CONTENTS

DATA MINING AND DATA WAREHOUSING

SYLLABUS

UNIT I

Delivery Process: Data warehousing delivery method – System process – Introduction – Overview – Typical process flow within a data warehouse – Extract and load process – Clean and transform data – Backup and archive process – Query management process.

Process Architecture: Introduction – Load manager – Warehouse manager – Query manager.

UNIT II

Introduction – Why you need tools to manage a data warehouse – System managers – Data warehouse – Process manager – Load manager – Warehouse manager – Query manager.

Capacity planning, tuning and testing – Introduction – Process – Estimating the load.

Tuning the data warehouse: Introduction – Accessing performance – Tuning the data load – Tuning queries.

UNIT III

Introduction – Basis of Data mining – Data mining versus knowledge discovery in database – Data mining Issues – Data Mining metrics – Social implications of data mining – Data mining from a database perspective.

UNIT IV

Database/OLAP system – Fuzzy sets and Fuzzy logic – Information retrieval – Decision support systems – Dimensional modeling – OLAP – Web search engines.

Data mining techniques: Introduction – A statistical Perspective on data mining – Similarity measures – Decision trees – Natural networks – Genetic algorithms.

UNIT V

Introduction – Large item sets – Basic Algorithm – Parallel and distributed algorithm – Comparing approaches – Incremental rules – Advanced association rule techniques – Measuring the quality of rules.

UNIT I

LESSON

1

DATA WAREHOUSING: SYSTEM PROCESS

CONTENTS

1.0 AIMS AND OBJECTIVE

After studying this lesson, you will be able to:

- Know meaning and characteristics of data warehouse
- Understand concept and structure of the data warehouse
- Explain granularity in data warehouse
- Describe how data is structured in warehouse
- Discuss ways of purging data
- Define reporting and the architected environment

1.1 INTRODUCTION

Data warehouse provides architectures and tools for business executives to systematically organise, understand, and use their data to make strategic decisions. In the last several years, many firms have spent millions of dollars in building enterprise-wide data warehouses as it is assumed a way to keep customers by learning more about their needs.

In simple terms, a data warehouse refers to a database that is maintained separately from an organisation's operational databases. Data warehouse systems allow for the integration of a variety of application systems. They support information processing by providing a solid platform of consolidated, historical data for analysis.

1.2 DATA WAREHOUSE

There are two major components to building a data warehouse: the design of the interface from operational systems and the design of the data warehouse itself. Yet, "*design*" is not entirely accurate because it suggests planning elements out in advance. The requirements for the data warehouse cannot be known until it is partially populated and in use and design approaches that have worked in the past will not necessarily suffice in subsequent data warehouses. Data warehouses are constructed in a heuristic manner, where one phase of development depends entirely on the results attained in the previous phase. First, one portion of data is populated. It is then used and scrutinized by the DSS analyst. Next, based on feedback from the end user, the data is modified and/or other data is added. Then another portion of the data warehouse is built, and so forth. This feedback loop continues throughout the entire life of the data warehouse.

A data warehouse is a subject-oriented, integrated, non volatile and time-variant collection of data in support of management's decisions. The data warehouse contains granular corporate data.

The subject orientation of the data warehouse is shown in Figure 1.1. Classical operations systems are organized around the applications of the company. For an insurance company, the applications may be auto, health, life, and casualty. The major subject areas of the insurance corporation might be customer, policy, premium, and claim. For a manufacturer, the major subject areas might be product, order, vendor, bill of material, and raw goods. For a retailer, the major subject areas may be product, SKU, sale, vendor, and so forth. Each type of company has its own unique set of subjects.

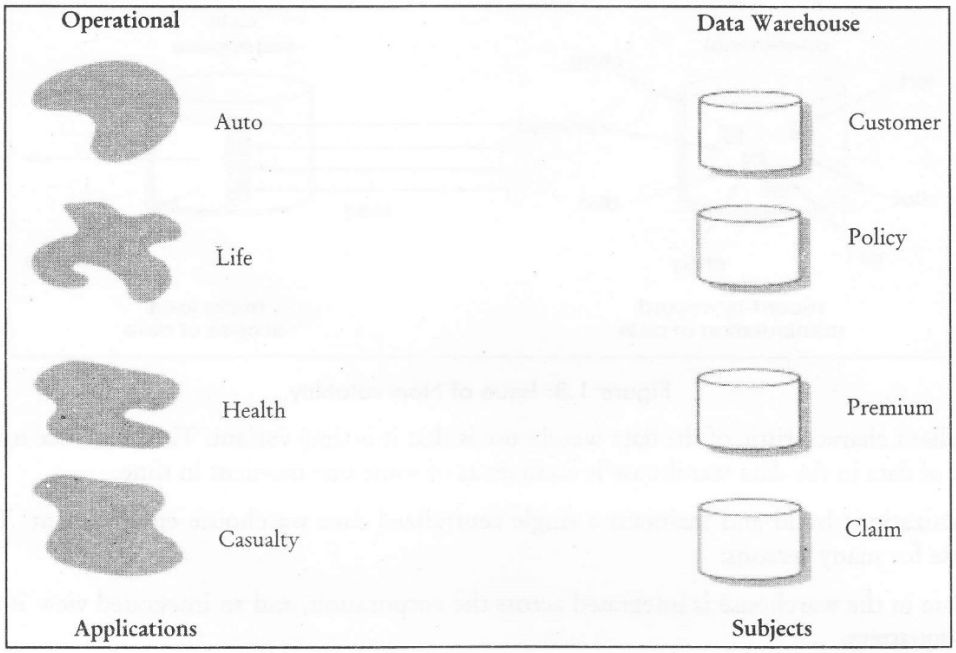


Figure 1.1: An Example of a Subject Orientation of Data

The second salient characteristic of the data warehouse is that it is integrated. Of all the aspects of a data warehouse, integration is the most important. Data is fed from multiple disparate sources into the data warehouse. As the data is fed it is converted, reformatted, resequenced, summarized, and so forth. The result is that data — once it resides in the data warehouse — has a single physical corporate image. Figure 1.2 illustrates the integration that occurs when data passes from the application-oriented operational environment to the data warehouse. Data is entered into the data warehouse in such a way that the many inconsistencies at the application level are undone. For example, in Figure 1.2, as far as encoding of gender is concerned, it matters little whether data in the warehouse is encoded as m/f or 1/0. What does matter is that regardless of method or source application, warehouse encoding is done consistently. If application data is encoded as X/Y, it is converted as it is moved to the warehouse.

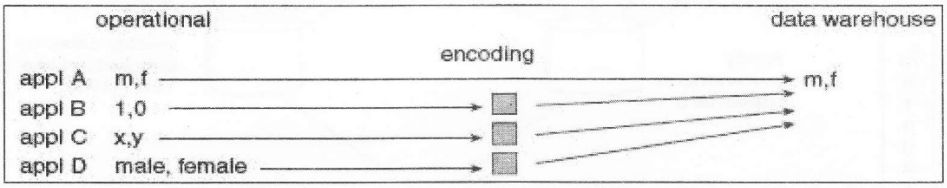


Figure 1.2: Issue of Integration

The third important characteristic of a data warehouse is that it is nonvolatile. Figure 1.3 illustrates non volatility of data and shows that operational data is regularly accessed and manipulated one record at a time. Data is updated in the operational environment as a regular matter of course, but data warehouse data exhibits a very different set of characteristics. Data warehouse data is loaded (usually en masse) and accessed, but it is not updated (in the general sense). Instead, when data in the data warehouse is loaded, it is loaded in a snapshot, static format. When subsequent changes occur, a new snapshot record is written. In doing so a history of data is kept in the data warehouse.

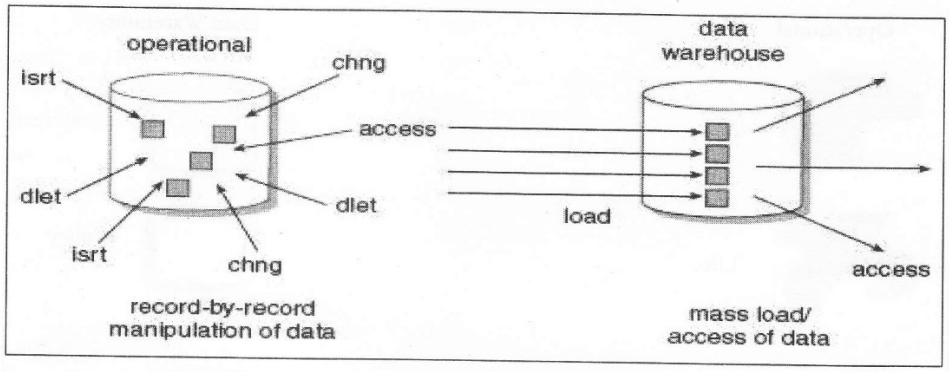


Figure 1.3: Issue of Non-volatility

The last salient characteristic of the data warehouse is that it is time variant. Time variance implies that every unit of data in the data warehouse is accurate as of some one moment in time.

Most organizations build and maintain a single centralized data warehouse environment. This setup makes sense for many reasons:

- The data in the warehouse is integrated across the corporation, and an integrated view is used only at headquarters.
- The corporation operates on a centralized business model.
- The volume of data in the data warehouse is such that a single centralized repository of data makes sense.
- Even if data could be integrated, if it were dispersed across multiple local sites, it would be cumbersome to access.

1.2.1 Structure of the Data Warehouse

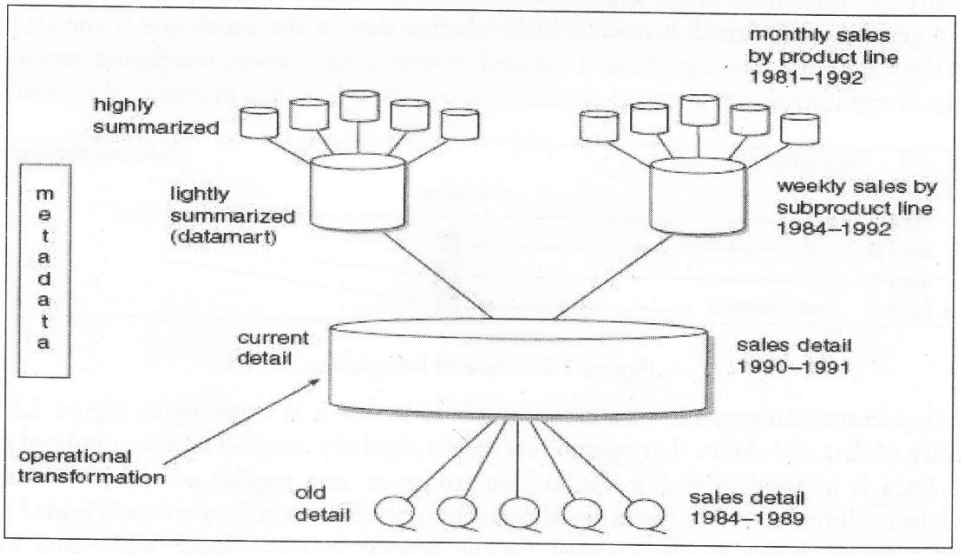


Figure 1.4: Structure of the Data Warehouse

Figure 1.4 shows that there are different levels of detail in the data warehouse. There is an older level of detail (usually on alternate, bulk storage), a current level of detail, a level of lightly summarized data (the data mart level), and a level of highly summarized data. *Data flows into the data warehouse from the operational environment.* Usually significant transformation of data occurs at the passage from the operational level to the data warehouse level. Once the data ages, it passes from current detail to older detail. As the data is summarized, it passes from current detail to lightly summarized data, then from lightly summarized data to highly summarized data.

1.2.2 Granularity

The single most important aspect of design of a data warehouse is the issue of granularity. Indeed, the issue of granularity permeates the entire architecture that surrounds the data warehouse environment. Granularity refers to the level of detail or summarization of the units of data in the data warehouse. The more detail there is, the lower the level of granularity. The less detail there is, the higher the level of granularity. For example, a simple transaction would be at a low level of granularity. A summary of all transactions for the month would be at a high level of granularity. Granularity of data has always been a major design issue. In early operational systems, granularity was taken for granted. When detailed data is being updated, it is almost a given that data be stored at the lowest level of granularity.

In almost all cases, data comes into the data warehouse at too high a level of granularity. This means that the developer must spend a lot of resources breaking the data apart. Occasionally, though, data enters the warehouse at too low a level of granularity.

Benefits of Granularity

The granular data found in the data warehouse is the key to reusability, because it can be used by many people in different ways.

For example, within a corporation, the same data might be used to satisfy the needs of marketing, sales, and accounting. All three departments look at the basic same data. Marketing may want to see sales on a monthly basis by geographic district, sales may want to see sales by salesperson by sales district on a weekly basis, and finance may want to see recognizable revenue on a quarterly basis by product line. All of these types of information are closely related, yet slightly different. With a data warehouse, the different organizations are able to look at the data as they wish to see it.

Looking at the data in different ways is only one advantage of having a solid foundation. A related benefit is the ability to reconcile data, if needed. Once there is a single foundation on which everyone relies, if there is a need to explain a discrepancy in analyses between two or more departments, then reconciliation is relatively simple.

Another benefit of granular data is that it contains a history of activities and events across the corporation. And the level of granularity is detailed enough that the data can be reshaped across the corporation for many different needs.

The largest benefit of a data warehouse foundation is that future unknown requirements can be accommodated.

Dual Levels of Granularity

Most of the time, there is a great need for efficiency in storing and accessing data, and for the ability to analyze data in great detail. (In other words, the organization wants to have its cake and eat it, too!)

When an organization has lots of data in the warehouse, it makes eminent sense to consider two (or more) levels of granularity in the detailed portion of the data warehouse. In fact, there is such a need for more than one level of granularity that a dual level of granularity design should be the default for almost every shop.

For example, a phone company—fits the needs of most shops. There is a tremendous amount of detail at the operational level. Most of this detail is needed for the billing systems. Up to 30 days of detail is stored in the operational level. The data warehouse in this example contains two types of data-lightly summarized data and "true archival" detail data. The data in the data warehouse can go back 10 years. The data that emanates from the data warehouse is "district" data that flows to the different districts of the telephone company. Each district then analyzes its data independently from other districts. Much heuristic analytical processing occurs at the individual level.

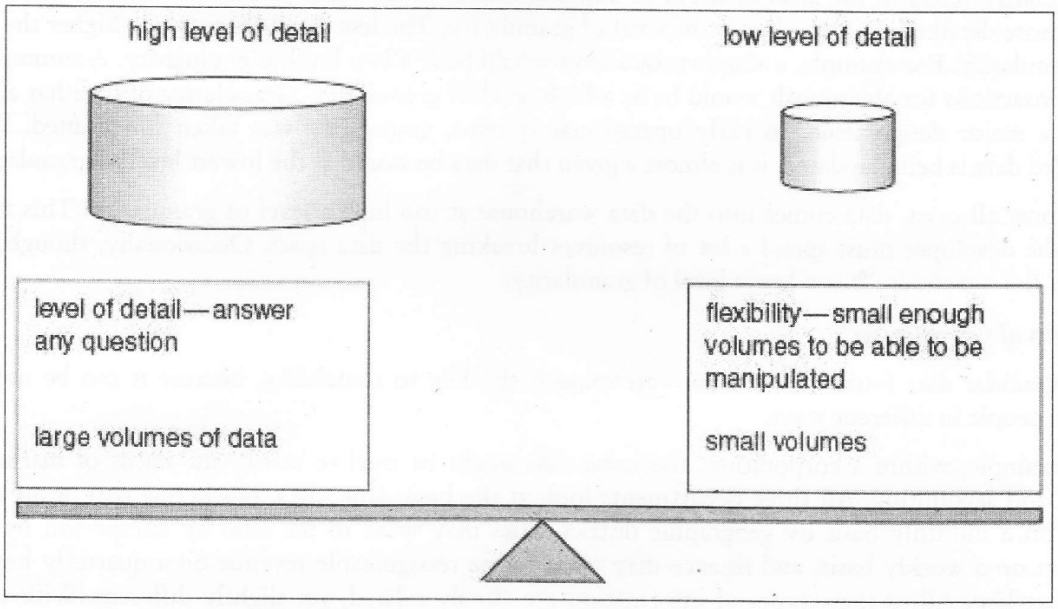


Figure 1.5: The Trade-off with Granularity is so Stark that for most Organizations the best Solution is some Form of Multiple Levels of Granularity

1.2.3 Structuring Data in the Data Warehouse

So far, we haven't gone into what the data structures found in the data warehouse really look like. Many kinds of structures are found in the data warehouse. We will look at some of the more common ones now. Perhaps the simplest and most common data structure found in the data warehouse is the simple cumulative structure, shown in Figure 1.6 shows the daily transactions being transported from the operational environment. After that, they are summarized into data warehouse records, which may be by customer, by account, or by any subject area in the data warehouse. The transactions in Figure 1.6 are summarized by day. In other words, all daily activity for a customer for an account are totaled and passed into the data warehouse on a day-by-day basis.

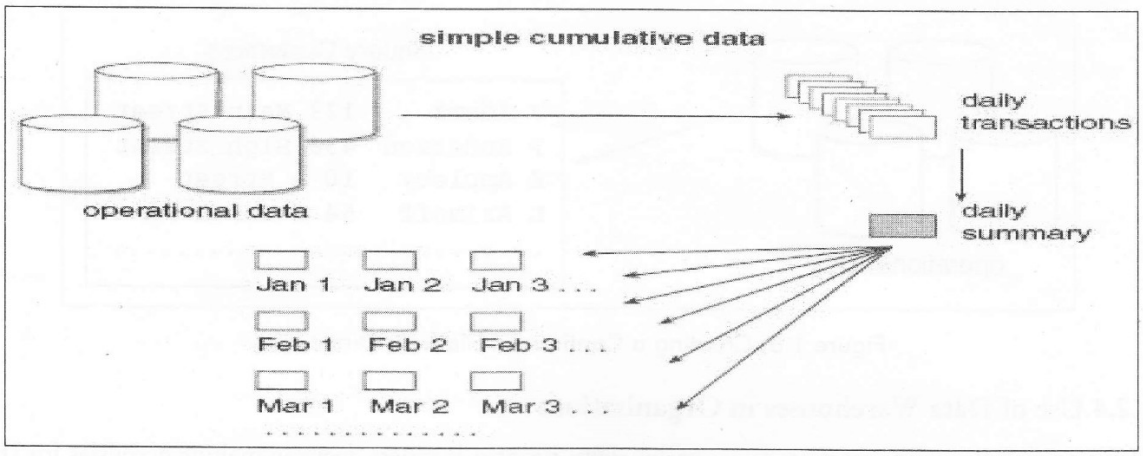


Figure 1.6: The Simplest Form of Data in the Data Warehouse is the Data that has been accumulated on a Record-by-record Basis, called Simple Cumulative Data

A rolling summary data structure handles many fewer units of data than does a simple cumulative structuring of data. A comparison of the advantages and the disadvantages of rolling summary versus simple cumulative structuring of data is shown in Figure 1.7

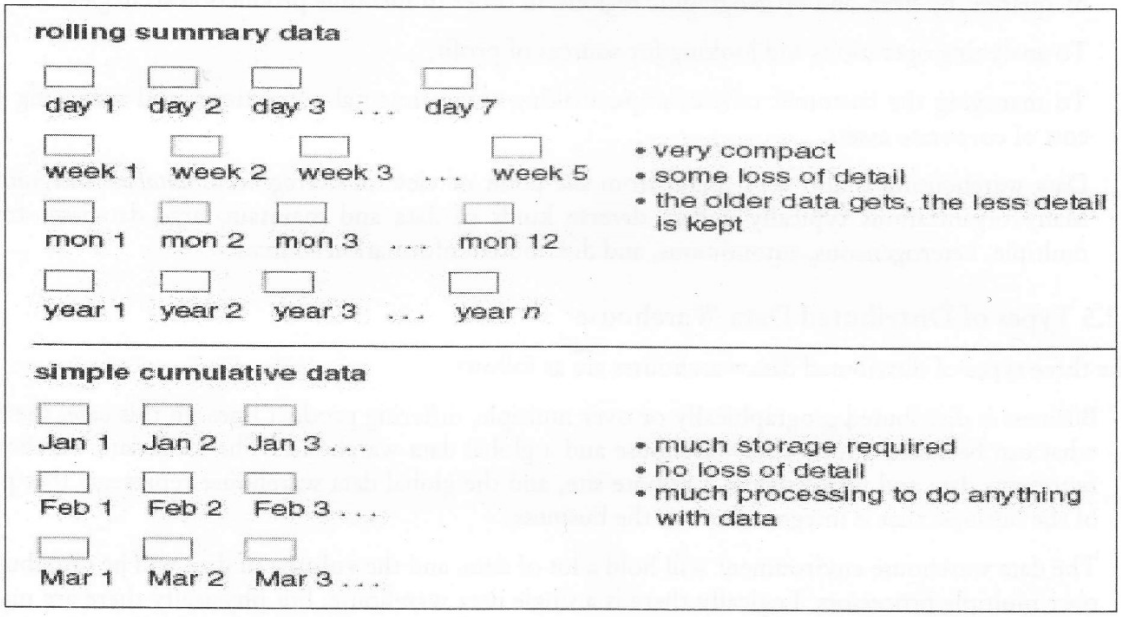


Figure 1.7: Comparing Simple Cumulative Data with Rolling Summary Data

Another possibility for the structuring of data warehouse data is the simple direct file, shown in Figure 1.8. Figure 1.8 shows that data is merely pulled from the operational environment to the data warehouse environment; there is no accumulation. In addition, the simple direct file is not done on a daily basis. Instead, it is done over a longer period of time, such as a week or a month. As such, the simple direct file represents a snapshot of operational data taken as of one instant in time.

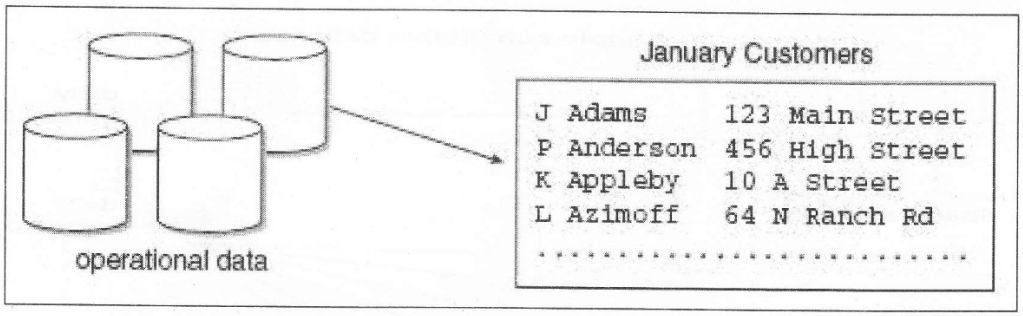


Figure 1.8: Creating a Continuous File from Direct Files

1.2.4 Use of Data Warehouses in Organisations

Many organizations are creating data warehouse to support business decision-making activities for the following reasons:

1. To increasing customer focus, which includes the analysis of customer buying patterns (such as buying preference, buying time, budget cycles, and appetites for spending),
2. To reposition products and managing product portfolios by comparing the performance of sales by quarter, by year, and by geographic regions, in order to fine-tune production strategies,
3. To analyzing operations and looking for sources of profit,
4. To managing the customer relationships, making environmental corrections, and managing the cost of corporate assets,
5. Data warehousing is also very useful from the point of view of *heterogeneous database integration.* Many organizations typically collect diverse kinds of data and maintain large databases from multiple, heterogeneous, autonomous, and distributed information sources.

1.2.5 Types of Distributed Data Warehouse

The three types of distributed data warehouses are as follows:

- Business is distributed geographically or over multiple, differing product lines. In this case, there is what can be called a local data warehouse and a global data warehouse. The local data warehouse represents data and processing at a remote site, and the global data warehouse represents that part of the business that is integrated across the business.
- The data warehouse environment will hold a lot of data, and the **volume** of data will be distributed over multiple processors. Logically there is a single data warehouse, but physically there are many data warehouses that are all tightly related but reside on separate processors. This configuration can be called the technologically distributed data warehouse.
- The data warehouse environment grows up in an uncoordinated manner — first one data warehouse appears, then another. The lack of coordination of the growth of the different data warehouses is usually a result of political and organizational differences. This case can be called the independently evolving distributed data warehouse.

Local Data Warehouse

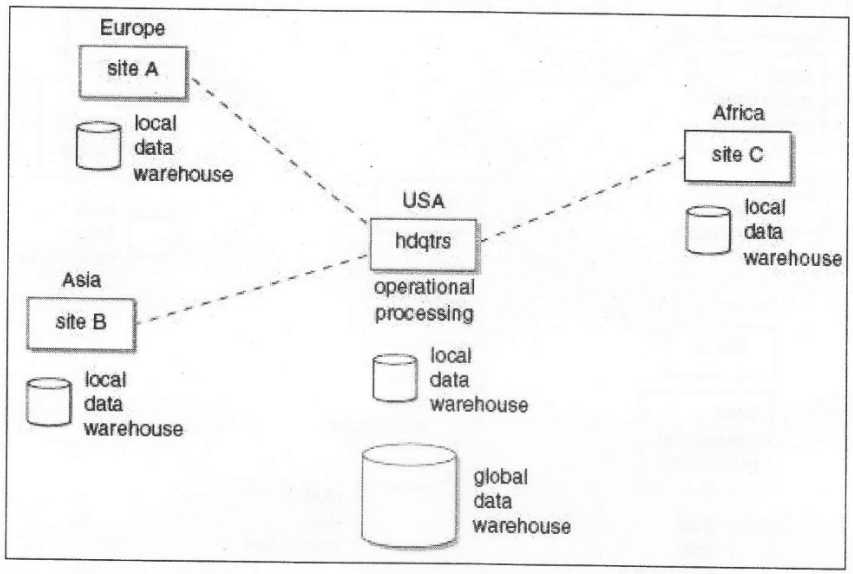


Figure 1.9: Local Data Warehouse

A form of data warehouse, known as a local data warehouse, contains data that is of interest only to the local level. There might be a local data warehouse for Brazil, one for France, and one for Hong Kong. Or there might be a local data warehouse for car parts, motorcycles, and heavy trucks. Each local data warehouse has its own technology, its own data, its own processor, and so forth. In Figure 1.9, a local data warehouse exists for different geographical regions or for different technical communities. The local data warehouse serves the same function that any other data warehouse serves, except that the scope of the data warehouse is local. For example, the data warehouse for Brazil does not have any information about business activities in France.

Global Data Warehouse

Of course, there can also be a global data warehouse, as shown in Figure 1.10. The global data warehouse has as its scope the corporation or the enterprise, while each of the local data warehouses within the corporation has as its scope the local site that it serves. For example, the data warehouse in Brazil does not coordinate or share data with the data warehouse in France, but the local data warehouse in Brazil does share data with the corporate headquarters data warehouse in Chicago. The scope of the global data warehouse is the business that is integrated across the corporation. In some cases, there is considerable corporate integrated data; in other cases, there is very little. The global data warehouse contains historical data, as do the local data warehouses.

The global data warehouse contains information that must be integrated at the corporate level. In many cases, this consists only of financial information. In other cases, this may mean integration of customer information, product information, and so on. While a considerable amount of information will be peculiar to and useful to only the local level, other corporate common information will need to be shared and managed corporately. The global data warehouse contains the data that needs to be managed globally.

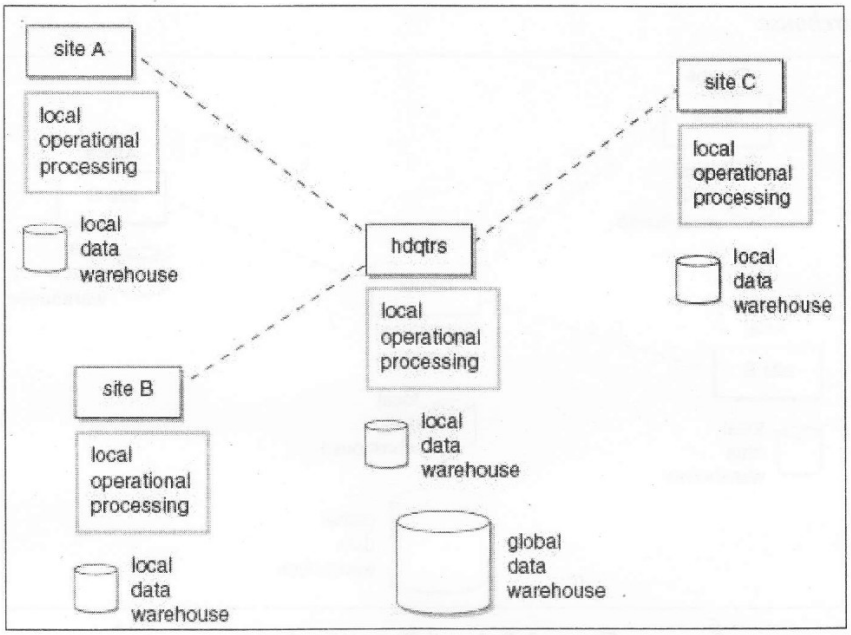


Figure 1.10: What a Typical Distributed Data Warehouse might look like

Intersection of Global and Local Data

Figure 1.11 shows that data is being fed from the local data warehouse environment to the global data warehouse environment. The data may be carried in both warehouses, and a simple transformation of data may occur as the data is placed in the global data warehouse.

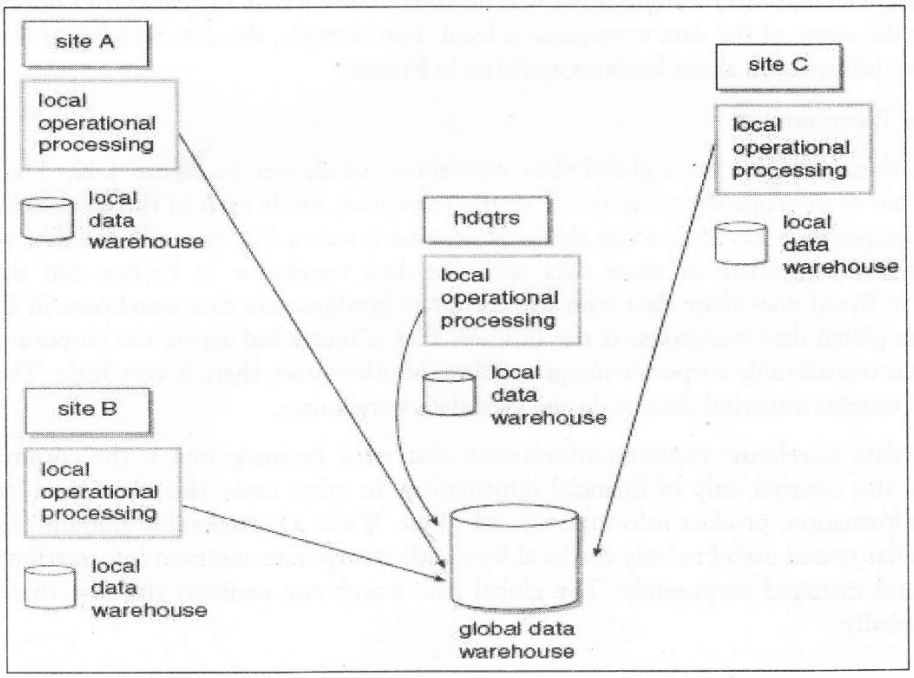


Figure 1.11: Intersection of Global and Local Data

Example: One local data warehouse may carry its information in the Hong Kong dollar but convert to the U.S. dollar on entering the global data warehouse. Or the French data warehouse may carry parts specifications in metric in the French data warehouse but convert metric to English measurements on entering the global data warehouse.

Redundancy

One of the issues of a global data warehouse and its supporting local data warehouses is redundancy or overlap of data on occasion, some detailed data will pass through to the global data warehouse untouched by any transformation or conversion In this case, a small overlap of data from the global data warehouse to the local data warehouse will occur.

A massive amount of redundancy of data between the local and the global data warehouse environments indicates that the scopes of the different warehouses probably have not been defined properly. When massive redundancy of data exists between the local and the global data warehouse environments, it is only a matter of time before spider web systems start to appear. With the appearance of such systems come many problems—reconciliation of inconsistent results, inability to create new systems easily, costs of operation, and so forth. For this reason, it should be a matter of policy that global data and local data be mutually exclusive with the exception of very small amounts of data that incidentally overlap between the two environments.

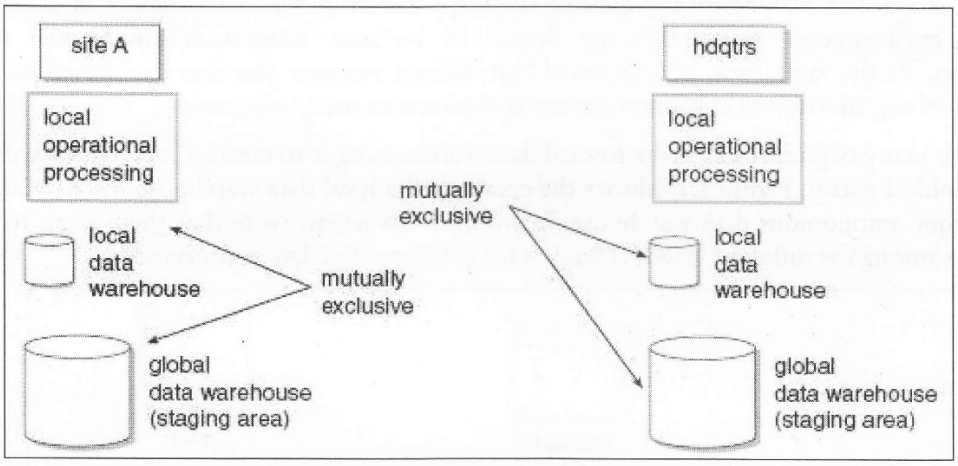


Figure 1.12: Data can exist in either the Local Data Warehouse or the Global Data Warehouse, but not both

Technologically Distributed Data Warehouse

The Data warehouse can be distributed technologically. Different type of a distributed warehouse is that of placing a data warehouse on the distributed technology of a vendor. Client/server technology fits this requirement nicely. The first advantage of a technologically distributed data warehouse is that the entry cost is cheap. The second advantage is that there is no theoretical limit to how much data can be placed in the data warehouse. If the volume of data inside the warehouse begins to exceed the limit of a single distributed processor, then another processor can be added to the network, and the progression of adding data continues in an unimpeded fashion. Whenever the data grows too large, another processor is added.

Example: Suppose one processor holds data for the year 1998, another processor for 1999, another for 2000, and yet another for 2001. When a request is made for data from 1998 to 2001, the result set for that query must pass over the boundaries of the processor that originally held the data. In this case, data from four processors must be gathered. In the process, data passes across the network and increases the traffic.

The Independently Evolving Distributed Data Warehouse

Yet a third flavor of distributed data warehouse is one in which independent data warehouses are developed concurrently and in an uncontrolled manner. The first step many corporations take in data warehousing is to put up a data warehouse for a financial or marketing organization. Once success follows the initial warehouse effort, other parts of the corporation naturally want to build on the successful first effort. In short order, the data warehouse architect has to manage and coordinate multiple data warehouse efforts within the organization.

1.2.6 Distributed Data Warehouse Development

A corporation has different sites in different parts of the world—one in the United States and Canada, one in South America, one in the Far East, and one in Africa. Each site has its own unique data, with no overlap of data—particularly of detailed transaction data—from one site to the next. The company wants to create a data warehouse for each of the disparate entities as a first effort in achieving an architected environment. There is some degree of business integration among the different organizations. At the same time, it is assumed that distinct business practices are carried on in each locale. Such an organization of corporate entities is common to many companies.

The first step many organizations make toward data warehousing is to create a local data warehouse at each geographical entity. Figure 1.13 shows the creation of a local data warehouse. Each locale builds its own unique autonomous data warehouse according to its needs. Note that there is no redundant detailed data among the different locals, at least as far as transaction data is concerned.

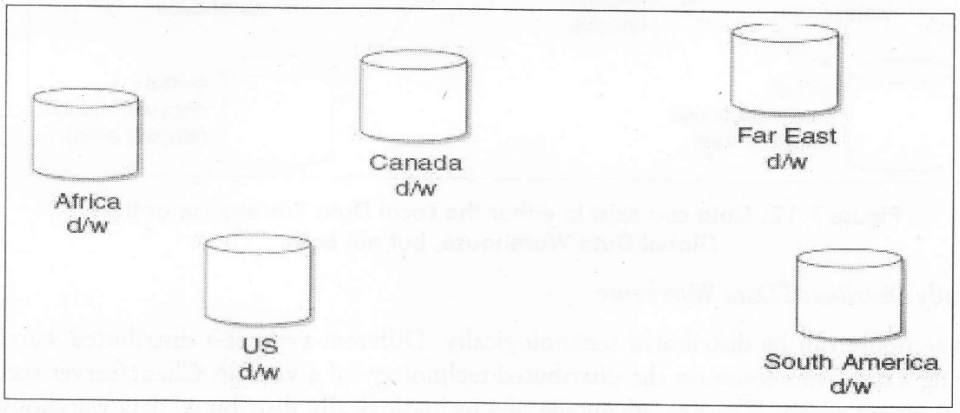


Figure 1.13: Local Data Warehouses are built at each of the Autonomous Operating Divisions

There are several pros and cons to this approach to building the distributed corporate data warehouse. One advantage is that it is quick to accomplish. Each local group has control over its design and resources.

Coordinating Development across Distributed Locations

An alternative approach is to try to coordinate the local data warehouse development efforts across the different local organizations. A separate data model is built to provide the foundation of the data warehouse design for each of the separate locales. The corporation decides to build a corporate data warehouse (see Figure 1.14).

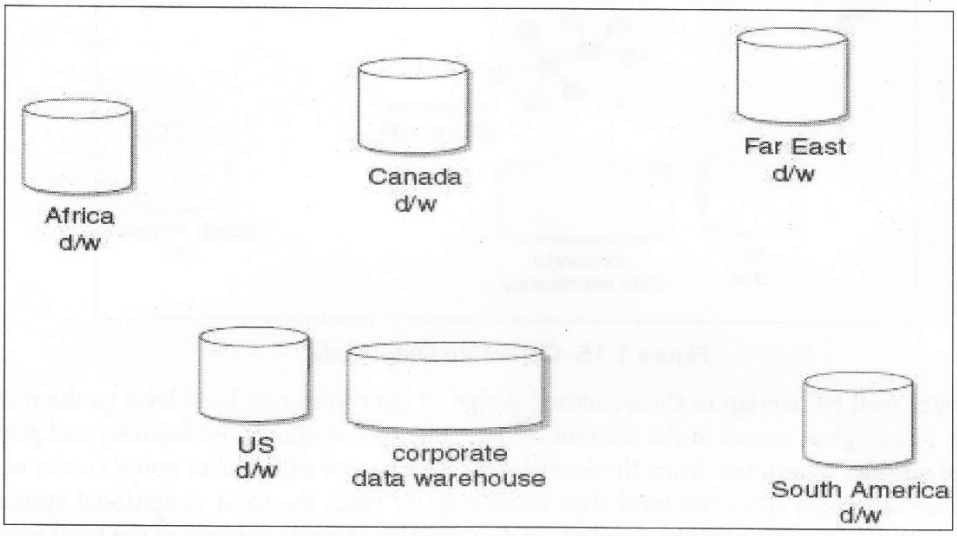


Figure 1.14: Coordinating Development across Distributed Locations

The corporate data warehouse will reflect the business integration across the different divisions and locales. The corporate data warehouse will be related to, but still distinct from, the local data warehouses. The first step in building the corporate data warehouse is to create a corporate data model for that portion of the business that will be reflected in the corporate data warehouse. As a general rule, the corporate data model that is built for the first iteration of the corporate data warehouse will be small and simple, and it will be limited to a subset of the business.

The Corporate Data Model Distributed

The corporate data warehouse will reflect the business integration across the different divisions and locales. The corporate data warehouse will be related to, but still distinct from, the local data warehouses. The first step in building the corporate data warehouse is to create a corporate data model for that portion of the business that will be reflected in the corporate data warehouse. As a general rule, the corporate data model that is built for the first iteration of the corporate data warehouse will be small and simple, and it will be limited to a subset of the business. Figure 1.15 illustrates the building of the corporate data model after which the corporate data warehouse will be shaped.

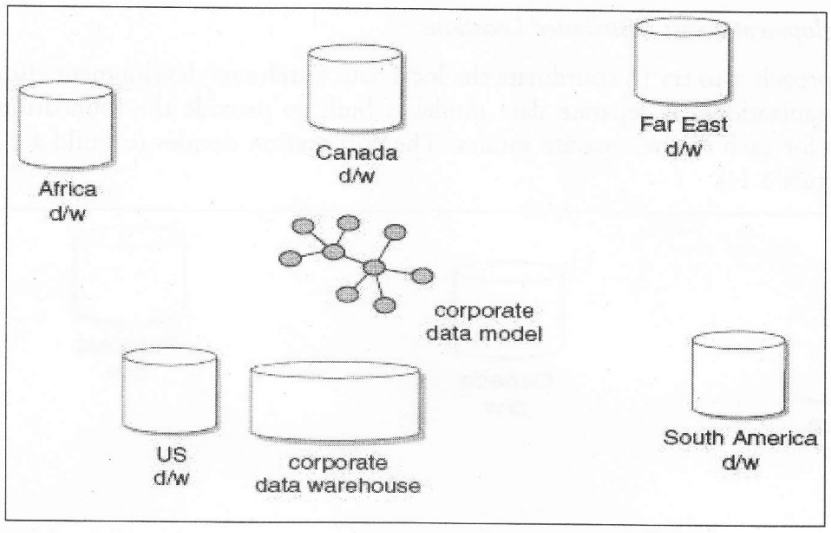


Figure 1.15: Corporate Data Model

There may very well be overlap in the structural design of data from one local level to the next, there is no overlap to any great extent in the content of data. Figure 1.16 shows the building and population of the corporate data warehouse from the local levels. The source of the data going to the corporate data warehouse can come from the local data warehouse or from the local operational systems. The determination of the system of record should be a decision that is made entirely at the local level.

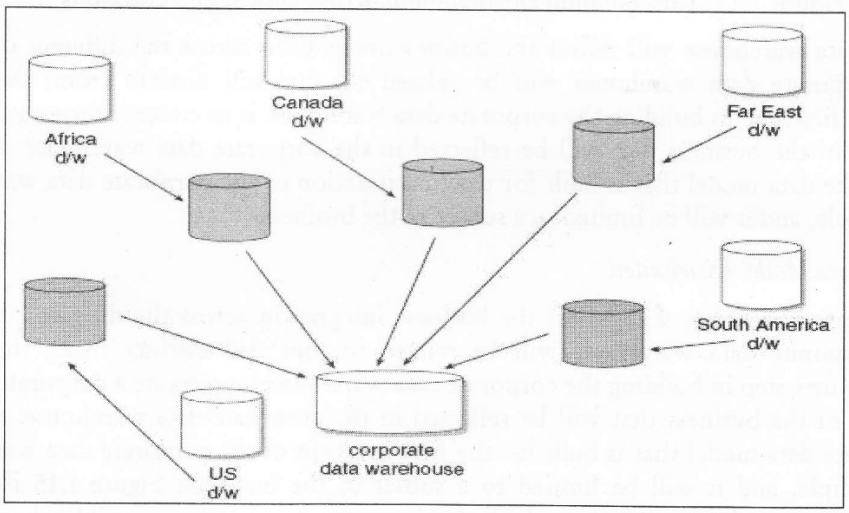


Figure 1.16: Corporate Data Warehouse is Loaded from the Different Autonomous Operating Companies

The corporate data warehouse of the distributed database can be contrasted with the corporate financial data warehouse of the completely unrelated companies. Figure 1.17 makes this comparison. In many ways, the data warehouse of the distributed corporation is very similar to the data warehouse of the unrelated companies. However, although there are similarities in design and operation, there is one major difference. The corporate distributed data warehouse extends into the business itself, reflecting the integration of customer, vendor, product, and so forth.

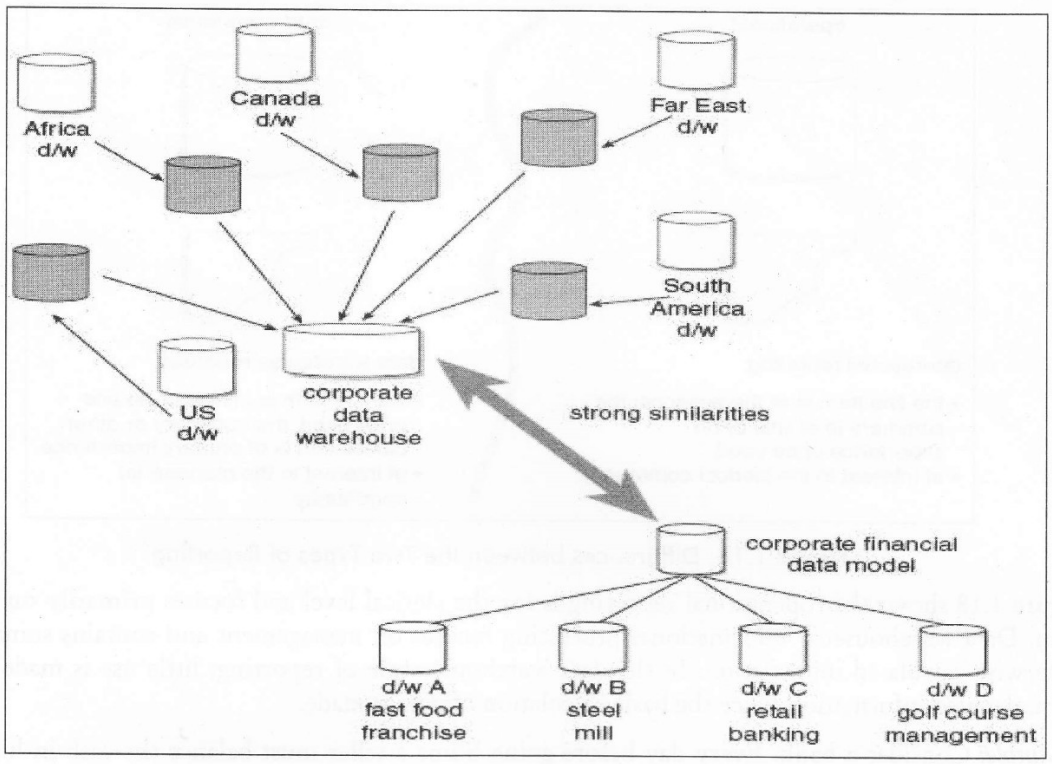


Figure 1.17

Meta Data in the Distributed Warehouse

Meta data plays a very important role across the distributed corporate data warehouse. It is through meta data that the coordination of the structure of data is achieved across the many different locations where the data warehouse is found. Not surprisingly, meta data provides the vehicle for the achievement of uniformity and consistency.

1.3 REPORTING AND THE ARCHITECTED ENVIRONMENT: DELIVERY PROCESS

It is a temptation to say that once the data warehouse has been constructed all reporting and informational processing will be done from there. That is simply not the case. There is a legitimate class of report processing that rightfully belongs in the domain of operational systems. Figure 1.18 shows where the different styles of processing should be located.

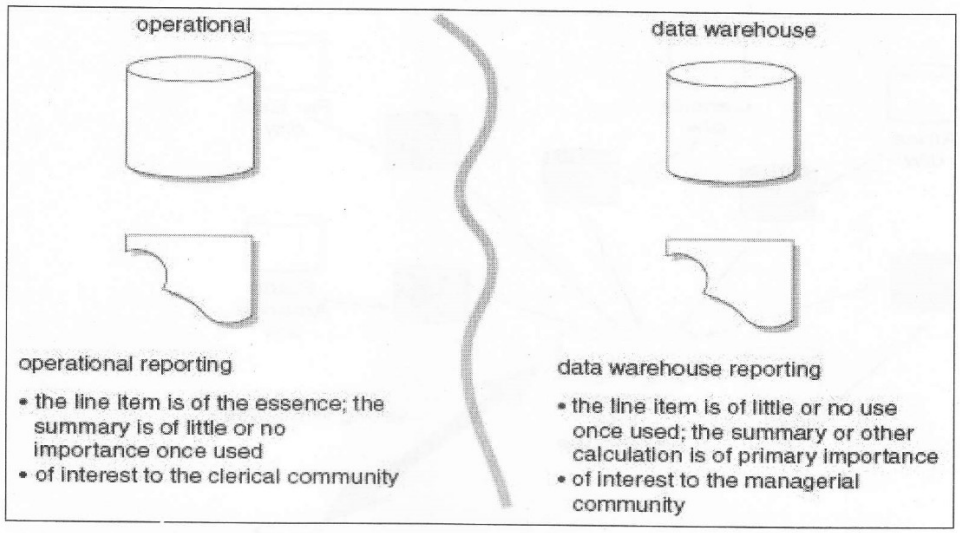


Figure 1.18: Differences between the Two Types of Reporting

Figure 1.18 shows that operational reporting is for the clerical level and focuses primarily on the line item. Data warehouse or informational processing focuses on management and contains summary or otherwise calculated information. In the data warehouse style of reporting, little use is made of line-item, detailed information, once the basic calculation of data is made.

Example: Consider a bank. Every day before going home a teller must balance the cash in his or her window. This means that the teller takes the starting amount of cash, tallies all the day's transactions, and determines what the day's ending cash balance should be. In order to do this, the teller needs a report of all the day's transactions. This is a form of operational reporting. Now consider the bank vice president who is trying to determine how many new ATMs to place in a newly developed shopping center. The banking vice president looks at a whole host of information, some of which comes from within the bank and some of which comes from outside the bank. The bank vice president is making a long-term, strategic decision and uses classical DSS information for his or her decision.

There is then a real difference between operational reporting and DSS reporting. Operational reporting should always be done within the confines of the operational environment.

1.4 SYSTEM PROCESS

Data warehouse Architecture is divided into:

- System Process
- Process Architecture

Here we will discuss system process. Process Architecture is discussed in next lesson.

System process is basically defined by ELT Process which is known as Extract, load, and Transform process.

These processes are discussed in subsequent sections.

Beginning with Operational Data

At the outset, operational transaction-oriented data is locked up in existing legacy systems. Though tempting to think that creating the data warehouse involves only extracting operational data and entering it into the warehouse, nothing could be further from the truth. Merely pulling data out of the legacy environment and placing it in the data warehouse achieves very little of the potential of data warehousing.

Figure 1.19 shows a simplification of how data is transferred from the existing legacy systems environment to the data warehouse. We see here that multiple applications contribute to the data warehouse. Most importantly, it does not take into account that the data in the operational environment is unintegrated.

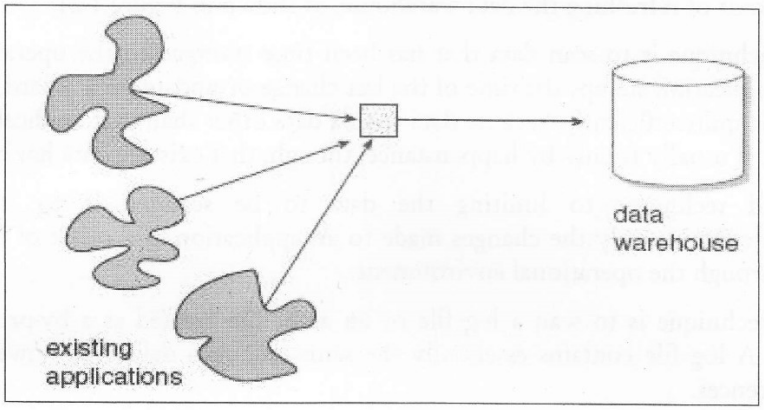


Figure 1.19: Moving from the Operational to the Data Warehouse Environment is not as Simple as mere Extraction

Figure 1.20 shows the lack of integration in a typical existing systems environment. Pulling the data into the data warehouse without integrating it is a grave mistake.

When the existing applications were constructed, no thought was given to possible future integration. Each application had its own set of unique and private requirements. It is no surprise, then, that some of the same data exists in various places with different names, some data is labeled the same way in different places, some data is all in the same place with the same name but reflects a different measurement, and so on.

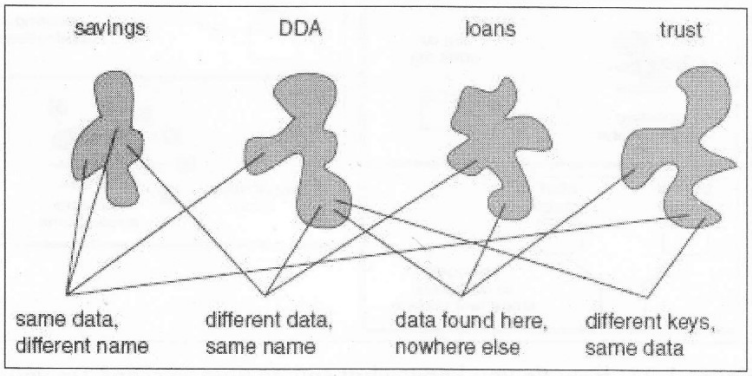


Figure 1.20: Data across the Different Applications is Severely Unintegrated

Three types of loads are made into the data warehouse from the operational environment:

- Archival data
- Data currently contained in the operational environment
- Ongoing changes to the data warehouse environment from the changes (updates) that have occurred in the operational environment since the last refresh.

As a rule, loading **archival data** from the legacy environment as the data warehouse is first loaded presents a minimal challenge for two reasons. First, it often is not done at all. Organizations find the use of old data not cost-effective in many environments. Second, even when archival data is loaded; it is a one time only event. Five common techniques are used to limit the amount of operational data scanned at the point of refreshing the data warehouse, as shown in Figure 1.21

1. The first technique is to scan data that has been time stamped in the operational environment. When an application stamps the time of the last change or update on a record, the data warehouse scan can run quite efficiently because data with a date other than that applicable does not have to be touched. It usually is only by happenstance, though, that existing data has been time stamped.
2. The second technique to limiting the data to be scanned is to scan a "delta" file. A delta file contains only the changes made to an application as a result of the transactions that have run through the operational environment.
3. The third technique is to scan a log file or an audit file created as a by-product of transaction processing. A log file contains essentially the same data as a delta file; however, there are some major differences.
4. The fourth technique for managing the amount of data scanned is to modify application code.
5. The last option (in most respects, a hideous one, mentioned primarily to convince people that there must be a better way) is rubbing a "before" and an "after" image of the operational file together. In this option, a snapshot of a database is taken at the moment of extraction.

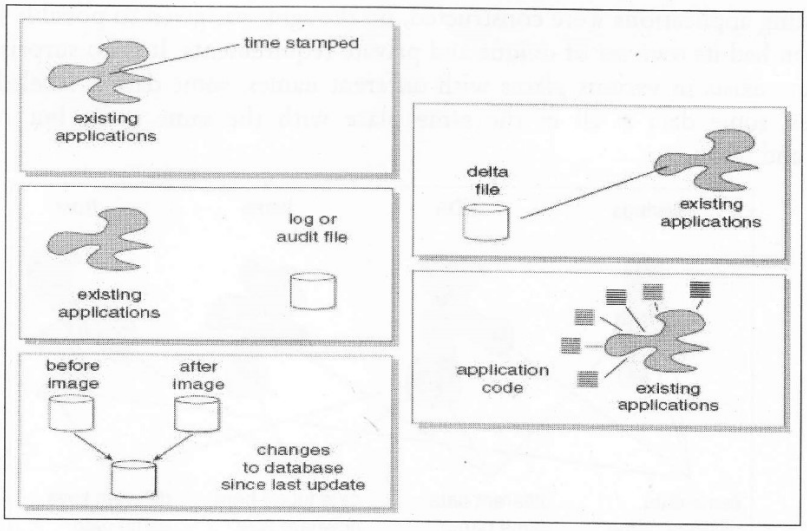


Figure 1.21: How do you know what source data to scan? Do you scan every in centre record every day/every week?

Archiving External Data

Every piece of information—external or otherwise—has a useful lifetime. Once past that lifetime, it is not economical to keep the information. An essential part of managing external data is deciding what the useful lifetime of the data is. Even after this is determined, there remains the issue of whether the data should be discarded or put into archives. As a rule, external data may be removed from the data warehouse and placed on less expensive storage. The meta data reference to the external data is updated to reflect the new storage place and is left in the meta data store. The cost of an entry into the meta data store is so low that once put there, it is best left there.

1.4.1 Data/Process Models and the Architected Environment

Before attempting to apply conventional database design techniques, the designer must understand the applicability and the limitations of those techniques.

1. The process model applies only to the operational environment.
2. The data model applies to both the operational environment and the data warehouse environment.

In general there are two types of models for the information systems environment— data models and process models. Data models are discussed in depth in the following section. For now, we will address process models. A process model typically consists of the following (in whole or in part):

- Functional decomposition
- Context-level zero diagram
- Data flow diagram
- Structure chart
- State transition diagram
- HIPO chart
- Pseudocode

There are many contexts and environments in which a process model is invaluable for instance, when building the data mart. However, because the process model is requirements-based, it is not suitable for the data warehouse. The process model assumes that a set of known processing requirements exists a priori—before the details of the design are established. With processes, such an assumption can be made. But those assumptions do not hold for the data warehouse. Many development tools, such as CASE tools, have the same orientation and as such are not applicable to the data warehouse environment.

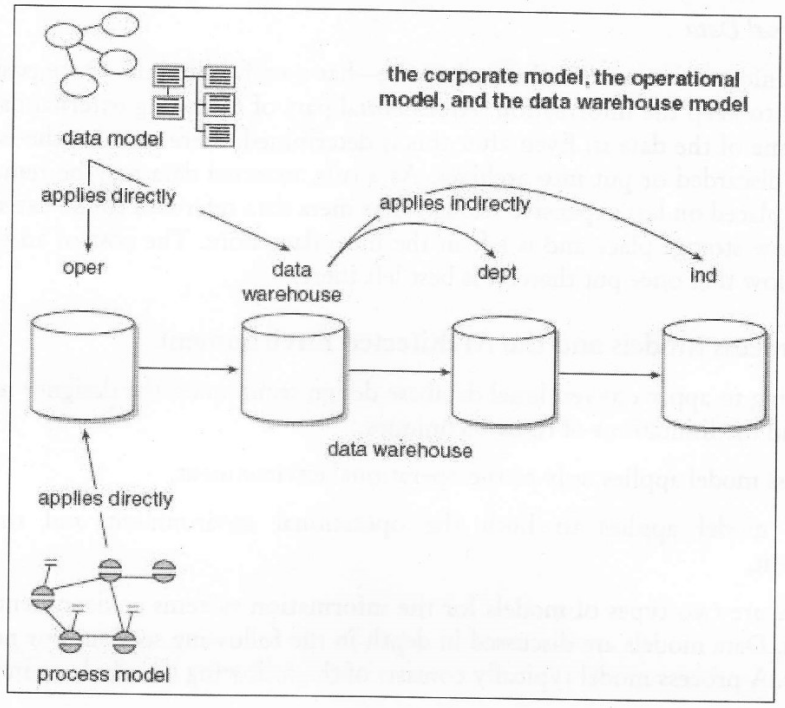


Figure 1.22: How the Different Types of Models apply to the Architected Environment

1.4.2 Data Warehouse and Data Models

The data model is applicable to both the existing systems environment and the data warehouse environment.

There are three levels of data modeling: high-level modeling (called the ERD, entity relationship level), midlevel modeling (called the data item set, or DIS), and low-level modeling (called the physical model).

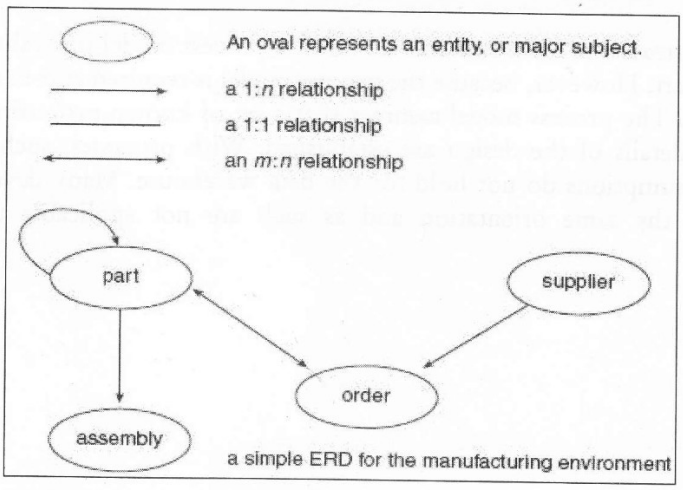


Figure 1.23 : Representing Entities and Relationships

High Level Data

The high level of modeling features entities and relationships, as shown in Figure 1.23. The name of the entity is surrounded by an oval. Relationships among entities are depicted with arrows. The direction and number of the arrowheads indicate the cardinality of the relationship, and only direct relationships are indicated. In doing so, transitive dependencies are minimized. The entities that are shown in the ERD level are at the highest level of abstraction.

Midlevel Data Model

After the high-level data model is created, the next level is established the midlevel model, or the DIS. For each major subject area, or entity, identified in the high-level data model, a midlevel model is created, as seen in Figure 1.24. The high-level data model has identified four entities, or major subject areas. Each area is subsequently developed into its own midlevel model.

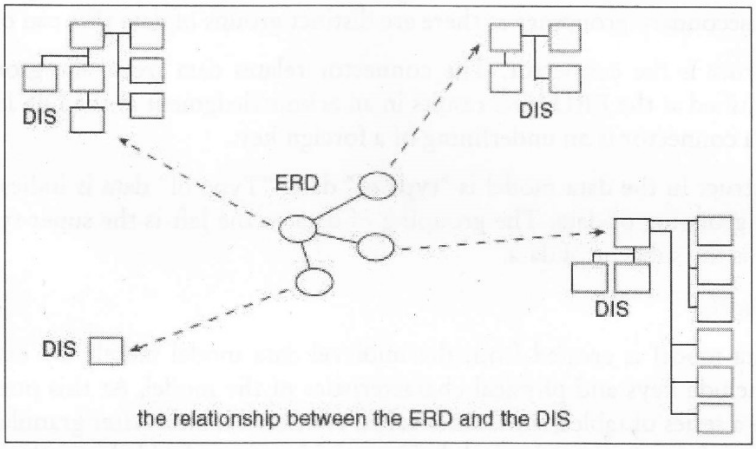


Figure 1.24: Each entity in the ERD is further defined by its own DIS

Interestingly, only very rarely are all of the midlevel models developed at once. The midlevel data model for one major subject area is expanded, then a portion of the model is fleshed out while other parts remain static, and so forth.

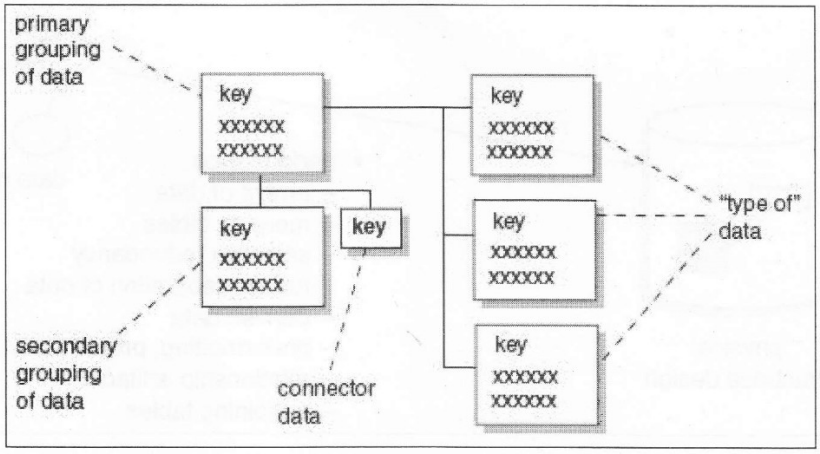


Figure 1.25: Four Constructs that make up the Midlevel Data Model

Shown in Figure 1.25, four basic constructs are found at the midlevel model:

1. A primary grouping of data
2. A secondary grouping of data
3. A connector, signifying the relationships of data between major subject areas
4. "Type of" data

The primary grouping exists once, and only once, for each major subject area. It holds attributes that exist only once for each major subject area. As with all groupings of data, the primary grouping contains attributes and keys for each major subject area.

The secondary grouping holds data attributes that can exist multiple times for each major subject area. This grouping is indicated by a line emanating downward from the primary grouping of data. There may be as many secondary groupings as there are distinct groups of data that can occur multiple times.

The third construct is the connector. The connector relates data from one grouping to another. A relationship identified at the ERD level results in an acknowledgment at the DIS level. The convention used to indicate a connector is an underlining of a foreign key.

The fourth construct in the data model is "type of" data. "Type of" data is indicated by a line leading to the right of a grouping of data. The grouping of data to the left is the super type. The grouping of data to the right is the subtype of data.

Physical Data Model

The physical data model is created from the midlevel data model merely by extending the midlevel data model to include keys and physical characteristics of the model. At this point, the physical data model looks like a series of tables, sometimes called relational tables. After granularity and partitioning are factored in, a variety of other physical design activities are embedded into the design, as outlined in Figure 1.26. At the heart of the physical design considerations is the usage of physical I/O (input/output). Physical I/O is the activity that brings data into the computer from storage or sends data to storage from the computer.

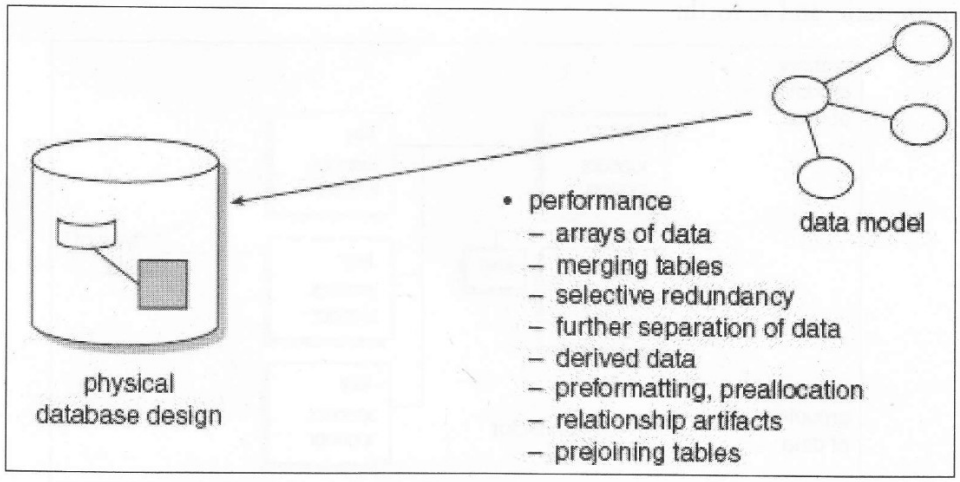


Figure 1.26: Getting Good Performance out of the Data Warehouse Environment

The job of the data warehouse designer is to organize data physically for the return of the maximum number of records from the execution of a physical I/O.

Example: Suppose a programmer must fetch five records. If those records are organized into different blocks of data on storage, then five I/Os will be required. But if the designer can anticipate that the records will be needed as a group and can physically juxtapose those records into the same block, then only one I/O will be required, thus making the program run much more efficiently.

1.4.3 Normalization/Denormalization

The output of the data model process is a series of tables, each of which contains keys and attributes. The normal output produces numerous tables, each with only a modicum of data. While there is nothing wrong—per se—with lots of little tables, there is a problem from a performance perspective. Consider the work the program has to do in order to interconnect the tables dynamically, as shown in Figure 1.27.

In Figure 1.27, a program goes into execution. First, one table is accessed, then another. To execute successfully, the program must jump around many tables. Each time the program jumps from one table to the next, I/O is consumed, in terms of both accessing the data and accessing the index to find the data. This cost a lot of redundant data.

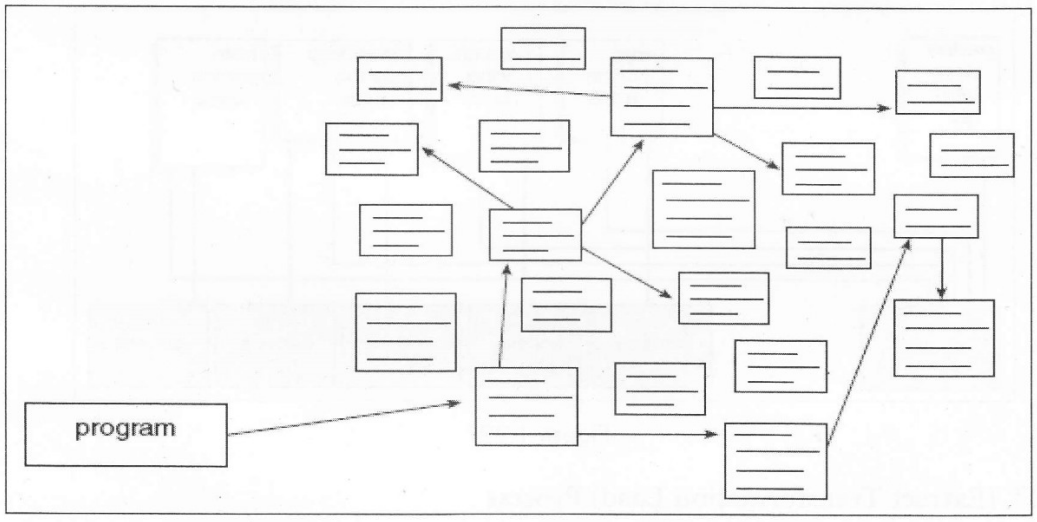


Figure 1.27

A more rational approach is to physically merge the tables so that minimal I/O is consumed, as seen in Figure 1.28. Now the same program operates as before, only it needs much less I/O to accomplish the same task.

Interestingly, in the data warehouse these circumstances occur regularly because of the time-based orientation of the data. Data warehouse data is always relevant to some moment in time, and units of time occur with great regularity.

In the data warehouse, creating an array by month, for example, is a very easy, natural thing to do. Another important design technique that is especially relevant to the data warehouse environment is the deliberate introduction of redundant data.

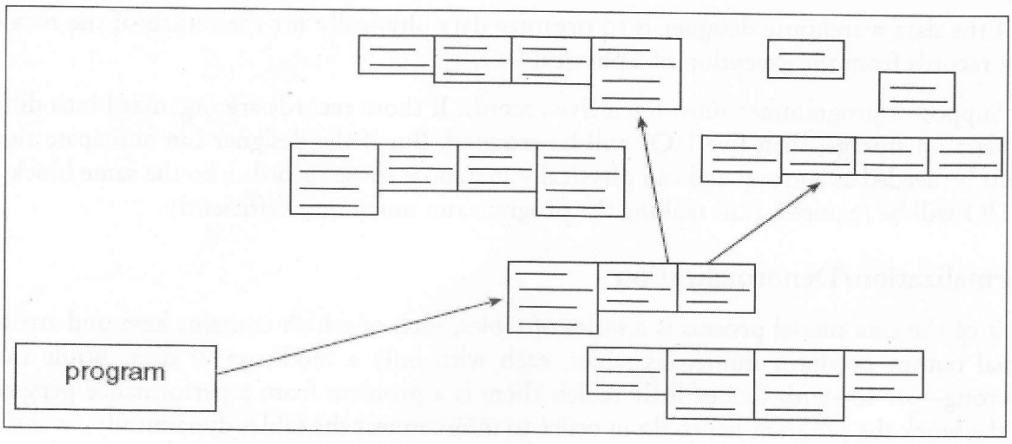


Figure 1.28

Figure 1.29 shows an example where the deliberate introduction of redundant data pays a big dividend. In the top of Figure 1.29, the field—description—is normalized and exists non redundantly. In doing so, all processes that must see the description of a part must access the base parts table. The access of the data is very expensive, although the update of the data is optimal.

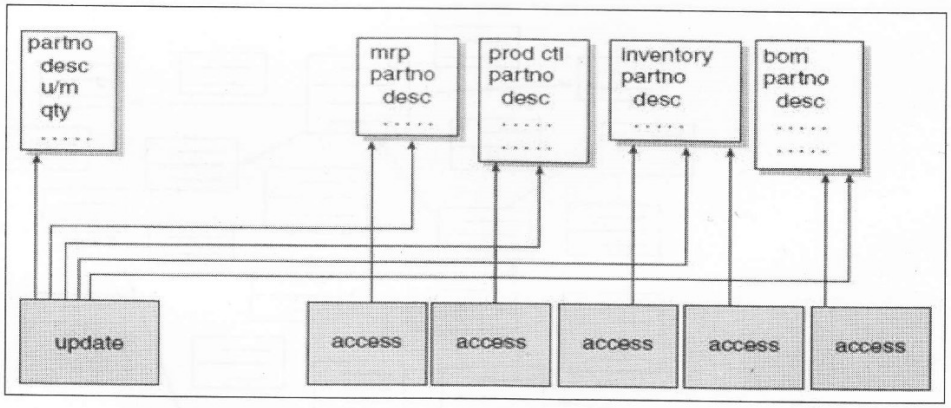


Figure 1.29

1.4.4 ETL (Extract Transformation Load) Process

System Process is basically an ETL Process.

In this section we will discussed about the four major system process of the data warehouse. They are extract (data from the operational systems and bring it to the data warehouse), transform (the data into internal format and structure of the data warehouse), cleanse (to make sure it is of sufficient quality to be used for decision making) and load (cleanse data is put into the data warehouse).

These four processes that is extraction, transformation, and loading is collectively known as **Data Staging**.

Extract

In this process data is extracted or taken from the operational system and this extracted data is handed over to data warehouse. User may find some of the data in the operational system or database, useful

for decision making. So, it is necessary to extract the relevant data from the operational database and bring it to the data warehouse. This process is done by various tools. One of them is Data Junction which is a commercial tool that helps in the extraction process.

The user of one of these tools typically has an easy-to-use windowed interface by which to specify the following:

- Which files and tables are to be accessed in the source database?
- Which fields are to be extracted from them? This is often done internally by SQL Select statement.
- What are those to be called in the resulting database?
- What is the target machine and database format of the output?
- On what schedule should the extraction process be repeated?

Loading

It often implies physical movement of the data from the computer(s) storing the source database(s) to that which will store the data warehouse database, assuming it is different. Loading takes place immediately after the extraction process. This process of loading is done through high-speed communication link which is one of the common channel for data movement. For example: Oracle Warehouse Builder is the API from Oracle, which provides the features to perform the ETL task on Oracle Data Warehouse.

Cleansing

After entering whole data in the data warehouse, it is very necessary to check the data for any errors. The developer cannot change the quality of the data but he is in position to make the data as error free as possible. If he finds that improvements can be done for the particular data, he will pass on the information to the higher authorities and they can look into the matter and improvements can be done in future.

This process of checking the data for errors is known as **Data Cleansing.**

Data Cleansing must deal with many types of possible errors. These include missing data and incorrect data at one source; inconsistent data and conflicting data when two or more source are involved.

Transform

Data that is stored in the operational database is based on some conditions according to its particular system. It is provided with a set of priorities, which keeps changing as per the requirements. Therefore, when this data is transferred to the data warehouse, it causes some inconsistency in that particular environment. So, the process that deals with this type of inconsistency (if any) is called Transformation.

One of the most common transformation issues is 'Attribute Naming Inconsistency'. It is used to make data consistent and make it common in the data warehouse by providing it a single name. So basically it prevents from any kind of confusion in the data warehouse. For example Customer Name may be CUST_NAME in one database and CNAME in the other. Thus one set of Data Names are picked and used consistently in the data warehouse. Once all the data elements are provided with their correct names, they must be converted to common formats. The conversion may consists of the following:

- Characters must be converted ASCII to EBCDIC or vise versa.
- Mixed Text may be converted to all uppercase for consistency.
- Numerical data must be converted in to a common format.
- Data Format has to be standardized.
- Measurement may have to convert. (Rs./ $)
- Coded data (Male/ Female, M/F) must be converted into a common format.

All these transformation activities are automated and many commercial products are available to perform the tasks. Another tool that is used for this process is **Data MAPPER**.

All these processes are discussed below in detail.

1.4.5 Purging Warehouse Data

Data does not just eternally pour into a data warehouse. It has its own life cycle within the warehouse as well. At some point in time, data is purged from the warehouse. The issue of purging data is one of the fundamental design issues that must not escape the data warehouse designer. In some senses, data is not purged from the warehouse at all. It is simply rolled up to higher levels of summary. There are several ways in which data is purged or the detail of data is transformed, including the following:

1. Data is added to a rolling summary file where detail is lost.
2. Data is transferred to a bulk storage medium from a high-performance medium such as DASD.
3. Data is actually purged from the system.
4. Data is transferred from one level of the architecture to another, such as from the operational level to the data warehouse level.

There are, then, a variety of ways in which data is purged or otherwise transformed inside the data warehouse environment. The life cycle of data — including its purge or final archival dissemination — should be an active part of the design process for the data warehouse.

1.4.6 Data Cleaning

Data cleaning can be applied to remove noise and correct inconsistencies in the data. It is a routine work to "clean" the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. Dirty data can cause confusion for the mining procedure. Although most mining routines have some procedures for dealing with incomplete or noisy data, they are not always robust. Instead, they may concentrate on avoiding overfitting the data to the function being modeled. Therefore, a useful preprocessing step is to run your data through some data cleaning routines. Some of the basic methods for data cleaning are as follows:

Data Cleaning for Missing Values

The following methods can be used to clean data for missing values in a particular attribute:

1. *Ignore the tuple:* This is usually done when the class label is missing (assuming the mining task involves classification or description). This method is not very effective, unless the tuple

contains several attributes with missing values. It is especially poor when the percentage of missing values per attribute varies considerably.

2. ***Fill in the missing value manually:*** In general, this approach is time-consuming and may not be feasible given a large data set with many missing values.
3. ***Use a global constant to fill in the missing value:*** Replace all missing attribute values by the same constant, such as a label like "Unknown". If missing values are replaced by, say, "Unknown", then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common — that of "Unknown". Hence, although this method is simple, it is not recommended.
4. ***Use the attribute mean to fill in the missing value:*** You can fill the missing values with the average value in that attribute.
5. ***Use the attribute mean for all samples belonging to the same class as the given tuple.***
6. ***Use the most probable value to fill in the missing value:*** This may be determined with inference-based tools using a Bayesian formalism or decision tree induction. For example, using the other customer attributes in your data set, you may construct a decision tree to predict the missing values for income.

Note: Methods 3 to 6 bias the data. The filled-in value may not be correct. Method 6, however, is a popular strategy. In comparison to the other methods, it uses the most information from the present data to predict missing values.

Noisy Data

Noise is a random error or variance in a measured variable.

Inconsistent Data

There may be inconsistencies in the data recorded for some transactions. Some data inconsistencies may be corrected manually using external references. For example, errors made at data entry may be corrected by performing a paper trace. This may be coupled with routines designed to help correct the inconsistent use of codes. Knowledge engineering tools may also be used to detect the violation of known data constraints. For example, known functional dependencies between attributes can be used to find values contradicting the functional constraints.

There may also be inconsistencies due to data integration, where a given attribute can have different names in different databases. Redundancies may also result.

1.4.7 Data Transformation

In data transformation, the data are transformed or consolidated into forms appropriate for mining. Data transformation can involve the following:

Normalization, where the attribute data is scaled so as to fall within a small specified range, such as -1.0 to 1.0, or 0 to 1.0.

Smoothing works to remove the noise from data. Such techniques include binning, clustering, and regression.

Aggregation, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for analysis of the data at multiple granularities.

Generalization of the data, where low level or 'primitive' (raw) data are replaced by higher level concepts through the use of concept hierarchies. For example, categorical attributes, like street, can be generalized to higher level concepts, like city or county. Similarly, values for numeric attributes, like age, may be mapped to higher level concepts, like young, middle-aged, and senior.

1.4.8 Query Management Process

For heterogeneous database integration, the traditional database implements query-driven approach, which requires complex information filtering and integration processes, and competes for resources with processing at local sources. It is inefficient and potentially expensive for frequent queries, especially for queries requiring aggregations.

In **query-driven approach**, data warehousing employs an update-driven approach in which information from multiple, heterogeneous sources is integrated in advance and stored in a warehouse for direct querying and analysis. In this approach, a data warehouse brings high performance to the integrated heterogeneous database system since data are copied, preprocessed, integrated, annotated, summarized, and restructured into one semantic data store. Furthermore, **query processing** in data warehouses does not interfere with the processing at local sources. Moreover, data warehouses can store and integrate historical information and support complex multidimensional queries. As a result, data warehousing has become very popular in industry.

1.4.9 Going from the Data Warehouse to the Operational Environment

The operational environment and the data warehouse environment are about as different as any two environments can be in terms of content, technology, usage, communities served, and a hundred other ways. The interface between the two environments is well documented. Data undergoes a fundamental transformation as it passes from the operational environment to the data warehouse environment. For a variety of reasons—the sequence in which business is conducted, the high performance needs of operational processing, the aging of data, the strong application orientation of operational processing, and so forth—the flow of data from the operational environment to the data warehouse environment is natural and normal. This normal flow of data is shown in Figure 1.30.

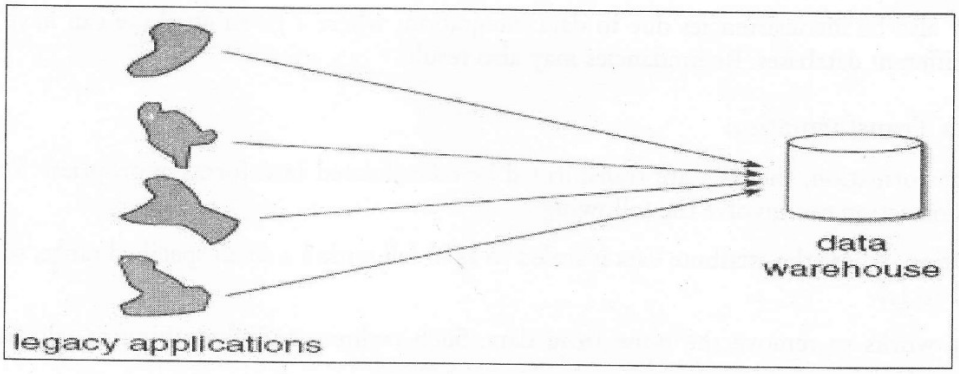


Figure 1.30

1.4.10 Incorrect Data in the Data Warehouse

The architect needs to know what to do about incorrect data in the data warehouse. The first assumption is that incorrect data arrives in the data warehouse on an exception basis. If data is being incorrectly entered in the data warehouse on a wholesale basis, then it is incumbent on the architect to find the offending ETL program and make adjustments. Occasionally, even with the best of ETL processing, a few pieces of incorrect data enter the data warehouse environment. How should the architect handle incorrect data in the data warehouse? There are at least three options. Each approach has its own strengths and weaknesses, and none are absolutely right or wrong. Instead, under some circumstances one choice is better than another.

Example: Suppose that on July 1 an entry for $5,000 is made into an operational system for account ABC. On July 2 a snapshot for $5,000 is created in the data warehouse for account ABC. Then on August 15 an error is discovered. Instead of an entry for $5,000, the entry should have been for $750. How can the data in the data warehouse be corrected?

Choice 1: Go back into the data warehouse for July 2 and find the offending entry. Then, using update capabilities, replace the value $5,000 with the value $750. This is a clean and neat solution when it works, but it introduces new issues:

The integrity of the data has been destroyed. Any report running between July 2 and Aug. 16 will not be able to be reconciled.

The update must be done in the data warehouse environment.

In many cases there is not a single entry that must be corrected, but many, many entries that must be corrected.

Choice 2: Enter offsetting entries. Two entries are made on August 16, one for $5,000 and another for _$750. This is the best reflection of the most up-to-date information in the data warehouse between July 2 and August 16. There are some drawbacks to this approach:

Many entries may have to be corrected, not just one. Making a simple adjustment may not be an easy thing to do at all.

Sometimes the formula for correction is so complex that making an adjustment cannot be done.

Choice 3: Reset the account to the proper value on August 16. An entry on August 16 reflects the balance of the account at that moment regardless of any past activity. An entry would be made for $750 on August 16. But this approach has its own drawbacks:

The ability to simply reset an account as of one moment in time requires application and procedural conventions.

Such a resetting of values does not accurately account for the error that has been made.

Choice 3 is what likely happens when you cannot balance your checking account at the end of the month. Instead of trying to find out what the bank has done, you simply take the bank's word for it and reset the account balance.

There are then at least three ways to handle incorrect data as it enters the data warehouse. Depending on the circumstances, one of the approaches will yield better results than another approach.

1.5 BUILDING THE WAREHOUSE ON MULTIPLE LEVELS

The simultaneous data warehouse development occurs when different development groups are building different levels of the data warehouse, as seen in Figure 1.31. This case is very different from the case of the distributed data warehouse development. In this case, Group A is building the high level of summarization of data, Group B is building the middle level of summarization, and Group C is building the current level of detail.

The scenario of multiple levels of data warehouse development is very common. Fortunately, it is the easiest scenario to manage, with the fewest risks.

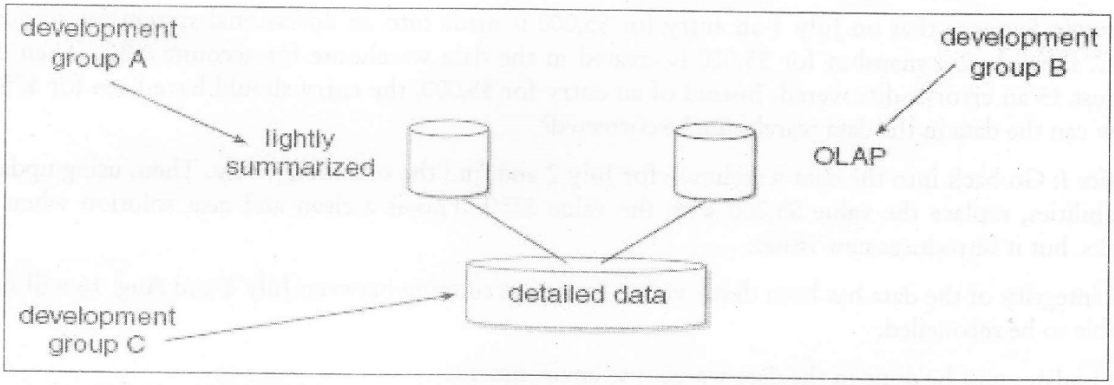


Figure 1.31

The primary concern of the data architect is to coordinate the efforts of the different development teams, both in terms of the specification of content and structure and in terms of the timing of development.

Example: If Group A is significantly ahead of Group B or C, there will be a problem: When Group A is ready to populate its databases at the summary level, there may be no detailed data to work with.

The data model for the data warehouse directly reflects the design and development effort by the group doing current-level detailed analysis and design. Of course, indirectly the data warehouse data model reflects the needs of all groups. But because other groups are summarizing from the data found at the current level of detail, they have their own interpretation of what is needed. In most cases, the groups working on the higher levels of summarization have their own data models that reflect their own specialized needs.

One of the issues of managing multiple groups building different levels of summarization is the technological platforms on which the data warehouse levels are built. Normally, different groups choose different technological platforms. In fact, for the different development groups to choose the same platform would be very unusual. There are several reasons for this, though the primary one is cost. The detailed level of data requires an industrial-strength platform because of the large **volume** of data that will have to be handled. The different levels of summarization will require much less data, especially at the higher levels of summarization. It is overkill (and expensive) to place the higher levels of summarized data on the same platform as the detailed data (although it can be done).

1.5.1 Multiple Groups Building the Current Level of Detail

An infrequent set of circumstances occurs when multiple development groups attempt to build the current level of detail in a data warehouse in a non distributed manner. Figure 1.32 illustrates this phenomenon.

As long as the groups that are developing the current level of detail are developing mutually exclusive sets of data, there is little difficulty. In this case, if the development groups are working from a common data model and the different groups' platforms are compatible, no problems should ensue. Unfortunately, mutually exclusive data sets are the exception rather than the rule. It is much more common for the multiple development groups to be designing and populating some or all of the same data.

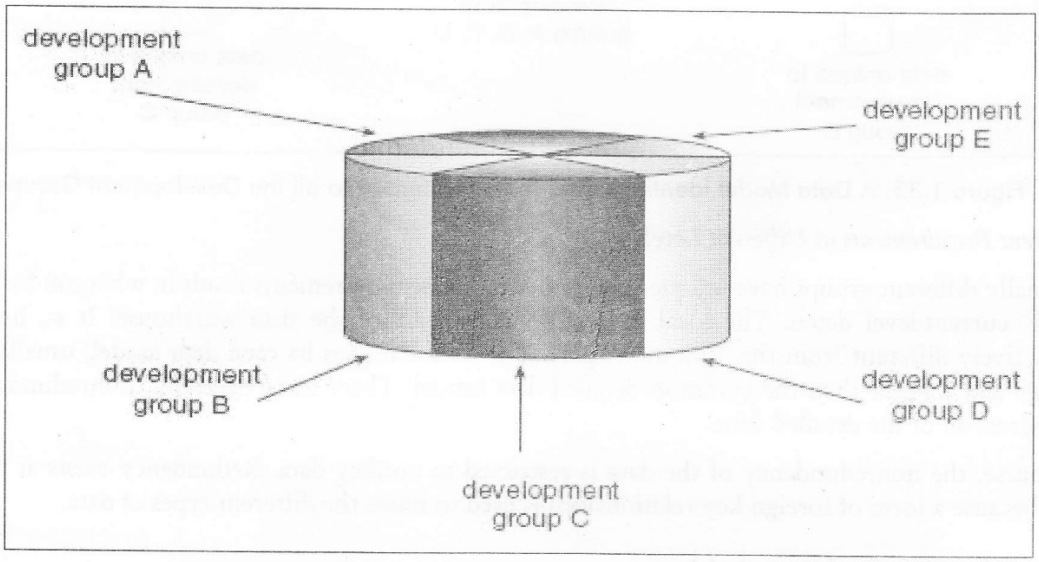


Figure 1.32

To ensure that no redundant data is developed, it is necessary to create a data model that reflects the common detailed data. Figure 1.33 shows that multiple development groups have combined their interests to create a common data model. In addition to the currently active development groups, other groups that will have future requirements but who are not currently in a development mode may also contribute their requirements. (Of course, if a group knows it will have future requirements but is unable to articulate them, then those requirements cannot be factored into the common detailed data model.) The common detailed data model reflects the collective need among the different groups for detailed data in the data warehouse.

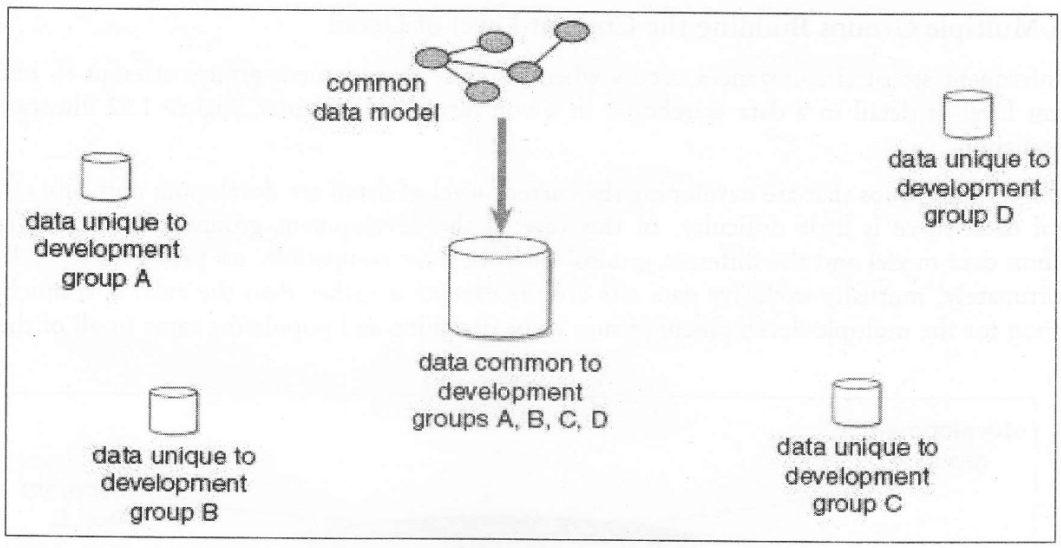


Figure 1.33: A Data Model identifies data that is Common to all the Development Groups

Different Requirements at Different Level

Normally different groups have unique requirements. These requirements result in what can be termed "local" current-level detail. The local data is certainly part of the data warehouse. It is, however, distinctively different from the "common" part. The local data has its own data model, usually much smaller and simpler than the common detailed data model. There is, of necessity, nonredundancy of data across all of the detailed data.

Of course, the nonredundancy of the data is restricted to nonkey data. Redundancy exists at the key level because a form of foreign key relationships is used to relate the different types of data.

1.5.2 Other Types of Detailed Data

Another strategy is to use different platforms for the different types of data found at the detailed level. Some of the local data is on one platform, the common data is on another platform, and other local data is on yet another. This option is certainly one that is valid, and it often satisfies the different political needs of the organization. With this option each group doing development can feel that it has some degree of control of at least its own peculiar needs. Unfortunately, this option has several major drawbacks. First, multiple technologies must be purchased and supported. Second, the end user needs to be trained in different technologies. And finally, the boundaries between the technologies may not be as easy to cross. Figure 1.34 illustrates this dilemma.

If there are to be multiple technologies supporting the different levels of detail in the data warehouse, it will be necessary to cross the boundaries between the technologies frequently. Software that is designed to access data across different technological platforms is available.

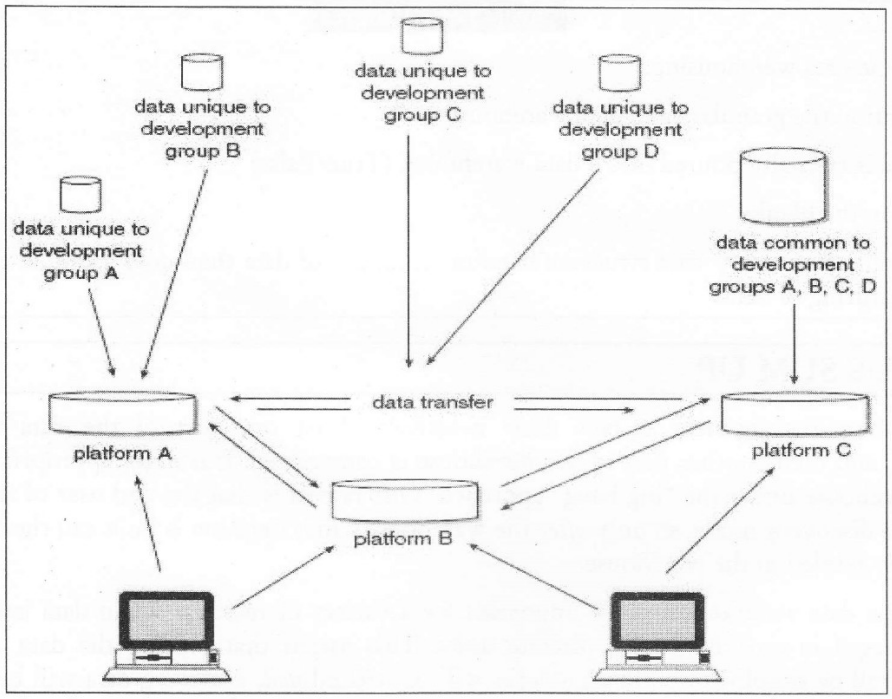


Figure 1.34: Some Problems with Interfacing Different Platforms

Detail Data

One other possibility worth mentioning is using multiple platforms for common detail of data. Figure 1.35 outlines this scenario.

While such a possibility is certainly an option, however, it is almost never a good choice. Managing common current detailed data is difficult enough. The volumes of data found at that level present their own unique problems for management. Adding the complication of having to cross multiple technological platforms merely makes life more difficult. Unless there are very unusual mitigating circumstances, this option is not recommended. The only advantage of multiple platforms for the management of common detail is that this option satisfies immediate political and organizational differences of opinion.

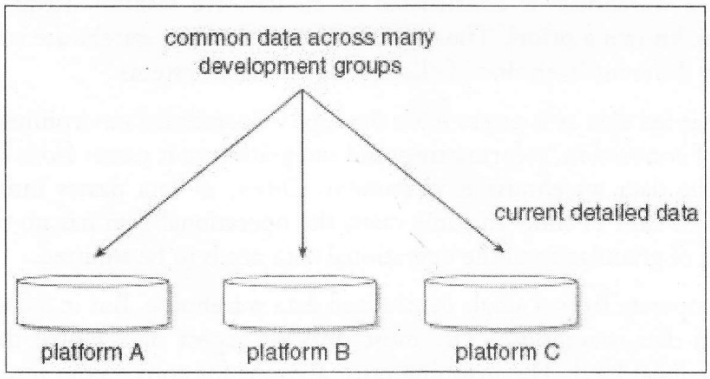


Figure 1.35: Common Detailed Data Across Multiple Platforms a Real Red Flag in all Cases

Check Your Progress

1. Define data warehousing.
2. Mention the granularity in data warehousing.
3. Data is eternally poured into a data warehouse. (True/False)
4. Fill in the blank:

   A rolling summary data structure handles ..... of data than does a simple cumulative structuring of data.

1.6 LET US SUM UP

Data warehouse development is best done iteratively. First one part of the data warehouse is constructed, and then another part of the warehouse is constructed. It is *never* appropriate to develop the data warehouse under the "big bang" approach. One reason is that the end user of the warehouse operates in a discovery mode, so only *after* the warehouse's first iteration is built can the developer tell what is really needed in the warehouse.

Partitioning a data warehouse is very important for a variety of reasons. When data is partitioned it can be managed in separate, small, discrete units. This means that loading the data into the data warehouse will be simplified, building indexes will be streamlined, archiving data will be easy, and so forth. There are at least two ways to partition data — at the DBMS/operating system level and at the application level. Each approach to partitioning has its own set of advantages and disadvantages.

Auditing can be done from a data warehouse, but auditing should not be done from a data warehouse. Instead, auditing is best done in the detailed operational transaction-oriented environment. When auditing is done in the data warehouse, data that would not otherwise be included is found there, the timing of the update into the data warehouse becomes an issue, and the level of granularity in the data warehouse is mandated by the need for auditing, which may not be the level of granularity needed for other processing. A normal part of the data warehouse life cycle is that of purging data. Often, developers neglect to include purging as a part of the specification of design. The result is a warehouse that grows eternally, which, of course, is impossibility.

The design of the data warehouse begins with the data model. The corporate data model is used for the design of the operational environment, and a variation of the corporate data model is used for the data warehouse. The data warehouse is constructed in an iterative fashion. Requirements for the data warehouse cannot be known a priori. The construction of the data warehouse is under a development life cycle completely different from that of classical operational systems.

The data warehouse is fed data as it passes from the legacy operational environment. Data goes through a complex process of conversion, reformatting, and integration as it passes from the legacy operational environment into the data warehouse environment. Often, as data passes into the data warehouse environment there is a shift of time. In some cases, the operational data has no time stamping, and in other cases, the level of granularity of the operational data needs to be adjusted.

Most environments operate from a single centralized data warehouse. But in some circumstances there can be a distributed data warehouse. The most difficult aspect of a global data warehouse is the mapping done at the local level. The mapping must account for conversion, integration, and different business practices. The mapping is done iteratively. In many cases, the global data warehouse will be

quite simple because only the corporate data that participates in business integration will be found in the global data warehouse.

The local data warehouses often are housed on different technologies. In addition, the global data warehouse may be on a different technology than any of the local data warehouses. The corporate data model acts as the glue that holds the different local data warehouses together, as far as their intersection at the global data warehouse is concerned. There may be local data warehouses that house data unique to and of interest to the local operating site.

The coordination and administration of the distributed data warehouse environment is much more complex than that of the single-site data warehouse.

1.7 KEYWORDS

Accuracy: A qualitative assessment of freedom from error or a quantitative measure of the magnitude of error, expressed as a function of relative error.

Data: A recording of facts, concepts, or instructions on a storage medium for communication, retrieval, and processing by automatic means and presentation as information that is understandable by human beings.

Data Structure: A logical relationship among data elements that is designed to support specific data manipulation functions (trees, lists, and tables).

Decision Support System (DSS): A system used to support managerial decisions. Usually DSS involves the analysis of many units of data in a heuristic fashion. As a rule, DSS processing does not involve the update of data.

Integrity: The property of a database that ensures that the data contained in the database is as accurate and consistent as possible.

Direct Access Storage Device (DASD): A data storage unit on which data can be accessed directly without having to progress through a serial file such as magnetic tapes file. A disk unit is a direct access storage device.

1.8 QUESTIONS FOR DISCUSSION

1. Explain the meaning of the term:
   (a) Granularity
   (b) Data warehouse
   (c) Structuring data
2. What do you mean by purging in data warehouse?
3. How can we define a structuring data in data warehouse?
4. What is the reporting and architectural environment?

Check Your Progress: Model Answers

1. A collection of integrated, subject-oriented databases designed to support the DSS function, where each unit of data is relevant to some moment in time. The data warehouse contains atomic data and lightly summarized data.
2. The level of detail contained in a unit of data. The more detail there is, the lower the level of granularity. The less detail there is, the higher the level of granularity.
3. False
4. Many fewer units

1.9 SUGGESTED READINGS

W. H. Inmon; Wiley, *Building the Data Warehouse* 2005.

Adelman, Sid *The Data Warehouse Database Explosion*. DMR (December 1996).

"Managing the Data Warehouse Environment." *Data Management Review* (February 1996). Defining who the data warehouse administrator is.

LESSON

2

PROCESS ARCHITECTURE

CONTENTS

2.0 AIMS AND OBJECTIVES

After studying this lesson, you will be able to:

- Discuss the models of data warehouse architecture
- Understand the components of process architecture.

2.1 INTRODUCTION

Data warehouse Architecture is divided into:

- System Process
- Process Architecture

System Process has been discussed in lesson 1. Process Architecture includes data warehouse models and its components which are discussed in this lesson.

2.2 DATA WAREHOUSE ARCHITECTURE

2.2.1 Process of Data Warehouse Design

A data warehouse can be built using three approaches:

- A top-down approach
- A bottom-up approach
- A combination of both approaches

The top-down approach starts with the overall design and planning. It is useful in cases where the technology is mature and well-known, and where the business problems that must be solved are clear and well-understood.

The bottom-up approach starts with experiments and prototypes. This is useful in the early stage of business modeling and technology development. It allows an organization to move forward at considerably less expense and to evaluate the benefits of the technology before making significant commitments.

In the combined approach, an organization can exploit the planned and strategic nature of the top-down approach while retaining the rapid implementation and opportunistic application of the bottom-up approach.

In general, the warehouse design process consists of the following steps:

1. Choose a *business process* to model, e.g., orders, invoices, shipments, inventory, account administration, sales, and the general ledger. If the business process is organizational and involves multiple, complex object collections, a data warehouse model should be followed. However, if the process is departmental and focuses on the analysis of one kind of business process, a data mart model should be chosen.
2. Choose the *grain* of the business process. The grain is the fundamental, atomic level of data to be represented in the fact table for this process, e.g., individual transactions, individual daily snapshots, etc.
3. Choose the *dimensions* that will apply to each fact table record. Typical dimensions are time, item, customer, supplier, warehouse, transaction type, and status.
4. Choose the *measures* that will populate each fact table record. Typical measures are numeric additive quantities like *dollars-sold* and *units-sold.*
5. Once a data warehouse is designed and constructed, the initial deployment of the warehouse includes initial installation, rollout planning, training and orientation. Platform upgrades and maintenance must also be considered. Data warehouse administration will include data refreshment, data source synchronization, planning for disaster recovery, managing access control and security, managing data growth, managing database performance, and data warehouse enhancement and extension.

2.2.2 Process Architecture

Process Architecture includes data warehouse models and its components which are discussed below.

Data Warehouses generally have a three-level (tier) architecture that includes:

- A bottom tier that consists of the Data Warehouse server, which is almost always a RDBMS. It may include several specialized data marts and a metadata repository.
- A middle tier that consists of an OLAP server for fast querying of the data warehouse. The OLAP server is typically implemented using either (1) a Relational OLAP (ROLAP) model, i.e., an extended relational DBMS that maps operations on multidimensional data to standard relational operations; or (2) a Multidimensional OLAP (MOLAP) model, i.e., a special purpose server that directly implements multidimensional data and operations.
- A top tier that includes front-end tools for displaying results provided by OLAP, as well as additional tools for data mining of the OLAP-generated data.

The overall DW architecture is shown in Figure 2.1.

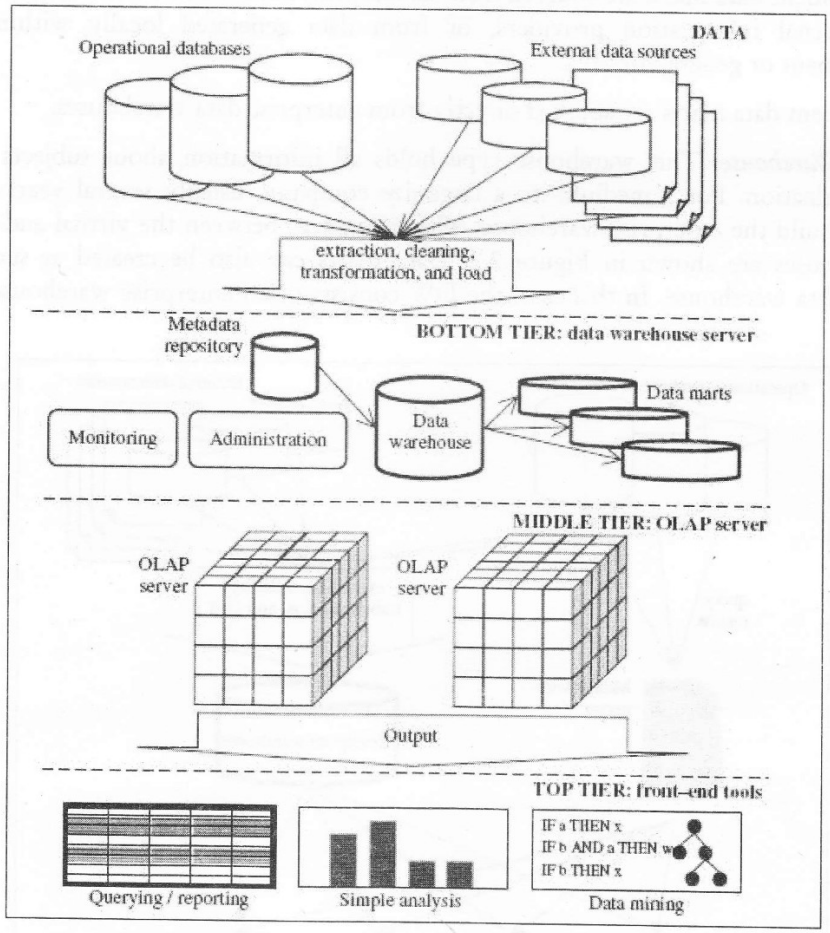


Figure 2.1: A three-tier Data Warehousing Architecture

2.2.3 Data Warehouse Models

From the architecture point of view, there are three data warehouse models: the virtual warehouse, the data *mart*, and the *enterprise warehouse.*

- ***Virtual Warehouse:*** A virtual warehouse is created based on a set of views defined for an operational RDBMS. This warehouse type is relatively easy to build but requires excess computational capacity of the underlying operational database system. The users directly access operational data via middleware tools. This architecture is feasible only if queries are posed infrequently, and usually is used as a temporary solution until a permanent data warehouse is developed.
- ***Data Mart:*** The data mart contains a subset of the organization-wide data that is of value to a small group of users, e.g., marketing or customer service. This is usually a precursor (and/or a successor) of the actual data warehouse, which differs with respect to the scope that is confined to a specific group of users.]

  Depending on the source of data, data marts can be categorized into the following two classes:

  - Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular department or geographic area.
  - Dependent data marts are sourced directly from enterprise data warehouses.
- ***Enterprise Warehouse:*** This warehouse type holds all information about subjects spanning the entire organization. For a medium- to a large-size company, usually several years are needed to design and build the enterprise warehouse. The differences between the virtual and the enterprise data warehouses are shown in Figure 2.2. Data marts can also be created as successors of an enterprise data warehouse. In this case, the DW consists of an enterprise warehouse and (several) data marts.

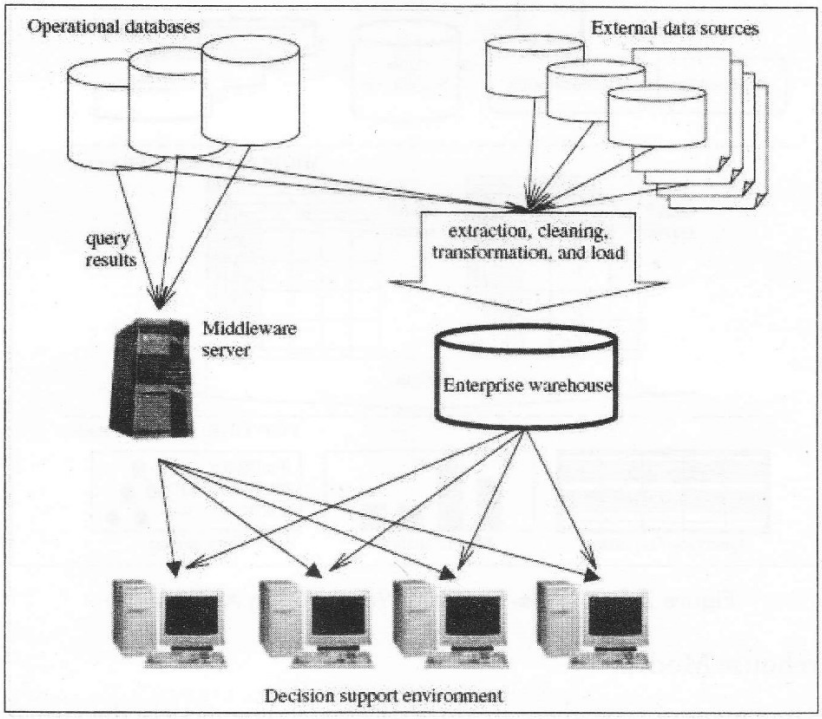


Figure 2.2: A Virtual data Warehouse and an Enterprise data Warehouse

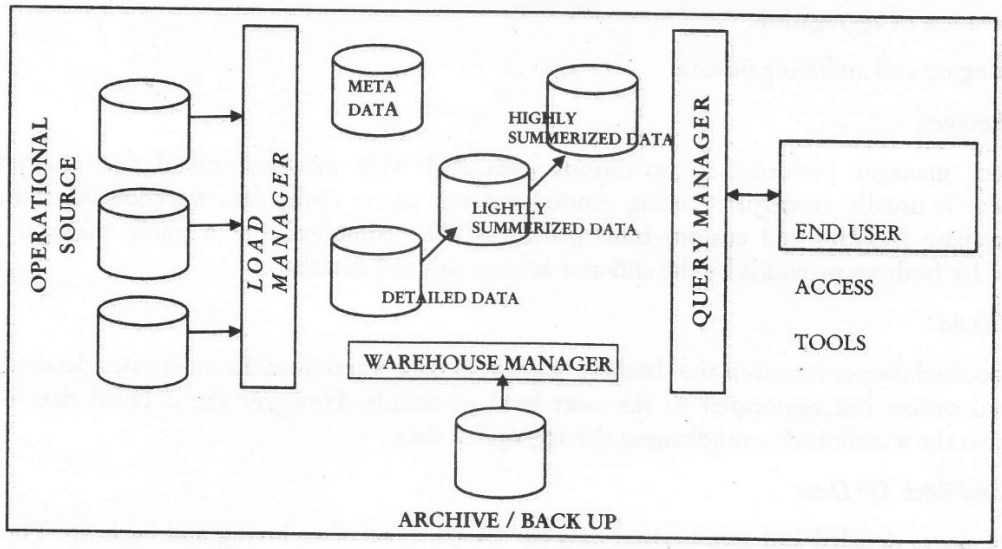
Note: The top-down development of an enterprise warehouse serves as a systematic solution and minimises integration problems. However, it is expensive, takes a long time to develop, and lacks flexibility due to the difficulty in achieving consistency and consensus for a common data model for the entire organisation.

The bottom-up approach to the design, development, and deployment of independent data marts provides flexibility, low cost, and rapid return.

2.2.4 Components of Data Warehouse Architecture

The data in a data warehouse comes from operational systems of the organization. These are referred to as **source systems**. The data that is extracted from source systems is stored in **data staging area**, where the data is cleaned, transformed, combined, to prepare the data. The data staging area does not provide any query or presentation services. As soon as a system provides query or presentation services, it is categorized as a ***presentation server***. The data is loaded on the presentation server and stored for direct querying by end users, report writers and other applications.

The data travels from source systems to presentation servers through the data staging area. The entire process is known as ETL (extract, transform, and load A typical architecture of a data warehouse is shown below:



Source: http://www.scribd.com/doc/36201881/Lecture-2-Data-Warehouse-Architecture

Figure 2.3: Data Warehouse Architecture

Each component and the tasks performed by them are explained below:

Operational Data

The data of the data warehouse comes from the operational databases which include:

- Data can also come from the relational DBMS like Oracle, Informix.
- Operational data also includes external data obtained from commercial databases and databases associated with supplier and customers.

Load Manager

While performing extraction and loading, some operations are to be done so as to make sure that these processes are executed in a correct manner. These operations are performed by Load Manager. The load manager performs all the operations associated with extraction and loading data into the data warehouse. These operations involve simple transformations of the data to prepare the data for entry into the warehouse. The size and complexity of this component will vary between data warehouses and may be constructed using a combination of vendor data loading tools and custom built programs.

Warehouse Manager

All the operations associated with the management of data in the warehouse are performed by Warehouse Manager. This operation is built using vendor data management tools and custom built programs. The operations performed by warehouse manager include:

- Analysis of data to ensure consistency.
- Transformation and merging the source data from temporary storage into data warehouse tables.
- Create indexes and views on the base table.
- Demoralization.
- Generation of aggregation.
- Backing up and archiving of data.

Query Manager

The query manager performs all operations associated with management of user queries. This component is usually constructed using vendor end-user access tools, data warehousing monitoring tools, database facilities and custom built programs. The complexity of a query manager can be examined by facilities provided by the end-user access tools and database.

Detailed Data

All the detailed data is stored in the database schema in data warehouse. In most cases detailed data is not stored online but aggregated to the next level of details. However the detailed data is added regularly to the warehouse to supplement the aggregated data.

Archive and Back Up Data

This area stores detailed and summarized data for the purpose of archiving and back up. The data is transferred to storage archives such as magnetic tapes or optical disks.

Lightly And Highly Summerized Data

All the predefined lightly and highly summarized data generated by the warehouse manager is stored in data warehouse. This area of the warehouse is transient as it will be subject to change on an ongoing basis in order to respond to the changing query profiles. The purpose of the summarized information is to speed up the query performance. The summarized data is updated continuously as new data is loaded into the warehouse.

Meta Data

The data warehouse also stores all the Meta data (data about data) definitions used by all processes in the warehouse. It is used for the following purposes.

- ***The extraction and loading process:*** Meta data is used to map data sources to a common view of information within the warehouse.
- ***The warehouse management process:*** Meta data is used to automate the production of summary tables.
- ***As part of query management process:*** Meta data is used to direct a query to the most appropriate data source.

The structure of Meta data will differ in each process, because the purpose is different. More about Meta data will be discussed in the later Lecture Notes.

End-User Access Tools

The users interact with the warehouse using end user access tools. The examples of some of the end user access tools can be:

- Reporting and Query Tools
- Application Development Tools
- Executive Information Systems Tools
- Online Analytical Processing Tools
- Data Mining Tools.

Check Your Progress

1. Fill in the blank:

   There are three data warehouse models: the *virtual warehouse*, the *data mart*, and ..... .
2. What is meta data?
3. What is Data Staging area?

2.3 LET US SUM UP

From the architecture point of view, there are three data warehouse models: the *virtual warehouse*, the *data mart*, and the *enterprise warehouse*.

2.4 KEYWORDS

Data Warehouse: A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process.

Virtual Warehouse: A virtual warehouse is created based on a set of views defined for an operational RDBMS.

Data Mart: The data mart contains a subset of the organisation-wide data that is of value to a small group of users, e.g., marketing or customer service. This is usually a precursor (and/or a successor) of the actual data warehouse, which differs with respect to the scope that is confined to a specific group of users.

Enterprise Warehouse: This warehouse type holds all information about subjects spanning the entire organisation. For a medium- to a large-size company, usually several years are needed to design and build the enterprise warehouse.

2.5 QUESTIONS FOR DISCUSSION

1. Draw the figure of 3-tier data warehouse.
2. Explain the difference between a data mart and a data warehouse.
3. Discuss the models of data warehouse architecture.
4. What are the different components of process architecture?

Check Your Progress: Model Answers

1. Enterprise warehouse
2. The warehouse management process - Meta data is used to automate the production of summary tables.
3. The data *extracted* from source systems is stored in a area called *data staging area*, where the data is cleaned, *transformed*, combined, deduplicated to prepare the data for us in the data warehouse.

2.6 SUGGESTED READINGS

Jiawei Han, Micheline Kamber, *Data Mining - Concepts and Techniques*, Morgan Kaufmann Publishers, First Edition, 2003.

Michael J. A. Berry, Gordon S Linoff, *Data Mining Techniques*, Wiley Publishing Inc, Second Edition, 2004.

Alex Berson, Stephen J. Smith, *Data warehousing, Data Mining & OLAP*, Tata McGraw Hill, Publications, 2004.

Sushmita Mitra, Tinku Acharya, *Data Mining - Multimedia, Soft Computing and Bioinformatics*, John Wiley & Sons, 2003.

Sam Anohory, Dennis Murray, *Data Warehousing in the Real World*, Addison Wesley, First Edition, 2000.

UNIT II

LESSON

3

MANAGING DATA WAREHOUSE, CAPACITY PLANNING AND DATA WAREHOUSE TUNING

CONTENTS

3.0 AIMS AND OBJECTIVES

After studying this lesson, you will be able to:

- Understand process of managing volume
- Know how to create profile records
- Discuss data warehouse tuning

3.1 INTRODUCTION

A data warehouse is a repository of an organization's electronically stored data. System managers includes process manager, load manager, warehouse manager, and query manager which have been discussed in this lesson.

The volume of data to be managed in the data warehouse is a significant issue. Creating profile records is an effective technique for managing the volume of data.

3.2 MANAGING DATA WAREHOUSE

Need to manage a Data Warehouse

A *data warehouse* is a repository of an organization's electronically stored data. Data warehouses are designed to facilitate reporting and analysis. It provides many advantages to business analysts as follows:

- A data warehouse may provide a *competitive advantage* by presenting relevant information from which to measure performance and make critical adjustments in order to help win over competitors.
- A data warehouse can enhance business *productivity* since it is able to quickly and efficiently gather information, which accurately describes the organization.
- A data warehouse facilitates *customer relationship marketing* since it provides a consistent view of customers and items across all lines of business, all departments, and all markets.
- A data warehouse may bring about *cost reduction* by tracking trends, patterns, and exceptions over long periods of time in a consistent and reliable manner.
- A data warehouse provides a common data model for all data of interest, regardless of the data's source. This makes it easier to report and analyze information than it would be if multiple data models from disparate sources were used to retrieve information such as sales invoices, order receipts, general ledger charges, etc.
- Because they are separate from operational systems, data warehouses provide retrieval of data without slowing down operational systems.

3.3 SYSTEM MANAGERS

Process Manager

Process manager is the person who is responsible for the whole flow of process in data warehouse.

Load Manager

While performing extraction and loading, some operations are to be done so as to make sure that these processes are executed in a correct manner. These operations are performed by Load Manager. The load manager performs all the operations associated with extraction and loading data into the data warehouse. These operations involve simple transformations of the data to prepare the data for entry into the warehouse. The size and complexity of this component will vary between data warehouses and may be constructed using a combination of vendor data loading tools and custom built programs.

Warehouse Manager

All the operations associated with the management of data in the warehouse are performed by Warehouse Manager. This operation is built using vendor data management tools and custom built programs. The operations performed by warehouse manager include:

- Analysis of data to ensure consistency
- Transformation and merging the source data from temporary storage into data warehouse tables
- Create indexes and views on the base table.

Query Manager

The query manager performs all operations associated with management of user queries. This component is usually constructed using vendor end-user access tools, data warehousing monitoring tools, database facilities and custom built programs. The complexity of a query manager can be examined by facilities provided by the end-user access tools and database.

3.4 CAPACITY PLANNING

In many cases, the volume of data to be managed in the data warehouse is a significant issue. Creating profile records is an effective technique for managing the volume of data. The reduction of the volume of data possible in moving detailed records in the operational environment into a profile record is remarkable. It is possible (indeed, normal) to achieve a 2-to-3 order-of-magnitude reduction of data by the creation of profile records in a data warehouse. Because of this benefit, the ability to create profile records is a powerful one that should be in the portfolio of every data architect approach (which can be used in conjunction with the iterative approach) to ensure that important detail is not permanently lost is to create an alternative level of historical detail along with the profile record, as shown in Figure 3.1 The alternative detail is not designed to be used frequently; it is stored on slow, inexpensive, sequential storage and is difficult to get to and awkward to work with. But the detail is there should it be needed. When management states that they must have a certain detail, however arcane, it can always be retrieved, albeit at a cost of time and money.

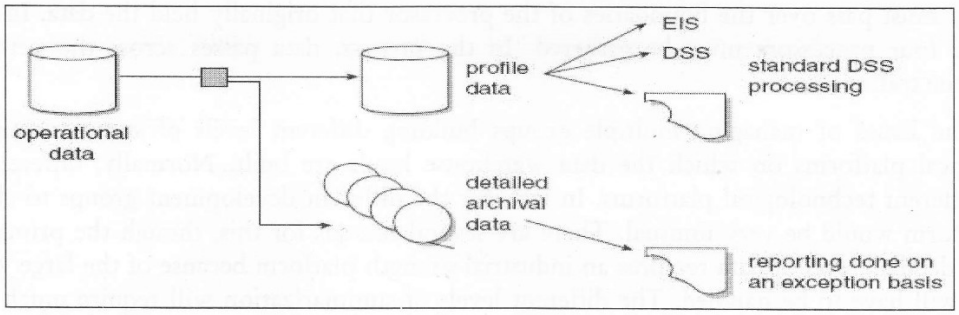


Figure 3.1

3.4.1 Create Multiple Profile Records

Multiple profile records can be created from the same detail. In the case of a phone company, individual call records can be used to create a customer profile record, a district traffic profile record, a line analysis profile record, and so forth.

The profile records can be used to go into the data warehouse or a data mart that is fed by the data warehouse. When the profile records go into a data warehouse, they are for general-purpose use. When the profile records go into the data mart, they are customized for the department thatuses the data mart.

The aggregation of the operational records into a profile record is almost always done on the operational server because this server is large enough to manage volumes of data and because that is where the data resides in any case. Usually creating the profile record involves sorting and merging data. Once the process of creating the snapshot becomes complicated and drawn out, whether the snapshot should be taken at all becomes questionable.

The meta data records written for profile records are very similar to the meta data records written for single activity snapshots with the exception that the process of aggregating the records becomes an important piece of meta data. (Technically speaking, the record of the process of aggregation is "meta process" information, not "meta data" information.)

The data warehouse environment will hold a lot of data, and the volume of data will be distributed over multiple processors. Logically there is a single data warehouse, but physically there are many data warehouses that are all tightly related but reside on separate processors. This configuration can be called the technologically distributed data warehouse.

Technologically Distributed Data Warehouse

The Data warehouse can be distributed technologically. Different type of a distributed warehouse is that of placing a data warehouse on the distributed technology of a vendor. Client/server technology fits this requirement nicely. The first advantage of a technologically distributed data warehouse is that the entry cost is cheap. The second advantage is that there is no theoretical limit to how much data can be placed in the data warehouse. If the volume of data inside the warehouse begins to exceed the limit of a single distributed processor, then another processor can be added to the network, and the progression of adding data continues in an unimpeded fashion. Whenever the data grows too large, another processor is added.

Example: Suppose one processor holds data for the year 1998, another processor for 1999, another for 2000, and yet another for 2001. When a request is made for data from 1998 to 2001, the result set for that query must pass over the boundaries of the processor that originally held the data. In this case, data from four processors must be gathered. In the process, data passes across the network and increases the traffic.

One of the issues of managing multiple groups building different levels of summarization is the technological platforms on which the data warehouse levels are built. Normally, different groups choose different technological platforms. In fact, for the different development groups to choose the same platform would be very unusual. There are several reasons for this, though the primary one is cost. The detailed level of data requires an industrial-strength platform because of the large volume of data that will have to be handled. The different levels of summarization will require much less data, especially at the higher levels of summarization. It is overkill (and expensive) to place the higher levels of summarized data on the same platform as the detailed data (although it can be done).

3.4.2 Estimating the Load

Estimating load basically involves the idea about your overall system-capacity requirements. This is done by establishing the highest number of pages and requests per second over a given time interval.

The minimum time period that you can measure is a 24-hour period, but Oracle suggests that you measure usage over a period of several weeks. If usage tends to spike during particular weeks of the year, try to obtain measurements from the busiest weeks. From this, you can better extract system-capacity requirements that will hold true even in the busiest period.

3.5 TUNING THE DATA WAREHOUSE

Following areas are mainly focused when tuning the data warehouse:

- Hardware considerations,
- Accessing performance,
- Data load tuning,
- Query tuning.

3.5.1 Hardware Considerations

Performance can be improved by having additional hardware, as it will increase the number of users and the size of the data warehouse. For example:

- It increases CPU bandwidth,
- It increases number of disks of I/O processing,
- And it add controllers if the controller bandwidth is exceeded.

It is necessary to understand that where a problem before assuming that any additional hardware will improve performance. For example, obtaining additional CPU bandwidth will not solve performance problems if the problems are caused by I/O bottlenecks. Here increasing the CPU bandwidth may worsen the bottleneck.

3.5.2 Accessing Performance

There are three major areas where a data warehousing system can use a little performance tuning:

- *ETL:* As we know that the data load is a very time-consuming process (and hence they are typically relegated to a nightly load job) and that data warehousing-related batch jobs are typically of lower priority, that means that the window for data loading is not very long. A data warehousing system that has its ETL process finishing right on-time is going to have a lot of problems simply because often the jobs do not get started on-time due to factors that is beyond the control of the data warehousing team. As a result, it is always an excellent idea for the data warehousing group to tune the ETL process as much as possible.
- *Query Processing:* Sometimes query performance can be an issue in a ROLAP environment or in a system where the reports are run directly against the relationship database. A study has shown that users generally lose their interest after 30 seconds of waiting for a report to return. ROLAP reports or reports that run directly against the RDBMS often exceed this time limit, and hence it is ideal for the data warehousing team to invest some time for query tuning, especially the most popularly ones. We discuss a number of query optimization ideas.

- *Report Delivery:* It is also possible that end users are experiencing significant delays in receiving their reports due to factors other than the query performance. For example, network traffic, server setup, etc. It is important for the data warehouse team to look into these areas for performance tuning.

Improving Performance in Data Warehouse

As data warehousing has become more and more important to businesses, increasing data warehouse performance has become vital. Many companies rely on numerous ways to improve data warehouse performance, including clearing obsolete data, increasing storage space and improving overall data warehouse architecture and design, to keep the data warehouse and the company functioning at their best.

Infrastructure is another important factor in data warehousing. Another approach that can help improve data warehouse performance is training. Data warehousing was designed to support decision making on a high level, but the overall importance of business intelligence has led to many other people using the data for a variety of purposes. In some cases, these employees have not received adequate training and do not know how to build efficient queries to get the information they need. For these employees, providing training on the use of the system and how to effectively query the data can lead to great improvement in data warehouse performance.

3.5.3 Load Tuning

Performance can be improved in the amount of time required to load data into the warehouse. Data loading is a critical process of overnight processing and serious complications can result if the data load is not completed in its particular window. Data Warehouse cannot be used until the data load is completed.

Maintaining Indexes

Maintenance of indexes is one of the costly areas of the data load, on data being loaded. If data is being loaded to tables that already contain data, it will prove more costly.

Indexes can be maintained side by side while data is being loaded. Also they can be dropped and then rebuilt once the data load is completed. Dropping indexes may be quicker, especially if the database management system allows indexes to be rebuilt in parallel.

The dropping method is the preferable one, even if it takes longer to load the data and then rebuild indexes.

There are two types of queries in the data warehousing environment. Fixed queries are clearly defined and well understood while ad hoc queries are unpredictable; in both frequency and quantity. Different techniques are applied to tune each type of query.

3.5.4 Query Tuning

It consists of fixed queries and *ad hoc* queries.

Fixed Queries

Fixed queries are the pre-defined queries and they include reports that are executed daily. These queries have predictable behavior and it generally follows traditional tuning that is done within a relational database environment.

Ad Hoc Queries

The number of users who perform ad hoc queries will have a significant effect on the performance of the data warehouse. The following information on users/user groups is considered.

- The number of different user groups that were previously identified when user access requirements were defined,
- Number of users in each group,
- Frequency and patterns of ad hoc querying,
- Peak usage days in weekly/monthly cycles,
- Query volume during peak times.

Other Considerations

Queries are most efficient when run against aggregations instead of directly against the fact data. Analyze ad hoc queries and create additional aggregations when appropriate.

Use query services to collect and analyze the following information on each query:

- Query structure,
- Query execution plan,
- Elapsed execution time of query,
- I/O, CPU, memory usage.

Check Your Progress

Fill in the blanks:

1. Creating .....is an effective technique for managing the volume of data.
2. The three major areas where a data warehousing system can use a little performance tuning are ETL, report delivery and..... .

3.6 LET US SUM UP

In capacity planning, the volume of data to be managed in the data warehouse is a significant issue. Creating profile records is an effective technique for managing the volume of data. The reduction of the volume of data possible in moving detailed records in the operational environment into a profile record is remarkable. Data loading is a critical process of overnight processing and serious complications can result if the data load is not completed in its particular window. Fixed queries are the pre-defined queries and they include reports that are executed daily. The number of users who perform ad hoc queries will have a significant effect on the performance of the data warehouse. As data warehousing has become more and more important to businesses, increasing data warehouse performance has become vital.

3.7 KEYWORDS

ETL: It is termed as extraction, transformation and load process.

Index: The portion of the storage structure maintained to provide efficient access to a record when its index key item is known.

3.8 QUESTIONS FOR DISCUSSION

1. Explain the meaning of the term – Profile Records.
2. How to manage data volume?
3. What are multiple profile records?
4. Discuss how the performance is accessed in data warehousing.

Check Your Progress: Model Answers

1. Profile records
2. Query processing

3.9 SUGGESTED READINGS

W. H. Inmon, *Building the Data Warehouse*; Wiley, 2005.

Adelman, Sid, *The Data Warehouse Database Explosion*, DMR (December 1996).

"Managing the Data Warehouse Environment." *Data Management Review* (February 1996). Defining who the data warehouse administrator is.

UNIT III

LESSON 4

DATA MINING AND KNOWLEDGE DISCOVERY

CONTENTS

4.0 AIMS AND OBJECTIVES

After studying this lesson, you will be able to:

- Discuss the concept of data mining
- Explain process of data mining
- Describe need and process of knowledge discovery
- Explain application of data mining and knowledge discovery

4.1 INTRODUCTION

In recent years data mining has attracted a great deal of attention in information industry due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge. The information and knowledge gained can be used for applications ranging from business management, production control, and market analysis, to engineering design and science exploration.

Data mining can be viewed as a result of the natural evolution of information technology. An evolutionary path has been witnessed in the database industry in the development of the following functionalities:

- Data collection and database creation,
- Data management (including data storage and retrieval, and database transaction processing), and
- Data analysis and understanding (involving data warehousing and data mining).

For instance, the early development of data collection and database creation mechanisms served as a prerequisite for later development of effective mechanisms for data storage and retrieval, and query and transaction processing. With numerous database systems offering query and transaction processing as common practice, data analysis and understanding has naturally become the next target.

By performing data mining, interesting knowledge, regularities, or high-level information can be extracted from databases and viewed or browsed from different angles. The discovered knowledge can be applied to decision-making, process control, information management, and query processing. Therefore, data mining is considered one of the most important frontiers in database and information systems and one of the most promising interdisciplinary developments in the information technology.

4.2 DATA MINING BASICS

Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

In simple words, data mining refers to extracting or "mining" knowledge from large amounts of data. Some other terms like knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging are also used for data mining. Many people treat data mining as a synonym for another popularly used term, Knowledge Discovery from Data, or KDD.

Some people view data mining as simply an essential step in the process of knowledge discovery. Knowledge discovery as a process and consists of an iterative sequence of the following steps:

1. ***Data deaning*** (to remove noise and inconsistent data)
2. ***Data integration*** (where multiple data sources may be combined)
3. ***Data selection*** (where data relevant to the analysis task are retrieved from the database)

4. *Data transformation* (where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations, for instance)
5. *Data mining* (an essential process where intelligent methods are applied in order to extract data patterns)
6. *Pattern evaluation* (to identify the truly interesting patterns representing knowledge based on some interestingness measures)
7. *Knowledge presentation* (where visualization and knowledge representation techniques are used to present the mined knowledge to the user).

The first four steps are different forms of data preprocessing, which are used for data preparation for mining. After this the data-mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user and may be stored as new knowledge in the knowledge base.

Although data mining is a relatively new term, the technology is not. Companies have used powerful computers to sift through volumes of supermarket scanner data and analyze market research reports for years. However, continuous innovations in computer processing power, disk storage, and statistical software are dramatically increasing the accuracy of analysis while driving down the cost.

Example: One Midwest grocery chain used the data mining capacity of *Oracle software* to analyze local buying patterns. They discovered that when men bought diapers on Thursdays and Saturdays, they also tended to buy beer. Further analysis showed that these shoppers typically did their weekly grocery shopping on Saturdays. On Thursdays, however, they only bought a few items. The retailer concluded that they purchased the beer to have it available for the upcoming weekend. The grocery chain could use this newly discovered information in various ways to increase revenue. For example, they could move the beer display closer to the diaper display. And, they could make sure beer and diapers were sold at full price on Thursdays.

Today, in industry, in media, and in the database research milieu, the term data mining is becoming more popular than the longer term of knowledge discovery from data. Therefore in a broader view of data mining functionality data mining can be defined as "the process of discovering interesting knowledge from large amounts of data stored in databases, data warehouses, or other information repositories."

4.2.1 Architecture of Data Mining

Based on the above definition, the architecture of a typical data mining system may have the following major components (Figure 4.1):

- *Information repository:* This is one or a set of databases, data warehouses, spreadsheets, or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.
- *Database or data warehouse server:* The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.
- *Knowledge base:* This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns.

- *Data mining engine:* This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterisation, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.
- *Pattern evaluation module:* This component typically employs interestingness measures and interacts with the data mining modules so as to *focus* the search toward interesting patterns.

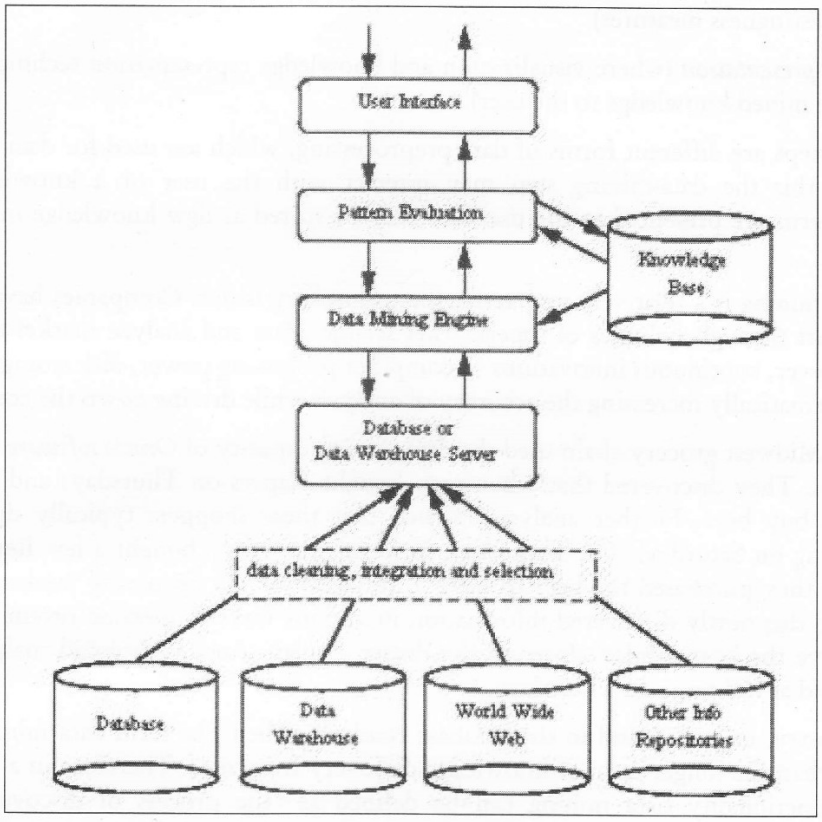


Figure 4.1: A Typical Architecture for Data Mining

User interface: This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse database and data warehouse schemas or data structures, evaluate mined patterns, and visualize the patterns in different forms.

Note that data mining involves an integration of techniques from multiple disciplines such as database and data warehouse technology, statistics, machine learning, high-performance computing, pattern recognition, neural networks, data visualization, information retrieval, image and signal processing, and spatial or temporal data analysis. In this book the emphasis is given on the database perspective that places on *efficient* and *scalable* data mining techniques.

For an algorithm to be scalable, its running time should grow approximately linearly in proportion to the size of the data, given the available system resources such as main memory and disk space.

4.2.2 What can Data Mining do?

Data mining is primarily used today by companies with a strong consumer focus - retail, financial, communication, and marketing organizations. It enables these companies to determine relationships among "internal" factors such as price, product positioning, or staff skills, and "external" factors such as economic indicators, competition, and customer demographics. And, it enables them to determine the impact on sales, customer satisfaction, and corporate profits. Finally, it enables them to "drill down" into summary information to view detail transactional data.

With data mining, a retailer could use point-of-sale records of customer purchases to send targeted promotions based on an individual's purchase history. By mining demographic data from comment or warranty cards, the retailer could develop products and promotions to appeal to specific customer segments.

Example: Blockbuster Entertainment mines its video rental history database to recommend rentals to individual customers.

- American Express can suggest products to its cardholders based on analysis of their monthly expenditures.
- WalMart is pioneering massive data mining to transform its supplier relationships.
- WalMart captures point-of-sale transactions from over 2,900 stores in 6 countries and continuously transmits this data to its massive 7.5 terabyte Teradata data warehouse.
- WalMart allows more than 3,500 suppliers, to access data on their products and perform data analyses.

These suppliers use this data to identify customer buying patterns at the store display level. They use this information to manage local store inventory and identify new merchandising opportunities.

4.2.3 How does Data Mining Work?

While large-scale information technology has been evolving separate transaction and analytical systems, data mining provides the link between the two. Data mining software analyzes relationships and patterns in stored transaction data based on open-ended user queries. Several types of analytical software are available: statistical, machine learning, and neural networks. Generally, any of four types of relationships are sought:

1. *Classes:* Stored data is used to locate data in predetermined groups. For example, a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order. This information could be used to increase traffic by having daily specials.
2. *Clusters:* Data items are grouped according to logical relationships or consumer preferences. For example, data can be mined to identify market segments or consumer affinities.
3. *Associations:* Data can be mined to identify associations. The beer-diaper example is an example of associative mining.
4. *Sequential patterns:* Data is mined to anticipate behavior patterns and trends. For example, an outdoor equipment retailer could predict the likelihood of a backpack being purchased based on a consumer's purchase of sleeping bags and hiking shoes.

Data mining consists of five major elements:

1. Extract, transform, and load transaction data onto the data warehouse system.
2. Store and manage the data in a multidimensional database system.
3. Provide data access to business analysts and information technology professionals.
4. Analyze the data by application software.
5. Present the data in a useful format, such as a graph or table.

Different levels of analysis are available:

1. *Artificial neural networks:* Non-linear predictive models that learn through training and resemble biological neural networks in structure.
2. *Genetic algorithms:* Optimization techniques that use processes such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.
3. *Decision trees:* Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific decision tree methods include Classification and Regression Trees (CART) and Chi Square Automatic Interaction Detection (CHAID). CART and CHAID are decision tree techniques used for classification of a dataset. They provide a set of rules that you can apply to a new (unclassified) dataset to predict which records will have a given outcome. CART segments a dataset by creating 2-way splits while CHAID segments using chi square tests to create multi-way splits. CART typically requires less data preparation than CHAID.
4. *Nearest neighbor method:* A technique that classifies each record in a dataset based on a combination of the classes of the k record(s) most similar to it in a historical dataset (where k 1). Sometimes called the k-nearest neighbor technique.
5. *Rule induction:* The extraction of useful if-then rules from data based on statistical significance.
6. *Data visualization:* The visual interpretation of complex relationships in multidimensional data. Graphics tools are used to illustrate data relationships.

4.2.4 Data Mining Process

The current process model for data mining provides an overview of the life cycle of a data mining project. It contains the corresponding phases of a project, their respective tasks, and relationships between these tasks. At this description level, it is not possible to identify all relationships. There possibly exist relationships between all data mining tasks depending on goals, background and interest of the user, and most importantly depending on the data.

The life cycle of a data mining project consists of six phases. The sequence of the phases is not strict. Moving back and forth between different phases is always required. It depends on the outcome of each phase which phase, or which particular task of a phase, that has to be performed next. The arrows indicate the most important and frequent dependencies between phases.

The outer circle in the figure symbolizes the cyclic nature of data mining itself. A data mining process continues after a solution has been deployed. The lessons learned during the process can trigger new, often more focused business questions. Subsequent data mining processes will benefit from the experiences of previous ones.

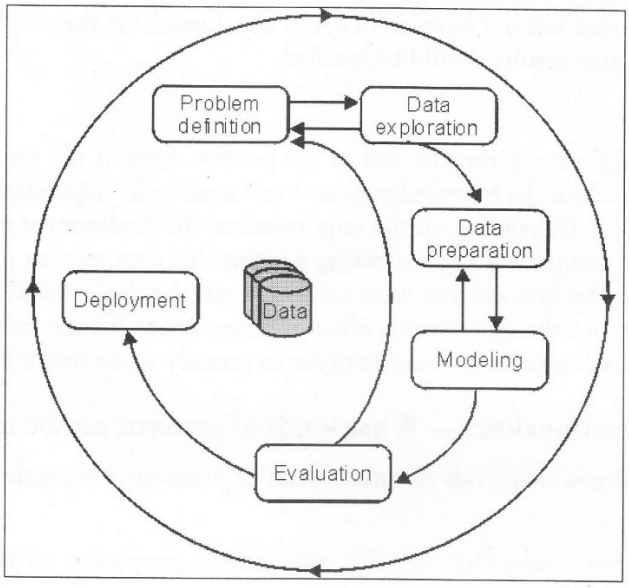


Figure 4.2

Business Understanding/Problem Definition

This initial phase focuses on understanding the project objectives and requirements from a business perspective, and then converting this knowledge into a data mining problem definition, and a preliminary plan designed to achieve the objectives.

Data Understanding/Exploration

The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data, or to detect interesting subsets to form hypotheses for hidden information.

Data Preparation

The data preparation phase covers all activities to construct the final dataset (data that will be fed into the modeling tool(s)) from the initial raw data. Data preparation tasks are likely to be performed multiple times, and not in any prescribed order. Tasks include table, record, and attribute selection as well as transformation and cleaning of data for modeling tools.

Modeling

In this phase, various modeling techniques are selected and applied, and their parameters are calibrated to optimal values. Typically, there are several techniques for the same data mining problem type. Some techniques have specific requirements on the form of data. Therefore, stepping back to the data preparation phase is often needed.

Evaluation

At this stage in the project you have built a model (or models) that appears to have high quality, from a data analysis perspective. Before proceeding to final deployment of the model, it is important to more thoroughly evaluate the model, and review the steps executed to construct the model, to be certain it properly achieves the business objectives. A key objective is to determine if there is some

important business issue that has not been sufficiently considered. At the end of this phase, a decision on the use of the data mining results should be reached.

Deployment

Creation of the model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process. In many cases it will be the customer, not the data analyst, who will carry out the deployment steps. However, even if the analyst will not carry out the deployment effort it is important for the customer to understand up front what actions will need to be carried out in order to actually make use of the created models.

4.2.5 Data Mining Functionalities — What kinds of patterns can be mined?

On mining over the databases two kinds of patterns can be discovered depending upon the data mining tasks employed:

- Descriptive data mining tasks that describe the general properties of the existing data. These include data characterization and discrimination.
- Predictive data mining tasks that attempt to do predictions based on inference on available data.

The data mining functionalities and the variety of knowledge they discover are briefly presented in the following list:

Characterization

Data characterization is a summarization of general features of objects in a target class, and produces what is called *characteristic rules*. The data relevant to a user-specified class are normally retrieved by a database query and run through a summarization module to extract the essence of the data at different levels of abstractions. For example, one may want to characterize the Our Video Store customers who regularly rent more than 30 movies a year. With concept hierarchies on the attributes describing the target class, the *attribute oriented induction* method can be used, for example, to carry out data summarization. Note that with a data cube containing summarization of data, simple OLAP operations fit the purpose of data characterization.

Discrimination

Data discrimination produces what are called *discriminant rules* and is basically the comparison of the general features of objects between two classes referred to as the *target class* and the *contrasting class*. For example, one may want to compare the general characteristics of the customers who rented more than 30 movies in the last year with those whose rental account is lower than 5. The techniques used for data discrimination are very similar to the techniques used for data characterization with the exception that data discrimination results include comparative measures.

Association Analysis

Association analysis is based on the *association rules*. It studies the frequency of items occurring together in transactional databases, and based on a threshold called *support*, identifies the frequent item sets. Another threshold, *confidence*, which is the conditional probability than an item appears in a transaction when another item appears, is used to pinpoint association rules. Association analysis is

commonly used for market basket analysis. For example, it could be useful for the OurVideoStore manager to know what movies are often rented together or if there is a relationship between renting a certain type of movies and buying popcorn or pop. The discovered association rules are of the form: P→Q [s, c], where P and Q are conjunctions of attribute value-pairs, and s (for support) is the probability that P and Q appear together in a transaction and c (for confidence) is the conditional probability that Q appears in a transaction when P is present. *For example, the hypothetic association rule*

Rent Type (X, "game") ∧ Age(X,"13-19") → Buys(X, "pop") [s=2% ,c=55%]

would indicate that 2% of the transactions considered are of customers aged between 13 and 19 who are renting a game and buying a pop, and that there is a certainty of 55% that teenage customers, who rent a game, also buy pop.

Classification

Classification is the processing of finding a set of models (or functions) that describe and distinguish data classes or concepts, for the purposes of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known). The derived model may be represented in various forms, such as *classification (IF-THEN) rules, decision trees, mathematical formulae,* or *neural networks.* For example, after starting a credit policy, the Our Video Store managers could analyze the customers' behaviors vis-à-vis their credit, and label accordingly the customers who received credits with three possible labels "safe", "risky" and "very risky". The classification analysis would generate a model that could be used to either accept or reject credit requests in the future.

Prediction

Classification can be used for predicting the class label of data objects. There are two major types of predictions: one can either try to predict (1) some unavailable data values or pending trends and (2) a class label for some data. The latter is tied to classification. Once a classification model is built based on a training set, the class label of an object can be foreseen based on the attribute values of the object and the attribute values of the classes. Prediction is however more often referred to the forecast of missing numerical values, or increase/ decrease trends in time related data. The major idea is to use a large number of past values to consider probable future values.

Note that Classification and prediction may need to be preceded by relevance analysis, which attempts to identify attributes that do not contribute to the classification or prediction process. These attributes can then be excluded.

Clustering

Similar to classification, clustering is the organization of data in classes. However, unlike classification, it is used to place data elements into related groups without advance knowledge of the group definitions i.e. class labels are unknown and it is up to the clustering algorithm to discover acceptable classes. Clustering is also called *unsupervised classification*, because the classification is not dictated by given class labels. There are many clustering approaches all based on the principle of maximizing the similarity between objects in a same class (*intra-class similarity*) and minimizing the similarity between objects of different classes (*inter-class similarity*). Clustering can also facilitate taxonomy formation, that is, the organization of observations into a hierarchy of classes that group similar events together. For example, for a data set with two attributes: AGE and HEIGHT, the following rule represents most of the data assigned to cluster 10:

If AGE >= 25 and AGE <= 40 and HEIGHT >= 5.0ft and HEIGHT <= 5.5ft then CLUSTER = 10

Evolution and Deviation Analysis

Evolution and deviation analysis pertain to the study of time related data that changes in time.

Evolution analysis models evolutionary trends in data, which consent to characterizing, comparing, classifying or clustering of time related data. For example, suppose that you have the major stock market (time-series) data of the last several years available from the New York Stock Exchange and you would like to invest in shares of high-tech industrial companies. A data mining study of stock exchange data may identify stock evolution regularities for overall stocks and for the stocks of particular companies. Such regularities may help predict future trends in stock market prices, contributing to your decision-making regarding stock investment.

Deviation analysis, on the other hand, considers differences between measured values and expected values, and attempts to find the cause of the deviations from the anticipated values. For example, a decrease in total *demand of CDs for rent at Video library* for the last month, in comparison to that of the same month of the last year, is a deviation pattern. Having detected a significant deviation, a data mining system may go further and attempt to explain the detected pattern (e.g., did the new comedy movies were released last year in comparison to the same period this year?).

4.2.6 Classification of Data Mining Systems

There are many data mining systems available or being developed. Some are specialized systems dedicated to a given data source or are confined to limited data mining functionalities, other are more versatile and comprehensive. Data mining systems can be categorized according to various criteria among other classification are the following:

- *Classification according to the kinds of data source mined:* this classification categorizes data mining systems according to the type of data handled such as spatial data, multimedia data, time-series data, text data, Worldwide Web, etc.
- *Classification according to the data model drawn on:* this classification categorizes data mining systems based on the data model involved such as relational database, object-oriented database, data warehouse, transactional, etc.
- *Classification according to the kind of knowledge discovered:* this classification categorizes data mining systems based on the kind of knowledge discovered or data mining functionalities, such as characterization, discrimination, association, classification, clustering, etc. Some systems tend to be comprehensive systems offering several data mining functionalities together.
- *Classification according to mining techniques used:* Data mining systems employ and provide different techniques. This classification categorizes data mining systems according to the data analysis approach used such as machine learning, neural networks, genetic algorithms, statistics, visualization, database-oriented or data warehouse-oriented, etc. The classification can also take into account the degree of user interaction involved in the data mining process such as query-driven systems, interactive exploratory systems, or autonomous systems. A comprehensive system would provide a wide variety of data mining techniques to fit different situations and options, and offer different degrees of user interaction.

4.3 NEED OF KNOWLEDGE DISCOVERY

The traditional method of turning data into knowledge relies on manual analysis and interpretation. For example, in the health-care industry, it is common for specialists to periodically analyze current trends and changes in health-care data, say, on a quarterly basis. The specialists then provide a report detailing the analysis to the sponsoring health-care organization; this report becomes the basis for future decision making and planning for health-care management. In a totally different type of application, planetary geologists sift through remotely sensed images of planets and asteroids, carefully locating and cataloging such geologic objects of interest as impact craters. Be it science, marketing, finance, health care, retail, or any other field, the classical approach to data analysis relies fundamentally on one or more analysts becoming innovaintimately familiar with the data and serving as an interface between the data and the users and products.

For these (and many other) applications, this form of manual probing of a data set is slow, expensive, and highly subjective. In fact, as data volumes grow dramatically, this type of manual data analysis is becoming completely impractical in many domains. Databases are increasing in size in two ways:

1. The number N of records or objects in the database and
2. The number d of fields or attributes to an object.

Hence, Knowledge Discover of Data is an attempt to address a problem that the digital information era made a fact of life for all of us data overload.

4.4 DATA MINING AND KNOWLEDGE DISCOVERY

The term *data mining* has mostly been used by statisticians, data analysts, and the Management Information Systems (MIS) communities. It has also gained popularity in the database field. The phrase *knowledge discovery in databases* was coined at the first KDD workshop in 1989 (Piatetsky-Shapiro 1991) to emphasize that knowledge is the end product of a data-driven discovery. It has been popularized in the AI and machine-learning fields.

KDD refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process. *Data mining* is the application of specific algorithms for extracting patterns from data. The distinction between the KDD process and the data-mining step (within the process) is a central point of this article. The additional steps in the KDD process, such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining, are essential to ensure that useful knowledge is derived from the data.

4.4.1 The Interdisciplinary Nature of Knowledge Discovery

KDD has evolved, and continues to evolve, from the intersection of research fields such as machine learning, pattern recognition, databases, statistics, AI, knowledge acquisition for expert systems, data visualization, and high-performance computing. The unifying goal is extracting high-level knowledge from low-level data in the context of large data sets.

The data-mining component of KDD currently relies heavily on known techniques from machine learning, pattern recognition, and statistics to find patterns from data in the data-mining step of the KDD process. A natural question is, How is KDD different from pattern recognition or machine

learning (and related fields)? The answer is that these fields provide some of the data-mining methods that are used in the data-mining step of the KDD process. KDD focuses on the overall process of knowledge discovery from data, including how the data are stored and accessed, how algorithms can be scaled to massive data sets.

4.4.2 Basic Definitions

KDD is the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.

Here, *data* are a set of facts (for example, cases in a database), and *pattern* is an expression in some language describing a subset of the data or a model applicable to the subset. Hence, in our usage here, extracting a pattern also designates fitting a model to data; finding structure from data; or, in general, making any high-level description of a set of data. The term *process* implies that KDD comprises many steps, which involve data preparation, search for patterns, knowledge evaluation, and refinement, all repeated in multiple iterations. By *nontrivial*, we mean that some search or inference is involved; that is, it is not a straightforward computation of predefined quantities like computing the average value of a set of numbers.

Data mining is a step in the KDD process that consists of applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns (or models) over the data. Note that the space of patterns is often infinite, and the enumeration of patterns involves some form of search in this space. Practical computational constraints place severe limits on the subspace that can be explored by a data-mining algorithm.

The KDD process involves using the database along with any required selection, preprocessing, sub sampling, and transformations of it; applying data-mining methods (algorithms) to enumerate patterns from it; and evaluating the products of data mining to identify the subset of the enumerated patterns deemed knowledge. The data-mining component of the KDD process is concerned with the algorithmic means by which patterns are extracted and enumerated from data.

4.5 KNOWLEDGE DISCOVERY PROCESS

The KDD process is interactive and iterative, involving numerous steps with many decisions made by the user. The knowledge discovery process is shown in Figure 4.3.

1. *First* is developing an understanding of the application domain and the relevant prior knowledge and identifying the goal of the KDD process from the customer's viewpoint.
2. *Second* is creating a target data set: selecting a data set, or focusing on a subset of variables or data samples, on which discovery is to be performed.
3. *Third* is data cleaning and preprocessing. Basic operations include removing noise if appropriate, collecting the necessary information to model or account for noise, deciding on strategies for handling missing data fields, and accounting for time-sequence information and known changes.
4. *Fourth* is data reduction and projection: finding useful features to represent the data depending on the goal of the task. With dimensionality reduction or transformation methods, the effective number of variables under consideration can be reduced, or invariant representations for the data can be found.

5. *Fifth* is matching the goals of the KDD process (step 1) to a particular data-mining method. For example, summarization, classification regression, clustering, and so on, is described later as well as in Fayyad, Piatetsky Shapiro, and Smyth (1996).
6. *Sixth* is exploratory analysis and model and hypothesis selection: choosing the data mining algorithm(s) and selecting method(s) to be used for searching for data patterns. This process includes deciding which models and parameters might be appropriate (for example, models of categorical data are different than models of vectors over the reals) and matching a particular data-mining method with the overall criteria of the KDD process (for example, the end user might be more interested in understanding the model than its predictive capabilities).
7. *Seventh* is data mining: searching for patterns of interest in a particular representational form or a set of such representations, including classification rules or trees, regression, and clustering. The user can significantly aid the data-mining method by correctly performing the preceding steps.
8. *Eighth* is interpreting mined patterns, possibly returning to any of steps 1 through 7 for further iteration. This step can also involve visualization of the extracted patterns and models or visualization of the data given the extracted models.
9. *Ninth* is acting on the discovered knowledge: using the knowledge directly, incorporating the knowledge into other system for further action, or simply documenting it and reporting it to interested parties. This process also includes checking for and resolving potential conflicts with previously believed (or extracted) knowledge.

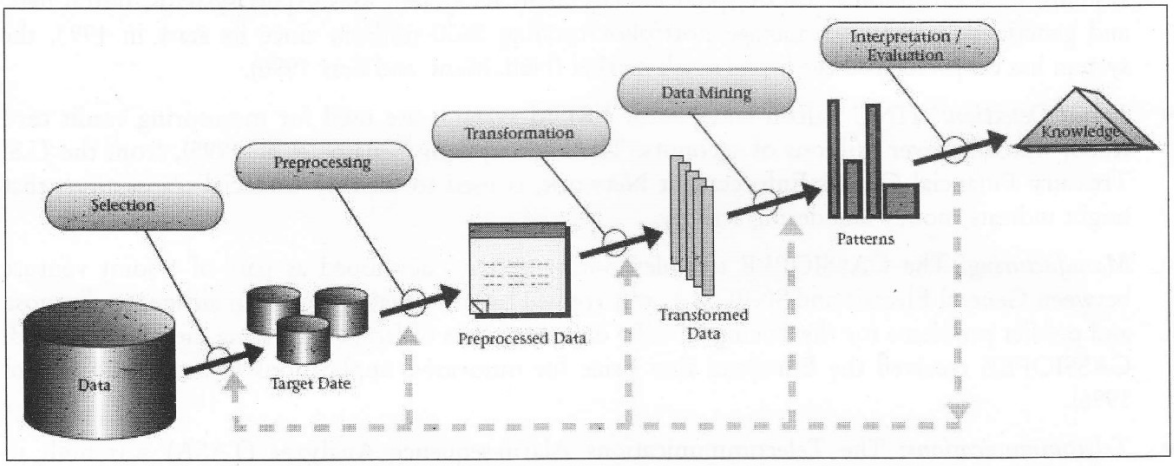


Figure 4.3

The KDD process can involve significant iteration and can contain loops between any two steps. The basic flow of steps (although not the potential multitude of iterations and loops) is illustrated in Figure 4.3. Most previous work on KDD has focused on step 7, the data mining. However, the other steps are as important (and probably more so) for the successful application of KDD in practice. Having defined the basic notions and introduced the KDD process, we now focus on the data-mining component, which has, by far, received the most attention in the literature.

4.6 DATA MINING AND KNOWLEDGE DISCOVERY IN REAL WORLD

A large degree of the current interest in KDD is the result of the media interest surrounding successful KDD applications, for example, the focus articles within the last two years in Business Week, Newsweek, Byte, PC Week, and other large-circulation periodicals. Unfortunately, it is not always easy to separate fact from media hype. Nonetheless, several well documented examples of successful systems can rightly be referred to as KDD applications and have been deployed in operational use on large-scale real-world problems in science and in business.

In business, main KDD application areas includes marketing, finance (especially investment), fraud detection, manufacturing, telecommunications, and Internet agents.

- *Marketing:* In marketing, the primary application is database marketing systems, which analyze customer databases to identify different customer groups and forecast their behavior. Business Week (Berry 1994) estimated that over half of all retailers are using or planning to use database marketing, and those who do use it have good results; for example, American Express reports a 10- to 15- percent increase in credit-card use. Another notable marketing application is market-basket analysis (Agrawal et al. 1996) systems, which find patterns such as, "If customer bought X, he/she is also likely to buy Y and Z." Such patterns are valuable to retailers.
- *Investment:* Numerous companies use data mining for investment, but most do not describe their systems. One exception is LBS Capital Management. Its system uses expert systems, neural nets, and genetic algorithms to manage portfolios totaling $600 million; since its start in 1993, the system has outperformed the broad stock market (Hall, Mani, and Barr 1996).
- *Fraud Detection:* HNC Falcon and Nestor PRISM systems are used for monitoring credit card fraud, watching over millions of accounts. The FAIS system (Senator et al. 1995), from the U.S. Treasury Financial Crimes Enforcement Network, is used to identify financial transactions that might indicate money laundering activity.
- *Manufacturing:* The CASSIOPEE troubleshooting system, developed as part of a joint venture between General Electric and SNECMA, was applied by three major European airlines to diagnose and predict problems for the Boeing 737. To derive families of faults, clustering methods are used. CASSIOPEE received the European first prize for innovative applications (Manago and Auriol 1996).
- *Telecommunications:* The Telecommunications Alarm-sequence Analyzer (TASA) was built in cooperation with a manufacturer of telecommunications equipment and three telephone networks (Mannila, Toivonen, and Verkamo 1995). The system uses a novel framework for locating frequently occurring alarm episodes from the alarm stream and presenting them as rules. Large sets of discovered rules can be explored with flexible information- retrieval tools supporting interactivity and iteration. In this way, TASA offers pruning, grouping, and ordering tools to refine the results of a basic brute-force search for rules.

Check Your Progress

1. How does data mining works?
2. Define data mining process.
3. State whether the following statements are true or false:
   (i) KDD is the nontrivial process.
   (ii) A data mining process continues after a solution has been deployed.
4. Fill in the blanks:
   (i) The data-mining component of KDD currently relies heavily on known techniques from .....
   (ii) ..... is the application of specific algorithms for extracting patterns from data.

4.7 LET US SUM UP

Data mining can be viewed as a result of the natural evolution of information technology. An evolutionary path has been witnessed in the database industry in the development of *data collection and database creation, data management and data analysis* functionalities.

Data mining refers to *extracting or "mining" knowledge from large amounts of data.* Some other terms like knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging are also used for data mining.

The KDD is an iterative process. Once the discovered knowledge is presented to the user, the evaluation measures can be enhanced, the mining can be further refined, new data can be selected or further transformed, or new data sources can be integrated, in order to get different, more appropriate results.

Data mining derives its name from the similarities between searching for valuable information in a large database and mining rocks for a vein of valuable ore. Both imply either sifting through a large amount of material or ingeniously probing the material to exactly pinpoint where the values reside.

4.8 KEYWORDS

Data Mining: It refers to extracting or "mining" knowledge from large amounts of data.

KDD: Many people treat data mining as a synonym for another popularly used term, Knowledge Discovery from Data, or KDD.

Data Cleaning: To remove noise and inconsistent data.

Data Integration: Multiple data sources may be combined.

Data Selection: Data relevant to the analysis task are retrieved from the database.

Data Transformation: Where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.

Pattern Evaluation: To identify the truly interesting patterns representing knowledge based on some interestingness measures.

Knowledge Presentation: Visualization and knowledge representation techniques are used to present the mined knowledge to the user.

Data Warehouse Server: The data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.

Knowledge Base: This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns.

Data Mining Engine: This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

Pattern Evaluation Module: This component typically employs interestingness measures and interacts with the data mining modules so as to focus the search toward interesting patterns.

4.9 QUESTIONS FOR DISCUSSION

1. What is data mining?
2. Explain the meaning of the term –KDD.
3. What are the steps of knowledge discover process?
4. Discuss data mining architecture.
5. Explain the process of data mining.

Check Your Progress: Model Answers

1. Data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information
2. It contains the corresponding phases of a project, their respective tasks, and relationships between these tasks.
3. (i) True (ii) True
4. (i) Machine learning, pattern recognition (ii) Data mining

4.10 SUGGESTED READINGS

Jiawei Han, Micheline Kamber, *Data Mining – Concepts and Techniques*, Morgan Kaufmann Publishers, First Edition, 2003.

Michael J. A. Berry, Gordon S Linoff, *Data Mining Techniques*, Wiley Publishing Inc, Second Edition, 2004.

Alex Berson, Stephen J. Smith, *Data Warehousing, Data Mining & OLAP*, Tata McGraw Hill, Publications, 2004.

Sushmita Mitra, Tinku Acharya, Data Mining – Multimedia, Soft Computing and Bioinformatics, John Wiley & Sons, 2003.

Sam Anohory, Dennis Murray, *Data Warehousing in the Real World*, Addison Wesley, First Edition, 2000.

Baeza-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*, Addison Wesley, 1999.

R. Bird, *Introduction to Functional Programming using Haskell*, Prentice Hall, 1998.

K. Rennolls, "An intelligent framework (O-SS-E) for data mining, knowledge discovery and business intelligence," in *Proc. 16th Int.*

Workshop on Database and Expert System Applications, 2005, pp. 715-719.

LESSON

5

DATA MINING ISSUES AND DATABASE SUPPORT

CONTENTS

5.0 AIMS AND OBJECTIVES

After studying this lesson, you should be able to:

- Describe major issues in data mining
- Discuss application of data mining
- Explain how database supports data mining

5.1 INTRODUCTION

Data mining issues are related to three categories such as Issues related to the mining methodology and user-interaction issues, Issues related to the performance, and Issues relating to the diversity of database types. These issues have been discussed in this lesson.

Data mining should be applicable to any kind of data repository, as well as to transient data, such as data streams. The data repository may include relational databases, data warehouses, transactional databases, advanced database systems, flat files, data streams, and the Worldwide Web. All these data repository systems are discussed in this lesson.

5.2 MAJOR ISSUES IN DATA MINING

We can divide the major issues in data mining in three categories as follows:

Issues related to the Mining Methodology and User-interaction Issues

1. *Mining different kinds of knowledge in databases:* On the same database, the different users can be interested in different kinds of knowledge. Therefore, the data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterisation, discrimination, association, classification, clustering, trend and deviation analysis, and similarity analysis.
2. *Interactive mining of knowledge at multiple levels of abstraction:* Since it is difficult to know exactly what can be discovered within a database, the data mining process should be interactive. Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results. This will help the user to view data and discovered patterns at multiple granularities and from different angles.
3. *Incorporation of background knowledge:*Domain knowledge related to databases, such as integrity constraints and deduction rules, can help focus and speed up a data mining process, or judge the interestingness of discovered patterns.
4. *Data mining query languages and ad-hoc data mining:* A high level data mining query language need to be developed which can be integrated with a database or a data warehouse query language to allow users to describe ad-hoc data mining tasks by facilitating the specification of the relevant sets of data for analysis, the domain knowledge, the kinds of knowledge to be mined, and the conditions and interestingness constraints to be enforced on the discovered patterns.
5. *Presentation and visualization of data mining results:* The Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that the knowledge can be easily understood and directly usable by humans.

6. *Handling outlier or incomplete data:* The data stored in a database may reflect outliers — noise, exceptional cases, or incomplete data objects which may cause the accuracy of the discovered patterns to be poor. Data cleaning methods and data analysis methods that can handle outliers are required.
7. *Pattern evaluation: the interestingness problem:* A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, representing common knowledge or lacking novelty. The use of interestingness measures to guide the discovery process and reduce the search space is another active area of research.

Issues related to the Performance

1. *Efficiency and scalability of data mining algorithms:* To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable. That is, the running time of a data mining algorithm must be predictable and acceptable in large databases.
2. *Parallel, distributed, and incremental updating algorithms:* The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of parallel and distributed data mining algorithms.

Issues relating to the Diversity of Database Types

1. *Handling of relational and complex types of data:* There are many kinds of data stored in databases and data warehouses such as relational databases, complex data objects, hypertext and multimedia data, spatial data, temporal data, or transaction data. It is unrealistic to expect one system to mine all kinds of data due to the diversity of data types and different goals of data mining. Therefore, specific data mining systems should be constructed for mining specific kinds of data.
2. *Mining information from heterogeneous databases and global information systems:* Local and wide-area computer networks (such as the Internet) connect many sources of data, forming huge, distributed, and heterogeneous databases. The discovery of knowledge from different sources of structured, semi-structured, or unstructured data with diverse data semantics poses great challenges to data mining. Data mining may help disclose high-level data regularities in multiple heterogeneous databases that are unlikely to be discovered by simple query systems and may improve information exchange and interoperability in heterogeneous databases.

5.3 DATA MINING METRICS

Data mining allows the discovery of knowledge potentially useful and unknown, which may be useful and interesting depending upon the application and the user. Data mining can generate or discover, a very large number of patterns or rules. In some cases the number of rules can reach the millions. One can even think of a meta-mining phase to mine the oversized data mining results. To reduce the number of patterns or rules discovered that have a high probability to be non-interesting, one has to put a measurement on the patterns. However, this raises the problem of completeness. The user would want to discover all rules or patterns, but only those that are interesting. The measurement of how interesting a discovery is, often called interestingness, can be based on quantifiable objective elements such as *validity* of the patterns when tested on new data with some degree of *certainty*, or on some subjective depictions such as *understandability* of the patterns, *novelty* of the patterns, or *usefulness*. Discovered patterns can also be found interesting if they confirm or validate a hypothesis sought to be confirmed or unexpectedly contradict a common belief. This brings the issue of describing what is

interesting to discover, such as meta-rule guided discovery that describes forms of rules before the discovery process, and interestingness refinement languages that interactively query the results for interesting patterns after the discovery phase. Typically, measurements for interestingness are based on thresholds set by the user. These thresholds define the completeness of the patterns discovered. Identifying and measuring the interestingness of patterns and rules discovered, or to be discovered, is essential for the evaluation of the mined knowledge and the KDD process as a whole. While some concrete measurements exist, assessing the interestingness of discovered knowledge is still an important research issue.

5.4 SOCIAL IMPLICATIONS OF DATA MINING

Data mining, over the past two decades, has focused on the analysis of massive data sets, with data contents ranging from numeric and string attributes, to text, to audio/video, images, web pages, text and any other form of information that can be captured. There has been considerable interest in data streams and temporal data mining as well. In our experience, the use of data mining as a part of business operations to support business processes is perhaps the most challenging and useful application. Business operations are messy, badly documented and require enormous amounts of domain knowledge. They change and morph almost on a daily basis. The data generated are complex (sys logs, streams of measurements, free text comments) and are beset with data quality issues. Even a partial solution can lead to big gains in efficiency and performance, highly valued by the management.

5.4.1 Data Mining in the Real World

Historically, data mining has been in the hands of small teams of expert statisticians who produce a few models per year. However, recently companies invested heavily in building huge data warehouses (from a few terabytes to peta-bytes) that contain millions of records and thousands of variables; for example, 5,000 variables on 150 million customers and prospects. That has changed the economics of data mining. Now businesses want a return on that investment and are looking well beyond reporting and basic statistics. When they review their business activities, they see the need for 100s or 1000s of predictive models per year. Of course, very few companies can produce that many today, due to a lack of expert staff and appropriate tools.

5.4.2 Data Mining in Companies Today

Data mining is widely used in companies today, mostly for CRM, fraud detection, credit scoring, web mining (see for example). Yet, data mining is still mostly seen as an art to be practiced only by experts (statisticians, data miners, analysts). The development of data mining-based applications is strongly linked to the ever-increasing availability of large volumes of data. In the last ten years, companies have very heavily invested in the implementation of very large data warehouses, where the largest data bases now a days reach 20 to 100 Tbytes and comprise millions of customers and thousands of variables.

The production of a data-mining model will usually involve a triangular relationship (Figure 5.1).

The business users have an operational role (for example, they are in charge of designing/planning/executing marketing campaigns). They will list their requirements and will use the model results into their business processes.

The Analytics Department serves the needs of 20-50 business users, producing 5-10 models/year with 5-10 data mining experts. Data mining experts usually are a very scarce resource in the company; they will produce the models in line with business users requirements.

The IT department is in charge of maintaining the data bases and executing the models transferred to them by analysts to produce the results (e.g. scores) used by the business users.

This process will result in a typical 3-8 weeks delivery time, which is hardly compatible with the reactivity needed in extreme data mining.

What we need is industrial data mining at a greatly speeded pace – we call it *Extreme Data Mining* – This requires that a company put in place a *model factory* capable of churning out hundreds of models per year on terabytes of data (millions of lines – customers – and thousands of variables). The right data mining technology is indeed a key ingredient and it will have to provide employees flexibility, ease-of-use and productivity.

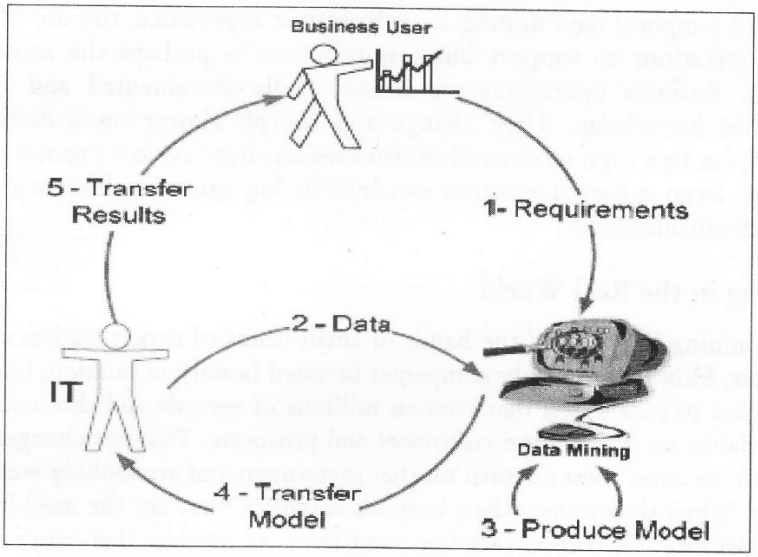


Figure 5.1: People involved in a Data Mining Project

5.4.3 Extreme Data Mining Examples

At KXEN, we have taken the *Extreme Data Mining* challenges at face value. Our tool KXEN Analytic framework has been developed to answer most of the previous requirements.

Cox Communications started using KXEN in September 2002 in its marketing department, to analyze its customers data base. It now produces hundreds of models for marketing campaigns in 26 regional markets from a data base of 10 million customers and 800 variables.

Barclays wanted to leverage the 100's of millions of inbound contacts; they had analyzed that that was where the greatest growth potential was. They thus incorporated events, triggers & predictive models to drive actions into the inbound channels, which needed acting in a timely way with a relevant contact (< 72 hours).

Vodafone wanted a complete analytic environment to support the easy creation and deployment of churn and cross sell/up sell models.

5.4.4 Data Mining in Procurement Business

Business Transformation Outsourcing (BTO) is rapidly becoming a popular way for enterprises (BTO clients) to streamline operations and reduce costs by offloading non-core operations to external BTO service providers. A commonly outsourced business operation is procurement, wherein a BTO service provider takes over the task of acquiring goods and services for the BTO client. The BTO service provider in such engagements can then do strategic sourcing by aggregating the spend of multiple BTO clients with its own spend, thereby increasing sales volumes, consolidating the supplier base, and negotiating better pricing deals by redirecting the bigger volumes to fewer, preferred suppliers. The BTO service provider shares a part of the savings generated with the BTO clients while retaining the bulk of the savings itself, making it a win-win situation for all parties. Data mining for performing such spend aggregation in IBM's procurement BTO practice, including the problems faced, techniques used, lessons learnt and open issues that still need to be addressed.

Introduction

Over the past few years, more and more enterprises are outsourcing some of their non-core business processes to external parties, a practice called Business Transformation Outsourcing (BTO). One commonly outsourced business operation is procurement, wherein a BTO service provider takes over the task of acquiring goods and services for the BTO client. Since enterprises invest a significant amount of resources in procurement activities, such outsourcing leads to immediate direct benefits across two fronts. First, by offloading non-core business processes to an external party, the enterprise can devote more resources to its core operations (in which it has maximum expertise), reduce the complexity and overhead of its business operations and streamline core-business processes by eliminating the processes in which it has little or no expertise. Second, substantial savings are immediately generated via reduction in human as well as non-human (e.g. procurement applications, h/w to run those applications) resources.

However, an additional significant chunk of savings comes indirectly via the BTO service provider who is able to acquire the same goods and services at substantially lower prices due to three main reasons. First, the service provider normally has significant expertise in procurement, and can utilize specialized and more efficient procurement processes. Second, the service provider can take advantages of economies of scale by taking on multiple, procurement BTO clients. Third, and most significantly, the service provider in such engagements can then do substantial strategic sourcing by aggregating the spend of the multiple BTO clients with its own spend, thereby increasing sales volumes, consolidating the supplier base, and negotiating better pricing deals by redirecting the bigger volumes to fewer, preferred suppliers. For a BTO service provider, such as IBM, that itself has a significant procurement spend, this allows substantial savings to be negotiated. Note that aggregation of spend across multiple BTO clients enables such strategic sourcing to be substantially enhanced over what could be done solely across a single enterprise. Figure 5.2 shows a cost function for volume-based deals that are typically negotiated with vendors. The function is commonly a step-function where the cost per unit decreases with increasing volume. Thus, if there are three different BTO clients that procure the same commodity with estimated volumes v1, v2 and v3, albeit from different suppliers, then the BTO service provider can negotiate a significantly better price by combining that volume ($v = v1 + v2 + v3$) and directing it all to a common, preferred supplier. The service provider shares a part of the savings generated with the BTO clients while retaining the bulk of the savings itself, making it a win-win situation for all parties.

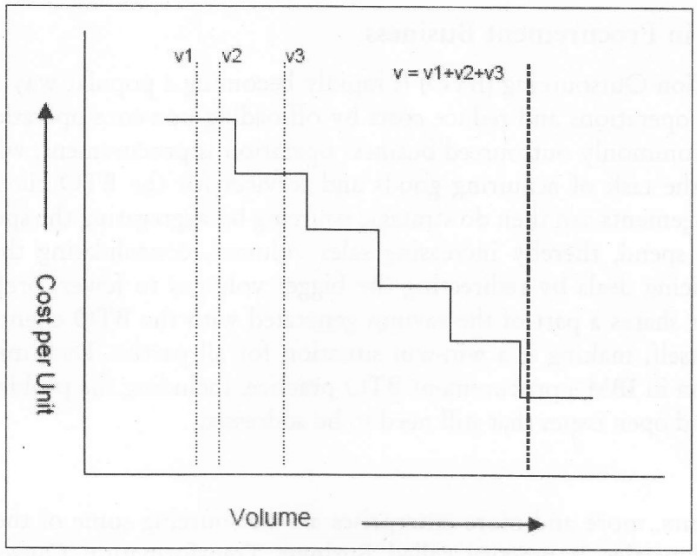


Figure 5.2

System Performance

The system has been used for a variety of companies across a wide spectrum of industry verticals, including, in addition to IBM itself, an electronics manufacturer (company 'B'), a transport company (company 'C'), a materials company (company 'D'), a consumer discretionary products manufacturer (company 'E'), and a procurement services company (company 'F', since acquired by IBM). Each company presented a different set of problems and issues. Some of them only had spend data linked to suppliers without any commodity information (B, E & F) while one had only commodity taxonomy information and no supplier data (D). Of these 5, only one has been an actual BTO spend aggregation engagement; the other 4 have been used to test and further develop the system.

Open Issues

We have applied the system for aggregating spends for a variety of companies across a wide spectrum of industry verticals with varying degree of success. While results for supplier normalization were generally good (often 80-90% accurate), commodity taxonomy mapping results were far more mixed (ranging from under 50% to generally around 70%). We have identified several different areas for improvement; some with known possible solutions; some without. First, there is a need to develop domain dictionaries to enable better matching for taxonomies across a wide spectrum of industry verticals. Second, we also need to enhance further the ability to better handle non-commodity taxonomy items, such as accounting categories that are often found in many company taxonomies. Third, we can likely get better performance by using the multiple applications of the system to various clients to develop a repository of "mapped" data and using it to learn supervised models, as discussed previously, in conjunction with the unsupervised methods and IR techniques. Fourth, we are now also focusing on improving the transactional commodity mapping capabilities of the system by building better indices and representations as well as improving our matching algorithms to better identify useful data in textual descriptions, as well as improved utilization of multiple descriptions for the same transaction to yield better matches.

5.5 DATABASE SUPPORT TO DATA MINING

Data mining comprises a step of the knowledge discovery process and is mainly concerned with methodologies for extracting knowledge artifacts, i.e. patterns, from large data repositories. Decision trees, association rules, clusters are some well known patterns coming from the data mining area. Patterns can also be found in other areas, such as Mathematics (e.g. patterns in sequences, in numbers, in graphs, in shapes etc.), Geometry, Signal Processing etc. Now a days, databases are huge, dynamic, come from different application domains and a lot of different and complex patterns can be extracted from those.

In order for someone to be able to exploit the information these patterns represent, an efficient and global (general) Pattern Base Management System (PBMS) for handling (storing/processing/retrieving) patterns is becoming necessary for a lot scientific areas apart from data mining.

There are two general paradigms for knowledge base construction: knowledge acquisition from a human expert, supported by appropriate expertise transfer tools, and empirical induction of knowledge from collections of examples. As noted by Gaines (1991), these approaches are fundamentally related in that domain experts have carefully constructed the examples used by machine learning programs. While most machine learning techniques can handle a limited amount of noisy or irrelevant data, they quickly break down if given an unstructured, unselective collection of information. Further, much general or common sense information is not stored in databases, and hence cannot be accessed by induction programs.

Expert system construction can clearly benefit from a combination of direct expert input and automated knowledge extraction. Human experts can provide structure, direction, and a commonsense understanding of the application domain. At the same time, however, the modeling rules formulated by human experts can include irrelevant and incorrect information. These biases and misconceptions can be mitigated by applying empirical *data mining* techniques to a domain database, to confirm, modify, or extend the human's domain model. These changes may then be incorporated into the expert system's knowledge base.

We use the term *data mining* to denote a shift from the machine-learning paradigm. Data mining does not generally produce a knowledge base; rather, the term refers to a technique that explores (mines) a rich, relatively unstructured set of data and retrieves from it unusual patterns, unexpected regularities, implicit information, etc. These interesting findings are generally not complete rules, but may suggest information that should be incorporated into a rule base.

Our approach to expert system construction is what Kerschberg (1992) refers to as "loosely coupled" in the sense that the database is not fully "understood" by the other components of the system. This is in contrast to an "expert deductive database system" (Schmidt 1991) which is tightly coupled to the database. Deductive databases comprise facts (database relations) and deductive rules which are general properties or constraining properties of the database relations. Such a system would, to all intents and purposes, be realized in a single program, perhaps in Prolog, with the database and rule-base working in union.

Production of an expert deductive database system from this scheme involves the separation of the deductive database (referred to as the object-level, defining the logical aspects of the problem) from the way in which the object-level is used to make deductions (referred to as the meta-level). An expert or an expert in conjunction with a knowledge engineer would produce the object-level of the system. The

meta-level would be produced by the knowledge engineer without reference to the expert since this part is less application dependent and only affects the efficiency of the deduction process.

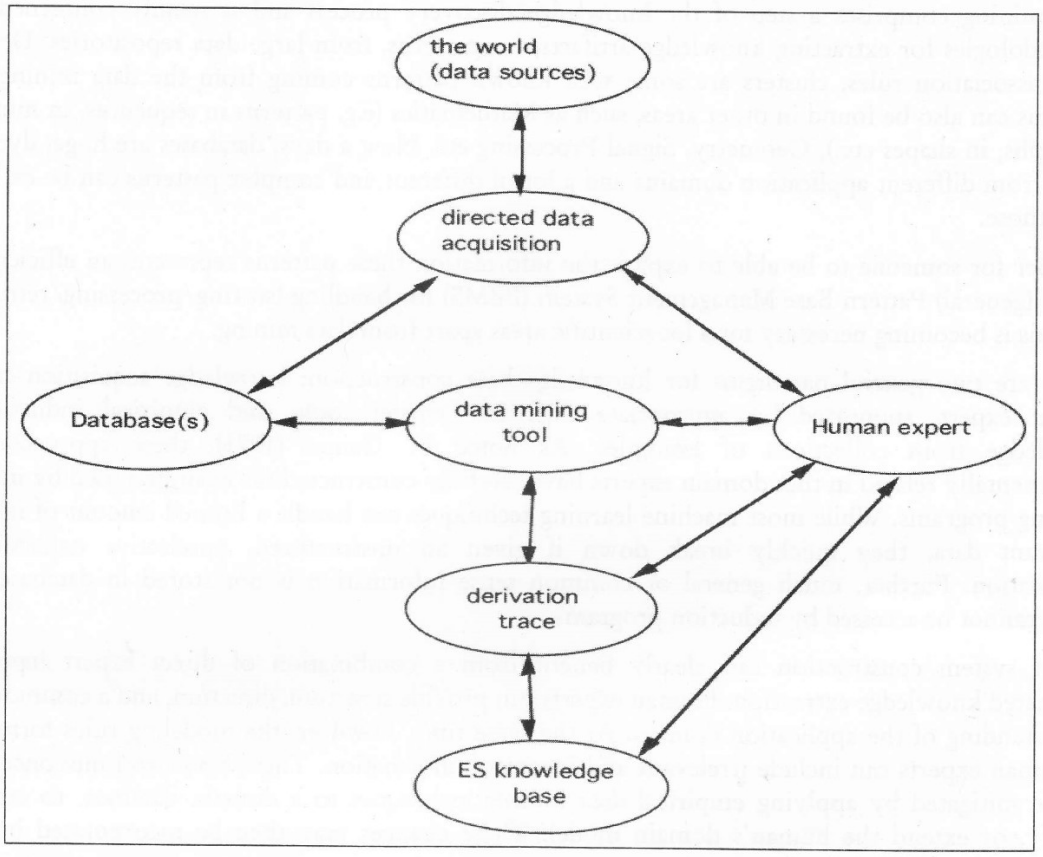


Figure 5.3

Figure 5.3 captures our model of the relationship between data mining and expert system construction. The construction process begins with an initial set of data, chosen by the human expert. As the expert system is constructed, the human expert may alter the composition of the databases by directing that additional or different data be collected. The human expert creates the knowledge base both through direct knowledge transfer and through mining one or more databases. Interactively guided by the expert, the mining tool efficiently organizes the search for findings in the database. These finding are then evaluated for inclusion in the knowledge base.

Data Mining - On What Kind of Data?

Data mining should be applicable to any kind of data repository, as well as to transient data, such as data streams. The data repository may include relational databases, data warehouses, transactional databases, advanced database systems, flat files, data streams, and the Worldwide Web. Advanced database systems include object-relational databases and specific application-oriented databases, such as spatial databases, time-series databases, text databases, and multimedia databases. The challenges and techniques of mining may differ for each of the repository systems.

A brief introduction to each of the major data repository systems listed above is given in subsequent section.

5.5.1 Flat Files

Flat files are actually the most common data source for data mining algorithms, especially at the research level. Flat files are simple data files in text or binary format with a structure known by the data mining algorithm to be applied. The data in these files can be transactions, time-series data, scientific measurements, etc.

5.5.2 Relational Databases

A database system or a Database Management System (DBMS) consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data. The software programs involve the following functions:

Mechanisms to create the definition of database structures:

- Data storage
- Concurrency control
- Sharing of data
- Distribution of data access
- Ensuring data consistency
- Security of the information stored, despite system crashes or attempts at unauthorized access.

A relational database is a collection of tables, each of which is assigned a unique name. Each table consists of a set of attributes (*columns* or *fields*) and usually stores a large set of tuples (*records* or *rows*). Each tuple in a relational table represents an object identified by a unique *key* and described by a set of attribute values. A semantic data model, such as an entity-relationship (ER) data model, is often constructed for relational databases. An ER data model represents the database as a set of entities and their relationships.

Some important points regarding the RDBMS are as follows:

- In RDBMS, tables can also be used to represent the relationships between or among multiple relation tables.
- Relational data can be accessed by database queries written in a relational query language, such as SQL, or with the assistance of graphical user interfaces.
- A given query is transformed into a set of relational operations, such as join, selection, and projection, and is then optimized for efficient processing.
- Trends and data patterns can be searched by applying data mining techniques on relational databases, we can go further by *searching for trends or data patterns.* For example, data mining systems can analyze customer data for a company to predict the credit risk of new customers based on their income, age, and previous credit information. Data mining systems may also detect deviations, such as items whose sales are far from those expected in comparison with the previous year.

- Relational databases are one of the most commonly available and rich information repositories, and thus they are a major data form in our study of data mining.

5.5.3 Data Warehouses

A data warehouse is a repository of information collected from multiple sources, stored under a unified schema, and that usually resides at a single site. Data warehouses are constructed via a process of data cleaning, data integration, data transformation, data loading, and periodic data refreshing. Figure 5.4 shows the typical framework for construction and use of a data warehouse for a manufacturing company.

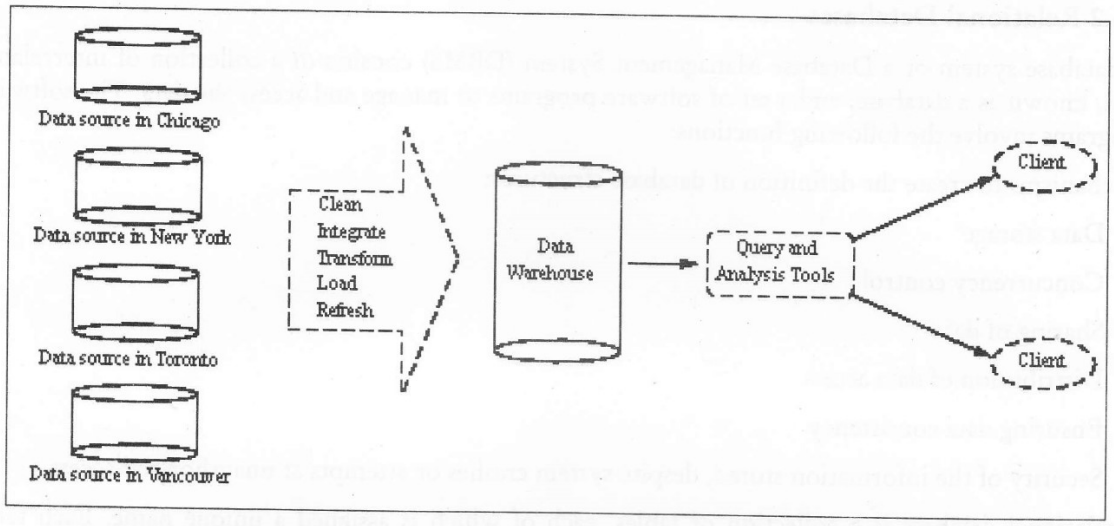


Figure 5.4: Typical Framework of a Data Warehouse for a Manufacturing Company

To facilitate decision making, the data in a data warehouse are *organized around major subjects*, such as customer, item, supplier, and activity. The data are stored to provide information from a *historical perspective* (such as from the past 510 years) and are typically *summarized*. For example, rather than storing the details of each sales transaction, the data warehouse may store a summary of the transactions per item type for each store or, summarized to a higher level, for each sales region.

A data warehouse is usually modeled by a multidimensional database structure, where each dimension corresponds to an attribute or a set of attributes in the schema, and each cell stores the value of some aggregate measure, such as *count* or *sales amount*. The actual physical structure of a data warehouse may be a relational data store or a multidimensional data cube. A data cube provides a multidimensional view of data and allows the precomputation and fast accessing of summarized data.

5.5.4 Data Cube

The data cube has a few alternative names or a few variants, such as, "multidimensional databases," "materialized views," and "OLAP (On-Line Analytical Processing)." The general idea of the approach is to materialize certain expensive computations that are frequently inquired, especially those involving aggregate functions, such as count, sum, average, max, etc., and to store such materialized views in a multi-dimensional database (called a "data cube") for decision support, knowledge discovery, and many other applications. Aggregate functions can be precomputed according to the grouping by different sets

or subsets of attributes. Values in each attribute may also be grouped into a hierarchy or a lattice structure. For example, "date" can be grouped into "day", "month", "quarter", "year" or "week", which forms a lattice structure. Generalization and specialization can be performed on a multiple dimensional data cube by "roll-up" or "drill-down" operations, where a roll-up operation reduces the number of dimensions in a data cube or generalizes attribute values to high-level concepts, whereas a drill-down operation does the reverse. Since many aggregate functions may often need to be computed repeatedly in data analysis, the storage of precomputed results in a multiple dimensional data cube may ensure fast response time and flexible views of data from different angles and at different abstraction levels.

For example, a relation with the schema "sales (part; supplier; customer; sale price)" can be materialized into a set of eight views as shown in Figure 5.5, where psc indicates a view consisting of aggregate function values (such as total sales) computed by grouping three attributes part, supplier, and customer, p indicates a view consisting of the corresponding aggregate function values computed by grouping part alone, etc.

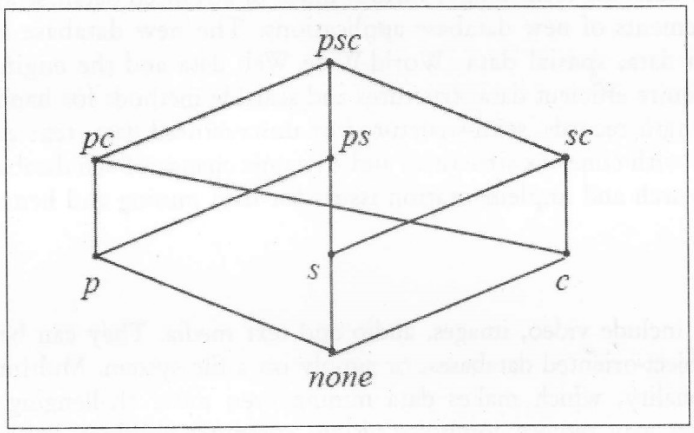


Figure 5.5: Eight Views of Data Cubes For Sales Information

5.5.5 Transaction Database

A transaction database is a set of records representing transactions, each with a time stamp, an identifier and a set of items. Associated with the transaction files could also be descriptive data for the items. For example, in the case of the video store, the rentals table such as shown in Figure 5.6, represents the transaction database. Each record is a rental contract with a customer identifier, a date, and the list of items rented (i.e. video tapes, games, VCR, etc.). Since relational databases do not allow nested tables (i.e. a set as attribute value), transactions are usually stored in flat files or stored in two normalised transaction tables, one for the transactions and one for the transaction items. One typical data mining analysis on such data is the so-called market basket analysis or association rules in which associations between items occurring together or in sequence are studied.

Rental				
Transaction ID	Date	Time	Customer ID	Item List
T12345	10/12/06	10:40	C1234	I1000

Figure 5.6: Fragment of a Transaction Database for the Rentals at our Video Store

5.5.6 Advanced Database Systems and Advanced Database Applications

With the advances of database technology, various kinds of advanced database systems have emerged to address the requirements of new database applications. The new database applications include handling multimedia data, spatial data, World-Wide Web data and the engineering design data. These applications require efficient data structures and scalable methods for handling complex object structures, variable length records, semi-structured or unstructured data, text and multimedia data, and database schemas with complex structures and dynamic changes. Such database systems may raise many challenging research and implementation issues for data mining and hence discussed in short as follows:

Multimedia Databases

Multimedia databases include video, images, audio and text media. They can be stored on extended object-relational or object-oriented databases, or simply on a file system. Multimedia is characterized by its high dimensionality, which makes data mining even more challenging. Data mining from multimedia repositories may require computer vision, computer graphics, image interpretation, and natural language processing methodologies.

Spatial Databases

Spatial databases are databases that, in addition to usual data, store geographical information like maps, and global or regional positioning. Such spatial databases present new challenges to data mining algorithms.

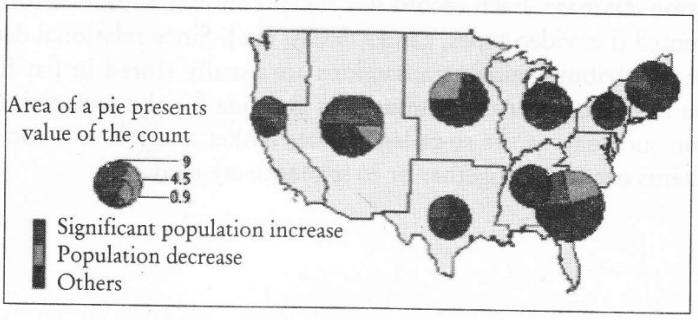


Figure 5.7 Visualization of Spatial OLAP

Time-series Databases

Time-series databases contain time related data such stock market data or logged activities. These databases usually have a continuous flow of new data coming in, which sometimes causes the need for a challenging real time analysis. Data mining in such databases commonly includes the study of trends and correlations between evolutions of different variables, as well as the prediction of trends and movements of the variables in time.

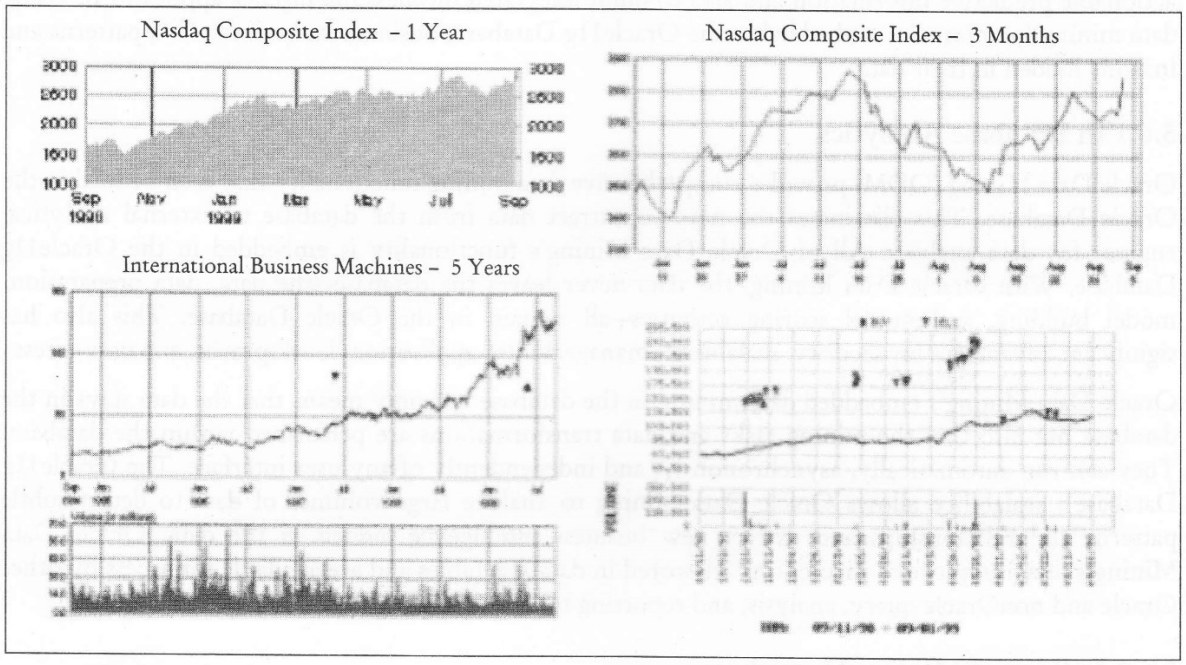


Figure 5.8 Shows some examples of Time-series Data

Worldwide Web

The Worldwide Web is the most heterogeneous and dynamic repository available. A very large number of authors and publishers are continuously contributing to its growth and metamorphosis, and a massive number of users are accessing its resources daily. Data in the Worldwide Web is organized in inter-connected documents. These documents can be text, audio, video, raw data, and even applications. Conceptually, the Worldwide Web is comprised of three major components: The content of the Web, which encompasses documents available; the structure of the Web, which covers the hyperlinks and the relationships between documents; and the usage of the web, describing how and when the resources are accessed. A fourth dimension can be added relating the dynamic nature or evolution of the documents. Data mining in the Worldwide Web, or web mining, tries to address all these issues and is often divided into web content mining, web structure mining and web usage mining.

Engineering Design Data

Database technology has evolved in parallel to the evolution of software to support engineering. In these applications relatively simple operations are performed on large volumes of data with uniform structure. The engineering world, on the other hand, is full of computationally intensive, logically complex applications requiring sophisticated representations. Recent developments in database technology

emphasize the need to provide general-purpose support for the type of functions involved in the engineering process such as the design of buildings, system components, or integrated circuits etc.

5.6 THE ORACLE DATABASE WITH DATA MINING OPTION

The Data Mining option to the Oracle Database 11g Release 1 enables enterprises to produce actionable predictive information and also to build integrated business intelligence applications. Using data mining functionality embedded in the Oracle11g Database, business analysts can find patterns and insights hidden in their data.

5.6.1 In Database Analytics

Oracle Data Mining (ODM) provides comprehensive data mining functionality that is embedded in the Oracle Database. This eliminates the need to extract data from the database to external analytical engines for data analysis. All of Oracle Data Mining's functionality is embedded in the Oracle11g Database. With Oracle Data Mining, the data never leaves the database—the data, data preparation, model building, and model scoring activities—all remain in the Oracle Database. This also has significant advantages for security, scalability, manageability, application development, and user access.

Oracle Data Mining's embedded data mining in the database not only means that the data stays in the database but also that the mining tasks and data transformations are performed within the database. They can run automatically, asynchronously, and independently of any user interface. The Oracle11g Database's scalability allows Oracle Data Mining to analyze large volumes of data to detect subtle patterns and relationships and extract new business intelligence hidden in the data. Oracle Data Mining's new insights and predictions are stored in database tables and are available for access by other Oracle and nonOracle query, analysis, and reporting tools and applications.

5.6.2 Full Set of Mining Algorithms

The Oracle Data Mining provides support for a wide range of data mining model building and evaluating functionality including: classification, regression, clustering, associations, anomaly detection, text mining, attribute importance and feature extraction. Text Mining, and Anomaly Detection use Support Vector Machines, Attribute Importance uses Minimum Description Length (MDL), Associations uses A Priori, and Feature Extraction uses Non-negative Matrix Factorization (NMF).

5.6.3 Mining Activities

The Oracle Data Miner Graphical User Interface (GUI) employs Mining Activities that not only prescribe the correct order of operations and perform all algorithm-required data transformations, but also provide intelligent settings and optimizations for all parameters; however, the expert can expose all parameters in order to override default values.

5.7 KNOWLEDGE BASE CONSTRUCTION AND MAINTENANCE

To construct an expert system that will predict an employee's expected salary range, given a description of the employee's work history, age, sex, etc. A database of information on current employees is available for mining.

5.7.1 Knowledge Base Construction

We take as our model an expert system that is constructed (in part) from concepts derived from one or more databases. The knowledge engineer guides the mining of these databases, and incorporates the concepts located into the expert system. This model differs from the classic machine-learning model of expert system construction, in which the expert system is automatically derived from a single database. Our model is advantageous when there is no single database containing all information needed to construct the knowledge base, and when common sense or other unstored information must be added to the system.

We use data mining to suggest information for possible inclusion in rules. The data may yield rules suitable for direct inclusion in the knowledge base. For example, if we wish to determine the employment categories currently receiving the highest salaries, the rule "employees holding the rank of vice-president or above receive the highest salary" can be directly extracted from the database. In some cases the information extracted must be augmented with additional knowledge held by the expert, or combined with information mined from a different database.

Example: We may extract the information that "graduates of the University of Texas receive higher than average salaries". The knowledge engineer may possess additional knowledge that it is the specialized training available at this university that provides a professional advantage, and that other universities are beginning to offer this training. A more general rule would be formed, then including a test for this training rather than a test for a University of Texas degree.

Alongside useful information, the data mining technique may also capture irrelevant information. One piece of information that can be mined from an employee database is that no one under the age of 16 receives a salary. A common sense understanding of the world allows a knowledge engineer to discard this fact, since the child labour laws guarantee that the company will never have an employee under 16, much less an under-aged employee earning a high salary.

5.7.2 Knowledge Base Maintenance

As the database is updated, changes in the data should be reflected in the rule base:

After the expert system is constructed, the employee database will continue to be updated, and as a consequence the knowledge base may have to be refined (Figure 5.9).

These changes to the database may be insignificant, or they may indicate major shifts in the high-level information implicit in the data. A link must be maintained between the portions of the database used to construct a rule, the Explora query that extracted the information relevant to the rule, and the rule itself. We currently adopt a naive approach to maintenance: data updates are monitored, and when it appears likely that "significant" changes have occurred, the Explora query is re-executed. The current query results are compared with previous results (stored in a rule derivation trace). If the results differ, then the rule and the derivation trace are presented to the knowledge engineer to determine whether the knowledge base should be modified. The following discussion illustrates the problems inherent in maintenance under a loosely-coupled architecture. Further work is necessary to provide a more principled method of supporting knowledge base maintenance.

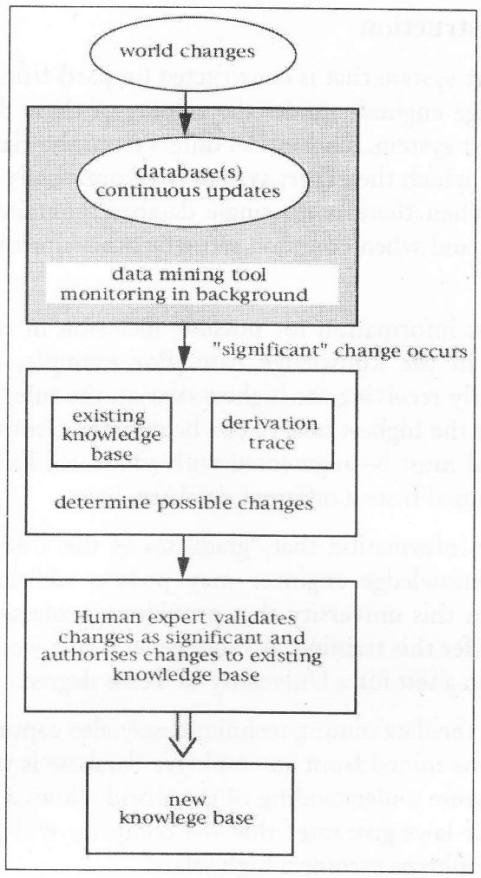


Figure 5.9

Example: Our initial query to Explora is, "what factors indicate that the employee will receive a high salary?" and the system's response is that "males with a high school education receive the highest salaries". We add the rule to the knowledge base.

male and high school education → high salary

5.8 DATABASE LANGUAGE AND QUERY EXECUTION

In Query Languages Supporting Descriptive Rule Mining provide a comparison of features of available relational query languages for data mining, such as DMQL, MSQL, MINE RULE, and standardization efforts for coupling database technology and data mining systems, such as OLEDB-DM and PMML.

Declarative Data Mining using SQL-3 shows a new approach, compared to existing SQL approaches, to mine association rules from an object-relational database: it uses a recursive join in SQL-3 that allows no restructuring or preprocessing of the data. It proposes a new mine by SQL-3 operator for capturing the functionality of the proposed approach.

Towards a Logic Query Language for Data Mining presents a logic database language with elementary data mining mechanisms, such as user-defined aggregates that provide a model, powerful and general as well, of the relevant aspects and tasks of knowledge discovery.

Data Mining Query Language for Knowledge Discovery in a Geographical Information System presents *SDMOQL* a spatial data mining query language for knowledge extraction from GIS. The language supports the extraction of classification rules and association rules, the use of background models, various interestingness measures and the visualization.

Towards Query Evaluation in Inductive Databases using Version Spaces studies inductive queries. These ones specify constraints that should be satisfied by the data mining patterns in which the user is interested. This work investigates the properties of solution spaces of queries with monotonic and anti-monotonic constraints and their boolean combinations.

5.9 SUPPORT FOR KNOWLEDGE DISCOVERY PROCESS

Data base knowledge discovery supports many ways to the data mining applications.

- *Interactivity, Scalability and Resource Control for Efficient KDD Support in DBMS* proposes a new approach for combining preprocessing and data mining operators in a KDD-aware implementation algebra. In this way data mining operators can be integrated smoothly into a database system, thus allowing interactivity, scalability and resource control. This framework is based on the extensive use of pipelining and is built upon an extended version of a specialized Database index.
- *Frequent itemset Discovery with SQL using Universal Quantification* investigates the integration of data analysis functionalities into two basic components of a database management system: query execution and optimization. It employs universal and existential quantifications in queries and a vertical layout to ease the set containment operations needed for frequent itemsets discovery.
- *Deducing Bounds on the Support of itemsets* provides a complete set of rules for deducing tight bounds on the support of an itemset if the support of all its subsets are known. These bounds can be used by the data mining system to choose the best access path to data and provide a better representation of the collection of frequent itemsets.
- *Model-independent Bounding of the Supports of Boolean Formulae in Binary Data* considers frequencies of arbitrary boolean formulas, a new class of aggregates: the summaries. These ones are computed for descriptive purposes on a sparse binary data set. This work considers the problem of finding tight upper bounds on these frequencies and gives a general formulation of the problem with a linear programming solution.
- *Integrity Constraints over Association Rules* investigates the notion of integrity constraints in inductive databases. This concept is useful in detecting inconsistencies in the results of common data mining tasks. This work proposes a form of integrity constraints called *association map constraints* that specify the allowed variations in confidence and support of association rules.

Check Your Progress

1. Fill in the blanks:
   (i) A ..... is a collection of tables, each of which is assigned a unique name.
   (ii) A ..... is a repository of information collected from multiple sources, stored under a unified schema, and that usually resides at a single site.
2. Write short notes on Transaction databases.

5.10 LET US SUM UP

The major components of a typical data mining system are- information repository, database or data warehouse server, knowledge base, data mining engine, pattern evaluation module and user interface.

Data mining should be applicable to any kind of data repository, as well as to transient data, such as data streams. The data repository may include relational databases, data warehouses, transactional databases, advanced database systems, flat files, data streams, and the Worldwide Web.

A data warehouse is a repository for long term storage of data from multiple sources, organised so as to facilitate management decision making.

Major issues in data mining are:

- Mining different kinds of knowledge in databases.
- Interactive mining of knowledge at multiple levels of abstraction.
- Incorporation of background knowledge.
- Data mining query languages and ad-hoc data mining.
- Presentation and visualisation of data mining results.
- Handling outlier or incomplete data.
- Pattern evaluation: the interestingness problem.
- Efficiency and scalability of data mining algorithms.
- Handling of relational and complex types of data.
- Mining information from heterogeneous databases and global information systems.

5.11 KEYWORDS

BPO: Business Process Outsourcing

Flat Files: Flat files are actually the most common data source for data mining algorithms, especially at the research level.

Relational Databases: A database system or a database management system (DBMS) consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data.

Data Warehouses: A data warehouse is a repository of information collected from multiple sources, stored under a unified schema, and that usually resides at a single site.

Data Cube: A data cube provides a multidimensional view of data and allows the precomputation and fast accessing of summarized data.

5.12 QUESTIONS FOR DISCUSSION

1. Why data mining is crucial to the success of a business?
2. Describe two challenges to data mining regarding performance issues.

3. Describe three challenges to data mining regarding data mining methodology and user-interaction issues.
4. What is a data mining application in real world?
5. Explain the business data mining applications.

Check Your Progress: Model Answers

1. (i) Relation database
   (ii) Data warehouse
2. A transaction database is a set of records representing transactions, each with a time stamp, an identifier and a set of items. Associated with the transaction files could also be descriptive data for the items.

5.13 SUGGESTED READINGS

Jiawei Han, Micheline Kamber, *Data Mining - Concepts and Techniques*, Morgan Kaufmann Publishers, First Edition, 2003.

Michael J. A. Berry, Gordon S Linoff, *Data Mining Techniques*, Wiley Publishing Inc, Second Edition, 2004.

Alex Berson, Stephen J. Smith, *Data Warehousing, Data Mining & OLAP*, Tata McGraw Hill, Publications, 2004.

Sushmita Mitra, Tinku Acharya, *Data Mining - Multimedia, Soft Computing and Bioinformatics*, John Wiley & Sons, 2003.

Sam Anohory, Dennis Murray, *Data Warehousing in the Real World*, Addison Wesley, First Edition, 2000.

Baeza-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*, Addison Wesley, 1999.

R. Bird, *Introduction to Functional Programming using Haskell*, Prentice Hall, 1998.

K. Rennolls, "An intelligent framework (O-SS-E) for data mining, knowledge discovery and business intelligence," in *Proc. 16th Int.*

Workshop on Database and Expert System Applications, 2005, pp. 715-719.

UNIT IV