make the data mart successful, one can design the mart and code the ETLs one time, verses not properly defining the requirements and designing and coding only to find out that the specifications were off and the process must be completed all over again, or at least major modifications will need to be done. If the requirements are understood, coding the ETL can be fairly straightforward and closely mirror the work completed on the source ETLs.

Before the mart ETLs are coded the following steps must be completed:

- Customer requirements gathered and communicated
- All customer-required aggregation routines need to be determined
- All the necessary data needs to be loaded into the CDW
- The CDW data needs to be verified to be complete
- The CDW data needs to be checked for quality
- Hardware architecture may need to be reviewed
- The design of the mart needs to be complete and refined

Implementing the design

After these steps are complete, the job of creating any new mart tables, inserting any static dimensional data needed, and coding the mart ETLs can be successful.

Once the design has been decided one will need to create tables in the mart that will be populated. Simply open the DB2 control center, open the appropriate database, right click **Tables**, then choose **Define** from the sub-menu. Fill in the name, description, and column definitions needed.

This of course can be accomplished through SQL data definition language as in Example 6-1. Note the naming of the tables in Example 6-1 as they must follow the Enablement Guide rules for data marts.

Example 6-1 Creation of data mart tables

```
--- this table must be in the cdw and the mart for us
CREATE TABLE "SPP"."F_AISEVTS_HOUR"  (
        "HOST_ID" INTEGER NOT NULL ,
        "HOST_STATE_STRT_DTTM" TIMESTAMP NOT NULL ,
        "METRIC_ID" INTEGER NOT NULL ,
        "MEAS_HOUR" TIMESTAMP NOT NULL ,
        "MIN_VALUE" DECIMAL(9,2) ,
        "MAX_VALUE" DECIMAL(9,2) ,
        "AVG_VALUE" DECIMAL(9,2) ,
        "TOTAL_VALUE" DECIMAL(9,2) )
       IN "USERSPACE1" ;

-- this table must be in the mart
CREATE TABLE "SPP"."F_AISEVTS_DAY"  (
        "HOST_ID" INTEGER NOT NULL ,
        "HOST_STATE_STRT_DTTM" TIMESTAMP NOT NULL ,
        "METRIC_ID" INTEGER NOT NULL ,
        "MEAS_HOUR" TIMESTAMP NOT NULL ,
        "MIN_VALUE" DECIMAL(9,2) ,
        "MAX_VALUE" DECIMAL(9,2) ,
        "AVG_VALUE" DECIMAL(9,2) ,
        "TOTAL_VALUE" DECIMAL(9,2) )
       IN "USERSPACE1" ;

-- this table must be in the mart
CREATE TABLE "SPP"."F_AISEVTS_WEEK"  (
        "HOST_ID" INTEGER NOT NULL ,
        "HOST_STATE_STRT_DTTM" TIMESTAMP NOT NULL ,
        "METRIC_ID" INTEGER NOT NULL ,
        "MEAS_HOUR" TIMESTAMP NOT NULL ,
        "MIN_VALUE" DECIMAL(9,2) ,
        "MAX_VALUE" DECIMAL(9,2) ,
        "AVG_VALUE" DECIMAL(9,2) ,
        "TOTAL_VALUE" DECIMAL(9,2) )
       IN "USERSPACE1" ;

-- this table must be in the mart
CREATE TABLE "SPP"."F_AISEVTS_MONTH"  (
        "HOST_ID" INTEGER NOT NULL ,
        "HOST_STATE_STRT_DTTM" TIMESTAMP NOT NULL ,
        "METRIC_ID" INTEGER NOT NULL ,
        "MEAS_HOUR" TIMESTAMP NOT NULL ,
        "MIN_VALUE" DECIMAL(9,2) ,
        "MAX_VALUE" DECIMAL(9,2) ,
        "AVG_VALUE" DECIMAL(9,2) ,
        "TOTAL_VALUE" DECIMAL(9,2) )
       IN "USERSPACE1" ;
```

Inserting data into the static tables can be done through various means, such as imports using the DB2 control center GUI, command line imports, or simple SQL inserts. In Example 6-2 we populated the stage table with new dimensional data, then the data was entered into the dimensional data in the mart automatically. This prevented us from loading directly into a table that will be reported on and allowed us to test dynamic changes to the dimensional data. In Example 6-2 we also inserted data into the extract control tables to allow the system to know of the data that had been inserted into the stage table and will need to be loaded.

These are just examples and the data can be loaded a number of different ways.

Example 6-2   Sample inserts of static data

```
db2 => insert into spp.stage_d_metric values (0, current timestamp, 'EvtCnt',
'Event Summary Count', 'QTY','EVT','N','N','N','Y','TEC')
DB20000I  The SQL command completed successfully.
db2 =>
db2 => insert into twg.extract_control values
('SPP.STAGE_D_METRIC','SPP.D_METRICS',null,null,31,31,current timestamp,
current timestamp)
DB20000I  The SQL command completed successfully.
db2 =>
```

Developing ETL code

Coding the mart ETL follows the same steps as coding the source ETLs in Section 5.3.10, “Step 10: Code the source ETL” on page 151. The source steps can be followed to create, test, and implement the mart ETLs. Changes will need to be made to comply with the naming conventions, which will now include an m in place of the c, as in the name of the ETL script ai1_m05_s010_buildMart where the m05 indicates that this is a mart script and is process number 05.

Developing aggregation routines

Aggregations are any mathematical functions performed on data that form a summarization of the data, therefore producing new rows and usually pertaining to summarizing, averaging, and taking the minimum or maximum. Aggregations produce new rows of data at a different granularities.

Some aggregation routines can be included in the ETLs. As data is moved from the source to the target, it may be simpler to add another query that will do the calculations and store the data in the target. In other situations the aggregation routine will be run as a separate step in the process in the Date Warehouse Center.

Important: Just to make sure the point is clear, the aggregation task should only populate the mart, and never the CDW.

Aggregation and rollup via the Tivoli-provided script

When the data mart ETL process runs, it populates the data mart F_<>_HOUR table with data. To support rollup or aggregation to the F_<>_DAY, F_<>_WEEK, and F_<>_MONTH tables, the user-defined program named rollup should be run. The rollup program is installed with Tivoli Enterprise Data Warehouse. It requires a table named STAGE_F_<>_HOUR to be created and populated with today's data in the application-specific schema in the central data warehouse. It then populates F_<>_DAY, F_<>_WEEK, and F_<>_MONTH tables in the data mart based on the data in STAGE_F_HOUR. It also populates RPI.SSUpdated to enable report scheduling. The process flow looks like the following:

```
twg.msmt .execsql.STAGE_F_HOUR and F_HOUR
                   |
                   ->.rollup.RPI.SSUpdated [optionally .runReport]
```

This assumes that the central data warehouse ETL is providing data summarized on an hourly basis; otherwise, the rollup.sh script does not work. In other words, if your application happens to report daily data only, then the rollup.sh script will not work for you. Also, you must place your star schema in the TWH_MART database or create the DAY_AGGREG table in your own data mart database.

The rollup program updates any of the aggregate tables as appropriate. It handles the updating of any aggregate tables if out of order data is detected. For any of the aggregate tables that are updated, it determines which star schemas have been updated and places an entry in the RPI.SSUpdated table. The report gets rerun when the runReport user-defined program is run if the following are true:

- The RPI.SSUpdated table has an entry for the star schema indicating that the data is new.
- When the user created a report in the Report Interface GUI, they selected the option to schedule reports.
- The parameters passed to rollup.sh are:
  - Schema name or product code
  - Name of hourly fact table

- Name of hourly stage table
- &SDB
- &SUID
- &SPWD
- &TDB
- &TUID
- &TPWD
- &STEPNAME

Note: The first two parameters are used to construct the temporary and log file names.

Most of the parameters need to be modified by users as they configure the ETL for their environment. (Initially, the parameters are modified by ETL developers, but users need to modify the parameters to fit their environment.) The others are set up using Data Warehouse Center variables and are filled in by the Data Warehouse Center, for example, &SDB, which is the warehouse source database name. In the Report Interface case, this is the data mart database.

Aggregation and rollup outside the Tivoli-provided script

Remember that the tables in the mart are simply RDBMS tables and can be used at the user's discretion. The study actually created a rollup script for the case study. This was for two major reasons. One was to show how a variation can be used to achieve desired results. This script could be aggregating data that was for a time frame of a half hour and therefore would have been out of scope for the Tivoli rollup script. And the second was that the Tivoli-provided rollup script would not work for summarizing TOT_VAL columns in the hour table. It would work for the min, max, and average, but not for the total value. This was addressed and we expect it to be corrected soon, so this should not be a problem for the reader.

The case study team followed the rules for the mart to conform so that the rollup.sh script could function if applied. This way we would have the advantages this schema and naming would provide, and also be flexible should anything change in the future. Plus, as new applications are provided they will be developed to work with the mart rules. Therefore, in the future if an application comes out that could provide the same or better features as what we have developed, we could migrate from our developed code into the Tivoli-supported code.

Provided in Example 6-3 on page 197 is a portion of the rollup code. The script simply did this process over again for each time frame needed.

Example 6-3 Code excerpt of rollup ETL developed in the case study

```
--  #EXECUTE_AT_TARGET
insert into spp.f_aisevts_day
select   d.host_id as host_id,
         current timestamp as host_state_strt_dttm,
         max(d.metric_id) as metric_id,
         char(date(d.Meas_hour),iso)||' 00:00:00.000000' as meas_hour,
         SUM(d.tot_val) as tot_val
from     spp.f_aisevts_hour d
group by d.host_id,host_state_strt_dttm, metric_id, date(d.meas_hour)
;
```

This script was passed as a parameter to the sqlscript.sh in a process step and, therefore, was set up just as all other ETL scripts. This is an excellent example of the power and flexibility of the sqlscript.sh combined with the CDW and mart designs.

Once this script completes, data is in the mart in an aggregated format as well, as at the detail level.

Implementing the Tivoli-provided rollup script

Do the following steps to implement the Tivoli-provided rollup script by creating a step inside the process.

1. Open the Date Warehouse Center and expand **Subject Areas** and **Process**.
2. Now to add a step to the process for the rollup, right click the process in the left window and follow the menu path to **Define -> User defined Programs and Transformers -> Tivoli -> Rollup** as in Figure 6-7 on page 198.

   This produces a dialog to configure the process, which has four tabs.

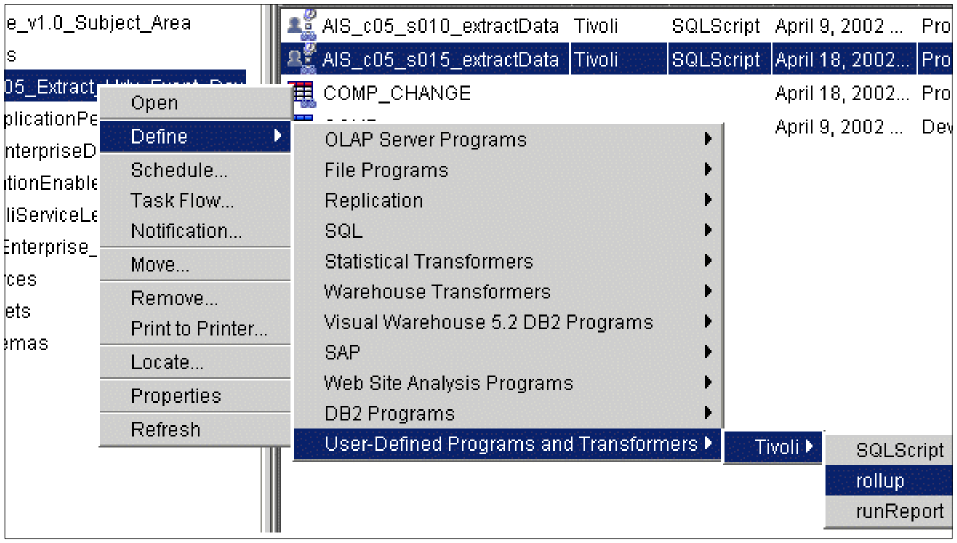
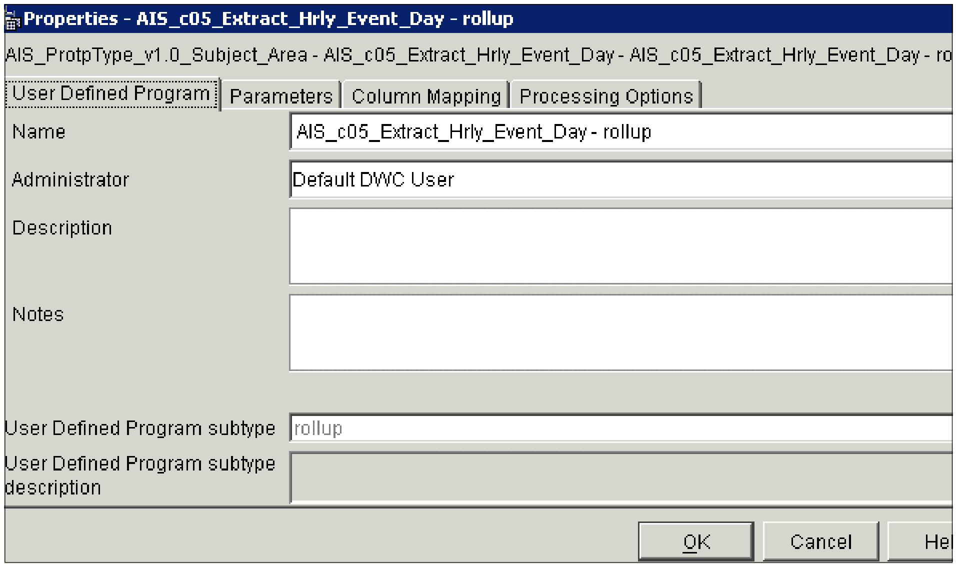


Figure 6-7 Sub-menu for defining a Tivoli rollup script

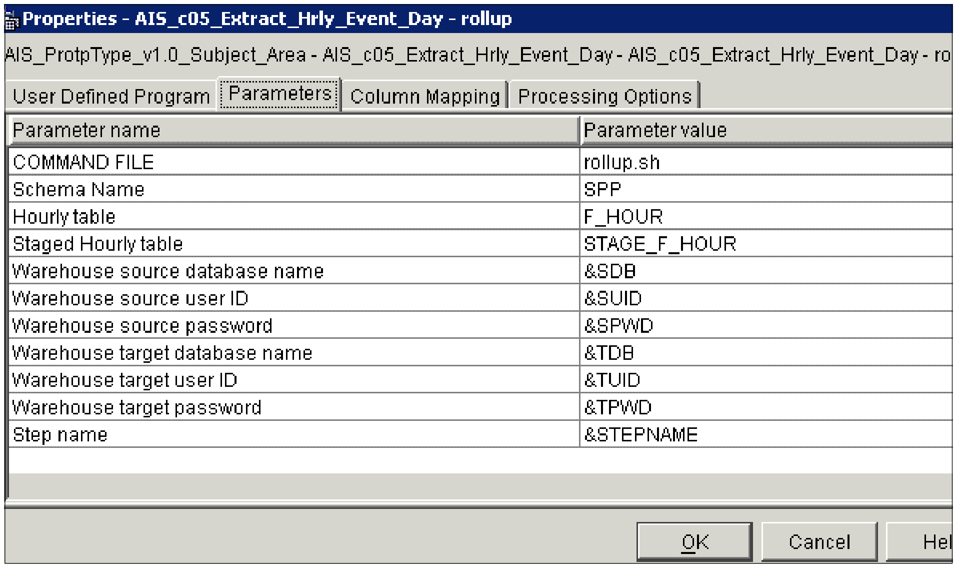
3. The first tab, as in Figure 6-8, contains the name, description, and notes and should be filled in appropriately. The naming of this process step should be chosen carefully, again referencing the Enablement Guide to review how the process steps should be named.


Properties - AIS_c05_Extract_Hrly_Event_Day - rollup
AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day - ro
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel
He

Name	AIS_c05_Extract_Hrly_Event_Day - rollup
Administrator	Default DWC User
Description	
Notes	
User Defined Program subtype	rollup
User Defined Program subtype description	

Figure 6-8 Defining a step with a rollup script

4. The second tab, shown in Figure 6-9, is the Parameters tab, which contains the details about the process step. The first field is the command file. This will be prefilled with the rollup.sh, which was filled when the user selected rollup script from the pop-up menu. This script is covered in-depth in the Enablement Guide.


Properties - AIS_c05_Extract_Hrly_Event_Day - rollup
AIS_ProtpType_v1.0_Subject_Area - AIS_c05_Extract_Hrly_Event_Day - AIS_c05_Extract_Hrly_Event_Day - ro
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel
Hel

Parameter name	Parameter value
COMMAND FILE	rollup.sh
Schema Name	SPP
Hourly table	F_HOUR
Staged Hourly table	STAGE_F_HOUR
Warehouse source database name	&SDB
Warehouse source user ID	&SUID
Warehouse source password	&SPWD
Warehouse target database name	&TDB
Warehouse target user ID	&TUID
Warehouse target password	&TPWD
Step name	&STEPNAME

Figure 6-9 Defining a step with a rollup script - Parameters tab

The schema name is the schema name where the data mart's hourly table exists. The hourly table is the hour fact table tablename, named similar to F_AISEVTS_HOUR. The stage hourly table is the staging table used on the CDW.

The remaining fields are the variables used for the sqlscript.sh to determine the target and source, which do not need to be adjusted.

5. The next tab is Column Mapping, which is not available for this type of process.
6. The last tab is the Processing Options and can used for debugging options.
7. Click **OK** to complete the process step creation.

   Once the step has been defined, the steps in the source ETL section can be used to complete the step. The step will need to have sources and targets added, as well as needing to be scheduled and put into production mode.

   The steps for testing source ETLs apply to testing the rollup ETL as well.

6.4.3 Step 13: Provide internationalization strings

Due to time constraints for the case studies, internationalization was left out of the studies. The documentation provided by the Enablement Guide in Chapter 7 explains how to prepare an application for international languages. Tivoli Enterprise Data Warehouse is written in Java, and international support is provided through the use of Java resource files. These resource files are a modular addition to Tivoli Enterprise Data Warehouse, and no changes are needed to the existing table entries. By default, Tivoli Enterprise Data Warehouse is in English.

6.4.4 Step 14: Create the warehouse enablement pack

The warehouse pack is used by Tivoli to separately package Tivoli Enterprise Data Warehouse applications. Since the authors were not making an application to be sold separately, no additional packaging was needed, and no enablement pack was created.

6.5 Data mart best practices

The following are best practices as one configures an implementation and creates new data marts for the Tivoli Enterprise Data Warehouse.

6.5.1 Break steps into the smallest steps possible

By breaking down the entire process into several small steps it is easier to debug the process when an error occurs. Also it provides a design that will be easy to understand and follow. Therefore, even in the design phase, the benefit will be that the development team will easily understand what each step will accomplish.

6.5.2 Data mart data should be kept at the lowest grain

In some development circles it is thought that data marts house only aggregated data. Some development teams feel they can save space by not duplicating the data from the warehouse to the mart. They propose that aggregation routines should be run to populate the data mart. However, this leaves the data mart without the ability to report on the lowest level of detail. Potentially leaving a customer with an answer that is not as detailed as needed.

By storing the data in the mart at the lowest grain level, the customer can be provided with both the aggregation data and the detailed information. This requires some more space and small amounts of effort on the part of the development team. However, the customer ends up with a better product.

6.5.3 Develop initial and incremental data loads together

There are two types of ETLs that must be constructed for each target and source of the data warehouse. The initial one-time load to prime the data warehouse and then the scheduled incremental loads to keep the data current. Since both require an understanding of the source data, target data, and any conversion processes, it makes sense to develop both sets of code at the same phase of the project. This may not mean simultaneously coding both scripts, but one can be coded immediately following the other.

7

OLAP integration

This chapter describes how you can integrate Tivoli Enterprise Data Warehouse data with OLAP tools such as Brio, Business Objects, and Cognos.

This chapter has the following sections:

- OLAP
- Brio implementation
- Business Objects implementation
- Cognos implementation

7.1 OLAP

Online Analytical Processing (OLAP) is a technology used in creating decision support software that allows application users to quickly analyze information that has been summarized into multidimensional views and hierarchies. By summarizing predicted queries into multidimensional views prior to run time, OLAP tools can provide the benefit of increased performance over traditional database access tools.

OLAP functionality is characterized by dynamic multi-dimensional analysis of consolidated enterprise data supporting end user analytical and navigational activities including:

- Calculations and modeling applied across dimensions, through hierarchies, and/or across members
- Trend analysis over sequential time periods
- Slicing subsets for on-screen viewing
- Drill-down to deeper levels of consolidation
- Reach-through to underlying detail data
- Rotation to new dimensional comparisons in the viewing area

7.2 Brio Intelligence

This section will give you an overview and components of Brio, and a basic example of integration with Brio with Tivoli Enterprise Data Warehouse and sample reports.

7.2.1 Brio overview

For query and analysis, Brio Intelligence has an easy-to-use set of tools available and some more advanced features. Leveraging data from existing enterprise information systems, such as Tivoli Enterprise Data Warehouse, Brio Intelligence provides executives, analysts, developers and employees with query and analysis capabilities supported by an intuitive, Web-enabled interface delivering business-critical information. For more information go to the Brio Web site at `http://www.brio.com`.

7.2.2 Brio components

Brio has client tools for both the client/server and the Web-based user, which can meet the needs of each user type across the enterprise. The Brio Intelligence product suite includes:

- Broadcast Server: Enables IT to create queries, analyses, and reports, and to schedule them for processing based on date and time or an event (e.g., a database update). The results can be published in a number of formats that best suit the end user's needs, including HTML, CSV, XLS, and Brio (BQY) files, as well as printed reports. The reports best suited for the Broadcast Server are those that involve queries that are resource intensive, and are thus best processed during off hours. With the ability to e-mail Excel files, extranet end users can also benefit.
- OnDemand Server: Allows Web browser-based end users to create and execute Brio files on demand. With no intervention from IT, end users can query the database and retrieve data to their desktops. The Adaptive Report functionality allows different end users to view the same report with different capabilities, thus minimizing the number of reports that IT needs to manage. One end user may only be able to view the data, while another may be able to re-query the database and analyze the data. The reports best suited for the OnDemand Server are those that require the end user to specify their selection parameters and those that are needed on an on-demand basis, and are therefore hard to schedule.
- Designer: The client/server developer's version of Brio. Designers can create Brio documents that incorporate not only reports for end users, but also data models, security, and auditing. Designers hold the "key" to the repository and control which documents are made available to end-users.
- Explorer: The client/server version of Brio that is used by those who will build a majority of the reports from the data models that have already been created using Brio Designer or by directly accessing the database, thus by-passing the repository. These are the power users who know the right business questions to ask and how to ask them using Brio.
- Navigator: The client/server version of Brio that is designed for those with relatively lightweight analysis needs, such as novice users. These users only need data access via the Brio repository, which contains pre-built documents created and populated by the Designer. They are protected from generating invalid queries by working in this predefined and controlled environment.

- Insight users: Web browser-based users that have varying levels of functionality based on the adaptive report and user security. They can retrieve documents posted by the Broadcast Server or in conjunction with the OnDemand Server they have the ability to query data over the Web and manipulate the results on their desktops. They have nearly identical functionality to the Brio Navigator user without the expense and maintenance of database connectivity middleware.
- Quickview users: Web browser-based users that only need to view and print Brio documents. The Broadcast Server delivers these documents, and the OnDemand Server can give them the ability to refresh the documents with new data on demand. For users that only need to view prepared data, and want to do so as easily and efficiently as possible, Quickview offers the ideal solution.

7.2.3 Brio integration with Tivoli Enterprise Data Warehouse

This section will only show the Brio Designer component integration with Tivoli Enterprise Data Warehouse. In this example we are using BrioQuery Designer Version 6.1. We will cover the setting up of the integration and running reports.

Setting up the integration

We installed Brio Designer onto a Windows 2000 system using the installation instructions supplied with the product. We also installed the DB2 Client software and configured an ODBC connection to the TWH_MART database.

We did the following to set up the integration:

1. Set up the OBDC connections to the TWH_MART and TWH_MD databases.
2. Start the Brio Designer program by clicking **Start -> Programs -> BrioQuery Designer** and select **BrioQuery Designer**.

   The Welcome to BrioQuery window will appear (Figure 7-1 on page 207).

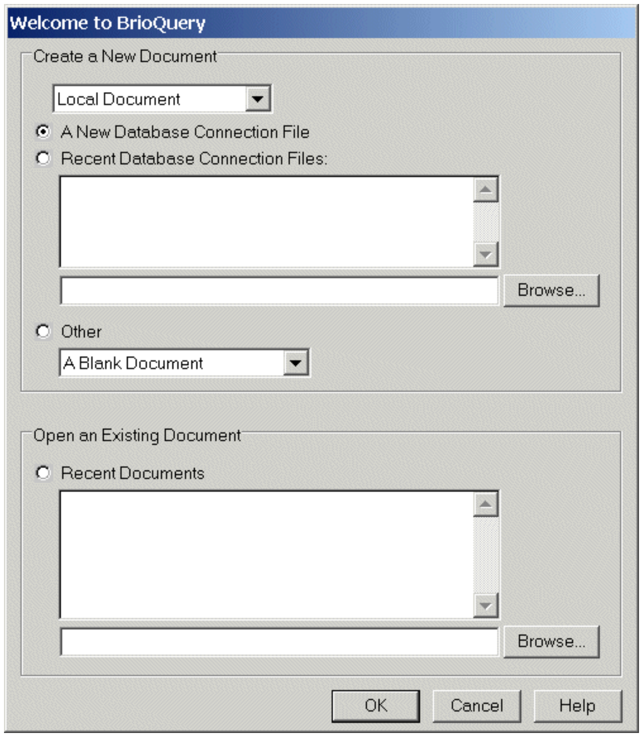
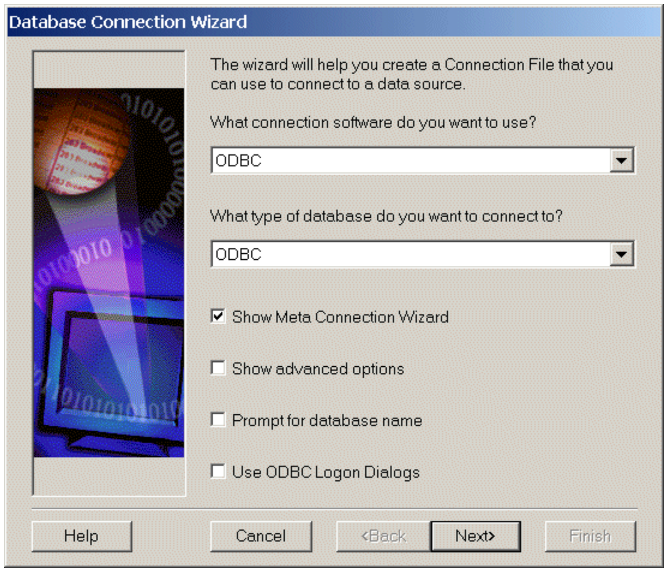


Figure 7-1 Welcome to BrioQuery window

3. Check **New Database Connection File** and click on **OK**. The Database Connection Wizard will appear (Figure 7-2 on page 208).


Database Connection Wizard

The wizard will help you create a Connection File that you can use to connect to a data source.

What connection software do you want to use?

ODBC

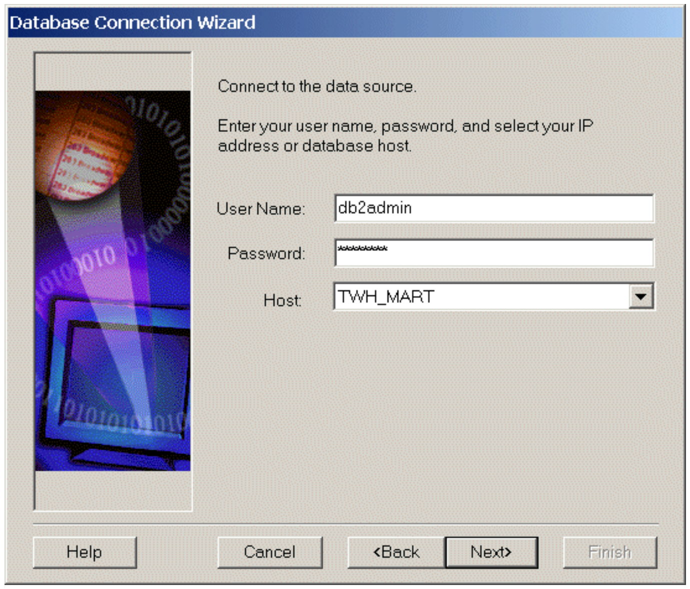
What type of database do you want to connect to?

ODBC

- [x] Show Meta Connection Wizard
- [ ] Show advanced options
- [ ] Prompt for database name
- [ ] Use ODBC Logon Dialogs

Figure 7-2 Connection Wizard window

4. Select **OBDC** from the drop-down bar and click **Next**. The Database Connection Wizard window will appear (Figure).


Database Connection Wizard

Connect to the data source.

Enter your user name, password, and select your IP address or database host.

User Name: db2admin

Password: ********

Host: TWH_MART

Figure 7-3 DB2 configuration window

5. Enter the connection user name and password and for Host and enter the ODBC connection name.
6. Click **Next** and the conformation window will appear. Select **Finish** to save the connection.

Creating reports with Brio

Now we are ready to set up and run a query. To do this we did the following:

1. Using the connection from the above step, expand the table and database tree and select the tables to run a query from, right click a table to query, and select **Add Selected Items** (Figure 7-4).

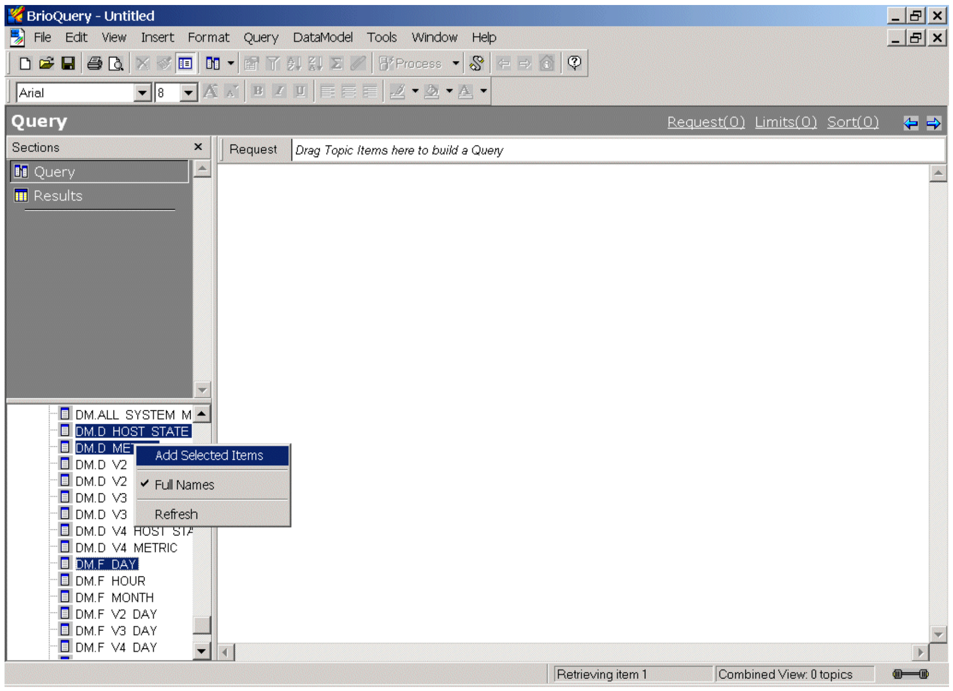


Figure 7-4 Add selected items window

2. The tables selected will now move to the right-hand window pane. Now select the items to be queried and drag and drop them into the **Request** tool bar (Figure 7-5 on page 210).

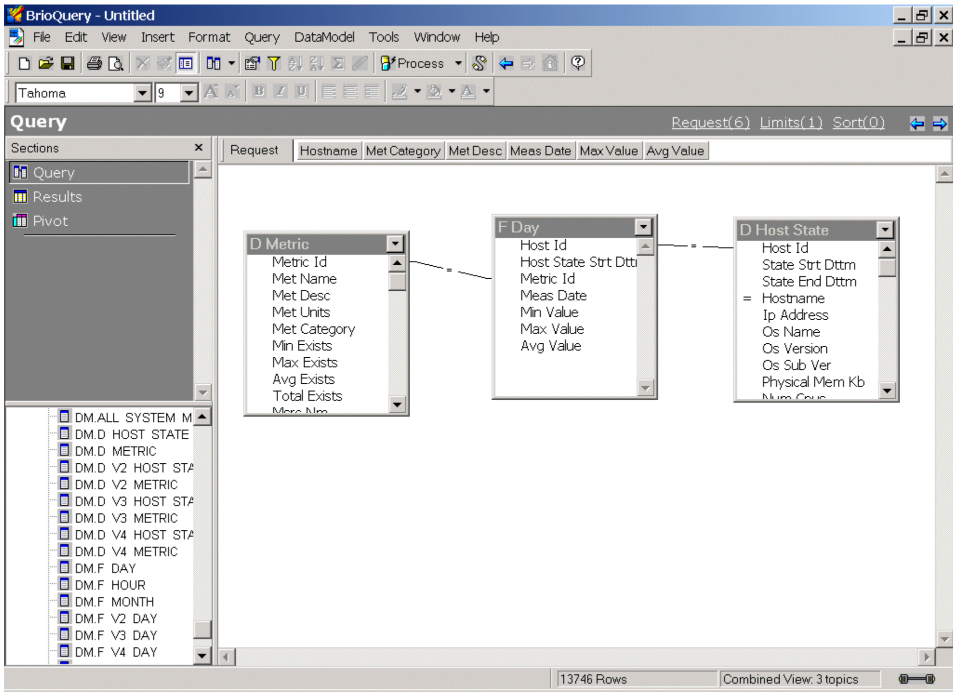
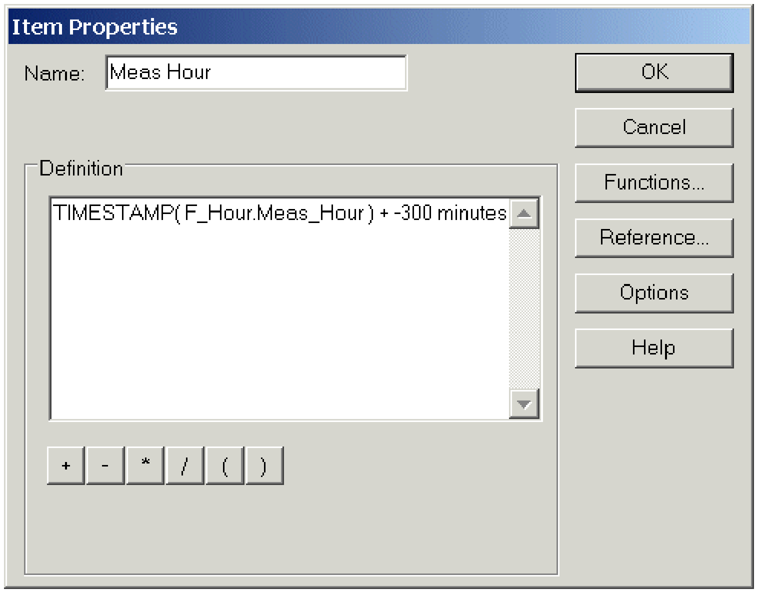


Figure 7-5 Item request window

3. When data is inserted into the TWH_MART database, time is converted to the UTC time format, so if you want to display the time in your local time, it needs to be converted to the correct time zone. To do this in Brio:

   a. Right click the date item (**Meas Hour**) in the Request tool bar.

   b. Select **Properties** and enter a SQL statement to calculate the correct time. See Figure 7-6 on page 211.



Name: Meas Hour

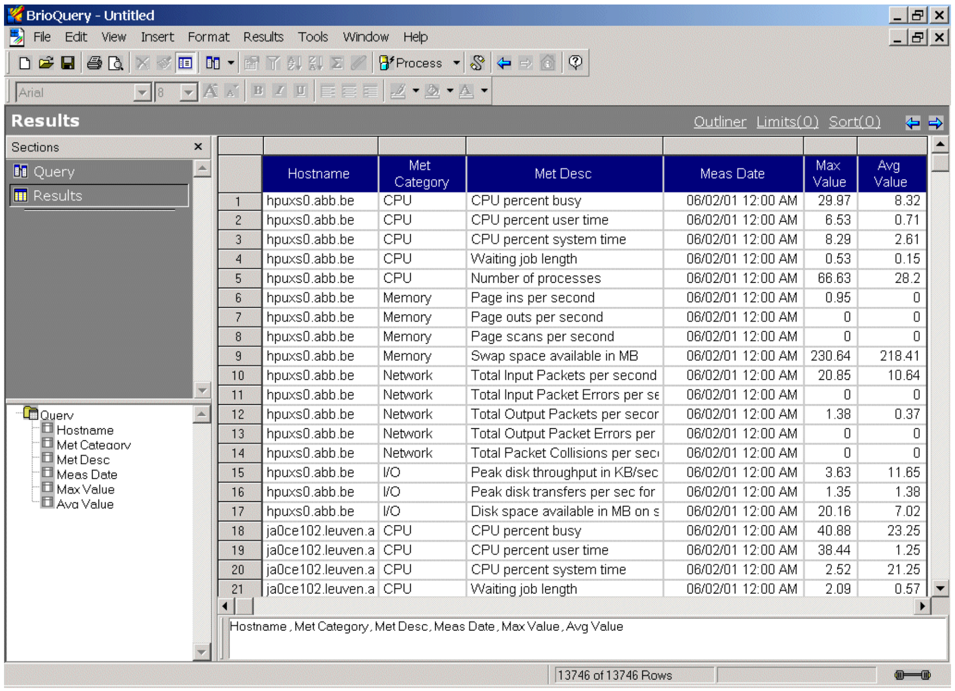
```
TIMESTAMP( F_Hour.Meas_Hour ) + -300 minutes
```

Figure 7-6 Change time window

Important: The above example assumes that there are five hours difference between the local time and the Greenwitch time. This is the case with the Eastern time. You need to substitute the correct value (or offset) depending on your location. For example, if you are in the central time zone, you need to use 360 minutes, since the difference between the Greenwitch time is six hours.

You do not need this step when using the Report Interface, because when installing Tivoli Enterprise Data Warehouse, the difference between the local time and the Greenwitch time is automatically recorded in the gmt-offset column of the TWD.DAY_AGGREG table. The Report Interface uses this value to automatically compensate the difference in time zones.

4. To process this query select the **Process** button. To view the results, select the **Results** section in the left hand frame (Figure 7-7 on page 212).



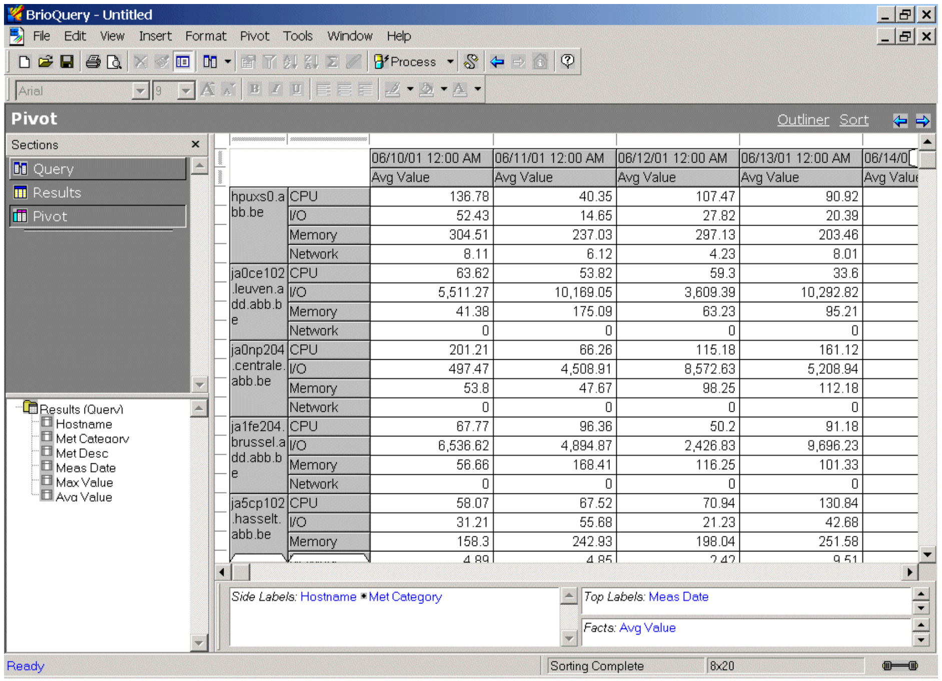
	Hostname	Met Category	Met Desc	Meas Date	Max Value	Avg Value
1	hpuxs0.abb.be	CPU	CPU percent busy	06/02/01 12:00 AM	29.97	8.32
2	hpuxs0.abb.be	CPU	CPU percent user time	06/02/01 12:00 AM	6.53	0.71
3	hpuxs0.abb.be	CPU	CPU percent system time	06/02/01 12:00 AM	8.29	2.61
4	hpuxs0.abb.be	CPU	Waiting job length	06/02/01 12:00 AM	0.53	0.15
5	hpuxs0.abb.be	CPU	Number of processes	06/02/01 12:00 AM	66.63	28.2
6	hpuxs0.abb.be	Memory	Page ins per second	06/02/01 12:00 AM	0.95	0
7	hpuxs0.abb.be	Memory	Page outs per second	06/02/01 12:00 AM	0	0
8	hpuxs0.abb.be	Memory	Page scans per second	06/02/01 12:00 AM	0	0
9	hpuxs0.abb.be	Memory	Swap space available in MB	06/02/01 12:00 AM	230.64	218.41
10	hpuxs0.abb.be	Network	Total Input Packets per second	06/02/01 12:00 AM	20.85	10.64
11	hpuxs0.abb.be	Network	Total Input Packet Errors per se	06/02/01 12:00 AM	0	0
12	hpuxs0.abb.be	Network	Total Output Packets per secor	06/02/01 12:00 AM	1.38	0.37
13	hpuxs0.abb.be	Network	Total Output Packet Errors per	06/02/01 12:00 AM	0	0
14	hpuxs0.abb.be	Network	Total Packet Collisions per seco	06/02/01 12:00 AM	0	0
15	hpuxs0.abb.be	I/O	Peak disk throughput in KB/sec	06/02/01 12:00 AM	3.63	11.65
16	hpuxs0.abb.be	I/O	Peak disk transfers per sec for	06/02/01 12:00 AM	1.35	1.38
17	hpuxs0.abb.be	I/O	Disk space available in MB on s	06/02/01 12:00 AM	20.16	7.02
18	ja0ce102.leuven.a	CPU	CPU percent busy	06/02/01 12:00 AM	40.88	23.25
19	ja0ce102.leuven.a	CPU	CPU percent user time	06/02/01 12:00 AM	38.44	1.25
20	ja0ce102.leuven.a	CPU	CPU percent system time	06/02/01 12:00 AM	2.52	21.25
21	ja0ce102.leuven.a	CPU	Waiting job length	06/02/01 12:00 AM	2.09	0.57

Hostname , Met Category , Met Desc , Meas Date , Max Value , Avg Value

13746 of 13746 Rows

Figure 7-7 Results section window

5. To create a report from this data select **Insert -> New Pivot** then drag and drop an item from the Results tree to the Side Labels, Top Labels, and Facts window panes as in Figure 7-8 on page 213.



		06/10/01 12:00 AM	06/11/01 12:00 AM	06/12/01 12:00 AM	06/13/01 12:00 AM	06/14/0
		Avg Value	Avg Value	Avg Value	Avg Value	Avg Valu
hpuxs0.abb.be	CPU	136.78	40.35	107.47	90.92	
	I/O	52.43	14.65	27.82	20.39	
	Memory	304.51	237.03	297.13	203.46	
	Network	8.11	6.12	4.23	8.01	
ja0ce102.leuven.add.abb.be	CPU	63.62	53.82	59.3	33.6	
	I/O	5,511.27	10,169.05	3,609.39	10,292.82	
	Memory	41.38	175.09	63.23	95.21	
	Network	0	0	0	0	
ja0np204.centrale.abb.be	CPU	201.21	66.26	115.18	161.12	
	I/O	497.47	4,508.91	8,572.63	5,208.94	
	Memory	53.8	47.67	98.25	112.18	
	Network	0	0	0	0	
ja1fe204.brussel.add.abb.be	CPU	67.77	96.36	50.2	91.18	
	I/O	6,536.62	4,894.87	2,426.83	9,696.23	
	Memory	56.66	168.41	116.25	101.33	
	Network	0	0	0	0	
ja5cp102.hasselt.abb.be	CPU	58.07	67.52	70.94	130.84	
	I/O	31.21	55.68	21.23	42.68	
	Memory	158.3	242.93	198.04	251.58	
		4.89	4.85	2.42	9.51	

Side Labels: Hostname • Met Category

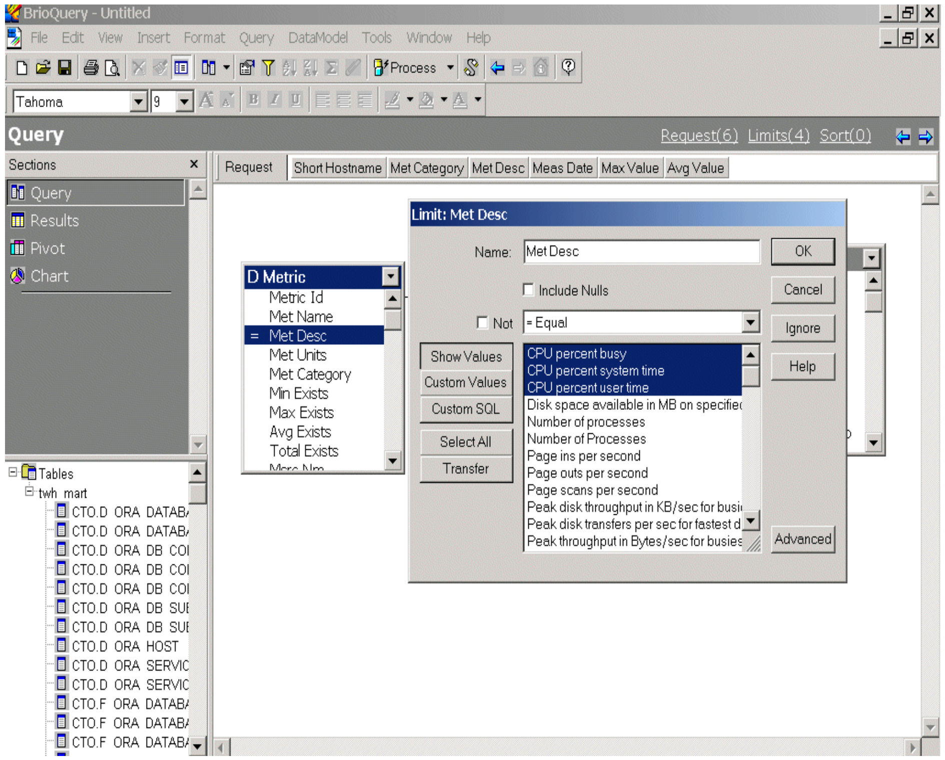
Top Labels: Meas Date

Facts: Avg Value

Figure 7-8 Crosstab report window

6. You would not want to create a graph of this data (because there is too much data), so we want to limit the amount of data. To limit the amount of data, go to the query section in the left-hand frame, select the item to limit the data, and click **Query -> Add Limits** the limit window will appear.

7. Select what you want to limit by using an SQL statement, We selected to limit only CPU metric descriptions for all host names over a seven-day period (Figure 7-9 on page 214).



Name: Met Desc

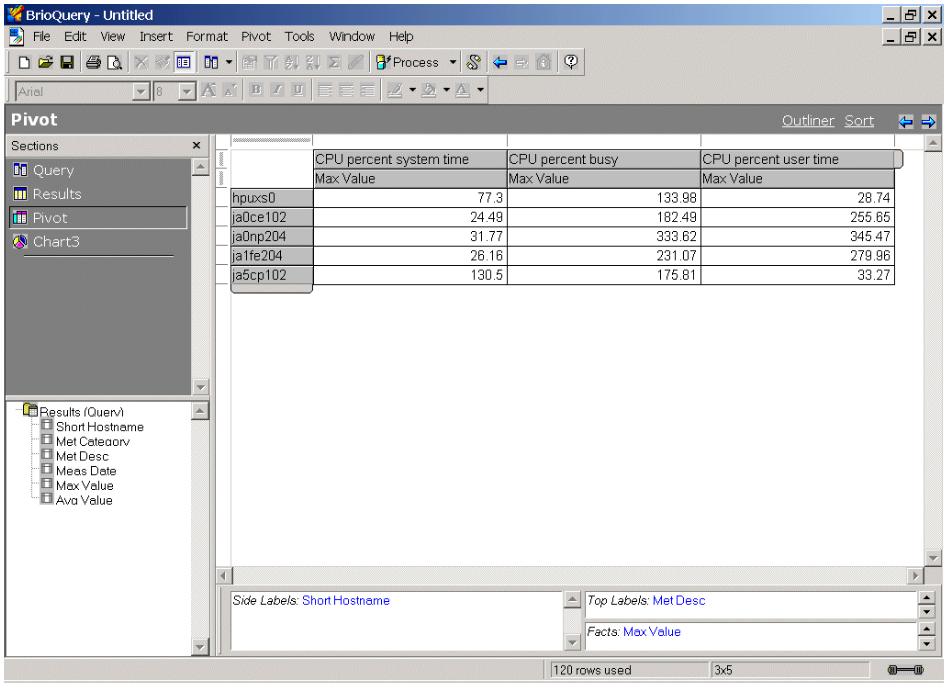
☐ Include Nulls

☐ Not = Equal

CPU percent busy
CPU percent system time
CPU percent user time
Disk space available in MB on specified
Number of processes
Number of Processes
Page ins per second
Page outs per second
Page scans per second
Peak disk throughput in KB/sec for busi
Peak disk transfers per sec for fastest d
Peak throughput in Bytes/sec for busies

Figure 7-9 Setting a limit window

8. We then rerun the query process and create a new pivot chart using the method already described in previous steps (Figure 7-10 on page 215).



	CPU percent system time	CPU percent busy	CPU percent user time
	Max Value	Max Value	Max Value
hpuxs0	77.3	133.98	28.74
ja0ce102	24.49	182.49	255.65
ja0np204	31.77	333.62	345.47
ja1fe204	26.16	231.07	279.96
ja5cp102	130.5	175.81	33.27

Side Labels: Short Hostname
Top Labels: Met Desc
Facts: Max Value
120 rows used
3x5

Figure 7-10 Crosstab report window

9. From this we can create a bar graph by clicking **Insert -> Chart This Pivot**. This chart shows the maximum daily value for each CPU description for all hosts over a period of seven days (Figure 7-11 on page 216).

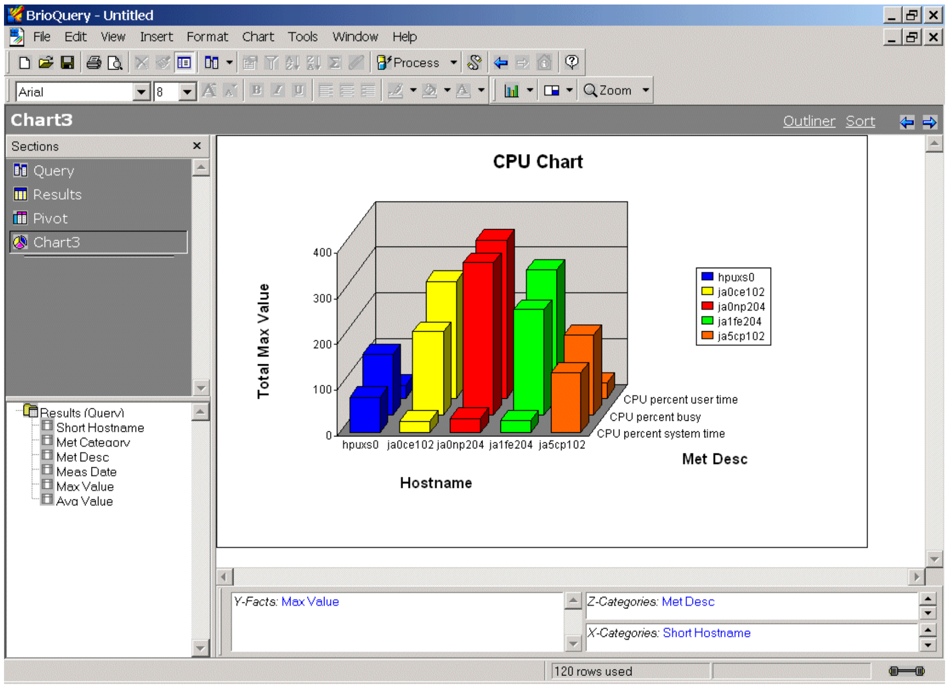


Figure 7-11 Bar graph window

10. For more detailed information by day select a host name then right click and select **Drill Anywhere -> Meas Date**. The chart will now show the maximum value for each CPU description for each day.

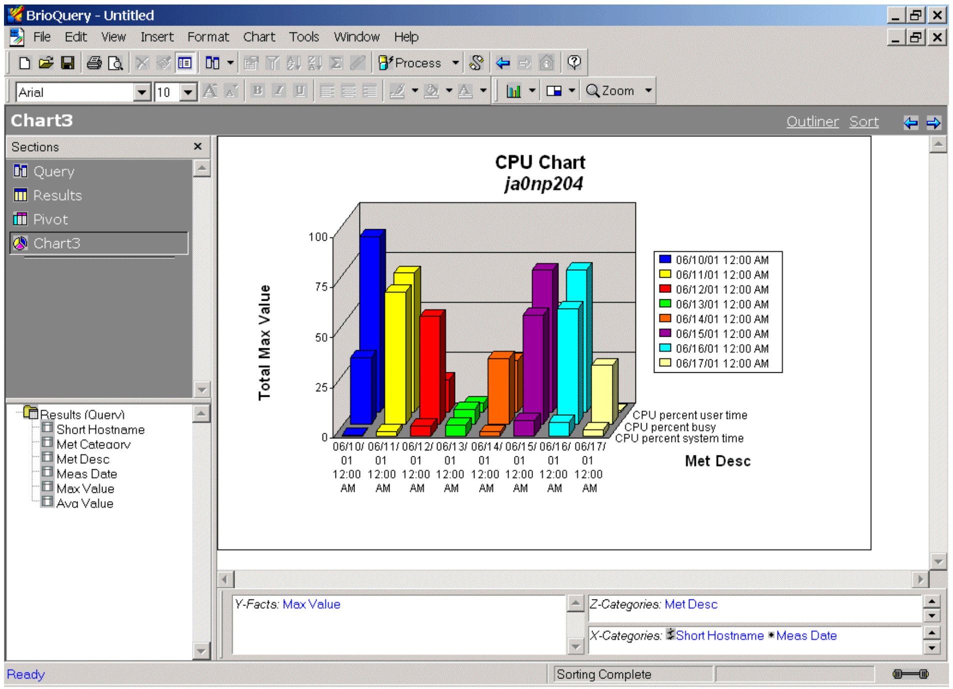


Figure 7-12 Bar graph drill-down window

We have shown in this section a very basic setup and use of Brio. For a more advanced use of Brio, refer to the user's guide that comes with the product. These reports can be exported to a Web server for viewing via a Web browser. In a more advanced installation of Brio, you can even configure the Brio server components to run the query automatically and generate reports, once a query setup is registered with the Brio server. These reports can then be viewed via the Brio Insight or Brio Quickview components.

7.2.4 Brio sample reports

In this section we will show sample reports we have created from Tivoli Distributed Monitoring data.

- The report in Figure 7-13 on page 218 shows what hosts have been over 90 percent utilization in the last day.

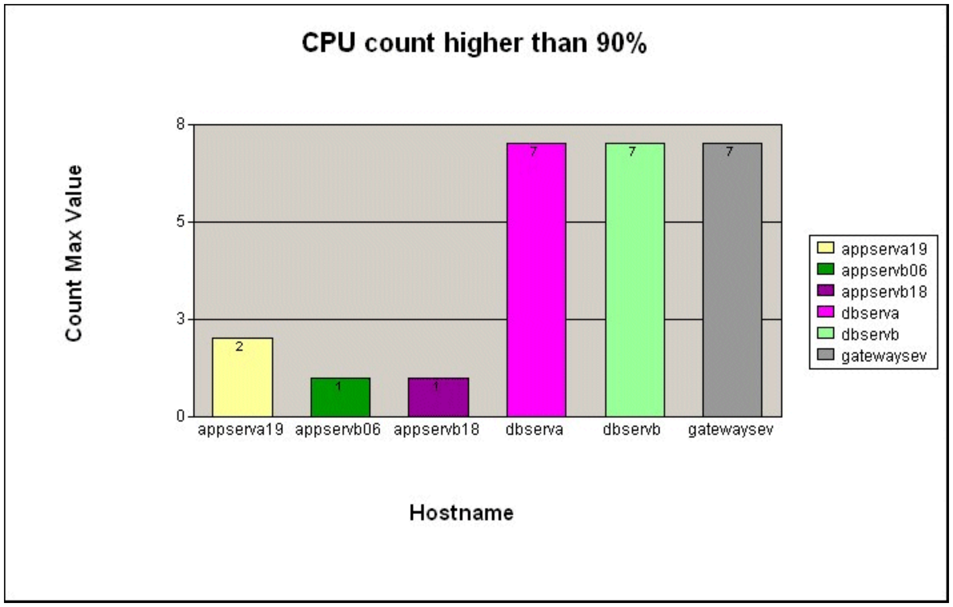


Figure 7-13 CPU over 90 percent window

- The report in Figure 7-14 on page 219 is a drill-down of the previous report to show what hours the host name appserva19 CPU exceeded 90 percent.

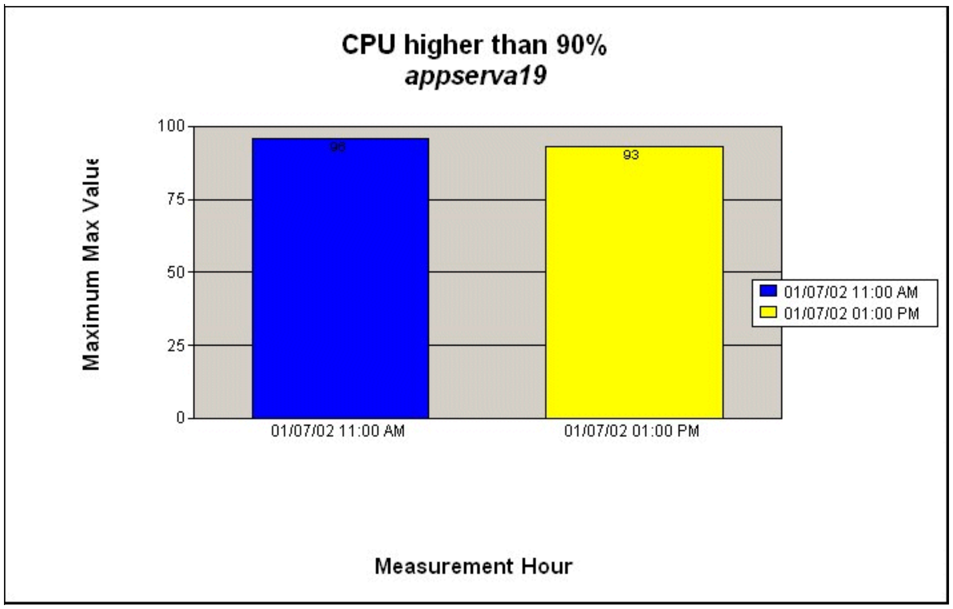
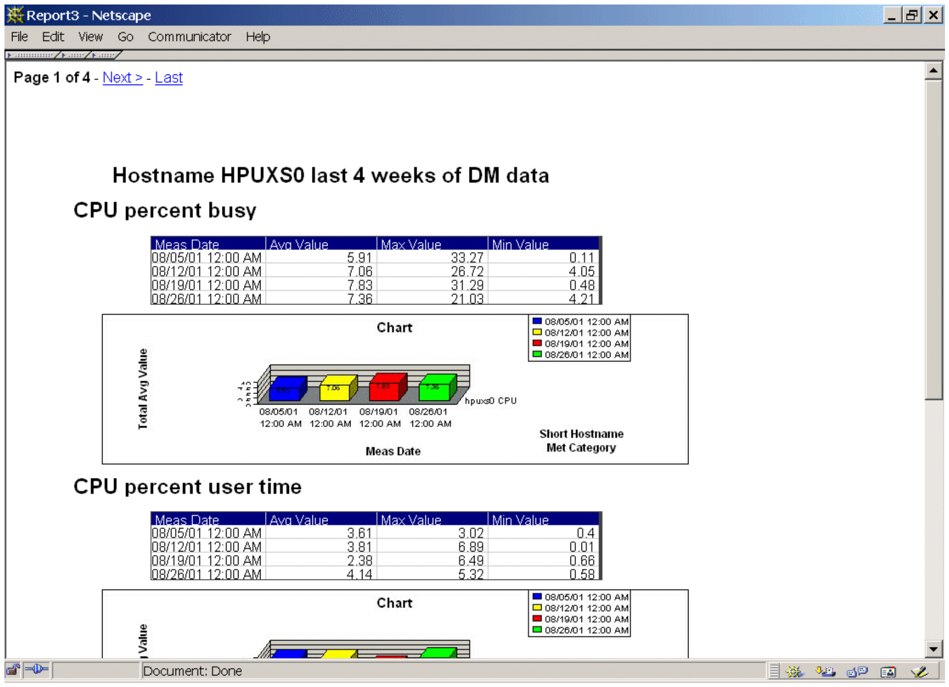


Figure 7-14 Drill-down of a host appserva19 window

- The following reports (Figure 7-15 on page 220 and Figure 7-16 on page 221) show a single machine and all of the metric measurements data for CPU, network, disk, and memory for a period of the last four weeks.



Page 1 of 4 - Next > - Last

Hostname HPUXS0 last 4 weeks of DM data

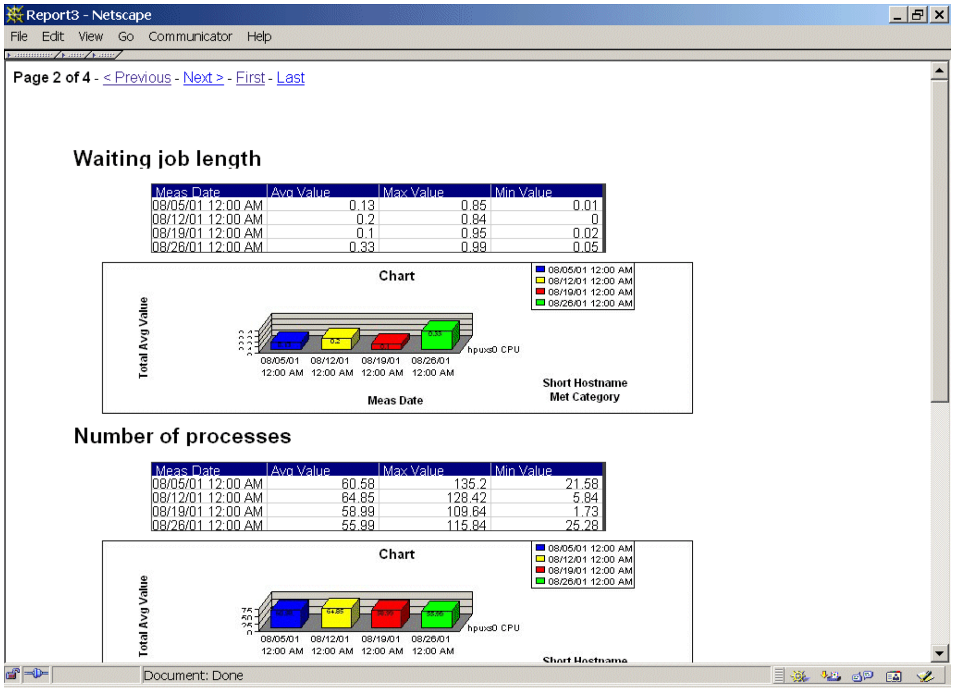
CPU percent busy

Meas Date	Avg Value	Max Value	Min Value
08/05/01 12:00 AM	5.91	33.27	0.11
08/12/01 12:00 AM	7.06	26.72	4.05
08/19/01 12:00 AM	7.83	31.29	0.48
08/26/01 12:00 AM	7.36	21.03	4.21

CPU percent user time

Meas Date	Avg Value	Max Value	Min Value
08/05/01 12:00 AM	3.61	3.02	0.4
08/12/01 12:00 AM	3.81	6.89	0.01
08/19/01 12:00 AM	2.38	6.49	0.66
08/26/01 12:00 AM	4.14	5.32	0.58

Figure 7-15 Page one of metric's for host HPUXS0 window



Report3 - Netscape

Page 2 of 4 - < Previous - Next > - First - Last

Waiting job length

Meas Date	Avg Value	Max Value	Min Value
08/05/01 12:00 AM	0.13	0.85	0.01
08/12/01 12:00 AM	0.2	0.84	0
08/19/01 12:00 AM	0.1	0.95	0.02
08/26/01 12:00 AM	0.33	0.99	0.05

Number of processes

Meas Date	Avg Value	Max Value	Min Value
08/05/01 12:00 AM	60.58	135.2	21.58
08/12/01 12:00 AM	64.85	128.42	5.84
08/19/01 12:00 AM	58.99	109.64	1.73
08/26/01 12:00 AM	55.99	115.84	25.28

Document: Done

Figure 7-16 Page two of metric's for host HPUXS0 window

7.3 Business Objects

This section will give you a overview of the components of Business Objects, a basic example of integration with Business Objects with Tivoli Enterprise Data Warehouse, and sample reports.

7.3.1 Business Objects overview

Business Objects is an integrated query, reporting, and analysis solution that allows you to access the data in your corporate databases directly from your desktop and present and analyze this information in a Business Objects document. For more information go to the Business Objects Web site at `http://www.businessobjects.com`.

7.3.2 Business Objects components

The Business Objects product line can be broken down into:

- Enterprise server products

  There are four main enterprise server products:

  – *Webintelligence* provides users with a light-weight, easy-to-use interface called the Web Panel to create and modify documents.

  – *Broadcast Agent* allows BUSINESSOBJECTS and WEBINTELLIGENCE users to automatically process and publish their Business Objects documents via the repository, an intranet, extranet, or the World Wide Web.

  – *Zero Admin Business Objects* is a new deployment of BUSINESSOBJECTS 5.1, installed on client machines from WEBINTELLIGENCE via a Web browser.

  – *Webintelligence SDK* is an extension of WEBINTELLIGENCE, which enables you to customize the look, behavior, and workflow of a WEBINTELLIGENCE deployment to match the needs of a wide user audience.

- Desktop products

  Business Objects desktop products include:

  – *Business Objects* is an integrated query, reporting, and analysis solution for business professionals that allows you to access the data in your corporate databases directly from your desktop and present and analyze this information in a BUSINESSOBJECTS document.

  – *Business Query* for Excel is an add-in tool that provides Microsoft Excel with fully functional database access. With BUSINESSQUERY, you can access your corporate databases from Excel using familiar business terms.

  – *Business Miner* works hand-in-hand with BUSINESSOBJECTS. Once you have used BUSINESSOBJECTS to access the information you need from your databases or data marts, you can use BUSINESSMINER to find the trends and patterns hidden in the data.

  – *Designer* allows administrators to create, manage, and distribute universes for particular groups of BUSINESSOBJECTS and WEBINTELLIGENCE users.

  – *Supervisor* allows you, as supervisor, to set up and maintain a secure environment for the overall Business Objects system.

  – *Developer Suite* provides application developers with tools to customize BUSINESSOBJECTS and DESIGNER.

- *Set Analyzer* allows nontechnical users to query large data stores without the lag time associated with standard relational online analytical processing (ROLAP) tools.
- *The Business Objects Services Administrator* gives you an overview of your distributed architecture system.
- *The Broadcast Agent Console* allows administrators to track and modify the scheduled automatic processing of documents through BROADCAST AGENT.

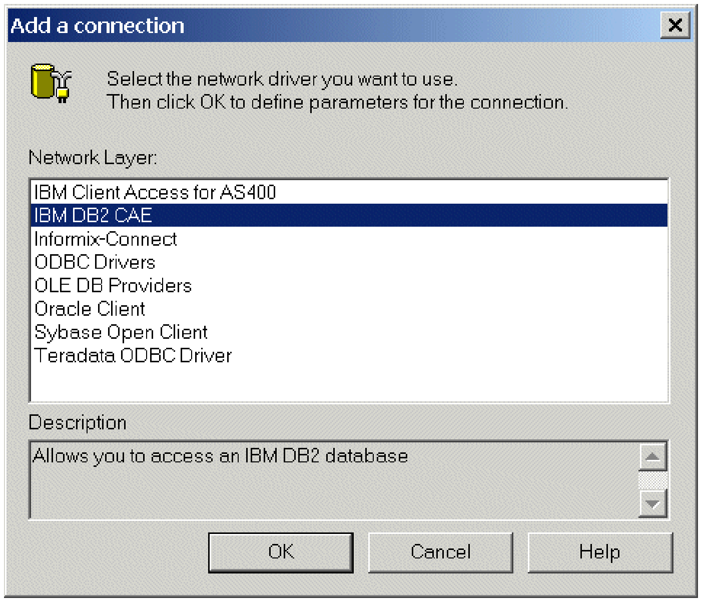
7.3.3 Business Objects integration

In this section we will use the Designer Version 5.1 and Business Object Version 5.1 components to show a simple integration scenario with Tivoli Enterprise Data Warehouse. We will cover setting up the integration and running reports.

Setting up the integration

We installed Designer Version 5.1 and Business Object Version 5.1 onto a Windows 2000 system using the installation instructions supplied with the product; also the DB2 Client software was installed and a ODBC connection to the TWH_MART database was configured. Once this is complete we did the following:

1. Set up a universe before we can use Business Objects. To do this click **Start -> Programs -> Business Objects 5.1 -> Designer** and a Quick Design Wizard window will appear
2. Click **Begin** and the Define the Universe parameters window will appear. Enter a universe name and click the **New** button and the Add a connection window appears (Figure 7-17 on page 224).


Add a connection

Select the network driver you want to use.
Then click OK to define parameters for the connection.

Network Layer:

IBM Client Access for AS400
IBM DB2 CAE
Informix-Connect
ODBC Drivers
OLE DB Providers
Oracle Client
Sybase Open Client
Teradata ODBC Driver

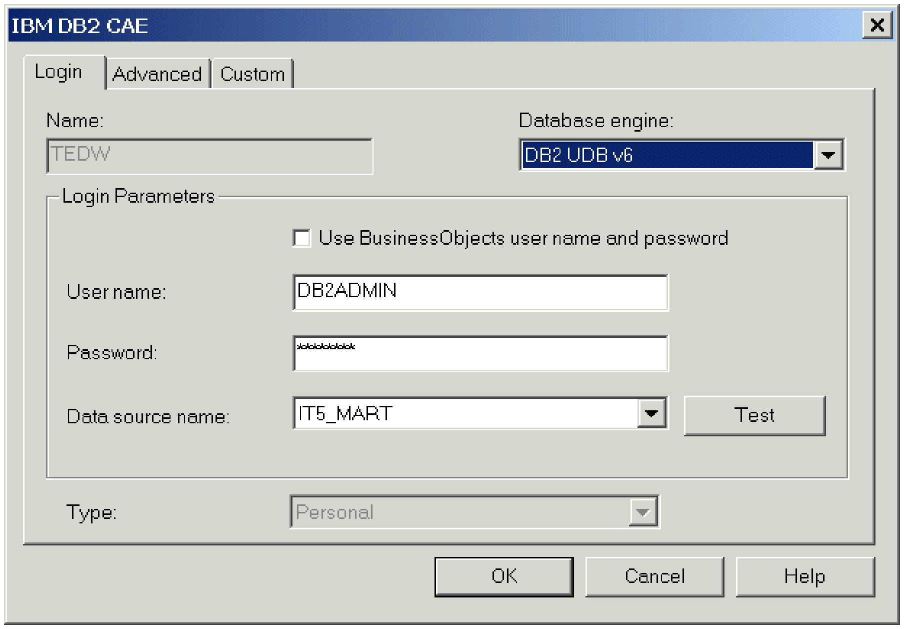
Description

Allows you to access an IBM DB2 database

Figure 7-17   Add a connection window

3. Select **IBM DB2 CAE** and click **OK**. The IBM DB2 CAE window will appear.
4. Enter a connection name and select **DB2 UDB V6** from the database engine drop-down menu.
5. Enter a DB2 user name and password and select the ODBC name of the TWH_Mart database connection from the data source name drop-down menu (Figure 7-18 on page 225).

Note: If the ODBC for the TWH_MART database is not in the drop-down list, that means that it has not been configured. You need to configure it first.


IBM DB2 CAE

Login | Advanced | Custom

Name: TEDW

Database engine: DB2 UDB v6

Login Parameters

☐ Use BusinessObjects user name and password

User name: DB2ADMIN

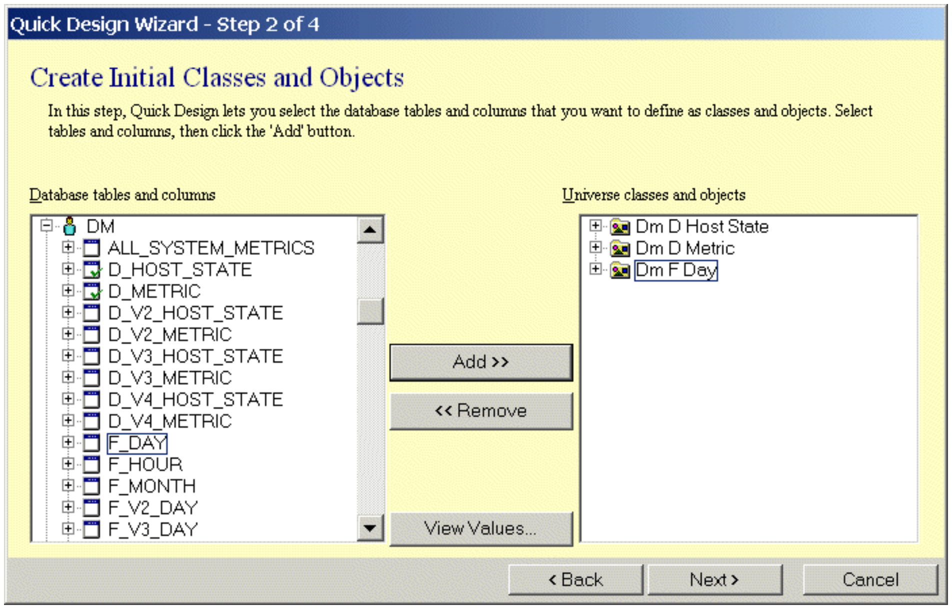
Password: ********

Data source name: IT5_MART

Type: Personal

Figure 7-18 IBM DB2 CAE window

6. Click **OK**. The Quick Design Wizard step 1 window will appear.
7. Click **Next** and the Quick Design Wizard step 2 will appear. **Add** the tables to run a query (Figure 7-19 on page 226).


Quick Design Wizard - Step 2 of 4

Create Initial Classes and Objects

In this step, Quick Design lets you select the database tables and columns that you want to define as classes and objects. Select tables and columns, then click the 'Add' button.

Database tables and columns

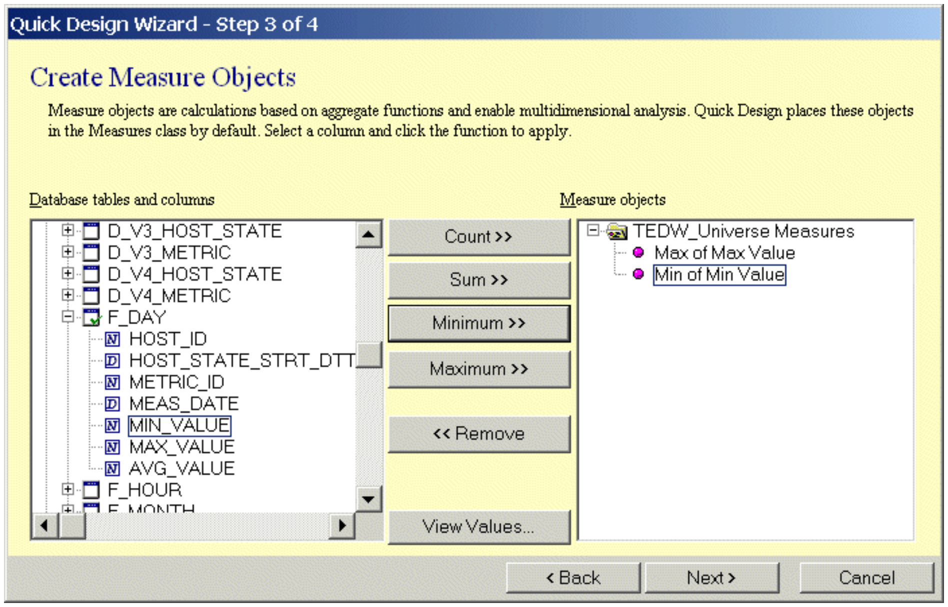
- DM
  - ALL_SYSTEM_METRICS
  - D_HOST_STATE
  - D_METRIC
  - D_V2_HOST_STATE
  - D_V2_METRIC
  - D_V3_HOST_STATE
  - D_V3_METRIC
  - D_V4_HOST_STATE
  - D_V4_METRIC
  - F_DAY
  - F_HOUR
  - F_MONTH
  - F_V2_DAY
  - F_V3_DAY

Universe classes and objects

- Dm D Host State
- Dm D Metric
- Dm F Day

Figure 7-19 Add tables to query window

8. Click **Next** and the Quick Design Wizard step 3 will appear. **Add** the measurement objects from the fact table (Figure 7-20 on page 227).


Quick Design Wizard - Step 3 of 4

Create Measure Objects

Measure objects are calculations based on aggregate functions and enable multidimensional analysis. Quick Design places these objects in the Measures class by default. Select a column and click the function to apply.

Database tables and columns

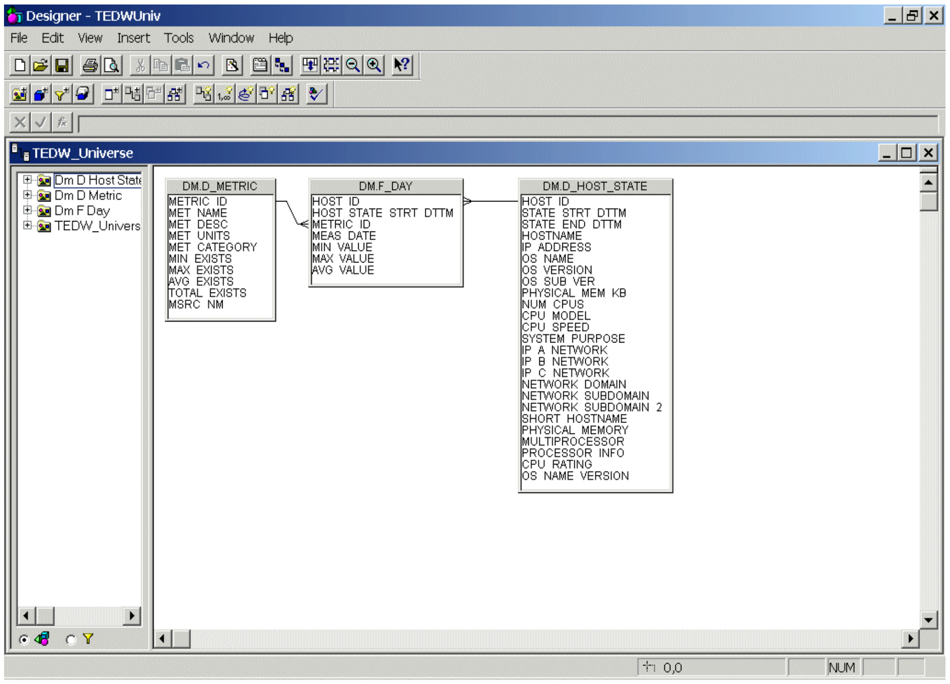
- D_V3_HOST_STATE
- D_V3_METRIC
- D_V4_HOST_STATE
- D_V4_METRIC
- F_DAY
  - HOST_ID
  - HOST_STATE_STRT_DTT
  - METRIC_ID
  - MEAS_DATE
  - MIN_VALUE
  - MAX_VALUE
  - AVG_VALUE
- F_HOUR
- F_MONTH

Measure objects

- TEDW_Universe Measures
  - Max of Max Value
  - Min of Min Value

Figure 7-20 Add the measurements from the fact table window

9. Click **Next** and the Quick Design Wizard step 4 will appear. Click **Finish**.
10. The universe window will appear showing the configuration we have set up. We need to create the table joins. Do this by clicking **Tools -> Detect Joins** or set the joins up manually (Figure 7-21 on page 228).



DM.D_METRIC
METRIC ID
MET NAME
MET DESC
MET UNITS
MET CATEGORY
MIN EXISTS
MAX EXISTS
AVG EXISTS
TOTAL EXISTS
MSRC NM

DM.F_DAY
HOST ID
HOST STATE STRT DTTM
METRIC ID
MEAS DATE
MIN VALUE
MAX VALUE
AVG VALUE

DM.D_HOST_STATE
HOST ID
STATE STRT DTTM
STATE END DTTM
HOSTNAME
IP ADDRESS
OS NAME
OS VERSION
OS SUB VER
PHYSICAL MEM KB
NUM CPUS
CPU MODEL
CPU SPEED
SYSTEM PURPOSE
IP A NETWORK
IP B NETWORK
IP C NETWORK
NETWORK DOMAIN
NETWORK SUBDOMAIN
NETWORK SUBDOMAIN 2
SHORT HOSTNAME
PHYSICAL MEMORY
MULTIPROCESSOR
PROCESSOR INFO
CPU RATING
OS NAME VERSION

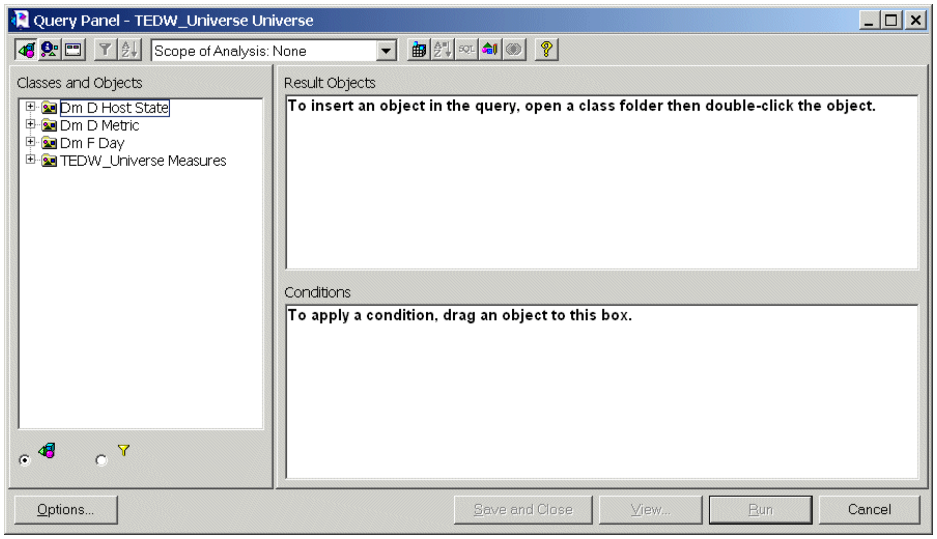
Figure 7-21 Setting up table joins window

The set up is now complete.

Creating reports with Business Objects

We now need to design a report using the above created universe. To do this do the following:

1. Click **Start -> Programs -> BusinessObjects 5.1 -> BusinessObjects**. The New Report Wizard appears.
2. Check the Generate a standard report and Specify how to access data buttons.
3. Click **Begin**, check the Universe button and click **Next**. Then select the Universe name we just created and click **Finish**. The Query Panel appears (Figure 7-22 on page 229).



To insert an object in the query, open a class folder then double-click the object.

To apply a condition, drag an object to this box.

Figure 7-22 Query Panel window

4. Now expand the **Classes and Objects** trees and drag and drop all the objects that will be part of a query to the **Result Objects** pane (Figure 7-23 on page 230).

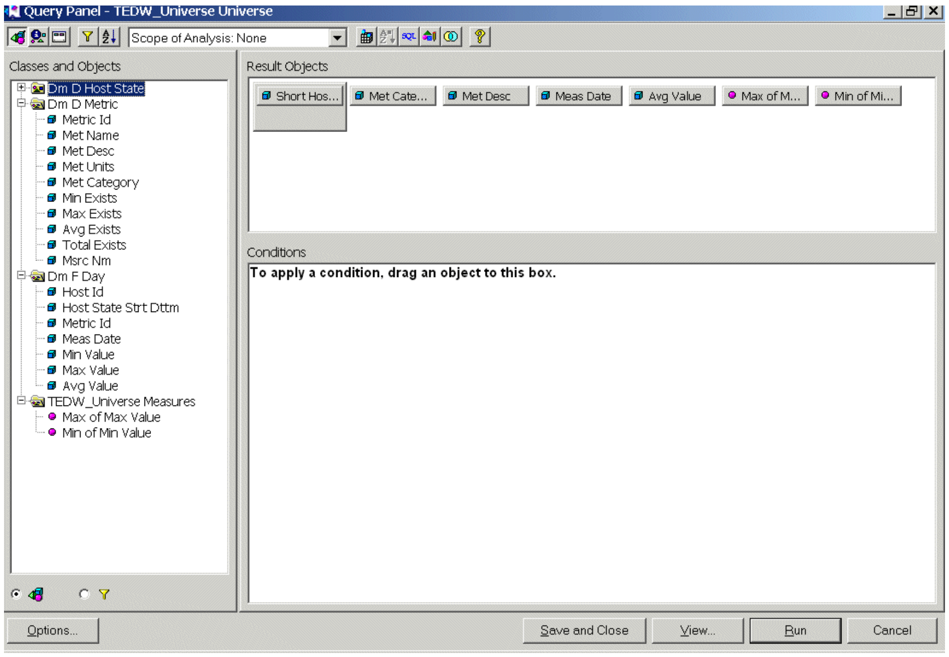


Figure 7-23 Results Objects window

5. Click **Run**. This will return the results of the query. This could return too much data to create a graph or the report may not be in a suitable format, so click **Analysis -> Slice and Dice**. The Slice and Dice Panel window appears.
6. To create a report with a different format, drag and drop a object such as host name above the dotted line or remove any unwanted objects (Figure 7-24 on page 231).

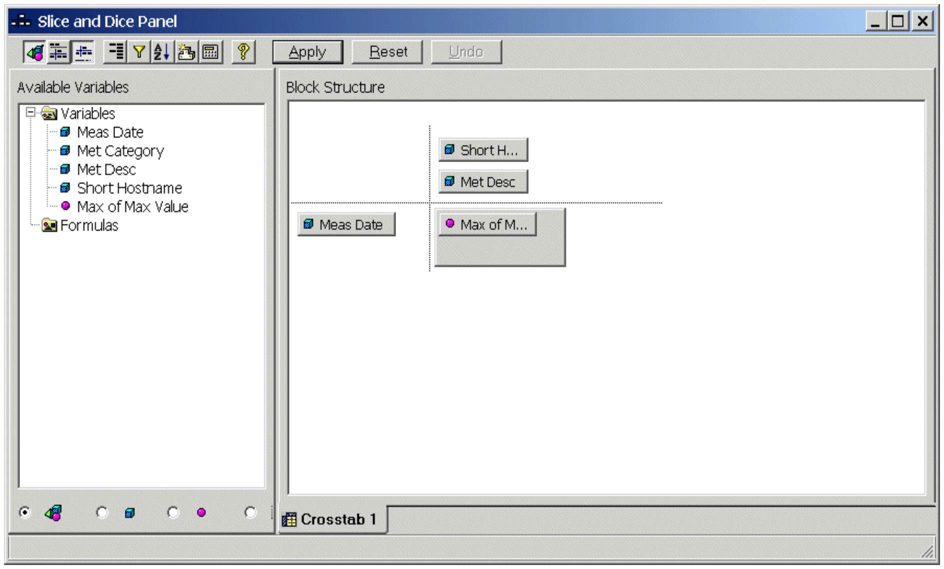
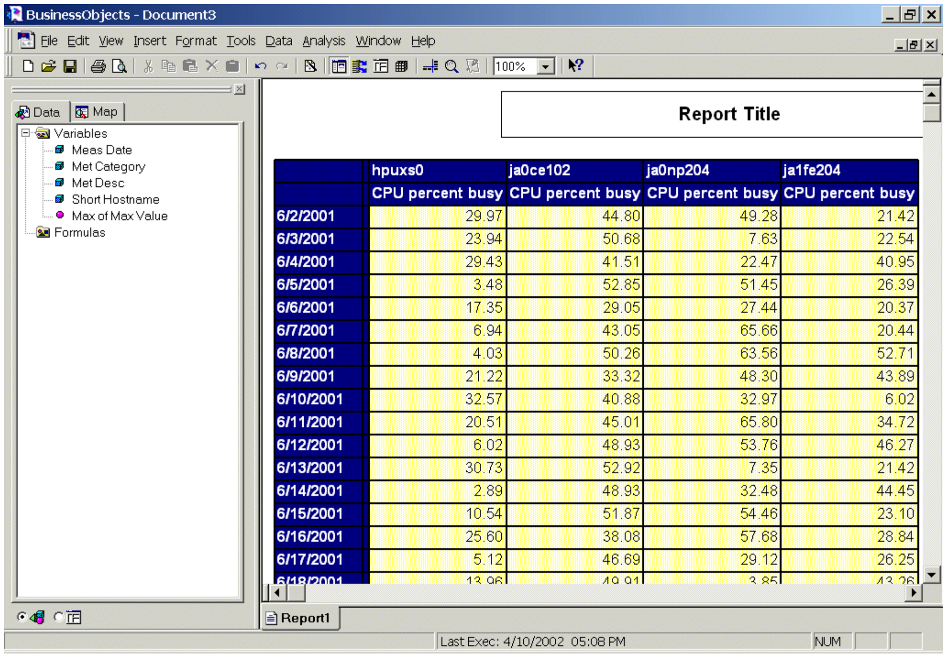


Figure 7-24 Slice and Dice Panel window

7. Click **Apply**. This will produce a report like the one in Figure 7-25 on page 232.



Report Title

	hpuxs0	ja0ce102	ja0np204	ja1fe204
	CPU percent busy	CPU percent busy	CPU percent busy	CPU percent busy
6/2/2001	29.97	44.80	49.28	21.42
6/3/2001	23.94	50.68	7.63	22.54
6/4/2001	29.43	41.51	22.47	40.95
6/5/2001	3.48	52.85	51.45	26.39
6/6/2001	17.35	29.05	27.44	20.37
6/7/2001	6.94	43.05	65.66	20.44
6/8/2001	4.03	50.26	63.56	52.71
6/9/2001	21.22	33.32	48.30	43.89
6/10/2001	32.57	40.88	32.97	6.02
6/11/2001	20.51	45.01	65.80	34.72
6/12/2001	6.02	48.93	53.76	46.27
6/13/2001	30.73	52.92	7.35	21.42
6/14/2001	2.89	48.93	32.48	44.45
6/15/2001	10.54	51.87	54.46	23.10
6/16/2001	25.60	38.08	57.68	28.84
6/17/2001	5.12	46.69	29.12	26.25
6/18/2001	13.96	49.91	3.85	43.26

Last Exec: 4/10/2002 05:08 PM

Figure 7-25 Crosstab report window

8. We now need to filter the data to show only seven days worth of data. Do this by clicking **Analysis -> Slice and Dice**. The Slice and Dice Panel window will appear.
9. Select an object such as **Meas Date** and click the filter button. The Apply a Filter window will appear. Select the values to show and click **OK** (Figure 7-26 on page 233).

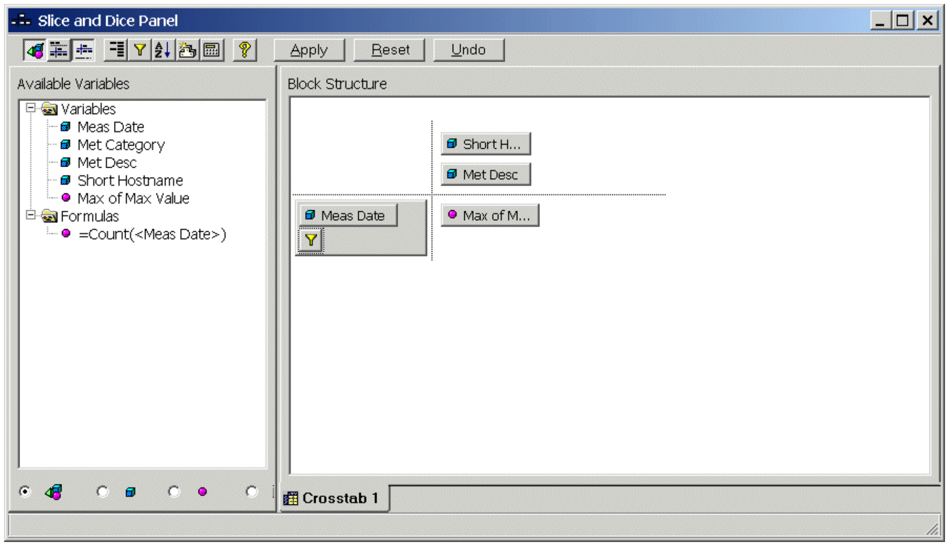


Figure 7-26 Filter applied window

10. Click **Apply** to finish and close the Slice and Dice Panel.
11. From this report we can create a graph. Do this by right clicking the report and selecting **Turn to Chart**. The Chart Auto Format window will appear.
12. Select a report type such as **Line** and click **OK**. This will display the graph (Figure 7-27 on page 234).

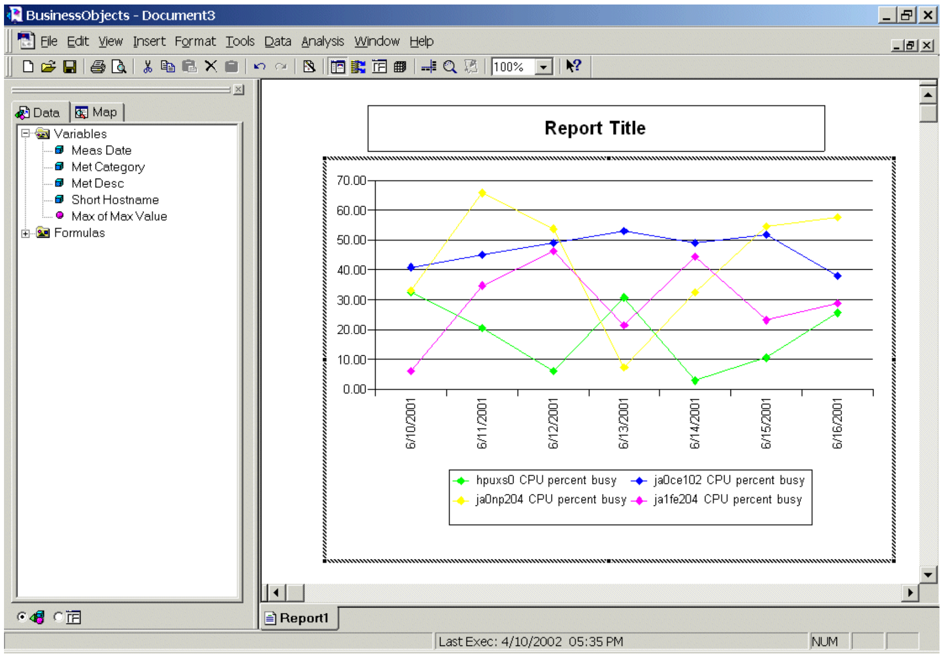


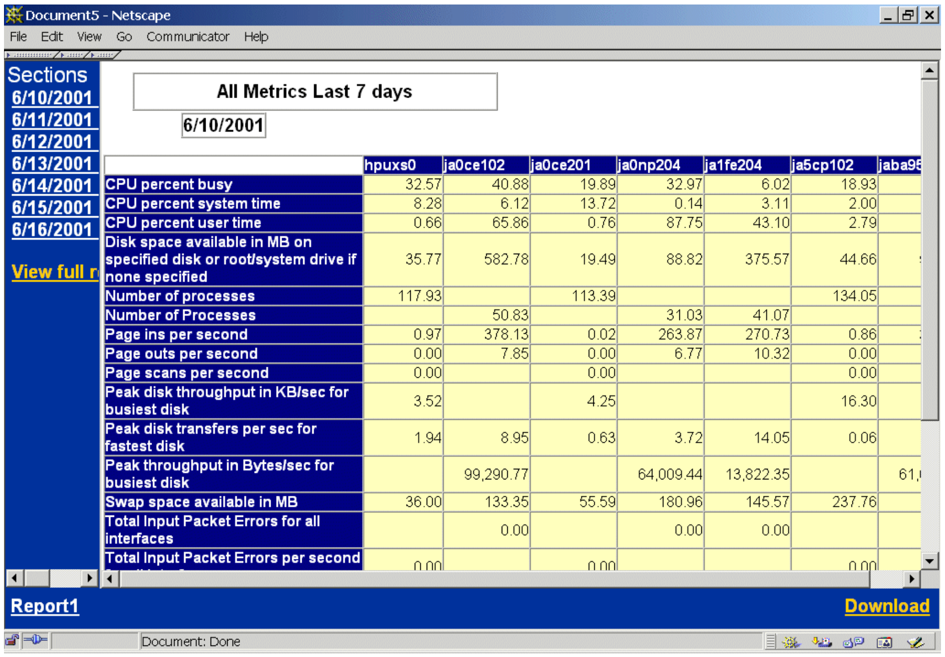
Figure 7-27 Line graph window

We have shown in this section a very basic set up and use of Business Objects. For a more advanced use of Business Objects refer to the user's guide that comes with the product. These reports can be exported to a Web server for viewing via a Web browser or, in a more advanced installation of Business Objects, the server components could be set up to provide automatic generation of reports, which can then be viewed via a Web browser or a Business Objects client.

7.3.4 Business Objects sample reports

In this section we will show sample reports we have created from Tivoli Distributed Monitoring data.

The following report (Figure 7-28 on page 235) shows all hosts and all of the measurement data (CPU, network, disk, and memory) for a period of the last seven days. You can navigate to what day you want by clicking the date in the left-hand frame.



All Metrics Last 7 days

6/10/2001

	hpuxs0	ja0ce102	ja0ce201	ja0np204	ja1fe204	ja5cp102	jaba95
CPU percent busy	32.57	40.88	19.89	32.97	6.02	18.93	
CPU percent system time	8.28	6.12	13.72	0.14	3.11	2.00	
CPU percent user time	0.66	65.86	0.76	87.75	43.10	2.79	
Disk space available in MB on specified disk or root/system drive if none specified	35.77	582.78	19.49	88.82	375.57	44.66	
Number of processes	117.93		113.39			134.05	
Number of Processes		50.83		31.03	41.07		
Page ins per second	0.97	378.13	0.02	263.87	270.73	0.86	
Page outs per second	0.00	7.85	0.00	6.77	10.32	0.00	
Page scans per second	0.00		0.00			0.00	
Peak disk throughput in KB/sec for busiest disk	3.52		4.25			16.30	
Peak disk transfers per sec for fastest disk	1.94	8.95	0.63	3.72	14.05	0.06	
Peak throughput in Bytes/sec for busiest disk		99,290.77		64,009.44	13,822.35		61,
Swap space available in MB	36.00	133.35	55.59	180.96	145.57	237.76	
Total Input Packet Errors for all interfaces		0.00		0.00	0.00		
Total Input Packet Errors per second	0.00		0.00			0.00	

Figure 7-28 Crosstab report window

The following reports (Figure 7-29 on page 236 and Figure 7-30 on page 237) show all the average daily disk measurements for the last month for selected hosts. You can click the metric measurement you want to view in the left-hand frame.

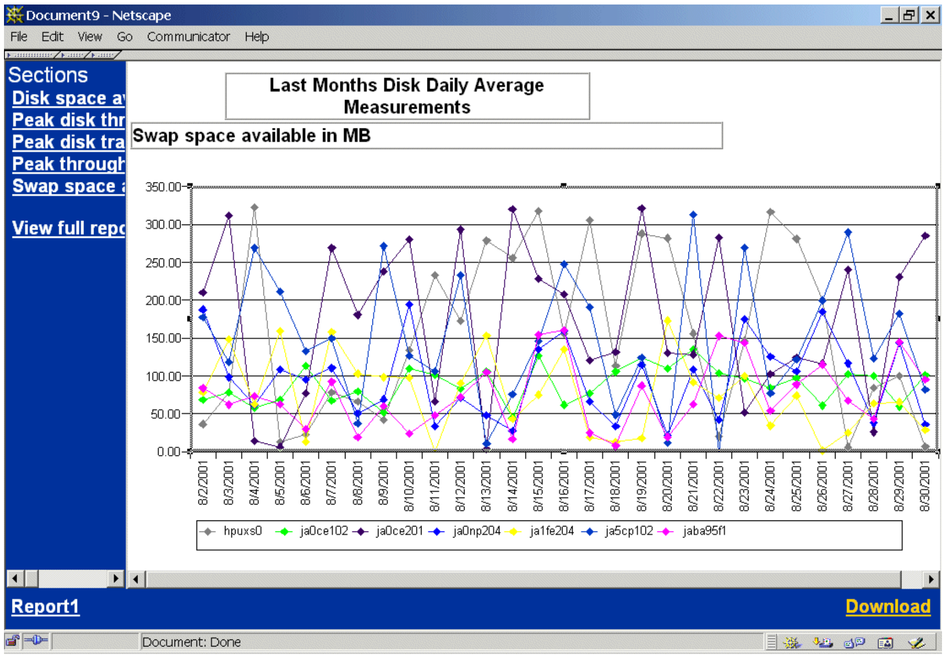


Figure 7-29 Line graph report window

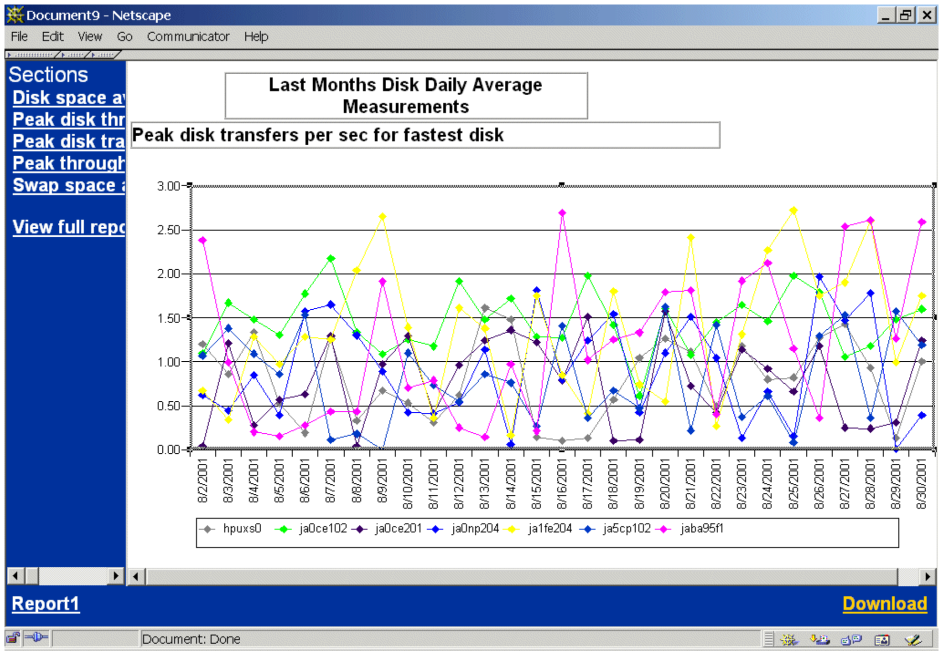


Figure 7-30 Line graph report window

7.4 Cognos

In this section we will give you an overview of the components of Cognos, and a basic example of integration with Cognos with Tivoli Enterprise Data Warehouse and sample reports. The target audience for this section is businesses that have deployed TDS and want to continue using Cognos as an OLAP software tool.

7.4.1 Cognos overview

Cognos PowerPlay enables you to create applications that read flat or two-dimensional data and manipulate it into a multidimensional presentation called a PowerCube. Each cube is in essence a data mart, which provides the information your users need to analyze their operations.

7.4.2 Cognos components

The Cognos PowerPlay architecture consists of the following client and server components:

- *PowerPlay for Windows* is used for standard report authoring on summarized data, for advanced exploration, and for ad hoc trend and exception reporting.
- *PowerPlay for Excel* is used for authoring reports using the built-in graphing functions and cell-based calculation capabilities of Microsoft Excel. Compatible with Microsoft Office 95 and Microsoft Office 97, this Excel add-in populates worksheets with the live data in a cube.
- *PowerPlay Web* is used for casual exploration and ad hoc trend and exception reporting. Online help is available, but no special training is needed to use this HTML-based client.
- *PowerPlay Enterprise Server* provides access to standard cubes, cubes stored in relational databases (either locally or distributed to other file servers), and third-party cubes stored on Essbase, Oracle Express, Microsoft, or IBM OLAP servers.
- *PowerPlay Personal Server* is used for saving and transporting cubes or sub-cubes on the personal computers of mobile users, and for accessing information stored in third-party cubes or in relational databases, locally or on a LAN.
- *PowerPlay Transformation Server (Windows Edition)* is used to provide full PowerPlay client functionality and the database storage capabilities formerly associated with the Administrator Database edition of Transformer 6.0.

7.4.3 Cognos integration with Tivoli Enterprise Data Warehouse

In this section we will use the Cognos PowerPlay Version 6.5 and Cognos PowerPlay Transformer Version 6.5 to show a simple integration scenario with Tivoli Enterprise Data Warehouse using Tivoli Distributed Monitoring data

Setting up the integration

We installed Cognos PowerPlay Version 6.5 and Cognos PowerPlay Transformer Version 6.5 from the Tivoli Decision Support (TDS) 2.1 CD. For installation instructions you can refer to the TDS manuals.

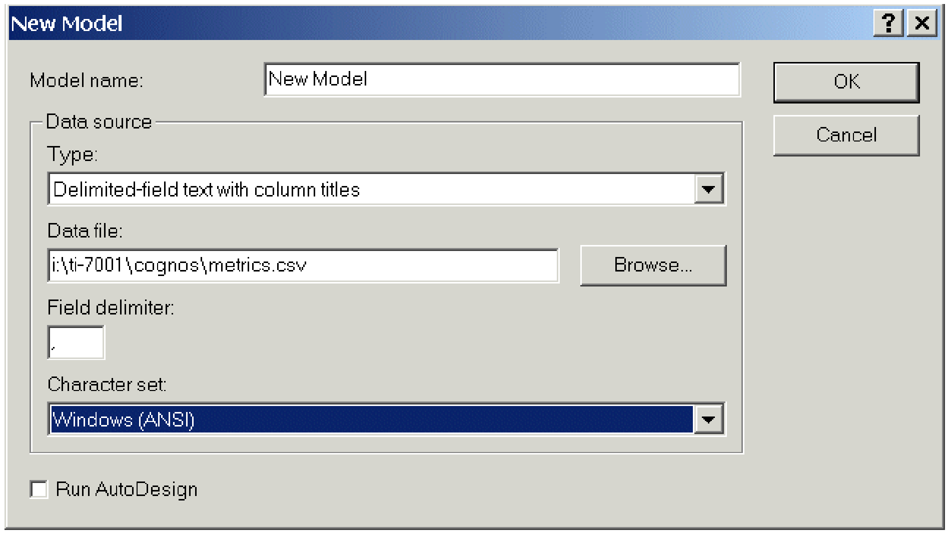
Important: The Cognos components that we used in this scenario are the ones that are shipped with Tivoli Decision Support 2.1. But if you want to use Cognos as your preferred OLAP tool to integrate with Tivoli Enterprise Data Warehouse, we recommend that you license the latest versions of Cognos products, because Cognos PowerPlay Version 6.5 and Cognos PowerPlay Transformer Version 6.5 that are shipped with Tivoli Decision Support 2.1 are not the latest versions.

Cognos uses an additional product called Impromptu for easy access to relational databases. Because Impromptu is not shipped with TDS, we opted to use the exporting function of DB2. If you decide to use Cognos as your OLAP tool for Tivoli Enterprise Data Warehouse data, it will be easier for you to use Impromptu, but you might need to get a license for it. Please contact Cognos Corporation for this.

In order to define our tables in Cognos we did the following:

1. We exported the following DM tables in our TWH_Mart database and created the corresponding CSV files for each:
   - Metrics table
   - Host table
   - Week tables
2. Next we defined these files in the PowerPlay Transformer as queries. Click **Start -> Programs -> Cognos 6.5 -> Transformer**. The PowerPlay Transformer program will start.
3. Select **File -> New** and the New Model window will appear.
4. Enter a model name, select **Delimited-field text with column titles** as the type, and select one of the exported CSV files as the data file and uncheck **Run AutoDesign** (Figure 7-31 on page 240).

   Tip: AutoDesign creates the model structure automatically, but it can only be used as a preliminary transformer model design. For most of the cases you need to modify it. The more data you have, the higher the accuracy of the model will be. You will also have a chance to run AutoDesign once you have imported all the queries.



Model name: New Model

Data source

Type: Delimited-field text with column titles

Data file: i:\ti-7001\cognos\metrics.csv

Field delimiter: ,

Character set: Windows (ANSI)

Run AutoDesign

Figure 7-31 New Model window

5. Click **OK**. The table will appear in the Queries window pane (Figure 7-32 on page 241).

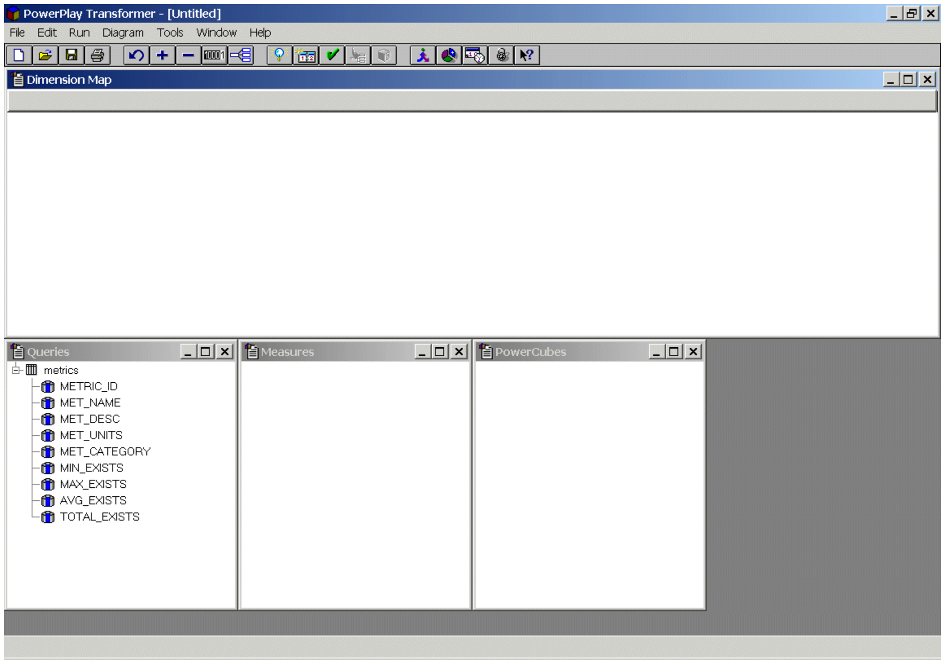


Figure 7-32 Create a query window

6. To add our other tables that we have exported to the query, right click the table already in the Queries window (Figure 7-33 on page 242).

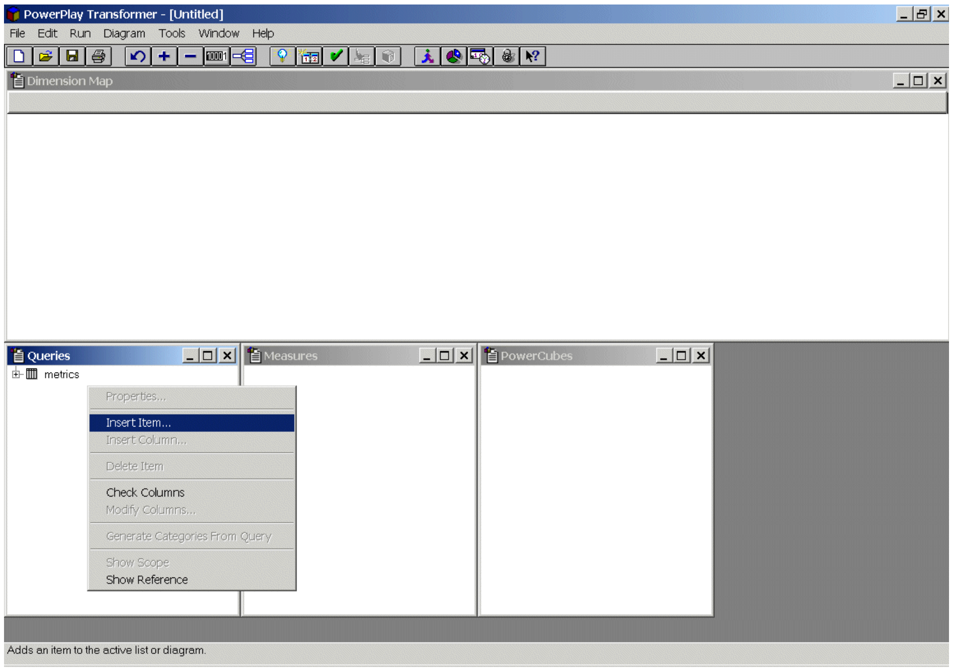


Figure 7-33 Add items window

7. Select **Insert Item**. The Query window will appear.
8. Enter a query name, enter the exported CSV file as the local data file, select **Delimited-field text with column titles** as the source type, and click **OK** (Figure 7-34 on page 243).

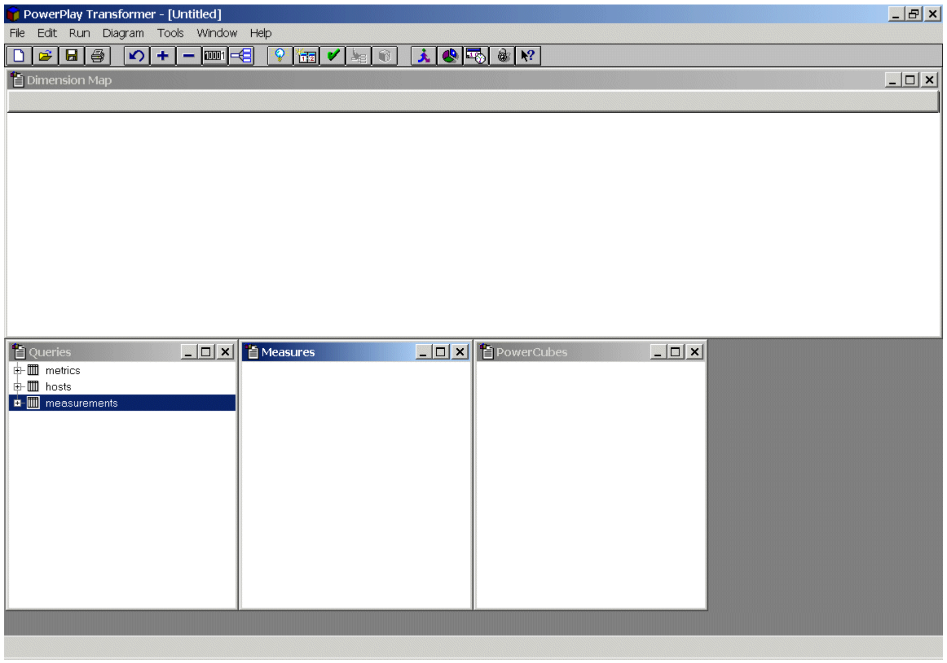

PowerPlay Transformer - [Untitled]
File Edit Run Diagram Tools Window Help
Dimension Map
Queries
metrics
hosts
measurements
Measures
PowerCubes

Figure 7-34 Create query window

Repeat this step until all tables are defined as PowerPlay Transformer, by adding them in the Queries window pane. Query names will be the same as table names by default, but you can change the name by right clicking the query and selecting **Properties**. For example, we changed the name of the query that corresponds to the weeks table from weeks to measurements.

Tip: You can change the *sequence* or *order* of the queries in the Queries list by dragging and dropping the queries with the mouse, in the Queries pane. By default Cognos PowerPlay places the queries in alphabetic order. But this might not be the optimum order. The recommended order depends on the type of the query. Cognos categorizes the query types into two:

- Structure: Queries that contain primarily text data (usually defines dimensions and levels in the model)
- Transaction: Queries that contain primarily numeric data (usually defines measures in the model)

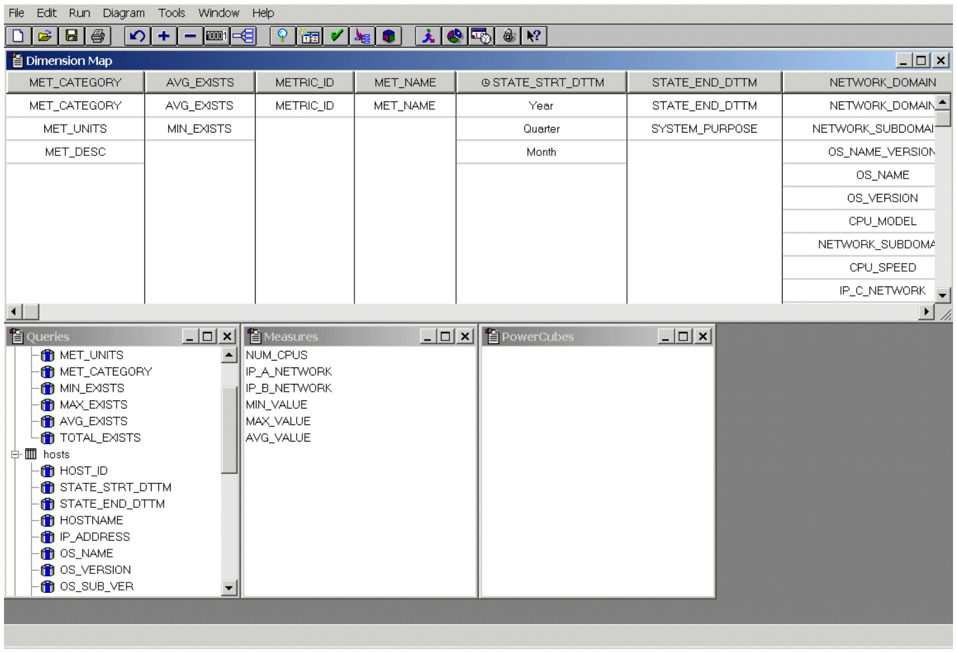
The recommended order for the queries in the Queries list is to place the structure queries before the transaction queries. This becomes important when generating the multidimensional cube.

In our example, we placed the measurements query last in the Queries list because it contains mostly numeric measurements about our data.

Creating reports with Cognos

Once we define our tables to the PowerPlay Transformer, we may start designing our model to run reports against out database.

1. First we need to create a model structure. At this point you can select **Tools -> AutoDesign** (or the *light bulb* button on the top menu bar) to ask PowerPlay Transformer to create a automatic design for you (Figure 7-35 on page 245).



MET_CATEGORY	AVG_EXISTS	METRIC_ID	MET_NAME	STATE_STRT_DTTM	STATE_END_DTTM	NETWORK_DOMAIN
MET_CATEGORY	AVG_EXISTS	METRIC_ID	MET_NAME	Year	STATE_END_DTTM	NETWORK_DOMAIN
MET_UNITS	MIN_EXISTS			Quarter	SYSTEM_PURPOSE	NETWORK_SUBDOMAI
MET_DESC				Month		OS_NAME_VERSION
						OS_NAME
						OS_VERSION
						CPU_MODEL
						NETWORK_SUBDOMA
						CPU_SPEED
						IP_C_NETWORK

Queries:

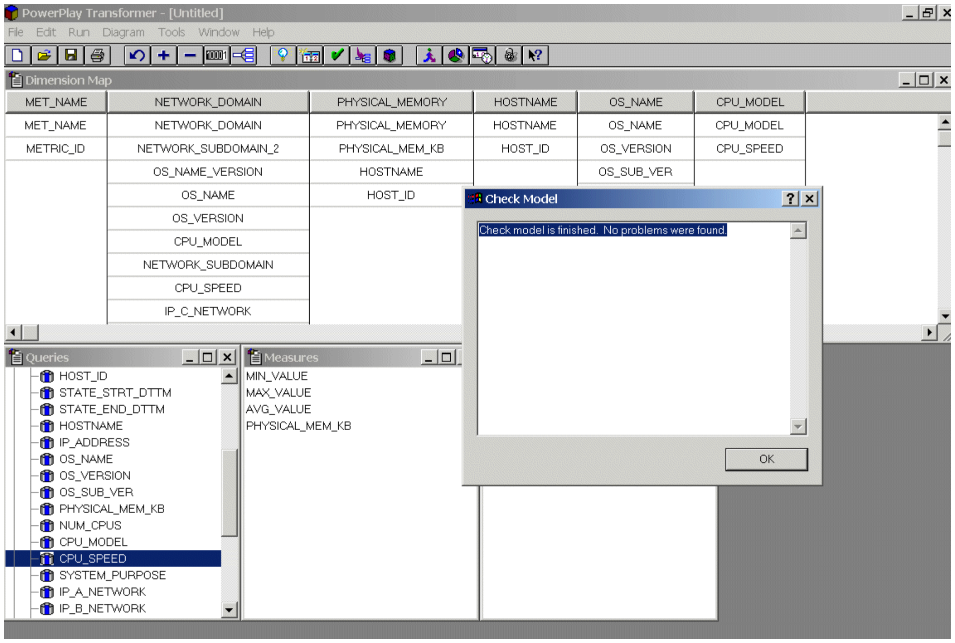
- MET_UNITS
- MET_CATEGORY
- MIN_EXISTS
- MAX_EXISTS
- AVG_EXISTS
- TOTAL_EXISTS
- hosts
  - HOST_ID
  - STATE_STRT_DTTM
  - STATE_END_DTTM
  - HOSTNAME
  - IP_ADDRESS
  - OS_NAME
  - OS_VERSION
  - OS_SUB_VER

Measures:

- NUM_CPUS
- IP_A_NETWORK
- IP_B_NETWORK
- MIN_VALUE
- MAX_VALUE
- AVG_VALUE

Figure 7-35 Create a structure window

2. AutoDesign will give you a preliminary design, but you still need to work on it to suit your needs. When designing the model structure drag and drop the columns containing text or date data from the Queries list to the Dimension Map. If you have a new dimension, drag and drop the column to a blank area on the **Dimension Map**. If you are creating additional levels in an existing dimension drag and drop the column under the existing levels. Also drag columns containing numeric data from the Queries list to the Measures list. See Figure 7-36 on page 246
3. You can also use the check model feature of Cognos by clicking **Tools -> Check Model**. This will verify the correctness of your model (Figure 7-36 on page 246).


PowerPlay Transformer - [Untitled]
File Edit Run Diagram Tools Window Help
Dimension Map
Queries
Measures
Check Model
OK

MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
METRIC_ID	NETWORK_SUBDOMAIN_2	PHYSICAL_MEM_KB	HOST_ID	OS_VERSION	CPU_SPEED
	OS_NAME_VERSION	HOSTNAME		OS_SUB_VER	
	OS_NAME	HOST_ID			
	OS_VERSION				
	CPU_MODEL				
	NETWORK_SUBDOMAIN				
	CPU_SPEED				
	IP_C_NETWORK				

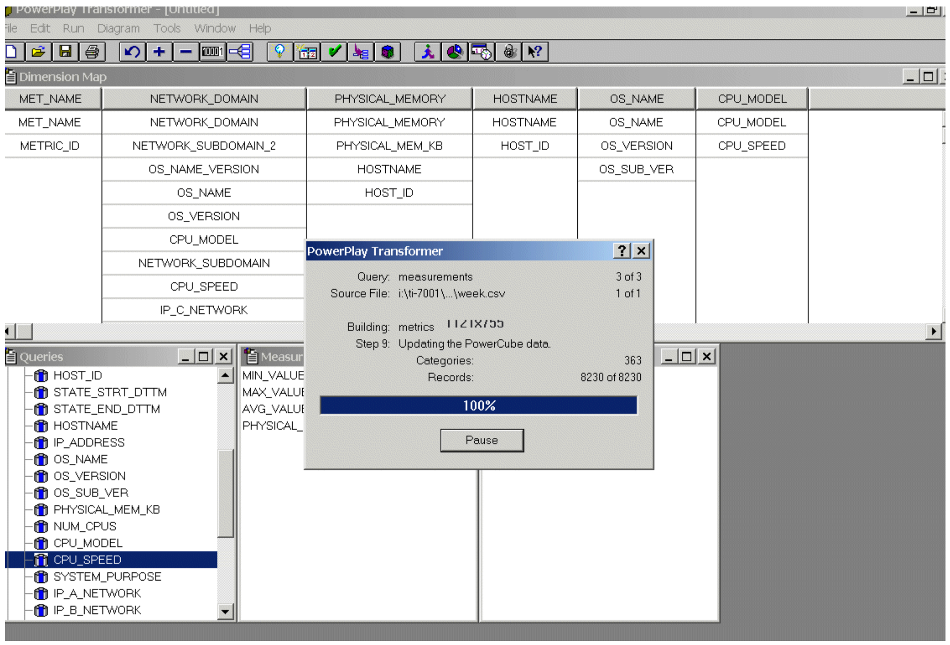
```
HOST_ID
STATE_STRT_DTTM
STATE_END_DTTM
HOSTNAME
IP_ADDRESS
OS_NAME
OS_VERSION
OS_SUB_VER
PHYSICAL_MEM_KB
NUM_CPUS
CPU_MODEL
CPU_SPEED
SYSTEM_PURPOSE
IP_A_NETWORK
IP_B_NETWORK
```

```
MIN_VALUE
MAX_VALUE
AVG_VALUE
PHYSICAL_MEM_KB
```

```
Check model is finished.  No problems were found.
```

Figure 7-36 Verifying that the model is OK

4. Now we create the cube using this model by clicking **Run -> Create PowerCubes** (Figure 7-37 on page 247).


PowerPlay Transformer - [Untitled]
File Edit Run Diagram Tools Window Help
Dimension Map

MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
METRIC_ID	NETWORK_SUBDOMAIN_2	PHYSICAL_MEM_KB	HOST_ID	OS_VERSION	CPU_SPEED
	OS_NAME_VERSION	HOSTNAME		OS_SUB_VER	
	OS_NAME	HOST_ID			
	OS_VERSION				
	CPU_MODEL				
	NETWORK_SUBDOMAIN				
	CPU_SPEED				
	IP_C_NETWORK				

PowerPlay Transformer

```
       Query:  measurements                       3 of 3
 Source File:  i:\ti-7001\...\week.csv            1 of 1

    Building:  metrics    1121x755
      Step 9:  Updating the PowerCube data.
             Categories:                          363
                Records:                  8230 of 8230

                        100%

                        Pause
```

Queries

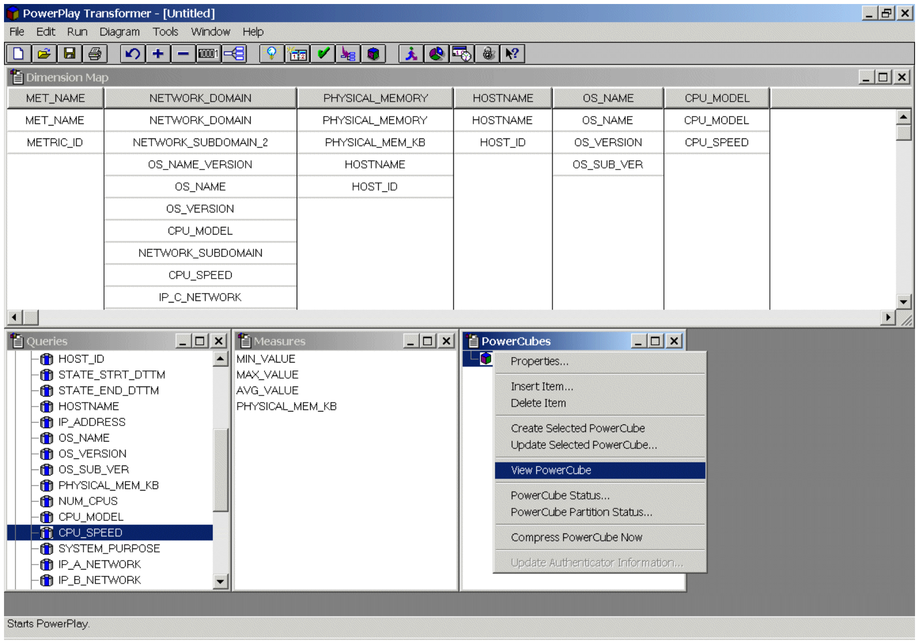
```
HOST_ID
STATE_STRT_DTTM
STATE_END_DTTM
HOSTNAME
IP_ADDRESS
OS_NAME
OS_VERSION
OS_SUB_VER
PHYSICAL_MEM_KB
NUM_CPUS
CPU_MODEL
CPU_SPEED
SYSTEM_PURPOSE
IP_A_NETWORK
IP_B_NETWORK
```

Measur

```
MIN_VALUE
MAX_VALUE
AVG_VALUE
PHYSICAL_
```

Figure 7-37 Building the cube

5. We want to view some reports from this cube by right clicking in the PowerCube pane and selecting **View PowerCube**.


PowerPlay Transformer - [Untitled]
File Edit Run Diagram Tools Window Help
Dimension Map
Queries
Measures
PowerCubes
Properties...
Insert Item...
Delete Item
Create Selected PowerCube
Update Selected PowerCube...
View PowerCube
PowerCube Status...
PowerCube Partition Status...
Compress PowerCube Now
Update Authenticator Information...
Starts PowerPlay.

MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
MET_NAME	NETWORK_DOMAIN	PHYSICAL_MEMORY	HOSTNAME	OS_NAME	CPU_MODEL
METRIC_ID	NETWORK_SUBDOMAIN_2	PHYSICAL_MEM_KB	HOST_ID	OS_VERSION	CPU_SPEED
	OS_NAME_VERSION	HOSTNAME		OS_SUB_VER	
	OS_NAME	HOST_ID			
	OS_VERSION				
	CPU_MODEL				
	NETWORK_SUBDOMAIN				
	CPU_SPEED				
	IP_C_NETWORK				

Queries:
- HOST_ID
- STATE_STRT_DTTM
- STATE_END_DTTM
- HOSTNAME
- IP_ADDRESS
- OS_NAME
- OS_VERSION
- OS_SUB_VER
- PHYSICAL_MEM_KB
- NUM_CPUS
- CPU_MODEL
- CPU_SPEED
- SYSTEM_PURPOSE
- IP_A_NETWORK
- IP_B_NETWORK

Measures:
- MIN_VALUE
- MAX_VALUE
- AVG_VALUE
- PHYSICAL_MEM_KB

Figure 7-38 Opening up the PowerPlay

6. The PowerPlay program, which is the GUI interface for Cognos, starts. From this you can create various crosstab or graph reports. Consider the following example, which shows minimum, maximum, and average values for all metrics and for all hosts in different layers (Figure 7-39 on page 249).

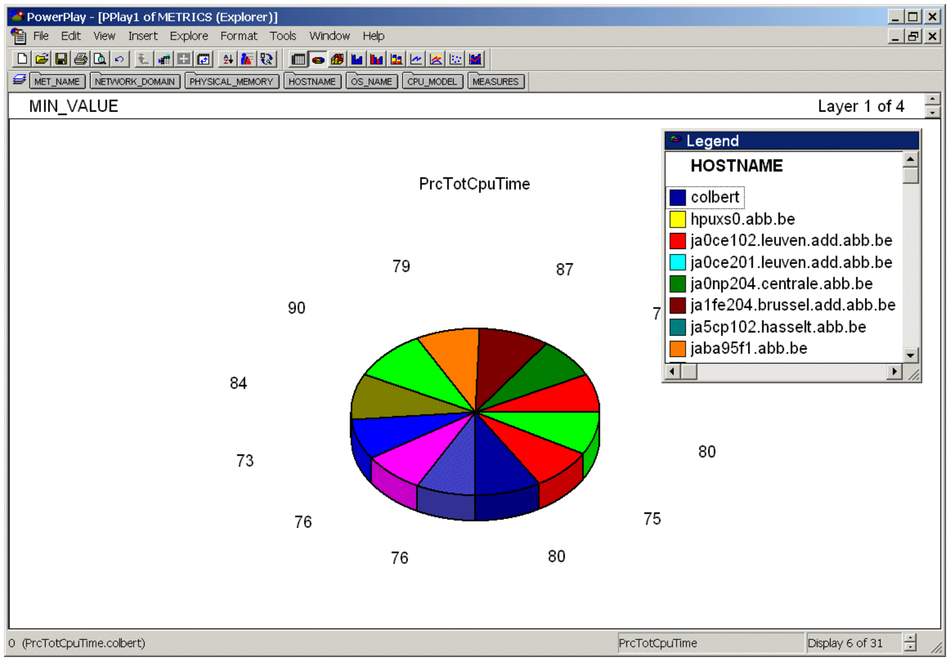


Figure 7-39   Create graph report

To get this graph we did the following:

a. Select the Pie chart icon from the top tool bar.

b. Drag and drop MET_NAME onto the pie chart.

c. Drag and drop the HOSTNAME into the Legend.

d. Drag and drop MEASURES on to the stacked paper icon in top left-hand corner.

7.4.4 Cognos sample reports

In this section we will show sample reports we have created from Tivoli Distributed Monitoring and Tivoli Inventory data.

Figure 7-40 on page 250 shows the average CPU run queue length for all hosts ranked by the highest CPU run queue length.

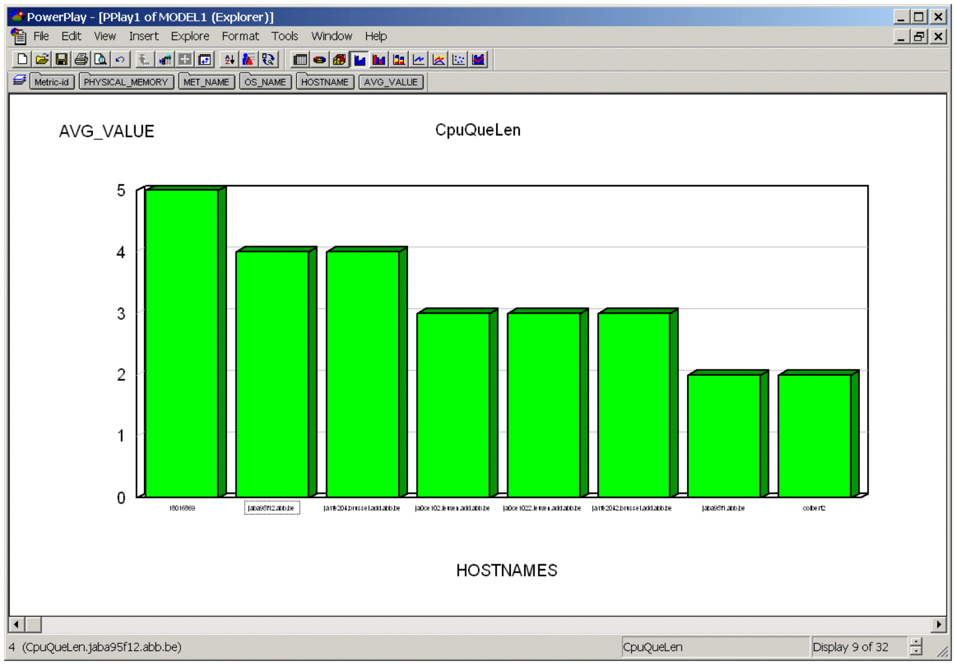

PowerPlay - [PPlay1 of MODEL1 (Explorer)]
File Edit View Insert Explore Format Tools Window Help
Metric-id
PHYSICAL_MEMORY
MET_NAME
OS_NAME
HOSTNAME
AVG_VALUE
AVG_VALUE
CpuQueLen
5
4
3
2
1
0
18016869
jaba95f12.abb.be
ja1fe204.brussel.add.abb.be
ja0ce102.leuven.add.abb.be
ja0ce1022.leuven.add.abb.be
ja1fe2042.brussel.add.abb.be
jaba95f1.abb.be
colbert2
HOSTNAMES
4 (CpuQueLen.jaba95f12.abb.be)
CpuQueLen
Display 9 of 32

Figure 7-40 Bar graph report

- Figure 7-41 on page 251 shows the average free swap space available in all AIX 4.3.3 machines per week.

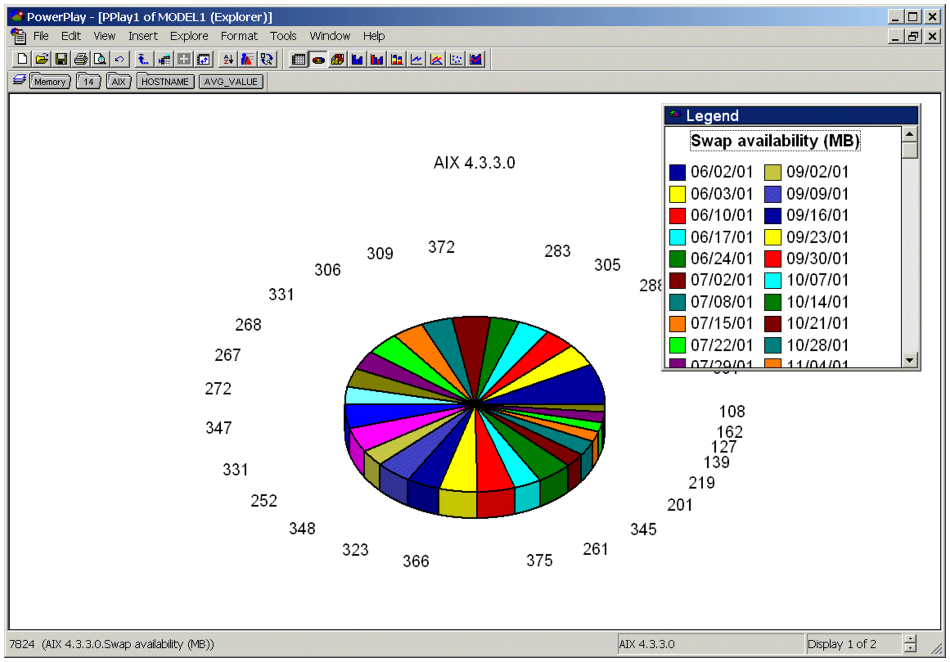


Figure 7-41 Pie chart graph

8

Real-life scenarios

This chapter will show how you can use Tivoli Enterprise Data Warehouse RPI interface and OLAP tools together to solve real-life problems.

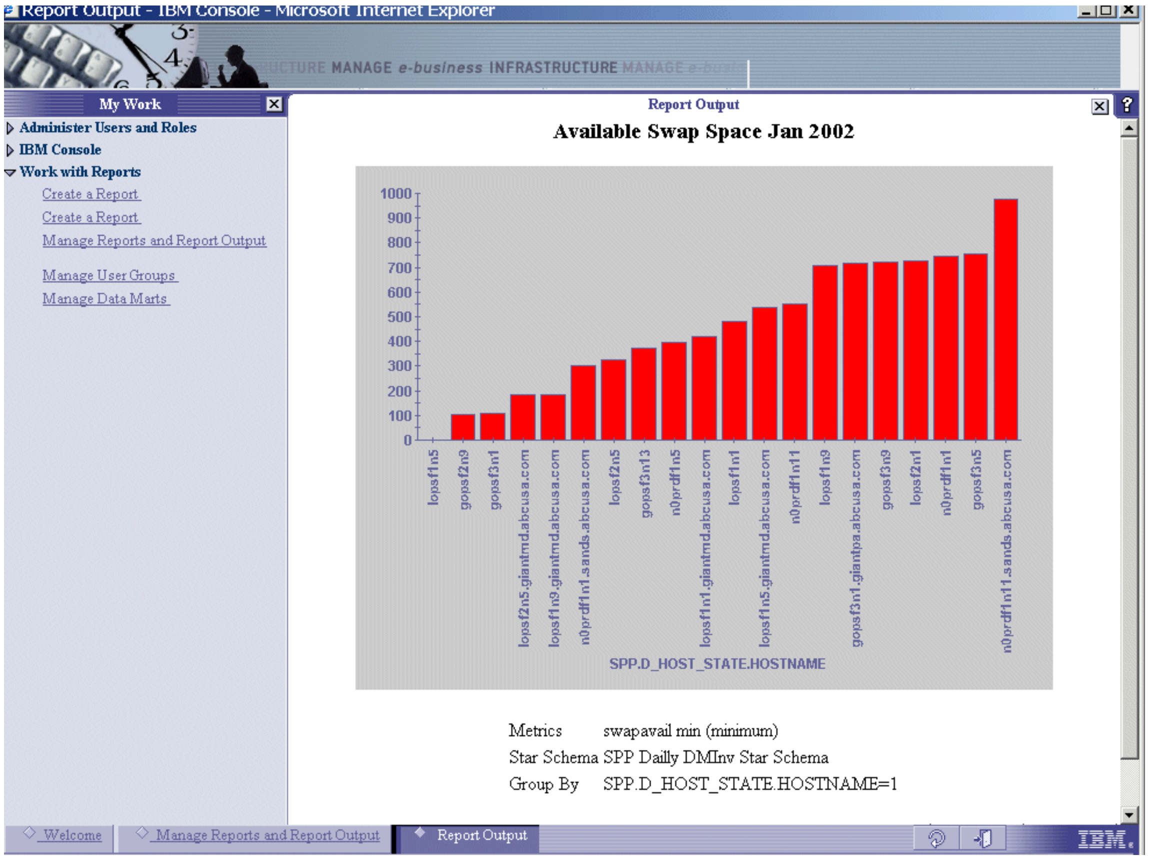
This scenario covers the following parts:

- ► Description of the environment
- ► Description of the scenario
- ► Implementation
- ► Wrap-up

8.1 Scenario 1 - Low swap space

In this first scenario we assume that our servers are monitored by Tivoli Distributed Monitoring (Version 3.7) and the data is imported into Tivoli Enterprise Data Warehouse. We use this data to produce scheduled monthly standard reports to evaluate the performance and to identify problems. In this example scenario we demonstrate how to find problems and how to create further specific reports to analyze the problem in more detail.

A good starting point for observing problems is extreme reports of basic system resources. In our example we consider the minimum available swap space (see Figure 8-1). The report displays the minimum value of all measurements of one month grouped by host names.


Report Output - IBM Console - Microsoft Internet Explorer
MANAGE e-business INFRASTRUCTURE
My Work
Administer Users and Roles
IBM Console
Work with Reports
Create a Report
Create a Report
Manage Reports and Report Output
Manage User Groups
Manage Data Marts
Report Output
Available Swap Space Jan 2002
1000
900
800
700
600
500
400
300
200
100
0
lopsf1n5
gopsf2n9
gopsf3n1
lopsf2n5.giantmd.abcusa.com
lopsf1n9.giantmd.abcusa.com
n0prdf1n1.sands.abcusa.com
lopsf2n5
gopsf3n13
n0prdf1n5
lopsf1n1.giantmd.abcusa.com
lopsf1n1
lopsf1n5.giantmd.abcusa.com
n0prdf1n11
lopsf1n9
gopsf3n1.giantpa.abcusa.com
gopsf3n9
lopsf2n1
n0prdf1n1
gopsf3n5
n0prdf1n11.sands.abcusa.com
SPP.D_HOST_STATE.HOSTNAME
Welcome
Manage Reports and Report Output
Report Output
IBM.

Metrics swapavail min (minimum)
Star Schema SPP Dailly DMInv Star Schema
Group By SPP.D_HOST_STATE.HOSTNAME=1

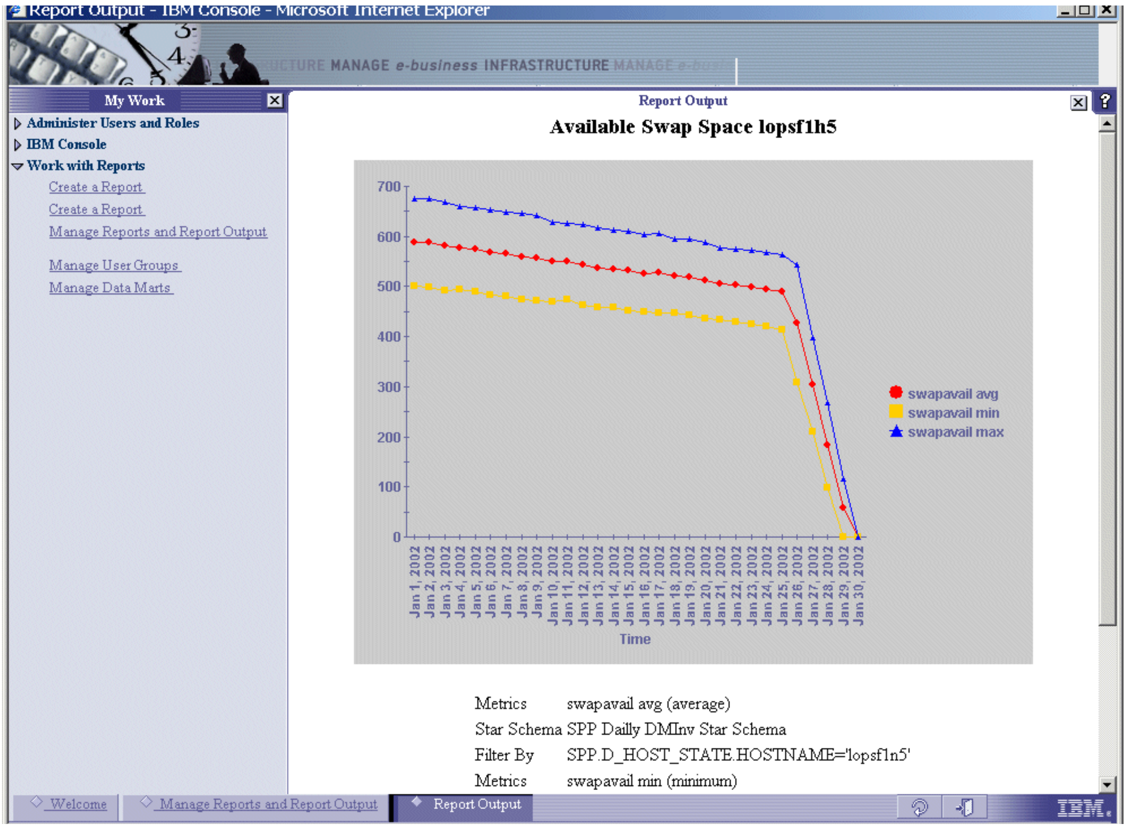
Figure 8-1 Monthly extreme report for available swap space

On the server lopsn1f5 we find a minimum swap space of 0 MB. It might have been a temporary problem, but the problem might also still exist. We now want to know when and for how long the problem occurred.

For this reason we create a health report for this particular server to gain further information. We choose for the metrics the minimum, maximum, and average value of available swap space. The filter is the host name found in the previous report. The time period for the report is the month used to create the extreme report.

Note: To complete this task a filter for host name must be applied. This is possible in the Tivoli Enterprise Data Warehouse Report Interface only, when there are less than 27 host names in the data mart. If you have more hosts in your environment you need to use an OLAP tool to generate this report.

The result of this report is depicted in Figure 8-2.


Report Output - IBM Console - Microsoft Internet Explorer
MANAGE e-business INFRASTRUCTURE
My Work
Administer Users and Roles
IBM Console
Work with Reports
Create a Report
Create a Report
Manage Reports and Report Output
Manage User Groups
Manage Data Marts
Report Output
Available Swap Space lopsf1h5
700
600
500
400
300
200
100
0
Jan 1, 2002 … Jan 30, 2002
Time
swapavail avg
swapavail min
swapavail max
Welcome
Manage Reports and Report Output
Report Output

Metrics	swapavail avg (average)
Star Schema	SPP Dailly DMInv Star Schema
Filter By	SPP.D_HOST_STATE.HOSTNAME='lopsf1n5'
Metrics	swapavail min (minimum)

Figure 8-2 Health report for server found in extreme report

We see that swap space has a bend on January 26th. Thus we can conclude that something has happened at this time. In this example a new software product has been installed, which has a memory leak. The installation of a patch for the new software product is needed.

In this example we have seen how we can obtain important information from the central warehouse data, which helps us to isolate and understand a problem in the IT infrastructure.

8.2 Scenario 2 - Slow application

For the second scenario, which is seen in Figure 8-3, we have to go into more detail about the environment. We have two applications that we call here, for the sake of simplicity, application A and application B, respectively. Each application has 20 load-balanced application servers (appserva01 to appserva20 for application A and appservb01 to appservb20 for application B), which are accessed by many clients in the enterprises network. Each application maintains its own database server (dbserva for application A and dbservb for application B).

Application A needs to import data from application B on a daily basis. The data is pulled from dbservb, processed, and pushed to dbserva by a server named gateway. The database update process is scheduled during the night as the applications are mainly used during the day.

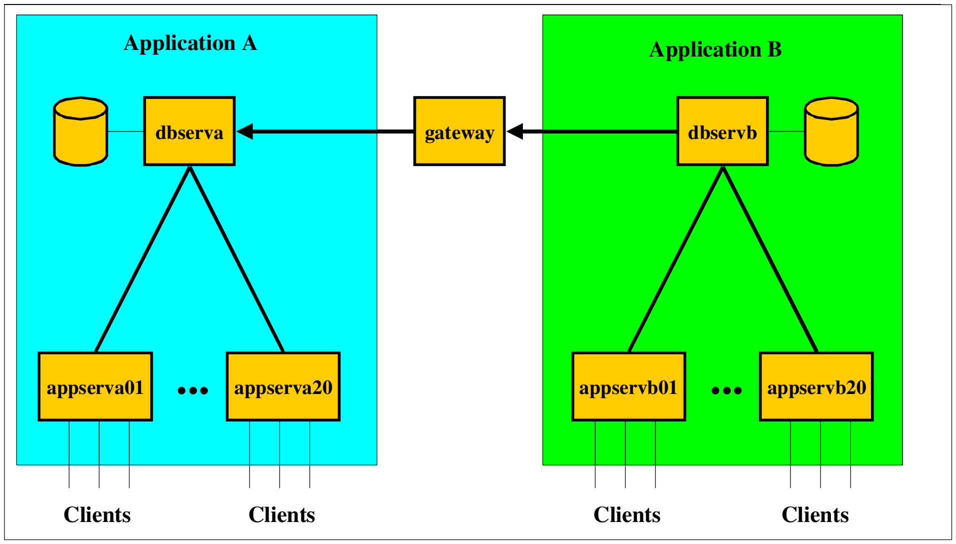
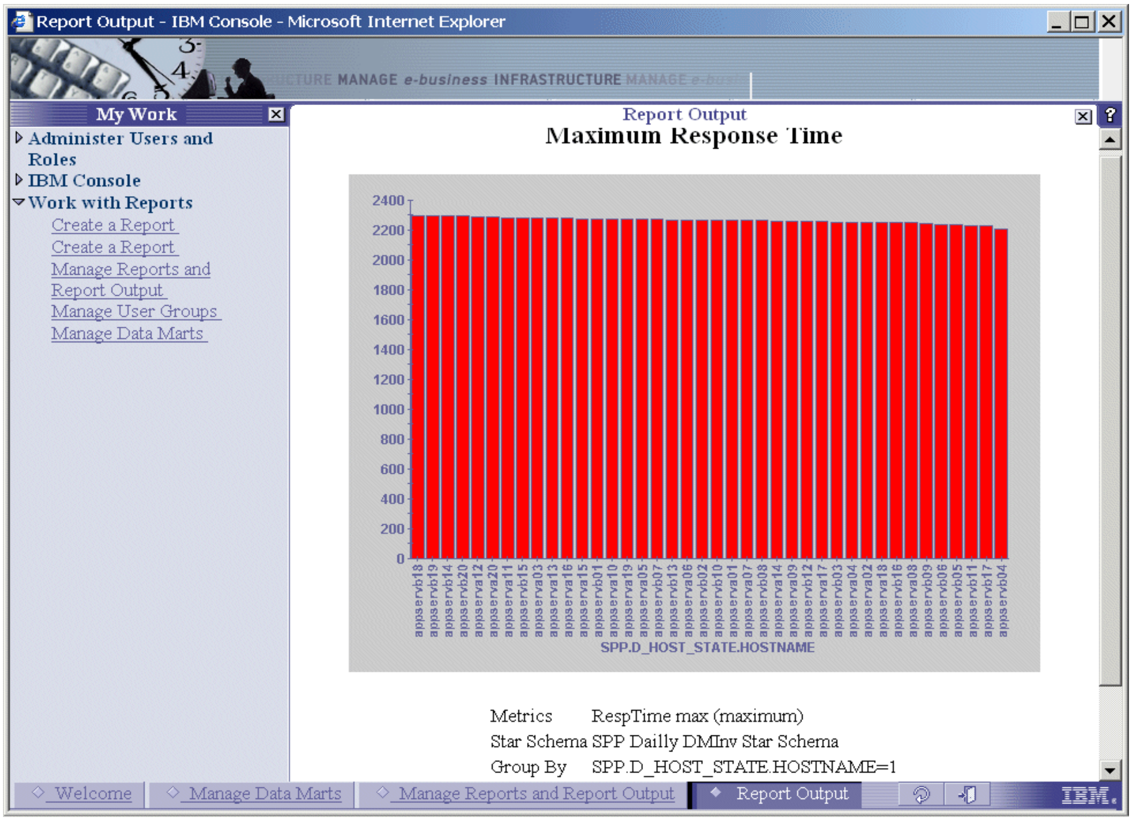


Figure 8-3 Environment for the second scenario

All servers in this environment are monitored with Tivoli Distributed Monitoring 3.7. Furthermore the response time of the applications on all application servers is measured by Tivoli Application Performance Management (TAPM). The data from both applications is imported to the central data warehouse. A data mart ETL has been set up that builds data marts that contain measurements from both applications.

In this environment we now have the problem reported by users that the applications are very slow, especially in the morning hours. We now have to isolate the reason for this slowdown using our data.

To quantify the problem we check the response time data of our applications. A first interesting question is whether all application servers are slow or whether the problem is experienced on a certain set of servers only. To answer this question we create a extreme report of the maximum value of the response time grouped by the host name from the daily star schema. This report is shown in Figure 8-4 on page 258.

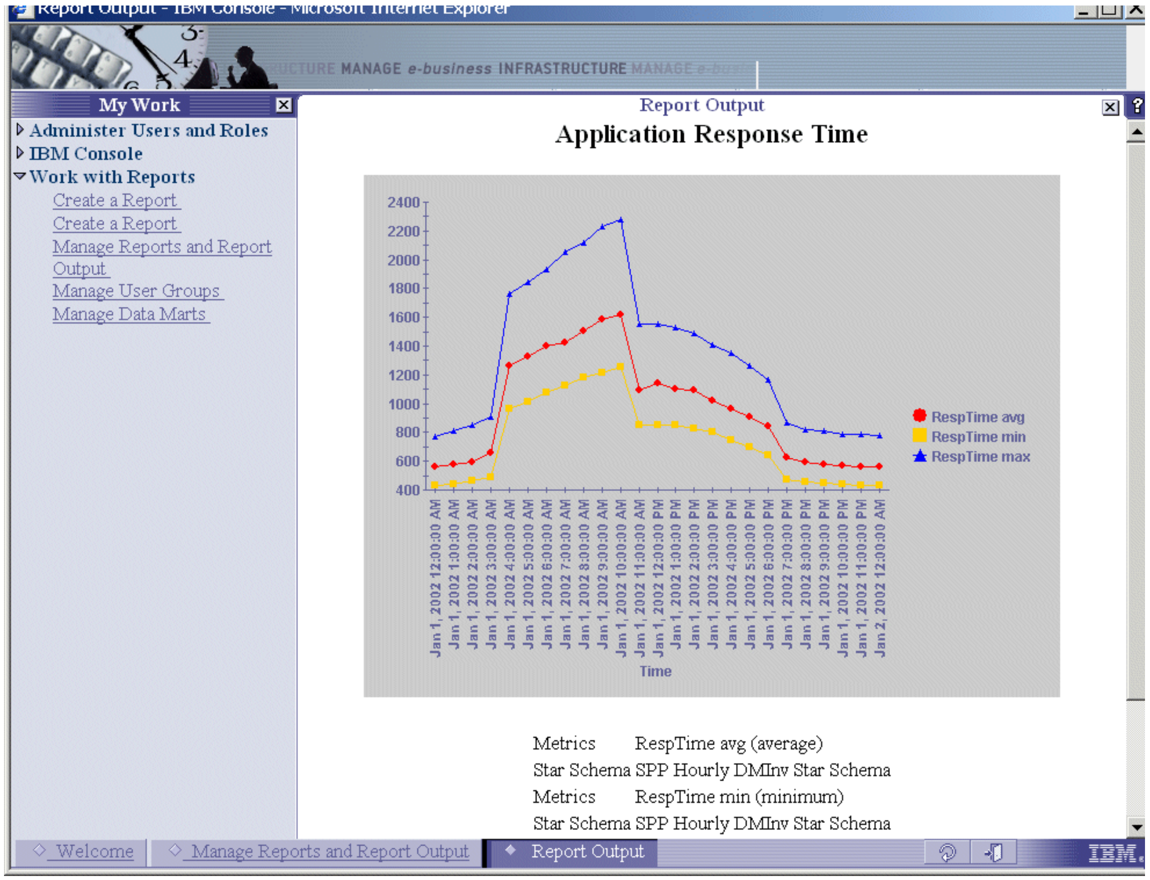


Metrics RespTime max (maximum)
Star Schema SPP Dailly DMInv Star Schema
Group By SPP.D_HOST_STATE.HOSTNAME=1

Figure 8-4 Extreme report of the maximum application response time

We see that all servers have roughly the same long maximum response times. As there are complaints only in the morning hours, we check the application response time distribution over one day.

In Figure 8-5 on page 259 we see a health report produced by the Tivoli Enterprise Data Warehouse Report Interface. It shows minimum, average, and maximum values of the application response time for one day averaged over all servers. The data is taken from the hourly star schema.



Metrics	RespTime avg (average)
Star Schema	SPP Hourly DMInv Star Schema
Metrics	RespTime min (minimum)
Star Schema	SPP Hourly DMInv Star Schema

Figure 8-5 Health report of the response time versus a one-day period

We find long response times in the morning increasing until 10:00 a.m. In the afternoon there are normal response times. Using OLAP tools, like Brio, we can check the daily response time behavior of single machines too. Such a report is shown in Figure 8-6 on page 260 where the response time of four randomly chosen application servers are considered.

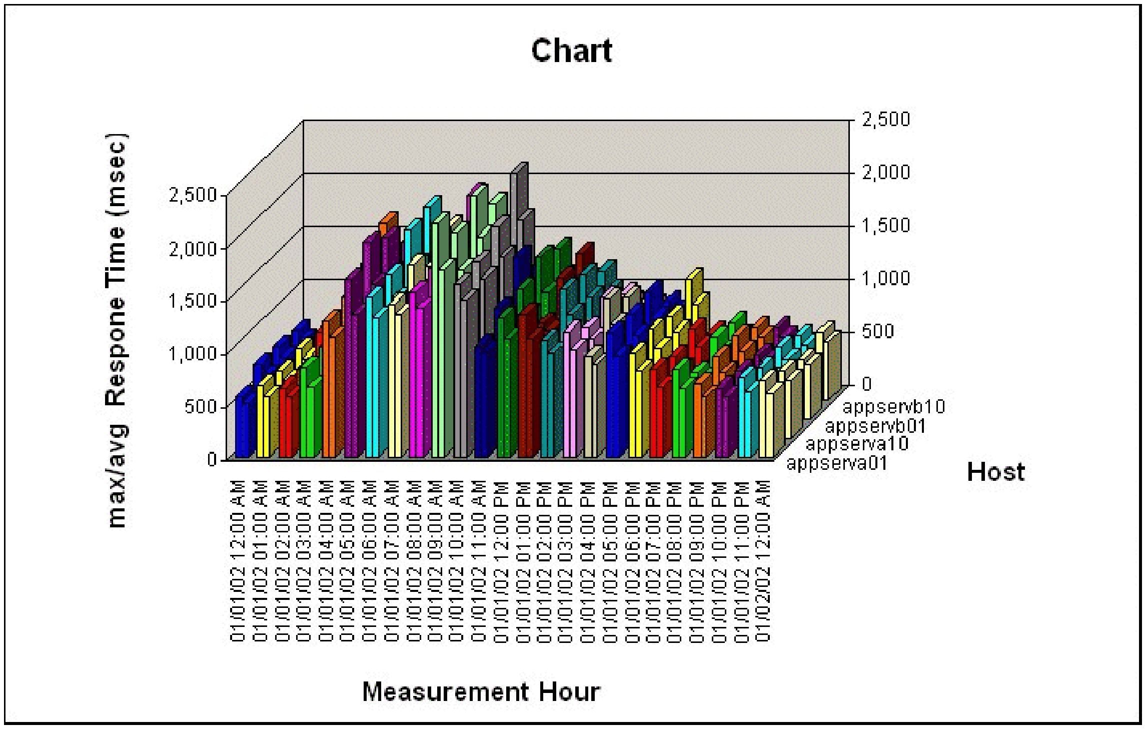


Figure 8-6 Application response time versus a one-day period

Again we find long response times until 11:00 a.m. but we still do not know, what the reason for this long response times is.

One reason could be that the workload of the application servers is too high during these hours. For this reason we now check the available swap space and the CPU usage of the application servers.

Here we again use Brio reports for four randomly chosen application servers. We could instead use average, minimum, or maximum reports over all application servers using the Tivoli Enterprise Data Warehouse Report Interface.

In Figure 8-7 on page 261 there is a Brio report for the minimum available swap space versus a one-day period for four application servers. We find that the minimum available swap space is roughly constant. It is at every time sufficient to run the application. Thus there is no problem with the swap space on our application servers that could be responsible for the long response times.

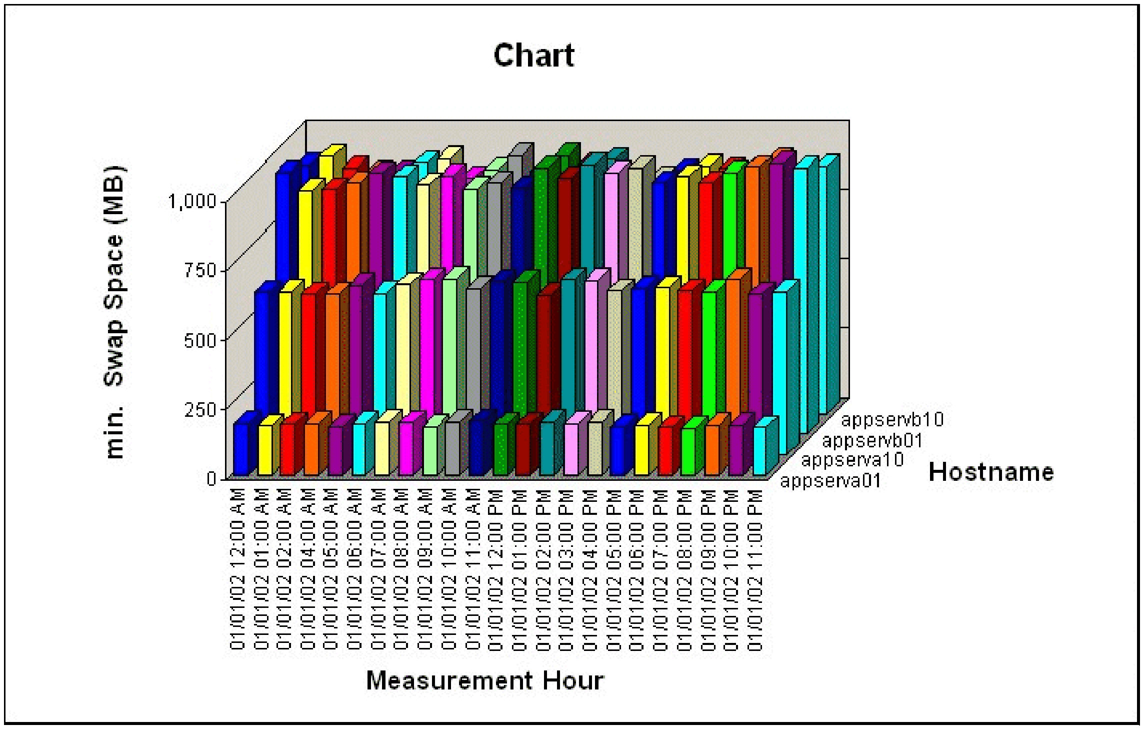


Figure 8-7 Minimum available swap space versus a one-day period

We now check the average CPU usage of the application servers. In Figure 8-8 on page 262 there is a Brio report for the maximum total CPU usage versus a one day period for four application servers. We find that the CPU usage increases during office hours but is also at every time sufficient to run the application. We conclude that there is no high workload of our application servers that causes the long response times.

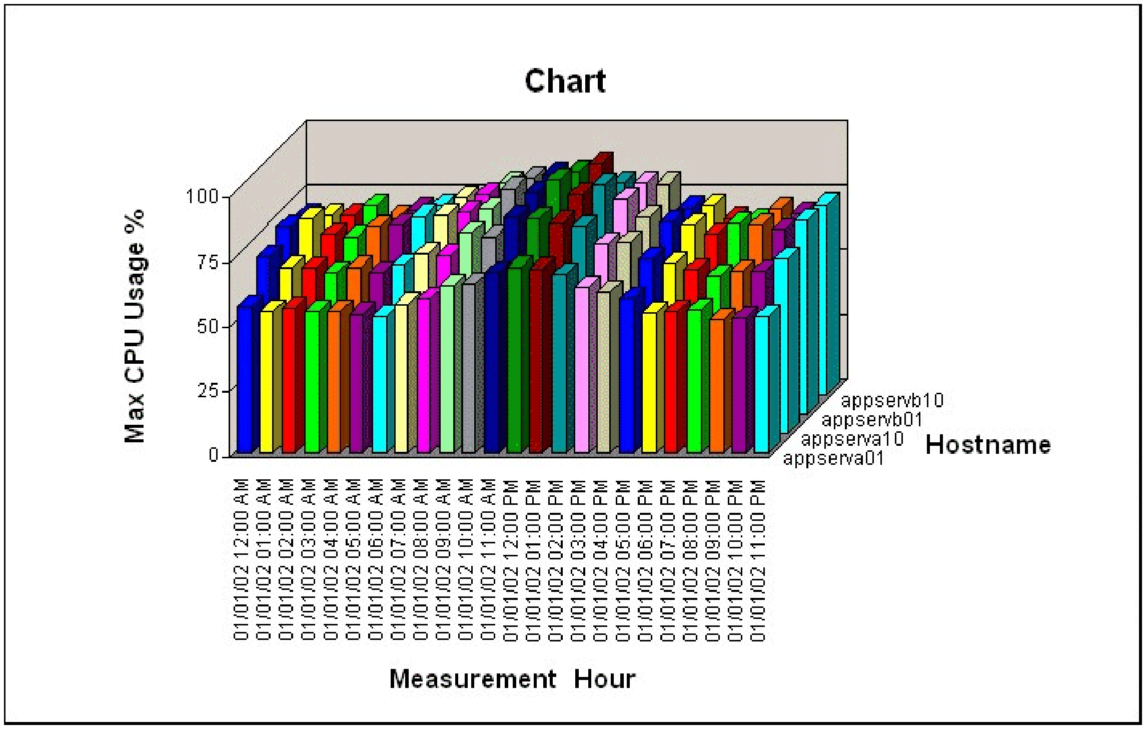


Figure 8-8 Maximum CPU usage versus a one-day period

Another possible source of the problem could be the workload of the database servers. Thus we now check the average CPU usage of the database servers (see Figure 8-9 on page 263). As the workload of the database server is also influenced by the database update process, we also consider the gateway server in this report.

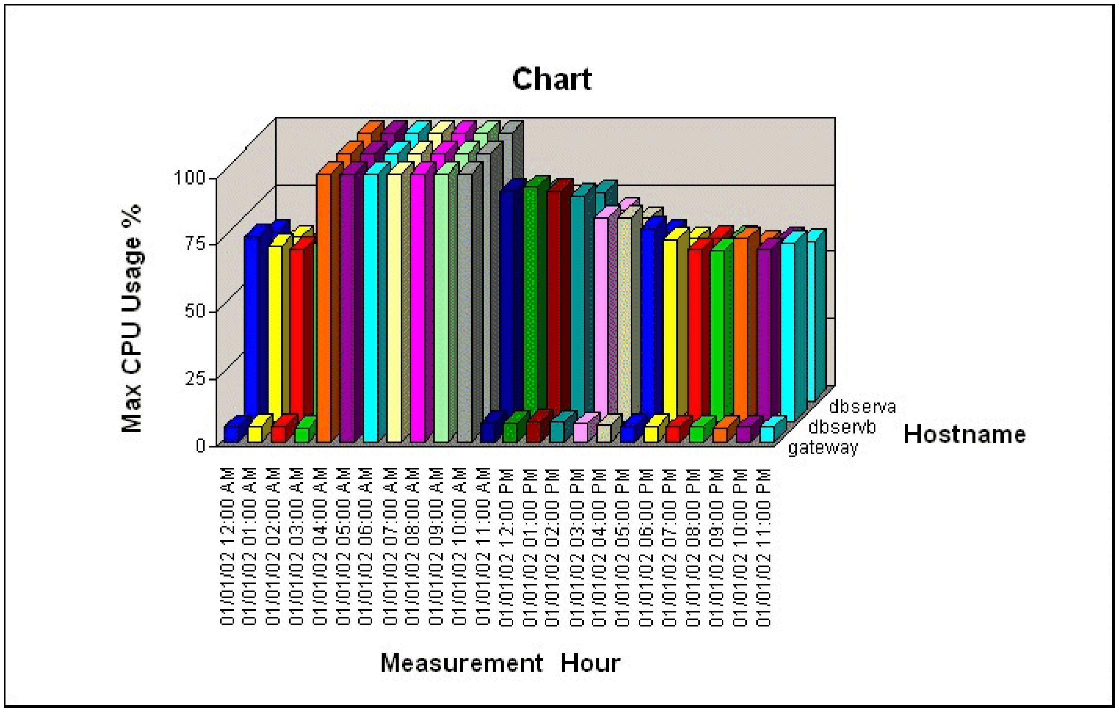


Figure 8-9 Maximum CPU usage versus a one-day period

In the graphs of the CPU usage of the database servers we find that both database servers are 100 percent busy between 4:00 a.m. and 10:00 a.m. We now compare these graphs with the one of the gateway server. We find that it is busy in exactly the same time interval.

Thus we can conclude that the database update, which is scheduled for each night, slows down our database servers. This in turn causes long response times of the applications (compare the time interval of increased response times in Figure 8-5 on page 259).

Apparently the duration of the database update has increased due to an increased data volume to be processed and now lasts until 11:00 a.m. Thus the workload of the database update and the workload from the application sum up in the early business hours, which leads to long response times.

We see that the import process is scheduled for 4:00 a.m. As the applications are rarely used at night, the solution of the problem is to schedule the database update earlier in the night so that it will be finished when the working hours begin.

This example has shown how the data in the Tivoli Enterprise Data Warehouse can be productively used to solve and isolate problems in the IT infrastructure. Different views and correlations of data in the reports produced by the Report Interface or other tools provide information about the state of many components. They help to quickly localize non-functional components, and also provide information that helps to exclude possible sources of error.

9 Multi-customer environments

This chapter describes how you can use Tivoli Enterprise Data Warehouse in a multi-customer environment (such as a Service Provider). We will show you how to customize Tivoli Enterprise Data Warehouse in such environments.

This chapter has the following sections:

- Multi-customers environments
- A typical Service Provider scenario

9.1 Multi-customer environments overview

The Tivoli Enterprise Data Warehouse's multi-customer and multi-center design makes the installation and management of a single customer or single center environment as simple as possible, while still providing features that enable a multi-customer or multi-center environment, when required. To facilitate the multi-customer and multi-center installations, some added complexity is required in the design and installation process that impacts the single customer or center environment. As part of the multi-customer support, two different application scenarios are provided:

- Application data sources that are input sources to the central data warehouse that contain information valid for one customer only.
- Application data sources that are input sources to the central data warehouse that contain data for multiple customers.

A single customer installation can smoothly migrate to a multi-customer environment. However, if any application data from a single customer environment is imported into the central data warehouse so that it cannot be identified with a particular customer, the central data warehouse cannot be used to support a multi-customer environment. In this scenario, a customer's recourse is to migrate their old central data warehouse data into the new environment to support the multi-customer environment.

Separate applications can use different data fields to distinguish between customers, but within applications the data fields should be consistent. *The application must have at least one data field that determines customer uniqueness.* Some applications might decide that two or more data fields are needed to distinguish customer data. Either case is permitted as long as the data fields used are consistent within a single application. Usually a fully qualified host name will be sufficient. Data from applications are assigned valid customer account codes by matching certain data fields in the incoming data with pre-identified values in a matching database table. Because each application can use different fields and different number of fields to identify customers, each application has its own matching table that it uses during the central data warehouse ETL process.

Before a central data warehouse ETL can be run for the multi-customer environment, the product_code.cust_lookup and TWG.Cust tables must be populated with data. The customer should populate the TWG.Cust and product_code.cust_lookup tables as part of the planning and installation of the central data warehouse. An outside process or script can be used to populate the tables, but it should be run as a distinct step, scheduled separately from the central data warehouse ETL process. The data contents of the TWG.Cust and product_code.cust_lookup tables should be fully understood by an administrator to avoid data mingling between customers.

If the users want to implement either multi-customer or multi-center functionality, they can do this by changing the contents of the lookup tables, cust_lookup or centr_lookup, and populating the TWG.Cust and TWG.Centr tables in the central data warehouse. For example, for a multi-customer environment the user must:

- Delete the single row in the product_code.cust_lookup table. This row has Cust_Acct_Cd='DEF', Value='@', or Cust_ID=1, Value='@'.
- Populate the product_code.cust_lookup table. The data in this table is static and should not be populated as part of an ETL process. The data might be based on part of the fully qualified domain name. For example, all hosts in the ford.myserviceprovider.com domain belong to the customer Ford.
- Populate the TWG.Cust table with valid values. If using the customer account code method, the Cust_Acct_Cd field in the TWG.Cust table must match the Cust_Acct_Cd field in the product_code.cust_lookup table. If using the customer ID, the Cust_ID field in the TWG.Cust table must match the Cust_ID field in the product_code.cust_lookup table.

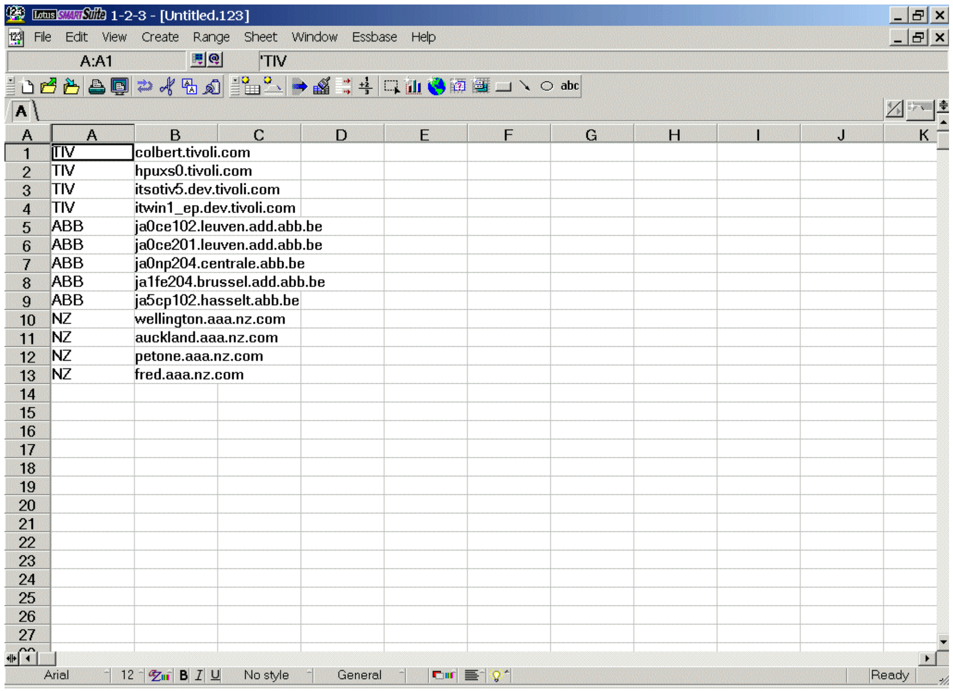
Once the performance data is moved through the Tivoli Enterprise Data Warehouse warehouses to the TWH_MART via the ETL processes, the views and users for each customer must be set up and configured.

9.2 Implementing a multi-customer scenario

In our multi-customer environment we have installed the Tivoli Distributed Monitoring Version 3.7 Warehouse Pack. All the scripts used are from this pack. You would follow a similar path for another application.

To implement a multi-customer environment you need to follow these steps:

1. Update the product_code.cust_lookup table for each application and the TWG.cust table in the TWH_CDW database.
   a. We created a spreadsheet for the cust_lookup table with two columns (see Figure 9-1 on page 268).
      - Column 1: CUST_ACCT_CD will be the customer account code.
      - Column 2: VALUE will be the machines fully qualified host name.



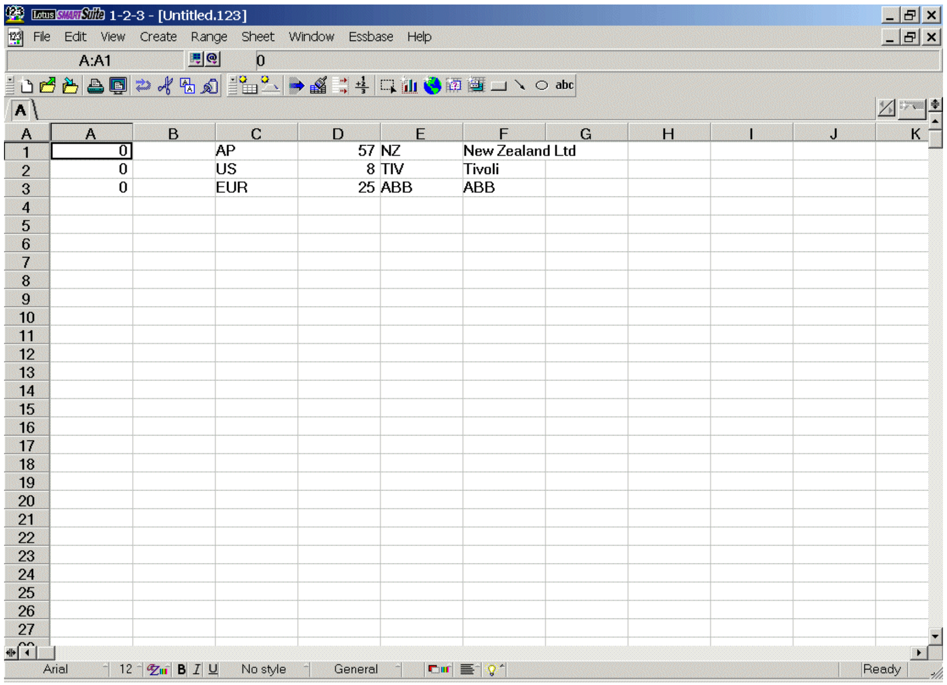
	A	B
1	TIV	colbert.tivoli.com
2	TIV	hpuxs0.tivoli.com
3	TIV	itsotiv5.dev.tivoli.com
4	TIV	itwin1_ep.dev.tivoli.com
5	ABB	ja0ce102.leuven.add.abb.be
6	ABB	ja0ce201.leuven.add.abb.be
7	ABB	ja0np204.centrale.abb.be
8	ABB	ja1fe204.brussel.add.abb.be
9	ABB	ja5cp102.hasselt.abb.be
10	NZ	wellington.aaa.nz.com
11	NZ	auckland.aaa.nz.com
12	NZ	petone.aaa.nz.com
13	NZ	fred.aaa.nz.com

Figure 9-1 Customer lookup table

b. This file is then saved as cust_lookup.csv.

c. We then created a spreadsheet for the cust table. This has six columns (see Figure 9-2 on page 269).

- Column 1: CUST_ID will be set by a trigger when a new customer is entered.
- Column 2: CUST_PARENT_CD.
- Column 3: GEOAREA_CD will be the geographic area from the GEOAREA table. See Appendix A, “GEOAREA and TMZON tables” on page 299.
- Column 4: TMZON_ID will be the time zone from the TMZON table. See Appendix A, “GEOAREA and TMZON tables” on page 299.
- Column 5: CUST_ACCT_CD will be the value from the CUST_LOOKUP table. See Figure 9-1.
- Column 6: CUST_NM will be the customer name.


Lotus SmartSuite 1-2-3 - [Untitled.123]
File Edit View Create Range Sheet Window Esbase Help
A:A1
0
Arial 12 No style General Ready

	A	B	C	D	E	F
1	0		AP	57	NZ	New Zealand Ltd
2	0		US	8	TIV	Tivoli
3	0		EUR	25	ABB	ABB

Figure 9-2 Configure customer table

d. This file is then saved as cust.csv.

e. We removed the current default value from the CUST_LOOKUP table by entering the following commands from a DB2 command window:

```
db2 connect to TWH_CDW user db2inst1
db2 DELETE FROM DMN.CUST_LOOKUP WHERE ((DMN.CUST_LOOKUP.VALUE='@'))
```

f. Then we inserted the values from the two spread sheets we created into the correct tables by running the following script from a DB2 command window:

```
db2 -f insert_new_cust.db2
```

The name of the script is insert_new_cust.db2. It contains the commands shown in Example 9-1.

Example 9-1 insert_new_cust.db2 script

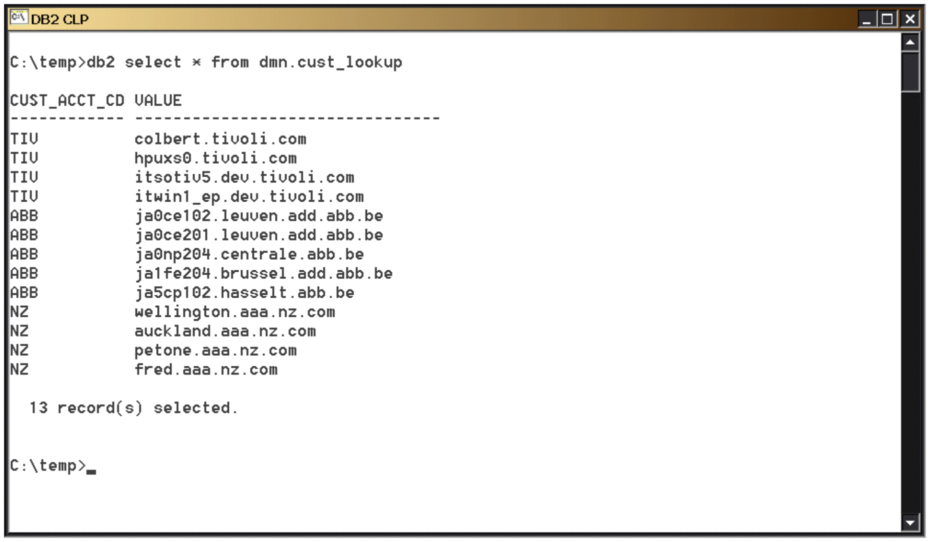
```
connect to twh_cdw user db2inst1
IMPORT FROM c:\temp\cust_lookup.csv OF DEL MODIFIED BY  chardel"" coldel,
decpt.  MESSAGES c:\temp\cust_lookup.log INSERT INTO DMN.CUST_LOOKUP
IMPORT FROM c:\temp\cust.csv OF DEL MODIFIED BY  chardel"" coldel, decpt.
MESSAGES c:\temp\cust.log INSERT INTO TWG.CUST
```

Where DB2inst1 is the user ID for the database, c:\temp\cust_lookup.csv is the location of the file we created (see Figure 9-1 on page 268), and c:\temp\cust.csv is the location of the file we created (see Figure 9-2 on page 269).

g. To check that the tables have the correct data inserted, run the following commands from a DB2 command window:

```
db2 select * from dmn.cust_lookup
```

The output of the command is seen in Figure 9-3.

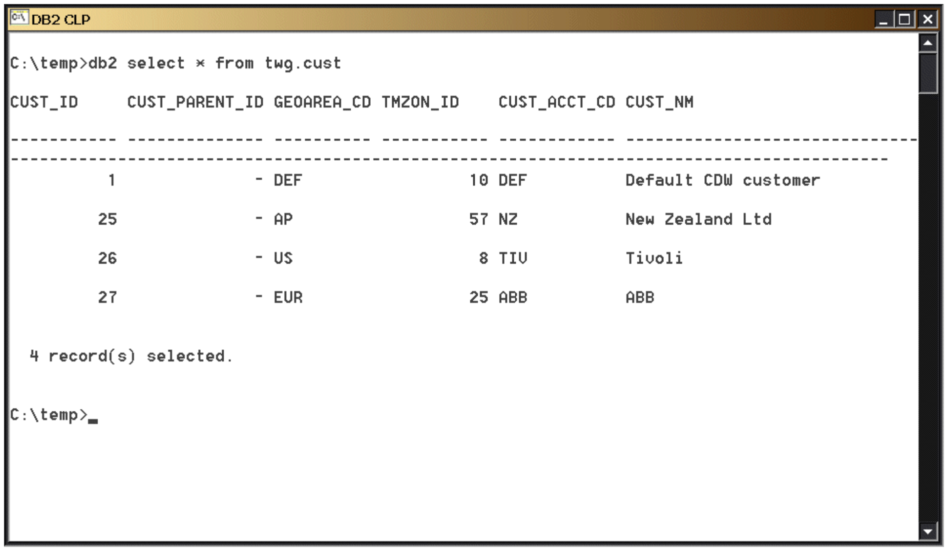


```
C:\temp>db2 select * from dmn.cust_lookup

CUST_ACCT_CD VALUE
------------ --------------------------------
TIV          colbert.tivoli.com
TIV          hpuxs0.tivoli.com
TIV          itsotiv5.dev.tivoli.com
TIV          itwin1_ep.dev.tivoli.com
ABB          ja0ce102.leuven.add.abb.be
ABB          ja0ce201.leuven.add.abb.be
ABB          ja0np204.centrale.abb.be
ABB          ja1fe204.brussel.add.abb.be
ABB          ja5cp102.hasselt.abb.be
NZ           wellington.aaa.nz.com
NZ           auckland.aaa.nz.com
NZ           petone.aaa.nz.com
NZ           fred.aaa.nz.com

  13 record(s) selected.


C:\temp>
```

Figure 9-3 Output of dmn.cust_lookout table

```
db2 select * from twg.cust
```

Output of the command is seen in Figure 9-4 on page 271.



```
C:\temp>db2 select * from twg.cust

CUST_ID     CUST_PARENT_ID GEOAREA_CD TMZON_ID    CUST_ACCT_CD CUST_NM

----------- -------------- ---------- ----------- ------------ ------------------------------
------------------------------------------------------------------------------------------
          1              - DEF                 10 DEF          Default CDW customer

         25              - AP                  57 NZ           New Zealand Ltd

         26              - US                   8 TIV          Tivoli

         27              - EUR                 25 ABB          ABB


  4 record(s) selected.


C:\temp>_
```

Figure 9-4 Output of twg.cust table

Once this is successful, the updates of the product_code.cust_lookup table and the twg.cust table in the TWH_CDW database are completed.

2. In a multi-customer environment we will be pulling data from multiple databases for the same application. Since the current version of Tivoli Enterprise Data Warehouse does not support an ETL process having two source databases of the same type of data (such as two source databases having Tivoli Distributed Monitoring data), we have to create or make a copy of an ETL process for each customer.

Note: This restriction will be fixed in a future release of Tivoli Enterprise Data Warehouse.

We did the following to create a copy of a process for each customer:

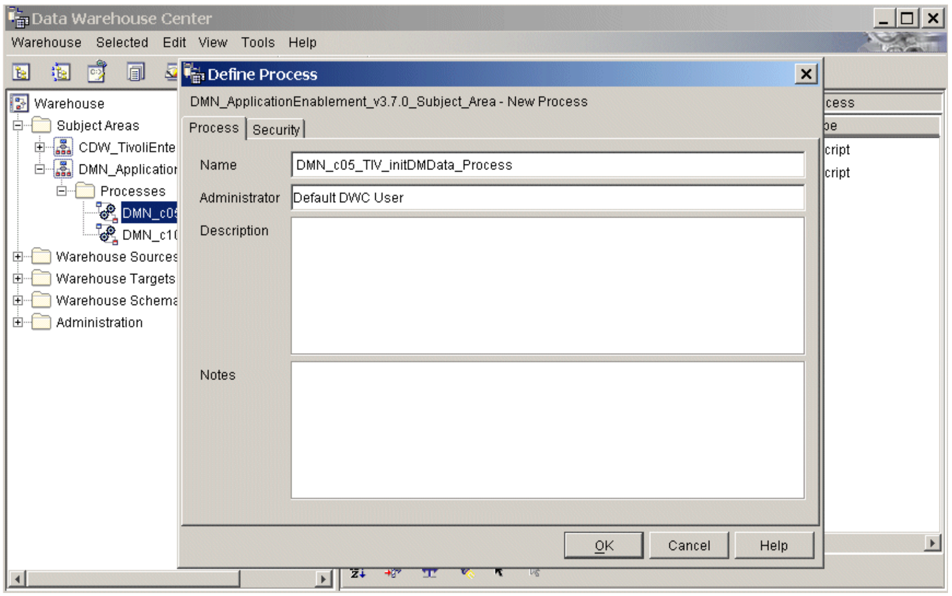
a. On the Windows taskbar, click **Start Programs -> IBM DB2 -> Control Center**. The Control Center window is displayed.

b. From the DB2 Control Center, start the DB2 Data Warehouse Center by clicking **Tools -> Data Warehouse Center**. The Data Warehouse Center Logon window is displayed.

c. Enter the user name and password and click **OK**. The Data Warehouse Center is displayed.

d. Expand the tree Subject Areas. There will be a list of applications. Expand the Application tree and expand the Processes tree; right click **Process** and select **Define**.

e. The Define Process window will appear (Figure 9-5). Enter a new process name and click **OK**.



DMN_ApplicationEnablement_v3.7.0_Subject_Area - New Process

Name: DMN_c05_TIV_initDMData_Process

Administrator: Default DWC User

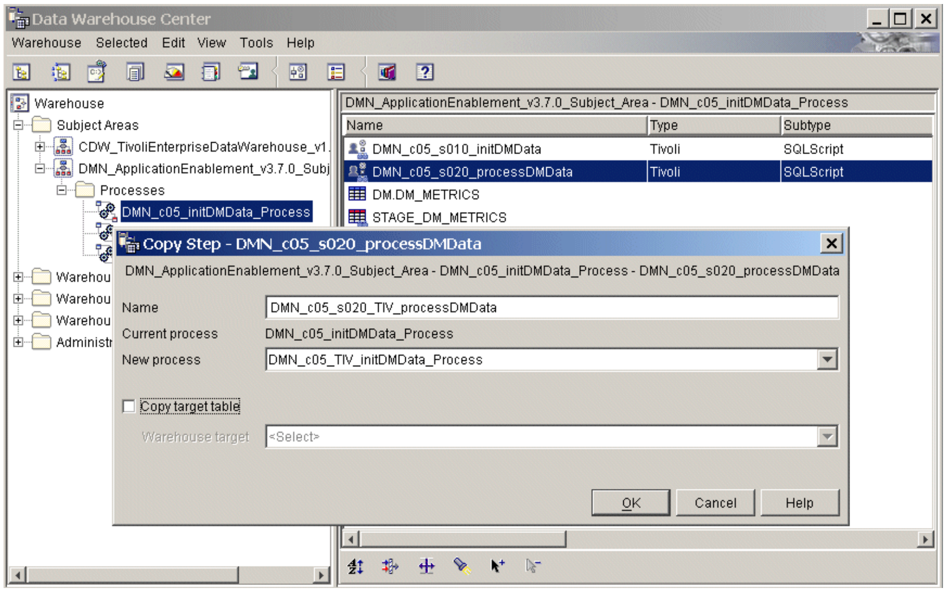
Description

Notes

Figure 9-5 Define Process window

f. Now select the process that we want to copy, right click the task in the right-hand pane, and select **Copy**.

g. Enter a new name, select the process from the previous step, uncheck Copy target table, and click **OK** (Figure 9-6 on page 273). Repeat this step for each task.


Data Warehouse Center

Warehouse Selected Edit View Tools Help

DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_c05_initDMData_Process

Name	Type	Subtype
DMN_c05_s010_initDMData	Tivoli	SQLScript
DMN_c05_s020_processDMData	Tivoli	SQLScript
DM.DM_METRICS		
STAGE_DM_METRICS		

Copy Step - DMN_c05_s020_processDMData

DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_c05_initDMData_Process - DMN_c05_s020_processDMData

Name	DMN_c05_s020_TIV_processDMData
Current process	DMN_c05_initDMData_Process
New process	DMN_c05_TIV_initDMData_Process

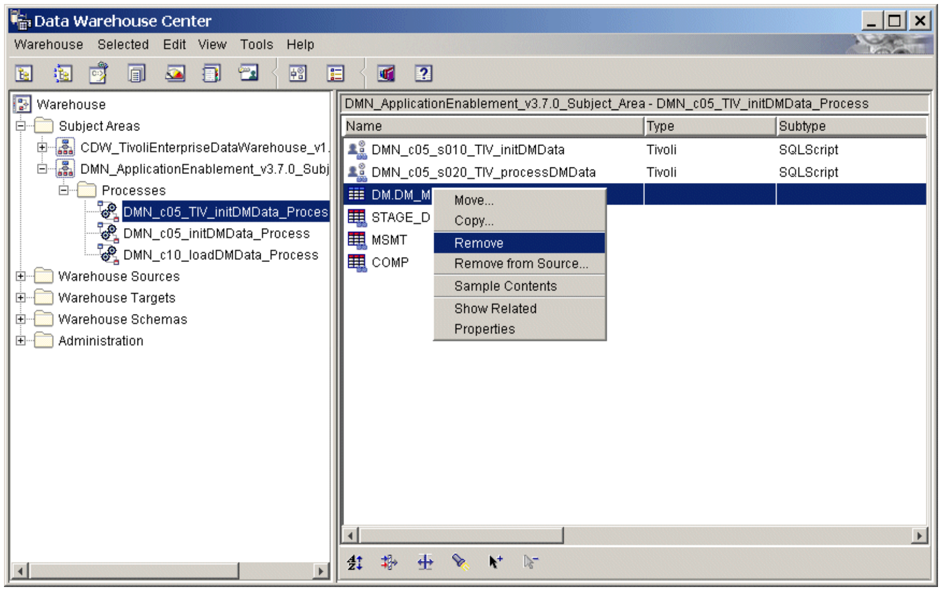
[ ] Copy target table

Warehouse target <Select>

Warehouse
Subject Areas
CDW_TivoliEnterpriseDataWarehouse_v1.
DMN_ApplicationEnablement_v3.7.0_Subj
Processes
DMN_c05_initDMData_Process
Warehou
Warehou
Warehou
Administr
OK
Cancel
Help

Figure 9-6 Copy Step window

h. Now our new process is set up, but has the wrong source database assigned, so we need to assign the source database for the TIV customer. Do this by left clicking the process we have just created for the customer, right click the source database, and select **Remove** (Figure 9-7 on page 274).


Data Warehouse Center
Warehouse
Selected
Edit
View
Tools
Help
Warehouse
Subject Areas
CDW_TivoliEnterpriseDataWarehouse_v1.
DMN_ApplicationEnablement_v3.7.0_Subj
Processes
DMN_c05_TIV_initDMData_Proces
DMN_c05_initDMData_Process
DMN_c10_loadDMData_Process
Warehouse Sources
Warehouse Targets
Warehouse Schemas
Administration
DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_c05_TIV_initDMData_Process
Move...
Copy...
Remove
Remove from Source...
Sample Contents
Show Related
Properties

Name	Type	Subtype
DMN_c05_s010_TIV_initDMData	Tivoli	SQLScript
DMN_c05_s020_TIV_processDMData	Tivoli	SQLScript
DM.DM_M		
STAGE_D		
MSMT		
COMP		

Figure 9-7 Remove source database window

i. Assign the new source database to the process task by right clicking **Process** and selecting **Open**.

j. The Process Model window will appear. Click the database button and drag it into the window.

k. The Add Data window will appear (Figure 9-8 on page 275). Select your source database table and click **OK**.

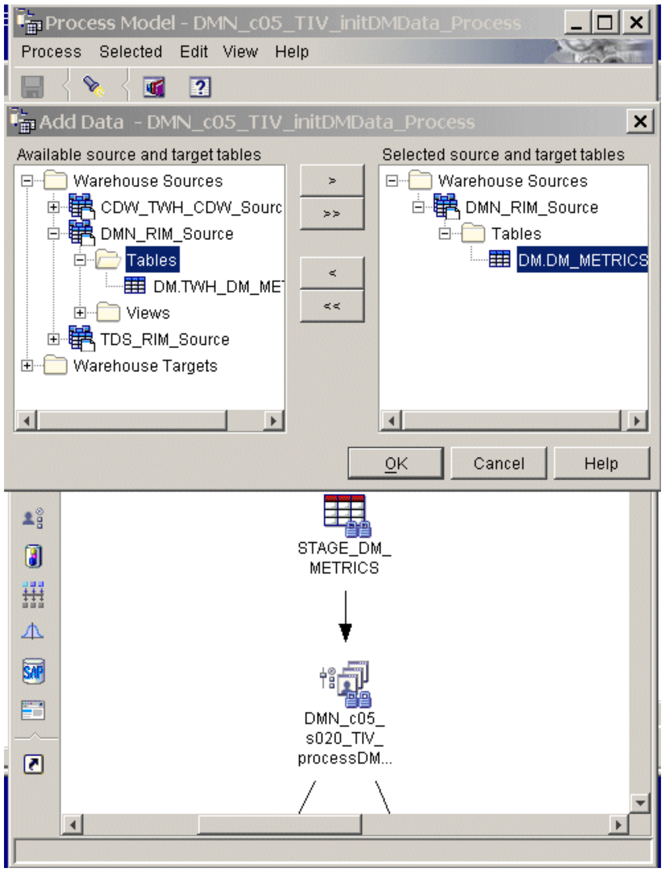


Figure 9-8 Add Data window

l. Now create a data link from the data source to the task, as in Figure 9-9 on page 276.

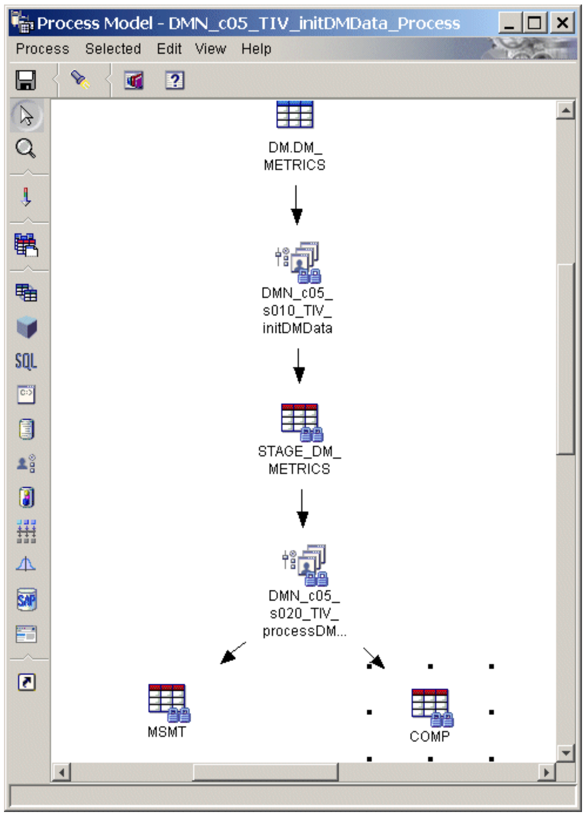

Process Model - DMN_c05_TIV_initDMData_Process
Process Selected Edit View Help
DM.DM_ METRICS
DMN_c05_ s010_TIV_ initDMData
STAGE_DM_ METRICS
DMN_c05_ s020_TIV_ processDM...
MSMT
COMP

Figure 9-9 Process Model window

m. In a multiple customer environment we need to be able to do a incremental extract from the source database. To do this, an extract control window is set up. *This is already implemented for you in a single customer environment but needs to be changed for each customer in a multi-customer environment.* We achieved this by adding a new row in the TWG.Extract_Control table in the TWH_CDW database. See the commands in Example 9-2.

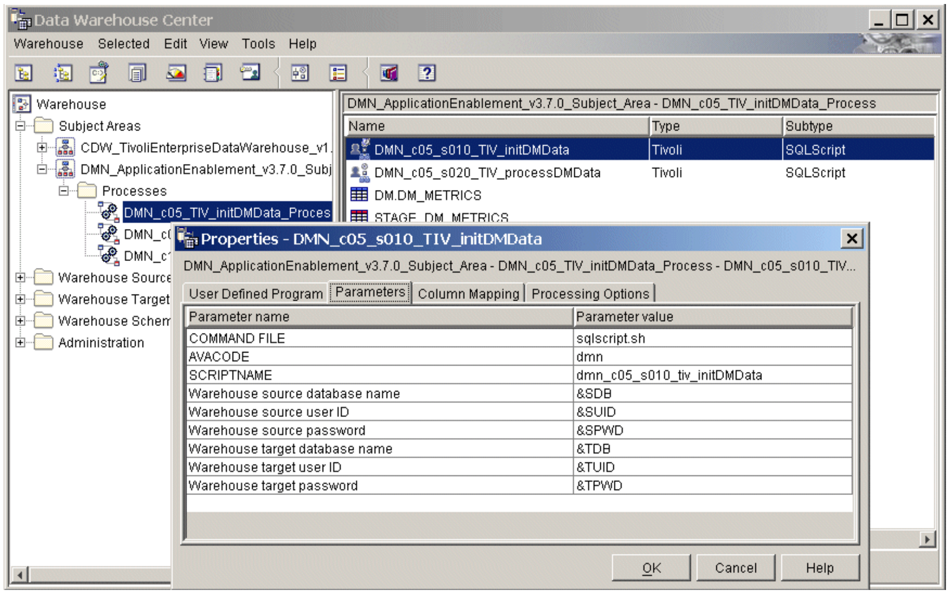
Example 9-2 Adding a new row in the TWG.Extract_Control table

```
connect to twh_cdw user db2admin
insert into twg.Extract_Control values (    'DM_METRICS_INIT_TIV' ,
'DMN.STAGE_DM_METRICS' ,    x'00000000000000000000',
x'99999999999999999999',   0,   2000000000,   '1970-01-01-00.00.00.000000',
'9999-01-01-00.00.00.000000')
```

```
insert into twg.Extract_Control values (    'DM_METRICS_TIV' ,
'DMN.STAGE_DM_METRICS' ,    x'00000000000000000000',
x'99999999999999999999',    0,    2000000000,    '1970-01-01-00.00.00.000000',
'9999-01-01-00.00.00.000000')
```

n. Then we need to edit the SQL script that runs the extract of data from the source database (see Appendix B, “Scripts” on page 303) and then we need to update our process model (see Figure 9-9 on page 276).

o. Right click the **Process** task selected, select **Properties -> Parameters** and enter the name of the edited SQL script.



Data Warehouse Center
Warehouse Selected Edit View Tools Help
Warehouse
Subject Areas
CDW_TivoliEnterpriseDataWarehouse_v1
DMN_ApplicationEnablement_v3.7.0_Subj
Processes
DMN_c05_TIV_initDMData_Proces
Warehouse Source
Warehouse Target
Warehouse Schem
Administration
DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_c05_TIV_initDMData_Process
DM.DM_METRICS
STAGE_DM_METRICS
Properties - DMN_c05_s010_TIV_initDMData
DMN_ApplicationEnablement_v3.7.0_Subject_Area - DMN_c05_TIV_initDMData_Process - DMN_c05_s010_TIV...
User Defined Program
Parameters
Column Mapping
Processing Options
OK
Cancel
Help

Name	Type	Subtype
DMN_c05_s010_TIV_initDMData	Tivoli	SQLScript
DMN_c05_s020_TIV_processDMData	Tivoli	SQLScript

Parameter name	Parameter value
COMMAND FILE	sqlscript.sh
AVACODE	dmn
SCRIPTNAME	dmn_c05_s010_tiv_initDMData
Warehouse source database name	&SDB
Warehouse source user ID	&SUID
Warehouse source password	&SPWD
Warehouse target database name	&TDB
Warehouse target user ID	&TUID
Warehouse target password	&TPWD

Figure 9-10 Process Properties window

p. Save this process model by clicking **Process -> Save**.

Note: Because of the changes made to the process model when a patch for the Tivoli Enterprise Data Warehouse is applied, these patches may not be applied to the customized process model, so you may need to recreate the process model again.

3. For multi-customer environments, a view needs to be created of each table in a star schema. The exception might be the D_METRIC table because the same metrics are likely to apply to all customers. The ETL designers will provide template insert statements such as in the

tedw_apps_etl/product_code/pkg/vnnn/mart/ddl/product_code_mart_schema .sql script (see Example 9-3).

Example 9-3 Template insert statements

```
Create view DM.D_V_host_TIV as (select * from DM.D_host where cust_id =’TIV’)
Create view DM.F_V_TIV_hour as (select * from DM.F_hour where host_id in
(select host_id from d_dm_host where cust_id =‘TIV’))
```

a. Create a view for every fact table that the customer will need access to for reporting, such as F_day and F_week tables, using the commands shown in Example 9-3 in a DB2 command window.

   Run this set of inserts to create views for each customer. You must replace TIV with your valid customer ID, and code inserts for each customer.

b. Once this is complete, we set up database users that include specific grants to only their applicable views. Using the commands shown in Example 9-4, we set up a user called TIV, and only assigned the views and tables that are applicable to this one customer. Run the commands shown in Example 9-4 from a DB2 command window.

Example 9-4 Grant statements

```
GRANT CONNECT ON DATABASE TO USER TIV
GRANT SELECT ON TABLE DM.D_METRIC TO USER TIV
GRANT SELECT ON DM.D_V_TIV_HOST_STATE TO USER TIV WITH GRANT OPTION
GRANT SELECT ON DM.F_V_TIV_HOUR TO USER TIV WITH GRANT OPTION
```

   Using the commands shown in Example 9-4, grant all the views you set up in the previous step for that customer, such as additional fact tables, F_day and F_week.

4. With a multi-customer implementation, you must also create star schemas made up of views in the Data Warehouse Center. This means that the IWH.StarSchema table contains both the view star schemas and the table star schemas (see “How star schemas are used to create reports” on page 104). To create a star schema, first you need to set up the view and table sources in the Data Warehouse Center and then you can set up your star schema. See the following:

   a. On the Windows taskbar, click **Start Programs -> IBM DB2 -> Control Center**. The Control Center window is displayed.

   b. From the DB2 Control Center, start the DB2 Data Warehouse Center by clicking **Tools -> Data Warehouse Center**. The Data Warehouse Center Logon window is displayed.

   c. Enter the user name and password and click **OK**. The Data Warehouse Center is displayed.

d. Expand the tree Warehouse Sources. Right click the application mart source, select **Properties**, and select **Tables and Views** from this window (Figure 9-11).

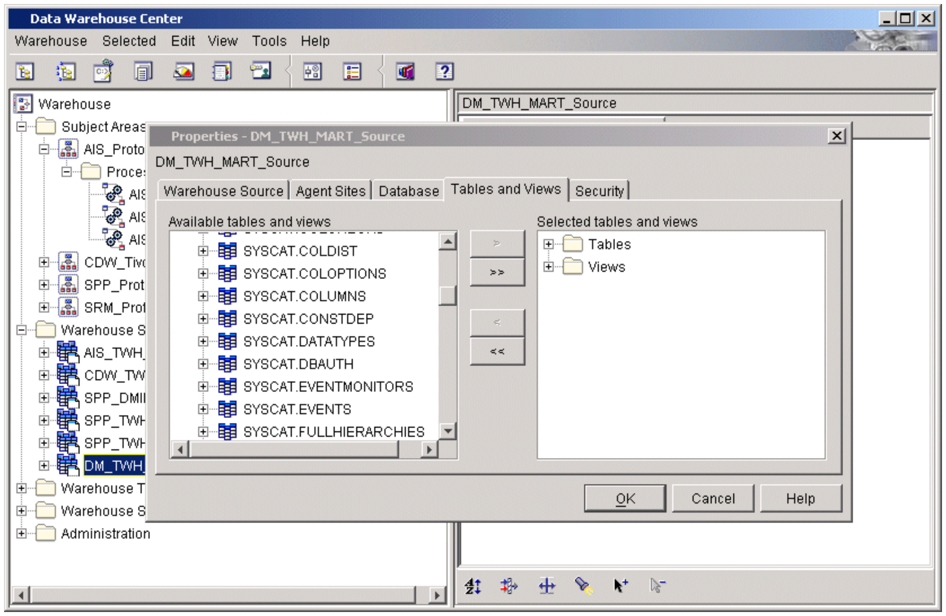
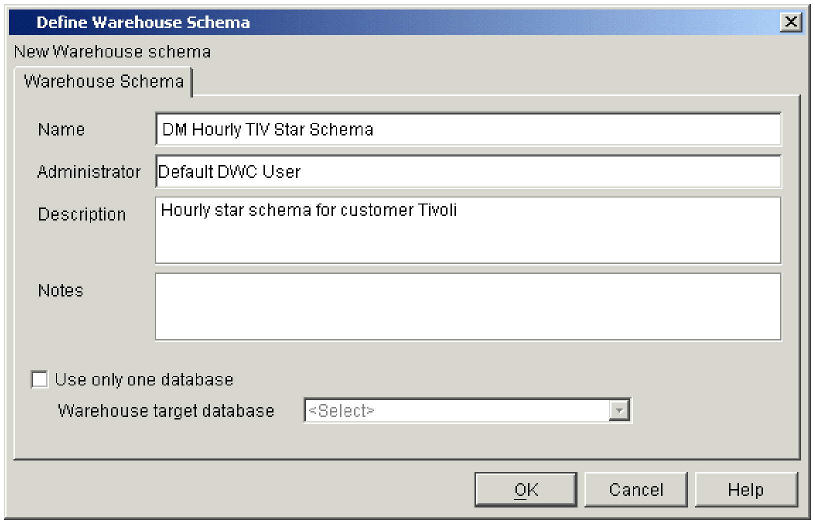


Figure 9-11 Add source window

e. Now select the views and tables for the star schemas, move them to the right side of the window, and click **OK**.

f. To set up the star schema right click **Warehouse Schemas** in the Warehouse Center and select **Define**. Enter a name and description (Figure 9-12 on page 280).



Define Warehouse Schema

New Warehouse schema

Warehouse Schema

Name: DM Hourly TIV Star Schema

Administrator: Default DWC User

Description: Hourly star schema for customer Tivoli

Notes:

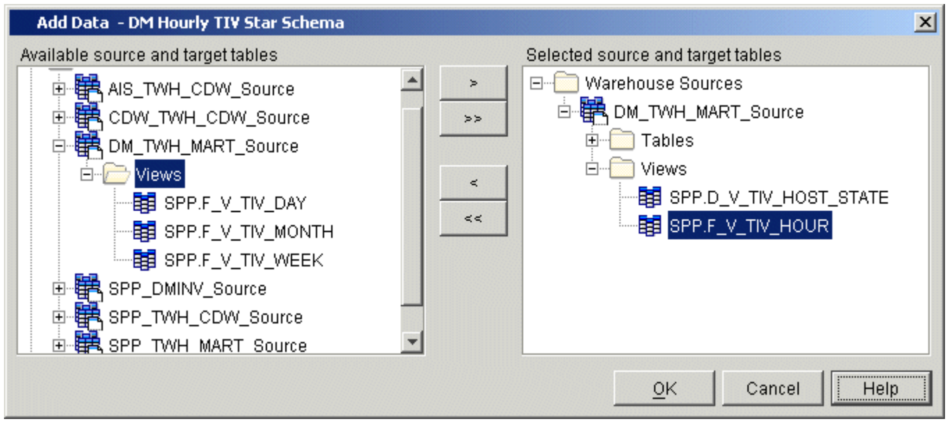
☐ Use only one database

Warehouse target database: <Select>

Figure 9-12 Define Warehouse Schema window

g. Click **OK**, expand the Warehouse Schemas tree, right click the schema name from the above step, and select **Add Table**.

h. Now add the table and views to your star schema from the source set up earlier (Figure 9-13).



Add Data - DM Hourly TIV Star Schema

Available source and target tables

```
AIS_TWH_CDW_Source
CDW_TWH_CDW_Source
DM_TWH_MART_Source
  Views
    SPP.F_V_TIV_DAY
    SPP.F_V_TIV_MONTH
    SPP.F_V_TIV_WEEK
SPP_DMINV_Source
SPP_TWH_CDW_Source
SPP_TWH_MART_Source
```

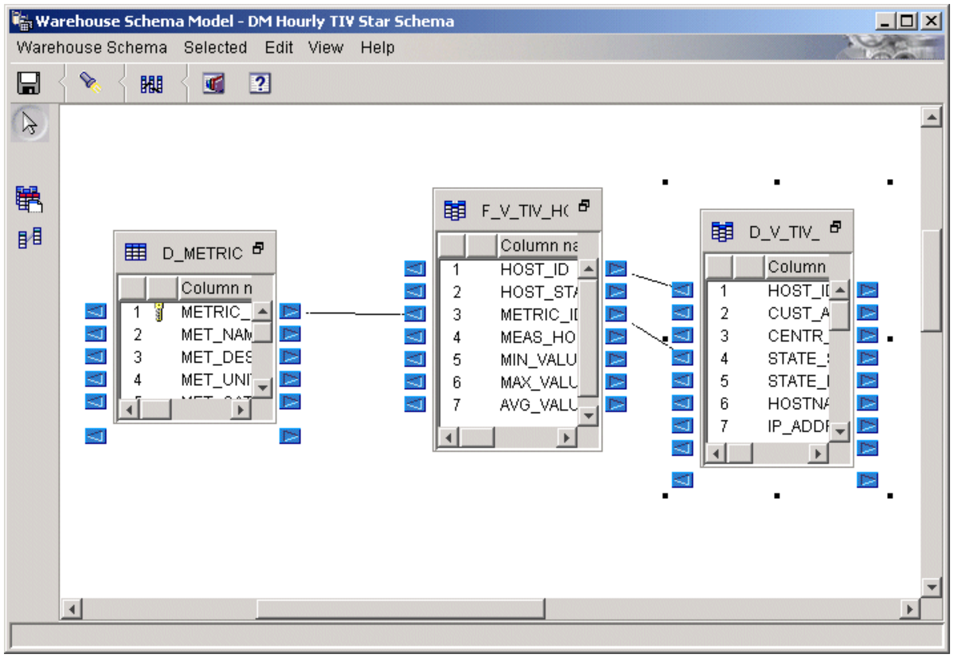
Selected source and target tables

```
Warehouse Sources
  DM_TWH_MART_Source
    Tables
    Views
      SPP.D_V_TIV_HOST_STATE
      SPP.F_V_TIV_HOUR
```

Figure 9-13 Add Data window

i. Click **OK**, right click the same schema, and select **Open**. The Warehouse Schema model will now open in this window. Set up your joins (Figure 9-14 on page 281).

Tip: The tables might be hidden behind each other. Select the top table and drag it to another part of the window.



	Column n
1	METRIC_
2	MET_NAM
3	MET_DES
4	MET_UNI

	Column na
1	HOST_ID
2	HOST_STA
3	METRIC_ID
4	MEAS_HO
5	MIN_VALU
6	MAX_VALU
7	AVG_VALU

	Column
1	HOST_ID
2	CUST_A
3	CENTR_
4	STATE_
5	STATE_
6	HOSTNA
7	IP_ADDR

Figure 9-14 Warehouse Schema Model window

j. Save this by clicking **Warehouse Schema** and selecting **Save**.

You may have more schemas to create, so use the same procedure. After creating these schemas, you are ready to create a report using these schemas. For report creation refer to Chapter 4, “Implementation of the Report Interface” on page 89.

10

Troubleshooting and maintenance

This chapter provides information on troubleshooting commands and techniques of Tivoli Enterprise Data Warehouse. It also covers maintenance of the Tivoli Enterprise Data Warehouse central repository and data marts.

This chapter has the following sections:

- Tivoli Enterprise Data Warehouse troubleshooting techniques
- Tivoli Enterprise Data Warehouse maintenance and back up

10.1 Troubleshooting techniques

In this section we give some useful tips for troubleshooting. In the first subsection we cover problems with installation. In the second subsection we give some hints for working with the Report Interface. In the last subsection we cover working with the Data Warehouse Center and installing your own ETLs and data marts.

10.1.1 Troubleshooting installation

Before you start the installation of Tivoli Enterprise Data Warehouse you should do the following:

- Check the *Tivoli Enterprise Data Warehouse Release Notes,* GI11-0857, for required prerequisites and patches for the operating systems and databases.
- If you are performing a silent installation of Tivoli Enterprise Data Warehouse on a UNIX system without a local X11 server, you must set and export the DISPLAY environment variable to a valid X11 server. The X11 server can be on a different system.
- For a distributed installation, the Domain Name Service (DNS) must be able to resolve host names from short names.

When all prerequisites are met you can start the installation. For detailed installation instructions refer to Chapter 3, “Installation and configuration” on page 47.

The Tivoli Enterprise Data Warehouse installer generates a log file named <TWH_TOPDIR>\TWH.log. Look into this file if the installation aborts with an error message or hangs. If the Tivoli Enterprise Data Warehouse installer fails, then all changes should be rolled back automatically. Thus, the reason for the install failure is usually not found in the last lines of the log file.

Additionally there is a log file, <TEMP>\twh_ibm_db2_runlog.log, which contains output and errors from any DB2 commands. In this log file you can search for MARKCORE, which marks the start of the core installation and MARK<AVA>, which marks the start of a warehouse pack installation (<AVA> is the three-letter product code of the product for which you install the warehouse pack). Note that these markers are created by an attempt to connect to a non-existing database with the name of the marker. So do not worry about the error messages containing the markers.

Following we describe some common installation problems:

- A common install error, especially in a single machine installation, is insufficient disk space. If you install all parts of Tivoli Enterprise Data Warehouse to the same drive and your TEMP directory is on the same drive, you should have 2.0 GB of free space. The reason for so much disk space is that the CD image is copied to TEMP to allow for CD swapping.

- The installation of the warehouse pack fails when the DB2 e-fix has not been installed. In this situation you have to install the DB2 e-fix on the Control server (see Section 3.3.1, “Windows DB2 Universal Database installation” on page 60). Un-install the warehouse pack that failed (see Section 10.3.2, “Un-install the warehouse packs” on page 297). Restore the TWH_MD database from the backup taken during the failed install:

  ```
  db2stop force;
  db2start
  cd <TWH_TOPDIR>\apps_backups\<ava>\install
  db2 restore db twh_md from .
  ```

  Reinstall the warehouse pack that failed.

- The installation of Tivoli Enterprise Data Warehouse might fail with the following message in the TWH.log file:

  ```
  ==>Testing DB2 exec path(F)
  CDWIC0024E Could not execute/locate DB2 command!!!
  ```

  This is because the PATH environment variable has become too long. The PATH environment variable is limited to 2075 characters in length.

- The installation of Presentation Services (PS) locks if ports are already in use. You must specify unused port numbers when you install Tivoli Enterprise Data Warehouse. In particular, if there is already a Web server on the system that you plan to install the Report server on, you must un-install it, disable it, or specify a different port number for the HTTP Server Port for Tivoli Presentation Services.

10.1.2 Troubleshooting the IBM Console and the Report Interface

In this section we discuss problems with the work of the Tivoli Enterprise Data Warehouse Report Interface and the IBM Console. We give useful tips as to where to start the troubleshooting. We also mention problems and known defects.

No connection to the IBM Console

If you have problems openning the IBM Console in your Web browser with the URL `http://`*`hostname:port`*`/IBMConsole`, and check the following:

1. See if the name of the Report server is correct. Try the fully qualified host name. Check the port of your Web server. The default value is port 80 if not changed during installation (see the entry IBM HTTP Server Port in Figure 3-4 on page 65).

2. If everything is correct and you still have no connection to the IBM Console, use the IP address of the Report server instead of its host name. If this works, you probably have a problem with your name resolution. Check NIS and DNS settings (check whether you can resolve the host name using the command **`nslookup hostname`**). Check the /etc/hosts file on UNIX or the C:\WINNT\system32\drivers\etc\hosts file on Windows machines.
3. If OK, check your network connection to the Report server (ping hostname).
4. Check if the Web server is running. Use the above URL without /IBMConsole. You should see a page displaying `Welcome to the IBM HTTP Server`. If not, check if the service Tivoli Presentation Services HTTP Server is started on the Report server. If not, try to start it manually.
5. If it is not possible to start this service, you can try to connect to the administration server (`http://hostname:8008`) and check the Web server configuration. You will probably have to create a user ID for the administration server first. Follow the instructions that are displayed after the log in to the administration server has failed several times.

   Tip: In our testing environment we experienced the problem, that the IP was changed by DHCP but the original IP was still in the web server configuration - do not use DHCP on your Report server.

6. If you can connect to the Web server (with the URL *without* /IBMConsole) but not to the IBM Console (URL *with* IBMConsole), check if the following services are started on the Report server:
   – Server for IBMConsole
   – Web Services for IBMConsole

   See also the log files of these two services which are in the directories PS_install_dir/log/fwp_wc and PS_install_dir/log/fwp_mcr, respectively.
7. If you can connect to the IBM Console from a Web browser running locally on the Report server but not from Web browsers running on (some) remote machines, check the following file on your Report server:

   ```
   PS_install_dir/ibmhttpd/conf/httpd.conf
   ```

   This file contains redirects for your IBM Console login window. If these redirects use the short host name for your Report server, you will have problems if your client cannot correctly resolve this short host name. This problem is not solved when you use the fully qualified host name in your Web browser.

   ```
   # Allow simpler sign-on URL.
   RedirectPermanent /IBMConsole/ http://host:80/servlet/com.tivoli.pf.wc...
   RedirectPermanent /IBMConsole http://host:80/servlet/com.tivoli.pf.wc...
   ```

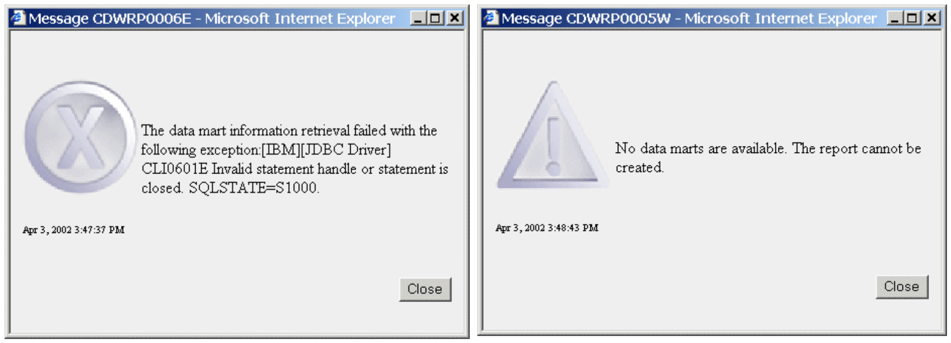
To solve this problem, change the host name to the fully qualified host name in the httpd.conf file and restart the service IBM Presentation Services HTTP Server.

Troubleshooting tips when using the Report Interface

The following are troubleshooting tips for the Report Interface.

1. If you work with the Report Interface, Java Script and style sheets must be enabled in the Web browser.
2. In the upper-right corner of each task panel you find the help button (**?**), which provides detailed information for the task.
3. If you have created an object (for example, a report or a user) and you do not find it in the appropriate list, you can try the following:
   - Click **Refresh**, if available.
   - If you see no objects or old objects only, check if the database service DB2-DB2 on the control server is running.

   If you stopped and restarted the Control server database and you see error messages like Figure 10-1, restart the following services on the Report server:
   - Server for IBMConsole
   - Web Services for IBMConsole


Message CDWRP0006E - Microsoft Internet Explorer
Close
Message CDWRP0005W - Microsoft Internet Explorer
Close

The data mart information retrieval failed with the following exception:[IBM][JDBC Driver] CLI0601E Invalid statement handle or statement is closed. SQLSTATE=S1000.

Apr 3, 2002 3:47:37 PM

No data marts are available. The report cannot be created.

Apr 3, 2002 3:48:43 PM

Figure 10-1 Error messages in the Report Interface after database restart

See also the leadoffs of these two services which are in the directories PS_install_dir/log/fwp_wc and PS_install_dir/log/fwp_mcr, respectively.

Problems creating data marts from customized star schemas

If you do not find your star schema in the Add Star Schemas to a Data Mart dialog, check if the star schema has been created in the Data Warehouse Center. Start the Data Warehouse Center from your DB2 Control Center using **Tools -> Data Warehouse Center**. Expand the tree **Warehouse Schemas** in the left-hand side panel. You will see all available star schemas there. You can create new star schemas in the Data Warehouse Center by right clicking **Warehouse Schemas**.

Problems creating the first report from new data marts

Following is a list of problems that you may run into when creating the first report from new data marts.

1. Check whether your star schema contains all the necessary tables (fact table, metric table, and dimension tables).
2. The Report Interface assumes certain column names in these tables. Check the naming conventions (see “Naming Conventions” in *Enabling an Application for Tivoli Enterprise Data Warehouse,* GC32-0745).
3. The connections between the tables have to be set up correctly in the star schema. The Report Interface uses these connections. They are written to the table rpi.strings in the control server database TWH_MD. They are updated by a trigger when you save the star schema in the Data Warehouse Center. The connections set up in the star schema will result in the where clauses in the SQL statements of your reports (see Example 4-1 on page 107, lines 4-6).
4. Check the SQL output from the reports pop-up menu (Show SQL...) for further hints locating a problem. This might be helpful when no data is found while running the report. Note that you must have sufficient roles to see the SQL output. If you do not have sufficient roles you will not see the Show SQL... entry in the pop-up menu of the reports.

10.1.3 Troubleshooting the customization

In this section we discuss some troubleshooting techniques for customizing. With customizing we refer to the process of creating ETLs and integrating them into Tivoli Enterprise Data Warehouse using the Data Warehouse Center. We give some tips on how to set up the Data Warehouse Center correctly for your own source application. The information given here can also be helpful for troubleshooting errors that occur during and after the installation of data warehouse packs.

Troubleshooting ETLs

The following are ideas for troubleshooting ETLs.

1. The Data Warehouse Center generates log files in the path defined by the DB2 environment variable %VWS_LOGGING%. This variable usually points to <db2dir>\sqllib\logging. In this directory you find the Warehouse Agent log file Agnt<nnnn>.log and the Warehouse Agent environment Agnt<nnnn>.set. Look for the most recent files.
2. When you run processes in the Data Warehouse Center you can see their status in the Work in Progress dialog (you can open this dialog using the Warehouse menu in the Data Warehouse Center). When you encounter errors in the process status you can gain more information by right clicking the failed step and selecting **Show Log**. Look for the first entry with the message Type `Run Time Error`. Right click this message and select **Show Details**.
3. If you have connection problems to remote databases try to connect to the source database using the CLI tools of the database, for example, DB2 CLP, sqlplus for Oracle, or dbaccess for Informix. You can also use the ODBC Data Source panels in Windows to test your database connection.

   You can also use the **`execsql`** command provided by Tivoli Enterprise Data Warehouse to test the database connection:

   ```
   execsql dummy dummy.out <ODBC Datasource name> user pwd
   ```

   If the name of the RDBMS vendor appears in dummy.out then the connect was successful.
4. It is recommended that you use the SQL execution engine execsql and its wrapper script sqlscript.sh provided by Tivoli Enterprise Data Warehouse in your own ETLs. You can get helpful troubleshooting information from the log file written by the **`execsql`** command. You find these log files under %VWS_LOGGING%\<stepname>.log (for example, apf_c05_s010_init.log). These log files show the following information (see also Example 10-1):
   – ODBC data sources used
   – SQL statements executed
   – Rows affected per SQL statement
   – Elapsed time per SQL statement

Example 10-1 An execsql log file example

```
======== Began 2001.12.21 18:58:47.818 ========
========================
= Source Datasource : oracle816b
= Source User Name  : scott
= DB Vendor         : Oracle 8 08.01.0006 Oracle 8.1.6.0.0
= DB Server Name    :
= Target Datasource : oracle816b
= Target User Name  : scott
= DB Vendor         : Oracle 8 08.01.0006 Oracle 8.1.6.0.0
```

```
= DB Server Name    :
= Input File        : e:/TWH/apps/apf/v1/etl/sql/apf_c05_s010_test.oracle
========================
= SOURCE SQL Statement: "insert into tab2 values ('a')"
= Elapsed Time        : 00:00:00.1000
= Rows Modified       : 1
= Successful Execution: No Errors
========================
= SOURCE SQL Statement: "insert into tab2 values ('b')"
= Elapsed Time        : 00:00:00.1000
= Rows Modified       : 1
= Successful Execution: No Errors
========================
======== Completed 2001.12.21 18:58:48.138 ========
```

5. Before attempting to run your custom ETL script from the Data Warehouse Center, you can run the script from the command line to validate the script. Start the bash program, which is installed with Tivoli Enterprise Data Warehouse, and enter the following command:

   ```
   sqlscript.sh product_code script_name source_db source_uid source_pwd
   target_db target_uid target_pw
   ```

   Where script_name is the name of your custom script.

6. If you get any errors from logging onto the Data Warehouse Center, check that the control database is set to TWH_MD. To do this click **Advanced** on the Data Warehouse Control logon panel. Make also sure that you set up the control database correctly in the Warehouse Control Database Management in the **Start -> Programs -> DB2** menu. Check the ODBC Data Source of the control database.

7. If you see errors in the Data Warehouse Center after a database restart, restart the vwserver and vwlogger services.

8. For Windows NT and Windows 2000, the vwserver and vwlogger services do not log on as the DB2 user, which causes ETL processes to fail.

   Workaround for Windows NT:

   a. Open the Services window.
   b. Select **Warehouse logger**.
   c. Select the **Startup** button.
   d. Click **This Account**.
   e. Type the DB2 user ID.
   f. Type the DB2 password in the Password field.
   g. Type the DB2 password in the Confirm Password field.
   h. Click **OK**.
   i. Repeat step a through step h for the Warehouse Server.
   j. Stop and then restart the vwserver and vwlogger services.

Workaround for Windows 2000:

a. Open the Services window.
b. Select **Warehouse logger -> Action -> Properties**.
c. Click the **Log On** tab.
d. Click **This account**.
e. Type the DB2 user ID.
f. Type the DB2 password in the Password field.
g. Type the DB2 password in the Confirm Password field.
h. Click **OK**.
i. Repeat step a through step h for the Warehouse Server.
j. Stop and then restart the vwserver and vwlogger services.

10.2 Maintenance and backup

The following are some issues to consider when maintaining and backing up Tivoli Enterprise Data Warehouse databases.

10.2.1 Removing old data from the Data Warehouse Center logs

You should regularly delete information in the Data Warehouse Center log files, IWH*.log, located in the directory specified by the %VWS_LOGGING% environment variable. These log files grow rapidly. You can only delete information from these files when the Data Warehouse Center services Warehouse Server and Warehouse Logger are stopped. You can do this in a script, as in Example 10-2.

Example 10-2 Purging script

```
net stop vwkernel
net stop vwlogger
<edit the necessary files >
net start vwlogger
net start vwkernel
```

10.2.2 Removing old data from the central data warehouse

You can control how often data is removed, or pruned, from the central data warehouse using a combination of triggers and warehouse processes. This is done by completing these tasks:

1. Scheduling the pruning process
2. Specifying the data to be pruned

For more information on data pruning processes and the database tables used, see *Enabling an Application for Tivoli Enterprise Data Warehouse,* GC32-0745.

Pruning processes for warehouse packs are defined by each application. See the documentation provided with each warehouse pack for information about pruning the data for that application.

10.2.3 Maintaining the warehouse database

This chapter describes how to use warehouse programs to maintain your warehouse database.

Reorganizing the data

You can use the DB2 reorganize warehouse program to rearrange a table in physical storage. This eliminates fragmentation and ensures that the table is stored efficiently in the database. You can also use reorganization to control the order in which the rows of a table are stored, usually according to an index.

You can define reorganization steps in a Data Warehouse process (see Figure 10-2 on page 293).

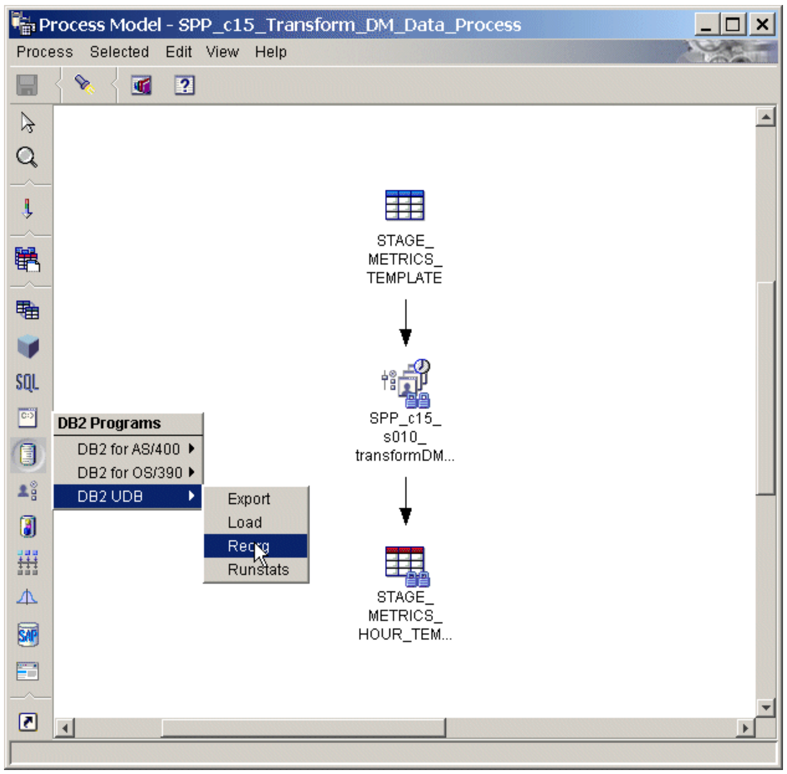


Figure 10-2 Create a reorganization step

You can use a warehouse source or target as a source for this step subtype. The REORG program writes to the source table.

To define values for a step that runs a DB2 UDB REORG warehouse program:

1. Open the Steps Property notebook.
2. Specify general information about the program.
3. *Optional:* On the Parameters page, specify information for the REORG step:
   - In the Using temporary table space field, type the name of the temporary table space that should be used during the REORG step.
   - In the Using index field, type the name of the index that should be used during the REORG step.
4. On the Processing Options page, provide information about how your step processes.
5. Click **OK** to save your changes and close the Step Property notebook.

Updating system catalog statistics

You can use the DB2 RUNSTATS warehouse program to gather statistics about the physical and logical characteristics of a table and its indexes. DB2 Universal Database uses these statistics to determine the best way to access your data.

You can use the DB2 UDB RUNSTATS warehouse program to create a step that can be used to update system catalog statistics on the data in a table, the data in the table indexes, or the data in both the table and its indexes. The optimizer uses these statistics to choose which path will be used to access the data. In general, you need to update statistics if there are extensive changes to the data in the table.

Create a RUNSTATS step in a process (see Figure 10-2 on page 293). The RUNSTATS program uses a warehouse target as a source and a target. Link a warehouse target to the step in the Process Model window before you define the values for the step. To define values for a step that runs a DB2 UDB RUNSTATS warehouse program:

1. Open the Steps Property notebook.
2. Specify the general information about the warehouse program.
3. *Optional:* On the Parameters page, specify information for the RUNSTATS warehouse program:
   a. Specify the level of statistics you want to gather for the table by clicking a radio button under Statistics for the table.
   b. Specify the level of statistics you want to gather for the table's indexes by selecting a radio button under Statistics for the indexes.
   c. Use the Share level radio buttons to specify the type of access you want other users to have to the table while the statistics are being gathered.

4. On the Processing Options page, provide information about how your step processes.
5. Click **OK** to save your changes and close the Step Properties notebook.

10.2.4 Backup

This section provides information about backing up and restoring the Tivoli Enterprise Data Warehouse databases. When planning back-up operations or performing restore operations, you must consider the relationships between the data in Tivoli Enterprise Data Warehouse databases. Some examples follow:

- If an older version of the control database is restored with a newer version of the central data warehouse database, log messages and ETL run status generated by the Data Warehouse Center are lost and will not match the state of the central data warehouse database.
- If a data mart database is completely lost and no backup exists, you might be able to recreate the data mart database from data in the central data warehouse. To recreate the data mart database, you must adjust the extract control information for the star schemas in the data mart database located in the central data warehouse, and then run the data mart ETL processes for each star schema in the lost data mart database. It is not possible to recreate a data mart database from data that has been pruned from the central data warehouse. Note that the extract control parameters in the database must be manually changed to ensure that all data is restored.
- If an old copy of the central data warehouse database is restored along with newer copies of the control database and newer copies of data mart databases, manual adjustment of extract control tables might be required to pull additional data from source applications to bring the central data warehouse database up-to-date. In some cases, source data might have been pruned after data was populated into the central data warehouse database, making it impossible to recover all of the data. Some data mart ETL processes might encounter problems as they attempt to reinsert records from the recovered central data warehouse database into more recent copies of data mart databases. In these cases, manual intervention by a database administrator might be required to fully recover the system.
- User definitions for the Report Interface are stored in the Tivoli Presentation Services directory. When the users are assigned to user groups in the Report Interface, a subset of user information is stored in the Tivoli Enterprise Data Warehouse control database.

10.3 Un-install components

Here we give the basic steps for the un-installation of the Tivoli Enterprise Data Warehouse core product and the Tivoli Enterprise Data Warehouse application packs. You can find more detailed information and troubleshooting hints for un-installation in *Installing and Configuring Tivoli Enterprise Data Warehouse*, GC 32-0744.

10.3.1 Un-install Tivoli Enterprise Data Warehouse core product

Tivoli Enterprise Data Warehouse might be installed on one machine or distributed on up to four machines. However, if you start the un-install process on one machine, you have to un-install all components on this machine. There is no option to select components to un-install.

If there is one component per machine, removing a component will allow a reinstall of the same components on that machine. However, if you remove the Control server then all other components on all other machines must be removed.

If you un-install a distributed environment, make sure that you un-install the Control server last. Otherwise you might not be able to un-install the remaining Tivoli Enterprise Data Warehouse components.

During the Tivoli Enterprise Data Warehouse un-installation process, all Tivoli Enterprise Data Warehouse-related databases will be dropped. This will fail if databases are locked by other processes. To make sure that the databases can be dropped successfully, you should stop and start the database before starting the uninstallation:

```
db2 stop force
db2 start.
```

To start the Tivoli Enterprise Data Warehouse un-installation process on a Windows machine type:

```
%TWH_TOPDIR%\uninstall\uninstall.exe
```

Or on a UNIX machine:

```
$TWH_TOPDIR/uninstall/uninstall.bin.
```

See %TWH_TOPDIR%/TWHUninstall.log for troubleshooting.

The Tivoli Enterprise Data Warehouse uninstaller does not un-install Presentation Services (PS), but merely removes the Tivoli Enterprise Data Warehouse components from PS. This should allow you to reinstall Tivoli Enterprise Data Warehouse. If it does not work, you have to manually un-install PS.

If required, use the PS uninstaller %PS_TOPDIR%\uninstall.bat to un-install PS. If this fails, you can manually un-install PS. Use regedt32 to remove the following NT services from HKLM\System\CurrentControlSet\Services:

- ps_mcr
- ps_wc
- TivoliPresentationServicesHTTPAdministration
- TivoliPresentationServicesHTTPServer

After this step you have to reboot your machine. Then remove the PS and TWH installation directories, if they exist. You have to edit the file vpd.properties, which is located in the %WINDIR% directory on Windows machines, in /usr/lib/objrepos on AIX machines, and in /root on Linux machines. This file tracks all products installed using InstallShield. Remove all lines beginning with Tivoli Enterprise Data Warehouse or with Tivoli_Enterprise_Data_Warehouse and entries from the Presentation Service.

The next step is to drop the DB2 databases TWH_MD, TWH_CDW and TWH_MART:

```
db2stop force
db2 drop db <dbname>.
```

If you get the error `database not found`, then catalog the databases and then drop them.

```
db2 catalog db <dbname>
```

10.3.2 Un-install the warehouse packs

Before you start the un-istallation you should do the same back ups as for the installation of warehouse packs. Warehouse packs can be un-installed with the command line utility:

```
<TWH_TOPDIR>\install\bin\twh_app_deinstall.sh.
```

Before you can un-install a warehouse pack you have to edit the configuration file `twh_app_deinstall.cfg` in the same directory. Follow the instructions in this file. You have to insert the product code and the DB2 password. When ready, run:

```
twh_app_deinstall.sh -c twh_app_deinstall.cfg
```

If you are un-installing a warehouse pack to recover from a failed installation you may be prompted to restore a backup of the control database. If so, you must do this restore before trying to run the install again.

A

GEOAREA and TMZON tables

This appendix provides views and descriptions of the GEOAREA and TMZON tables in the TWH_CDW database.

Table TMZON

Use Table A-1 to select the correct GEOAREA_CD that the customer data is being imported from.

Table A-1 Table TMZON

GEOAREA_CD	GEOAREA_PARENT_CD	GEOTYP_ID	TMZON_ID	GEOAREA_NM
DEF		2		Default GeoArea
NA		2		North America
US		4		United States
US-N		3		US North
US-S		3		US South
US-W		3		US West
CAN		4		Canada
LA		2		Latin America
EMEA		1		Europe, Middle East & Africa
EUR		2		Europe
ME		2		Middle East
AFR		2		Africa
AP		2		Asia Pacific
AP-N		3		Asia Pacific - North
AP-S		3		Asia Pacific - South

Table GEOAREA

Use Table A-2 on page 301 to select the TMZON that the customer data is being imported from.

Table A-2 Table GEOAREA

TMZON_ID	TMZON_NM	TMZON_GMT_OFFSET	TMZON_CDW_DIFF
1	Eniwetok, Kwajalein	-120000	60000
2	Midway Island, Samoa	-110000	50000
3	Hawaii	-100000	40000
4	Alaska	-90000	30000
5	Pacific Time (US & Canada); Tijuana	-80000	20000
6	Arizona	-70000	10000
7	Mountain Time (US & Canada)	-70000	10000
8	Central Time (US & Canada)	-60000	0
9	Mexico City, Tegucigalpa	-60000	0
10	Saskatchewan	-60000	0
11	Bogota, Lima, Quito	-50000	-10000
12	Eastern Time (US & Canada)	-50000	-10000
13	Indiana (East)	-50000	-10000
14	Atlantic time (Canada)	-40000	-20000
15	Caracas, La Paz	-40000	-20000
16	Newfoundland Standard Time	-33000	-27000
17	Brasilia	-30000	-30000
18	Buenos Aires, Georgetown	-30000	-30000
19	Mid-Atlantic	-20000	-40000
20	Azores, Cape Verde Is.	-10000	-50000
21	Casablanca, Monrovia	0	-60000
22	Grenwich Mean Time: Dublin, Edinburg, Lisbon, London	0	-60000
23	Amsterdam, Copenhagen, Madrid, Paris, Vilnius	10000	-70000
24	Belgrade, Sarajevo, Skopje, Sofija, Zagreb	10000	-70000
25	Brussels, Berlin, Bern, Rome, Stockholm, Vienna	10000	-70000
26	Athens, Istanbul, Minsk	20000	-80000
27	Bucharest	20000	-80000
28	Cairo	20000	-80000
29	Harare, Pretoria	20000	-80000
30	Helsinki, Riga, Tallinn	20000	-80000
31	Baghdad, Kuwait, Riyadh	30000	-90000
32	Moscow, St. Petersburg, Volgograd	30000	-90000
33	Tehran	33000	-93000
34	Abu Dhabi, Muscat	40000	-100000
35	Baku, Tbilisi	40000	-100000
36	Kabul	43000	-103000
37	Islamabad, Karachi, Tashkent	50000	-110000
38	Bombay, Calcutta, Madras, New Delhi	53000	-113000
39	Almaty, Dhaka	60000	-120000
40	Colombo	60000	-120000
41	Bangkok, Hanoi, Jakarta	70000	-130000
42	Beijing, Chongqing, Hong Kong, Urumqi	80000	-140000
43	Perth	80000	-140000
44	Singapore	80000	-140000
45	Taipei	80000	-140000
46	Osaka, Sapporo, Tokyo	90000	-150000
47	Seol	90000	-150000
48	Yakutsk	90000	-150000
49	Adelaide	93000	-153000
50	Darwin	93000	-153000
51	Brisbane	100000	-160000
52	Canberra, Melbourne, Sydney	100000	-160000
53	Guam, Port Moresby	100000	-160000
54	Hobart	100000	-160000
55	Vladivostok	100000	-160000
56	Magadan, Solomon Is., New Caledonia	110000	-170000
57	Auckland, Wellington	120000	-180000
58	Fiji, Kamchatka, Marshall Is.	120000	-180000

B

Scripts

This appendix contains scripts mentioned in this redbook. All these scripts can also be downloaded from the Redbooks Web site, as well. For downloading insctructions, please refer to Appendix C, “Additional material” on page 365.

dmn_c10_s010_tiv_loadDMData.db2

Example B-1 is the edited script for the extract of Tivoli Distributing Monitoring Version 3.7 data for the multiple customer environment. The original script name, which is used for Tivoli Distributing Monitoring Version 3.7 in a single customer environment, is dmn_c05_s010_initDMdata. We changed this script for the multiple customer environment. The changes are highlighted in boldface.

Example: B-1   dmn_c10_s010_tiv_loadDMData.db2 script

```
*************Begin Copyright - Do not add comments here********************
*
* Licensed Materials - Property of IBM
* 5724-C40
* (C) Copyright IBM Corp. 2002.  All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*
*************End Copyright*************************************************

-------------------------------------------------------------------------------
----
Here we perform an incremental extract using date/time to control the
extraction
window because we could be interested in old DM data; i.e. to DM data
retrieved
before the TWH  patch that  introduce the  raw/integer  sequence numbers to
keep
track of every record added to the DM_METRICS table.
The source table is DM_METRICS and is located on an DB2 RIM Database
(ODBC Src=db2_rim); the target table is DMN.STAGE_DM_METRICS and is located on
a
target DB2 Database (ODBC Tgt=twh_cdw).
-------------------------------------------------------------------------------
----

--
//////////////// here we define the oldest dt_stamp to be considered during the
//////////////// fist time ETL as the current timestamp minus the number of
days
//////////////// specified in the PMSMTC_AGE_IN_DAYS field of the
Prune_Msmt_Control
//////////////// table for the DMN application.
//////////////// You have to update the Prune_Msmt_Control table before running
//////////////// this initial ETL; e.g. if you are interested into the latest
//////////////// 2 months and 8 days of DM Data, the SQL statement is:
////////////////
```

```
////////////////     UPDATE TWG.Prune_Msmt_Control
////////////////        SET PMSMTC_AGE_IN_DAYS = 208
////////////////      WHERE TMSUM_CD = 'H'
////////////////        AND MSRC_CD  = 'DMN'
////////////////     ;
////////////////
//////////////// because PMSMTC_AGE_IN_DAYS is a DECIMAL(8,0) field that holds
a
//////////////// date duration in the format yyyymmdd (208 -> yyyy=0 years,
mm=2
//////////////// months, dd=8 days).
//////////////// The initial update of the Extract_Control table at the CDW
database
//////////////// is performed only if the extraction window is open, i.e. if we
are
//////////////// running the initial ETL for the first time; this condition is
intended
//////////////// in order to prevent multiple runs of the initialization
script, that
//////////////// may lead to data integrity violation in the TWG.Msmt table.
--

--#EXECUTE_AT_TARGET
UPDATE TWG.Extract_Control
  SET extctl_from_intseq = (
      SELECT (DAYS(current date - C.PMSMTC_AGE_IN_DAYS ) - DAYS('1970-01-01'))
* 86400
        FROM TWG.PRUNE_MSMT_CONTROL C
       WHERE TMSUM_CD = 'H' AND
             MSRC_CD  = 'DMN' )
WHERE
  extctl_source='DM_METRICS_INIT_TIV' AND
  extctl_target='DMN.STAGE_DM_METRICS' AND
   extctl_from_intseq <> extctl_to_intseq
;

--
//////////////// drop temporary table to hold the extract window
--
//////////////// at: db2_rim
--
--#IGNORE_ERROR
DROP TABLE DMN.temp_extract_control
;
--
//////////////// create temporary table to hold the extract window
--
//////////////// at: db2_rim
--
```

```
CREATE TABLE DMN.temp_extract_control (
  extctl_from_intseq   INTEGER NOT NULL,
  extctl_to_intseq     INTEGER NOT NULL )
;
--
//////////////// get the from_intseq from the Extract_Control in the CDW
--
//////////////// from: twh_cdw to: db2_rim
--
--#INSERT_INTO_SOURCE
INSERT INTO DMN.temp_extract_control
SELECT
  INT(extctl_from_intseq),
  INT(2000000000)
FROM
  TWG.Extract_Control
WHERE
  extctl_source='DM_METRICS_INIT_TIV' AND
  extctl_target='DMN.STAGE_DM_METRICS'
;
//////////////// update extctl_to_intseq with the max of dt_stamp
--
//////////////// at: db2_rim
--
UPDATE DMN.temp_extract_control
  SET extctl_to_intseq = (SELECT MAX(dt_stamp) FROM DM_METRICS)
;
//////////////// drop the staging table in twh_cdw (this is faster then
deleting all records)
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
--#IGNORE_ERROR
DROP TABLE DMN.stage_dm_metrics
;
//////////////// create the stage table in twh_cdw according to the predefined
template
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
CREATE TABLE DMN.stage_dm_metrics LIKE DMN.stage_dm_metrics_template
;
//////////////// Dropping the table DMN.stage_dm_metrics makes inoperative the
//////////////// trigger responsible to insert the metric_id unique sequence
number.
//////////////// Issuing a CREATE TRIGGER statement with the same trigger-name
//////////////// as the inoperative trigger will cause that inoperative trigger
```

```
//////////////// to be replaced with a warning (SQLSTATE 01595), so we need to
//////////////// instruct the execsql to IGNORE_ERROR.
--
--#EXECUTE_AT_TARGET
--#IGNORE_ERROR
CREATE TRIGGER DMN.metric_id_trig NO CASCADE
  BEFORE INSERT ON DMN.stage_dm_metrics
  REFERENCING  NEW AS N
  FOR EACH ROW
  MODE DB2SQL
  SET N.metric_id = NEXTVAL FOR DMN.metric_id_seq
;
--
//////////////// perform the extraction from the DM_METRICS table to the
staging
//////////////// table DMN.stage_dm_metrics in the CDW database.
--
//////////////// from: db2_rim to: twh_cdw
--
//////////////// NOTE: the constant metric_id value selected from the
DM_METRICS table
//////////////// is always overridden by the unique sequence number provided by
the
//////////////// before insert trigger DMN.metric_id_trig
--
//////////////// The initial ETL process assumes the local GMT offset;
//////////////// if you want to change it, you can change the last constant
//////////////// field retrieved by the following select to a more suitable
value
//////////////// to be chosen in the range [-720,+720].
--
--#INSERT_INTO_TARGET
INSERT INTO DMN.stage_dm_metrics
SELECT
  0 AS metric_id,
  D.COLLECTION_DATE, D.HOSTNAME, D.ENDPOINT,
   D.PROFILE_COLLECTION, D.PROBE_COLLECTION,
   D.PROBE, D.PROBE_DESC, D.PROBE_ARG,
   D.MIN_VALUE_00, D.MAX_VALUE_00, D.AVG_VALUE_00,
   D.MIN_VALUE_01, D.MAX_VALUE_01, D.AVG_VALUE_01,
   D.MIN_VALUE_02, D.MAX_VALUE_02, D.AVG_VALUE_02,
   D.MIN_VALUE_03, D.MAX_VALUE_03, D.AVG_VALUE_03,
   D.MIN_VALUE_04, D.MAX_VALUE_04, D.AVG_VALUE_04,
   D.MIN_VALUE_05, D.MAX_VALUE_05, D.AVG_VALUE_05,
   D.MIN_VALUE_06, D.MAX_VALUE_06, D.AVG_VALUE_06,
   D.MIN_VALUE_07, D.MAX_VALUE_07, D.AVG_VALUE_07,
   D.MIN_VALUE_08, D.MAX_VALUE_08, D.AVG_VALUE_08,
   D.MIN_VALUE_09, D.MAX_VALUE_09, D.AVG_VALUE_09,
   D.MIN_VALUE_10, D.MAX_VALUE_10, D.AVG_VALUE_10,
    D.MIN_VALUE_11, D.MAX_VALUE_11, D.AVG_VALUE_11,
    D.MIN_VALUE_12, D.MAX_VALUE_12, D.AVG_VALUE_12,
    D.MIN_VALUE_13, D.MAX_VALUE_13, D.AVG_VALUE_13,
    D.MIN_VALUE_14, D.MAX_VALUE_14, D.AVG_VALUE_14,
    D.MIN_VALUE_15, D.MAX_VALUE_15, D.AVG_VALUE_15,
    D.MIN_VALUE_16, D.MAX_VALUE_16, D.AVG_VALUE_16,
    D.MIN_VALUE_17, D.MAX_VALUE_17, D.AVG_VALUE_17,
    D.MIN_VALUE_18, D.MAX_VALUE_18, D.AVG_VALUE_18,
    D.MIN_VALUE_19, D.MAX_VALUE_19, D.AVG_VALUE_19,
    D.MIN_VALUE_20, D.MAX_VALUE_20, D.AVG_VALUE_20,
    D.MIN_VALUE_21, D.MAX_VALUE_21, D.AVG_VALUE_21,
    D.MIN_VALUE_22, D.MAX_VALUE_22, D.AVG_VALUE_22,
    D.MIN_VALUE_23, D.MAX_VALUE_23, D.AVG_VALUE_23,
  D.MIN_DAILY_VALUE, D.MAX_DAILY_VALUE, D.AVG_DAILY_VALUE,
  D.MIN_HOURLY_AVG, D.MAX_HOURLY_AVG,
   (current timezone * 60 / 10000)
FROM
  DM_METRICS D,
  DMN.temp_extract_control ec
WHERE
  D.DT_STAMP > ec.extctl_from_intseq AND
  D.DT_STAMP <= ec.extctl_to_intseq
;
--
//////////////// sample check for invalid data if multi customer support is
implemented
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
INSERT INTO DMN.Invalid_Data
SELECT *
  FROM DMN.stage_dm_metrics N
 WHERE NOT EXISTS ( SELECT 1
                      FROM DMN.CUST_LOOKUP CUST_LOOKUP
                     WHERE CUST_LOOKUP.VALUE = N.HOSTNAME OR
                           CUST_LOOKUP.VALUE = '@' )
       OR
       NOT EXISTS ( SELECT 1
                      FROM DMN.CENTR_LOOKUP CENTR_LOOKUP
                     WHERE CENTR_LOOKUP.VALUE = N.HOSTNAME OR
                           CENTR_LOOKUP.VALUE = '@' )
;
--
//////////////// drop temp_to_intseq temporary table in CDW
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
```

```
--#IGNORE_ERROR
DROP TABLE DMN.temp_to_intseq
;
//////////////// create temp_to_intseq temporary table in CDW
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
CREATE TABLE DMN.temp_to_intseq (
  extctl_to_intseq   INTEGER NOT NULL )
;
//////////////// copy extctl_to_intseq from temp_extract_control table
//////////////// temp_to_intseq temporary table in the CDW
--
//////////////// from: db2_rim to: twh_cdw
--
--#INSERT_INTO_TARGET
INSERT INTO DMN.temp_to_intseq
  SELECT extctl_to_intseq FROM DMN.temp_extract_control
;
//////////////// update extract control with the extctl_to_intseq value
//////////////// retrieved from the temporary table temp_to_intseq
--
//////////////// at: twh_cdw
--
--#EXECUTE_AT_TARGET
UPDATE TWG.Extract_Control
  SET extctl_to_intseq = (SELECT extctl_to_intseq FROM DMN.temp_to_intseq)
WHERE
  extctl_source='DM_METRICS_INIT_TIV' AND
  extctl_target='DMN.STAGE_DM_METRICS'
;
//////////////// write extract info to extract log table; this fires the
trigger
//////////////// TWG.extract_ctl_upd that updates the extract control table to
//////////////// close the window
--
//////////////// at: twh_cdw

--#EXECUTE_AT_TARGET
INSERT INTO TWG.Extract_Log
SELECT
  extctl_source, extctl_target,
  current timestamp - current timezone,
  extctl_from_rawseq, extctl_to_rawseq,
  extctl_from_intseq, extctl_to_intseq,
  extctl_from_dttm, extctl_to_dttm
FROM
  TWG.Extract_Control ec
```

```
WHERE
  ec.extctl_source='DM_METRICS_INIT_TIV' AND
  ec.extctl_target='DMN.STAGE_DM_METRICS' ;
```

Insert_cust_control.db2

Example B-2 is an SQL script for inserting an extract control row into the TWD_CDW database for a customer.

Example: B-2 Insert_cust_control.db2 script

```
connect to TWH_CDW user db2inst1

insert into twg.Extract_Control values (    'DM_METRICS_INIT_TIV' ,
'DMN.STAGE_DM_METRICS' ,    x'00000000000000000000',
x'99999999999999999999',   0,   2000000000,   '1970-01-01-00.00.00.000000',
'9999-01-01-00.00.00.000000')

insert into twg.Extract_Control values (    'DM_METRICS_TIV' ,
'DMN.STAGE_DM_METRICS' ,    x'00000000000000000000',
x'99999999999999999999',   0,   2000000000,   '1970-01-01-00.00.00.000000',
'9999-01-01-00.00.00.000000')
```

srm_c05_s010_extractInvData

The script in Example B-3 was written for case study 1 ETL1 process. It is the first script in the process to run. Its responsibility is to make sure that all the components are available in the CDW comp table. It moves data into several stage tables in the CDW from the legacy source application data. It then moves data into the comp tables. Then it uses several joins and unions to determine which data to place into the compattr tables. Also it makes use of the invalid_data tables to ensure that invalid data is captured.

Example: B-3 srm_c05_s010_extractInvData script

```
--#EXECUTE_AT_TARGET
drop table inv.stage_ipnetwork_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_ipnetwork_copy like inv.stage_ipnetwork;
--#INSERT_INTO_TARGET
insert into inv.stage_ipnetwork_copy
SELECT
   'sysid_' CONCAT SUBSTR(perform.server.display_name,1,31) AS HWARE_SYS_ID,
   SUBSTR(perform.server.fqhn,1,31) AS NET_NODE_NAME,
   perform.server.primary_ip_addr AS NET_NODE_ADDR
FROM
   PERFORM.SERVER
WHERE
   PERFORM.SERVER.DISPLAY_NAME LIKE 'srm%'
ORDER BY
   HWARE_SYS_ID,
   NET_NODE_NAME,
   NET_NODE_ADDR
;

--#EXECUTE_AT_TARGET
drop table inv.stage_ostype_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_ostype_copy like inv.stage_ostype;
--#INSERT_INTO_TARGET
insert into inv.stage_ostype_copy
SELECT
   'sysid_' CONCAT SUBSTR(perform.server.display_name,1,31) AS HWARE_SYS_ID,
   perform.server.op_sys AS BOOTED_OS_NAME,
   'AIX' AS BOOTED_OS_VER,
   'v1.x' AS COMPUTER_KRNL_VER,
   'Service Pack X' AS NT_SVC_PACK
FROM
   PERFORM.SERVER
WHERE
   PERFORM.SERVER.DISPLAY_NAME LIKE 'srm%'
;

--#EXECUTE_AT_TARGET
drop table inv.stage_processor_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_processor_copy like inv.stage_processor;
--#INSERT_INTO_TARGET
insert into inv.stage_processor_copy
SELECT
   'sysid_' CONCAT SUBSTR(perform.server.display_name,1,31) AS HWARE_SYS_ID,
   perform.server.processors as NUM_PROCESSORS,
   'Model X' AS PROCESSOR_MODEL,
   '1000' AS PROCESSOR_SPEED
FROM
   PERFORM.SERVER
WHERE
   PERFORM.SERVER.DISPLAY_NAME LIKE 'srm%'
ORDER BY
   HWARE_SYS_ID
;

--#EXECUTE_AT_TARGET
```

```
drop table inv.stage_memory_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_memory_copy like inv.stage_memory;
--#INSERT_INTO_TARGET
insert into inv.stage_memory_copy
SELECT
   'sysid_' CONCAT SUBSTR(perform.server.display_name,1,31) AS HWARE_SYS_ID,
   perform.server.inst_mem AS PHYSICAL_MEM_KB
FROM
   PERFORM.SERVER
WHERE
   PERFORM.SERVER.DISPLAY_NAME LIKE 'srm%'
;

--#EXECUTE_AT_TARGET
drop table inv.stage_net_node_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_net_node_copy like inv.stage_net_node;
--#EXECUTE_AT_TARGET
insert into inv.stage_net_node_copy
select
  value(lower(T2.NET_NODE_NAME),'') as net_node_name,
  value(T2.NET_NODE_ADDR,'') as net_node_addr,
  T1.HWARE_SYS_ID,
  value(T1.BOOTED_OS_NAME,'') as booted_os_name,
  'v1.1' as booted_os_ver,
  'Service Pack Y' as os_sub_ver,
  value(T4.PHYSICAL_MEM_KB,-1) as physical_mem_kb,
  value(T6.NUM_PROCESSORS,-1) as num_processors,
  value(T6.PROCESSOR_MODEL,'') as processor_model,
  value(T6.PROCESSOR_SPEED,-1) as processor_speed,
  'Unknown' as system_purpose
FROM
  inv.stage_OSTYPE_copy T1       LEFT OUTER JOIN
  inv.stage_IPNETWORK_copy  T2
      ON     T1.HWARE_SYS_ID = T2.HWARE_SYS_ID,
  inv.stage_OSTYPE_copy T3       LEFT OUTER JOIN
  inv.stage_MEMORY_copy T4
      ON     T3.HWARE_SYS_ID = T4.HWARE_SYS_ID,
  inv.stage_OSTYPE_copy T5       LEFT OUTER JOIN
  inv.stage_PROCESSOR_copy T6
      ON     T5.HWARE_SYS_ID = T6.HWARE_SYS_ID
WHERE
  T1.HWARE_SYS_ID=T3.HWARE_SYS_ID      AND
  T1.HWARE_SYS_ID=T5.HWARE_SYS_ID
;


--#EXECUTE_AT_TARGET
```

```
--sample check for invalid data
insert into inv.invalid_data
select * from inv.stage_net_node_copy n
where
  not exists(
    select 1 from inv.cust_lookup
    where inv.cust_lookup.value = n.net_node_name OR
          inv.cust_lookup.value = '@') OR
  not exists(
    select 1 from inv.centr_lookup
    where inv.centr_lookup.value = n.net_node_name OR
          inv.centr_lookup.value = '@')
;

--Note that the insert of IP_HOST may be changed to SPP_HOST by a trigger
--  if the hostname is not fully qualified.
--#EXECUTE_AT_TARGET
INSERT INTO TWG.COMP(COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
select
 0 as Comp_Id,
 'IP_HOST' as CompTyp_Cd ,
 centr.Centr_Cd as Centr_Cd,
 c.Cust_ID as Cust_ID,
 n.net_node_name as Comp_Nm,
 current timestamp - current timezone as Comp_Strt_DtTm ,
 '9999-01-01-00.00.00.000000' as Comp_End_DtTm,
 '' as Comp_Ds
from
 inv.stage_net_node_copy n,
 inv.cust_lookup cust,
 inv.centr_lookup centr,
 twg.cust c
where
 n.net_node_name != '' and
 not exists (
  select 1
  from
   twg.comp c
  where
   c.comp_nm=n.net_node_name
   and c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
   and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm ) AND
 ((cust.value = n.net_node_name and cust.cust_acct_cd = c.cust_acct_cd)
     OR cust.value = '@') AND
 (centr.value = n.net_node_name OR centr.value = '@')
;
```

```
--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id,
 'INOSNM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.booted_os_name as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.booted_os_name != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.booted_os_name
   and ca.AttrTyp_Cd='INOSNM'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INOSVR' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.booted_os_ver as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.booted_os_ver != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.booted_os_ver
   and ca.AttrTyp_Cd='INOSVR'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INOSSV' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.os_sub_ver as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.os_sub_ver != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.os_sub_ver
   and ca.AttrTyp_Cd='INOSSV'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id as comp_id,
 'INADDR' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.net_node_addr as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.net_node_addr != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.net_node_addr
   and ca.AttrTyp_Cd='INADDR'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id,
 'INPMEM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.physical_mem_kb) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.physical_mem_kb != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.physical_mem_kb)
   and ca.AttrTyp_Cd='INPMEM'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
```

```
 0 as compattr_id,
 c.comp_id,
 'INCPUM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.processor_model as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.processor_model != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.processor_model
   and ca.AttrTyp_Cd='INCPUM'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INNCPU' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.num_processors) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.num_processors != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.num_processors)
   and ca.AttrTyp_Cd='INNCPU'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INCPUS' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.processor_speed) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.processor_speed != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.processor_speed)
   and ca.AttrTyp_Cd='INCPUS'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;
```

srm_c15_s010_transformDMData

Example B-4 is the second script to run for case study 1. It queries data from the source application, then changes the format of the data so that it will map to the CDW tables and CDW granularity. It does this by loading data into the CDW staging tables; no CDW tables are updated.

Example: B-4 srm_c15_s010_transformDMData script

```
-- Now that we have gotten the new data from the source DB, transform it to
hourly data.

--#EXECUTE_AT_TARGET
drop table spp.stage_srm_r_hour;
```

```
--#EXECUTE_AT_TARGET
create table spp.stage_srm_r_hour like spp.stage_srm_r_hour_template;

--#INSERT_INTO_TARGET
insert into spp.stage_srm_r_hour
SELECT *
  FROM inv.srm_r_hour;

--#EXECUTE_AT_TARGET
--#IGNORE_ERROR
drop table spp.stage_metrics_hour;
--#EXECUTE_AT_TARGET
create table spp.stage_metrics_hour like spp.stage_metrics_hour_template;

--#EXECUTE_AT_TARGET
--#IGNORE_ERROR
drop trigger spp.stage_hour_id_trig;
--#EXECUTE_AT_TARGET
CREATE TRIGGER spp.stage_hour_id_trig
NO CASCADE
BEFORE INSERT ON spp.stage_metrics_hour
REFERENCING NEW AS N
FOR EACH ROW
MODE DB2SQL
SET N.metric_id = NEXTVAL for spp.stage_metrics_hour_id_seq;

-- CPU Busy
-- Note: we're using the timezone of the center where the data was collected
--       to calculte the GMT time.
--#EXECUTE_AT_TARGET
insert into spp.stage_metrics_hour
select
 0 as metric_id, current timestamp - current timezone as insert_dttm,
 DATE(TSTAMP), time(tstamp) - t.TmZon_GMT_Offset, lower(fqhn),
SUBSTR(fqhn,1,10), 'SPR_UnixProfile',
 'Unix_Sentry', 'avgcpubusy', 'none', '5',
 cpu_busy * 0.7 as Msmt_Min_Val,
 cpu_busy * 1.1 as Msmt_Max_Val,
 cpu_busy as Msmt_Avg_Val,
 0 as Msmt_Tot_Val, 0 as Msmt_Smpl_Cnt
from
 spp.stage_srm_r_hour dm,
 twg.tmzon t,
 spp.centr_lookup c,
 twg.centr ce
where
 dm.fqhn = c.value and
 c.centr_cd = ce.centr_cd and
 ce.tmzon_id = t.tmzon_id
```

```
;

-- CPU System
-- Note: we're using the timezone of the center where the data was collected
--       to calculte the GMT time.
--#EXECUTE_AT_TARGET
insert into spp.stage_metrics_hour
select
 0 as metric_id, current timestamp - current timezone as insert_dttm,
 DATE(TSTAMP), time(tstamp) - t.TmZon_GMT_Offset, lower(fqhn),
SUBSTR(fqhn,1,10), 'SPR_UnixProfile',
 'Unix_Sentry', 'avgcpusys', 'none', '5',
 cpu_s * 0.7 as Msmt_Min_Val,
 cpu_s * 1.1 as Msmt_Max_Val,
 cpu_s as Msmt_Avg_Val,
 0 as Msmt_Tot_Val, 0 as Msmt_Smpl_Cnt
from
 spp.stage_srm_r_hour dm,
 twg.tmzon t,
 spp.centr_lookup c,
 twg.centr ce
where
 dm.fqhn = c.value and
 c.centr_cd = ce.centr_cd and
 ce.tmzon_id = t.tmzon_id
;

-- CPU User
-- Note: we're using the timezone of the center where the data was collected
--       to calculte the GMT time.
--#EXECUTE_AT_TARGET
insert into spp.stage_metrics_hour
select
 0 as metric_id, current timestamp - current timezone as insert_dttm,
 DATE(TSTAMP), time(tstamp) - t.TmZon_GMT_Offset, lower(fqhn),
SUBSTR(fqhn,1,10), 'SPR_UnixProfile',
 'Unix_Sentry', 'avgcpuusr', 'none', '5',
 cpu_u * 0.7 as Msmt_Min_Val,
 cpu_u * 1.1 as Msmt_Max_Val,
 cpu_u as Msmt_Avg_Val,
 0 as Msmt_Tot_Val, 0 as Msmt_Smpl_Cnt
from
 spp.stage_srm_r_hour dm,
 twg.tmzon t,
 spp.centr_lookup c,
 twg.centr ce
where
 dm.fqhn = c.value and
 c.centr_cd = ce.centr_cd and
 ce.tmzon_id = t.tmzon_id
;

-- Run Q length
-- Note: we're using the timezone of the center where the data was collected
--       to calculte the GMT time.
--#EXECUTE_AT_TARGET
insert into spp.stage_metrics_hour
select
 0 as metric_id, current timestamp - current timezone as insert_dttm,
 DATE(TSTAMP), time(tstamp) - t.TmZon_GMT_Offset, lower(fqhn),
SUBSTR(fqhn,1,10), 'SPR_UnixProfile',
 'Unix_Sentry', 'runqjobs', 'none', '5',
 runq * 0.7 as Msmt_Min_Val,
 runq * 1.1 as Msmt_Max_Val,
 runq as Msmt_Avg_Val,
 0 as Msmt_Tot_Val, 0 as Msmt_Smpl_Cnt
from
 spp.stage_srm_r_hour dm,
 twg.tmzon t,
 spp.centr_lookup c,
 twg.centr ce
where
 dm.fqhn = c.value and
 c.centr_cd = ce.centr_cd and
 ce.tmzon_id = t.tmzon_id
;
```

srm_c20_s010_loadDMData

Example B-5 is the third script to run in the process for case study 1. Its responsibility is to load the measurement dimensional data. It loads stage tables in the CDW, then moves data into comp, in case the data is not already there, and then into the msmt table. It uses invalid data options.

Example: B-5 srm_c20_s010_loadDMData script

```
-- Populate the CDW tables with DM data.

-- EC is not needed for comp table because we use where not exists....the data
will only be inserted once even if rerun.

--#EXECUTE_AT_TARGET
--sample check for invalid data
insert into spp.invalid_data
select * from spp.stage_metrics_hour n
where
```

```
  not exists(
    select 1 from spp.cust_lookup
    where spp.cust_lookup.value = n.hostname OR
          spp.cust_lookup.value = '@') OR
  not exists(
    select 1 from spp.centr_lookup
    where spp.centr_lookup.value = n.hostname OR
          spp.centr_lookup.value = '@')
;

-- Note that the insert of IP_HOST might be changed to SPP_HOST by a trigger
--   if the name of the host isn't a fully qualified hostname.
--#EXECUTE_AT_TARGET
insert into twg.comp (COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
select
  0,
 'IP_HOST' as CompTyp_Cd,
 centr.Centr_Cd as Centr_Cd,
 c.Cust_ID as Cust_ID,
 d.hostname as Comp_Nm,
 current timestamp - current timezone as Comp_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as Comp_End_DtTm,
 '' as Comp_Ds
from
 spp.stage_metrics_hour d,
 inv.cust_lookup cust,
 spp.centr_lookup centr,
 twg.cust c
where
 not exists (
  select 1
  from
   twg.comp c
  where
   c.comp_nm=d.hostname
   and c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
   and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm ) AND
 ((cust.value = d.hostname and cust.cust_acct_cd = c.cust_acct_cd)
    OR cust.value = '@') AND
 (centr.value = d.hostname OR centr.value = '@')
group by
 d.hostname, centr.centr_cd, c.cust_id
;

-- For the prototype, we really don't need to use extract control when
-- inserting into msmt since we only have 1 step that populates the msmt
-- table. Use of EC is an example if you had multiple sql statements populating
-- the Msmt table. Then each statement would need EC to determine whether data
-- has already been loaded if the step gets rerun.

-- set the high value for the data to be extracted
--  #EXECUTE_AT_TARGET
--update TWG.Extract_Control set
-- extctl_to_intseq = (select max(metric_id) from SPP.STAGE_METRICS_HOUR),
-- extctl_to_dttm = (select max(insert_dttm) from SPP.STAGE_METRICS_HOUR)
--where
-- extctl_source='SPP.STAGE_METRICS_HOUR' and
-- extctl_target='TWG.MSMT' and
-- (select count(*) from SPP.STAGE_METRICS_HOUR) > 0
--;

--#EXECUTE_AT_TARGET
insert into twg.msmt
select
 0 as Msmt_ID,
 c.Comp_ID       ,
 m.MsmtTyp_ID    ,
 s.TmSum_Cd      ,
 d.collection_date as Msmt_Strt_Dt,
 d.collection_time as Msmt_Strt_Tm,
 avg(d.Msmt_Min_Val) as  Msmt_Min_Val,
 avg(d.Msmt_Max_Val)  as Msmt_Max_Val,
 avg(d.Msmt_Avg_Val) as Msmt_Avg_Val   ,
 avg(d.Msmt_Tot_Val) as Msmt_Tot_Val ,
 avg(d.Msmt_Smpl_Cnt) as Msmt_Smpl_Cnt,
 0 as Msmt_Err_Cnt
from
 spp.stage_metrics_hour d,
 twg.extract_control ec,
 twg.comp c,
 twg.msmttyp m,
 twg.msmtrul mr,
 twg.tmsum s
where
 c.comp_nm=d.hostname and
 m.MsmtTyp_Nm = d.probe and
 m.msmttyp_id = mr.msmttyp_id and
 mr.comptyp_cd IN ('SPP_HOST', 'IP_HOST') and
 s.TmSum_Nm = 'Hourly'
group by
 c.Comp_ID       ,
 m.MsmtTyp_ID    ,
 s.TmSum_Cd      ,
 d.collection_date,
 d.collection_time
;
```

```
-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.

--  #EXECUTE_AT_TARGET
--insert into twg.extract_log
--select
-- ec.extctl_source as extlog_source,
-- ec.extctl_target as extlog_target,
-- current timestamp as extlog_done_dttm,
-- ec.extctl_from_rawseq as extlog_from_rawseq,
-- ec.extctl_to_rawseq as extlog_to_rawseq,
-- ec.extctl_from_intseq as extlog_from_intseq,
-- ec.extctl_to_intseq as extlog_to_intseq,
-- ec.extctl_from_dttm as extlog_from_dttm,
-- ec.extctl_to_dttm as extlog_to_dttm
--from
-- TWG.Extract_control ec
--where
-- ec.extctl_source = 'SPP.STAGE_METRICS_HOUR' and
-- ec.extctl_target = 'TWG.MSMT'
--;
```

srm_m05_s010_buildMart

The script in Example B-6 moves the SRM data from the CDW into the data mart and is used for case study 1 ETL 2. First it moves the metrics data into the d_metrics table, then it moves data into the stage table, then it moves data into the attribute tables ip_address and host name views, and finally the fact data into the f_hour table. It utilizes the extract control system.

Example: B-6   srm_m05_s010_buildMart script

```
-------------------------------
-- insert into the metric dimension table
------------------------------
--NOTE: This process will only happen once. We initially insert the data into
the staging table in the cdw/dml script.
--
--
-- 3/22/02Chip Crane
-- Changed to meet SRM needs and added documentation
-- Note:This script, nor does any other ETL, load the spp.stage_d_metrics
table.
```

```
--    It is loaded through "some" static load process.  This should be
corrected.
--
--
-- set the high value for the data to be extracted
update TWG.Extract_Control set
 extctl_to_intseq = (select max(metric_id) from SPP.STAGE_D_METRIC),
 extctl_to_dttm = (select max(insert_dttm) from SPP.STAGE_D_METRIC)
where
 extctl_source='SPP.STAGE_D_METRIC' and
 extctl_target='SPP.D_METRIC' and
 (select count(*) from SPP.STAGE_D_METRIC) > 0
;

--#INSERT_INTO_TARGET
insert into spp.d_metric
select
  d.metric_id as metric_id,
  d.Met_Name as Met_Name,
  d.Met_Desc as Met_Desc,
  d.Met_units as Met_units,
  d.Met_Category as Met_Category,
  d.min_exists as MIN_Exists,
  d.max_exists as MAX_Exists,
  d.avg_exists as AVG_Exists,
  d.total_exists as Total_Exists,
  d.Msrc_Nm as MSrc_Nm
from
  spp.stage_d_metric d,
  twg.extract_control ec
where
  ec.extctl_source = 'SPP.STAGE_D_METRIC' and
  ec.extctl_target = 'SPP.D_METRIC' and
  d.metric_id > ec.extctl_from_intseq and
  d.metric_id <= ec.extctl_to_intseq
;

-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.

insert into twg.extract_log
select
 ec.extctl_source as extlog_source,
 ec.extctl_target as extlog_target,
 current timestamp - current timezone as extlog_done_dttm,
 ec.extctl_from_rawseq as extlog_from_rawseq,
 ec.extctl_to_rawseq as extlog_to_rawseq,
 ec.extctl_from_intseq as extlog_from_intseq,
 ec.extctl_to_intseq as extlog_to_intseq,
 ec.extctl_from_dttm as extlog_from_dttm,
 ec.extctl_to_dttm as extlog_to_dttm
from
 TWG.Extract_control ec
where
 ec.extctl_source = 'SPP.STAGE_D_METRIC' and
 ec.extctl_target = 'SPP.D_METRIC'
;

--#IGNORE_ERROR
drop table spp.stage_host_state;

create table spp.stage_host_state like spp.stage_host_state_template;

insert into spp.stage_host_state
select
  c1.comp_id as host_id,
  v1.compattr_strt_dttm as host_state_start_dttm,
  '9999-01-01-00.00.00.000000' as host_state_end_dttm,
  c1.comp_nm as hostname,
  ca1.CompAttr_Val as ip_address,
  ca2.CompAttr_Val as os_name,
  ca3.CompAttr_Val as os_version,
  ca4.CompAttr_Val as os_sub_ver,
  ca5.CompAttr_Val as physical_mem_kb,
  ca6.CompAttr_Val as num_cpus,
  ca7.CompAttr_Val as cpu_model,
  ca8.CompAttr_Val as cpu_speed,
  'Unknown' as system_purpose
from
  ( select
      comp_id,
      min(compattr_strt_dttm) as compattr_strt_dttm
    from spp.host_state_temp group by comp_id ) v1,
  twg.cur_comp c1,
  twg.cur_compattr ca1,
  twg.cur_compattr ca2,
  twg.cur_compattr ca3,
  twg.cur_compattr ca4,
  twg.cur_compattr ca5,
  twg.cur_compattr ca6,
  twg.cur_compattr ca7,
  twg.cur_compattr ca8
where
  c1.comp_id = v1.comp_id and
  c1.comptyp_cd IN ('SPP_HOST', 'IP_HOST') and
  c1.comp_id = ca1.comp_id and
  ca1.attrtyp_cd = 'INADDR' and
  c1.comp_id = ca2.comp_id and
  ca2.attrtyp_cd = 'INOSNM' and
  c1.comp_id = ca3.comp_id and
  ca3.attrtyp_cd = 'INOSVR' and
  c1.comp_id = ca4.comp_id and
  ca4.attrtyp_cd = 'INOSSV' and
  c1.comp_id = ca5.comp_id and
  ca5.attrtyp_cd = 'INPMEM' and
  c1.comp_id = ca6.comp_id and
  ca6.attrtyp_cd = 'INNCPU' and
  c1.comp_id = ca7.comp_id and
  ca7.attrtyp_cd = 'INCPUM' and
  c1.comp_id = ca8.comp_id and
  ca8.attrtyp_cd = 'INCPUS'
;

-- We could base the views on stage_host_state_template, but then we would
-- have to "delete from stage_host_state_template" and "insert into
stage_host_state_template
-- select * from stage_host_state." The problem with this is the space
-- isn't reclaimed until you do a db reorg, so I chose to recreate the views
here.

drop view spp.ip_address_v1;
create view spp.ip_address_v1 as
select
  ip_address,
  LOCATE('.',ip_address) - 1 as addr_w_end,
  LOCATE('.',ip_address,LOCATE('.',ip_address) + 1) -1 as addr_x_end,
  LOCATE('.',ip_address,LOCATE('.',ip_address,LOCATE('.',ip_address) + 1) + 1)
-1 as addr_y_end
from
  spp.stage_host_state
group by
  ip_address
;

drop view spp.hostname_v1;
create view spp.hostname_v1 as
select
  hostname,
  LOCATE('.',hostname) + 1 as name_b_beg,
  LOCATE('.',hostname,LOCATE('.',hostname) + 1) + 1 as name_c_beg,
  LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname) + 1) + 1) + 1 as
name_d_beg,

LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname
) + 1) + 1) + 1) +1 as name_e_beg
```

```
from
  spp.stage_host_state
group by
  hostname
;

drop view spp.hostname_v2;
create view spp.hostname_v2 as
select
  hostname,
  name_b_beg,
  name_c_beg,
  name_d_beg,
  name_e_beg,
  case  when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg > 1 and
name_e_beg > 1) then 5
      when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg > 1 and name_e_beg
= 1) then 4
      when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg = 1 ) then 3
      when (name_b_beg > 1 and name_c_beg = 1 ) then 2
      when (name_b_beg = 1 ) then 1
  end   as name_parts
from
  spp.hostname_v1
;

--#IGNORE_ERROR
drop table spp.stage_ip_address;

create table spp.stage_ip_address like spp.stage_ip_address_template;

insert into spp.stage_ip_address
select
  ip_address,
  CASE
     when ip_address is null THEN null
    when addr_w_end > 0 THEN SUBSTR(ip_address, 1, addr_w_end)
    else null
  END  as IP_A_NETWORK,
  CASE
    when ip_address is null THEN null
    when addr_x_end > 0 THEN SUBSTR(ip_address, 1, addr_x_end)
    else null
  END  as IP_B_NETWORK,
  CASE
    when ip_address is null THEN null
    when addr_y_end > 0 THEN SUBSTR(ip_address, 1, addr_y_end)
    else null
  END  as IP_C_NETWORK
```

```
from spp.ip_address_v1
;

--#IGNORE_ERROR
drop table spp.stage_hostname;

create table spp.stage_hostname like spp.stage_hostname_template;

insert into spp.stage_hostname
select
  hostname,
  CASE
    when name_parts = 5 then substr(hostname,name_d_beg)
    when name_parts = 4 then substr(hostname,name_c_beg)
    when name_parts = 3 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_DOMAIN,
  CASE
    when name_parts = 5 then substr(hostname,name_c_beg)
    when name_parts = 4 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_SUBDOMAIN,
  CASE
    when name_parts = 5 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_SUBDOMAIN_2,
  CASE
    when hostname is null THEN null
    when name_b_beg > 1 THEN SUBSTR(hostname, 1, name_b_beg - 2)
    else hostname
  END  as short_hostname
from   spp.hostname_v2
;

----------------------------------------------------------------------
-- 3/22/02 CC added:  This will allow us to do the "not in" function
--         in the next query.
-- Moves data from the CDW to the Mart in bulk/raw form into a
-- a stage table.  This data will then be allowed to be compared to
-- the already existing data.
-- This is in response to the unique constraint violation we were
-- getting because it attempts to insert ALL records from CDW into
-- a demension table that has already been populated with at least
-- some record.

-- Drop the staging table
--#EXECUTE_AT_SOURCE
DROP TABLE spp.mart_stage_host_state;
```

```
-- Create the staging table
--#EXECUTE_AT_SOURCE
CREATE TABLE SPP.mart_stage_host_state
  (     HOST_ID INTEGER NOT NULL ,
        CUST_ACCT_CD CHAR(10) NOT NULL ,
        CENTR_CD CHAR(6) NOT NULL ,
        STATE_STRT_DTTM TIMESTAMP NOT NULL ,
        STATE_END_DTTM TIMESTAMP NOT NULL ,
        HOSTNAME VARCHAR(64) NOT NULL ,
        IP_ADDRESS VARCHAR(32) NOT NULL ,
        OS_NAME VARCHAR(16) NOT NULL ,
        OS_VERSION VARCHAR(16) NOT NULL ,
        OS_SUB_VER VARCHAR(32) ,
        PHYSICAL_MEM_KB VARCHAR(30) NOT NULL ,
        NUM_CPUS VARCHAR(30) NOT NULL ,
        CPU_MODEL VARCHAR(32) NOT NULL ,
        CPU_SPEED VARCHAR(30) NOT NULL ,
        SYSTEM_PURPOSE VARCHAR(32) NOT NULL ,
        IP_A_NETWORK VARCHAR(32) NOT NULL ,
        IP_B_NETWORK VARCHAR(32) NOT NULL ,
        IP_C_NETWORK VARCHAR(32) NOT NULL ,
        NETWORK_DOMAIN VARCHAR(64) ,
        NETWORK_SUBDOMAIN VARCHAR(64) ,
        NETWORK_SUBDOMAIN_2 VARCHAR(64) ,
        SHORT_HOSTNAME VARCHAR(64) ,
        PHYSICAL_MEMORY VARCHAR(32) ,
        MULTIPROCESSOR VARCHAR(32) ,
        PROCESSOR_INFO VARCHAR(64) ,
        CPU_RATING VARCHAR(64) ,
        OS_NAME_VERSION VARCHAR(64) )
       IN USERSPACE1 ;


-- Insert ALL records into staging table
-- Yes this is redundant, since this will only change when new host are added
-- in hopes that development can help later
--#INSERT_INTO_SOURCE
INSERT INTO spp.mart_stage_host_state
SELECT *
  FROM spp.d_host_state;

-- 3/22/02 CC - End of CC add ------------

-- 3/22/02 CC changed from INSERT_INTO_TARGET to EXECUTE_AT_TARGET
--#INSERT_INTO_TARGET
insert into spp.d_host_state
select
  hst.host_id,
  cust.cust_acct_cd,
  centr.centr_cd,
  hst.state_strt_dttm,
  hst.state_end_dttm,
  hst.hostname,
  hst.ip_address,
  hst.os_name,
  hst.os_version,
  hst.os_sub_ver,
  hst.physical_mem_kb,
  hst.num_cpus,
  hst.cpu_model,
  hst.cpu_speed,
  hst.system_purpose,
  ip.IP_A_NETWORK,
  ip.IP_B_NETWORK,
  ip.IP_C_NETWORK,
  h.NETWORK_DOMAIN,
  h.NETWORK_SUBDOMAIN,
  h.NETWORK_SUBDOMAIN_2,
  h.SHORT_HOSTNAME,
  case
   when integer(physical_mem_kb) > 0 and integer(physical_mem_kb) <= 32768
      then '0-32 Megabytes'
  when integer(physical_mem_kb) > 32768 and integer(physical_mem_kb) <= 65536
      then '32-64 Megabytes'
    when integer(physical_mem_kb) > 65536 and integer(physical_mem_kb) <=
262144
      then '64-256 Megabytes'
    when integer(physical_mem_kb) > 262144 and integer(physical_mem_kb) <=
1048576
      then '256-1024 Megabytes'
    when integer(physical_mem_kb) > 1048576 and integer(physical_mem_kb) <=
4194304
       then '1-4 Gigabytes'
    else 'unknown'
  end as PHYSICAL_MEMORY,
  case
    when integer(hst.num_cpus) = 1 then 'Single Processor'
    else 'Multi Processor'
  end as MULTIPROCESSOR,
  case
    when integer(hst.num_cpus) = 1 then hst.cpu_model concat ' '
         concat rtrim(char(hst.cpu_speed)) concat ' Mhz Single Processor'
    else hst.cpu_model concat ' ' concat rtrim(char(hst.cpu_speed))
         concat ' Mhz Multi Processor'
  end as PROCESSOR_INFO,
  ct.cpu_rating,
  hst.os_name concat ' ' concat hst.os_version as os_name_version
from
```

```
  spp.stage_host_state hst,
  spp.stage_ip_address ip,
  spp.stage_hostname h,
  inv.cpu_term ct,
  inv.cust_lookup cust,
  spp.centr_lookup centr
where
  hst.ip_address=ip.ip_address and
  hst.hostname=h.hostname and
  hst.cpu_model=ct.cpu_model and
  hst.cpu_speed=ct.cpu_speed and
  hst.num_cpus=ct.num_cpus and
  hst.hostname=cust.value and
  hst.hostname=centr.value
  AND 0 = ( SELECT count(1)
         FROM spp.mart_stage_host_state s1
        WHERE s1.ip_address = hst.ip_address
          AND s1.hostname   = hst.hostname
          AND s1.os_name    = hst.os_name
               AND s1.host_id = hst.host_id
          AND s1.cust_acct_cd = cust.cust_acct_cd
          AND s1.centr_cd= centr.centr_cd
          AND s1.os_version= hst.os_version
          AND s1.os_sub_ver= hst.os_sub_ver
          AND s1.physical_mem_kb= hst.physical_mem_kb
          AND s1.num_cpus= hst.num_cpus
          AND s1.cpu_model= hst.cpu_model
          AND s1.cpu_speed= hst.cpu_speed
          AND s1.system_purpose= hst.system_purpose
          AND s1.ip_a_network= ip.IP_A_NETWORK
          AND s1.ip_b_network= ip.IP_B_NETWORK
          AND s1.ip_c_network= ip.IP_C_NETWORK
          AND s1.network_domain    = h.NETWORK_DOMAIN
          AND s1.network_subdomain = h.NETWORK_SUBDOMAIN
          AND s1.network_subdomain_2 = h.NETWORK_SUBDOMAIN_2
          AND s1.short_hostname= h.SHORT_HOSTNAME)
;

-- set the high value for the data to be extracted
update TWG.extract_control set
 extctl_to_intseq = ( select max(msmt_id) from twg.msmt)
where
 extctl_source='TWG.MSMT' and
 extctl_target='SPP.STAGE_F_HOUR' and
 (select count(*) from twg.msmt) > 0
;

--#IGNORE_ERROR
--#EXECUTE_AT_TARGET
```

```
drop table spp.stage_f_hour;

--#EXECUTE_AT_TARGET
create table SPP.stage_f_hour like SPP.f_hour
;

--The following should be part of the where clause. However, our simulated
--dm data is static from June and our inventory data gets created everytime
-- the process is run, so the dates are out of sync

--  m.msmt_strt_dt between date(h.state_strt_dttm) and date(h.state_end_dttm)
and
--#INSERT_INTO_TARGET
insert into spp.stage_f_hour
select
  m.comp_id as host_id,
  h.state_strt_dttm as host_state_strt_dttm,
  m.msmttyp_id as metric_id,
  timestamp(m.msmt_strt_dt, m.msmt_strt_tm) as meas_hour,
  m.msmt_min_val as min_value,
  m.msmt_max_val as max_value,
  m.msmt_avg_val as avg_value
from
  twg.msmt m,
  twg.msmttyp mt,
  twg.extract_control ec,
  spp.stage_host_state h
where
  m.msmt_id > ec.extctl_from_intseq and
  m.msmt_id <= ec.extctl_to_intseq and
  ec.extctl_source='TWG.MSMT' and
  ec.extctl_target='SPP.STAGE_F_HOUR' and
  (m.msmttyp_id = mt.msmttyp_id and mt.msrc_cd = 'SPP') and
  m.tmsum_cd = 'H' and
  m.comp_id = h.host_id
;

--#EXECUTE_AT_TARGET
insert into spp.f_hour
select * from spp.stage_f_hour
;

-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.

insert into twg.extract_log
select
```

```
 ec.extctl_source as extlog_source,
 ec.extctl_target as extlog_target,
 current timestamp - current timezone as extlog_done_dttm,
 ec.extctl_from_rawseq as extlog_from_rawseq,
 ec.extctl_to_rawseq as extlog_to_rawseq,
 ec.extctl_from_intseq as extlog_from_intseq,
 ec.extctl_to_intseq as extlog_to_intseq,
 ec.extctl_from_dttm as extlog_from_dttm,
 ec.extctl_to_dttm as extlog_to_dttm
from
 TWG.Extract_control ec
where
 ec.extctl_source = 'TWG.MSMT' and
 ec.extctl_target = 'SPP.STAGE_F_HOUR'
;

--The f_day, f_week, f_month tables are populated by executing
--the rollup.sh UDP.
```

ai1_c05_s010_extractData

Example B-7 is an ETL script for moving data from the CDW stage tables to the CDW tables for case study 2.

Example: B-7   ai1_c05_s010_extractData script

```
------------------------------------------------------------

Date/Author 3/25/02 CC
Description ETL for moving data from the CDW stage tables to the CDW tables.
Code SourcesCopied from spp_c05_s010_extractInvData.db2
--    Adapted to suit needs
Parameters(See Datawarehouse Center)
Inputstwh_cdw.inv.stage_ais_sp_hrly_evt_sev
Outputstwh_cdw.inv.invalid_data ( bad data )
--    twh_cdw.twg.comp(good components)
Project
Mod log
-------------------------------------------------------------

-Dont think we need to stage again, therfore, doc'd out
-This can be used as an example to go through the staging process
#EXECUTE_AT_TARGET
--drop table inv.stage_ipnetwork_copy;
#EXECUTE_AT_TARGET
--create table inv.stage_ipnetwork_copy like inv.stage_ipnetwork;
```

```
#INSERT_INTO_TARGET
--insert into inv.stage_ipnetwork_copy
--SELECT
  inv.COMPUTER_SYS.HWARE_SYS_ID AS HWARE_SYS_ID,
  inv.NET_NODE.NET_NODE_NAME AS NET_NODE_NAME,
  inv.NET_NODE.NET_NODE_ADDR AS NET_NODE_ADDR
--FROM
  inv.COMPUTER_SYS LEFT OUTER JOIN inv.NET_NODE
     ON ( inv.COMPUTER_SYS.HWARE_SYS_ID = inv.NET_NODE.HWARE_SYS_ID )
--WHERE
  (
    ( inv.NET_NODE.NET_PROTOCOL = 'TCP' )
 OR ( inv.NET_NODE.NET_PROTOCOL is null )
AND (( inv.NET_NODE.CFG_CHG_TYPE = 'INSERT' OR  inv.NET_NODE.CFG_CHG_TYPE =
'UPDATE' )
 OR ( inv.NET_NODE.CFG_CHG_TYPE is null ))
  )
--ORDER BY
  HWARE_SYS_ID,
  NET_NODE_NAME,
  NET_NODE_ADDR
--;

#EXECUTE_AT_TARGET
--drop table inv.stage_ostype_copy;
#EXECUTE_AT_TARGET
--create table inv.stage_ostype_copy like inv.stage_ostype;
#INSERT_INTO_TARGET
--insert into inv.stage_ostype_copy
--SELECT
  inv.COMPUTER_SYS.HWARE_SYS_ID AS HWARE_SYS_ID,
  inv.COMPUTER_SYS.BOOTED_OS_NAME AS BOOTED_OS_NAME,
  inv.COMPUTER_SYS.BOOTED_OS_VER AS BOOTED_OS_VER,
  inv.COMPUTER_SYS.COMPUTER_KRNL_VER AS COMPUTER_KRNL_VER,
  inv.NT_INFO.NT_SVC_PACK AS NT_SVC_PACK
--FROM
  inv.NT_INFO RIGHT OUTER JOIN inv.COMPUTER_SYS
     ON ( inv.NT_INFO.HWARE_SYS_ID = inv.COMPUTER_SYS.HWARE_SYS_ID )
--;

#EXECUTE_AT_TARGET
--drop table inv.stage_processor_copy;
#EXECUTE_AT_TARGET
--create table inv.stage_processor_copy like inv.stage_processor;
#INSERT_INTO_TARGET
--insert into inv.stage_processor_copy
--SELECT
  i.HWARE_SYS_ID,
  count(*),
  MIN(p.PROCESSOR_MODEL),
  MIN(p.PROCESSOR_SPEED)
--FROM
  inv.PROCESSOR p,
  inv.INST_PROCESSOR i
--WHERE
  i.PROCESSOR_ID = p.PROCESSOR_ID
--GROUP BY
  i.HWARE_SYS_ID
--;

#EXECUTE_AT_TARGET
--drop table inv.stage_memory_copy;
#EXECUTE_AT_TARGET
--create table inv.stage_memory_copy like inv.stage_memory;
#INSERT_INTO_TARGET
--insert into inv.stage_memory_copy
--SELECT
  HWARE_SYS_ID,
  PHYSICAL_MEM_KB
--FROM
  inv.COMPUTER_SYS_MEM
--;

--commit;

#EXECUTE_AT_TARGET
--drop table inv.stage_net_node_copy;
#EXECUTE_AT_TARGET
--create table inv.stage_net_node_copy like inv.stage_net_node;
#EXECUTE_AT_TARGET
--insert into inv.stage_net_node_copy
--select
 value(lower(T2.NET_NODE_NAME),'') as net_node_name,
 value(T2.NET_NODE_ADDR,'') as net_node_addr,
 T1.HWARE_SYS_ID,
 value(T1.BOOTED_OS_NAME,'') as booted_os_name,
 value(T1.BOOTED_OS_VER,'') as booted_os_ver,
 case
   when T1.BOOTED_OS_NAME in ('Windows NT','Windows 2000')
      then value(T1.NT_SVC_PACK,'')
   when T1.BOOTED_OS_NAME in ('AIX', 'SunOS', 'HP-UX', 'Linux')
      then value(T1.COMPUTER_KRNL_VER,'')
   else ''
 end as os_sub_ver,
 value(T4.PHYSICAL_MEM_KB,-1) as physical_mem_kb,
 value(T6.NUM_PROCESSORS,-1) as num_processors,
 value(T6.PROCESSOR_MODEL,'') as processor_model,
 value(T6.PROCESSOR_SPEED,-1) as processor_speed,
 'Unknown' as system_purpose
--FROM
 inv.stage_OSTYPE_copy T1       LEFT OUTER JOIN
 inv.stage_IPNETWORK_copy  T2
     ON     T1.HWARE_SYS_ID = T2.HWARE_SYS_ID,
 inv.stage_OSTYPE_copy T3       LEFT OUTER JOIN
 inv.stage_MEMORY_copy T4
     ON     T3.HWARE_SYS_ID = T4.HWARE_SYS_ID,
 inv.stage_OSTYPE_copy T5       LEFT OUTER JOIN
 inv.stage_PROCESSOR_copy T6
     ON     T5.HWARE_SYS_ID = T6.HWARE_SYS_ID
--WHERE
 T1.HWARE_SYS_ID=T3.HWARE_SYS_ID      AND
 T1.HWARE_SYS_ID=T5.HWARE_SYS_ID
--;

----------------------------------
Move data that does not have a
cust or center ids into invalid
data table.  Filled with text
since we do not have all the values.
----------------------------------
--#EXECUTE_AT_TARGET
--sample check for invalid data
insert into inv.invalid_data
select  n.hostname,'AIS_NULL', 'AIS_NULL','AIS_NULL','AIS_NULL',
   'AIS_NULL',0,0,'AIS_NULL',0,'AIS_NULL'
  from inv.stage_ais_sp_hrly_evt_sev n
 where not exists(
    select 1
     from inv.cust_lookup
     where inv.cust_lookup.value = n.hostname
       OR inv.cust_lookup.value = '@')
   OR
   not exists(
     select 1 from inv.centr_lookup
      where inv.centr_lookup.value = n.hostname
        OR inv.centr_lookup.value = '@')
;
-----------------------------------------------------------------------
Need to get the distinct hostname to put into comp for some reason
a record in the stage table produces a record in the comp table,
which produces 100+ comp of the same hostname.
Therefore moving them into another stage of just hostnames
-----------------------------------------------------------------------
--#EXECUTE_AT_TARGET
DROP TABLE inv.stage_ais_sp_host;
--#EXECUTE_AT_TARGET
CREATE TABLE inv.stage_ais_sp_host(hostname VARCHAR(32));
```

```
--#EXECUTE_AT_TARGET
INSERT INTO inv.stage_ais_sp_host
SELECT DISTINCT hostname
  FROM inv.stage_ais_sp_hrly_evt_sev;
------------------------------------------------------------------------
Insert data into the twg.comp(component table).  This is the real work
of the entire script
--
This will not duplicate rows because it does a "not exist" in comp table
--
--
--Note that the insert of IP_HOST may be changed to SPP_HOST by a trigger
 if the hostname is not fully qualified.
-------------------------------------------------------------------------
--#EXECUTE_AT_TARGET
INSERT INTO TWG.COMP(COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
select 0 as Comp_Id,
   'IP_HOST' as CompTyp_Cd ,
   centr.Centr_Cd as Centr_Cd,
   c.Cust_ID as Cust_ID,
   n.hostname as Comp_Nm,
   current timestamp - current timezone as Comp_Strt_DtTm ,
   '9999-01-01-00.00.00.000000' as Comp_End_DtTm,
   '' as Comp_Ds
from inv.stage_ais_sp_host n,
   inv.cust_lookup cust,
   inv.centr_lookup centr,
   twg.cust c
where n.hostname != ''
  and   0 = (
      select count(1)
        from twg.comp c
       where c.comp_nm=n.hostname
         and c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
         and current timestamp - current timezone >= (c.comp_strt_dttm - 1
day)
         and current timestamp - current timezone <  c.comp_end_dttm )
  AND ((cust.value = n.hostname and cust.cust_acct_cd = c.cust_acct_cd)
   OR cust.value = '@')
  AND (centr.value = n.hostname OR centr.value = '@')
;

---------------------
since we only have
hostname, we have
nothing to update
compattr with, so
we doc it out
```

```
---------------------
#EXECUTE_AT_TARGET
--insert into twg.compattr
--select
0 as compattr_id,
c.comp_id,
'INOSNM' as AttrTyp_Cd,
current timestamp - current timezone as CompAttr_Strt_DtTm,
'9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
n.booted_os_name as CompAttr_Val
--from
inv.stage_ais_sp_hrly_evt_sev n,
twg.comp c
--where
c.comp_nm=n.hostname
and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
and n.booted_os_name != ''
and not exists (
 select 1
 from
  twg.compattr ca,
  twg.comp c2
 where
 n.hostname=c2.comp_nm
  and ca.AttrTyp_Cd='INOSNM'
  and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
--union
--select
0 as compattr_id,
c.comp_id,
'INOSVR' as AttrTyp_Cd,
current timestamp - current timezone as CompAttr_Strt_DtTm,
'9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
n.booted_os_ver as CompAttr_Val
--from
inv.stage_net_node_copy n,
twg.comp c
--where
c.comp_nm=n.net_node_name
and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
and n.booted_os_ver != ''
and not exists (
 select 1
 from
  twg.compattr ca,
  twg.comp c2
 where
  n.net_node_name=c2.comp_nm
  and ca.CompAttr_Val=n.booted_os_ver
  and ca.AttrTyp_Cd='INOSVR'
  and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
--union
--select
0 as compattr_id,
c.comp_id,
'INOSSV' as AttrTyp_Cd,
current timestamp - current timezone as CompAttr_Strt_DtTm,
'9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
n.os_sub_ver as CompAttr_Val
--from
inv.stage_net_node_copy n,
twg.comp c
--where
c.comp_nm=n.net_node_name
and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
and n.os_sub_ver != ''
and not exists (
 select 1
 from
  twg.compattr ca,
  twg.comp c2
 where
  n.net_node_name=c2.comp_nm
  and ca.CompAttr_Val=n.os_sub_ver
  and ca.AttrTyp_Cd='INOSSV'
  and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
--;
-Removed large amounts of code that moved data into compattr.
```

ai1_c20_s010_LoadEvData

The script in Example B-8 is used to populate the CDW tables with DM data for case study 2.

Example: B-8 ai1_c20_s010_LoadEvData script

```
-- Populate the CDW tables with DM data.
-------------------------------------------------------------
-- Date/Author 3/26/02 CC
-- Description ETL for moving data from the CDW stage tables
--    to the CDW tables.
--
-- Code SourcesCopied from spp_c20_s010_extractDMData.db2
--    Adapted to suit needs
-- Parameters(See Datawarehouse Center)
-- Inputstwh_cdw.inv.stage_ais_sp_hrly_evt_sev
-- Outputstwh_cdw.inv.invalid_data ( bad data )
--    twh_cdw.twg.msmt(good data)
-- Project
-- Mod log
-------------------------------------------------------------

-- EC isn't needed for comp table because we use where not exists....the data
will only
-- be inserted once even if rerun.


--------------------------------------------
-- Move any invalid data ( data without
-- customer or center) into invalid data table
--------------------------------------------
-- #EXECUTE_AT_TARGET
--insert into spp.invalid_data
--select  0,current timestamp, current date, current time,
n.hostname,n.hostname, 'AIS_NULL','AIS_NULL','AIS_NULL',
-- 'AIS_NULL',0,0,0,0,evt_cnt
--  from inv.stage_ais_sp_hrly_evt_sev n
--where
--  not exists(
--    select 1 from spp.cust_lookup
--   where inv.cust_lookup.value = n.hostname OR
--          inv.cust_lookup.value = '@') OR
--  not exists(
--    select 1 from spp.centr_lookup
--    where inv.centr_lookup.value = n.hostname OR
--          inv.centr_lookup.value = '@')
--;

------------------------------------------------------------------------
-- Need to get the distinct hostname to put into comp for some reason
-- a record in the stage table produces a record in the comp table,
-- which produces 100+ comp of the same hostname.
-- Therefore moving them into another stage of just hostnames
------------------------------------------------------------------------
--#EXECUTE_AT_TARGET
DROP TABLE inv.stage_ais_sp_host;

--#EXECUTE_AT_TARGET
CREATE TABLE inv.stage_ais_sp_host(hostname VARCHAR(32));
```

```
--#EXECUTE_AT_TARGET
INSERT INTO inv.stage_ais_sp_host
SELECT DISTINCT hostname
  FROM inv.stage_ais_sp_hrly_evt_sev;

------------------------------------------------
-- Insert any components that did not already
-- exist in the components table
------------------------------------------------
-- Note that the insert of IP_HOST might be changed to SPP_HOST by a trigger
--   if the name of the host isn't a fully qualified hostname.
--#EXECUTE_AT_TARGET
insert into twg.comp (COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
select  0,
   'IP_HOST' as CompTyp_Cd,
   centr.Centr_Cd as Centr_Cd,
   c.Cust_ID as Cust_ID,
   d.hostname as Comp_Nm,
   current timestamp - current timezone as Comp_Strt_DtTm,
   '9999-01-01-00.00.00.000000' as Comp_End_DtTm,
   '' as Comp_Ds
from inv.stage_ais_sp_host d,
   inv.cust_lookup cust,
   inv.centr_lookup centr,
   twg.cust c
where  not exists (
   select 1
     from twg.comp c
    where c.comp_nm=d.hostname
      and c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
      and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm )
  AND  ((cust.value = d.hostname and cust.cust_acct_cd = c.cust_acct_cd)
   OR  cust.value = '@')
  AND (centr.value = d.hostname OR centr.value = '@')
group by d.hostname, centr.centr_cd, c.cust_id
;


--- CC 3/26/02 We will solve this incremental extrac with the query below
---
-- For the prototype, we really don't need to use extract control when
-- inserting into msmt since we only have 1 step that populates the msmt
-- table. Use of EC is an example if you had multiple sql statements populating
-- the Msmt table. Then each statement would need EC to determine whether data
-- has already been loaded if the step gets rerun.
-- set the high value for the data to be extracted
-- #EXECUTE_AT_TARGET
--update TWG.Extract_Control set
-- extctl_to_intseq = (select max(metric_id) from SPP.STAGE_METRICS_HOUR),
-- extctl_to_dttm = (select max(insert_dttm) from SPP.STAGE_METRICS_HOUR)
--where
-- extctl_source='SPP.STAGE_METRICS_HOUR' and
-- extctl_target='TWG.MSMT' and
-- (select count(*) from SPP.STAGE_METRICS_HOUR) > 0
--;

--#EXECUTE_AT_TARGET
insert into twg.msmt
select 0 as Msmt_ID,
   c.Comp_ID       ,
   m.MsmtTyp_ID    ,
   s.TmSum_Cd      ,
   date(d.hr_strt) as Msmt_Strt_Dt,
   time(d.hr_strt) as Msmt_Strt_Tm,
   0 as Msmt_Min_Val,
   0 as Msmt_Max_Val,
   0 as Msmt_Avg_Val   ,
   avg(d.evt_cnt) as Msmt_Tot_Val ,
   0 as Msmt_Smpl_Cnt,
   0 as Msmt_Err_Cnt
from inv.stage_ais_sp_hrly_evt_sev d,
   twg.comp c,
   twg.msmttyp m,
   twg.msmtrul mr,
   twg.tmsum s
where  c.comp_nm=d.hostname
--  and  d.hr_strt >  (  select max(timestamp(msmt_strt_dt,msmt_strt_tm))
--         from twg.msmt
--        where msmttyp_id=49)
  and   m.MsmtTyp_Nm = 'AIS_Events'
  and   m.msmttyp_id = mr.msmttyp_id
  and  mr.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
  and s.TmSum_Nm = 'Hourly'
group by c.Comp_ID, m.MsmtTyp_ID, s.TmSum_Cd, date(d.hr_strt), time(d.hr_strt)
;

-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.
-----------------------------------
-- Changed to pull min and max date
-- from the stage tables
-----------------------------------
```

```
--#EXECUTE_AT_TARGET
insert into twg.extract_log
select 'inv.stage_ais_sp_hrly_evt_sev' as extlog_source,
   'twg.msmt' as extlog_target,
   current timestamp as extlog_done_dttm,
   0 as extlog_from_rawseq,
   0 as extlog_to_rawseq,
   0 as extlog_from_intseq,
   0 as extlog_to_intseq,
   MIN(ec.hr_strt) as extlog_from_dttm,
   MAX(ec.hr_strt) as extlog_to_dttm
from  inv.stage_ais_sp_hrly_evt_sev ec
;
```

ai1_m05_s010_buildMart

Example B-9 is the ETL script for moving data from the CDW tables to the data mart tables for case study 2.

Example: B-9   ai1_m05_s010_buildMart script

```
-------------------------------------------------------------
-- Date/Author 3/26/02 CC
-- Description ETL for moving data from the CDW tables
--    to the data mart tables.
--
-- Code SourcesCopied from spp_m05_s010_buildMart.db2
--    Adapted to suit needs
-- Parameters(See Datawarehouse Center)
-- Inputstwh_cdw.comp
--    twh_cdw.msmt
--    twh_cdw.compattr
-- Outputstwh_mart.d_metrics
--    twh_mart.d_host_state
--    twh_mart.f_aisevt_hour
-- Project
-- Mod log
-------------------------------------------------------------


---------------------------------
-- insert into the metric dimension table
------------------------------
--NOTE: This process will only happen once. We initially insert the data into
the
--     staging table in the cdw/dml script.
```

```
---------------------------------------------------------
-- CC Left alone since we have already conformed  in CDW
-- this should work out of the box
---------------------------------------------------------
-- set the high value for the data to be extracted
update TWG.Extract_Control set
 extctl_to_intseq = (select max(metric_id) from SPP.STAGE_D_METRIC),
 extctl_to_dttm = (select max(insert_dttm) from SPP.STAGE_D_METRIC)
where
 extctl_source='SPP.STAGE_D_METRIC' and
 extctl_target='SPP.D_METRIC' and
 (select count(*) from SPP.STAGE_D_METRIC) > 0
;

----------------------------------------------
-- Made staic insert into spp.stage_d_metrics
-- table, therefore, this should work without
-- adjustment.
----------------------------------------------
--#INSERT_INTO_TARGET
insert into spp.d_metric
select  d.metric_id as metric_id,
   d.Met_Name as Met_Name,
   d.Met_Desc as Met_Desc,
   d.Met_units as Met_units,
   d.Met_Category as Met_Category,
   d.min_exists as MIN_Exists,
   d.max_exists as MAX_Exists,
   d.avg_exists as AVG_Exists,
   d.total_exists as Total_Exists,
   d.Msrc_Nm as MSrc_Nm
from  spp.stage_d_metric d,
   twg.extract_control ec
where   ec.extctl_source = 'SPP.STAGE_D_METRIC'
  and   ec.extctl_target = 'SPP.D_METRIC'
  and d.metric_id > ec.extctl_from_intseq
  and   d.metric_id <= ec.extctl_to_intseq
;

-------------------
-- No Changes
-------------------

-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.

insert into twg.extract_log
```

```
select
 ec.extctl_source as extlog_source,
 ec.extctl_target as extlog_target,
 current timestamp - current timezone as extlog_done_dttm,
 ec.extctl_from_rawseq as extlog_from_rawseq,
 ec.extctl_to_rawseq as extlog_to_rawseq,
 ec.extctl_from_intseq as extlog_from_intseq,
 ec.extctl_to_intseq as extlog_to_intseq,
 ec.extctl_from_dttm as extlog_from_dttm,
 ec.extctl_to_dttm as extlog_to_dttm
from
 TWG.Extract_control ec
where
 ec.extctl_source = 'SPP.STAGE_D_METRIC' and
 ec.extctl_target = 'SPP.D_METRIC'
;

--#IGNORE_ERROR
drop table spp.stage_host_state;

create table spp.stage_host_state like spp.stage_host_state_template;

insert into spp.stage_host_state
select
  c1.comp_id as host_id,
  v1.compattr_strt_dttm as host_state_start_dttm,
  '9999-01-01-00.00.00.000000' as host_state_end_dttm,
  c1.comp_nm as hostname,
  ca1.CompAttr_Val as ip_address,
  ca2.CompAttr_Val as os_name,
  ca3.CompAttr_Val as os_version,
  ca4.CompAttr_Val as os_sub_ver,
  ca5.CompAttr_Val as physical_mem_kb,
  ca6.CompAttr_Val as num_cpus,
  ca7.CompAttr_Val as cpu_model,
  ca8.CompAttr_Val as cpu_speed,
  'Unknown' as system_purpose
from
  ( select
      comp_id,
      min(compattr_strt_dttm) as compattr_strt_dttm
    from spp.host_state_temp group by comp_id ) v1,
  twg.cur_comp c1,
  twg.cur_compattr ca1,
  twg.cur_compattr ca2,
  twg.cur_compattr ca3,
  twg.cur_compattr ca4,
  twg.cur_compattr ca5,
  twg.cur_compattr ca6,
  twg.cur_compattr ca7,
  twg.cur_compattr ca8
where
  c1.comp_id = v1.comp_id and
  c1.comptyp_cd IN ('SPP_HOST', 'IP_HOST') and
  c1.comp_id = ca1.comp_id and
  ca1.attrtyp_cd = 'INADDR' and
  c1.comp_id = ca2.comp_id and
  ca2.attrtyp_cd = 'INOSNM' and
  c1.comp_id = ca3.comp_id and
  ca3.attrtyp_cd = 'INOSVR' and
  c1.comp_id = ca4.comp_id and
  ca4.attrtyp_cd = 'INOSSV' and
  c1.comp_id = ca5.comp_id and
  ca5.attrtyp_cd = 'INPMEM' and
  c1.comp_id = ca6.comp_id and
  ca6.attrtyp_cd = 'INNCPU' and
  c1.comp_id = ca7.comp_id and
  ca7.attrtyp_cd = 'INCPUM' and
  c1.comp_id = ca8.comp_id and
  ca8.attrtyp_cd = 'INCPUS'
;

-- We could base the views on stage_host_state_template, but then we would
-- have to "delete from stage_host_state_template" and "insert into
stage_host_state_template
-- select * from stage_host_state." The problem with this is the space
-- isn't reclaimed until you do a db reorg, so I chose to recreate the views
here.

drop view spp.ip_address_v1;
create view spp.ip_address_v1 as
select
  ip_address,
  LOCATE('.',ip_address) - 1 as addr_w_end,
  LOCATE('.',ip_address,LOCATE('.',ip_address) + 1) -1 as addr_x_end,
  LOCATE('.',ip_address,LOCATE('.',ip_address,LOCATE('.',ip_address) + 1) + 1)
-1 as addr_y_end
from
  spp.stage_host_state
group by
  ip_address
;

drop view spp.hostname_v1;
create view spp.hostname_v1 as
select
  hostname,
  LOCATE('.',hostname) + 1 as name_b_beg,
  LOCATE('.',hostname,LOCATE('.',hostname) + 1) + 1 as name_c_beg,
  LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname) + 1) + 1) + 1 as
name_d_beg,

LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname,LOCATE('.',hostname
) + 1) + 1) + 1) +1 as name_e_beg
from
  spp.stage_host_state
group by
  hostname
;

drop view spp.hostname_v2;
create view spp.hostname_v2 as
select
  hostname,
  name_b_beg,
  name_c_beg,
  name_d_beg,
  name_e_beg,
  case  when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg > 1 and
name_e_beg > 1) then 5
      when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg > 1 and name_e_beg
= 1) then 4
      when (name_b_beg > 1 and name_c_beg > 1 and name_d_beg = 1 ) then 3
      when (name_b_beg > 1 and name_c_beg = 1 ) then 2
      when (name_b_beg = 1 ) then 1
  end   as name_parts
from
  spp.hostname_v1
;

--#IGNORE_ERROR
drop table spp.stage_ip_address;

create table spp.stage_ip_address like spp.stage_ip_address_template;

insert into spp.stage_ip_address
select
  ip_address,
  CASE
     when ip_address is null THEN null
    when addr_w_end > 0 THEN SUBSTR(ip_address, 1, addr_w_end)
    else null
  END  as IP_A_NETWORK,
  CASE
    when ip_address is null THEN null
    when addr_x_end > 0 THEN SUBSTR(ip_address, 1, addr_x_end)
    else null
  END  as IP_B_NETWORK,
  CASE
    when ip_address is null THEN null
    when addr_y_end > 0 THEN SUBSTR(ip_address, 1, addr_y_end)
    else null
  END  as IP_C_NETWORK
from spp.ip_address_v1
;

--#IGNORE_ERROR
drop table spp.stage_hostname;

create table spp.stage_hostname like spp.stage_hostname_template;

insert into spp.stage_hostname
select
  hostname,
  CASE
    when name_parts = 5 then substr(hostname,name_d_beg)
    when name_parts = 4 then substr(hostname,name_c_beg)
    when name_parts = 3 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_DOMAIN,
  CASE
    when name_parts = 5 then substr(hostname,name_c_beg)
    when name_parts = 4 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_SUBDOMAIN,
  CASE
    when name_parts = 5 then substr(hostname,name_b_beg)
    else 'no value'
  END as NETWORK_SUBDOMAIN_2,
  CASE
    when hostname is null THEN null
    when name_b_beg > 1 THEN SUBSTR(hostname, 1, name_b_beg - 2)
    else hostname
  END  as short_hostname
from   spp.hostname_v2
;


----------------------------------------------------------------------
-- 3/25/02 CC Copied from previous script
-- 3/22/02 CC added:  This will allow us to do the "not in" function
--        in the next query.
-- Moves data from the CDW to the Mart in bulk/raw form into a
-- a stage table.  This data will then be allowed to be compared to
-- the already existing data.
-- This is in response to the unique constraint violation we were
-- getting because it attempts to insert ALL records from CDW into
-- a demension table that has already been populated with at least
-- some record.

-- Drop the staging table
-- #EXECUTE_AT_SOURCE -- This is default so need need to have a directive
DROP TABLE spp.mart_stage_host_state;

-- Create the staging table
-- #EXECUTE_AT_SOURCE -- This is default so need need to have a directive
CREATE TABLE SPP.mart_stage_host_state
  (     HOST_ID INTEGER NOT NULL ,
        CUST_ACCT_CD CHAR(10) NOT NULL ,
        CENTR_CD CHAR(6) NOT NULL ,
        STATE_STRT_DTTM TIMESTAMP NOT NULL ,
        STATE_END_DTTM TIMESTAMP NOT NULL ,
        HOSTNAME VARCHAR(64) NOT NULL ,
        IP_ADDRESS VARCHAR(32) NOT NULL ,
        OS_NAME VARCHAR(16) NOT NULL ,
        OS_VERSION VARCHAR(16) NOT NULL ,
        OS_SUB_VER VARCHAR(32) ,
        PHYSICAL_MEM_KB VARCHAR(30) NOT NULL ,
        NUM_CPUS VARCHAR(30) NOT NULL ,
        CPU_MODEL VARCHAR(32) NOT NULL ,
        CPU_SPEED VARCHAR(30) NOT NULL ,
        SYSTEM_PURPOSE VARCHAR(32) NOT NULL ,
        IP_A_NETWORK VARCHAR(32) NOT NULL ,
        IP_B_NETWORK VARCHAR(32) NOT NULL ,
        IP_C_NETWORK VARCHAR(32) NOT NULL ,
        NETWORK_DOMAIN VARCHAR(64) ,
        NETWORK_SUBDOMAIN VARCHAR(64) ,
        NETWORK_SUBDOMAIN_2 VARCHAR(64) ,
        SHORT_HOSTNAME VARCHAR(64) ,
        PHYSICAL_MEMORY VARCHAR(32) ,
        MULTIPROCESSOR VARCHAR(32) ,
        PROCESSOR_INFO VARCHAR(64) ,
        CPU_RATING VARCHAR(64) ,
        OS_NAME_VERSION VARCHAR(64) )
       IN USERSPACE1 ;


-- Insert ALL records into staging table
-- Yes this is redundant, since this will only change when new host are added
-- in hopes that development can help later
--#INSERT_INTO_SOURCE
INSERT INTO spp.mart_stage_host_state
SELECT *
  FROM spp.d_host_state;
```

```
-- 3/22/02 CC - End of CC add ------------
------------------------------------------
-- 3/25/02 CC Added the subselect in the
-- where to prevent duplicate records
-- being inserted
------------------------------------------
--#INSERT_INTO_TARGET
insert into spp.d_host_state
select
  hst.host_id,
  cust.cust_acct_cd,
  centr.centr_cd,
  hst.state_strt_dttm,
  hst.state_end_dttm,
  hst.hostname,
  hst.ip_address,
  hst.os_name,
  hst.os_version,
  hst.os_sub_ver,
  hst.physical_mem_kb,
  hst.num_cpus,
  hst.cpu_model,
  hst.cpu_speed,
  hst.system_purpose,
  ip.IP_A_NETWORK,
  ip.IP_B_NETWORK,
  ip.IP_C_NETWORK,
  h.NETWORK_DOMAIN,
  h.NETWORK_SUBDOMAIN,
  h.NETWORK_SUBDOMAIN_2,
  h.SHORT_HOSTNAME,
  case
   when integer(physical_mem_kb) > 0 and integer(physical_mem_kb) <= 32768
      then '0-32 Megabytes'
   when integer(physical_mem_kb) > 32768 and integer(physical_mem_kb) <= 65536
      then '32-64 Megabytes'
    when integer(physical_mem_kb) > 65536 and integer(physical_mem_kb) <=
262144
      then '64-256 Megabytes'
    when integer(physical_mem_kb) > 262144 and integer(physical_mem_kb) <=
1048576
      then '256-1024 Megabytes'
    when integer(physical_mem_kb) > 1048576 and integer(physical_mem_kb) <=
4194304
       then '1-4 Gigabytes'
    else 'unknown'
  end as PHYSICAL_MEMORY,
  case
    when integer(hst.num_cpus) = 1 then 'Single Processor'
    else 'Multi Processor'
  end as MULTIPROCESSOR,
  case
    when integer(hst.num_cpus) = 1 then hst.cpu_model concat ' '
         concat rtrim(char(hst.cpu_speed)) concat ' Mhz Single Processor'
    else hst.cpu_model concat ' ' concat rtrim(char(hst.cpu_speed))
         concat ' Mhz Multi Processor'
  end as PROCESSOR_INFO,
  ct.cpu_rating,
  hst.os_name concat ' ' concat hst.os_version as os_name_version
from
  spp.stage_host_state hst,
  spp.stage_ip_address ip,
  spp.stage_hostname h,
  inv.cpu_term ct,
  spp.cust_lookup cust,
  spp.centr_lookup centr
where
  hst.ip_address=ip.ip_address and
  hst.hostname=h.hostname and
  hst.cpu_model=ct.cpu_model and
  hst.cpu_speed=ct.cpu_speed and
  hst.num_cpus=ct.num_cpus and
  hst.hostname=cust.value and
  hst.hostname=centr.value
  AND 0 = ( SELECT count(1)
         FROM spp.mart_stage_host_state s1
        WHERE s1.ip_address = hst.ip_address
          AND s1.hostname   = hst.hostname
          AND s1.os_name    = hst.os_name
               AND s1.host_id = hst.host_id
          AND s1.cust_acct_cd = cust.cust_acct_cd
          AND s1.centr_cd= centr.centr_cd
          AND s1.os_version= hst.os_version
          AND s1.os_sub_ver= hst.os_sub_ver
          AND s1.physical_mem_kb= hst.physical_mem_kb
          AND s1.num_cpus= hst.num_cpus
          AND s1.cpu_model= hst.cpu_model
          AND s1.cpu_speed= hst.cpu_speed
          AND s1.system_purpose= hst.system_purpose
          AND s1.ip_a_network= ip.IP_A_NETWORK
          AND s1.ip_b_network= ip.IP_B_NETWORK
          AND s1.ip_c_network= ip.IP_C_NETWORK
          AND s1.network_domain    = h.NETWORK_DOMAIN
          AND s1.network_subdomain = h.NETWORK_SUBDOMAIN
          AND s1.network_subdomain_2 = h.NETWORK_SUBDOMAIN_2
          AND s1.short_hostname= h.SHORT_HOSTNAME)

;
```

```
-- set the high value for the data to be extracted
update TWG.extract_control set
 extctl_to_intseq = ( select max(msmt_id) from twg.msmt)
where
 extctl_source='TWG.MSMT' and
 extctl_target='SPP.STAGE_F_AISEVT_HOUR' and
 (select count(*) from twg.msmt) > 0
;

--#IGNORE_ERROR
--#EXECUTE_AT_TARGET
drop table spp.stage_f_aisevt_hour;

--#EXECUTE_AT_TARGET
create table SPP.stage_f_ais_evt_hour like SPP.f_aisevts_hour
;

--The following should be part of the where clause. However, our simulated
--dm data is static from June and our inventory data gets created everytime
-- the process is run, so the dates are out of sync

--  m.msmt_strt_dt between date(h.state_strt_dttm) and date(h.state_end_dttm)
and
--#INSERT_INTO_TARGET
insert into spp.stage_f_aisevts_hour
select
  m.comp_id as host_id,
  h.state_strt_dttm as host_state_strt_dttm,
  m.msmttyp_id as metric_id,
  timestamp(m.msmt_strt_dt, m.msmt_strt_tm) as meas_hour,
  m.msmt_tot_val as tot_val
from
  twg.msmt m,
  twg.msmttyp mt,
  twg.extract_control ec,
  spp.stage_host_state h
where
  m.msmt_id > ec.extctl_from_intseq and
  m.msmt_id <= ec.extctl_to_intseq and
  ec.extctl_source='TWG.MSMT' and
  ec.extctl_target='SPP.STAGE_F_HOUR' and
  (m.msmttyp_id = mt.msmttyp_id and mt.msrc_cd = 'TEC') and
  m.tmsum_cd = 'H' and
  m.comp_id = h.host_id
;

--#EXECUTE_AT_TARGET
insert into spp.f_aisevts_hour
```

```
select * from spp.stage_f_aisevts_hour
;

-- Confirm the insertion of data by inserting a row into TWG.EXTRACT_LOG.
-- A trigger gets fired which updates the extract control table's from data
with
-- the current to data.

insert into twg.extract_log
select
 ec.extctl_source as extlog_source,
 ec.extctl_target as extlog_target,
 current timestamp - current timezone as extlog_done_dttm,
 ec.extctl_from_rawseq as extlog_from_rawseq,
 ec.extctl_to_rawseq as extlog_to_rawseq,
 ec.extctl_from_intseq as extlog_from_intseq,
 ec.extctl_to_intseq as extlog_to_intseq,
 ec.extctl_from_dttm as extlog_from_dttm,
 ec.extctl_to_dttm as extlog_to_dttm
from
 TWG.Extract_control ec
where
 ec.extctl_source = 'TWG.MSMT' and
 ec.extctl_target = 'SPP.STAGE_F_HOUR'
;

--The f_day, f_week, f_month tables are populated by executing
--the rollup.sh UDP.
```

ai1_c05_s010_extractData

Example B-10 on page 355 is the original ETL script for the SPP prototype.

Example: B-10   ai1_c05_s010_extractData script

```
------------------------------------------------------------
-- 3/25/02 CC ETL for moving data from the CDW stage tables
--    to the CDW tables.
------------------------------------------------------------
--- Dont think we need to stage again, therfore, doc'd out
-- #EXECUTE_AT_TARGET
--drop table inv.stage_ipnetwork_copy;
-- #EXECUTE_AT_TARGET
--create table inv.stage_ipnetwork_copy like inv.stage_ipnetwork;
-- #INSERT_INTO_TARGET
--insert into inv.stage_ipnetwork_copy
--SELECT
--   inv.COMPUTER_SYS.HWARE_SYS_ID AS HWARE_SYS_ID,
--   inv.NET_NODE.NET_NODE_NAME AS NET_NODE_NAME,
--   inv.NET_NODE.NET_NODE_ADDR AS NET_NODE_ADDR
--FROM
--   inv.COMPUTER_SYS LEFT OUTER JOIN inv.NET_NODE
--      ON ( inv.COMPUTER_SYS.HWARE_SYS_ID = inv.NET_NODE.HWARE_SYS_ID )
--WHERE
--   (
--     ( inv.NET_NODE.NET_PROTOCOL = 'TCP' )
--  OR ( inv.NET_NODE.NET_PROTOCOL is null )
-- AND (( inv.NET_NODE.CFG_CHG_TYPE = 'INSERT' OR  inv.NET_NODE.CFG_CHG_TYPE =
'UPDATE' )
--  OR ( inv.NET_NODE.CFG_CHG_TYPE is null ))
--   )
--ORDER BY
--   HWARE_SYS_ID,
--   NET_NODE_NAME,
--   NET_NODE_ADDR
--;

--#EXECUTE_AT_TARGET
drop table inv.stage_ostype_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_ostype_copy like inv.stage_ostype;
--#INSERT_INTO_TARGET
insert into inv.stage_ostype_copy
SELECT
   inv.COMPUTER_SYS.HWARE_SYS_ID AS HWARE_SYS_ID,
   inv.COMPUTER_SYS.BOOTED_OS_NAME AS BOOTED_OS_NAME,
   inv.COMPUTER_SYS.BOOTED_OS_VER AS BOOTED_OS_VER,
   inv.COMPUTER_SYS.COMPUTER_KRNL_VER AS COMPUTER_KRNL_VER,
   inv.NT_INFO.NT_SVC_PACK AS NT_SVC_PACK
FROM
   inv.NT_INFO RIGHT OUTER JOIN inv.COMPUTER_SYS
      ON ( inv.NT_INFO.HWARE_SYS_ID = inv.COMPUTER_SYS.HWARE_SYS_ID )
```

```
;

--#EXECUTE_AT_TARGET
drop table inv.stage_processor_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_processor_copy like inv.stage_processor;
--#INSERT_INTO_TARGET
insert into inv.stage_processor_copy
SELECT
   i.HWARE_SYS_ID,
   count(*),
   MIN(p.PROCESSOR_MODEL),
   MIN(p.PROCESSOR_SPEED)
FROM
   inv.PROCESSOR p,
   inv.INST_PROCESSOR i
WHERE
   i.PROCESSOR_ID = p.PROCESSOR_ID
GROUP BY
   i.HWARE_SYS_ID
;

--#EXECUTE_AT_TARGET
drop table inv.stage_memory_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_memory_copy like inv.stage_memory;
--#INSERT_INTO_TARGET
insert into inv.stage_memory_copy
SELECT
   HWARE_SYS_ID,
   PHYSICAL_MEM_KB
FROM
   inv.COMPUTER_SYS_MEM
;

commit;

--#EXECUTE_AT_TARGET
drop table inv.stage_net_node_copy;
--#EXECUTE_AT_TARGET
create table inv.stage_net_node_copy like inv.stage_net_node;
--#EXECUTE_AT_TARGET
insert into inv.stage_net_node_copy
select
  value(lower(T2.NET_NODE_NAME),'') as net_node_name,
  value(T2.NET_NODE_ADDR,'') as net_node_addr,
  T1.HWARE_SYS_ID,
  value(T1.BOOTED_OS_NAME,'') as booted_os_name,
  value(T1.BOOTED_OS_VER,'') as booted_os_ver,
  case
    when T1.BOOTED_OS_NAME in ('Windows NT','Windows 2000')
       then value(T1.NT_SVC_PACK,'')
    when T1.BOOTED_OS_NAME in ('AIX', 'SunOS', 'HP-UX', 'Linux')
       then value(T1.COMPUTER_KRNL_VER,'')
    else ''
  end as os_sub_ver,
  value(T4.PHYSICAL_MEM_KB,-1) as physical_mem_kb,
  value(T6.NUM_PROCESSORS,-1) as num_processors,
  value(T6.PROCESSOR_MODEL,'') as processor_model,
  value(T6.PROCESSOR_SPEED,-1) as processor_speed,
  'Unknown' as system_purpose
FROM
  inv.stage_OSTYPE_copy T1       LEFT OUTER JOIN
  inv.stage_IPNETWORK_copy  T2
      ON     T1.HWARE_SYS_ID = T2.HWARE_SYS_ID,
  inv.stage_OSTYPE_copy T3       LEFT OUTER JOIN
  inv.stage_MEMORY_copy T4
      ON     T3.HWARE_SYS_ID = T4.HWARE_SYS_ID,
  inv.stage_OSTYPE_copy T5       LEFT OUTER JOIN
  inv.stage_PROCESSOR_copy T6
      ON     T5.HWARE_SYS_ID = T6.HWARE_SYS_ID
WHERE
  T1.HWARE_SYS_ID=T3.HWARE_SYS_ID      AND
  T1.HWARE_SYS_ID=T5.HWARE_SYS_ID
;


--#EXECUTE_AT_TARGET
--sample check for invalid data
insert into inv.invalid_data
select * from inv.stage_net_node_copy n
where
  not exists(
    select 1 from inv.cust_lookup
    where inv.cust_lookup.value = n.net_node_name OR
          inv.cust_lookup.value = '@') OR
  not exists(
    select 1 from inv.centr_lookup
    where inv.centr_lookup.value = n.net_node_name OR
          inv.centr_lookup.value = '@')
;

--Note that the insert of IP_HOST may be changed to SPP_HOST by a trigger
--  if the hostname is not fully qualified.
--#EXECUTE_AT_TARGET
INSERT INTO TWG.COMP(COMP_ID, COMPTYP_CD, CENTR_CD, CUST_ID, COMP_NM,
COMP_STRT_DTTM, COMP_END_DTTM, COMP_DS)
select
```

```
 0 as Comp_Id,
 'IP_HOST' as CompTyp_Cd ,
 centr.Centr_Cd as Centr_Cd,
 c.Cust_ID as Cust_ID,
 n.net_node_name as Comp_Nm,
 current timestamp - current timezone as Comp_Strt_DtTm ,
 '9999-01-01-00.00.00.000000' as Comp_End_DtTm,
 '' as Comp_Ds
from
 inv.stage_net_node_copy n,
 inv.cust_lookup cust,
 inv.centr_lookup centr,
 twg.cust c
where
 n.net_node_name != '' and
 not exists (
  select 1
  from
   twg.comp c
  where
   c.comp_nm=n.net_node_name
   and c.comptyp_cd IN ('SPP_HOST', 'IP_HOST')
   and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm ) AND
 ((cust.value = n.net_node_name and cust.cust_acct_cd = c.cust_acct_cd)
     OR cust.value = '@') AND
 (centr.value = n.net_node_name OR centr.value = '@')
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id,
 'INOSNM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.booted_os_name as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.booted_os_name != ''
 and not exists (
  select 1
  from
```

```
    twg.compattr ca,
    twg.comp c2
   where
    n.net_node_name=c2.comp_nm
    and ca.CompAttr_Val=n.booted_os_name
    and ca.AttrTyp_Cd='INOSNM'
    and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INOSVR' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.booted_os_ver as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.booted_os_ver != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.booted_os_ver
   and ca.AttrTyp_Cd='INOSVR'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INOSSV' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.os_sub_ver as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.os_sub_ver != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.os_sub_ver
   and ca.AttrTyp_Cd='INOSSV'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id as comp_id,
 'INADDR' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.net_node_addr as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.net_node_addr != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.net_node_addr
   and ca.AttrTyp_Cd='INADDR'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
```

```
 0 as compattr_id,
 c.comp_id,
 'INPMEM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.physical_mem_kb) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.physical_mem_kb != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.physical_mem_kb)
   and ca.AttrTyp_Cd='INPMEM'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;

--#EXECUTE_AT_TARGET
insert into twg.compattr
select
 0 as compattr_id,
 c.comp_id,
 'INCPUM' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 n.processor_model as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.processor_model != ''
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=n.processor_model
   and ca.AttrTyp_Cd='INCPUM'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INNCPU' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.num_processors) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.num_processors != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.num_processors)
   and ca.AttrTyp_Cd='INNCPU'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
union
select
 0 as compattr_id,
 c.comp_id,
 'INCPUS' as AttrTyp_Cd,
 current timestamp - current timezone as CompAttr_Strt_DtTm,
 '9999-01-01-00.00.00.000000' as CompAttr_End_DtTm,
 char(n.processor_speed) as CompAttr_Val
from
 inv.stage_net_node_copy n,
 twg.comp c
where
 c.comp_nm=n.net_node_name
 and current timestamp - current timezone between c.comp_strt_dttm and
c.comp_end_dttm
 and n.processor_speed != -1
 and not exists (
  select 1
  from
   twg.compattr ca,
   twg.comp c2
  where
   n.net_node_name=c2.comp_nm
   and ca.CompAttr_Val=char(n.processor_speed)
   and ca.AttrTyp_Cd='INCPUS'
   and current timestamp - current timezone between ca.compattr_strt_dttm and
ca.compattr_end_dttm )
;
```

C

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246607

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select **Additional materials** and open the directory that corresponds with the redbook form number, SG246607.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
SG246607.zip	Zipped Code Samples

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space	10 MB minimum
Operating system	Windows/UNIX

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Abbreviations and acronyms

AIS	ABC Information Services
API	Application programmers interface
BI	Business Intelligence
CDE	Common Desktop Environment
CDW	Central data warehouse
CIM	Common Information Model
CVS	Comma separated file
CWM	Common Warehouse Metadata
DDL	Data definition language
DM	Distributed Monitoring
DNS	Domain Name System
ETL	Extract, transform, and load
GNOME	GNU Network Object Model Environment
IBM	International Business Machines Corporation
ITSO	International Technical Support Organization
KDE	K Desktop Environment
NIS	Network Information Service
ODBC	Open Database Connectivity
ODS	Operational data source
OLAP	Online analytical processing
OLTP	Online transaction processing
RDBMS	Relational database management system
RI	Report Interface
ROLAP	Relational online analytical processing
SDC	Service Delivery Center
SPP	Server Performance Prediction
SRM	Server Resource Management
SSL	Secure Sockets Layer
TAPM	Tivoli Application Performance Management
TBSM	Tivoli Business Systems Manager
TDS	Tivoli Decision Support
TEC	Tivoli Event Console
TSLA	Tivoli Service Level Advisor
TWH	Tivoli Warehouse
TWSA	Tivoli Web Services Analyser
TWSM	Tivoli Web Services Manager
UDB	Universal Database

Related publications

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

IBM Redbooks

For information on ordering these publications, see "How to get IBM Redbooks" on page 370.

- *Business Intelligence Certification Guide*, SG24-5747
- *DB2 Warehouse Management: High Availability and Problem Determination Guide,* SG24-6544

Other resources

These publications are also relevant as further information sources:

- *DB2 UDB Quick Beginnings for UNIX,* GC09-2970
- *DB2 UDB Quick Beginnings for Windows Version 7*, GC09-2971
- *Enabling an Application for Tivoli Enterprise Data Warehouse*, GC32-0745
- *Installing and Configuring Tivoli Enterprise Data Warehouse,* GC32-0744
- *Tivoli Enterprise Data Warehouse Release Notes,* GI11-0857

Referenced Web sites

These Web sites are also relevant as further information sources:

- Brio Software Web site

  `http://www.brio.com`
- Business Objects Web site

  `http://www.businessobjects.com`
- IBM Software Support Web site

  `http://www.ibm.com/software/sysmgmt/products/support`
- J2SE download site

  `http://java.sun.com/j2se/1.3/install-solaris-patches.html`

- Open Management Group Web site

  `http://www.omg.org`
- SunSolve Web site

  `http://sunsolve.sun.com`

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Introduction to Tivoli Enterprise Data Warehouse

Introduction to Tivoli Enterprise Data Warehouse

Insider's guide to Tivoli Enterprise Data Warehouse

Best practices for creating data marts

Integration with all major OLAP tools

Tivoli Enterprise Data Warehouse is a brand new product from Tivoli, which allows customers to get cross application reports from various Tivoli and customer applications. The infrastructure enables a set of extract, transform, and load (ETL) utilities to extract and move data from Tivoli application data stores to a central data warehouse database. This redbook gives a broad understanding of the Tivoli Enterprise Data Warehouse. Some of the topics that are covered in this redbook are:

-Concepts behind the Tivoli Enterprise Data Warehouse
-Architecture and installation
-Tips for using the Report Interface
-Writing your own ETLs
-Best practices in creating data marts
-Integrating Tivoli Enterprise Data Warehouse with OLAP tools such as Brio, Business Objects, and Cognos PowerPlay
-Implementing a multi-customer environment
-Operational considerations and troubleshooting

Most of the topics are explained using real customer implementations. We think that this redbook will be a major reference for Tivoli specialists and customers who are responsible for implementing Tivoli Enterprise Data Warehouse in a real environment.

SG24-6607-00 ISBN 0738425516