Dataware Housing and Datamining

DCAP603

LOVELY
PROFESSIONAL
UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

DATA WAREHOUSING AND DATA MINING

Produced & Printed by
EXCEL BOOKS PRIVATE LIMITED
A-45, Naraina, Phase-I,
New Delhi-110028
for
Lovely Professional University
Phagwara

SYLLABUS

Data Warehousing and Data Mining

In this, Students study the issues involved in planning, designing, building, populating, and maintaining a successful data warehouse. It's objective is to:

- Analyze query plans of execution.
- Manage data loads and transactions
- Describe methods and tools for accessing and analyzing data warehouse.
- Describe various technologies to implement data warehouse.

S. No.	Description
1	***Data Warehouse Practice:*** Data warehouse components, Designing the Data Warehouse, Getting Heterogeneous Data into the Warehouse, Getting Multidimensional Data out of the Warehouse.
2	***Data Warehouse Research-Issues and Research:*** Data Extraction and Reconciliation, Data Aggregation and Customization, Query Optimization, Update Propagation, Modelling and Measuring Data Warehouse Quality, Some Major Research Projects in Data Warehousing, Three Perspectives of Data Warehouse Metadata.
3	***Source Integration:*** The Practice of Source Integration, Research in Source Integration, Towards Systematic Methodologies for Source Integration.
4	***Data Warehouse Refreshment:*** Data Warehouse Refreshment, Incremental Data Extraction, Data Cleaning,
5	***Data Warehouse Refreshment:*** Update Propagation into Materialized Views, Towards a Quality-Oriented Refreshment Process, Implementation of the Approach
6	***Multidimensional Data Models and Aggregation:*** Multidimensional View of Information, ROLAP Data Model, MOLAP Data Model, Logical Models for Multidimensional Information, Conceptual Models for Multidimensional Information
7	***Query Processing and Optimization:*** Description and Requirements for Data Warehouse Queries, Query Processing Techniques.
8	***Metadata and Warehouse Quality***: Matadata Management in Data Warehouse Practice, A repository Model for the DWQ Framework, Defining Data Warehouse Quality.
9	***Metadata and Data Warehouse Quality:*** Representing and Analyzing Data Warehouse Quality, Quality Analysis in Data Staging.
10	***Quality-Driven Data Warehouse Design:*** Interactions between Quality Factors and DW Tasks, The DWQ Data Warehouse Design Methodology, Optimizing the Materialization of DW Views

CONTENTS

Unit 1: Data Warehouse Practice

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Know data warehouse concept
- Explain data warehouse components
- Describe data warehouse architecture

Notes

Introduction

Remember using Lotus 1-2-3-? This was your first taste of "What if?" processing on the desktop. This is what a data warehouse is all about-using information your business has gathered to help it react better, smarter, quicker and more efficiently.

To expand upon this definition, a data warehouse is a collection of corporate information, derived directly from operational systems and some external data sources. Its specific purpose is to support business decisions, not business operations. This is what a data warehouse is all about, helping your business ask "What if?" questions. The answers to these questions will ensure your business is proactive, instead of reactive, a necessity in today's information age.

The industry trend today is moving towards more powerful hardware and software configurations. With these more powerful configurations, we now have the ability to process vast volumes of information analytically, which would have been unheard of ten or even five years ago. A business today must be able to use this emerging technology or rum the risk of being information under-loaded. You read that correctly under-loaded the opposite of overloaded. Overloaded means you are so overwhelmed by the enormous gult of information. It's hard to wade through it to determine what is important. If you are under-loaded, you are information deficient. You cannot cope with decision-making exceptions because you do not know where you stand. You are missing critical pieces of information required to make informed decisions.

In today's world, you do not want to be the country mouse. In today's world, full of vast amounts of unfiltered information, a business that does not effectively use technology to shift through that information will not survive the information age. Access to and the understanding of information is power. This power equates to a competitive advantage are survival.

1.1 What is a Data Warehouse?

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 organization'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.

According to W. H. Inman, a leading architect in the construction of data warehouse systems, "a data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process." The four keywords, subject-oriented, integrated, time-variant, and non-volatile, distinguish data warehouses from other data repository systems, such as relational database systems, transaction processing systems, and file systems. Let us understand the four key words in more detail as follows:

1. ***Subject-oriented:*** A data warehouse focuses on the modeling and analysis of data for decision makers. Therefore, data warehouses typically provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

2. ***Integrated:*** As the data warehouse is usually constructed by integrating multiple heterogeneous sources, such as relational databases, flat files, and on-line transaction records, the data cleaning and data integration techniques need to be applied to ensure consistency in naming conventions, encoding structures, attribute measures, and so on.

3. ***Time-variant:*** Data are stored to provide information from a historical perspective (e.g., the past 5-10 years). Every key structure in the data warehouse contains, either implicitly or explicitly, an element of time.

4. ***Non-volatile:*** A data warehouse is always a physically separate store of data transformed from the application data found in the operational environment. Due to this separation, a data warehouse does not require transaction processing, recovery, and concurrency control mechanisms. It usually requires only two operations in data accessing: initial loading of data and access of data.

Example: A typical data warehouse is organised around major subjects, such as customer, vendor, product, and sales rather than concentrating on the day-to-day operations and transaction processing of an organization.

1.1.1 Use of Data Warehouses in Organisations

Many organisations 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, and

5. Data warehousing is also very useful from the point of view of heterogeneous database integration. Many organisations typically collect diverse kinds of data and maintain large databases from multiple, heterogeneous, autonomous, and distributed information sources.

1.1.2 Query Driven Approach versus Update Driven Approach for Heterogeneous Database Integration

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, summarised, 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.

Notes

1.1.3 Differences between Operational Database Systems and Data Warehouses

The first major stepping stone in understanding Data Warehousing is to grasp the concepts and differences between the two overall database categories. The type most of us are used to dealing with is the On Line Transactional Processing (OLTP) category. The other major category is On Line Analytical Processing (OLAP).

OLTP is what we characterise as the ongoing day-to-day functional copy of the database. It is where data is added and updated but never overwritten or deleted. The main needs of the OLTP operational database being easily controlled insertion and updating of data with efficient access to data manipulation and viewing mechanisms. Typically only single record or small record-sets should be manipulated in a single operation in an OLTP designed database. The main thrust here is to avoid having the same data in different tables. This basic tenet of Relational Database modeling is known as "normalising" data.

OLAP is a broad term that also encompasses data warehousing. In this model data is stored in a format, which enables the efficient creation of data mining/reports. OLAP design should accommodate reporting on very large record sets with little degradation in operational efficiency. The overall term used to describe taking data structures in an OLTP format and holding the same data in an OLAP format is "Dimensional Modeling" It is the primary building block of Data Warehousing.

The major distinguishing features between OLTP and OLAP are summarised as follows:

Feature	OLTP System	OLAP System
Characteristic	Operational Processing	Informational Processing
Users	Clerks, clients, and information technology professionals.	Knowledge workers, including managers, executives, and analysts.
System orientation	Customer oriented and used for transaction and query processing Day to day operations	Market-oriented and used for data analysis long term informational requirements, decision support.
Data contents	Manages current data that typically, are too detailed to be easily used for decision making.	Manages large amounts of historical data, provides facilities for summarisation and aggregation, and stores and manages information at different levels of granularity.
Database design	Adopts an entity-relationship (ER) data model and an application-oriented database design	Adopts either a star or snowflake model and a subject-oriented database design.
View	Focuses mainly on the current data within an enterprise or department, without referring to historical data or data in different organisations.	In contrast, an OLAP system often spans multiple versions of a database schema, due to the evolutionary process of an organisation. OLAP systems also deal with information that originates from different organisations, integrating information from many data stores.
Volume of data	Not very large	Because of their huge volume, OLAP data are stored on multiple storage media.
Access patterns	Consists mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms.	Accesses to OLAP systems are mostly read-only operations (since most data warehouses store historical rather than up-to-date information), although many could be complex queries.

Contd...

Access mode	Read/write	Mostly write
Focus	Data in	Information out
Operations	Index/hash on primary key	Lots of scans
Number of records accessed	Tens	Millions
Number of users	Thousands	Hundreds
DB size	100 MB to GB	100 GB to TB
Priority	High performance, high availability	High flexibility, end-user autonomy
Metric	Transaction throughput	Query response time

1.1.4 Need to Build a Data Warehouse

You know that data warehouse queries are often complex. They involve the computation of large groups of data at summarised levels and may require the use of special data organisation, access, and implementation methods based on multidimensional views. Processing OLAP queries in operational databases would substantially degrade the performance of operational tasks. Moreover, an operational database supports the concurrent processing of several transactions as well recovery mechanism such as locking and logging to ensure the consistency and robustness of transactions. An OLAP query often needs read-only access of data records for summarisation and aggregation. Concurrency control and recovery mechanisms, if applied for such OLAP operations, may jeopardise the execution of concurrent transactions and thus substantially reduce the throughput of an OLTP system.

1.2 Data Warehousing

Data mining potential can be enhanced if the appropriate data has been collected and stored in a data warehouse. A data warehouse is a relational database management system (RDBMS) designed specifically to meet the needs of transaction processing systems. It can be loosely defined as any centralized data repository which can be queried for business benefits but this will be more clearly defined later. Data warehousing is a new powerful technique making it possible to extract archived operational data and overcome inconsistencies between different legacy data formats. As well as integrating data throughout an enterprise, regardless of location, format, or communication requirements it is possible to incorporate additional or expert information. It is,

The logical link between what the managers see in their decision support EIS applications and the company's operational activities.

John McIntyre of SAS Institute Inc.

In other words the data warehouse provides data that is already transformed and summarized, therefore making it an appropriate environment for more efficient DSS and EIS applications.

1.3 Characteristics of Data Warehouse

According to Bill Inmon, author of Building the data Warehouse and the guru who is widely considered to be the originator of the data warehousing concept, there are generally four characteristics that describe a data warehouse:

1. ***Subject oriented:*** Data are organized according to subject instead of application e.g. an insurance company using a data warehouse would organize their data by customer, premium, and claim, instead of by different products (auto, life, etc,). The data organized by subject contain only the information necessary for decision support processing.

Notes

2. ***Integrated:*** When data resides in many separate applications in the operational environment, encoding of data is often inconsistent. For instance, in one application, gender might be coded as "m" and "f" in another by 0 and 1. When data are moved from the operational environment into the data warehouse, they assume a consistent coding convention e.g. gender data is transformed to "m" and "f".

3. ***Time variant:*** The data warehouse contains a place for storing data that are five to 10 years old, or older, to be used for comparisons, trends, and forecasting. These data are not updated.

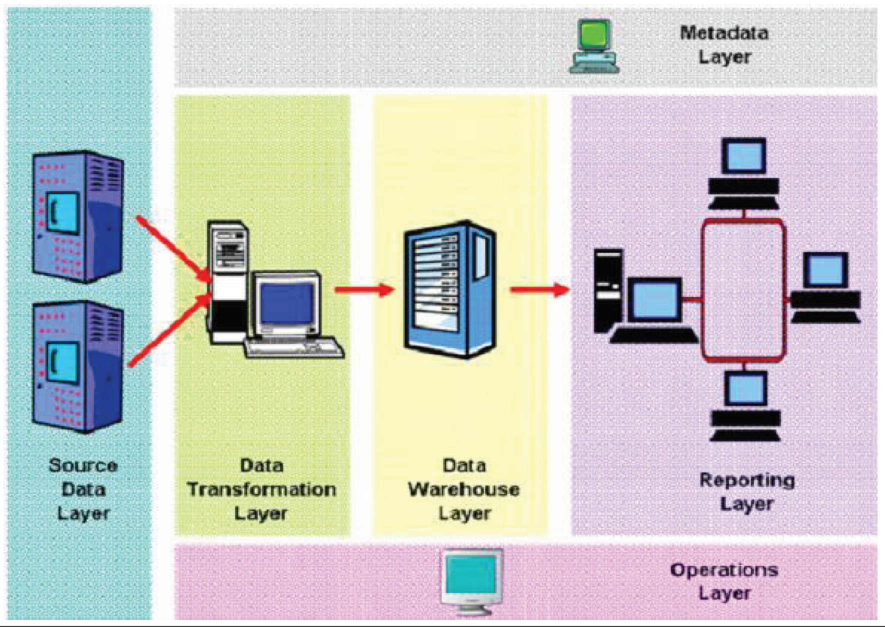
4. ***Non volatile:*** Data are not updated or changed in any way once they enter the data warehouse, but are only loaded and accessed.

Task You know about store room and warehouse. Exactly what the difference between warehouse and data warehouse? Explain with the suitable of suitable example.

1.4 Data Warehouse Components

The primary components of the majority of data warehouses are shown in the Figure 1.1 and described in more detail below:

Figure 1.1: Components of Data Warehouse



Data Sources

Data sources refer to any electronic repository of information that contains data of interest for management use or analytics. This definition covers mainframe databases (e.g. IBM DB2, ISAM, Adabas, Teradata, etc.), client-server databases (e.g. Teradata, IBM DB2, Oracle database, Informix, Microsoft SQL Server, etc.), PC databases (e.g. Microsoft Access, Alpha Five), spreadsheets (e.g. Microsoft Excel) and any other electronic store of data. Data needs to be passed from these systems to the data warehouse either on a transaction-by-transaction basis for real-time data warehouses or on a regular cycle (e.g., daily or weekly) for offline data warehouses.

Data Warehouse

The data warehouse is normally (but does not have to be) a relational database. It must be organized to hold information in a structure that best supports not only query and reporting, but also advanced analysis techniques, like data mining. Most data warehouses hold information for at least 1 year and sometimes can reach half century, depending on the business/operations data retention requirement. As a result these databases can become very large.

Reporting

The data in the data warehouse must be available to the organisation's staff if the data warehouse is to be useful. There are a very large number of software applications that perform this function, or reporting can be custom-developed. Examples of types of reporting tools include:

1. ***Business intelligence tools:*** These are software applications that simplify the process of development and production of business reports based on data warehouse data.
2. ***Executive information systems (known more widely as Dashboard (business)***: These are software applications that are used to display complex business metrics and information in a graphical way to allow rapid understanding.
3. ***OLAP Tools***: OLAP tools form data into logical multi-dimensional structures and allow users to select which dimensions to view data by.
4. ***Data Mining***: Data mining tools are software that allow users to perform detailed mathematical and statistical calculations on detailed data warehouse data to detect trends, identify patterns and analyze data.

Metadata

Metadata, or "data about data", is used not only to inform operators and users of the data warehouse about its status and the information held within the data warehouse, but also as a means of integration of incoming data and a tool to update and refine the underlying DW model.

Example: Data warehouse metadata include table and column names, their detailed descriptions, their connection to business meaningful names, the most recent data load date, the business meaning of a data item and the number of users that are logged in currently.

Operations

A data warehouse operation is comprised of the processes of loading, manipulating and extracting data from the data warehouse. Operations also cover user management, security, capacity management and related functions.

Optional Components

In addition, the following components exist in some data warehouses:

1. ***Dependent Data Marts:*** A dependent data mart is a physical database (either on the same hardware as the data warehouse or on a separate hardware platform) that receives all its information from the data warehouse. The purpose of a Data Mart is to provide a sub-set of the data warehouse's data for a specific purpose or to a specific sub-group of the organization. A data mart is exactly like a data warehouse technically, but it serves a different business purpose: it either holds information for only part of a company (such as a division), or it holds a small selection of information for the entire company (to support

extra analysis without slowing down the main system). In either case, however, it is not the organization's official repository, the way a data warehouse is.

2. ***Logical Data Marts:*** A logical data mart is a filtered view of the main data warehouse but does not physically exist as a separate data copy. This approach to data marts delivers the same benefits but has the additional advantages of not requiring additional (costly) disk space and it is always as current with data as the main data warehouse. The downside is that Logical Data Marts can have slower response times than physicalized ones.

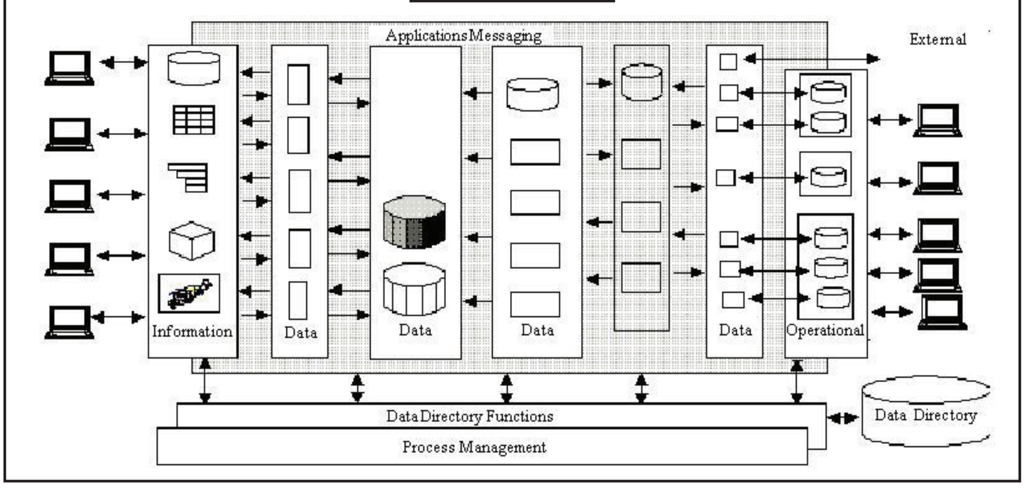
3. ***Operational Data Store:*** An ODS is an integrated database of operational data. Its sources include legacy systems, and it contains current or near-term data. An ODS may contain 30 to 60 days of information, while a data warehouse typically contains years of data. ODSs are used in some data warehouse architectures to provide near-real-time reporting capability in the event that the Data Warehouse's loading time or architecture prevents it from being able to provide near-real-time reporting capability.

1.5 Designing the Data Warehouse

Designing data warehouses is very different from designing traditional operational systems. For one thing, data warehouse users typically don't know nearly as much about their wants and needs as operational users. Second, designing a data warehouse often involves thinking in terms of much broader, and more difficult to define, business concepts than does designing an operational system. In this respect, data warehousing is quite close to Business Process Reengineering (BPR). Finally, the ideal design strategy for a data warehouse is often outside-in as opposed to top-down.

But while data warehouse design is different from what we have been used to, it is no less important. The fact that end-users have difficulty defining what they need as a bare minimum is no less necessary. In practice, data warehouse designers find that they have to use every trick in the book to help their users "visualize" their requirements. In this respect, robust working prototypes are essential.

Figure 1.2



The job of designing and implementing a data warehouse is very challenging and difficult one, even though at the same time, there is a lot of focus and importance attached to it. The designer of a data warehouse may be asked by the top management: "The all enterprise data and build a data warehouse such that the management can get answers to all their questions". This is a daunting task with responsibility being visible and exciting. But how to get started? Where to start? Which data should be put first? Where is that data available? Which queries should be answered? How would bring down the scope of the project to something smaller and manageable, yet be scalable to gradually upgrade to build a comprehensive data warehouse environment finally?

The recent trend is to build data marts before a real large data warehouse is built. People want something smaller, so as to get manageable results before proceeding to a real data warehouse.

Ralph Kimball identified a nine-step method as follows:

Step 1: Choose the subject matter (one at a time)

Step 2: Decide what the fact table represents

Step 3: Identify and conform the dimensions

Step 4: Choose the facts

Step 5: Store pre-calculations in the fact table

Step 6: Define the dimensions and tables

Step 7: Decide the duration of the database and periodicity of updation

Step 8: Track slowly the changing dimensions

Step 9: Decide the query priorities and the query modes.

All the above steps are required before the data warehouse is implemented. The final step or step 10 is to implement a simple data warehouse or a data mart. The approach should be 'from simpler to complex',

First, only a few data marts are identifies, designed and implemented. A data warehouse then will emerge gradually.

Let us discuss the above mentioned steps in detail. Interaction with the users is essential for obtaining answers to many of the above questions. The users to be interviewed include top management, middle management, executives as also operational users, in addition to sales-force and marketing teams. A clear picture emerges from the entire project on data warehousing as to what are their problems and how they can possibly be solved with the help of the data warehousing.

The priorities of the business issues can also be found. Similarly, interviewing the DBAs in the organization will also give a clear picture as what are the data sources with clean data, valid data and consistent data with assured flow for several years.

Task Discuss various factors play vital role to design a good data warehouse.

1.6 Data Warehouse Architecture

1.6.1 Why do Business Analysts need 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:

1. 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.

2. A data warehouse can enhance business productivity since it is able to quickly and efficiently gather information, which accurately describes the organization.
3. 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.
4. 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.
5. 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.
6. Because they are separate from operational systems, data warehouses provide retrieval of data without slowing down operational systems.

1.6.2 Process of Data Warehouse Design

A data warehouse can be built using three approaches:

1. A top-down approach
2. A bottom-up approach
3. 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 organisation to move forward at considerably less expense and to evaluate the benefits of the technology before making significant commitments.

In the combined approach, an organisation 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 organisational 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.

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 synchronisation, planning for disaster recovery, managing access control and security, managing data growth, managing database performance, and data warehouse enhancement and extension.

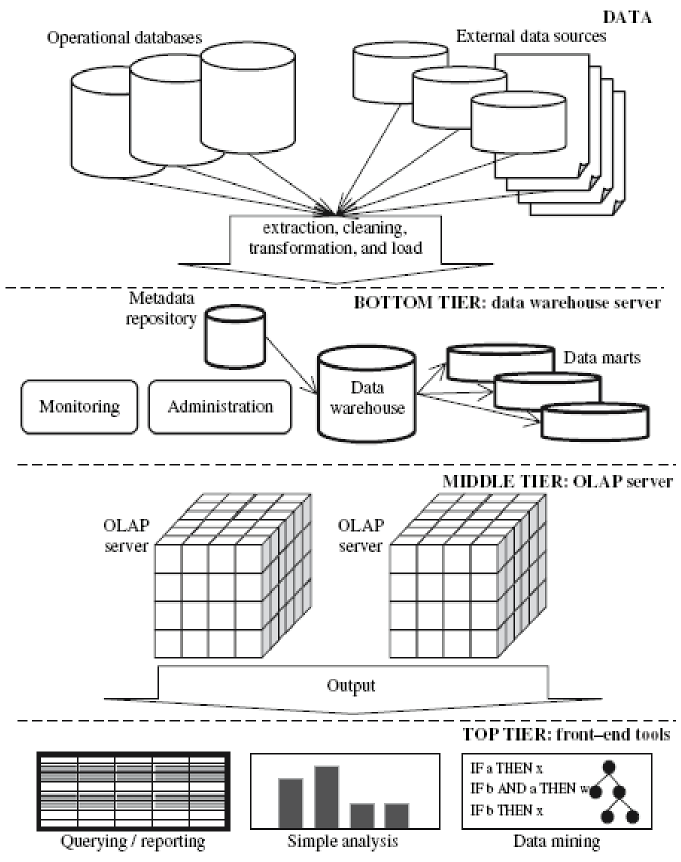
1.6.3 A Three-tier Data Warehouse Architecture

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

1. A bottom tier that consists of the Data Warehouse server, which is almost always a RDBMS. It may include several specialised data marts and a metadata repository,
2. 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.
3. 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 1.3.

Figure 1.3: A three-tier Data Warehousing Architecture



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 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.

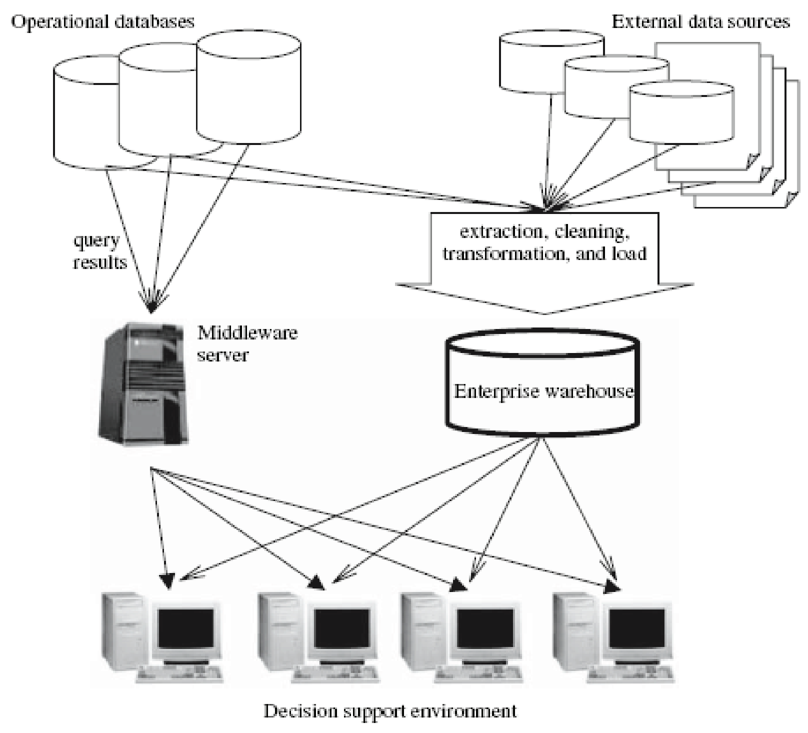
Depending on the source of data, data marts can be categorized into the following two classes:

1. 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.
2. Dependent data marts are sourced directly from enterprise data warehouses.

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.

The differences between the virtual and the enterprise DWs are shown in Figure 1.4. 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 1.4: A Virtual Data Warehouse and an Enterprise Data Warehouse



1.6.4 OLAP Server Architectures

We describe here the physical implementation of an OLAP server in a Data Warehouse. There are three different possible designs:

1. Relational OLAP (ROLAP)
2. Multidimensional OLAP (MOLAP)
3. Hybrid OLAP (HOLAP)

ROLAP

ROLAP stores the data based on the already familiar relational DBMS technology. In this case, data and the related aggregations are stored in RDBMS, and OLAP middleware is used to implement handling and exploration of data cubes. This architecture focuses on the optimisation of the RDBMS back end and provides additional tools and services such as data cube navigation logic. Due to the use of the RDBMS back end, the main advantage of ROLAP is scalability in handling large data volumes.

Example: ROLAP engines include the commercial IBM Informix Metacube (www.ibm.com) and the Micro-strategy DSS server (www.microstrategy.com), as well as the open-source product Mondrian (mondrian.sourceforge.net).

MOLAP

In contrast to ROLAP, which uses tuples as the data storage unit, the MOLAP uses a dedicated n-dimensional array storage engine and OLAP middleware to manage data. Therefore, OLAP queries are realised through a direct addressing to the related multidimensional views (data cubes). Additionally, this architecture focuses on pre-calculation of the transactional data into the aggregations, which results in fast query execution performance. More specifically, MOLAP precalculates and stores aggregated measures at every hierarchy level at load time, and stores and indexes these values for immediate retrieval. The full precalculation requires a substantial amount of overhead, both in processing time and in storage space. For sparse data, MOLAP uses sparse matrix compression algorithms to improve storage utilisation, and thus in general is characterised by smaller on-disk size of data in comparison with data stored in RDBMS.

Example: MOLAP products are the commercial Hyperion Ebasse (www.hyperion.com) and the Applix TM1 (www.applix.com), as well as Palo (www.opensourceolap.org), which is an open-source product.

HOLAP

To achieve a tradeoff between ROLAP's scalability and MOLAP's query performance, many commercial OLAP servers are based on the HOLAP approach. In this case, the user decides which portion of the data to store in the MOLAP and which in the ROLAP. For instance, often the low-level data are stored using a relational database, while higher-level data, such as aggregations, are stored in a separate MOLAP. An example product that supports all three architectures is Microsoft's OLAP Services (www.microsoft.com/), which is part of the company's SQL Server.

1.7 Getting Heterogeneous Data into the Warehouse

All data warehouses store their data grouped together by subject areas that reflect the general usage of the data (Customer, Product, Finance etc.). The general principle used in the majority of data warehouses is that data is stored at its most elemental level for use in reporting and information analysis.

Within this generic intent, there are two primary approaches to organising the data in a data warehouse.

The first is using a "dimensional" approach. In this style, information is stored as "facts" which are numeric or text data that capture specific data about a single transaction or event, and "dimensions" which contain reference information that allows each transaction or event to be classified in various ways. As an example, a sales transaction would be broken up into facts such as the number of products ordered, and the price paid, and dimensions such as date, customer, product, geographical location and salesperson. The main advantages of a dimensional approach is that the Data Warehouse is easy for business staff with limited information technology experience to understand and use. Also, because the data is pre-processed into the dimensional form, the Data Warehouse tends to operate very quickly. The main disadvantage of the dimensional approach is that it is quite difficult to add or change later if the company changes the way in which it does business.

The second approach uses database normalization. In this style, the data in the data warehouse is stored in third normal form. The main advantage of this approach is that it is quite straightforward to add new information into the database the primary disadvantage of this approach is that it can be rather slow to produce information and reports.

Task Database means to keep records in a particular format. What about data warehouse?

1.8 Getting Multidimensional Data out of the Warehouse

Data warehouse, generally speaking, is a kind of analytical tool used in business area. Since the amount of data of such a database is usually far beyond our imagination, and the inter-relation among those data are often intertwined, multidimensional data, with its multidimensional structure, are very popular for database as described.

A key feature of multidimensional data is that it can be viewed from different aspects, leading to a broader picture of the whole data and a clear view of the trend. Therefore, such kind of data is often used as part of OLAP to server users with a high-speed and effective query.

Getting multidimensional data out of the data warehouse are as follows:

1. Large data volumes, e.g., sales, telephone calls
   (a) Giga-, Tera-, Peta-, Exa-byte
2. OLAP = On-Line Analytical Processing
   (a) Interactive analysis
   (b) Explorative discovery
   (c) Fast response times required
3. OLAP operations
   (a) Aggregation of data
   (b) Standard aggregations operator, e.g., SUM
   (c) Starting level, (Quarter, City)

(d) Roll Up: Less detail, Quarter->Year

(e) Drill Down: More detail, Quarter->Month

(f) Slice/Dice: Selection, Year=1999

(g) Drill Across: "Join"

These are the various methods used for getting multidimensional data out of the data warehouse.

Case Study

Fast Food

The Fast Food industry is highly competitive, one where a very small change in operations can have a significant impact on the bottom line. For this reason, quick access to comprehensive information for both standard and on-demand reporting is essential.

Exclusive Ore designed and implemented a data warehouse and reporting structure to address this requirement for Summerwood Corporation, a fast food franchisee operating approximately 80 Taco Bell and Kentucky Fried Chicken restaurants in and around Philadelphia. The Summerwood Data Warehouse now provides strategic and tactical decision support to all levels of management within Summerwood.

The data warehouse is implemented in Microsoft SQL Server 2000, and incorporates data from two principal sources:

1. Daily sales information automatically polled by the TACO system DePol utility.
2. Period based accounting information from the Dynamics (Microsoft Great Plains) accounting database.

This data is automatically refreshed periodically (or on-demand if required) and is maintained historically over several years for comparative purposes.

For reporting and analysis purposes, the data in the warehouse is processed into OLAP Cubes. The cubes are accessed through Excel by using BusinessQuery MD. Data can be analyzed (sliced and diced) by store, by company, by zone and area, by accounting year, quarter and period (as far back as 1996), and by brand and concept. The available cubes and some example analyses are shown below. While each represents an area of analytical focus, cross cube analysis is also possible.

1. ***PL Cube.*** Contains Profit & Loss, Cash Flow and EBIDTA statements for Summerwood. Amounts can be viewed for any period as a period, quarter-to-date, year-to-date, or rolling 13 period amount, and can be compared to either of two budgets, compared to the corresponding period from the prior year, or as a percent of sales.
2. ***BS Cube.*** Contains the Balance Sheet for Summerwood. Balances can be viewed as of any period, and can be compared to the preceding period or the corresponding period in the prior year.
3. ***SalesMix Cube.*** Contains daily sales of all menu items in all stores. In addition to the standard analysis parameters, this data can also be sliced and diced by brand, by item category or by menu item, by calendar year, month and week, and by pricing tier. This cube can be used to compute sales amounts and counts, costs and variance from list price.

Contd...

Notes

4. ***SalesDayPart Cube.*** Contains sales amounts and counts at 15 minute intervals. In addition to the standard analysis parameters, the data in this cube can also be analyzed by calendar year, month and week, and by eight-hour, four-hour, two-hour, one-hour and 15 minute intervals, or by specific meal (e.g., lunch, dinner, breakfast, between-meals, etc.).
5. ***SalesOps Cube.*** Contains daily sales summary for each store. In addition to the standard analysis parameters, this data can also be sliced and diced by a comparable indicator, by calendar year, month and week, and by pricing tier. Gross sales, taxable sales, non-tax sales, manual over/under, deletions, labor, cash over/short, deposits and average check are available.

Many amounts can be viewed optionally as variances, as a percent of sales, or summarized as week-to-date, period-to-date, year-to-date, or rolling 52-week amountsReportCard Cube. Contains the daily report card amounts. Some of these are also in the SalesOps cube. In addition, the Report Card contains speed-of-service and peak -hour information.

The data structure implemented for Summerwood allows them to maintain several distinct organizational structures in order to properly represent each store in (1) the corporate structure, i.e. the subsidiary to which they belong, (2) the operations structure, i.e. the zone/area and (3) the concept structure, i.e., KFC, TB-PH (a Taco Bell – Pizza Hut combination restaurant), etc.

The Summerwood data warehouse and the resulting OLAP cubes permit investigation along any of these corporate hierarchies – i.e., by operating company, by zone or area, or by brand or concept. This permits comparisons between concepts, say, or of all stores within a concept. Similarly, it is easy to do area-toareacomparisons, or zone-to-zone comparisons, or to view the performance of all stores within an area.

The data warehouse also supports a time dimension based on the 13 period calendar under which Summerwood operates. This calendar has been built into the warehouse, permitting easy comparison of any period to the prior period or to the same period in a prior year. Instead of comparing at the period level, comparisons and trends can be done at quarterly or annual levels. Lower level examination is also possible, e.g., comparing week -to-week or even day-to-day.

The PL and BS cubes contain the full Profit and Loss and Balance Sheet statements for each period during the last five years (65 periods in all), down to the account level. This makes it easy for Summerwood to evaluate trends in any expense category, comparing store-to-store, period-to-period, zone-to-zone, or concept-to-concept.

The SalesOps and SalesMix cubes are updated overnight and contain up-to-the-minute (through yesterday) information captured by the cash registers (POS) in each store. This enables managers to evaluate and compare trends in speed of service, labor usage, over/under rings, employee food purchases, etc., by store, zone, area, concept, subsidiary, etc. Because sales and counts are recorded in 15-minute intervals, called day parts, managers can use this to find strange sales patterns, possibly suggestive of employee theft, during the midnight hours.

1.9 Summary

- Data warehousing is the consolidation of data from disparate data sources into a single target database to be utilized for analysis and reporting purposes.
- The primary goal of data warehousing is to analyze the data for business intelligence purposes. For example, an insurance company might create a data warehouse to capture policy data for catastrophe exposure.

- The data is sourced from front-end systems that capture the policy information into the data warehouse.
- The data might then be analyzed to identify windstorm exposures in coastal areas prone to hurricanes and determine whether the insurance company is overexposed.
- The goal is to utilize the existing information to make accurate business decisions.

1.10 Keywords

Data Sources: Data sources refer to any electronic repository of information that contains data of interest for management use or analytics.

Data Warehouse Architecture: It is a description of the elements and services of the warehouse, with details showing how the components will fit together and how the system will grow over time.

Data Warehouse: It is a relational database that is designed for query and analysis rather than for transaction processing.

Job Control: This includes job definition, job scheduling (time and event), monitoring, logging, exception handling, error handling, and notification.

Metadata: Metadata, or "data about data", is used not only to inform operators and users of the data warehouse about its status and the information held within the data warehouse

1.11 Self Assessment

Choose the appropriate answers:

1. OLTP stands for
   (a) On Line Transactional Processing
   (b) On Link Transactional Processing
   (c) On Line Transnational Processing
   (d) On Line Transactional Program
2. RDBMS stands for
   (a) Relational Database Manager System
   (b) Relational Database Management System
   (c) Relative Database Management System
   (d) Relational Design Management System
3. BPR stands for
   (a) Business Program Reengineering
   (b) Business Preview Reengineering
   (c) Basic Process Reengineering
   (d) Business Process Reengineering
4. ROLAP stands for
   (a) Relational On Line Transition Processing
   (b) Relative On Line Transactional Processing

(c) Relational On Line Transactional Processing

(d) Relational On Line Transactional Program

Fill in the blanks:

5. A .... warehouse is created based on a set of views defined for an operational RDBMS.
6. The .... contains a subset of the organisation-wide data that is of value to a small group of users.
7. .... stores the data based on the already familiar relational DBMS technology.
8. Data warehouse is a kind of analytical tool used in ....
9. A data warehouse focuses on the modeling and analysis of data for ....
10. The data warehouse is normally a ....
11. A .... is a physical database that receives all its information from the data warehouse.

1.12 Review Questions

1. What is a data warehouse? How is it better than traditional information-gathering techniques?
2. Describe the data warehouse environment.
3. List and explain the different layers in the data warehouse architecture.
4. What are metadata? Why are metadata so important to a data warehouse?
5. Describe operational components of data warehouse.
6. Explain why business analysts need data warehouse.
7. Describe the process of data warehouse design.
8. Differentiate between ROLAP and MOLAP.
9. "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process." Discuss.
10. What are the various uses of data warehouses in organizations?
11. Differences between operational database systems and data warehouses.
12. Describe three-tier data warehouse architecture in detail.

Answers: Self Assessment

1. (a)	2. (b)
3. (d)	4. (c)
5. virtual	6. data mart
7. ROLAP	8. business area
9. decision makers	10. relational database
11. dependent data mart	

1.13 Further Readings

Notes

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata McGraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 2: Data Mining Concept

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain data mining concept
- Describe architecture of data mining
- Know data mining functionalities
- Describe data mining system classifications

Introduction

This unit provides an introduction to the multidisciplinary field of data mining. It discusses the evolutionary path of database technology, which led up to the need for data mining, and the importance of its application potential. The basic architecture of data mining systems is described, and a brief introduction to the concepts of database systems and data warehouses is given. A detailed classification of data mining tasks is presented, based on the different kinds of knowledge to be mined. A classification of data mining systems is presented, and major challenges in the field are discussed.

With the increased and widespread use of technologies, interest in data mining has increased rapidly. Companies are now utilized data mining techniques to exam their database looking for trends, relationships, and outcomes to enhance their overall operations and discover new patterns that may allow them to better serve their customers. Data mining provides numerous benefits to businesses, government, society as well as individual persons. However, like many technologies, there are negative things that caused by data mining such as invasion of privacy right. In addition, the ethical and global issues regarding the use of data mining will also be discussed.

2.1 Motivation for Data Mining: Why is it Important?

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:

1. Data collection and database creation,
2. Data management (including data storage and retrieval, and database transaction processing), and
3. 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

Notes

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.

2.2 What is Data Mining?

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 cleaning*** (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 visualisation and knowledge representation techniques are used to present the mined knowledge to the user).

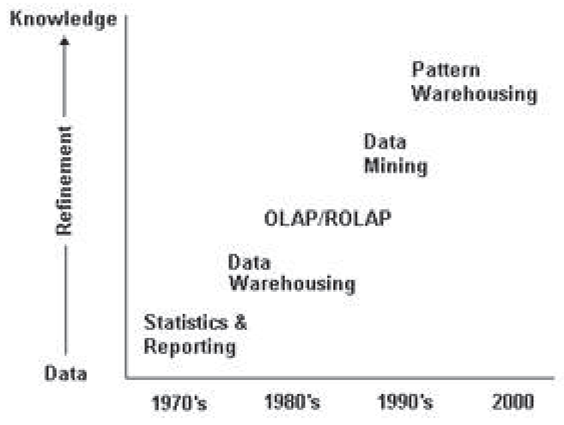
The first four steps are different forms of data preprocessing, which are used for data preparation for mining. After this the data-mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user and may be stored as new knowledge in the knowledge base.

2.3 Definition of Data Mining

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."

For many years, statistics have been used to analyze data in an effort to find correlations, patterns, and dependencies. However, with an increased in technology more and more data are available, which greatly exceed the human capacity to manually analyze them. Before the 1990's, data collected by bankers, credit card companies, department stores and so on have little used. But in recent years, as computational power increases, the idea of data mining has emerged. Data mining is a term used to describe the "process of discovering patterns and trends in large data sets in order to find useful decision-making information." With data mining, the information obtained from the bankers, credit card companies, and department stores can be put to good use

Figure 2.1: Data Mining Chart

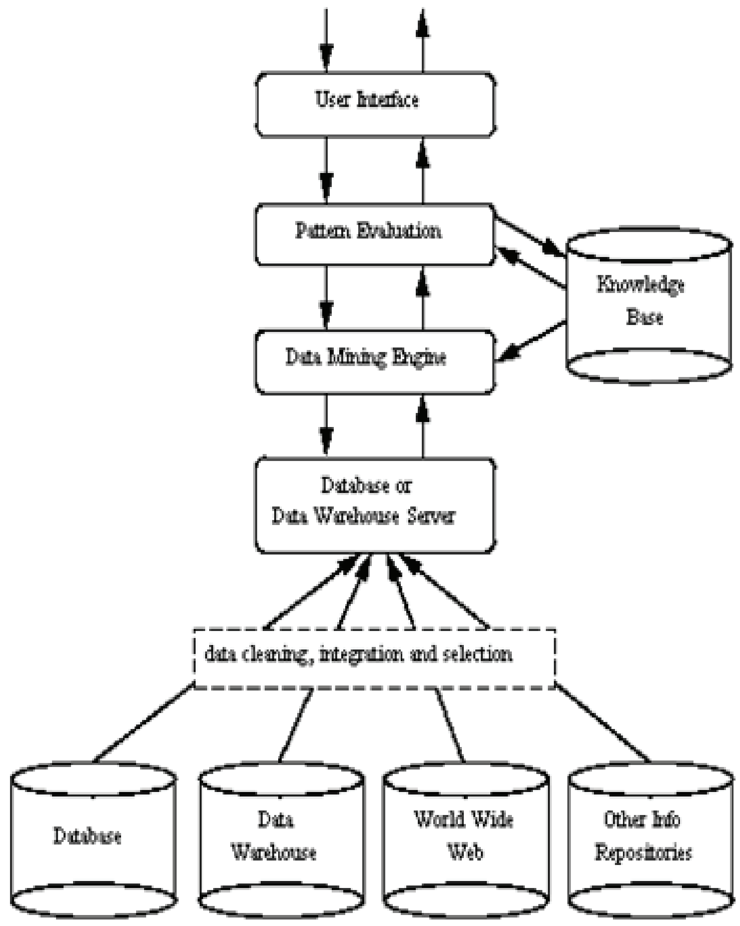


2.4 Architecture of Data Mining

Based on the above definition, the architecture of a typical data mining system may have the following major components (Figure 2.2):

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.
2. ***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.
3. ***Knowledge base:*** This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns.
4. ***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.
5. ***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 2.2: A Typical Architecture for Data Mining



6. ***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 visualise the patterns in different forms.

Note 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 visualisation, 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.

Task Data mining is a term used to describe the "process of discovering patterns and trends in large data sets in order to find useful decision-making information." Discuss.

2.5 How Data Mining Works?

Data mining is a component of a wider process called "knowledge discovery from database". It involves scientists and statisticians, as well as those working in other fields such as machine learning, artificial intelligence, information retrieval and pattern recognition.

Before a data set can be mined, it first has to be "cleaned". This cleaning process removes errors, ensures consistency and takes missing values into account. Next, computer algorithms are used to "mine" the clean data looking for unusual patterns. Finally, the patterns are interpreted to produce new knowledge.

How data mining can assist bankers in enhancing their businesses is illustrated in this example. Records include information such as age, sex, marital status, occupation, number of children, and etc. of the bank's customers over the years are used in the mining process. First, an algorithm is used to identify characteristics that distinguish customers who took out a particular kind of loan from those who did not. Eventually, it develops "rules" by which it can identify customers who are likely to be good candidates for such a loan. These rules are then used to identify such customers on the remainder of the database. Next, another algorithm is used to sort the database into cluster or groups of people with many similar attributes, with the hope that these might reveal interesting and unusual patterns. Finally, the patterns revealed by these clusters are then interpreted by the data miners, in collaboration with bank personnel.

2.6 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.

2.6.1 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.

2.6.2 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:

1. Mechanisms to create the definition of database structures:
2. Data storage

3. Concurrency control
4. Sharing of data
5. Distribution of data access
6. Ensuring data consistency
7. Security of the information stored, despite system crashes or attempts at unauthorised 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:

1. In RDBMS, tables can also be used to represent the relationships between or among multiple relation tables.
2. 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.
3. A given query is transformed into a set of relational operations, such as join, selection, and projection, and is then optimised for efficient processing.
4. 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.
5. 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.

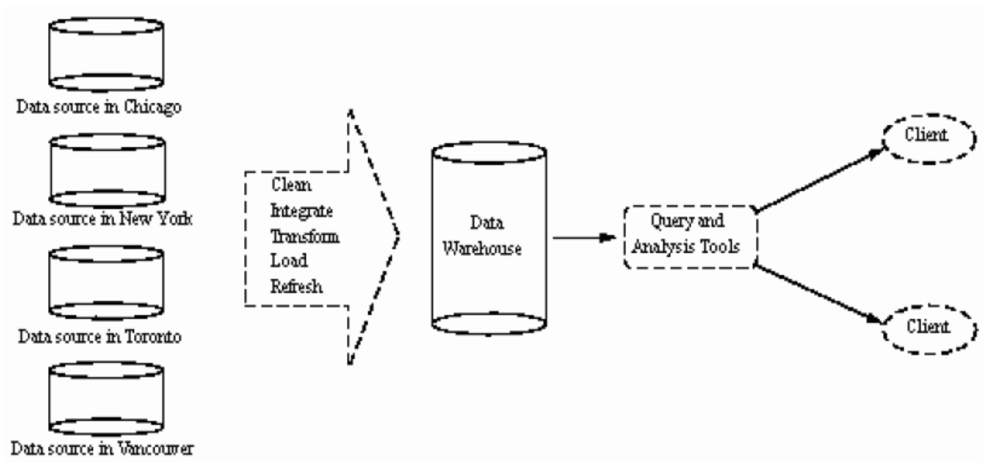
Example: Data mining systems can analyse customer data for a company to predict the credit risk of new customers based on their income, age, and previous credit information. Data mining systems may also detect deviations, such as items whose sales are far from those expected in comparison with the previous year.

2.6.3 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 2.3 shows the typical framework for construction and use of a data warehouse for a manufacturing company.

To facilitate decision making, the data in a data warehouse are organised 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 summarised. 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, summarised to a higher level, for each sales region.

Figure 2.3: Typical Framework of a Data Warehouse for a Manufacturing Company



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 summarised data.

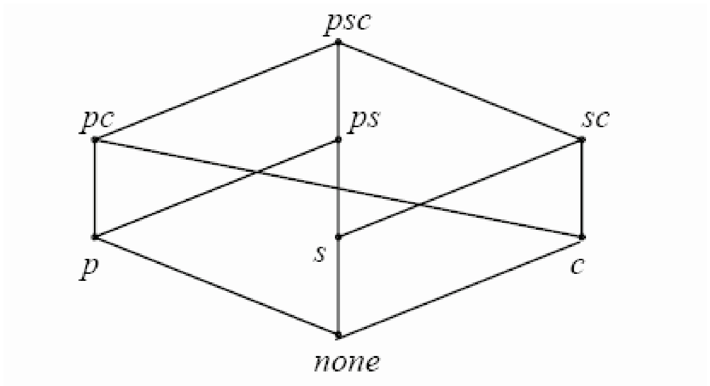
2.6.4 Data Cube

The data cube has a few alternative names or a few variants, such as, "multidimensional databases," "materialised views," and "OLAP (On-Line Analytical Processing)." The general idea of the approach is to materialise certain expensive computations that are frequently inquired, especially those involving aggregate functions, such as count, sum, average, max, etc., and to store such materialised 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.

Example: "Date" can be grouped into "day", "month", "quarter", "year" or "week", which forms a lattice structure.

Generalisation and specialisation 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 generalises 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.

Figure 2.4: Eight Views of Data Cubes for Sales Information



For example, a relation with the schema "sales(part; supplier; customer; sale price)" can be materialised into a set of eight views as shown in Figure 2.4, 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.

2.6.5 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 2.5, 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.

Figure 2.5: Fragment of a Transaction Database for the Rentals at Our Video Store

Rental				
Transaction ID	Date	Time	Customer ID	Item ID
T12345	10/12/06	10:40	C12345	11000

2.6.6 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:

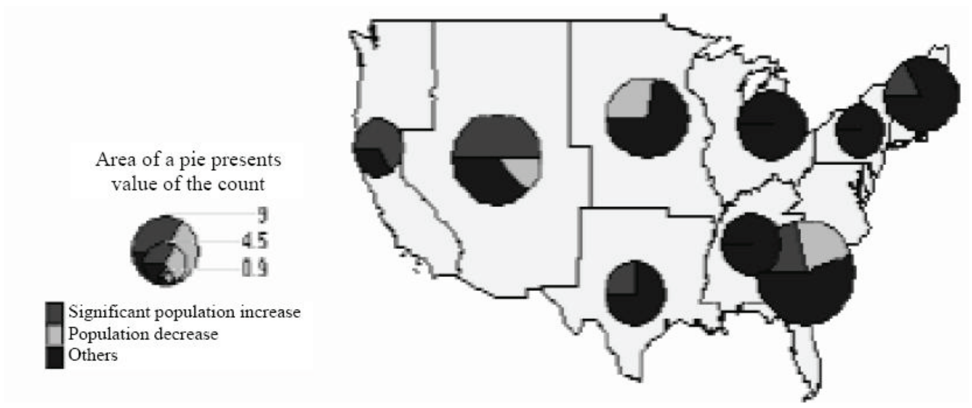
Multimedia Databases

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 characterised 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.

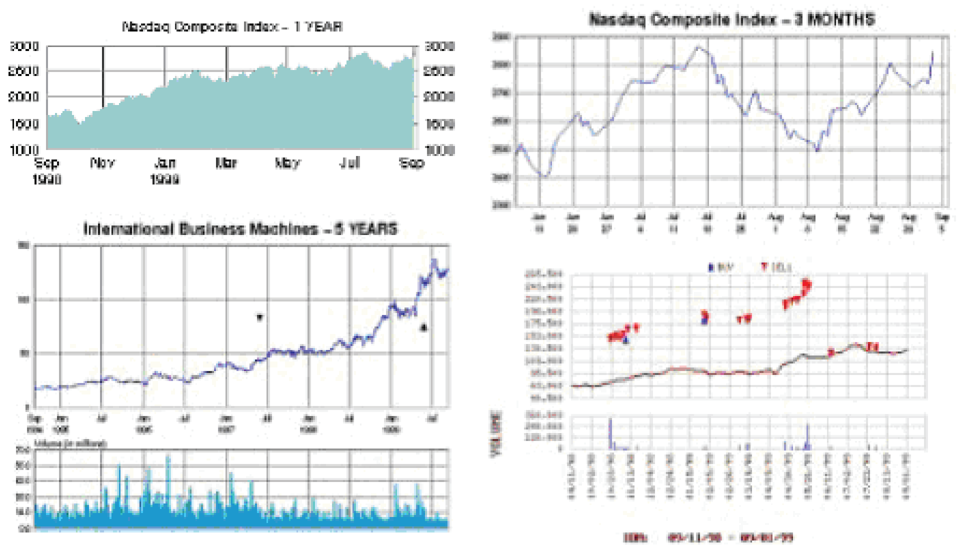
Figure 2.6: Visualisation of Spatial OLAP


Area of a pie presents value of the count
9
4.5
0.9
Significant population increase
Population decrease
Others

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 2.7: 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 organised 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 emphasise 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.

Task Discuss the purpose of ER diagram in database management system.

2.7 Data Mining Functionalities — What Kinds of Patterns can be Mined?

We have studied that the data mining can be performed on various types of data stores and database systems. On mining over the databases two kinds of patterns can be discovered depending upon the data mining tasks employed:

1. ***Descriptive data mining*** tasks that describe the general properties of the existing data. These include data characterisation and discrimination.

2. ***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:

Characterisation

Data characterisation is a summarisation 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 summarisation module to extract the essence of the data at different levels of abstractions.

Example: One may want to characterise the OurVideoStore customers who regularly rent more than 30 movies a year. With concept hierarchies on the attributes describing the target class, the attributeoriented induction method can be used, for example, to carry out data summarisation. Note that with a data cube containing summarisation of data, simple OLAP operations fit the purpose of data characterisation.

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.

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 characterisation 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.

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: $P \rightarrow Q$ [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 analyse 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 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 organisation 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 maximising the similarity between objects in a same class (intra-class similarity) and minimising the similarity between objects of different classes (inter-class similarity). Clustering can also facilitate taxonomy formation, that is, the organisation of observations into a hierarchy of classes that group similar events together.

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 characterising, 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?).

2.8 A Classification of Data Mining Systems

There are many data mining systems available or being developed. Some are specialised 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 categorised according to various criteria among other classification are the following:

1. ***Classification according to the kinds of data source mined:*** This classification categorises data mining systems according to the type of data handled such as spatial data, multimedia data, time-series data, text data, Worldwide Web, etc.
2. ***Classification according to the data model drawn on:*** This classification categorises data mining systems based on the data model involved such as relational database, object-oriented database, data warehouse, transactional, etc.
3. ***Classification according to the kind of knowledge discovered:*** This classification categorises data mining systems based on the kind of knowledge discovered or data mining functionalities, such as characterisation, discrimination, association, classification, clustering, etc. Some systems tend to be comprehensive systems offering several data mining functionalities together.
4. ***Classification according to mining techniques used:*** Data mining systems employ and provide different techniques. This classification categorises data mining systems according to the data analysis approach used such as machine learning, neural networks, genetic algorithms, statistics, visualisation, 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.

2.9 Advantages of Data Mining

Advantages of data mining from different point of view are:

Marketing/Retailing

Data mining can aid direct marketers by providing them with useful and accurate trends about their customers' purchasing behavior. Based on these trends, marketers can direct their marketing attentions to their customers with more precision. For example, marketers of a software company may advertise about their new software to consumers who have a lot of software purchasing history. In addition, data mining may also help marketers in predicting which products their customers may be interested in buying. Through this prediction, marketers can surprise their customers and make the customer's shopping experience becomes a pleasant one.

Retail stores can also benefit from data mining in similar ways. For example, through the trends provide by data mining, the store managers can arrange shelves, stock certain items, or provide a certain discount that will attract their customers.

Banking/Crediting

Data mining can assist financial institutions in areas such as credit reporting and loan information. For example, by examining previous customers with similar attributes, a bank can estimated the level of risk associated with each given loan. In addition, data mining can also assist credit card issuers in detecting potentially fraudulent credit card transaction. Although the data mining technique is not a 100% accurate in its prediction about fraudulent charges, it does help the credit card issuers reduce their losses.

Law Enforcement

Data mining can aid law enforcers in identifying criminal suspects as well as apprehending these criminals by examining trends in location, crime type, habit, and other patterns of behaviors.

Researchers

Data mining can assist researchers by speeding up their data analyzing process; thus, allowing them more time to work on other projects.

2.10 Disadvantages of Data Mining

Disadvantages of data mining are:

Privacy Issues

Personal privacy has always been a major concern in this country. In recent years, with the widespread use of Internet, the concerns about privacy have increase tremendously. Because of the privacy issues, some people do not shop on Internet. They are afraid that somebody may have access to their personal information and then use that information in an unethical way; thus causing them harm.

Although it is against the law to sell or trade personal information between different organizations, selling personal information have occurred. For example, according to Washing Post, in 1998, CVS had sold their patient's prescription purchases to a different company. In addition, American Express also sold their customers' credit care purchases to another company. What CVS and American Express did clearly violate privacy law because they were selling personal information without the consent of their customers. The selling of personal information may also bring harm to these customers because you do not know what the other companies are planning to do with the personal information that they have purchased.

Security Issues

Although companies have a lot of personal information about us available online, they do not have sufficient security systems in place to protect that information. For example, recently the Ford Motor credit company had to inform 13,000 of the consumers that their personal information including Social Security number, address, account number and payment history were accessed by hackers who broke into a database belonging to the Experian credit reporting agency. This incidence illustrated that companies are willing to disclose and share your personal information, but they are not taking care of the information properly. With so much personal information available, identity theft could become a real problem.

Misuse of Information/Inaccurate Information

Trends obtain through data mining intended to be used for marketing purpose or for some other ethical purposes, may be misused. Unethical businesses or people may used the information obtained through data mining to take advantage of vulnerable people or discriminated against a certain group of people. In addition, data mining technique is not a 100 percent accurate; thus mistakes do happen which can have serious consequence.

2.11 Ethical Issues of Data Mining

As with many technologies, both positives and negatives lie in the power of data mining. There are, of course, valid arguments to both sides. Here is the positive as well as the negative things about data mining from different perspectives.

2.11.1 Consumers' Point of View

According to the consumers, data mining benefits businesses more than it benefit them. Consumers may benefit from data mining by having companies customized their product and service to fit the consumers' individual needs. However, the consumers' privacy may be lost as a result of data mining.

Data mining is a major way that companies can invade the consumers' privacy. Consumers are surprised as how much companies know about their personal lives. For example, companies may know your name, address, birthday, and personal information about your family such as how many children you have. They may also know what medications you take, what kind of music you listen to, and what are your favorite books or movies. The lists go on and on. Consumers are afraid that these companies may misuse their information, or not having enough security to protect their personal information from unauthorized access. For example, the incidence about the hackers in the Ford Motor company case illustrated how insufficient companies are at protecting their customers' personal information. Companies are making profits from their customers' personal data, but they do not want to spend a lot amount of money to design a sophisticated security system to protect that data. At least half of Internet users interviewed by Statistical Research, Inc. claimed that they were very concerned about the misuse of credit care information given online, the selling or sharing of personal information by different web sites and cookies that track consumers' Internet activity.

Data mining that allows companies to identify their best customers could just be easily used by unscrupulous businesses to attack vulnerable customer such as the elderly, the poor, the sick, or the unsophisticated people. These unscrupulous businesses could use the information unethically by offering these vulnerable people inferior deals. For example, Mrs. Smith's husband was diagnosis with colon cancer, and the doctor predicted that he is going to die soon. Mrs. Smith was so worry and depressed. Suppose through Mrs. Smith's participation in a chat room or mailing list, someone predicts that either she or someone close to her has a terminal illness. Maybe through this prediction, Mrs. Smith started receiving email from some strangers stating that they know a cure for colon cancer, but it will cause her a lot of money. Mrs. Smith who is desperately wanted to save her husband, may fall into their trap. This hypothetical example illustrated that how unethical it is for somebody to use data obtained through data mining to target vulnerable person who are desperately hoping for a miracle.

Data mining can also be used to discriminate against a certain group of people in the population. For example, if through data mining, a certain group of people were determine to carry a high risk for a deathly disease (eg. HIV, cancer), then the insurance company may refuse to sell insurance policy to them based on this information. The insurance company's action is not only unethical, but may also have severe impact on our health care system as well as the individuals involved. If these high risk people cannot buy insurance, they may die sooner than expected because they cannot afford to go to the doctor as often as they should. In addition, the government may have to step in and provide insurance coverage for those people, thus would drive up the health care costs.

Data mining is not a flawless process, thus mistakes are bound to happen. For example, a file of one person may get mismatch to another person file. In today world, where we replied heavily on the computer for information, a mistake generated by the computer could have serious consequence. One may ask is it ethical for someone with a good credit history to get reject for a loan application because his/her credit history get mismatched with someone else bearing the same name and a bankruptcy profile? The answer is "NO" because this individual does not do anything wrong. However, it may take a while for this person to get his file straighten out. In the mean time, he or she just has to live with the mistake generated by the computer. Companies might say that this is an unfortunate mistake and move on, but to this individual this mistake can ruin his/her life.

2.11.2 Organizations' Point of View

Data mining is a dream comes true to businesses because data mining helps enhance their overall operations and discover new patterns that may allow companies to better serve their customers. Through data mining, financial and insurance companies are able to detect patterns of fraudulent credit care usage, identify behavior patterns of risk customers, and analyze claims. Data mining would help these companies minimize their risk and increase their profits. Since companies are able to minimize their risk, they may be able to charge the customers lower interest rate or lower premium. Companies are saying that data mining is beneficial to everyone because some of the benefit that they obtained through data mining will be passed on to the consumers.

Data mining also allows marketing companies to target their customers more effectively; thus, reducing their needs for mass advertisements. As a result, the companies can pass on their saving to the consumers. According to Michael Turner, an executive director of a Directing Marking Association "Detailed consumer information lets apparel retailers market their products to consumers with more precision. But if privacy rules impose restrictions and barriers to data collection, those limitations could increase the prices consumers pay when they buy from catalog or online apparel retailers by 3.5% to 11%".

When it comes to privacy issues, organizations are saying that they are doing everything they can to protect their customers' personal information. In addition, they only use consumer data for ethical purposes such as marketing, detecting credit card fraudulent, and etc. To ensure that personal information are used in an ethical way, the chief information officers (CIO) Magazine has put together a list of what they call the Six Commandments of Ethical Date Management. The six commandments include:

1. Data is a valuable corporate asset and should be managed as such, like cash, facilities or any other corporate asset;
2. The CIO is steward of corporate data and is responsible for managing it over its life cycle (from its generation to its appropriate destruction);
3. The CIO is responsible for controlling access to and use of data, as determined by governmental regulation and corporate policy;
4. The CIO is responsible for preventing inappropriate destruction of data;
5. The CIO is responsible for bringing technological knowledge to the development of data management practices and policies;
6. The CIO should partner with executive peers to develop and execute the organization's data management policies.

Since data mining is not a perfect process, mistakes such as mismatching information do occur. Companies and organizations are aware of this issue and try to deal it. According to Agrawal, a IBM's researcher, data obtained through mining is only associated with a 5 to 10 percent loss in accuracy. However, with continuous improvement in data mining techniques, the percent in inaccuracy will decrease significantly.

2.11.3 Government Point of View

The government is in dilemma when it comes to data mining practices. On one hand, the government wants to have access to people's personal data so that it can tighten the security system and protect the public from terrorists, but on the other hand, the government wants to protect the people's privacy right. The government recognizes the value of data mining to the society, thus wanting the businesses to use the consumers' personal information in an ethical way. According to the government, it is against the law for companies and organizations to trade data they had collected for money or data collected by another organization. In order to protect the people's privacy right, the government wants to create laws to monitor the data mining practices. However, it is extremely difficult to monitor such disparate resources as servers, databases, and web sites. In addition, Internet is global, thus creating tremendous difficulty for the government to enforce the laws.

2.11.4 Society's Point of View

Data mining can aid law enforcers in their process of identify criminal suspects and apprehend these criminals. Data mining can help reduce the amount of time and effort that these law enforcers have to spend on any one particular case. Thus, allowing them to deal with more problems. Hopefully, this would make the country becomes a safer place. In addition, data mining may also help reduce terrorist acts by allowing government officers to identify and locate potential terrorists early. Thus, preventing another incidence likes the World Trade Center tragedy from occurring on American soil.

Data mining can also benefit the society by allowing researchers to collect and analyze data more efficiently. For example, it took researchers more than a decade to complete the Human Genome Project. But with data mining, similar projects could be completed in a shorter amount of time. Data mining may be an important tool that aid researchers in their search for new medications, biological agents, or gene therapy that would cure deadly diseases such as cancers or AIDS.

2.12 Analysis of Ethical Issues

After looking at different views about data mining, one can see that data mining provides tremendous benefit to businesses, governments, and society. Data mining is also beneficial to individual persons, however, this benefit is minor compared to the benefits obtain for the companies. In addition, in order to gain this benefit, individual persons have to give up a lot of their privacy right.

If we choose to support data mining, then it would be unfair to the consumers because their right of privacy may be violated. Currently, business organizations do not have sufficient security systems to protect the personal information that they obtained through data mining from unauthorized access. Utilitarian, however, would supported this view because according to them, "An action is right from ethical point of view, if and only if, the sum total of utilities produced by that act is greater than the sum total of utilities produced by any other act the agent could have performed in its place." From the utilitarian view, data mining is a good thing because it enables corporations to minimize risk and increases profit; helps the government strengthen the security system; and benefit the society by speeding up the technological advancement. The only downside to data mining is the invasion of personal privacy and the risk of having people misuse the data. Based on this theory, since the majority of the party involves benefit from data mining, then the use of data mining is morally right.

If we choose to restrict data mining, then it would be unfair to the businesses and the government that use data mining in an ethical way. Restricting data mining would affect businesses' profits, national security, and may cause a delay in the discovery of new medications, biological agents, or gene therapy that would cure deadly diseases such as cancers or AIDS. Kant's categorical imperative, however, would supported this view because according to him "An action is morally right for a person if, and only if, in performing the action, the person does not use others merely as a means for advancing his or her own interests, but also both respects and develops their capacity to choose freely for themselves." From Kant's view, the use of data mining is unethical because the corporation and the government to some extent used data mining to advance their own interests without regard for people's privacy. With so many web sites being hack and private information being stolen, the benefit obtained through data mining is not good enough to justify the risk of privacy invasion.

As people collect and centralize data to a specific location, there always existed a chance that these data may be hacked by someone sooner or later. Businesses always promise they would treat the data with great care and guarantee the information will be save. But time after time, they have failed to keep their promise. So until companies able to develop better security system to safeguard consumer data against unauthorized access, the use of data mining should be restricted because there is no benefit that can outweigh the safety and wellbeing of any human being.

Task Do you think data mining provide help in marketing and research? Discuss with an example.

2.13 Global Issues of Data Mining

Since we are living in an Internet era, data mining will not only affect the US, instead it will have a global impact. Through Internet, a person living in Japan or Russia may have access to the personal information about someone living in California. In recent years, several major international hackers have break into US companies stealing hundred of credit card numbers. These hackers have used the information that they obtained through hacking for credit card fraud, black mailing purpose, or selling credit card information to other people. According to the

FBI, the majority of the hackers are from Russia and the Ukraine. Though, it is not surprised to find that the increase in fraudulent credit card usage in Russian and Ukraine is corresponded to the increase in domestic credit card theft.

After the hackers gained access to the consumer data, they usually notify the victim companies of the intrusion or theft, and either directly demanded the money or offer to patch the system for a certain amount of money. In some cases, when the companies refuse to make the payments or hire them to fix system, the hackers have release the credit card information that they previous obtained onto the Web. For example, a group of hackers in Russia had attacked and stolen about 55,000 credit card numbers from merchant card processor CreditCards.com. The hackers black mailed the company for $ 100,000, but the company refused to pay. As a result, the hackers posted almost half of the stolen credit card numbers onto the Web. The consumers whose card numbers were stolen incurred unauthorized charges from a Russian-based site. Similar problem also happened to CDUniverse.com in December 1999. In this case, a Russian teenager hacked into CDUniverse.com site and stolen about 300,000 credit card numbers. This teenager, like the group mentioned above also demanded $100,000 from CDUniverse.com. CDUniverse.com refused to pay and again their customers' credit card numbers were released onto the Web called the Maxus credit card pipeline.

Besides hacking into e-commerce companies to steal their data, some hackers just hack into a company for fun or just trying to show off their skills.

Example: A group of hacker called "BugBear" had hacked into a security consulting company's website. The hackers did not stole any data from this site, instead they leave a message like this "It was fun and easy to break into your box."

Besides the above cases, the FBI is saying that in 2001, about 40 companies located in 20 different states have already had their computer systems accessed by hackers. Since hackers can hack into the US e-commerce companies, then they can hack into any company worldwide. Hackers could have a tremendous impact on online businesses because they scared the consumers from purchasing online. Major hacking incidences liked the two mentioned above illustrated that the companies do not have sufficient security system to protect customer data. More efforts are needed from the companies as well as the government to tighten security against these hackers. Since the Internet is global, efforts from different governments worldwide are needed. Different countries need to join hand and work together to protect the privacy of their people.

Case Study

Data Warehousing and Data Mining using the SAS® System

As a management information systems department, our charge, simply stated, is to answer questions. We answer questions reactively when needed and proactively when possible. Being able to turn data into information, information into action and action into value is our reason for being. As technology changes, industry competition tightens, and our clients become increasingly computer savvy, our department must rapidly seek new technology and methods to meet the needs of our wide and varied list of clients.

Beyond the online mainframe systems, printouts, and electronic file attachments, we desire to provide internet based, intelligent, integrated systems that give our end users the best management information systems in the world. By using formal data warehousing practices, tools and methodologies, state of the art data extraction, transformation and

Contd...

Notes

summarization tools and thin client application deployment, we want to move beyond "data reporting" to "data mining."

According to the authors of Data Mining Techniques for Marketing, Sales and Customer Support, "to really achieve its promise, data mining needs to become an essential business process, incorporated into other processes, including marketing, sales, customer support, product design and inventory control. The 'virtuous cycle' incorporates data mining into the larger context of other business processes. It focuses on action based discovery and not the discovery mechanism itself."

To this end, MIS is developing a customized process to re-engineer existing MIS applications into a data warehousing environment where significant improvements and benefits for end users and the corporation can be realized. The process is founded in accepted data warehousing principles using an iterative rapid application development methodology, which is reusable across systems, functions and business solutions.

Data Warehousing

To successfully engage data mining in our processes, the first step is to know who our customers are. We are able to list them by name, job title, function, and business unit, and communicate with them regularly.

Next we must be able to identify the appropriate business opportunities. In MIS, our priorities are based on business needs as articulated to us by our clients through ad hoc requests and project management meetings and processes. Constant communication, integration and feedback are required to ensure we are investing our resources in proper ways.

Once having identified our customer base and business cases, we must be able to transform data into useful information. Transforming and presenting data as information is our primary function in the corporation. We are constantly looking for new and improved ways to accomplish this directive. The latest evolution in efficiently transforming and presenting data is formal data warehousing practices with browser based front ends.

Source data is crucial to data quality and mining efforts. As each new on-line transactional system and data base plat form is introduced the complexity of our tasks increases. "Using operational data presents many challenges to integrators and analysts such as bad data formats, confusing data fields, lack of functionality, legal ramifications, organizational factors, reluctance to change, and conflicting timelines (Berry, 25)." Also, the more disparate the input data sources, the more complicated the integration.

A clear definition of the business need is also required to ensure the accuracy of the end results. Defining a logical view of the data needed to supply the correct information, independent of source data restraints, is necessary. Here clients and analysts get the opportunity to discuss their business needs and solutions proactively.

Next, a mapping from the physical source data to the logical view is required and usually involves some compromises from the logical view due to physical data constraints. Then questions about presentation can begin to be answered. Who needs it? How often? In what format? What technology is available?

The first iteration of our SAS Data Warehousing solution accesses five operational systems existing on six data platforms. In addition to printed reports the users expect, the data warehouse is also accessible through MDDB OLAP technology over the intranet. Users can now ask and answer their own questions, enabling the creativity needed for successful data mining. With help from the SAS System, we are busily integrating additional data, accessing more data platforms and streamlining our processes.

2.14 Summary

- 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 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.
- Knowledge discovery as a process and consists of an iterative sequence of the data cleaning, data integration, data selection data transformation, data mining, pattern evaluation and knowledge presentation.

2.15 Keywords

Data cleaning: To remove noise and inconsistent data.

Data integration: Multiple data sources may be combined.

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

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.

KDD: Many people treat data mining as a synonym for another popularly used term.

Knowledge presentation: Visualisation and knowledge representation techniques are used to present the mined knowledge to the user.

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

2.16 Self Assessment

Choose the appropriate answers:

1. KDD stands for
   - (a) Knowledge Design Database
   - (b) Knowledge Discovery Database
   - (c) Knowledge Discovery Design
   - (d) Knowledge Design Development
2. DBMS stands for
   - (a) Database Management Spot
   - (b) Design Management System
   - (c) Database Management System
   - (d) Database Manager System

Fill in the blanks:

3. A .... is a collection of tables, each of which is assigned a unique name.
4. Data warehouses are constructed via a process of data cleaning, data integration, data transformation, data loading and ....
5. A .... is a set of records representing transactions, each with a time stamp, an identifier and a set of items.
6. .... databases include video, images, audio and text media.
7. Time-series databases contain time related data such ....
8. Data .... is a summarisation of general features of objects in a target class, and produces what is called characteristic rules.
9. .... is based on the association rules.
10. Clustering is also called .... classification.
11. .... is a term used to describe the "process of discovering patterns and trends in large data sets in order to find useful decision-making information."

2.17 Review Questions

1. What is data mining? How does data mining differ from traditional database access?
2. Discuss, in brief, the characterization of data mining algorithms.
3. Briefly explain the various tasks in data mining.
4. What is visualization? Discuss, in brief, the different visualization techniques.
5. Discuss the evolution of data mining as a confluence of disciplines.
6. What issues should be addressed by data mining algorithms and products? Why are they relevant?
7. Discuss the need for metrics in data mining.
8. "Data mining can often have far reaching social implications." Discuss this statement.
9. Discuss, in brief, important implementation issues in data mining.
10. Distinguish between the KDD process and data mining.
11. Discuss how database and OLTP systems are related to data mining.
12. Write a short note on the ER model. What advantage does it offer over a trivial DBMS?

Answers: Self Assessment

1. (b)	2. (c)
3. relational database	4. periodic data refreshing
5. transaction database	6. Multimedia
7. stock market data	8. characterization
9. Association analysis	10. unsupervised
11. Data mining	

2.18 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata McGraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 3: Data Mining Techniques

CONTENTS

Objectives

After studying this unit, you will be able to:

- Know data mining techniques
- Describe statistical perspectives on data mining
- Explain decision trees

Introduction

Data mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally were too time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

Most companies already collect and refine massive quantities of data. Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources, and can be integrated with new products and systems as they are brought on-line. When implemented on high performance client/server or parallel processing computers, data mining tools can analyze massive databases to deliver answers to questions such as, "Which clients are most likely to respond to my next promotional mailing, and why?"

3.1 Statistical Perspective on Data Mining

The information age has been matched by an explosion of data. This surfeit has been a result of modern, improved and, in many cases, automated methods for both data collection and storage. For instance, many stores tag their items with a product-specific bar code, which is scanned in when the corresponding item is bought. This automatically creates a gigantic repository of information on products and product combinations sold. Similar databases are also created by automated book-keeping, digital communication tools or by remote sensing satellites, and aided by the availability of affordable and effective storage mechanisms – magnetic tapes, data warehouses and so on. This has created a situation of plentiful data and the potential for new and deeper understanding of complex phenomena. The very size of these databases however means that any signal or pattern may be overshadowed by "noise".

Consider for instance the database created by the scanning of product bar codes at sales checkouts. Originally adopted for reasons of convenience, this now forms the basis for gigantic databases as large stores maintain records of products bought by customers in any transaction. Some businesses have gone further: by providing customers with an incentive to use a magnetic-striped frequent shopper card, they have created a database not just of product combinations but also time-sequenced information on such transactions. The goal behind collecting such data is the ability to answer questions such as "If potato chips and ketchup are purchased together, what is the item that is most likely to be also bought?", or "If shampoo is purchased, what is the most common item also bought in that same transaction?". Answers to such questions result in what are called association rules. Such rules can be used, for instance, in deciding on store layout or on promotions of certain brands of products by offering discounts on select combinations. Applications of association rules transcend sales transactions data indeed.

An oft-stated goal of data mining is the discovery of patterns and relationships among different variables in the database. This is no different from some of the goals of statistical inference: consider for instance, simple linear regression. Similarly, the pair-wise relationship between the products sold above can be nicely represented by means of an undirected weighted graph, with products as the nodes and weighted edges for the presence of the particular product pair in as many transactions as proportional to the weights. While undirected graphs provide a graphical display, directed a cyclic graphs are perhaps more interesting – they provide understanding of the phenomena driving the relationships between the variables. The nature of these relationships can be analyzed using classical and modern statistical tools such as regression, neural networks and so on.

Another aspect of knowledge discovery is supervised learning. Statistical tools such as discriminant analysis or classification trees often need to be refined for these problems. Some additional methods to be investigated here are k-nearest neighbor methods, bootstrap aggregation or bagging, and boosting which originally evolved in the machine learning literature, but whose statistical properties have been analyzed in recent years by statisticians. Boosting is particularly useful in the context of data streams – when we have rapid data flowing into the system and real-time classification rules are needed. Such capability is especially desirable in the context of financial data, to guard against credit card and calling card fraud, when transactions are streaming in from several sources and an automated split-second determination of fraudulent or genuine use has to be made, based on past experience.

Notes

Another important aspect of knowledge discovery is unsupervised learning or clustering, which is the categorization of the observations in a dataset into an a priori unknown number of groups, based on some characteristic of the observations. This is a very difficult problem, and is only compounded when the database is massive. Hierarchical clustering, probability based methods, as well as optimization partitioning algorithms are all difficult to apply here. Maitra (2001) develops, under restrictive Gaussian equal-dispersion assumptions, a multipass scheme which clusters an initial sample, filters out observations that can be reasonably classified by these clusters, and iterates the above procedure on the remainder. This method is scalable, which means that it can be used on datasets of any size.

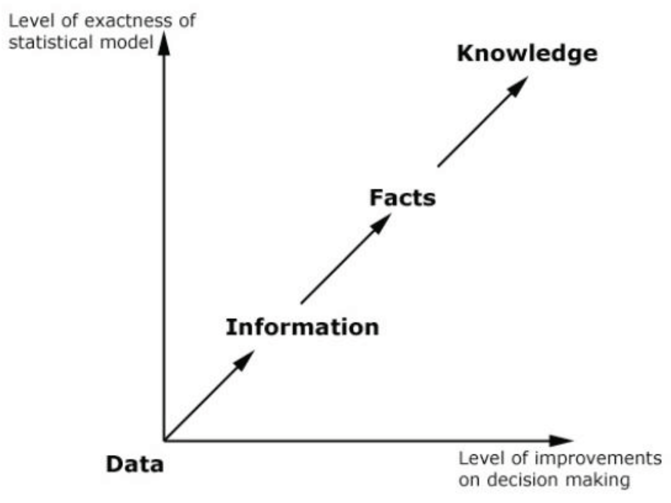
The field of data mining, like statistics, concerns itself with "learning from data" or "turning data into information".

3.2 What is Statistics and why is Statistics needed?

Statistics is the science of learning from data. It includes everything from planning for the collection of data and subsequent data management to end-of-the-line activities such as drawing inferences from numerical facts called data and presentation of results. Statistics is concerned with one of the most basic of human needs: the need to find out more about the world and how it operates in face of variation and uncertainty. Because of the increasing use of statistics, it has become very important to understand and practice statistical thinking. Or, in the words of H.G. Wells: "*Statistical thinking will one day be as necessary for efficient citizenship as the ability to read and write*".

But, why is statistics needed? Knowledge is what we know. Information is the communication of knowledge. Data are known to be crude information and not knowledge by themselves. The sequence from data to knowledge is as follows: from data to information (data become information when they become relevant to the decision problem); from information to facts (information becomes facts when the data can support it); and finally, from facts to knowledge (facts become knowledge when they are used in the successful competition of the decision process). Figure 3.1 illustrate this statistical thinking process based on data in constructing statistical models for decision making under uncertainties. That is why we need statistics. Statistics arose from the need to place knowledge on a systematic evidence base. This required a study of the laws of probability, the development of measures of data properties and relationships, and so on.

Figure 3.1: The Statistical Thinking Process based on Data in Constructing Statistical Models for Decision-making under Uncertainties


Level of exactness of statistical model
Knowledge
Facts
Information
Data
Level of improvements on decision making

3.3 Similarity Measures

Similarity measures provide the framework on which many data mining decision are based. Tasks such as classification and clustering usually assume the existence of some similarity measure, while fields with poor methods to compute similarity often find that searching data is a cumbersome task. Several classic similarity measures are discussed, and the application of similarity measures to other field are addressed.

3.3.1 Introduction

The goal of information retrieval (IR) systems is to meet users needs. In practical terms, a need is usually manifested in the form of a short textual query entered in the text box of some search engine online. IR systems typically do not directly answer a query, instead they present a ranked list of documents that are judged relevant to that query by some similarity measure. Sine similarity measures have the effect of clustering and classifying information with respect to a query, users will commonly find new interpretations of their information need that may or may not be useful to them when reformulating their query. In the case when the query is a document from the initial collection, similarity measures can be used to cluster and classify documents within a collection. In short, similarity measure can add a rudimentary structure to a previously unstructured collection.

3.3.2 Motivation

Similarity measures used in IR systems can distort one's perception of the entire data set. For example, if a user types a query into a search engine and does not find a satisfactory answer in the top ten returned web pages, then he/she will usually try to reformulate his/her query once or twice. If a satisfactory answer is still not returned, then the user will often assume that one does not exist. Rarely does a user understand or care what ranking scheme a particular search engine employs.

An understanding of the similarity measures, however, is crucial in today's business world. Many business decisions are often based on answers to questions that are posed in a way similar to how queries are given to search engines. Data miners do not have the luxury of assuming that the answers given to them from a database or IR system are correct or all-inclusive they must know the drawbacks of any similarity measure used and adjust their business decisions accordingly.

3.3.3 Classic Similarity Measures

A similarity measure is defined as a mapping from a pair of tuples of size k to a scalar number. By convention, all similarity measures should map to the range [-1, 1] or [0, 1], where a similarity score of 1 indicates maximum similarity. Similarity measure should exhibit the property that their value will increase as the number of common properties in the two items being compared increases.

A popular model in many IR applications is the vector-space model, where documents are represented by a vector of size n, where n is the size of the dictionary. Thus, document I is represented by a vector $d_i = (w_{1i},\ldots,w_{ki})$, where w_{ki} denotes the weight associated with term k in document i. in the simplest case, w_{ki} is the frequency of occurrence of term k in document i. Queries are formed by creating a pseudo-document vector q of size n, where w_{kq} is assumed to be non-zero if and only if term k occurs in the query.

Given two similarity scores sim $(q, d_i) = s_1$ and sim $(q, d_j) = s_2$, $s_1 > s_2$ means that document i is judged m ore relevant than document j to query q. since similarity measure are a pairwise measure, the values of s_1 and s_2 do not imply a relationship between documents i and j themselves.

Notes

From a set theoretic standpoint, assume that a universe Ω exists from which subsets A, B are generated. From the IR perspective, Ω is the dictionary while A and B are documents with A usually representing the query. Some similarity measures are more easily visualize via set theoretic notation.

As a simple measure, A∩B denotes the number of shared index terms. However, this simple coefficient takes no information about the sizes of A and B into account. The Simple coefficient is analogous to the binary weighting scheme in IR that can be thought of as the frequency of term co-occurrence with respect to two documents. Although the Simple coefficient is technically a similarity measure,

Most similarity measures are themselves evaluated by precision and recall, let A denote the set of retrieved documents and B denote the set of relevant documents. Define precision and recall as

$$P(A, B) = \frac{|A \cap B|}{|A|}$$

And

$$P(A, B) = \frac{|A \cap B|}{|A|}$$

respectively. Informally, precision is the ratio of returned relevance documents to the total number of documents returned, while recall is the ratio of returned relevant documents to the total number of relevant documents to the total number of relevant, documents. Precision is often evaluated at varying levels of recall (namely, I = 1, …., $|B|$) to produce a precision-recall graph. Ideally, IR systems generate high precision at all levels of recall. In practice, however, most systems exhibits lower precision values at higher levels of recall.

While the different notation styles may not yield exactly the same numeric values for each pair of items, the ordering of the items within a set is preserved.

3.3.4 Dice

The dice coefficient is a generalization of the harmonic mean of the precision and recall measures. A system with a high harmonic mean should theoretically by closer to an ideal retrieval system in that it can achieve high precision values at high levels of recall. The harmonic mean for precision and recall is given by

$$E = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

while the Dice coefficient is denoted by

$$\text{sim}(d, d_j) = D(A, B) = \frac{|A \cap B|}{\alpha |A| + (1 - \alpha)|B|}$$

$$\cong \frac{\alpha \sum_{k=1}^{n} w_{kq} w_{kj}}{\alpha \sum_{k=1}^{n} w_{kq}^2 + (1 - \alpha) \sum_{k=1}^{n} w_{kj}^2},$$

with α ε [0, 1]. To show that the Dice coefficient is a weighted harmonic mean, let α = ½.

3.3.5 Overlap

As its name implies, the Overlap coefficient attempts to determine the degree to which two sets overlap. The Overlap coefficient is compared as

$$\text{sim}(d, d_j) = D(A, B) = \frac{|A \cap B|}{\min(|A|, |B|)}$$

$$\cong \frac{\sum_{k=1}^{n} w_{kq} w_{kj}}{\min\left(\sum_{k=1}^{n} w_{kq}^2 + \sum_{k=1}^{n} w_{kj}^2\right)}$$

The Overlap coefficient is sometimes calculated using the max operator in place of the min.

Note The denominator does not necessarily normalize the similarity values produced by this measure. As a result, the Overlap values are typically higher in magnitude than other similarity measures.

Task "Statistics is mathematics but it's very useful in data mining." Discuss

3.4 Decision Trees

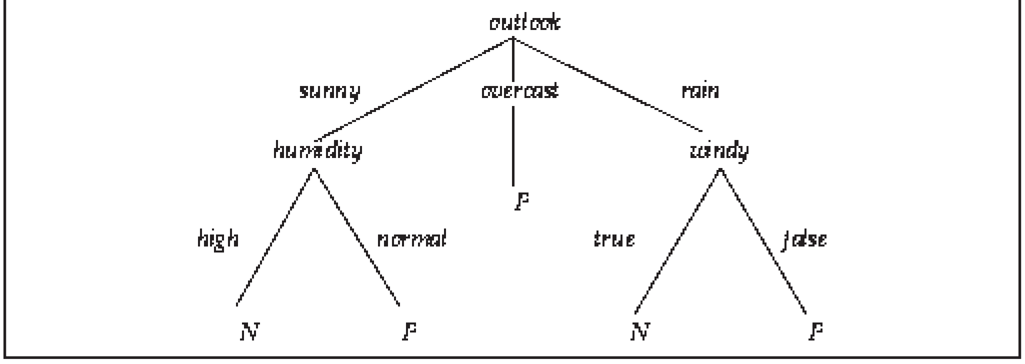
A decision tree is a structure that can be used to divide up a large collection of records into successively smaller sets of records by applying a sequence of simple decision rules. With each successive division, the members of the resulting sets become more and more similar to one another. The familiar division of living things into kingdoms, phyla, classes, orders, families, genera, and species, invented by the Dwedish botanist Carl Linnaeous in the 1730s, provides a good example. Within the animal kingdom, a particular animal is assigned to the phylum chordata if it has a spinal cord. Additional characteristics are used to further sub-divided the chordates into the birds, mammals, reptiles, and so on. These classes are further subdivided until, at the lowest level in the taxonomy, members of the same species are not only morphologically similar, they are capable of breeding and producing fertile offspring.

Decision trees are simple knowledge representation and they classify examples to a finite number of classes, the nodes are labelled with attribute names, the edges are labelled with possible values for this attribute and the leaves labelled with different classes. Objects are classified by following a path down the tree, by taking the edges, corresponding to the values of the attributes in an object.

The following is an example of objects that describe the weather at a given time. The objects contain information on the outlook, humidity etc. Some objects are positive examples denote by P and others are negative i.e. N. Classification is in this case the construction of a tree structure, illustrated in the following diagram, which can be used to classify all the objects correctly.

Notes

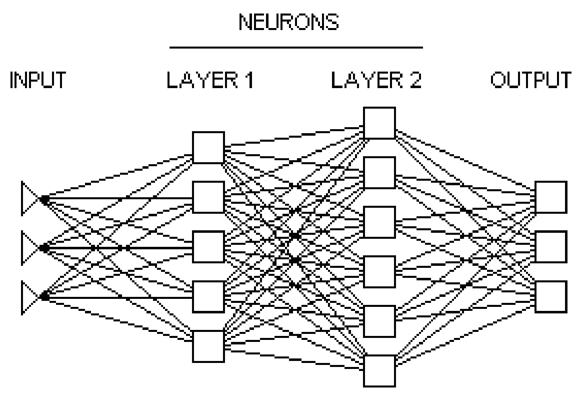
Figure 3.2: Decision Tree Structure



3.5 Neural Networks

Neural Networks are analytic techniques modeled after the (hypothesized) processes of learning in the cognitive system and the neurological functions of the brain and capable of predicting new observations (on specific variables) from other observations (on the same or other variables) after executing a process of so-called learning from existing data. Neural Networks is one of the Data Mining techniques.

Figure 3.3: Neural Network



The first step is to design a specific network architecture (that includes a specific number of "layers" each consisting of a certain number of "neurons"). The size and structure of the network needs to match the nature (e.g., the formal complexity) of the investigated phenomenon. Because the latter is obviously not known very well at this early stage, this task is not easy and often involves multiple "trials and errors."

The new network is then subjected to the process of "training." In that phase, neurons apply an iterative process to the number of inputs (variables) to adjust the weights of the network in order to optimally predict (in traditional terms one could say, find a "fit" to) the sample data on which the "training" is performed. After the phase of learning from an existing data set, the new network is ready and it can then be used to generate predictions.

Neural networks have seen an explosion of interest over the last few years, and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology and physics. Indeed, anywhere that there are problems of prediction, classification or control, neural networks are being introduced. This sweeping success can be attributed to a few key factors:

1. ***Power:*** Neural networks are very sophisticated modeling techniques capable of modeling extremely complex functions. In particular, neural networks are nonlinear (a term which is discussed in more detail later in this section). For many years linear modeling has been the commonly used technique in most modeling domains since linear models have well-known optimization strategies. Where the linear approximation was not valid (which was frequently the case) the models suffered accordingly. Neural networks also keep in check the curse of dimensionality problem that bedevils attempts to model nonlinear functions with large numbers of variables.

2. ***Ease of use:*** Neural networks learn by example. The neural network user gathers representative data, and then invokes training algorithms to automatically learn the structure of the data. Although the user does need to have some heuristic knowledge of how to select and prepare data, how to select an appropriate neural network, and how to interpret the results, the level of user knowledge needed to successfully apply neural networks is much lower than would be the case using (for example) some more traditional nonlinear statistical methods.

Neural networks are also intuitively appealing, based as they are on a crude low-level model of biological neural systems. In the future, the development of this neurobiological modeling may lead to genuinely intelligent computers.

Applications for Neural Networks

Neural networks are applicable in virtually every situation in which a relationship between the predictor variables (independents, inputs) and predicted variables (dependents, outputs) exists, even when that relationship is very complex and not easy to articulate in the usual terms of "correlations" or "differences between groups." A few representative examples of problems to which neural network analysis has been applied successfully are:

1. ***Detection of medical phenomena:*** A variety of health-related indices (e.g., a combination of heart rate, levels of various substances in the blood, respiration rate) can be monitored. The onset of a particular medical condition could be associated with a very complex (e.g., nonlinear and interactive) combination of changes on a subset of the variables being monitored. Neural networks have been used to recognize this predictive pattern so that the appropriate treatment can be prescribed.

2. ***Stock market prediction:*** Fluctuations of stock prices and stock indices are another example of a complex, multidimensional, but in some circumstances at least partially-deterministic phenomenon. Neural networks are being used by many technical analysts to make predictions about stock prices based upon a large number of factors such as past performance of other stocks and various economic indicators.

3. ***Credit assignment:*** A variety of pieces of information are usually known about an applicant for a loan. For instance, the applicant's age, education, occupation, and many other facts may be available. After training a neural network on historical data, neural network analysis can identify the most relevant characteristics and use those to classify applicants as good or bad credit risks.

4. ***Monitoring the condition of machinery:*** Neural networks can be instrumental in cutting costs by bringing additional expertise to scheduling the preventive maintenance of machines. A neural network can be trained to distinguish between the sounds a machine makes when it is running normally ("false alarms") versus when it is on the verge of a problem. After this training period, the expertise of the network can be used to warn a technician of an upcoming breakdown, before it occurs and causes costly unforeseen "downtime."

Notes

5. *Engine management:* Neural networks have been used to analyze the input of sensors from an engine. The neural network controls the various parameters within which the engine functions, in order to achieve a particular goal, such as minimizing fuel consumption.

3.6 Genetic Algorithms

Genetic algorithms are mathematical procedures utilizing the process of genetic inheritance. They have been usefully applied to a wide variety of analytic problems. Data mining can combine human understanding with automatic analysis of data to detect patterns or key relationships. Given a large database defined over a number of variables, the goal is to efficiently find the most interesting patterns in the database. Genetic algorithms have been applied to identify interesting patterns in some applications. They usually are used in data mining to improve the performance of other algorithms, one example being decision tree algorithms, another association rules.

Genetic algorithms require certain data structure. They operate on a population with characteristics expressed in categorical form. The analogy with genetics is that the population (genes) consist of characteristic. One way to implement genetic algorithms is to apply operators (reproduction, crossover, selection) with the feature of mutation enhance generation of potentially better combinations. The genetic algorithm process is thus:

1. Randomly select parents
2. Reproduce through crossover, Reproduction is the choosing which individual entities will survive. In other words, some objective function or selection characteristic is needed to determine survival. Crossover relates to change in future generations of entities.
3. Select survivors for the next generation through a fitness function.
4. Mutation is the operation by which randomly selected attributes of randomly selected entities in subsequent operations are changed.
5. Iterate until either a given fitness level is attained, or the present number of iteration is reached.

Genetic algorithm parameters include population size, crossover rate, and the mutation rate.

Advantages of Genetic Algorithm

Genetic algorithms are very easy to develop and to validate which makes them highly attractive of they apply. The algorithm is parallel, meaning that it can applied to large populations efficiently. The algorithm is also efficient in that if it begins with a poor original solution, it can rapidly progress to good solutions. Use of mutation makes the method capable of identifying global optima even in very nonlinear problem domains. The method does not require knowledge about the distribution of the data.

Disadvantages of Genetic Algorithms

Genetic algorithms require mapping data sets to a from where attributes have discrete values for the genetic algorithm to work with. This is usually possible, but can be lose a great deal of detail information when dealing with continuous variables. Coding the data into categorical from can unintentionally lead to biases in the data.

There are also limits to the size of data set that can be analyzed with genetic algorithms. For very large data sets, sampling will be necessary, which leads to different results across different runs over the same data set.

Task Discuss the use of neural networks in data mining.

3.7 Application of Genetic Algorithms in Data Mining

Genetic algorithms have been applies to data mining in two ways. External support is through evaluation or optimization of some parameter for another learning system, often hybrid systems using other data mining tools such as clustering or decision trees. In this sense, genetic algorithms help other data mining tools operate more efficiently. Genetic algorithms can also be directly applied to analysis, where the genetic algorithm generates descriptions, usually as decision rules or decision trees. Many applications of genetic algorithms within data mining have been applied outside of business. Specific examples include medical data mining and computer network intrusion detection. In business, genetic algorithms have been applied to customer segmentation, credit scoring, and financial security selection.

Genetic algorithms can be very useful within a data mining analysis dealing with more attributes and many more observations. It says the brute force checking of all combinations of variable values, which can make some data mining algorithms more effective. However, application of genetic algorithms requires expression of the data into discrete outcomes, with a calculate functional value which to base selection. This does not fit all data mining applications. Genetic algorithms are useful because sometimes if does fit.

Case Study

Business Reporting & Customer Information Datamart Architecture Setup & Roll-out for a Global Technology Company

To scope, define & design a structured & systemized architecture for capturing & reporting Business & Financial performance metrics for APJCC:

1. Evaluate current processes & methodology for capturing & reporting Business & financial performance metrics
2. Scope framework for systemized data capture & reporting of Business & financial performance metrics
3. Key Areas identified: (1) ABC (2) OPEX (3) Revenue (4) Profitability Analysis
4. Define processes, systems & tools for systemized data capture & reporting of Business & financial performance metrics
5. Design systems & tools for systemized data capture & reporting of Business & financial performance metrics

To scope, define & design framework for building a Customer Information Datamart Architecture for APJCC

1. Evaluate current processes & systems for capturing all customer contact information for APJCC
2. Scope framework for systemized data capture of customer intelligence data

Contd...

3. Scope for pilot: KL Hub (for actual data capture, UAT & roll-out), but framework must incorporate APJCC perspective
4. Define data definitions, DWH structure, data capture processes, business logics & system rules, applications & tools for Datamart.
5. Design & implement

3.8 Summary

- In this unit, you learnt about the data mining technique. Data Mining is an analytic process designed to explore data (usually large amounts of data - typically business or market related) in search of consistent patterns and/or systematic relationships between variables, and then to validate the findings by applying the detected patterns to new subsets of data.
- The ultimate goal of data mining is prediction - and predictive data mining is the most common type of data mining and one that has the most direct business applications.
- The process of data mining consists of three stages: (1) the initial exploration, (2) model building or pattern identification with validation/verification, and (3) deployment (i.e., the application of the model to new data in order to generate predictions).
- In this unit you also learnt about a statistical perspective of data mining, similarity measures, decision tree and many more.

3.9 Keywords

Decision Tree: A decision tree is a structure that can be used to divide up a large collection of records into successively smaller sets of records by applying a sequence of simple decision rules.

Dice: The dice coefficient is a generalization of the harmonic mean of the precision and recall measures.

Genetic Algorithms: Genetic algorithms are mathematical procedures utilizing the process of genetic inheritance.

Similarity Measures: Similarity measures provide the framework on which many data mining decision are based.

3.10 Self Assessment

Fill in the blanks:

1. .... is the science of learning from data.
2. .... are known to be crude information and not knowledge by themselves.
3. .... provide the framework on which many data mining decision are based.
4. The goal of .... systems is to meet user needs.
5. The .... is sometimes calculated using the max operator in place of the min.
6. .... are very sophisticated modeling techniques capable of modeling extremely complex functions.
7. .... are mathematical procedures utilizing the process of genetic inheritance.

State whether the following statements are true or false:

8. Data are known to be crude information and not knowledge by themselves.
9. Neural Networks is not a type of Data Mining techniques.
10. A variety of pieces of information are usually known about an applicant for a loan.
11. Genetic algorithms require certain data structure.

3.11 Review Questions

1. Write a short note on regression and correlation.
2. Define similarity. List the commonly used similarity measures.
3. What is meant by distance or dissimilarity measures? Elaborate.
4. Define a decision tree and a decision tree model.
5. Present and discuss a prediction technique using decision trees.
6. Write short notes on the following terms:
   (a) Neural network
   (b) Neural network model
   (c) Artificial neural network
   (d) Activation function
7. What is an activation function? Explain the different types of activation functions.
8. Write a short note on genetic algorithms.
9. Define a genetic algorithm. Present and explain a genetic algorithm.
10. Discuss the merits and demerits of genetic algorithms.
11. What do you mean by genetic algorithms?
12. Describe statistical perspective on data mining.
13. What do you mean by statistics and why it's so important?
14. Explain similarity measures in detail.
15. Describe various applications of genetic algorithms in data mining.

Answers: Self Assessment

1.	Statistics	2.	Data
3.	Similarity measures	4.	information retrieval (IR)
5.	Overlap coefficient	6.	Neural networks
7.	Genetic algorithms	8.	True
9.	False	10.	True
11.	True		

Notes

3.12 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 4: Data Mining Classification

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe the concept of data mining classification
- Discuss basic knowledge of different classification techniques
- Explain rule based algorithms

Introduction

Classification is a data mining (machine learning) technique used to predict group membership for data instances.

Example: You may wish to use classification to predict whether the weather on a particular day will be "sunny", "rainy" or "cloudy". Popular classification techniques include decision trees and neural networks.

Data classification is a two step process. In the first step, a model is built describing a predetermined set of data classes or concepts. The model is constructed by analyzing database tuples described by attributes. Each tuple is assumed to belong to a predefined class, as determined by one of the attributes, called the class label attribute. In the context of classification, data tuples are also referred to as samples, examples, or objects. The data tuples analyzed to build the model collectively form the training data set. The individual tuples making up the training set are referred to as training samples and are randomly selected from the sample population.

Since the class label of each training sample is provided, this step is also known as supervised learning (i.e., the learning of the model is 'supervised' in that it is told to which class each training sample belongs). It contrasts with unsupervised learning (or clustering), in which the class labels of the training samples are not known, and the number or set of classes to be learned may not be known in advance.

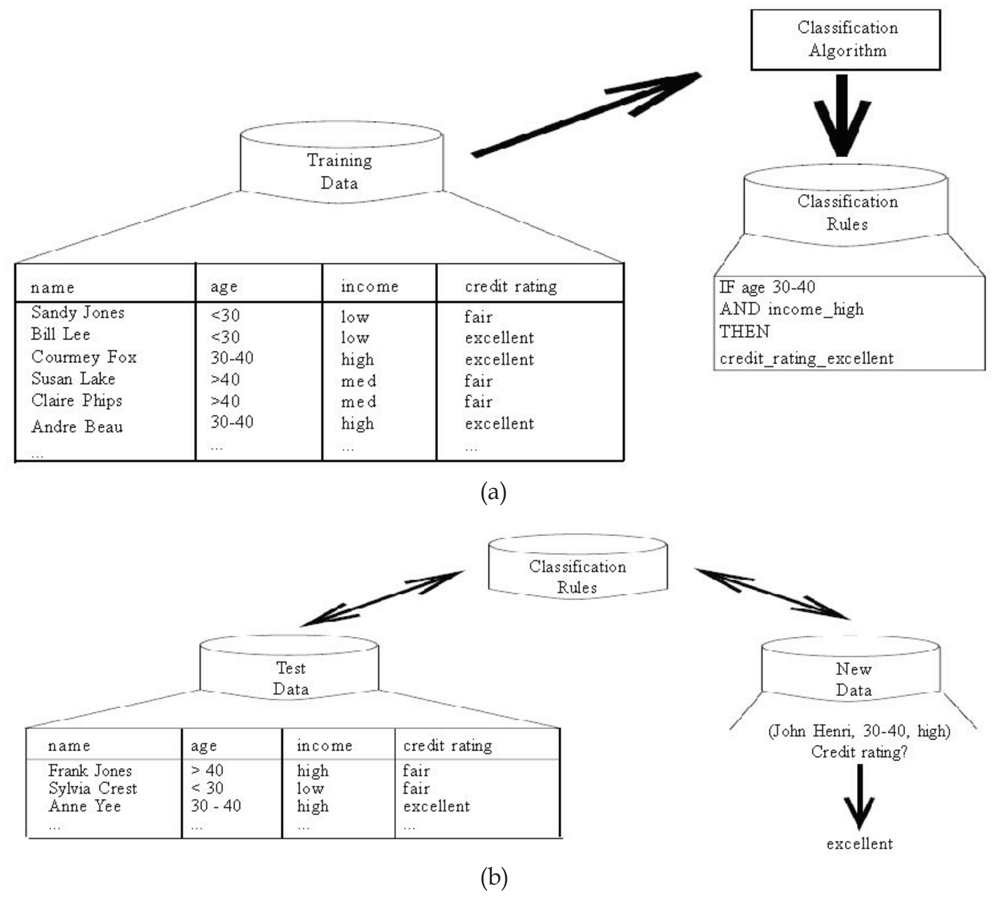
Typically, the learned model is represented in the form of classification rules, decision trees, or mathematical formulae. For example, given a database of customer credit information, classification rules can be learned to identify customers as having either excellent or fair credit ratings (Figure 4.1a). The rules can be used to categorize future data samples, as well as provide a better understanding of the database contents. In the second step (Figure 4.1b), the model is used for classification. First, the predictive accuracy of the model (or classifier) is estimated.

The holdout method is a simple technique which uses a test set of class-labeled samples. These samples are randomly selected and are independent of the training samples. The accuracy of a model on a given test set is the percentage of test set samples that are correctly classified by the model. For each test sample, the known class label is compared with the learned model's class prediction for that sample. Note that if the accuracy of the model were estimated based on the training data set, this estimate could be optimistic since the learned model tends to over fit the data (that is, it may have incorporated some particular anomalies of the training data which are not present in the overall sample population). Therefore, a test set is used.

(a) ***Learning:*** Training data are analyzed by a classification algorithm. Here, the class label attribute is credit rating, and the learned model or classifier is represented in the form of classification rule.

(b) ***Classification:*** Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

Figure 4.1: The Data Classification Process



name	age	income	credit rating
Sandy Jones	<30	low	fair
Bill Lee	<30	low	excellent
Courmey Fox	30-40	high	excellent
Susan Lake	>40	med	fair
Claire Phips	>40	med	fair
Andre Beau	30-40	high	excellent
...	...	...	...

IF age 30-40
AND income_high
THEN
credit_rating_excellent

name	age	income	credit rating
Frank Jones	> 40	high	fair
Sylvia Crest	< 30	low	fair
Anne Yee	30 - 40	high	excellent
...	...	...	...

4.1 What is Classification and Prediction?

4.1.1 Classification

Classification is a data mining technique used to predict group membership for data instances.

Example: You may wish to use classification to predict whether the weather on a particular day will be "sunny", "rainy" or "cloudy". Popular classification techniques include decision trees and neural networks.

In classification, there is a target categorical variable, such as income bracket, which, for example, could be partitioned into three classes or categories: high income, middle income, and low income. The data mining model examines a large set of records, each record containing information on the target variable as well as a set of input or predictor variables. For example, consider the excerpt from a data set shown in Table 4.1.

Notes

Table 4.1: Excerpt from Data Set for Classifying Income

Subject	Age	Gender	Occupation	Income Bracket
001	47	F	Software engineer	High
002	28	M	Marketing consultant	Middle
003	35	M	Unemployed	Low

Suppose that the researcher would like to be able to classify the income brackets of persons not currently in the database, based on other characteristics associated with that person, such as age, gender, and occupation. This task is a classification task, very nicely suited to data mining methods and techniques. The algorithm would proceed roughly as follows. First, examine the data set containing both the predictor variables and the (already classified) target variable, income bracket. In this way, the algorithm (software) "learns about" which combinations of variables are associated with which income brackets. For example, older females may be associated with the high-income bracket. This data set is called the training set. Then the algorithm would look at new records, for which no information about income bracket is available. Based on the classifications in the training set, the algorithm would assign classifications to the new records. For example, a 63-year-old female professor might be classified in the high-income bracket.

Examples of classification tasks in business and research include:

1. Determining whether a particular credit card transaction is fraudulent
2. Placing a new student into a particular track with regard to special needs
3. Assessing whether a mortgage application is a good or bad credit risk
4. Diagnosing whether a particular disease is present
5. Determining whether a will was written by the actual deceased, or fraudulently by someone else
6. Identifying whether or not certain financial or personal behavior indicates a possible terrorist threat

Supervised Learning and Unsupervised Learning

The learning of the model is 'supervised' if it is told to which class each training sample belongs. In contrasts with unsupervised learning (or clustering), in which the class labels of the training samples are not known, and the number or set of classes to be learned may not be known in advance.

Typically, the learned model is represented in the form of classification rules, decision trees, or mathematical formulae.

4.1.2 Prediction

Prediction is similar to classification, except that for prediction, the results lie in the future. Examples of prediction tasks in business and research include:

1. Predicting the price of a stock three months into the future
2. Predicting the percentage increase in traffic deaths next year if the speed limit is increased
3. Predicting the winner of this fall's baseball World Series, based on a comparison of team statistics

4. Predicting whether a particular molecule in drug discovery will lead to a profitable new drug for a pharmaceutical company.

Any of the methods and techniques used for classification may also be used, under appropriate circumstances, for prediction. These include the traditional statistical methods of point estimation and confidence interval estimations, simple linear regression and correlation, and multiple regression.

4.2 Issues regarding Classification and Prediction

To prepare the data for classification and prediction, following preprocessing steps may be applied to the data in order to help improve the accuracy, efficiency, and scalability of the classification or prediction process.

1. ***Data cleaning:*** This refers to the preprocessing of data in order to remove or reduce noise (by applying smoothing techniques, for example), and the treatment of missing values (e.g., by replacing a missing value with the most commonly occurring value for that attribute, or with the most probable value based on statistics). Although most classification algorithms have some mechanisms for handling noisy or missing data, this step can help reduce confusion during learning.

2. ***Relevance analysis:*** Many of the attributes in the data may be irrelevant to the classification or prediction task. For example, data recording the day of the week on which a bank loan application was filled is unlikely to be relevant to the success of the application. Furthermore, other attributes may be redundant. Hence, relevance analysis may be performed on the data with the aim of removing any irrelevant or redundant attributes from the learning process. In machine learning, this step is known as feature selection. Including such attributes may otherwise slow down, and possibly mislead, the learning step. Ideally, the time spent on relevance analysis, when added to the time spent on learning from the resulting "reduced" feature subset, should be less than the time that would have been spent on learning from the original set of features. Hence, such analysis can help improve classification efficiency and scalability.

3. ***Data transformation:*** The data can be generalized to higher-level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous-valued attributes. For example, numeric values for the attribute income may be generalized to discrete ranges such as low, medium, and high. Similarly, nominal-valued attributes, like street, can be generalized to higher-level concepts, like city. Since generalization compresses the original training data, fewer input/output operations may be involved during learning. The data may also be normalized, particularly when neural networks or methods involving distance measurements are used in the learning step. Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1.0 to 1.0, or 0 to 1.0. In methods which use distance measurements, for example, this would prevent attributes with initially large ranges (like, say income) from outweighing attributes with initially smaller ranges (such as binary attributes).

4.3 Statistical based Algorithms

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given sample belongs to a particular class.

Bayesian classification is based on Bayes theorem. Bayesian classifiers exhibited high accuracy and speed when applied to large databases.

Studies comparing classification algorithms have found a simple Bayesian classifier known as the naive Bayesian classifier to be comparable in performance with decision tree and neural network classifiers. Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved, and in this sense, is considered "naive". Bayesian belief networks are graphical models, which unlike naive Bayesian classifiers, allow the representation of dependencies among subsets of attributes.

Apply Bayes Rule: *c* is the class, {*v*} observed attribute values:

$$P(c\,|\,\{v\}) = \frac{P(\{v\}\,|\,c)P(c)}{P(\{v\})}$$

If we assume *k* possible disjoint diagnoses, $c_1, \ldots, c_K$

$$P(c_k\,|\,\{v\}) = \frac{P(\{c_k\}P(\{v\}\,|\,c_k)}{P(\{v\})}$$

P({v}) may not be known, but total probability of diagnoses is 1

P({v}) (the evidence): $\sum_k \frac{P(\{c_k\}P(\{v\}\,|\,c_k)}{P(\{v\})} = 1$

$\Rightarrow$ P({v}) = Σ_k | P(c_k) P({v} | c_k |

Need to know P(c_k), P({v} | c_k) for all k

Bayes Rules: $\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$

MAP vs. ML

Rather than computing full posterior, can simplify computation if interested in classification:

1. ML (Maximum Likelihood) Hypothesis

   assume all hypotheses equiprobable a priori – simply maximize data likelihood:

$$c_{\text{ML}} = \arg\max_{c \in C} P(\{v\}\,|\,c)$$

2. MAP (Maximum A Posteriori) Class Hypothesis

$$c_{\text{MAP}} = \arg\max_{c \in C} P(c\,|\,\{v\})$$

$$= \arg\max_{c \in C} \frac{P(\{v\}c\,|\,P(c))}{P(\{v\})}$$

   can ignore denominator because same for all *c*

Bayes Theorem

Bayes' theorem relates the conditional and marginal probabilities of events A and B, where B has a non-vanishing probability:

$$P(A\,|\,B) = \frac{P(B\,|\,A)P(A)}{P(B)}.$$

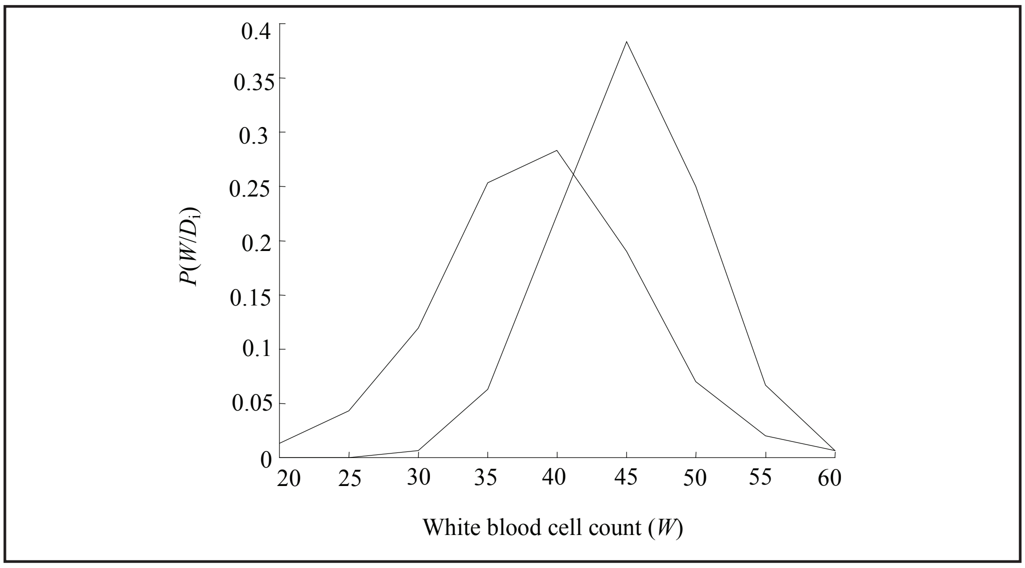
Each term in Bayes' theorem has a conventional name:

1. P(A) is the prior probability or marginal probability of A. It is "prior" in the sense that it does not take into account any information about B.
2. P(A | B) is the conditional probability of A, given B. It is also called the posterior probability because it is derived from or depends upon the specified value of B.
3. P(B | A) is the conditional probability of B given A.
4. P(B) is the prior or marginal probability of B, and acts as a normalizing constant.

Intuitively, Bayes' theorem in this form describes the way in which one's beliefs about observing 'A' are updated by having observed 'B'.

Bayes Theorem: Example

Use training examples to estimate class-conditional probability density functions for white-blood cell count (*W*)



Could use these to select maximum likelihood hypothesis.

4.4 Naive Bayesian Classification

Suppose your data consist of fruits, described by their color and shape. Bayesian classifiers operate by saying "If you see a fruit that is red and round, which type of fruit is it most likely to be, based on the observed data sample? In future, classify red and round fruit as that type of fruit."

A difficulty arises when you have more than a few variables and classes - you would require an enormous number of observations (records) to estimate these probabilities.

Naive Bayes classification gets around this problem by not requiring that you have lots of observations for each possible combination of the variables. Rather, the variables are assumed to be independent of one another and, therefore the probability that a fruit that is red, round, firm, 3" in diameter, etc. will be an apple can be calculated from the independent probabilities that a fruit is red, that it is round, that it is firm, that it is 3" in diameter, etc.

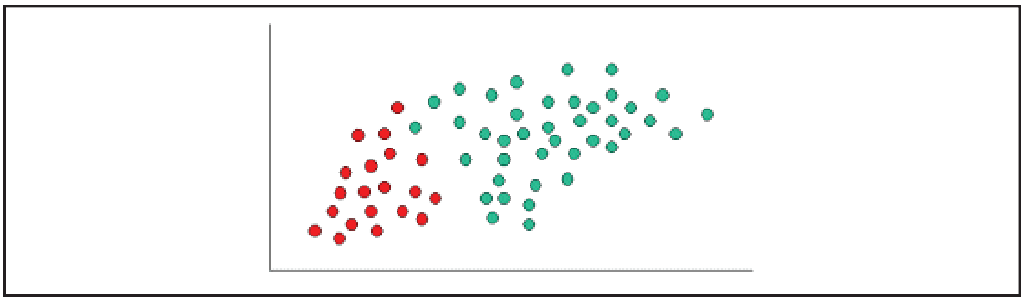
Notes

In other words, Naïve Bayes classifiers assume that the effect of a variable value on a given class is independent of the values of other variable. This assumption is called class conditional independence. It is made to simplify the computation and in this sense considered to be "Naïve".

This assumption is a fairly strong assumption and is often not applicable. However, bias in estimating probabilities often may not make a difference in practice - it is the order of the probabilities, not their exact values, that determine the classifications.

Naive Bayes Classifier

The Naive Bayes Classifier technique is based on the so-called Bayesian theorem and is particularly suited when the dimensionality of the inputs is high. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.



To demonstrate the concept of Naïve Bayes Classification, consider the example displayed in the illustration above. As indicated, the objects can be classified as either GREEN or RED. Our task is to classify new cases as they arrive, i.e., decide to which class label they belong, based on the currently exiting objects.

Since there are twice as many GREEN objects as RED, it is reasonable to believe that a new case (which hasn't been observed yet) is twice as likely to have membership GREEN rather than RED. In the Bayesian analysis, this belief is known as the prior probability. Prior probabilities are based on previous experience, in this case the percentage of GREEN and RED objects, and often used to predict outcomes before they actually happen.

Thus, we can write:

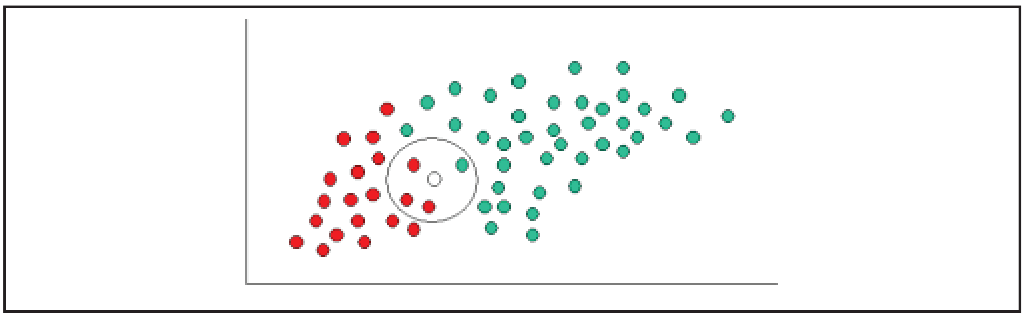
$$\text{Prior probability for Green} \propto \frac{\text{Number of green objects}}{\text{Total number of objects}}$$

$$\text{Prior probability for Red} \propto \frac{\text{Number of red objects}}{\text{Total number of objects}}$$

Since there is a total of 60 objects, 40 of which are GREEN and 20 RED, our prior probabilities for class membership are:

$$\text{Prior probability for Green } \mu \ \frac{40}{60}$$

$$\text{Prior probability for Red } \mu \ \frac{20}{60}$$



Having formulated our prior probability, we are now ready to classify a new object (WHITE circle). Since the objects are well clustered, it is reasonable to assume that the more GREEN (or RED) objects in the vicinity of X, the more likely that the new cases belong to that particular color. To measure this likelihood, we draw a circle around X which encompasses a number (to be chosen a priori) of points irrespective of their class labels. Then we calculate the number of points in the circle belonging to each class label. From this we calculate the likelihood:

$$\text{Likelihood of X given Green} \propto \frac{\text{Number of Green in the vicinity of X}}{\text{Total number of Green cases}}$$

$$\text{Likelihood of X given Red} \propto \frac{\text{Number of Red in the vicinity of X}}{\text{Total number of Red cases}}$$

From the illustration above, it is clear that Likelihood of X given GREEN is smaller than Likelihood of X given RED, since the circle encompasses 1 GREEN object and 3 RED ones. Thus:

Probability of X given Green µ $\frac{1}{40}$

Probability of X given Red µ $\frac{3}{20}$

Although the prior probabilities indicate that X may belong to GREEN (given that there are twice as many GREEN compared to RED) the likelihood indicates otherwise; that the class membership of X is RED (given that there are more RED objects in the vicinity of X than GREEN). In the Bayesian analysis, the final classification is produced by combining both sources of information, i.e., the prior and the likelihood, to form a posterior probability using the so-called Bayes' rule (named after Rev. Thomas Bayes 1702-1761).

Posterior probability of X being Green µ

Prior probability of Green × Likelihood of X given Green

$$= \frac{4}{6} \times \frac{1}{40} = \frac{1}{60}$$

Posterior probability of X being Red µ

Prior probability of Red × Likelihood of X given Red

$$= \frac{2}{6} \times \frac{3}{20} = \frac{1}{20}$$

Finally, we classify X as RED since its class membership achieves the largest posterior probability.

How Effective are Bayesian Classifiers?

In theory, Bayesian classifiers have the minimum error rate in comparison to all other classifiers. However, in practice this is not always the case owing to inaccuracies in the assumptions made for its use, such as class conditional independence, and the lack of available probability data. However, various empirical studies of this classifier in comparison to decision tree and neural network classifiers have found it to be comparable in some domains.

Bayesian classifiers are also useful in that they provide a theoretical justification for other classifiers which do not explicitly use Bayes theorem. For example, under certain assumptions, it can be shown that many neural network and curve fitting algorithms output the maximum posteriori hypothesis, as does the naive Bayesian classifier.

Task "Classification is a data mining technique used to predict group membership for data instances". Discuss.

4.5 Distance-based Algorithms

Distance-based algorithms assume that an object is more similar to the objects within the same class as opposed to objects from other classes. Therefore, the classification of the target object is affected by the objects that are similar to it. The concept of distance is used to measure the dissimilarity between objects. In other words, two similar objects can be considered close to each other in the sample space. The two key issues in distance-based classification are choosing the proper distance function and the design of the classification algorithm. Many kinds of distance functions can be used, such as city block distance or Euclidean distance. Different distances have different characteristics, which fit various types of data. Classification algorithms must determine the class of target according to objects close to it. One of the most effective techniques is K-Nearest Neighbors (KNN). Using the K-closest objects, the target object is assigned the class that contains the most objects. KNN is widely used in text classification, web mining and stream data mining.

4.6 Distance Functions

Distance-based algorithms rely on distance functions to measure the dis-similarity between the objects. Selecting a distance function is not only the first step of the algorithms, but also a critical step. Different distance functions have different characteristics, which fit various types of data. There does not exist a distance function that can deal with every type of data. So the performance of the algorithm heavily depends on whether a proper distance function is chosen for that particular data. For a set X, the distance function $d: X \times X \rightarrow R$, for all x, y, z € X, satisfies

$d(x, y) \geq 0$,

$d(x, y) = 0$ if and only if $x = y$,

$d(x, y) = d(y, x)$ (symmetry law), and

$d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality).

Interestingly, several distance functions used in practice do not necessarily satisfy all four of the constraints listed above. For example, the squared Euclidean distance does not satisfy the triangle inequality and the Kullback-Leibler distance function used in document clustering is not symmetric. A good distance function should be invariant to the natural data transformations that do not affect the class of the objects.

City Block Distance

City block distance, sometimes called Manhattan distance is defines as

Let x, y € X, where x = {x_1, x_2, …., x_k}and y = {y_1, y_2, …., y_k}.

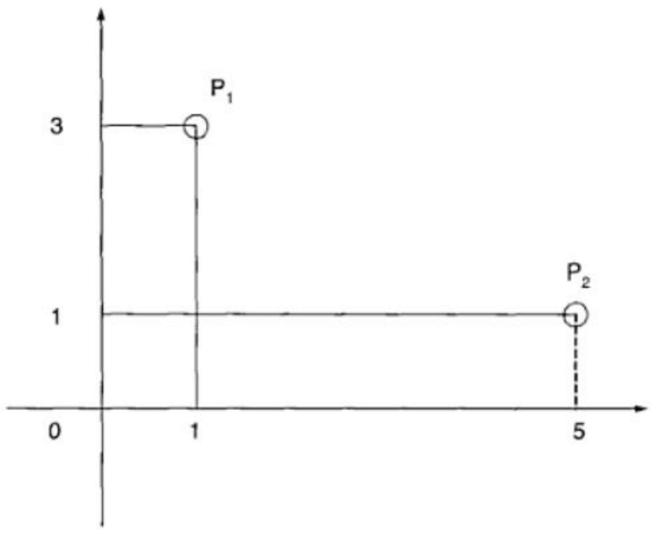
Then, $d_{CityBlock}$ (x, y) = $\sum^{k}_{I=1} | x_i - y_i |$

This measure reflects the sum of the absolute distances along each coordinate axis. In Figure 4.2, the city block distance between P_1 and P_2 is given by

$D(P_1, P_2) = |1 - 5| + |3 - 1| = 6$

Although the city block distance is easy to compute, it is variant to scaling, rotation and many other transformations. In other words, the similarity is not preserved by the city block distance after these transformations. Such a distance measure would not be appropriate for many types of data (e.g., images) which may be invariant to rotation and scaling.

Figure 4.2: City Clock Distance between two Points in 2D Space



Euclidean Distance

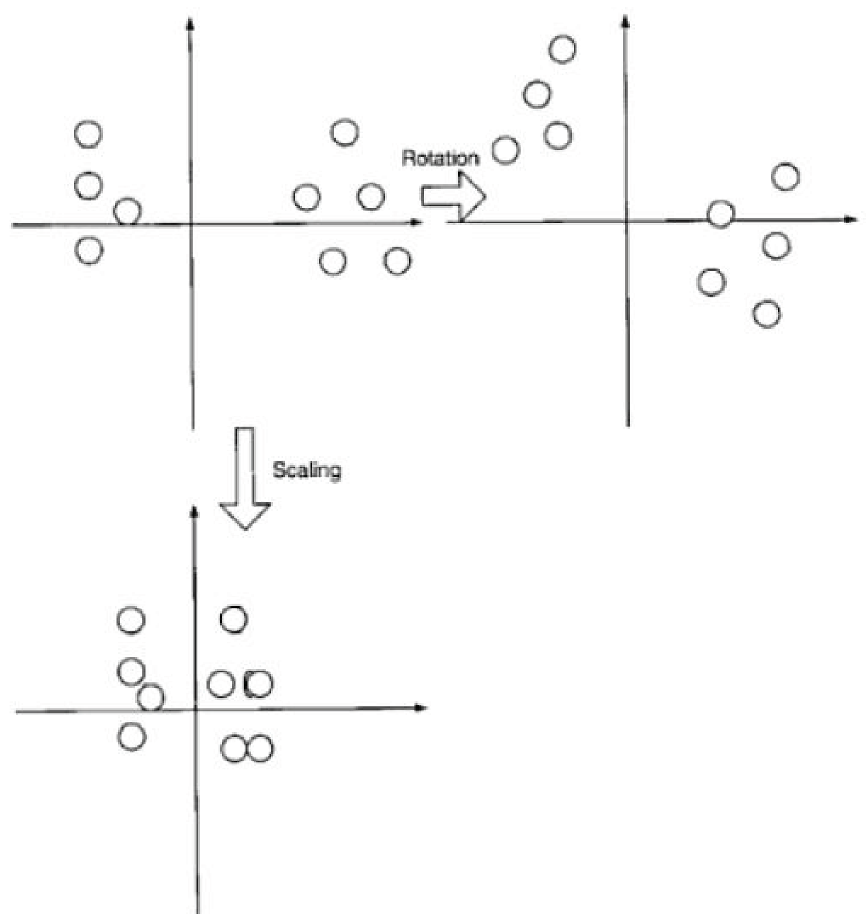
Euclidean distance is the most common distance used as the dissimilarity measure. It is defined as

$$d_{Euclidean}(x, y) = \left(\sum_{i=1}^{k} |x_i - y_i|^2 \right)^{1/2} .$$

Figure 4.3 illustrate the effects the rotations of scaling on Euclidean distance in a 2D space. It is obvious from Figure 4.3 that dissimilarity is preserved after rotation. But after scaling the x-axis, the dissimilarity between objects is changed. So Euclidean distance is invariant to rotation, but not to scaling. If rotation is the only acceptable operation for an image database, Euclidean distance would be a good choice.

Notes

Figure 4.3: Effects of Rotation and Scaling on Euclidean Distance



Other Distances

There are also many other distances that can be used for different data. Edit distance fits sequence and text data. The Tanimoto distance is suitable for data with binary-valued features.

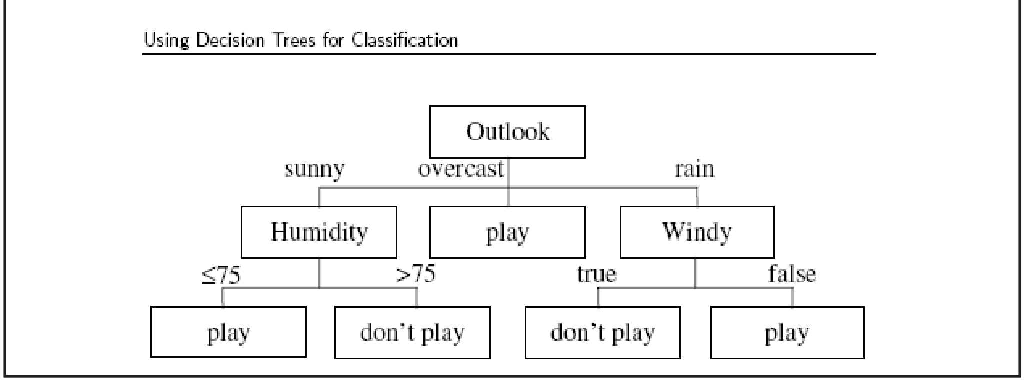
Actually, data normalization is one way to overcome the limitation of the distance functions. Functions. For example, normalizing the data to the same scale can overcome the scaling problem of Euclidean distance, however, normalization may lead to information loss and lower classification accuracy.

4.7 Classification by Decision Tree

A decision tree is a flow-chart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distributions. The topmost node in a tree is the root node.

For example, to decide whether play golf or not, let us consider the following decision tree (see Figure)

Figure 4.4: Decision Tree for the Golf Example



In order to determine the decision (classification) for a given set of weather conditions from the decision tree, first look at the value of Outlook. There are three possibilities.

1. If the value of Outlook is sunny, next consider the value of Humidity. If the value is less than or equal to 75 the decision is play. Otherwise the decision is don't play.
2. If the value of Outlook is overcast, the decision is play.
3. If the value of Outlook is rain, next consider the value of Windy. If the value is true the decision is don't play, otherwise the decision is play.

Decision Trees are useful for predicting exact outcomes. Applying the decision trees algorithm to a training dataset results in the formation of a tree that allows the user to map a path to a successful outcome. At every node along the tree, the user answers a question (or makes a "decision"), such as "play" or "don't play".

The decision trees algorithm would be useful for a bank that wants to ascertain the characteristics of good customers. In this case, the predicted outcome is whether or not the applicant represents a bad credit risk. The outcome of a decision tree may be a Yes/No result (applicant is/is not a bad credit risk) or a list of numeric values, with each value assigned a probability.

The training dataset consists of the historical data collected from past loans. Attributes that affect credit risk might include the customer's educational level, the number of kids the customer has, or the total household income. Each split on the tree represents a decision that influences the final predicted variable. For example, a customer who graduated from high school may be more likely to pay back the loan. The variable used in the first split is considered the most significant factor. So if educational level is in the first split, it is the factor that most influences credit risk.

Decision trees have been used in many application areas ranging from medicine to game theory and business. Decision trees are the basis of several commercial rule induction systems.

4.7.1 Basic Algorithm for Learning Decision Trees

Algorithm: Generate a decision tree from the given training data.

Input: The training samples, samples, represented by discrete-valued attributes; the set of candidate attributes, attribute-list.

Output: A decision tree.

Notes

Method

```
1.  Create a node N;
2.  If samples are all of the same class, C then
3.  Return N as a leaf node labeled with the class C
4.  If attribute-list is empty then
5.  Return N as a leaf node labeled with the most common class in samples; // majority
    voting
6.  Select test-attribute, the attribute among attribute-list with the highest information gain
7.  Label node N with test-attribute
8.  For each known value $a_i$ of test-attribute // partition the samples
9.  Grow a branch from node N for the condition test-attribute=$a_i$
10. Let $s_i$ be the set of samples in samples for which test-attribute=$a_i$; // a partition
11. If $s_i$ is empty then
12. Attach a leaf labeled with the most common class in samples
13. Else attach the node returned by Generate decision tree ($s_i$, attribute-list - test-attribute).
```

4.7.2 Decision Tree Induction

The automatic generation of decision rules from examples is known as rule induction or automatic rule induction.

Generating decision rules in the implicit form of a decision tree are also often called rule induction, but the terms tree induction or decision tree inductions are sometimes preferred.

The basic algorithm for decision tree induction is a greedy algorithm, which constructs decision trees in a top-down recursive divide-and-conquer manner. The basic algorithm for learning decision trees, is a version of ID3, a well-known decision tree induction algorithm.

The basic strategy is as follows:

1. The tree starts as a single node representing the training samples (step 1).
2. If the samples are all of the same class, then the node becomes a leaf and is labeled with that class (steps 2 and 3).
3. Otherwise, the algorithm uses an entropy-based measure known as information gain as a heuristic for selecting the attribute that will best separate the samples into individual classes (step 6). This attribute becomes the "test" or "decision" attribute at the node (step 7). In this version of the algorithm, all attributes are categorical, i.e., discrete-valued. Continuous-valued attributes must be discretized.
4. A branch is created for each known value of the test attribute, and the samples are partitioned accordingly (steps 8-10).
5. The algorithm uses the same process recursively to form a decision tree for the samples at each partition. Once an attribute has occurred at a node, it need not be considered in any of the node's descendents (step 13).

6. The recursive partitioning stops only when any one of the following conditions is true:
   (a) All samples for a given node belong to the same class (steps 2 and 3), or
   (b) There are no remaining attributes on which the samples may be further partitioned (step 4). In this case, majority voting is employed (step 5). This involves converting the given node into a leaf and labeling it with the class in majority among samples. Alternatively, the class distribution of the node samples may be stored; or
   (c) There are no samples for the branch test-attribute=at (step 11). In this case, a leaf is created with the majority class in samples (step 12).

Decision tree induction algorithms have been used for classification in a wide range of application domains. Such systems do not use domain knowledge. The learning and classification steps of decision tree induction are generally fast. Classification accuracy is typically high for data where the mapping of classes consists of long and thin regions in concept space.

Attribute Selection Measure

The information gain measure is used to select the test attribute at each node in the tree. Such a measure is also referred to as an *attribute selection measure.* The attribute with the highest information gain is chosen as the test attribute for the current node. This attribute minimizes the information needed to classify the samples in the resulting partitions and reflects the least randomness or "impurity" in these partitions. Such an information-theoretic approach minimizes the expected number of tests needed to classify an object and guarantees that a simple (but not necessarily the simplest) tree is found.

Let *S* be a set consisting of *s* data samples. Suppose the class label attribute has *m* distinct values defining *m* distinct classes, *d* (for *i* = 1,..., *m).* Let *s,* be the number of samples of *S* in class *d.* The expected information needed to classify a given sample is given by:

$$I(s_1, s_2 \ldots, s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

where *p,* is the probability than an arbitrary sample belongs to class *d* and is estimated by S_i/s. Note that a *log* function to the base 2 is used since the information is encoded in bits.

Attribute *A* can be used to partition *S* into *v* subsets, $\{S_1, S_2, \ldots, S_V\}$, where S_j contains those samples in *S* that have value a_j of *A*. If *A* were selected as the test attribute (i.e., best attribute for splitting), then these subsets would correspond to the branches grown from the node containing the set *S*. Let s_j be the number of samples of class *d* in a subset S_j. The entropy, or expected information based on the partitioning into subsets by *A* is given by:

$$E(A) = \sum_{i=1}^{v} \frac{s_{1j} + \ldots + s_{mj}}{s} I(s_{1j} \ldots s_{mj}).$$

The term $\sum_{i=1}^{v} \frac{s_{1j} + \ldots + s_{mj}}{s}$ acts as the weight of the jth subset and is the number of samples in the subset (i.e., having value a_j of A) divided by the total number of samples in S. The smaller the entropy value is, the greater the purity of the subset partitions. The encoding information that would be gained by branching on A is

Gain(A) = $I(s_l, s_2, \ldots, s_m)$ - E(A).

In other words, *Gain(A)* is the expected reduction in entropy caused by knowing the value of attribute *A*.

The algorithm computes the information gain of each attribute. The attribute with the highest information gain is chosen as the test attribute for the given set *S.* A node is created and labeled with the attribute, branches are created for each value of the attribute, and the samples are partitioned accordingly.

4.7.3 Tree Pruning

After building a decision tree a tree pruning step can be performed to reduce the size of the decision tree. Decision trees that are too large are susceptible to a phenomenon called as overfitting. Pruning helps by trimming the branches that reflects anomalies in the training data due to noise or outliers and helps the initial tree in a way that improves the generalization capability of the tree. Such methods typically use statistical measures to remove the least reliable branches, generally resulting in faster classification and an improvement in the ability of the tree to correctly classify independent test data.

There are two common approaches to tree pruning.

Pre-pruning Approach

In the pre-pruning approach, a tree is "pruned" by halting its construction early (e.g., by deciding not to further split or partition the subset of training samples at a given node). Upon halting, the node becomes a leaf. The leaf may hold the most frequent class among the subset samples, or the probability distribution of those samples.

When constructing a tree, measures such as statistical significance, x^2, information gain, etc., can be used to assess the goodness of a split. If partitioning the samples at a node would result in a split that falls below a pre-specified threshold, then further partitioning of the given subset is halted. There are difficulties, however, in choosing an appropriate threshold. High thresholds could result in oversimplified trees, while low thresholds could result in very little simplification.

Post-pruning Approach

The post-pruning approach removes branches from a "fully grown" tree. A tree node is pruned by removing its branches.

The *cost complexity* pruning algorithm is an example of the post-pruning approach. The pruned node becomes a leaf and is labeled by the most frequent class among its former branches. For each non-leaf node in the tree, the algorithm calculates the expected error rate that would occur if the subtree at that node were pruned. Next, the expected error rate occurring if the node were not pruned is calculated using the error rates for each branch, combined by weighting according to the proportion of observations along each branch. If pruning the node leads to a greater expected error rate, then the subtree is kept. Otherwise, it is pruned. After generating a set of progressively pruned trees, an independent test set is used to estimate the accuracy of each tree. The decision tree that minimizes the expected error rate is preferred.

Example: A company is trying to decide whether to bid for a certain contract or not. They estimate that merely preparing the bid will cost £10,000. If their company bid then they estimate that there is a 50% chance that their bid will be put on the "short-list", otherwise their bid will be rejected.

Once "short-listed" the company will have to supply further detailed information (entailing costs estimated at £5,000). After this stage their bid will either be accepted or rejected.

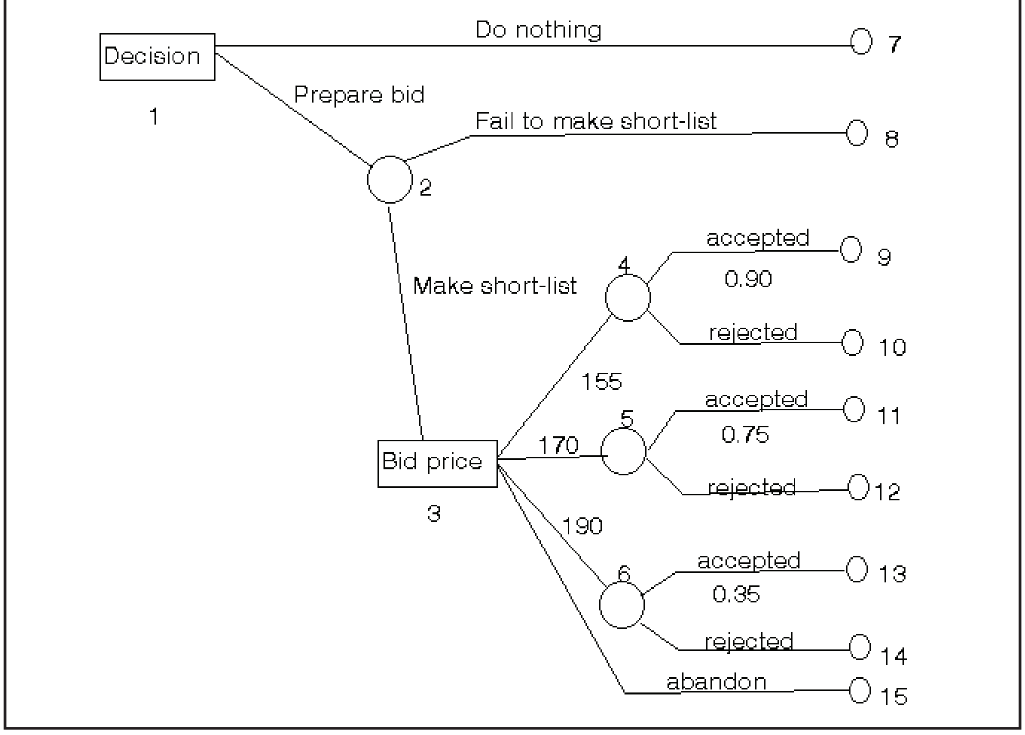
The company estimate that the labour and material costs associated with the contract are £127,000. They are considering three possible bid prices, namely £155,000, £170,000 and £190,000. They estimate that the probability of these bids being accepted (once they have been short-listed) is 0.90, 0.75 and 0.35 respectively.

What should the company do and what is the expected monetary value of your suggested course of action?

Solution

The decision tree for the problem is shown in Figure 4.5:

Figure 4.5: Decision Tree



Below we carry out step 1 of the decision tree solution procedure which (for this example) involves working out the total profit for each of the paths from the initial node to the terminal node (all figures in £'000).

1. Path to terminal node 7 – the company do nothing

   Total profit = 0

2. Path to terminal node 8 – the company prepare the bid but fail to make the short-list

   Total cost = 10 Total profit = –10

3. Path to terminal node 9 – the company prepare the bid, make the short-list and their bid of £155K is accepted

   Total cost = 10 + 5 + 127 Total revenue = 155 Total profit = 13

4. Path to terminal node 10 – the company prepare the bid, make the short-list but their bid of £155K is unsuccessful

   Total cost = 10 + 5 Total profit = –15

5. Path to terminal node 11 – the company prepare the bid, make the short-list and their bid of £170K is accepted

   Total cost = 10 + 5 + 127 Total revenue = 170 Total profit = 28

6. Path to terminal node 12 - the company prepare the bid, make the short-list but their bid of £170K is unsuccessful

   Total cost = 10 + 5 Total profit = –15

7. Path to terminal node 13 - the company prepare the bid, make the short-list and their bid of £190K is accepted

   Total cost = 10 + 5 + 127 Total revenue = 190 Total profit = 48

8. Path to terminal node 14 - the company prepare the bid, make the short-list but their bid of £190K is unsuccessful

   Total cost = 10 + 5 Total profit = –15

9. Path to terminal node 15 - the company prepare the bid and make the short-list and then decide to abandon bidding (an implicit option available to the company)

   Total cost = 10 + 5 Total profit = –15

Hence we can arrive at the table below indicating for each branch the total profit involved in that branch from the initial node to the terminal node.

Terminal node	Total profit £
7	0
8	–10
9	13
10	–15
11	28
11	–15
13	48
14	–15
15	–15

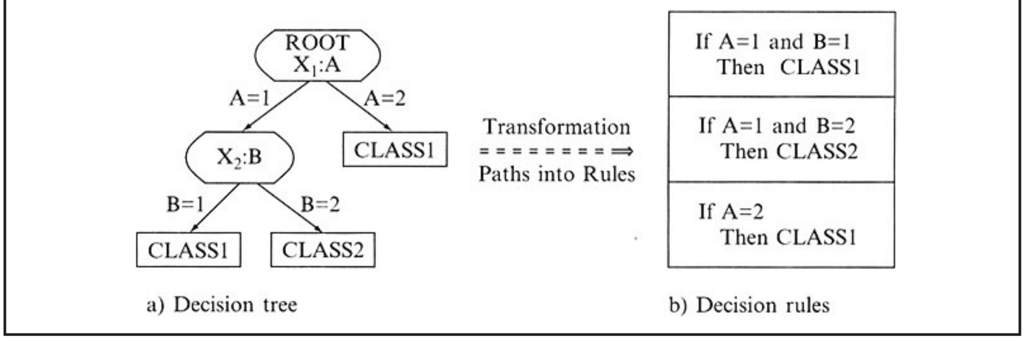
We can now carry out the second step of the decision tree solution procedure where we work from the right-hand side of the diagram back to the left-hand side.

Task “Different distance functions have different characteristics, which fit various types of data.” Explain

4.7.4 Extracting Classification Rules from Decision Trees

Even though the pruned trees are more compact than the originals, they can still be very complex. Large decision trees are difficult to understand because each node has a specific context established by the outcomes of tests at antecedent nodes. To make a decision-tree model more readable, a path to each leaf can be transformed into an IF-THEN production rule. The IF part consists of all tests on a path, and the THEN part is a final classification. Rules in this form are called decision rules, and a collection of decision rules for all leaf nodes would classify samples exactly as the tree does. As a consequence of their tree origin, the IF parts of the rules would be mutually exclusive and exhaustive, so the order of the rules would not matter. An example of the transformation of a decision tree into a set of decision rules is given in Figure 4.6, where the two given attributes, A and B, may have two possible values, 1 and 2, and the final classification is into one of two classes.

Figure 4.6: Transformation of a Decision Tree into Decision Rules



A rule can be "pruned" by removing any condition in its antecedent that does not improve the estimated accuracy of the rule. For each class, rules within a class may then be ranked according to their estimated accuracy. Since it is possible that a given test sample will not satisfy any rule antecedent, a default rule assigning the majority class is typically added to the resulting rule set.

4.8 Neural Network based Algorithms

Artificial Neural Network

An artificial neural network is a system based on the operation of biological neural networks, in other words, is an emulation of biological neural system. Why would be necessary the implementation of artificial neural networks? Although computing these days is truly advanced, there are certain tasks that a program made for a common microprocessor is unable to perform; even so a software implementation of a neural network can be made with their advantages and disadvantages.

Advantages of Neural Network

The advantages of neural network are:

1. A neural network can perform tasks that a linear program can not.
2. When an element of the neural network fails, it can continue without any problem by their parallel nature.
3. A neural network learns and does not need to be reprogrammed.
4. It can be implemented in any application.
5. It can be implemented without any problem.

Disadvantages of Neural Network

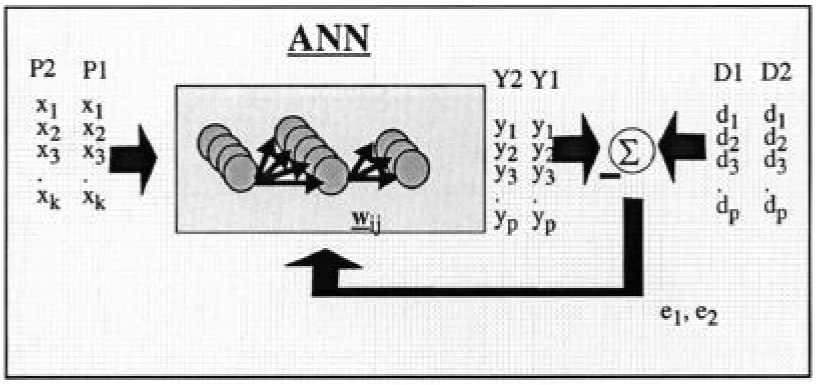
The disadvantages of neural network are:

1. The neural network needs training to operate.
2. The architecture of a neural network is different from the architecture of microprocessors therefore needs to be emulated.
3. Requires high processing time for large neural networks.

Notes

Another aspect of the artificial neural networks is that there are different architectures, which consequently requires different types of algorithms, but despite to be an apparently complex system, a neural network is relatively simple.

Artificial Neural Networks (ANN) are among the newest signal-processing technologies in the engineer's toolbox. The field is highly interdisciplinary, but our approach will restrict the view to the engineering perspective. In engineering, neural networks serve two important functions: as pattern classifiers and as nonlinear adaptive filters. We will provide a brief overview of the theory, learning rules, and applications of the most important neural network models. Definitions and Style of Computation An Artificial Neural Network is an adaptive, most often nonlinear system that learns to perform a function (an input/output map) from data. Adaptive means that the system parameters are changed during operation, normally called the training phase. After the training phase the Artificial Neural Network parameters are fixed and the system is deployed to solve the problem at hand (the testing phase). The Artificial Neural Network is built with a systematic step-by-step procedure to optimize a performance criterion or to follow some implicit internal constraint, which is commonly referred to as the learning rule . The input/ output training data are fundamental in neural network technology, because they convey the necessary information to "discover" the optimal operating point. The nonlinear nature of the neural network processing elements (PEs) provides the system with lots of flexibility to achieve practically any desired input/output map, i.e., some Artificial Neural Networks are universal mappers . There is a style in neural computation that is worth describing.

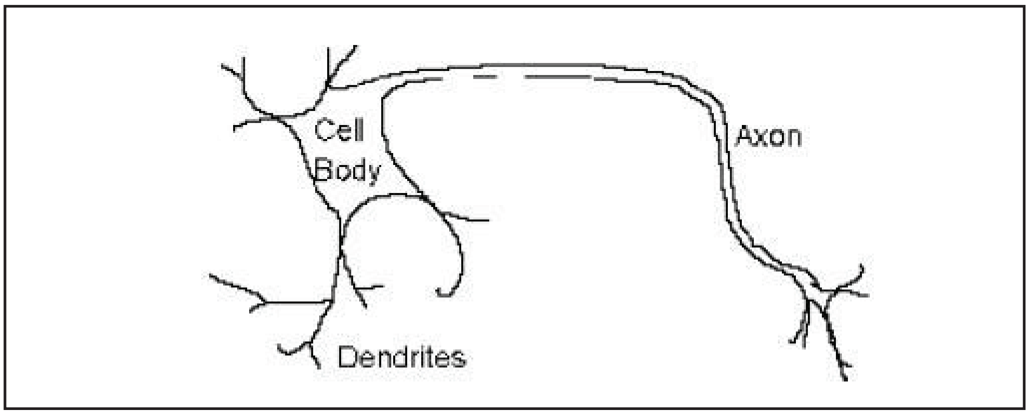


The style of neural computation.

An input is presented to the neural network and a corresponding desired or target response set at the output (when this is the case the training is called supervised). An error is composed from the difference between the desired response and the system output. This error information is fed back to the system and adjusts the system parameters in a systematic fashion (the learning rule). The process is repeated until the performance is acceptable. It is clear from this description that the performance hinges heavily on the data. If one does not have data that cover a significant portion of the operating conditions or if they are noisy, then neural network technology is probably not the right solution. On the other hand, if there is plenty of data and the problem is poorly understood to derive an approximate model, then neural network technology is a good choice. This operating procedure should be contrasted with the traditional engineering design, made of exhaustive subsystem specifications and intercommunication protocols. In artificial neural networks, the designer chooses the network topology, the performance function, the learning rule, and the criterion to stop the training phase, but the system automatically adjusts the parameters. So, it is difficult to bring a priori information into the design, and when the system does not work properly it is also hard to incrementally refine the solution. But ANN-based solutions are extremely efficient in terms of development time and resources, and in many difficult problems artificial neural networks provide performance that is difficult to match with other technologies. Denker 10 years ago said that "artificial neural networks are the second best way to implement a solution" motivated by the simplicity of their design and because of their universality, only shadowed by the traditional design obtained by studying the physics of the problem. At present, artificial neural networks are emerging as the technology of choice for many applications, such as pattern recognition, prediction, system identification, and control.

The Biological Model

Artificial neural networks emerged after the introduction of simplified neurons by McCulloch and Pitts in 1943 (McCulloch & Pitts, 1943). These neurons were presented as models of biological neurons and as conceptual components for circuits that could perform computational tasks. The basic model of the neuron is founded upon the functionality of a biological neuron. "Neurons are the basic signaling units of the nervous system" and "each neuron is a discrete cell whose several processes arise from its cell body".

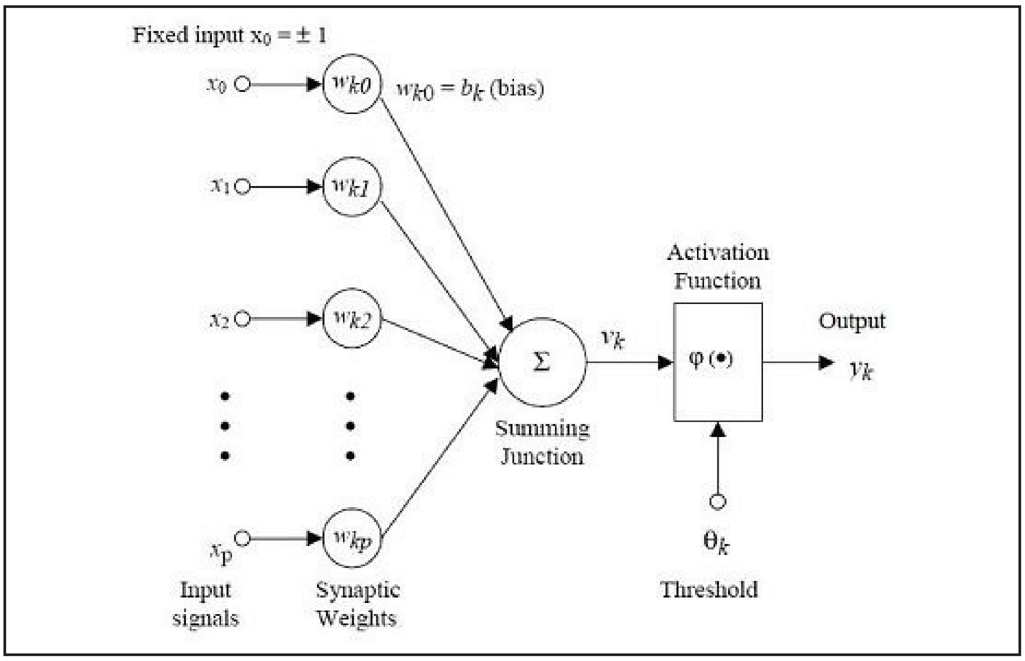


The neuron has four main regions to its structure. The cell body, or soma, has two offshoots from it, the dendrites, and the axon, which end in presynaptic terminals. The cell body is the heart of the cell, containing the nucleus and maintaining protein synthesis. A neuron may have many dendrites, which branch out in a treelike structure, and receive signals from other neurons. A neuron usually only has one axon which grows out from a part of the cell body called the axon hillock. The axon conducts electric signals generated at the axon hillock down its length. These electric signals are called action potentials. The other end of the axon may split into several branches, which end in a presynaptic terminal. Action potentials are the electric signals that neurons use to convey information to the brain. All these signals are identical. Therefore, the brain determines what type of information is being received based on the path that the signal took. The brain analyzes the patterns of signals being sent and from that information it can interpret the type of information being received. Myelin is the fatty tissue that surrounds and insulates the axon. Often short axons do not need this insulation. There are uninsulated parts of the axon. These areas are called Nodes of Ranvier. At these nodes, the signal traveling down the axon is regenerated. This ensures that the signal traveling down the axon travels fast and remains constant (i.e. very short propagation delay and no weakening of the signal). The synapse is the area of contact between two neurons. The neurons do not actually physically touch. They are separated by the synaptic cleft, and electric signals are sent through chemical 13 interaction. The neuron sending the signal is called the presynaptic cell and the neuron receiving the signal is called the postsynaptic cell. The signals are generated by the membrane potential, which is based on the differences in concentration of sodium and potassium ions inside and outside the cell membrane. Neurons can be classified by their number of processes (or appendages), or by their function. If they are classified by the number of processes, they fall into three categories. Unipolar neurons have a single process (dendrites and axon are located on the same stem), and are most common in invertebrates. In bipolar neurons, the dendrite and axon are the neuron's two separate processes. Bipolar neurons have a subclass called pseudo-bipolar neurons, which are used to send sensory information to the spinal cord. Finally, multipolar neurons are most common in mammals. Examples of these neurons are spinal motor neurons, pyramidal cells and Purkinje cells (in the cerebellum). If classified by function, neurons again fall into three separate categories. The first group is sensory, or afferent, neurons, which provide information for perception and motor coordination. The second group provides information (or instructions) to muscles and glands and is therefore called motor neurons. The last group, interneuronal, contains all other neurons and has two subclasses. One group called relay or projection interneurons have long axons and connect different parts of the brain. The other group called local interneurons are only used in local circuits.

The Mathematical Model

When creating a functional model of the biological neuron, there are three basic components of importance. First, the synapses of the neuron are modeled as weights. The strength of the connection between an input and a neuron is noted by the value of the weight. Negative weight values reflect inhibitory connections, while positive values designate excitatory connections [Haykin]. The next two components model the actual activity within the neuron cell. An adder sums up all the inputs modified by their respective weights. This activity is referred to as linear combination. Finally, an activation function controls the amplitude of the output of the neuron. An acceptable range of output is usually between 0 and 1, or -1 and 1.

Mathematically, this process is described in the figure



From this model the interval activity of the neuron can be shown to be:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j$$

The output of the neuron, y_k, would therefore be the outcome of some activation function on the value of v_k.

Activation Functions

As mentioned previously, the activation function acts as a squashing function, such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). In general, there are three types of activation functions, denoted by Φ(.) . First, there is the Threshold Function which takes on a value of 0 if the summed input is less than a certain threshold value (v), and the value 1 if the summed input is greater than or equal to the threshold value.

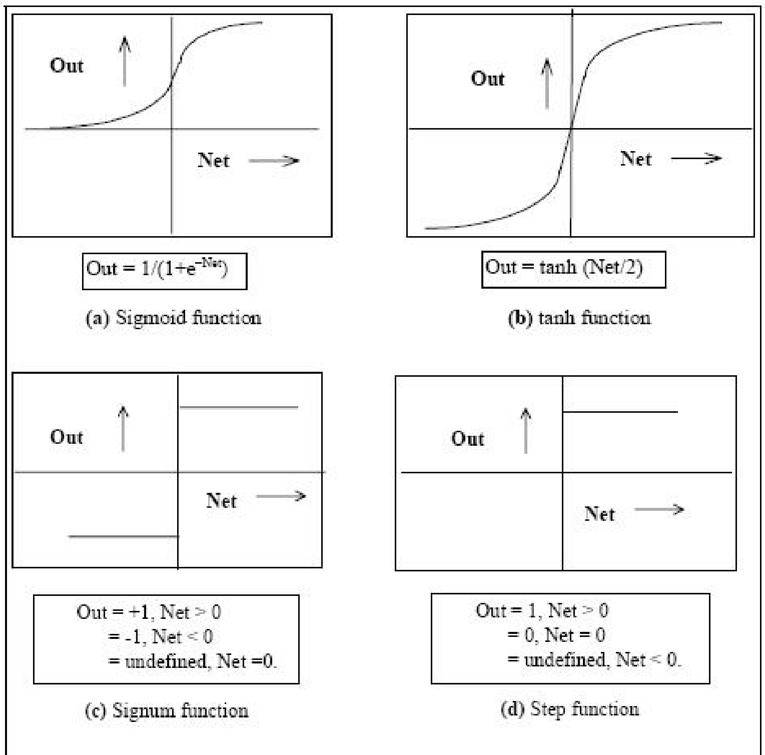
$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$

Secondly, there is the Piecewise-Linear function. This function again can take on the values of 0 or 1, but can also take on values between that depending on the amplification factor in a certain region of linear operation.

$$\varphi(v) = \begin{cases} 1 & v \geq \frac{1}{2} \\ v & -\frac{1}{2} > v > \frac{1}{2} \\ 0 & v \leq -\frac{1}{2} \end{cases}$$

Thirdly, there is the sigmoid function. This function can range between 0 and 1, but it is also sometimes useful to use the -1 to 1 range. An example of the sigmoid function is the hyperbolic tangent function.

$$\varphi(v) = \tanh\left(\frac{v}{2}\right) = \frac{1 - \exp(-v)}{1 + \exp(-v)}$$



Common non-linear functions used for synaptic inhibition. Soft non-linearity: (a) Sigmoid and (b) tanh; Hard non-linearity: (c) Signum and (d) Step.

The artifcial neural networks which we describe are all variations on the parallel distributed processing (PDP) idea. The architecture of each neural network is based on very similar building blocks which perform the processing. In this unit we first discuss these processing units and discuss diferent neural network topologies. Learning strategies as a basis for an adaptive system

4.9 Rule-based Algorithms

One of the most well-studied methods for producing sets of classification rules from examples is rule algorithms. They attempt to cover all instances of each class while excluding instances not in the class. The main point is that covering algorithms (rule-based) work on a specific class at a time, ignoring the rest of the classes. For instance, if a rule is desired to classify the weather as warm, then the covering algorithm attempts to x in the statement.

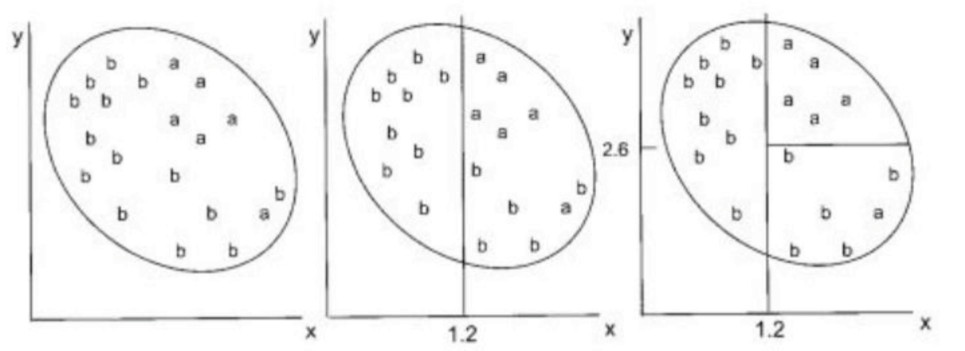
If x, then class = warm,

With the condition that produces the best probability for the weather to be warm. Covering algorithms follows these three steps:

1. Generate rule R on training data S
2. Remove the training data covered by rule R
3. Repeat the process

This method can be visualized in the 2D space of instances illustrated in Figure 4.7. First, a rule is constructed to cover a’s by splitting the space vertically at x = 1.2 and then further splitting it horizontally at y = 2.6, leading to the rule.

Figure 4.7: Covering Algorithm Demonstration



If x > 1.2 AND y > 2.6, then class = a

Second, the following procedure is used to construct rules to cover b’s:

If x ≤ 1.2, then class = b

If x > 1.2 and y ≤ 2.6, then class = b

Note that one a is incorrectly covered by these rules, and more tests can be added to exclude that a from b’s cover and include it in the a’s cover.

1R Algorithm

One of the simple approaches used to find classification rules is called 1R, as it generated a one level decision tree. This algorithm examines the "rule that classify an object on the basis of a single attribute".

The basic idea is that rules are constructed to test a single attribute and branch for every value of that attribute. For each branch, the class with the best classification is the one occurring most often in the training data. The error rate of the rules is then determined by counting the number of instances that do not have the majority class in the training data. Finally, the error rate for each attribute's rule set is evaluated, and the rule set with the minimum error rate is chosen.

A comprehensive comparative evaluation of the performance of 1R and other methods on 16 datasets (many of which were most commonly used in machine learning research) was performed. Despite it simplicity, 1R produced surprisingly accurate rules, just a few percentage points lower in accuracy than the decision produced by the state of the art algorithm (C4). The decision tree produced by C4 were in most cases considerably larger than 1R's rules, and the rules generated by 1R were much easier to interpret. 1R teherfore provides a baseline performance using a rudimentary technique to be used before progressing to more sophisticated algorithms.

Other Algorithms

Basic covering algorithms construct rules that classify training data perfectly, that is, they tend to overfit the training set causing insufficient generalization and difficulty for processing new data. However, for applications in real world domains, methods for handling noisy data, mechanisms for avoiding overfitting even on training data, and relaxation requirements of the constraints are needed. Pruning is one of the ways of dealing with these problems, and it approaches the problem of overfitting by learning a general concept from the training set "to improve the prediction of unseen instance". The concept of Reduced Error Pruning (REP) was developed by, where some of the training examples were withheld as a test set and performance of the rule was measured on them. Also, Incremental Reduced Error Pruning (IREP) has proven to be efficient in handling over-fitting, and it form the basis of RIPPER. SLIPPER (Simple Learner with Iterative Pruning to Produce Error Reduction) uses "confidence-rated boosting to learn an ensemble of rules."

Applications of Rule-based Algorithms

Rule based algorithms are widely used for deriving classification rules applied in medical sciences for diagnosing illnesses, business planning, banking government and different disciplines of science. Particularly, covering algorithms have deep roots in machine learning. Within data mining, covering algorithms including SWAP-1, RIPPER, and DAIRY are used in text classification, adapted in gene expression programming for discovering classification rules.

Task Explain how will you remove the training data covered by rule R.

4.10 Combining Techniques

Data mining is an application-driven field where research questions tend to be motivated by real-world data sets. In this context, a broad spectrum of formalisms and techniques has been proposed by researchers in a large number of applications. Organizing them in inherently rather difficult; that is why we highlight the central role played by the various types of data motivating the current research.

Notes

We begin with what us perhaps the best-known data type in traditional data analysis, namely, d-dimensional vectors x of measurements on N objects or individual, or N objects where for each of which we have d measurements or attributes. Such data is often referred to as multivariate data and can be thought of as an N x d data matrix. Classical problems in data analysis involving multivariate data include classification (learning a functional mapping from a vector x to y where y is a categorical, or scalar, target variable of interest), regression (same as classification, except y, which takes real values), clustering (learning a function that maps x into a set of categories, where the categories are unknown a priori), and density estimation (estimating the probability density function, or PDF, for x, p (x)).

The dimensionality d of the vectors x plays a significant role in multivariate modeling. In problems like text classification and clustering of gene expression data, d can be as large 10^3 and 10^4 dimensions. Density estimation theory shows that the amount of data needed to reliably to estimate a density function scales exponentially in d (the so-called "curse of dimensionality"). Fortunately, many predictive problems including classification and regression do not need a full d dimensional estimate of the PDF p(x), relying instead on the simpler problem of determining of a conditional probability density function p(y/x), where y is the variable whose value the data minor wants to predict.

Recent research has shown that combining different models can be effective in reducing the instability that results form predictions using a single model fit to a single set of data. A variety of model-combining techniques (with exotic names like bagging, boosting, and stacking) combine massive computational search methods with variance-reduction ideas from statistics; the result is relatively powerful automated schemes for building multivariate predictive models. As the data minor's multivariate toolbox expands, a significant part of the data mining is the practical intuition of the tools themselves.

Case Study

Hideaway Warehouse Management System (WMS)

The Company

Hideaway Beds – Wall Bed Company offers the Latest Designs wall beds. Wall Beds have been around since 1918 in American and Europe. The company ships their products to approximately 100 retailers in Australia as well as taking online orders directly from individual consumers.

Key Benefits

1. Order accuracy increases from 80% to 99.9 %
2. Order picking times reduced by one third

Contd...

Notes

3. Real-time visibility into warehouse inventory Levels.
4. Unit items shipped per day up from 50 to 200.

The Challenge

Hideaway managed 8,000 square-foot warehouse using a paper based system. All the assembly items required to make final products i.e. beds are done in the warehouse. Lots of subassembly items come into the warehouse required to assemble the end items. The real challenge is the lack of system since most of the picking and receiving of assembly items are done using Microsoft Office, in which manual logs were kept noting the location and movement of goods throughout the facility. When it came time to provide reports for the senior management team, the reports were built manually on an excel spreadsheet- a laborious and potentially error prone process. This cumbersome process results into late orders and deteriorating customer service levels. A system is needed to streamline the logistics situation and improve the bottom-line profits.

The Solution

Naxtor warehouse management solution (WMS) was selected to solve Company challenges.

"Several systems were explored during the initial search for a perfect fit," explains Von, Managing Director. "We are not a large company and the cost /benefit analysis couldn't justify a huge investment, so cost played a considerable role in our decision to go ahead with Naxtor WMS".

With the implementation of Naxtor WMS, Hideaway immediately saw an increase in productivity, processing picking and receiving of assembly items were done in half the time it had taken previously. Warehouse staffs were equipped with wireless handhelds to pick

Contd...

Notes

order for assembling. Once the final product is ready it is then shipped to customer and all phases of their assembly processes are thereby recorded into the warehouse management system. There was no longer a need for paper recording or manual data entry. Using RF handhelds and barcode technology Naxtor WMS allow Hideaway to track and trace every item as it is received, put-away, picked and shipped. Each member of the warehouse staff became an expert on stock and assembly location and warehouse processes.

Hideaway beds at times also required generating their own barcodes, and with Naxtor WMS with a click of a button the application generates EAN32 and many generic barcode read by most of the barcode reader. The printed labels are then attached to the incoming assembly items ready to be put away to their respective locations.

"Having the warehouse system in place means that instead of product location and warehouse layout sitting in someone's head, it's now in the system, and that makes it much more transferable so that every staff member is an expert," says Von

After only three week installation and training process, the warehouse was able to leverage the full capabilities of Naxtor WMS. The company was able to equip warehouse staff with PSC falcon line or wireless data collection and printing technologies.

4.11 Summary

- Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given sample belongs to a particular class.
- Bayesian classification is based on Bayes theorem. Bayesian classifiers exhibited high accuracy and speed when applied to large databases.
- Bayesian belief networks are graphical models, which unlike naive Bayesian classifiers, allow the representation of dependencies among subsets of attributes.
- Bayes' theorem relates the conditional and marginal probabilities of events A and B, where B has a non-vanishing probability.
- Naïve Bayes classifiers assume that the effect of a variable value on a given class is independent of the values of other variable. This assumption is called class conditional independence. It is made to simplify the computation and in this sense considered to be "Naïve".
- In theory, Bayesian classifiers have the minimum error rate in comparison to all other classifiers.
- Classification and prediction are two forms of data analysis, which can be used to extract models describing important data classes or to predict future data trends. Classification predicts categorical labels (or discrete values) where as, prediction models continuous-valued functions.
- The learning of the model is 'supervised' if it is told to which class each training sample belongs. In contrasts with unsupervised learning (or clustering), in which the class labels of the training samples are not known, and the number or set of classes to be learned may not be known in advance.
- Prediction is similar to classification, except that for prediction, the results lie in the future.
- Any of the methods and techniques used for classification may also be used, under appropriate circumstances, for prediction.

- Data cleaning, relevance analysis and data transformation are the preprocessing steps that may be applied to the data in order to help improve the accuracy, efficiency, and scalability of the classification or prediction process.
- Classification and prediction methods can be compared and evaluated according to the criteria of Predictive accuracy, Speed, Robustness, Scalability and Interpretability.
- A decision tree is a flow-chart-like tree structure, where each *internal node* denotes a test on an attribute, each *branch* represents an outcome of the test, and *leaf nodes* represent classes or class distributions. The topmost node in a tree is the *root* node.

4.12 Keywords

Bayes theorem: Bayesian classification is based on Bayes theorem.

Bayesian belief networks: These are graphical models, which unlike naive Bayesian classifiers, allow the representation of dependencies among subsets of attributes

Bayesian classification: Bayesian classifiers are statistical classifiers.

Classification: Classification is a data mining technique used to predict group membership for data instances.

Data cleaning: Data cleaning refers to the preprocessing of data in order to remove or reduce *noise* (by applying smoothing techniques, for example), and the treatment of *missing values* (e.g., by replacing a missing value with the most commonly occurring value for that attribute, or with the most probable value based on statistics).

Decision Tree: A decision tree is a flow-chart-like tree structure, where each *internal node* denotes a test on an attribute, each *branch* represents an outcome of the test, and *leaf nodes* represent classes or class distributions. The topmost node in a tree is the *root* node.

Decision tree induction: The automatic generation of decision rules from examples is known as *rule induction* or *automatic rule induction*.

Interpretability: This refers is the level of understanding and insight that is provided by the learned model.

Naive Bayesian classifiers: They assume that the effect of an attribute value on a given class is independent of the values of the other attributes.

Overfitting: Decision trees that are too large are susceptible to a phenomenon called as overfitting.

Prediction: Prediction is similar to classification, except that for prediction, the results lie in the future.

Predictive accuracy: This refers to the ability of the model to correctly predict the class label of new or previously unseen data.

Scalability: This refers to the ability of the learned model to perform efficiently on large amounts of data.

Supervised learning: The learning of the model is 'supervised' if it is told to which class each training sample belongs.

Tree pruning: After building a decision tree a tree pruning step can be performed to reduce the size of the decision tree.

Unsupervised learning: In unsupervised learning, the class labels of the training samples are not known, and the number or set of classes to be learned may not be known in advance.

Notes

4.13 Self Assessment

Fill in the blanks:

1. Bayesian classifiers are .... classifiers.
2. Bayesian classification is based on .... theorem
3. .... classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes.
4. Bayesian belief networks are .... models.
5. In theory .... classifiers have the minimum error rate in comparison to all other classifiers.
6. .... is a data mining technique used to predict group membership for data instances.
7. The learning of the model is .... if it is told to which class each training sample belongs.
8. .... refers to the ability of the model to correctly predict the class label of new or previously unseen data.
9. A .... is a flow-chart-like tree structure.
10. The .... measure is used to select the test attribute at each node in the tree.

4.14 Review Questions

1. What do you mean by classification in data mining? Write down the applications of classification in business.
2. What do you mean by prediction? Write down the applications of classification in business.
3. What is the difference between the classification and predicate?
4. Discuss the issues regarding the classification and predicate.
5. Differentiate between the supervised and unsupervised learning.
6. What do you mean by data cleaning?
7. Write down the criteria to compare and evaluate the classification and prediction methods.
8. What is a decision tree? Explain with the help of a suitable example.
9. Write down the basic algorithm for decision learning trees.
10. How effective are Bayesian classifiers?
11. Write short notes on the followings:
    (a) Bayesian classification
    (b) Bayes theorem
    (c) Naive Bayesian classification

Answers: Self Assessment

1. Statistical	2. Bayes
3. Naïve Bayesian	4. Graphical
5. Bayesian	6. Classification
7. Supervised	8. Predictive accuracy
9. Decision tree	10. Information gain

4.15 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 5: Data Warehouse Research – Issues and Research

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe data extraction and reconciliation
- Know data aggregation and customization
- Explain query optimization
- Know modelling and measuring data warehouse quality
- Explain three perspectives of data warehouse metadata

Introduction

An enterprise data warehouse often fetches records from several disparate systems and store them centrally in an enterprise-wide warehouse. But what is the guarantee that the quality of data will not degrade in the process of centralization?

Many of the data warehouses are built on n-tier architecture with multiple data extraction and data insertion jobs between two consecutive tiers. As it happens, the nature of the data changes as it passes from one tier to the next tier. Data reconciliation is the method of reconciling or tie-up the data between any two consecutive tiers (layers).

5.1 Data Extraction

Extraction is the operation of extracting data from a source system for further use in a data warehouse environment. This is the first step of the ETL process. After the extraction, this data can be transformed and loaded into the data warehouse.

The source systems for a data warehouse are typically transaction processing applications.

Example: One of the source systems for a sales analysis data warehouse might be an order entry system that records all of the current order activities.

Designing and creating the extraction process is often one of the most time-consuming tasks in the ETL process and, indeed, in the entire data warehousing process. The source systems might be very complex and poorly documented, and thus determining which data needs to be extracted can be difficult. The data has to be extracted normally not only once, but several times in a periodic manner to supply all changed data to the data warehouse and keep it up-to-date. Moreover, the source system typically cannot be modified, nor can its performance or availability be adjusted, to accommodate the needs of the data warehouse extraction process.

These are important considerations for extraction and ETL in general. This unit, however, focuses on the technical considerations of having different kinds of sources and extraction methods. It assumes that the data warehouse team has already identified the data that will be extracted, and discusses common techniques used for extracting data from source databases.

Extraction Methods in Data Warehouses

The extraction method you should choose is highly dependent on the source system and also from the business needs in the target data warehouse environment. Very often, there is no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems. Sometimes even the customer is not allowed to add anything to an out-of-the-box application system.

The estimated amount of the data to be extracted and the stage in the ETL process (initial load or maintenance of data) may also impact the decision of how to extract, from a logical and a physical perspective. Basically, you have to decide how to extract data logically and physically.

Logical Extraction Methods

There are two types of logical extraction:

1. ***Full Extraction:*** The data is extracted completely from the source system. Because this extraction reflects all the data currently available on the source system, there's no need to keep track of changes to the data source since the last successful extraction. The source data will be provided as-is and no additional logical information (for example, timestamps) is

necessary on the source site. An example for a full extraction may be an export file of a distinct table or a remote SQL statement scanning the complete source table.

2. ***Incremental Extraction:*** At a specific point in time, only the data that has changed since a well-defined event back in history will be extracted. This event may be the last time of extraction or a more complex business event like the last booking day of a fiscal period. To identify this delta change there must be a possibility to identify all the changed information since this specific time event. This information can be either provided by the source data itself such as an application column, reflecting the last-changed timestamp or a change table where an appropriate additional mechanism keeps track of the changes besides the originating transactions. In most cases, using the latter method means adding extraction logic to the source system.

Many data warehouses do not use any change-capture techniques as part of the extraction process. Instead, entire tables from the source systems are extracted to the data warehouse or staging area, and these tables are compared with a previous extract from the source system to identify the changed data. This approach may not have significant impact on the source systems, but it clearly can place a considerable burden on the data warehouse processes, particularly if the data volumes are large.

Physical Extraction Methods

Depending on the chosen logical extraction method and the capabilities and restrictions on the source side, the extracted data can be physically extracted by two mechanisms. The data can either be extracted online from the source system or from an offline structure. Such an offline structure might already exist or it might be generated by an extraction routine.

There are the following methods of physical extraction:

1. ***Online Extraction:*** The data is extracted directly from the source system itself. The extraction process can connect directly to the source system to access the source tables themselves or to an intermediate system that stores the data in a preconfigured manner (for example, snapshot logs or change tables).

Note The intermediate system is not necessarily physically different from the source system.

With online extractions, you need to consider whether the distributed transactions are using original source objects or prepared source objects.

2. ***Offline Extraction:*** The data is not extracted directly from the source system but is staged explicitly outside the original source system. The data already has an existing structure (for example, redo logs, archive logs or transportable tablespaces) or was created by an extraction routine.

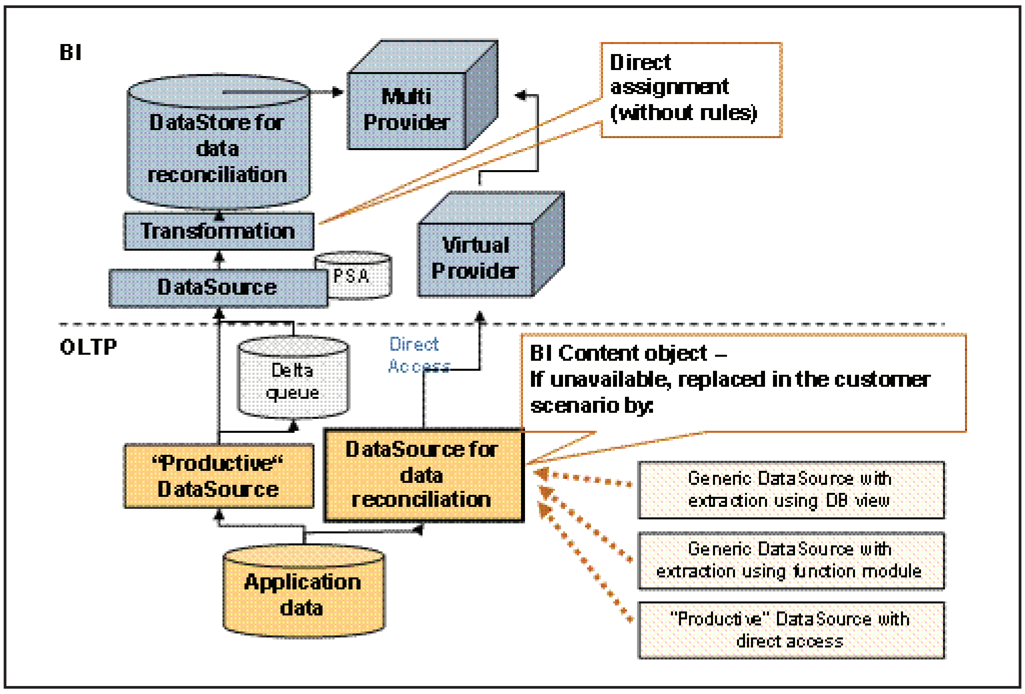
5.2 Data Reconciliation

An important aspect in ensuring the quality of data in business intelligence is the consistency of the data. As a data warehouse, business intelligence integrates and transforms data and stores it so that it is made available for analysis and interpretation. The consistency of the data between the various process steps has to be ensured. Data reconciliation for DataSources allows you to ensure the consistency of data that has been loaded into business intelligence and is available and used productively there. You use the scenarios that are described below to validate the loaded data. Data reconciliation is based on a comparison of the data loaded into business intelligence and the application data in the source system. You can access the data in the source system directly to perform this comparison.

The term productive DataSource is used for DataSources that are used for data transfer in the productive operation of business intelligence. The term data reconciliation DataSource is used for DataSources that are used as a reference for accessing the application data in the source directly and therefore allow you to draw comparisons to the source data.

5.2.1 Model

Figure below shows the data model for reconciling application data and loaded data in the data flow with transformation. The data model can also be based on 3.x objects (data flow with transfer rules).


BI
DataStore for data reconciliation
Multi Provider
Direct assignment (without rules)
Transformation
PSA
Virtual Provider
DataSource
OLTP
Delta queue
Direct Access
BI Content object – If unavailable, replaced in the customer scenario by:
'Productive' DataSource
DataSource for data reconciliation
Generic DataSource with extraction using DB view
Generic DataSource with extraction using function module
"Productive" DataSource with direct access
Application data

The productive DataSource uses data transfer to deliver the data that is to be validated to BI. The transformation connects the DataSource fields with the InfoObject of a DataStore object that has been created for data reconciliation, by means of a direct assignment. The data reconciliation DataSource allows a VirtualProvider direct access to the application data. In a MultiProvider, the data from the DataStore object is combined with the data that has been read directly. In a query that is defined on the basis of a MultiProvider, the loaded data can be compared with the application data in the source system.

In order to automate data reconciliation, we recommend that you define exceptions in the query that proactively signal that differences exist between the productive data in BI and the reconciliation data in the source. You can use information broadcasting to distribute the results of data reconciliation by e-mail.

5.2.2 Modelling Aspects

Data reconciliation for DataSources allows you to check the integrity of the loaded data by, for example, comparing the totals of a key figure in the DataStore object with the corresponding totals that the VirtualProvider accesses directly in the source system.

In addition, you can use the extractor or extractor error interpretation to identify potential errors in the data processing. This function is available if the data reconciliation DataSource uses a different extraction module to the productive DataSource.

We recommend that you keep the volume of data transferred as small as possible because the data reconciliation DataSource accesses the data in the source system directly. This is best performed using a data reconciliation DataSource delivered by business intelligence Content or a generic DataSource using function modules because this allows you to implement an aggregation logic. For mass data, you generally need to aggregate the data or make appropriate selections during extraction.

The data reconciliation DataSource has to provide selection fields that allow the same set of data to be extracted as the productive DataSource.

Task "The source systems for a data warehouse are typically transaction processing applications." Discuss

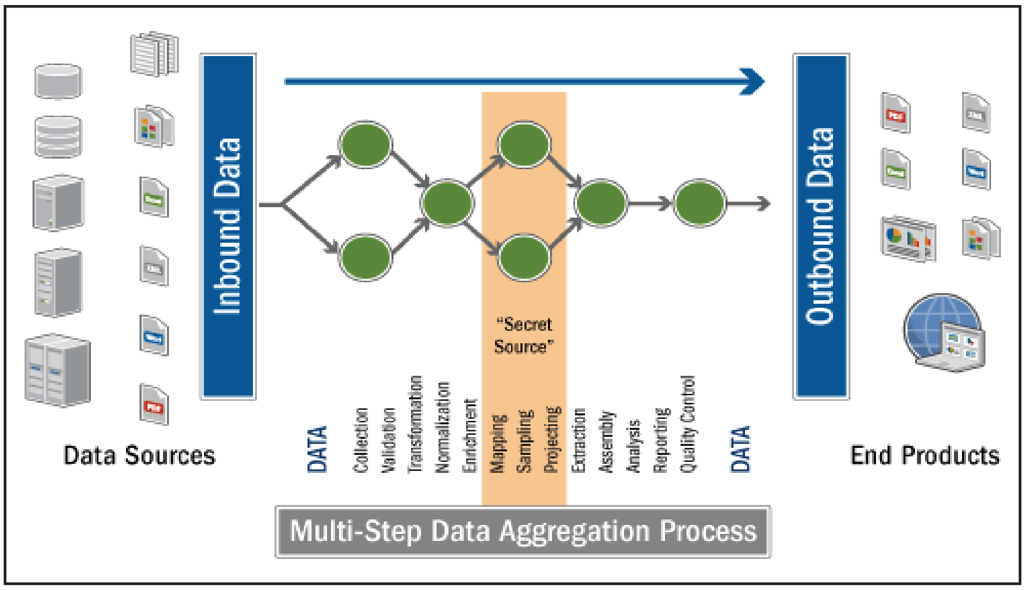
5.3 Data Aggregation and Customization

Data aggregation is any process in which information is gathered and expressed in a summary form, for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as age, profession, or income. The information about such groups can then be used for Web site personalization to choose content and advertising likely to appeal to an individual belonging to one or more groups for which data has been collected.

Example: A site that sells music CDs might advertise certain CDs based on the age of the user and the data aggregate for their age group. Online analytic processing (OLAP) is a simple type of data aggregation in which the marketer uses an online reporting mechanism to process the information.

Data aggregation can be user-based: personal data aggregation services offer the user a single point for collection of their personal information from other Web sites. The customer uses a single master personal identification number (PIN) to give them access to their various accounts (such as those for financial institutions, airlines, book and music clubs, and so on). Performing this type of data aggregation is sometimes referred to as "screen scraping."

As a data aggregator, data is your business not a byproduct of your business. You buy data, transform it, scrub it, cleanse it, standardize it, match it, validate it, analyze it, statistically project it, and sell it. You need a rock-solid data aggregation solution as the foundation of your operations.



The ideal data aggregation solution should:

1. Fully automate key steps of the data aggregation process so that your IT team can boost productivity, accelerate delivery times, and dramatically reduce costs. That automation lets your company focus on what it does best—your proprietary data sampling, data projecting, and data assembling core competencies that distinguish you from your competitors

2. Transform any data, in any format, from any source so your company can create new products for new markets

3. Detect data quality problems early so your company can synthesize high-quality marketing research, point-of-sale analysis, and other data products more quickly and at a lower cost.

5.4 Query Optimization

For any production database, SQL query performance becomes an issue sooner or later. Having long-running queries not only consumes system resources that makes the server and application run slowly, but also may lead to table locking and data corruption issues. So, query optimization becomes an important task.

Basic principles of query optimization are:

1. ***Understand how your database is executing your query:*** Nowadays all databases have their own query optimizer, and offers a way for users to understand how a query is executed. For example, which index from which table is being used to execute the query? The first step to query optimization is understanding what the database is doing. Different databases have different commands for this. For example, in MySQL, one can use "EXPLAIN [SQL Query]" keyword to see the query plan. In Oracle, one can use "EXPLAIN PLAN FOR [SQL Query]" to see the query plan.

2. ***Retrieve as little data as possible:*** The more data returned from the query, the more resources the database needs to expand to process and store these data. So for example, if you only need to retrieve one column from a table, do not use 'SELECT *'.

Notes

3. ***Store intermediate results:*** Sometimes logic for a query can be quite complex. Often, it is possible to achieve the desired result through the use of subqueries, inline views, and UNION-type statements. For those cases, the intermediate results are not stored in the database, but are immediately used within the query. This can lead to performance issues, especially when the intermediate results have a large number of rows.

The way to increase query performance in those cases is to store the intermediate results in a temporary table, and break up the initial SQL statement into several SQL statements. In many cases, you can even build an index on the temporary table to speed up the query performance even more. Granted, this adds a little complexity in query management (i.e., the need to manage temporary tables), but the speedup in query performance is often worth the trouble.

Specific query optimization strategies:

1. ***Use Index:*** Using an index is the first strategy one should use to speed up a query. In fact, this strategy is so important that index optimization is also discussed.
2. ***Aggregate Table:*** Pre-populating tables at higher levels so less amount of data need to be parsed.
3. ***Vertical Partitioning:*** Partition the table by columns. This strategy decreases the amount of data a SQL query needs to process.
4. ***Horizontal Partitioning:*** Partition the table by data value, most often time. This strategy decreases the amount of data a SQL query needs to process.
5. ***De-normalization:*** The process of de-normalization combines multiple tables into a single table. This speeds up query performance because fewer table joins are needed.
6. ***Server Tuning:*** Each server has its own parameters and often tuning server parameters so that it can fully take advantage of the hardware resources can significantly speed up query performance.

5.5 Update Propagation

Data Propagation is the distribution of data from one or more source data warehouses to one or more local access databases, according to propagation rules. Data warehouses need to manage big bulks of data every day. A data warehouse may start with a few data, and starts to grow day by day by constant sharing and receiving from various data sources.

As data sharing continues, data warehouse management becomes a big issue. Database administrators need to manage the corporate data more efficiently and in different subsets, groupings and time frames. As a company grows further, it may implement more and more data sources especially if the company expansions goes outside its current geographical location.

Data warehouses, data marts and operational data stores are becoming indispensable tools in today’s businesses. These data resources need to be constantly updated and the process of updating involves moving large volumes of data from one system to another and forth and back to a business intelligence system. It is common for data movement of high volumes to be performed in batches within a brief period without sacrificing performance of availability of operation applications or data from the warehouse.

The higher the volume of data to be moved, the more challenging and complex the process becomes. As such, it becomes the responsibility of the data warehouse administrator to find means of moving bulk data more quickly and identifying and moving only the data which has changed since the last data warehouse update.

From these challenges, several new data propagation methods have been developed in business enterprises resulting in data warehouses and operational data stores evolving into mission-critical, real-time decision support systems. Below are some of the most common technological methods developed to address the problems related to data sharing through data propagation.

Bulk Extract: In this method of data propagation, copy management tools or unload utilities are being used in order to extract all or a subset of the operational relational database. Typically, the extracted data is the transported to the target database using file transfer protocol (FTP) any other similar methods. The data which has been extracted may be transformed to the format used by the target on the host or target server.

The database management system load products are then used in order to refresh the database target. This process is most efficient for use with small source files or files that have a high percentage of changes because this approach does not distinguish changed versus unchanged records. Apparently, it is least efficient for large files where only a few records have changed.

File Compare: This method is a variation of the bulk move approach. This process compares the newly extracted operational data to the previous version. After that, a set of incremental change records is created. The processing of incremental changes is similar to the techniques used in bulk extract except that the incremental changes are applied as updates to the target server within the scheduled process. This approach is recommended for smaller files where there only few record changes.

Change Data Propagation: This method captures and records the changes to the file as part of the application change process. There are many techniques that can be used to implement Change Data Propagation such as triggers, log exits, log post processing or DBMS extensions. A file of incremental changes is created to contain the captured changes. After completion of the source transaction, the change records may already be transformed and moved to the target database. This type of data propagation is sometimes called near real time or continuous propagation and used in keeping the target database synchronized within a very brief period of a source system.

5.6 Modelling and Measuring Data Warehouse Quality

Data quality has been defined as the fraction of performance over expectancy, or as the loss imparted to society from the time a product is shipped. We believe, though, that the best definition is the one found in: data quality is defined as "fitness for use". The nature of this definition directly implies that the concept of data quality is relative. For example, data semantics (the interpretation of information) is different for each distinct user. As mentions "the problem of data quality is fundamentally intertwined in how [...] users actually use the data in the system", since the users are actually the ultimate judges of the quality of the data produced for them: if nobody actually uses the data, then nobody will ever take care to improve its quality.

As a decision support information system, a data warehouse must provide high level quality of data and quality of service. Coherency, freshness, accuracy, accessibility, availability and performance are among the quality features required by the end users of the data warehouse. Still, too many stakeholders are involved in the lifecycle of the data warehouse; all of them seem to have their quality requirements. As already mentioned, the Decision Maker usually employs an OLAP query tool to get answers interesting to him. A decision-maker is usually concerned with the quality of the stored data, their timeliness and the ease of querying them through the OLAP tools. The Data Warehouse Administrator needs facilities such as error reporting, metadata accessibility and knowledge of the timeliness of the data, in order to detect changes and reasons for them, or problems in the stored information. The Data Warehouse Designer needs to measure the quality of the schemata of the data warehouse environment (both existing and newly produced) and the quality of the metadata as well. Furthermore, he needs software evaluation standards to test the software packages he considers purchasing. The Programmers of Data Warehouse Components can make good use of software implementation standards in order to accomplish and evaluate their work. Metadata reporting can also facilitate their job, since they can avoid mistakes related to schema information. Based on this analysis, we can safely argue that different roles imply a different collection of quality aspects, which should be ideally treated in a consistent and meaningful way.

From the previous it follows that, on one hand, the quality of data is of highly subjective nature and should ideally be treated differently for each user. At the same time, the quality goals of the involved stakeholders are highly diverse in nature. They can be neither assessed nor achieved directly but require complex measurement, prediction, and design techniques, often in the form of an interactive process. On the other hand, the reasons for data deficiencies, non-availability or reachability problems are definitely objective, and depend mostly on the information system definition and implementation. Furthermore, the prediction of data quality for each user must be based on objective quality factors that are computed and compared to users' expectations. The question that arises, then, is how to organize the design, administration and evolution of the data warehouse in such a way that all the different, and sometimes opposing, quality requirements of the users can be simultaneously satisfied. As the number of users and the complexity of data warehouse systems do not permit to reach total quality for every user, another question is how to prioritize these requirements in order to satisfy them with respect to their importance. This problem is typically illustrated by the physical design of the data warehouse where the problem is to find a set of materialized views that optimize user requests response time and the global data warehouse maintenance cost at the same time.

Before answering these questions, though, it should be useful to make a clear-cut definition of the major concepts in these data warehouse quality management problems. To give an idea of the complexity of the situation let us present a verbal description of the situation. The interpretability of the data and the processes of the data warehouse is heavily dependent on the design process (the level of the description of the data and the processes of the warehouse) and the expressive power of the models and the languages which are used. Both the data and the systems architecture (i.e. where each piece of information resides and what the architecture of the system is) are part of the interpretability dimension. The integration process is related to the interpretability dimension, by trying to produce minimal schemata. Furthermore, processes like query optimization (possibly using semantic information about the kind of the queried data -e.g. temporal, aggregate, etc.), and multidimensional aggregation (e.g. containment of views, which can guide the choice of the appropriate relations to answer a query) are dependent on the interpretability of the data and the processes of the warehouse. The accessibility dimension of quality is dependent on the kind of data sources and the design of the data and the processes of the warehouse. The kind of views stored in the warehouse, the update policy and the querying processes are all influencing the accessibility of the information. Query optimization is related to the accessibility dimension, since the sooner the queries are answered, the higher the transaction availability is. The extraction of data from the sources is also influencing (actually determining) the availability of the data warehouse. Consequently, one of the primary goals of the update propagation policy should be to achieve high availability of the data warehouse (and the sources). The update policies, the evolution of the warehouse (amount of purged information) and the kind of data sources are all influencing the timeliness and consequently the usefulness of data. Furthermore, the timeliness dimension influences the data warehouse design and the querying of the information stored in the warehouse (e.g., the query optimization could possibly take advantage of possible temporal relationships in the data warehouse). The believability of the data in the warehouse is obviously influenced from the believability of the data in the sources. Furthermore, the level of the desired believability influences the design of the views and processes of the warehouse. Consequently, the source integration should take into account the believability of the data, whereas the data warehouse design process should also take into account the believability of the processes. The validation of all the processes of the data warehouse is another issue, related with every task in the data warehouse environment and especially with the design process. Redundant information in the warehouse can be used from the aggregation, customization and query optimization processes in order to obtain information faster. Also, replication issues are related to these tasks.

Finally, quality aspects influence several factors of data warehouse design. For instance, the required storage space can be influenced by the amount and volume of the quality indicators needed (time, believability indicators etc.). Furthermore, problems like the improvement of query optimization through the use of quality indicators (e.g. ameliorate caching), the modeling of incomplete information of the data sources in the data warehouse, the reduction of negative effects schema evolution has on data quality and the extension of data warehouse models and languages, so as to make good use of quality information have to be dealt with.

Models and tools for the management of data warehouse quality can build on substantial previous work in the fields of data quality.

5.6.1 Quality Definition

A definition and quantification of quality is given, as the fraction of Performance over Expectance. Taguchi defined quality as the loss imparted to society from the time a product is shipped. The total loss of society can be viewed as the sum of the producer's loss and the customer's loss. It is well known that there is a tradeoff between the quality of a product or service and a production cost and that an organization must find an equilibrium between these two parameters. If the equilibrium is lost, then the organization loses anyway (either by paying too much money to achieve a certain standard of quality, called "target", or by shipping low quality products of services, which result in bad reputation and loss of market share).

5.6.2 Data Quality Research

Quite a lot of research has been done in the field of data quality. Both researchers and practitioners have faced the problem of enhancing the quality of decision support systems, mainly by ameliorating the quality of their data. In this section we will present the related work on this field, which more or less influenced our approach for data warehouse quality. A detailed presentation can be found in Wang et al., presents a framework of data analysis, based on the ISO 9000 standard. The framework consists of seven elements adapted from the ISO 9000 standard: management responsibilities, operation and assurance cost, research and development, production, distribution, personnel management and legal function. This framework reviews a significant part of the literature on data quality, yet only the research and development aspects of data quality seem to be relevant to the cause of data warehouse quality design. The three main issues involved in this field are: analysis and design of the data quality aspects of data products, design of data manufacturing systems (DMS's) that incorporate data quality aspects and definition of data quality dimensions. We should note, however, that data are treated as products within the proposed framework. The general terminology of the framework regarding quality is as follows: Data quality policy is the overall intention and direction of an organization with respect to issues concerning the quality of data products. Data quality management is the management function that determines and implements the data quality policy. A data quality system encompasses the organizational structure, responsibilities, procedures, processes and resources for implementing data quality management. Data quality control is a set of operational techniques and activities that are used to attain the quality required for a data product. Data quality assurance includes all the planed and systematic actions necessary to provide adequate confidence that a data product will satisfy a given set of quality requirements.

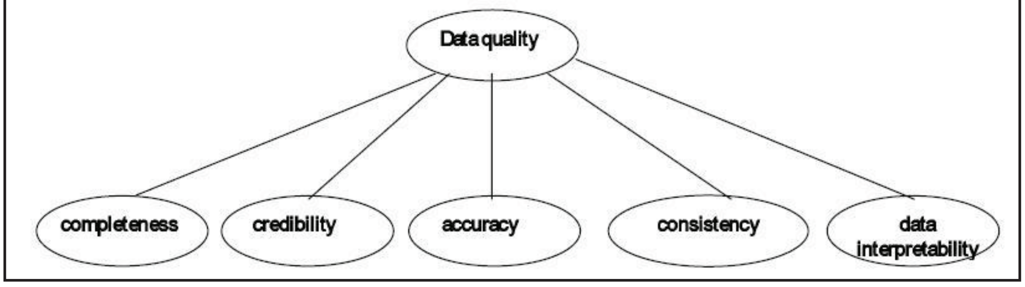
5.6.3 Data Quality

The quality of the data that are stored in the warehouse, is obviously not a process by itself; yet it is influenced by all the processes which take place in the warehouse environment. As already mentioned, there has been quite a lot of research on the field of data quality, in the past. We define data quality as a small subset of the factors proposed in other models. For example, our notion of

Notes

data quality, in its greater part, is treated as a second level factor, namely believability. Yet, in our model, the rest of the factors proposed elsewhere, are treated as process quality factors.

Figure 5.1: Data Quality Factors



The completeness factor describes the percentage of the interesting real-world information entered in the sources and/or the warehouse. For example, completeness could rate the extent to which a string describing an address did actually fit in the size of the attribute which represents the address. The credibility factor describes the credibility of the source that provided the information. The accuracy factor describes the accuracy of the data entry process which happened at the sources. The consistency factor describes the logical coherence of the information with respect to logical rules and constraints. The data interpretability factor is concerned with data description (i.e. data layout for legacy systems and external data, table description for relational databases, primary and foreign keys, aliases, defaults, domains, explanation of coded values, etc.)

Factor	Methods of Measurement	Formulae
Completeness	Performance of statistic checks	The percentage of stored information detected to be incomplete with respect to the real world values
Credibility	Documentation of the source which provided the information	Percentage of inaccurate information provided by each specific source
Accuracy	Documentation of the person or machine which entered the information and performance of statistical checks	The percentage of stored information detected to be inaccurate with respect to the real world values, due to data entry reasons
Consistency	Performance of statistic checks	The percentage of stored information detected to be inconsistent
Data Interpretability	Data description (i.e. data layout for legacy systems and external data, table description for relational databases, primary and foreign keys, aliases, defaults, domains, explanation of coded values, etc.)	Number of pieces of information not fully described

5.7 Major Research Project in Data Warehousing

Major research project in data warehousing are:

1. Michael Hahne, Lothar Burow, and Torben Elvers, XML-Datenimport in das SAP Business Information Warehouse bei Bayer MaterialScience, Auf dem Weg zur Integration Factory, 231-251, 2005.
2. H. Fan and A. Poulovassilis, Using AutoMed Metadata in Data Warehousing Environments, DOLAP, 2003.
3. Michael Hahne, Transformation mehrdimensionaler Datenmodelle, Vom Data Warehouse zum Corporate Knowledge Center, 399-420, 2002.

Notes

4. D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati, Source Integration in Data Warehousing, DEXA'98, 192—197, 1998.

5. G. Zhou, R. Hull, R. King, and J.-C. Franchitti, Supporting Data Integration and Warehousing Using H2O, IEEE Data Engineering Bulletin, 18:2, 29-40, 1995.

6. Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh, Finding an application-appropriate model for XML data warehouses, Information Systems, 35:6, 662—687, 2010.

7. J. Gray, H. Schek, M. Stonebraker, and J. Ullman, Lowell Report, 2003.

8. Stefano Rizzi, Open problems in data warehousing: eight years later, DMDW, 2003 (Keynote slides).

9. Eva Kuhn, The Zero-Delay Data Warehouse: Mobilizing Heterogeneous Databases, VLDB, 2003.

10. Rob Weir, Taoxin Peng, and Jon Kerridge, Best Practice for Implementing a Data Warehouse: A Review for Strategic Alignment, DMDW, 2003.

11. Surajit Chaudhuri, Umeshwar Dayal, and Venkatesh Ganti, Database technology for decision support systems, IEEE Computer, 48—55, 2001.

12. Surajit Chaudhuri and Umesh Dayal, An Overview of Data Warehousing and OLAP Technology, ACM SIGMOD Record, 26:1, 1997.

13. M.C. Wu and A.P. Buchmann, Research Issues in Data Warehousing, BTW, 1997.

14. Phillip M. Fernandez and Donovan Schneider, The Ins and Outs (and everything in between) of Data Warehousing, ACM SIGMOD, 1996 (SIGMOD membership required).

15. Jennifer Widom, Research Problems in Data Warehousing, Int'l Conf. on Information and Knowledge Management, 1995.

16. P. Raj, Database Design Articles, Overviews, and Resources.

17. Data Warehousing Knowledge Center.

18. Larry Greenfield, Data Warehousing Information Center.

19. R. Kimball, Various articles from Intelligent Enterprise magazine.

20. Data Warehousing Online.

Task Online analytic processing (OLAP) is a simple type of data aggregation in which the marketer uses an online reporting mechanism to process the information. Discuss how OLAP used.

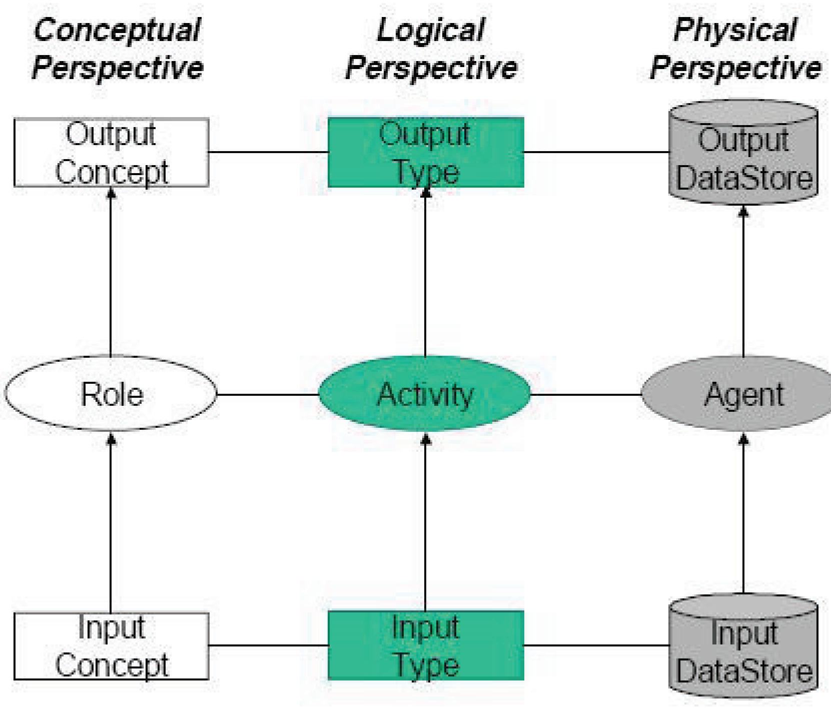
5.8 Three Perspectives of Data Warehouse Metadata

In Figure 5.2 we offer an intuitive view for the categorization of the entities in the process metamodel. The model has three different perspectives covering distinct aspects of a process: the conceptual, logical and physical perspective. The categorization is following the separation of the framework proposed, and fits naturally with the adopted architecture model, since the perspectives of the process model operate on objects of the respective perspective of the architecture model. As mentioned there are different ways to view a process: what steps it consists of (logical perspective), how they are to be performed (physical perspective) and why these steps exist (conceptual perspective). Thus, we view a data warehouse process from three perspectives: a central logical part of the model, which captures the basic structure of a process, its physical

Notes

counterpart which provides specific details over the actual components that execute the activity and the conceptual perspective which abstractly represents the basic interrelationships between data warehouse stakeholders and processes in a formal way.

Figure 5.2: The Reasoning behind the 3 Perspectives of the Process Metamodel



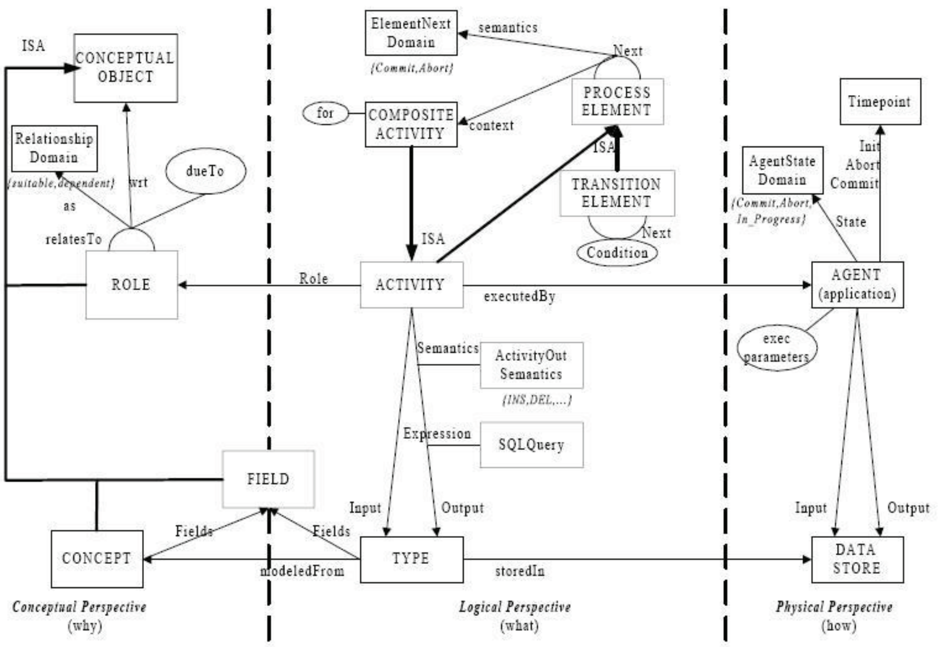
Typically, the information about how a process is executed concerns stakeholders who are involved in the everyday use of the process. The information about the structure of the process concern stakeholders that manage it and the information relevant to the reasons behind this structure concern process engineers who are involved in the monitoring or evolution of the process environment. In the case of data warehouses it is expected that all these roles are covered by the data warehouse administration team, although one could also envision different schemes. Another important issue shown in Figure 5.2 is that there is also a data flow at each of the perspectives: a type description of the incoming and outcoming data at the logical level, where the process acts as an input/output function, a physical description of the details of the physical execution for the data involved in the activity and a relationship to the conceptual entities related to these data, connected through a corresponding role.

Once again, we have implemented the process metamodel in the Telos language and specifically in the ConceptBase metadata repository. The implementation of the process metamodel in ConceptBase is straightforward, thus we choose to follow an informal, bird's-eye view of the model, for reasons of presentation and lack of space. Wherever definitions, constraints, or queries of ConceptBase are used in the chapter, they will be explained properly in natural language, too.

In the sequel, when we present the entities of the metamodel, this will be done in the context of a specific perspective. In the Telos implementation of the metamodel, this is captured by specializing the generic classes ConceptualObject, LogicalObject and PhysicalObject with ISA relationships, accordingly. We will start the presentation of the metamodel from the logical perspective. First, we will show how it deals with the requirements of structure complexity and capturing of data semantics in the next two sections. In the former, the requirement of trace logging will be fulfilled too. The full metamodel is presented in Figure 5.3.

Notes

Figure 5.3: The Data Warehouse Operational Process Metamodel



Case Study

Data Warehousing System for a Logistics Regulatory Body

Business Challenge

The customer, the sole regulatory body for ports and maritime affairs of a leading republic in the Asia Pacific, protects the maritime and ports interests of the republic in the international arena by regulating and licensing port and marine services and facilities. It also manages vessel traffic in the port, ensures navigational safety and port/maritime security, and a clean marine environment.

In the wake of increasing vessel traffic through the port and booming number of users, the customer required a system that would provide an integrated framework in which end-users could perform online queries, prepare ad hoc reports and analyze data with user-friendly tools.

Mahindra Satyam Role

Mahindra Satyam identified Data Warehouse (DW) as a potential and essential tool for high-level analysis and planning, and for faster and easier information management. The role involved developing a data warehouse (DW) application that has the capability of comparative or variance analysis, trend analysis, slice and dice, ranking, drill down, visualization on table, alerts and job scheduling etc.

Business Benefits

1. Helped the customer promote the nation as an important hub port and an International maritime centre
2. Facilitated easier monitoring of the increased vessel movement through the port

Contd...

3. Enabled better understanding of customer behavior and customer profiles
4. Helped customer increase its market share and understand competition better

5.9 Summary

- In the process of extracting data from one source and then transforming the data and loading it to the next layer, the whole nature of the data can change considerably.
- It might also happen that some information is lost while transforming the data. A reconciliation process helps to identify such loss of information.
- One of the major reasons of information loss is loading failures or errors during loading.
- Data reconciliation is often confused with the process of data quality testing. Even worse, sometimes data reconciliation process is used to investigate and pin point the data issues.
- While data reconciliation may be a part of data quality assurance, these two things are not necessarily same.
- Scope of data reconciliation should be limited to identify, if at all, there is any issue in the data or not.
- The scope should not be extended to automate the process of data investigation and pin pointing the issues.

5.10 Keywords

Data Aggregation: Data aggregation is any process in which information is gathered and expressed in a summary form, for purposes such as statistical analysis.

Data Extraction: Extraction is the operation of extracting data from a source system for further use in a data warehouse environment.

Data Quality: Data quality has been defined as the fraction of performance over expectancy, or as the loss imparted to society from the time a product is shipped.

Update Propagation: Data Propagation is the distribution of data from one or more source data warehouses to one or more local access databases, according to propagation rules.

5.11 Self Assessment

Choose the appropriate answers:

1. SQL stands for:
   (a) Structured Query Language
   (b) Structured Query League
   (c) Systematic Query Language
   (d) Structured Queue Language
2. PIN stands for:
   (a) Permanent Identification Number
   (b) Permanent Index Number
   (c) Personal Identification Number
   (d) Personal Index Number

Fill in blanks:

3. The source systems for a .... are typically transaction processing applications.
4. The data is extracted completely from the source system known as ....
5. .... is based on a comparison of the data loaded into business intelligence and the application data in the source system.
6. Data reconciliation for DataSources allows you to check the .... of the loaded data.
7. The .... of the data and the processes of the data warehouse is heavily dependent on the design process.
8. .... is related to the accessibility dimension, since the sooner the queries are answered.
9. .... defined quality as the loss imparted to society from the time a product is shipped.
10. Three perspective of data warehouse metadata are logical, physical and ....

5.12 Review Questions

1. What do you mean by data extraction?
2. Describe logical extraction methods of data.
3. Distinguish between online and offline extraction.
4. Explain modeling aspects of data reconciliation.
5. What do you mean by data aggregation?
6. Describe basic principles of query optimization.
7. Explain various strategies of query optimization.
8. "As a decision support information system, a data warehouse must provide high level quality of data and quality of service". Explain
9. Write a short note on data quality research.
10. What are the three perspectives of data warehouse metadata?

Answers: Self Assessment

1. (a)	2. (c)
3. data warehouse	4. Full extraction
5. Data reconciliation	6. integrity
7. interpretability	8. Query optimization
9. Taguchi	10. Conceptual

Notes

5.13 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 6: Source Integration

Notes

CONTENTS

Notes

Objectives

After studying this unit, you will be able to:

- Know enterprise application integration
- Describe practice of source integration
- Explain research in source integration
- Describe methodologies of source integration

Introduction

Enterprise Application Integration deals with approaches and mechanisms to integrate multiple applications, and deliver processes, and each business process supports a specific business use case. And, each business process includes multiple sub processes, which also perform some related tasks for achieving the overall goal of the business process. In addition, as each business process works across multiple functional areas (which are typically realized using multiple applications), there is also a need for performing specific business activities that would support the overall business process in the context of the functional area.

6.1 Enterprise Application Integration

EAI solutions provide an integrated approach to connecting the different components of IT infrastructure- people, applications, platforms and databases to enable secure, intra and inter enterprise collaboration. EAI solutions enable an organization to integrate business processes internally and externally with business partners to create dynamic environments that support current and evolving business requirements, thereby creating a global organization.

EAI assists in unrestricted sharing of data and business processes among any connected applications or data sources in the enterprise without making major changes to the applications or data structures. EAI integrates multiple, independently developed applications using incompatible technologies into a single enterprise wide system with information flowing seamlessly.

6.2 Reasons for Emergence of EAI

The reasons for emergence of EAI need are varied. Efforts by the leading Enterprise Business Application Suppliers seeking to establish themselves as the primary provider of the business and the IT backbone that supports the enterprise's operations. But the core driving forces behind EAI is of following categories, which are within themselves inter-related.

Mergers & Acquisitions

Mergers & Acquisitions to be successful require overnight integration of dissimilar business processes of two or more companies, so that they can work as a single corporation. EAI is the only solution, which will enable such a rapid integration.

e-business

e-business requires connecting of customers, suppliers and partners across the world, so as to form an integrated value and supply chain over the Internet.

Industry Regulation & De-regulation

Opening up of business processes to share information and allow market access requires information to flow transparently and seamlessly both externally and internally.

Business Process Automation

Business Process Automation requires new products and services to be integrated with already existent applications so as to improve efficiency, operating costs and customer services across an organization.

Growth in Implementation of ERP Packages

ERP vendors are coming up with a product line complete with interfaces/ adapters to assist the ERP solution to be integrated with other applications as they have realized that ERP solutions to be effective should be integrated with the back end legacy applications.

Supply Chain Management & Customer Relationship Management

There is a movement towards virtual enterprise linking application systems from various companies in the supply chain. Significant developments in peer-to-peer networking and distributed processing have made it possible for businesses to integrate better their own functional departments as well as integrate with their partners and suppliers for better SCM & CRM. Re-engineering of business processes by organizations for greater customer focus requires close cooperation between standalone applications.

Zero Latency Enterprise

Zero latency enterprise refers to an organization that can change its business rules in real time to act on new market opportunities and customer demands. An enterprise application integration solution accelerates responses and facilitates business changes in the zero latency enterprise.

Reduction of Business Process Life Cycle

In the today's competitive business environment the need to align business systems with business goals is all the more a reality. Business processes evolve continuously requiring new methods and data, which in turn require integration with the existing ones. These new applications should start operations quickly moving IT management to shorter application lifecycles. This is made possible because of EAI solutions, which help in integrating different applications and also assist in changing the business rules as required in minimum amount of time.

Intranet/Internet Explosion

The Intranet/Internet explosion is leading to surge in the demand for a new class of human active applications that require integration with back end legacy applications. This feature again is enabled by EAI solution which can integrated the front end and back end applications.

6.3 Advantages of Implementing EAI

The advantages of EAI are:

1. Assists in Supply Chain Management and has the ability to adapt to business changes like Mergers and Acquisitions as it unifies/ integrates applications in no time.
2. Presents user applications with an unified view of information for better decision making thereby achieving cross system consistency.

3. Assists in formation of Zero Latency Enterprise - when all functions within the organization work with the same up-to-data information, latency between applications is eliminated/ reduced.
4. Updating and integrating of applications is possible whenever required. New applications can be created by integrating real time data from different parts of the enterprise.
5. Assists in rapid business process change.
6. Enables creation of virtual corporations with virtual supply chains and operations through sharing of data beyond the organization.
7. Makes possible for legacy or proprietary systems to function on web.
8. Enhancements to standard applications can be made rapidly.

6.4 EAI Functioning

The EAI solution works at both data level and business process level and assists in sharing data of different applications. This sharing of data involves different types business process depending on the type of data sharing involved.

The various logical steps for data sharing are as given below:

1. Unload raw data from source database
2. Validate raw data against source business model
3. Transform source business model data into target business data
4. Validate business data against target business model
5. Load data into Target database

The various integration processes are as follows:

1. Data to data
2. Business model to business model
3. Business model to data model
4. Data model to business model

6.5 Source Integration

A clear trend is the movement away from information-oriented to service-based integration. Information-oriented integration provides an inexpensive mechanism to integrate applications because, in most instances, there is no need to change the applications.

While information-oriented integration provides a functional solution for many application integration problem domains, it is the integration of both application services and application methods that generally provides more value in the long run.

Example: A trading community looking to automate the processing of ordering raw materials may find that simply sharing information (order goes out, and confirmation comes in) is just fine to solve their integration problem. However, in another trading community, there may be a need to access remote services, such as the calculation of duty for intercountry trades. Again, you have to leverage the right approach for the business problem you are looking to solve.

Service-based application integration is not a new approach. We've been looking for mechanisms to bind applications together at the service level for years, including frameworks, transactions, and distributed objects all in wide use today. However, the new notion of Web services, such as Microsoft's .NET strategy, is picking up steam as we attempt to identify a new mechanism that's better able to leverage the power of the Internet to provide access to remote application services through a well-defined interface and directory service: Universal Description, Discovery and Integration (UDDI).

The uses for this type of integration are endless, including the creation of composite applications, or applications that aggregate the processes and information of many applications. For example, using this paradigm, application developers simply need to create the interface and add the application services by binding the interface to as many Internet-connected application services as are required.

The downside, at least with service-based integration, is that this makes it necessary to change the source and target applications or, worse in a number of instances, to create a new application (a composite application). This has the effect of adding cost to the application integration project and is the reason many choose to stay at the information level.

Still, the upside of this approach is that it is consistent with the "baby step" approach most enterprises find comfortable when implementing solutions to integration problems. Service-based solutions tend to be created in a series of small, lower-risk steps. This type of implementation can be successful from the department to the enterprise to the trading community, but never the other way around from the trading community to the department.

6.6 The Practice of Source Integration

Service-oriented Application Integration

Service-oriented integration (SOI) is defined as integrating computing entities using only service interactions in a service-oriented architecture. Service-oriented integration addresses problems with integrating legacy and inflexible heterogeneous systems by enabling IT organizations to offer the functionality locked in existing applications as reusable services.

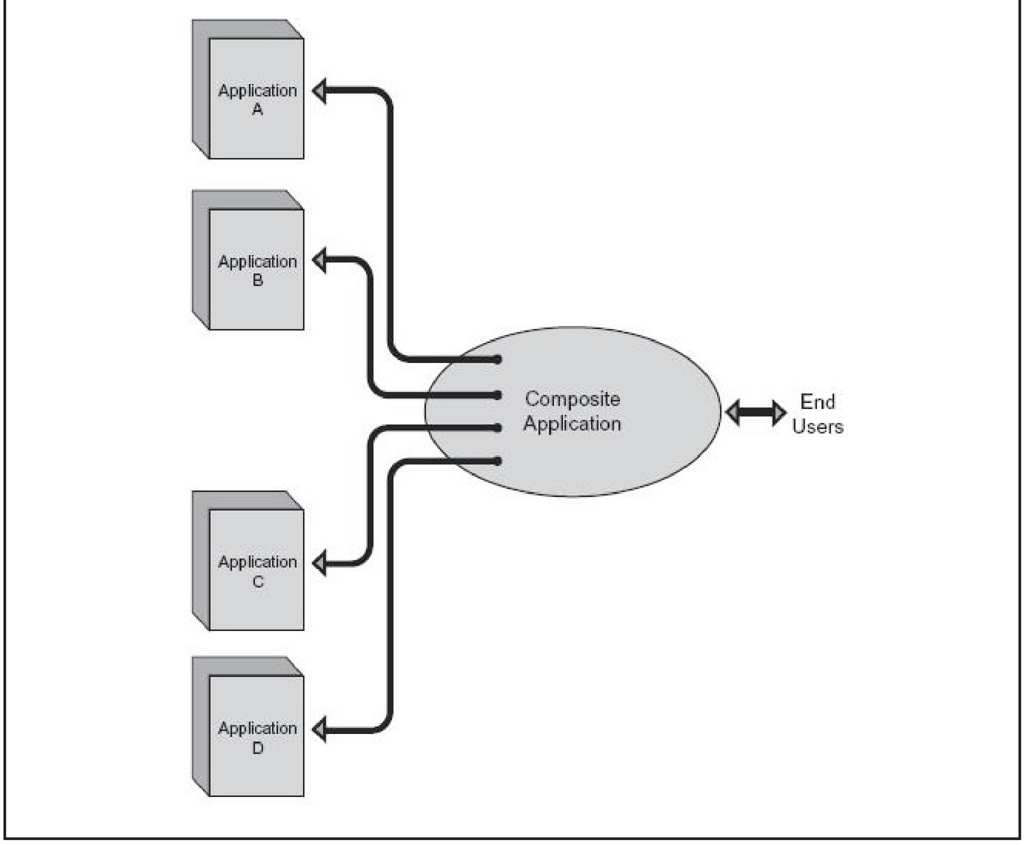
In contrast to traditional enterprise application integration (EAI), the significant characteristics of services-oriented integration are:

1. ***Well-defined, standardized interfaces:*** Consumers are provided with easily understood and consistent access to the underlying service.
2. ***Opaqueness:*** The technology and location of the application providing the functionality is hidden behind the service interface. In fact, there is no need for a fixed services provider.
3. ***Flexibility:*** Both the providers of services and consumers of services can change - the service description is the only constant. Provided both the provider and consumer continue to adhere to the service description, the applications will continue to work.

Service-oriented Application Integration (SOAI) allows applications to share common business logic or methods. This is accomplished either by defining methods that can be shared, and therefore integrated, or by providing the infrastructure for such method sharing such as Web services (Figure 6.1). Methods may be shared either by being hosted on a central server, by accessing them interapplication (e.g., distributed objects), or through standard Web services mechanisms, such as .NET.

Notes

Figure 6.1: Service-oriented Application Integration



Service-oriented Application Integration provides mechanisms to create composite applications, leveraging services found in many remote systems.

Attempts to share common processes have a long history, one that began more than ten years ago with the multi-tiered client/server a set of shared services on a common server that provided the enterprise with the infrastructure for reuse and, now, for integration and the distributed object movement. "Reusability" is a valuable objective. A common set of methods among enterprise applications invites reusability and, as a result, significantly reduces the need for redundant methods and/or applications.

While most methods exist for single-organization use, we are learning that there are times when it makes sense to share between organizations. In a new twist on the longstanding practice of reusability, we are now hoping to expand this sharing beyond intraenterprise to trading partners, as well; for example, sharing a common logic to process credit requests from customers or to calculate shipping costs using a set of Web services.

Unfortunately, absolute reuse has yet to be achieved on the enterprise level. It is an even more distant goal between trading partners. The reasons for this failure are primarily political. They range from internal politics to the inability to select a consistent technology set. In most cases, the actual limit on reuse results directly from a lack of enterprise architecture and central control.

Utilizing the tools and techniques of application integration gives us the opportunity to learn how to share common methods. More than that, these tools and techniques create the infrastructure that can make such sharing a reality. By taking advantage of this opportunity, we are integrating applications so that information can be shared, even as we provide the infrastructure for the reuse of business logic.

Sounds great, doesn't it? The downside might give you pause, however. This "great-sounding" application integration solution also confronts us with the most invasive level of application integration, thus the most costly. This is no small matter if you're considering Web services, distributed objects, or transactional frameworks.

While IOAI generally does not require changes to either the source or target applications, SOAI requires that most, if not all, enterprise applications be changed in order to take advantage of the paradigm. Clearly, this downside makes SOAI a tough sell. However, it is applicable in many problem domains. You just need to make sure you leverage SOAI only when you need it.

Changing applications is a very expensive proposition. In addition to changing application logic, there is the need to test, integrate, and redeploy the application within the enterprise a process that often causes costs to spiral upward. This seems to be the case, no matter if you're approaching SOAI with older technologies such as Common Object Request Broker Architecture (CORBA), or new technologies such as .NET, the latest service-based architecture to come down the road.

Before embracing the invasiveness and expense of SOAI, enterprises must clearly understand both its opportunities and its risks. Only then can its value be evaluated objectively. The opportunity to share application services that are common to many applications and therefore making it possible to integrate those applications represents a tremendous benefit. However, that benefit comes with the very real risk that the expense of implementing SOAI will outpace its value.

Task "The EAI solution works at both data level and business process level and assists in sharing data of different applications." Discuss.

6.7 Service-oriented Architecture

Business Needs

IT organization fulfills two major business needs:

1. Automating internal business processes. Here, the focus is not providing the automation to the clients (users), rather it allows the clients to initiate a paperless transaction. Examples are web-based applications for customers and business partners.
2. Providing B2B e-commerce so that business partners can integrate their systems with the systems of their clients.

Importance of EAI

Typically, a business process involves interactions among various organizational units, which translates into a business process automation requiring interactions with the various applications in an organization. The major challenges faced by the IT organization when integrating these applications relate to the integration of different domains, architecture and technologies. These challenges necessitate a well planned EAI strategy and architecture. There are two main forms of EAI. The first one integrates applications within a company (intra-EAI) and serves the first business need. The second form (inter-EAI) relates to B2B integration and serves the second business need.

There are several strategies available for EAI. They are listed below.

1. ***Data Level Integration***: Data are shared or moved among the applications.
2. ***Applications Interface Integration***: One application can share some functionality residing in other applications. It allows sharing application components.

3. ***Business Method Integration:*** One application can share business services provided by the other applications.
4. ***Presentation Integration***: Provides unified view of data to the end user.
5. ***B2B Integration:*** Provides integration of applications residing in two different organizations.

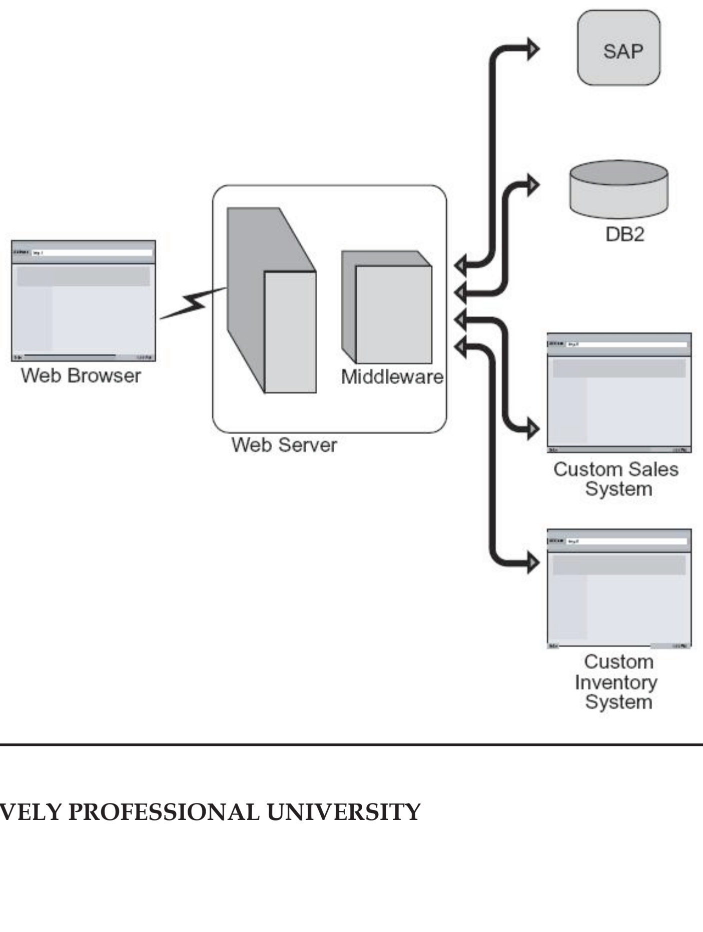
Role of SOA

The best strategy for EAI is Business Method Integration, which allows one application to use the business services provided by the other applications. It makes B2B integration easier, which boils down to the choice of technologies for protocol and transport. A protocol defines the 'language' for the communication and transport carries the messages as per the protocol from one application to another. The service oriented Architecture (SOA) acts like an enabler to Business Method Integration strategy. SOA is the proponent of business driven application architecture, rather than technology driven application architecture, where a business service can be readily mapped to a technology component in an application. Embracing SOA along with supports for multi-protocols (native, SOAP, XML) and multi-transports (TCP/IP, MQ, HTTP) for the service invocations would truly allow us to implement a flexible and extensible EAI architecture.

6.8 Portal-oriented Application Integration

Portal-oriented Application Integration (POAI) allows us to view a multitude of systems both internal enterprise systems and external enterprise systems through a single user interface or application. POAI avoids the back-end integration problem altogether by extending the user interface of each system to a common user interface most often a web browser (Figure 6.2). As a result, POAI integrates all participating systems through the browser, although is does not directly integrate the applications within or between the enterprise.

Figure 6.2: Portal-oriented Application Integration



Portals have become so common and so much has been written about them that we will cover just the basic concepts here. The important point to remember in the context of application integration is that portals have become the primary mechanism by which we accomplish application integration. Whether that is good, bad, or indifferent doesn't really matter. It is simply the way it is. Trading partners have extended the reach of internal enterprise systems by utilizing the familiar web browser interface.

POAI by Example

An example of POAI is an automobile parts supplier that would like to begin selling parts to retail stores (B2B) using a portal. This portal would allow the retail stores to access catalog information, place orders, and track orders over the web, currently the parts supplier leverages SAP as its preferred inventory control system and a common built mainframe application written in COBOL/DB2 serves as its sales order system. Information from each system is required for the B2B portal, and the portal users to update those back-end systems as well.

In order to create a portal, the parts supplier must design the portal application, including the user interface and application behavior, as well as determine which information contained within the back-end systems (SAP and the mainframe) needs to be shared with the portal application. The portal application requires a traditional analysis and design life cycle and a local database. This portal application must be able to control user interaction, capturing and processing errors and controlling the transaction from the user interface all the way to the back-end systems.

Although you can employ many types of enabling technologies when creating portals, most portals are built using application servers. Application servers provide the interface development environments for designing the user interface, a programming environment to define application behavior, and back-end connectors to move information in and out of back-end systems, including SAP and mainframe systems. Although not integrating the application directly the portal externalizes the information to the trading partner in this case, the owner of a retail auto parts store and also update the back-end system, in this case with order placed by the store owner perhaps with the status of existing orders.

Other examples of portals include entire enterprise that are integrated with a single portal application. As many as a dozen companies may use that portal, B2B to purchase goods and services from many companies at the same time. The same type of architecture and enabling technology applies in this case, however, the number of systems integrated with the portal application greatly increases.

Portal Power

The use of portals to integrate enterprises has many advantages. The primary one is that there is no need to integrate back-end systems directly between companies or within enterprises, which eliminates the associated cost or risk. What's more you usually don't have to worry about circumventing firewalls or application-to-application security, because portals typically do nothing more than web-enable existing systems from a single enterprise. With portals, you simple connect to each back-end system through a point of integration and externalize the information into a common user interface. Of course portals themselves are applications and must be designed, built, and tested like any other enterprise application.

Portal-oriented B2B application integration also provides a good facility for Web-enabling existing enterprise systems for any purpose, including B2B and business-to-consumer (B2C) selling over the Web. If you need to move information to a user interface for any reason, this is the best approach.

In many B2B application integration problem domains, the users prefer to interact with the back-end systems through a user interface rather than have the systems automatically exchange information behind the scenes (as in data-oriented B2B application integration). Today more B2B information flows through user interfaces (portal-oriented B2B application integration) than automatically through back-end integration. However, the trend is moving from portals to real-time information exchange. We will eventually remove from the equation the end user, who is the most obvious point of latency when considering portal-oriented B2B application integration.

The advantages of portal-oriented integration are clear:

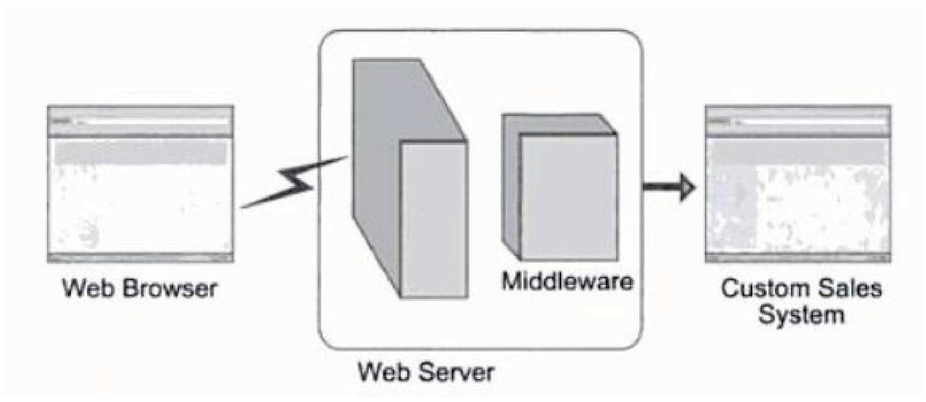
1. It supports a true noninvasive approach, allowing other organizations to interact with a company's internal systems through a controlled interface accessible over the Web.
2. It is typically much faster to implement than real-time information exchange with back-end systems, such as the data-, method-, and application interface–oriented approaches.
3. Its enabling technology is mature, and many examples of portal-oriented B2B application integration exist to learn from.
4. However, there are also disadvantages to portal-level B2B application integration:
5. Information does not flow in real time and thus requires human interaction. As a result, systems do not automatically react to business events within a trading community (such as the depletion of inventory).
6. Information must be abstracted, most typically through another application logic layer (such as an application server). As a result, some portal-oriented solutions actually add complexity to the solution.
7. Security is a significant concern when enterprise and trading community data is being extended to users over the Web.
8. The notion of POAI has gone through many generations, including single system portals, multiple-enterprise-system portals, and now, enterprise portals.

Single-system Portals

Single-system portals, as you might expect are single enterprise systems that have their user interface extended to the web (Figure 6.3).

A number of approaches exist to create a portal for a single enterprise system, including application servers, page servers, and technology for translating simple screens to HTML.

Figure 6.3: Single-system Portal



Notes

Multiple-Enterprise-System Portals

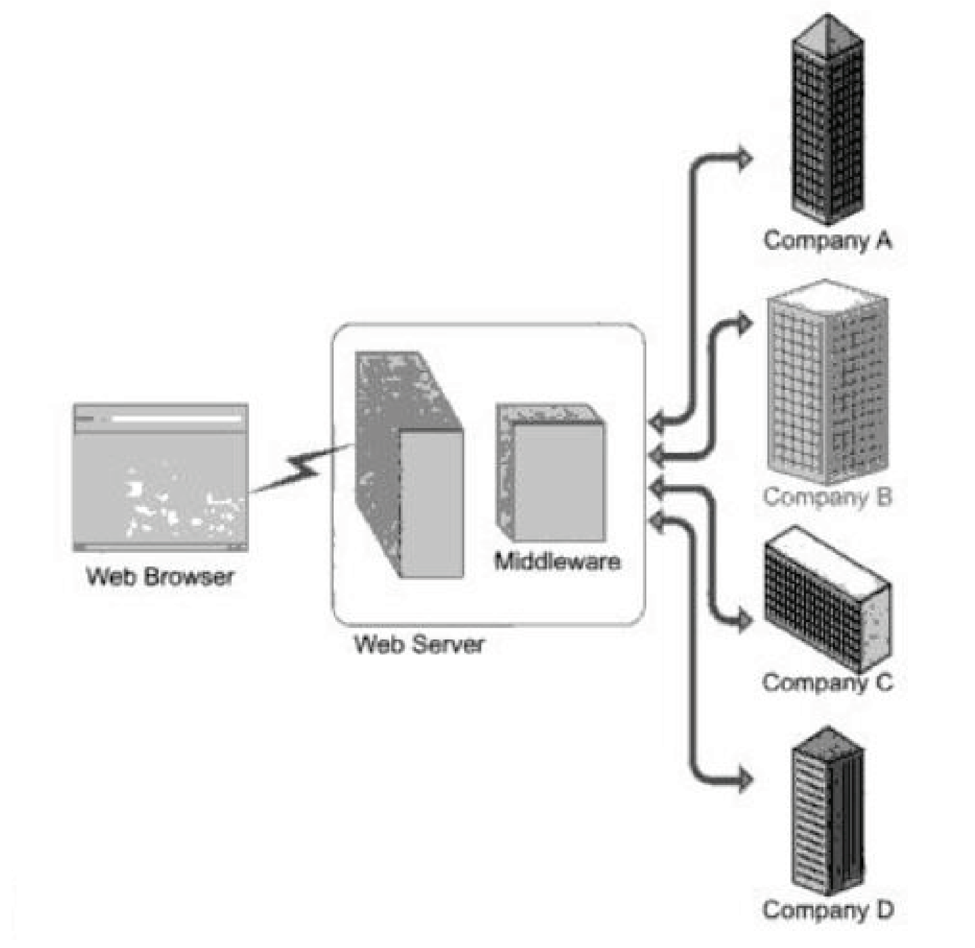
Extending a single-system portal architecture to multiple enterprise systems results in a multiple-enterprise-system portal.

This type of portal represents a classic application server architecture, where information is funneled from several enterprise systems such as SAP R/3, mainframes, PeopleSoft, and inventory systems through a single web-enabled application. Users are able to extract information from these systems and update them through a single web browser interface accessed over an extranet or over the web.

Trading Community

When the multiple-enterprise-system portal is extended to include systems that exist within many companies, the result is an enterprise portal (Figure 6.4).

Figure 6.4: Trading Community Portal



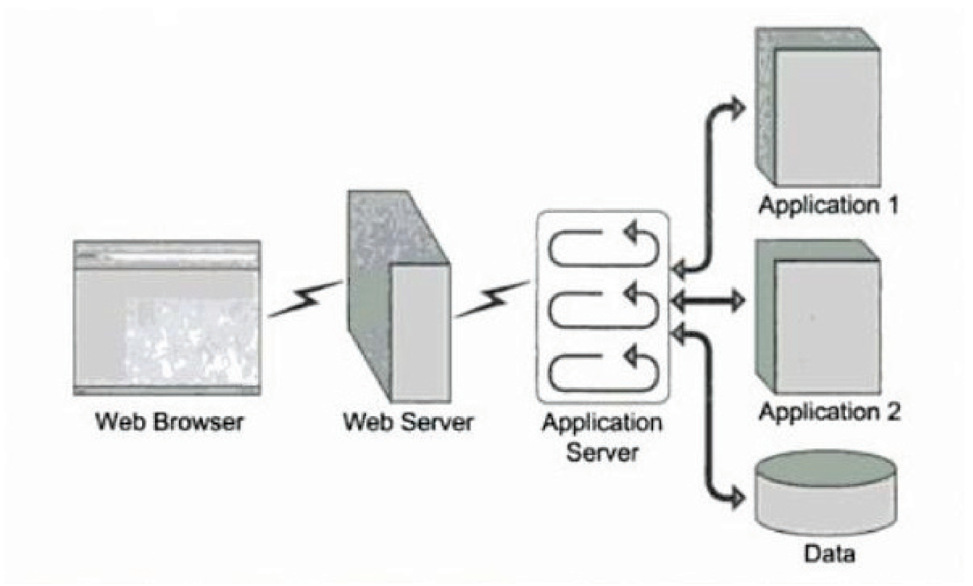
Notes

6.9 Portal Architecture

It should be apparent that portals are really web-enabled application. Given that reality, it might be a good idea to discuss the architecture and components of portals. The following components comprise portal architecture (Figure 6.5).

1. Web clients
2. Web servers
3. Database servers
4. Back-end applications
5. Application servers

Figure 6.5: Portal Architecture and Components



Web Clients

The Web client is a PC or any device that runs a web browser and is capable of displaying HTML and graphics. The web browser makes requests to the web server and processes the files the web server returns. Rather than exchanging message, the web client exchanges entire files. Unfortunately, the process is insufficient and resource intensive. Still, with the web as our preferred common application platform, these drawbacks are also inevitable.

Today, web browsers need not run on PCs. They can also run wireless devices such as personal digital assistants (PDAs) and cellular phones.

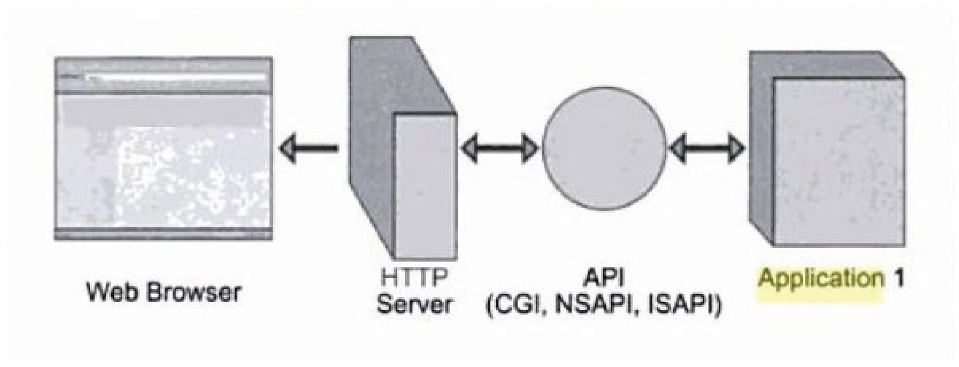
Web Servers

Web servers, at their core, are file servers. Like traditional file servers, they respond to requests from web clients, then send requested file. Web servers are required with portals because the information coming from the application server must be converted into HTML and pumped down to the web browser using HTTP, HTML, graphics, and multimedia files (audio, video and animation) have been traditionally stored on web servers.

Notes

Today's web servers pull double duty. Not only do they serve up file content to hordes of web clients, but they perform rudimentary application processing, as well. With enabling technologies such as Common Gateway Interface (CGI), Netscape Application Programming Interface (NSAPI), and Internet Server Application Programming Interface (ISAPI), web servers can query the web client for information, and then, using web server APIs, they can pass that information to an external process that runs on the web server (Figure 6.6). In many cases, this means users can access information on a database server or on application servers.

Figure 6.6: Using the Web Server API to Customize Information sent to the Client Level



Database Servers

Database servers, when leveraged with portals work just as they do in more traditional client/ server architectures they respond to requests and return information. Sometimes the requests come from web servers that communicate with the database server through a process existing on the web server. Sometimes they come directly from web client communication with the database server via a Call-Level Interface (CLI), such as JDBC for ActiveX.

Back-end Applications

Back-end applications are enterprise applications existing either within a single enterprise or across many enterprises. These are typically a mix of ERP systems, such as SAP R/3 or PeopleSoft, custom applications existing on mainframes, and newer client/server systems. Portals gather the appropriate information from these back-end systems and externalize this information through the user interface.

Although the mechanism employed to gather back-end information varies from technology to technology, typically, portal development environments provide connectors or adaptors to link to various back-end systems, or they provide APIs to allow developers to bind the back-end systems to the portal technology.

Application Servers

Application servers work with portal applications by providing a middle layer between the back-end applications, databases, and the web server. Application servers communicate with both the web server and the resource server using transaction-oriented application development. As with three-tier client/servers, application servers bring load-balancing recovery services and fail-over capabilities to the portal.

6.10 Portals and Application Integration

Portals are nothing new, we've been building them since the web took off and we first understood the value of externalizing existing enterprise data through web interfaces. That was the birth of B2C e-Business, which led the way to B2B e-Business which is after all what POAI is all about. What is new is the reliance of POAI to support huge transactions and to support enterprises that no longer have to pick up the phone or send a fax to buy, sell, or trade. POAI removes the need for human interaction to support a business transaction. The only requirement is an interaction with a web browser.

Unfortunately, although they are a necessity today, portals do not support the ultimate goal of application integration the exchange of business information in real time in support of business events that do not require human interaction. The goals of application integration are to design the user out of the process, thus removing the greatest source of latency in the exchange of information and to support the new event-driven economy.

6.11 Portal-oriented B2B Application Integration

Portal-oriented B2B application integration allows us to view a multitude of systems both internal enterprise systems and external, trading community systems through a single user interface or application. Portal-oriented B2B application integration benefits us through avoiding the back-end integration problem altogether; it extends the user interface of each system to a common user interface (aggregated user interface), most often a Web browser. As a result, it integrates all participating systems through the browser, although the applications are not directly integrated within or between the enterprises.

Portals have become so common, and so much has been written about them that we will cover just the basic concepts here. The important point to remember in the context of B2B application integration is that portals have become the primary mechanism by which B2B application integration is being accomplished. Whether that is good, bad, or indifferent doesn't really matter. This is simply the way it is. The reach of internal enterprise systems has been extended to trading partners by utilizing the familiar Web browser interface.

Task Check near by you and choose a small business owner apply POAI on that business.

6.12 Research in Source Integration

EAI is considered strategic because its potential contribution is measured in terms of attaining or exceeding key performance and competitive benchmarks for entire industries, as noted in the following examples.

Banking

The basis of competition in banking and financial services is customer retention. Customers with multiple accounts are less likely to change, but most financial institutions have stovepipe systems for credit cards, checking, savings, mortgage, brokerage, and other services. An EAI implementation would integrate the systems so that a data warehouse can aggregate account data, provide a single view to the customer, and recommend what additional products the customer should be offered. In EAI systems instituted at Bank of America (Bank of America (NYSE: BAC TYO: 8648) is the largest commercial bank in the United States in terms of deposits, and the largest company of its kind in the world.) and Royal Bank of Canada (Canada's central bank, established under the Bank of Canada Act (1934). It was founded during the Great Depression to regulate credit and currency. The Bank acts as the Canadian government's fiscal agent and has the sole right to issue paper money.) a transaction in one account triggers an event in another process for a marketing contact.

Manufacturing

Manufacturing's key competitive measure is cost reduction through inventory control and just-in-time processes. Allowing outside suppliers to view inventory data is possible with enterprise resource (Any software system designed to support and automate the business processes of medium and large businesses.) applications, but EAI provides a way for the manufacturer's systems to communicate with external systems to track sales, forecast demand, and maintain a detailed view of pricing and availability among many suppliers. General Motors' Covisint is a custom vehicle order management system that interfaces with inventory, planning, and logistics applications in the supply chain.

Life Science

In the life sciences industries, being first to market is critical since patents on new compounds expire within 17 years from the time the first documents are submitted to the Food and Drug Administration. EAI applications in pharmaceuticals, biotechnology companies, and research institutions integrate data from diverse laboratory systems with clinical data and other core systems so that it can be available to analytical applications. Several such projects are quietly underway within U.S. companies working on human and animal genome projects.

6.13 Towards Systematic Methodologies for Source Integration

EAI technology based on:

1. EAI technology trends
2. EAI tools/products
3. EAI solution types
4. OSI model and EAI

6.13.1 EAI Technology Trends

The initial focus of EAI was at the data-level i.e., moving or replicating data among databases, but it is evolving into business process automation. The present EAI technology is different to the earlier EAI solutions as its focus is on integrating enterprise applications and not data or assortment of different application types. Also the EAI solution can be reused for many other needs, not just on the same platform but also across heterogeneous platforms and networks and between multiple suppliers' packaged applications. The other differences in the past and present EAI solutions are that the integration is now at business process and practices level, not at application level or database level and the middleware is transparent to the user, so specific expertise in particular application-infrastructure technologies not required.

The Enterprise Application Integration trends are as follows:

1. Point-to-point Interfaces
2. Integration with Packaged integration brokers

Point-to-Point Interfaces

The traditional approach to integration is creation of point-to-point interfaces. The integration is handled through tools like extract programs, file transfers and update programs with screen-scraping tools/messaging system/TCP/IP socket connections. But good documentation is required for further integration or for incorporating changes.

Disadvantages/Constraints

1. If the number applications connected are many this leads to inter application spaghetti.
2. The approach is labor intensive and involves high cost and risk. It also does not assist if applications need to be changed or added.
3. The maintenance costs are also huge.

Integration with Packaged Integration Brokers

Integration brokers/message brokers are a type of high-level middleware. They work as follows:

1. Data level integration assists in accessing and updating of data in different applications by directly accessing the source and target applications' data either in files or database management systems.
2. Program level integration invokes existing application code to implicitly access the target application's data.

Tasks Performed by Integration Brokers

1. Interfacing-to move data to and fro applications
2. Transforming-to convert the semantic content of the data
3. Distributing-to move data between applications
4. Routing-to determine the destination of the data
5. Managing-to instrument the process and provide a toolset to support operator control of the process.

Characteristics of Message Broker/ Integration Broker

1. Scalability
   (a) For content based and subject based routing
   (b) For incrementing applications
2. Advanced team development and management development capability-version control, source code management etc
3. Handle batch as well as near real time integrations
4. Handle integration of mainframe as well as client/server capability
5. Low to moderate learning curve
6. Strong service and support capabilities to assist with project management
7. Vendor reputation

6.13.2 EAI Tools/Products

There are many types of products that have one or more functionalities of EAI. These are MOM (message-oriented middleware) systems; publish/subscribe systems, Transaction Processing monitors, application servers, data warehouse and data mart systems and logical integration systems. On the basis of the level of integration the tools perform the EAI solutions can be broadly categorized into Data level products and Business Model level products.

Data Level Products

The various products, which support the movement of data between applications, are:

1. File transfer tools
2. Copy management
3. Data propagation
4. Schema-specific data synchronization
5. Database replication
6. Extraction/Transformation

Only extraction/transformation products are capable of getting data directly into and/or out of an application's data store and can also change the format of the source data so as to fit the target product group of EAI solutions.

Extraction/transformation products are of three types:

1. Code Generators
2. Transformation Engines
3. Data Warehouse and Data mart Loaders

Code Generator

The code generator assists in the manual coding of programs by extracting data from an application and transforming it for loading into another application. This is useful for simple application network.

Disadvantages

1. The resulting program is not independent of the source or target system, so for integrating with more than one system extra programming / processing is required.
2. The desired level of data movement cannot be achieved, so modifications have to be done to the generated code
3. Language used for the generated program may differ from system to system
4. Scalability is a major concern as the integration is point-to-point
5. Modifying an application can require major regenerations and modifications to existing interfaces.

Transformation Engines/ Hubs

They use application metadata to create export-transform-load programs like code generators. But the difference is that all code is executed at a central location independent of the source

Notes

and target. This works by getting the source data and moving it to a separate location where transformation takes place.

Advantages

1. Centralized approach assists in scalability
2. Rapid interface development
3. Data staging
4. For large volumes of data some tools have transient data store, where excess data is processed.
5. The same development environment and tools can be used for all application interfaces, so there is minimal impact on the source and target systems.
6. It is very useful for large data volumes

Disadvantage

As transformation is done in a centralized location the tools are not scalable.

Data Warehouse and Data Mart Loaders

The Data warehouse and Data mart loaders can be found in either code generator or engine/ hub forms. The focus is in transforming operational data into a form that can be loaded into a very specific type of data store. Data aggregation is required so as to transform data in an application network.

Disadvantages

Warehouse loaders do not have the fault tolerance or performance requirements that make them viable for linking together a host of operational systems.

Business Model Level Products

The various products are:

1. Remote Procedure Calls
2. Stored Procedure Calls
3. Object Request Brokers
4. Transaction Processing Monitors
5. Database Triggers
6. Message Queuing
7. Message Broker
8. Asynchronous RPCs
9. Publish and Subscribe

At business model level two applications can be integrated through the use of function calls i.e., one application sends data to the other by calling a function over a network. In the PRC mechanism the source application calls the function of another by specifically naming the target application and its function.

In the Message Broker the application calls a logical function that it wishes to be executed. The broker then maps this to a specific function in another application. Neither the source nor the

Notes

target applications know in advance which application is involved. This makes the message broker the most scalable and best EAI option as the source and/ or the target applications and/ or the business process logic can be changed without interrupting the whole system. The Message broker is a set of software components that allow applications to communicate with each other through non-invasive bi-directional exchange of messages.

6.13.3 EAI Solution Types

There are primarily two types of EAI solution at high level- data level integration and message-based integration. Data level integration basically assists applications in the exchange and sharing of data across a common data store. For inter-enterprise application integration at data level Extensible Markup Language (XML) is very useful. Message based application integration is based on messaging software that is network aware. This is nearer to the complete EAI solution. Message oriented middleware products are thus becoming increasingly popular. Most EAI software offer tools to model business processes and link the applications with middleware that can make each application communicate via data message.

6.13.4 OSI Model for EAI

The Open System Interconnection Model for EAI contains 12 layers as against the seven-layered structure for network applications. The various layers are as follows (Table 6.1):

Figure 6.1: The Open System Interconnection Model for EAI

Layer	Name	Description	Best Source of Knowledge
Layer 12	Business Process	Defines company specific business processes	Operational users
Layer 11	Business semantics	Holds company specific data definition and structures	Operational users and IS staff
Layer 10	Application Semantics	Contains in-depth knowledge of application structure and meaning	Application vendor's staff and vendors staff
Layer 9	Interface Syntax	Defines methods for sending/ receiving information to and for applications	Middleware Vendor's staff
Layer 8	Integration Middleware multiple applications	Architecture for integrating	Middleware Vendor's staff
Layer 7	Application	Provides standardized services	
Layer 6	Presentation	Encodes, encrypts and specifies data transfer formats	
Layer 5	Session	Manages session protocols	
Layer 4	Transport	Manages network layer connections and delivers packets	
Layer 3	Network	Addresses and routes packets	
Layer 2	Data link	Frames packets and controls physical layer data flow	
Layer 1	Physical	Electrical and mechanical specifications	

6.14 EAI Architecture

EAI architecture reduces the number of interfaces and provides a standard methodology for application integration. Layering the different transport technologies does this. The black box EAI solution employs an array of middleware comprising message broker, transaction processing integration, Database Remote Procedure Calls, Screen scrapers, Java applets, Active X Controls, etc.

The EAI architecture also provides services such as application development tools, repository management, routing, publish/subscribe services, data flow, data transformation, security services, recoverability and workload balancing.

Hub and spoke architecture is the most common. All applications connect to a central hub, which connects to many application spokes. The hub provides centralized services while connectors or adapters provide the services for each spoke or integration point. Adapters provide integration with the centralized hub for a specific resource like relational database or a java application, enabling information or invocation of a process against a specific resource.

EAI assists by causing existing and new applications to exchange data via messages governed by the rules of the business process. The business process is to be modeled and rules defined for the applications to follow. A message Broker routes the messages according to these rules. The data in the messages is transformed into the format required by the target application along the way. As the EAI software is independent of the individual applications it connects, the business processes can change and grow without requiring changes to the application.

6.14.1 Layers of EAI

The EAI solutions can be categorized as a three-layer solution on the basis of the level of integration and functionality. The three specific layers to EAI solution are:

1. Communications
2. Routing and brokering
3. Business Intelligence

Communications

The communications layer comprises of tools that assist in accessing data sources, inter-process communications, network transports and representations of messages that pass between applications. It includes the facilities for distributing processing over a network and includes the following technologies: TCP/IP, publish and subscribe, database server protocols and middleware, multicast IP, asynchronous messaging, remote procedure calls, etc. The communications layer essentially views the world as a set of data sources.

Routing and Brokering

In this layer some amount of decision-making and processing capabilities can be found. The primary job of this layer is to aggregate, broker, transform, filter, and format data so it can be understood by the other systems that are connected by the EAI solution.

Business Intelligence

The Business Intelligence layer plays a critical role in achieving the virtual application. This layer provides an environment that responds to messages from the routing and brokering layer. It then uses a set of declarative rules to make intelligent business decisions based on company goals. This layer connects to rules analyzers and on-line analytical processing (OLAP) services to assist in the decision making process. It is essential for companies to build this layer for a more proactive and competitive approach to conducting business.

6.14.2 EAI Software Topology

The integration topology is a major consideration when building an EAI architecture to meet the diverse set of integration requirements in an organization. Selecting the right topology will assist in integration performance, event management and maintenance costs.

Types of software topology are:

1. Hub/star topology
2. Bus topology
3. Point-to-point topology
4. Pipeline topology
5. Network topology

Hub/Star Topology

Hub typology is useful for creating a central point of control. Messages are sent from source to central hub, which is often in the machine itself. Hub typology works well if business events are independent and if the Message Oriented Middleware (MOM) on which the typology is based is from a single vendor. Here the source application sends a single message in one format and the hub reformats the message as necessary and relays it to the various spokes connected to the hub.

Advantages

1. Reduces re-entry of the data as it is centralized
2. Promotes re-use of the data
3. As all data must pass through the hub it is easy to monitor and audit data flows across the network from the hub
4. Scalability is more

Disadvantages

1. Mostly the hubs available cannot handle incoming transaction from any other source than the middleware on which they operate.
2. They cannot manage integration events involving multiple sources and destinations
3. If database is required, it would become a source of processing or routing bottlenecks as volumes grow and integration rules become complex.

Bus Topology

Bus typology is useful for distributing information to many destinations. Source applications put messages onto a system-wide logical software bus that is accessible to other applications. One or more applications can then selectively subscribe to the messages broadcast on the bus. Traffic does not need to flow through the central switching point. This is possible in publish and subscribe middleware only. Bus typology circumvents the problem of bottlenecks.

Point-to-Point Topology

Point-to-point topology enables applications to communicate directly with one another. This is useful when synchronous communication and persistence are required. Applications with pre-built integration for ERP applications use this topology. Too much of the point-to-point integration in an organizations IT structure leads to inter application. Benefit of this topology is its ability to take full advantage of the context and semantics of the original data as it is transformed into one or more target structures. The major constraint for this topology is if there is any change in either of the applications like up gradation, etc then the whole integration has to be changed.

Pipeline Topology

Pipeline topology is useful if dynamic configuration is not required and multiple pipelines are independent of each other. The information flows will be based on the First In First Out approach. This is a very simple level of integration.

Network Topology

Network topology is the best option available if there is a lot of asynchronous activity and independent transactions must coexist with one another. For this topology to work well, the interfaces must be well defined and robust. If there is a snag at the interface level then the entire network communication can fail.

Task E-business is a business done on internet. Discuss something about B2B and B2C business.

6.15 EAI Solution

The solution for EAI consists of technology for supporting both data level and business model level interfacing. Design patterns are used to identify, categorize and reuse interfaces so as to ensure that the selected method of application-to-application communications is the best. Effective EAI solutions reduce the up front costs of implementation and provide open, seamless integration of business processes with any type of technical infrastructure. This also results in a Zero Latency Enterprise.

6.15.1 Requirements for Effective EAI Solution

1. IT strategy needs to be mapped out according to the business strategy and the objectives
2. Complete understanding of the business processes data models and the supporting systems and applications currently in place
3. Planning for the whole process' right from need identification, vendor selection to implementation and future requirements
4. The EAI architecture, viz., process models and integration requirements, has to be formulated from the IT strategy and architecture
5. Evaluate the EAI tools and the vendors
6. Accountability and ownership has to be established
7. Evaluate the solutions and the scope of integration covered by the technology

8. Invest in systems management and administration
9. Right implementers with right skill set are required.

EAI implementation requires careful planning. This is because EAI is more than moving data from source to a target; it is a function of application semantics. EAI involves transformation of application content as data moves among the various systems. This requires a top-down approach' focusing on integrating application at a business context level and not just at technical level. Business level integration is concerned with business processes and the associated business rules.

6.15.2 EAI Software Flexibility

EAI software must be implemented with five layers of technology for flexibility. The different layers are as follows:

1. Business Process Support
2. Transportation
3. Services
4. Interfaces
5. Transformation

Business Process Support

EAI solution set has tools, which let the users visually diagram the business processes so as to let the users declare rules for each message. This is useful to visualize the business processes and thereby control different activities and ease the flow of information. Intelligent routing capability that can look at a message and figure out the nest course of action is required in an EAI solution.

Transportation

Data can be routed point-to-point or with an architecture called publish/ subscribe, in which applications send messages to other applications that have registered interest with the message broker. The application sending information is the publisher and that receiving information is the subscriber. Depending on the network and platforms the application resides on this can be done with middleware such as database drivers, component object models or messaging middleware.

Services

This characteristic is required by messages to carry out missions successfully. The different services that are to be present are:

1. Queuing to store messages if receiving application is slower than the sending one
2. Transactional Integrity- to confirm that the transaction has completed before a message is sent or acknowledged as received.
3. Message priority, error handling and hooks to let the network management tools control the network traffic

Interfaces

Access to application is through the interfaces. Interfaces interact with the application either via descriptions they provide to their platforms component model or by taking advantage of the program Application Programming Interface. Thus the interfaces play an important role in selecting an EAI tool as they should be such that no/minimum coding will be required while integrating.

Transformation

As data format is not same for all applications, tools are required that let users visually map, coordinate one application data format with the another application data format and transform the information as required.

6.16 EAI Solution Evalution

6.16.1 Criteria for Selection

The EAI solution should have the following functionalities:

1. Workflow Management- facility for designing transaction work flows across applications
2. Seamless Data Transformation- full and simultaneous transformation of application content among multiple sources and destinations, regardless of application complexity.
3. Intelligent Content Based Routing- powerful, rules based routing of messages, files and other data objects based on content from any part of the transaction and centralized management of the routing rules
4. Business Rule Management- graphical environment for definition and management of business rules to support business process that cross application boundaries
5. Resource Adapters- for seamless technology integration with wide array of data sources and destinations
6. Functional capabilities and characteristics
7. Flexibility, ease of response to change in integrated applications over time
8. Support for components, including standard component architectures like COM and CORBA
9. Simplicity or complexity of tools for configuring integration behavior
10. The extensibility provided via traditional programming
11. Development tools provided
12. The other integrators and partners supported
13. The legacy application integration capability and access to mainframe transactions
14. Support provided for heterogeneous messaging infrastructure

6.16.2 EAI Solution Evaluation Methodology

Information Group has come out with an evaluation methodology for EAI solution on the basis of seven criteria, which can be used to compare different solutions. The point to note here is that these criteria are customer specific i.e., dependent on the customer requirements the importance of each criterion varies. The criteria to be checked are:

1. Adapter/ Connector Fit
2. Tools Productivity/ Quality
3. Runtime Quality/ Scalability
4. Runtime Fit to Purpose
5. Business Process Support
6. Integrator Resources
7. Purchase and Ownership Cost

Adapter/Connector Fit

The extent of provision of pre built or pre configured adapters or connectors in the solution. The rating is dependent on what packaged applications and legacy environments are required to be integrated and the quantitative and qualitative assessment of the adapters/ connectors. The important point to consider in assessment is the impact of the adapter/ connectors available on the time to market, where a high rating means the amount of pre built connectivity will accelerate the integration project.

Tools Productivity/ Quality

The productivity of the integration development work is dependent on the quality of tools provided and thus this criterion's impact is more in the case where the adapter/ connectors are not available. If the amount of custom integration work to do is more then this criterion increases in vitality. This also determines the flexibility and the maintenance cost of the system.

Runtime Quality/ Scalability

Scalability is important as it determines the speed of the system. Quality of service includes the level of reliability or guarantee of delivery and the level of transaction integrity. Throughput, latency and efficiency may also be considered for assessment of quality of service.

Runtime Fit to Purpose

There are four main points, which are required in different combinations:

1. Transactional real-time component integration
2. Queued messaging model
3. Publish and subscribe messaging
4. Bulk data movement

Business Process Support

All integration scenarios require business process support. There are two ways by which this can be taken care of are:

1. Facilities to model business processes and generate or execute an automated version of the process are include in the integration solution
2. Specific business processes are already pre configured as part of the solution

Integrator Resources

They can be provided by the vendor/ partners/ the Organization itself.

Purchase and Ownership Cost

The price sensitivity is high in this category as the differentiation is very less.

6.17 EAI Software Checklist

The ability to support evolving integration topologies is important, as there are rapid changes in the business requirements. EAI software if chosen right will play a key role in integrating the business processes. For meeting this requirement of business process integration a typical EAI solution should satisfy the following criteria:

1. Topology Independence
2. Support for multiple operating platforms
3. Support for multiple middleware systems
4. Connectivity to databases and files
5. Content-based application adapters
6. Process flow control
7. Event coordination and event management
8. Integration without programming
9. High performance
10. Proven implementation

Topology Independence

The architecture to select for connecting an integrated process depends on various factors/ issues like performance, timing requirement, event coordination etc. Therefore an open EAI topology has to be chosen, not restricting only to Hub or Bus or any other approach. Flexibility is the key word.

Support for Multiple Operating Systems

Business processes often are required to be platform independent. So the EAI software should be flexible enough to execute the process on any platform.

Support for Multiple Middleware Systems

The EAI software should focus on the business process and not on the underlying technology that is used to transfer the data. Good EAI software provides pre-built adaptability for all middleware categories, like MOM; publish/subscribe middleware and ORB

Connectivity to Databases and Files

The EAI software should support not only message routing but also provide direct access to databases, files, email systems etc without separate steps i.e., it should be a part of the integrated process.

Content-based Application Adapters

The EAI software should not only create and maintain the adapters from applications metadata, but also provide descriptions with semantics and syntax, eliminating the need for coding.

Process Flow Control

The EAI software should provide a graphical environment to describe the processes and also should have provision for acknowledging events, trigger execution, intelligently route data and ensure transactional integrity across entire integration scenario.

Event Coordination and Management

Real time events triggering business processes have to be monitored and managed to ensure that they achieve a coordinated result. The software should also include a run time environment, which supports active listening, event coordination and multi threaded processing.

Integration without Programming

EAI software should handle the complexities of the Business process integration by itself without resorting to hand coding.

High Performance

As business process involves high transaction volumes or complex rules, the EAI software should prevent bottleneck and should have features like multi-threading and multi-processing along with performance monitoring tools.

Proven Implementation

The EAI software should be proven and in use by other customers so as to minimize risk, as business process integration is a mission critical task.

6.18 EAI Market Segmentation

EAI Solutions are moving from middleware messaging systems to Business process Integration. The EAI market as of now is concentrated mainly on the layers 8 & 9 of the OSI model for EAI, viz., Integration middleware and Interface Syntax. The main reason for the focus on these two layers is the immaturity of the EAI market and also that profits are easier to achieve in these two layers.

The EAI product market can be differentiated into:

1. Platform Integration
2. Data Integration
3. Component Integration
4. Application Integration
5. Process Integration

Platform Integration

This provides connectivity among heterogeneous hardware, operating systems and application platforms. The various technologies providing platform integration are:

1. ***Messaging:*** this is for asynchronous connectivity
2. ***Remote Procedure Calls:*** Synchronous connectivity
3. ***Object Request Brokers:*** Both types of connectivity

The logic for connecting each application must be defined either through code or pre coded applications adapters. Additional functionality is required to reconcile the differences in data representation in the system. This can be done by hand coding or by the use of data translations and transformation products. Logic is required for message routing and this can be provided either through hand coding or by a Message Broker.

Monitoring and management of end-to-end business process has to be done through hand coding or automated process management tools.

Data Integration

Data integration is of Two types:

1. Database gateways like Sybase DirectConnect, Information Builders EDA SQL and Oracle Open Gateway which provide SQL access to heterogeneous data sources. They are synchronous data access products and require application developers with knowledge in the database schemas
2. Tools for Extracting, Transforming, Moving and Loading Data- ETML tools:
   (a) They are batch or point in time solutions suitable for initial loading of data warehouse or large batch transfers. They extract and load data directly bypassing application logic. ETML vendors are extending functionality through messaging support.

The solution set of system-to-system data map. A new system must be mapped to all other systems it's integrated with. Changes in application impact mapping to every other systems it's integrating with. Tools that provide impact analysis simplify change management.

Component Integration

Hub and spoke integration-hub provides some of the integration. Application servers are used to provide data access to variety of relational database sources applications Adapters to packaged applications and middleware services like messaging are also available.

Application Integration

Application integration provides a framework for technology for near real time processing. The framework includes:

1. Underlying platform integration technology
2. Event integration through Message Broker that provide data translation
3. Transformation & rules based routing
4. Application interface integration provided through application adapters to packages
5. Custom applications

Integration frameworks assist in reducing the complexity of creating, managing and changing integration solution. The advantage is faster time to market through pre built adapters and reusable integration infrastructure.

Process Integration

This provides the highest level of abstraction and adaptability for an EAI solution. This enables managers to define, monitor and change business processes through a graphical modeling interface.

Business Process Modeling helps business users and analysts to define how information flows across systems and organizational boundaries through a graphical model and declarative language instead of programming. The integration solution is generated from the model. When changes are required, they can be made in the model and the same can be regenerated in the solution. Simulation can also be done before the implementation of the solution.

6.19 EAI Implementation

Understanding the organizations business processes and data is essential to select which processes and data require integration. There are four different scenarios for EAI viz.,

1. Database Linking
2. Application Linking
3. Data Warehousing
4. Common Virtual System

Database Linking

This is basically linking two or more databases so that information is shared between the databases at some point. The information can be exchanged and duplicate information maintained or information can be shared. This is the simplest and initial form of EAI.

Application Linking

This one is more complex than database linking. Application linking means both processes and data between two or more applications are integrated. The advantage of this there is redundant business processes are not created as the processes are shared between applications.

Data Warehousing

This is similar to database linking. Data Warehousing is the collection of meaningful data from several data sources to support decision-making efforts within an organization. The data from different data stores is extracted, aggregated and migrated into a data mart or data warehouse. EAI assists in real time data warehousing.

Common Virtual System

A virtual system means that for any transaction, the information required for it will be available irrespective of where the information exists. EAI helps to integrate diverse systems so that they appear as one monolithic and unified application.

Notes

6.20 Types of EAI

There are two types of EAI on the basis of the level of integration done, type of applications integrated, viz.

1. Data level
2. Business Model level

Data Level

Data level EAI is the process/technology to move data between data stores, i.e., extracting information from one database, processing the information if needed and updating the information in another database. Business logic may also be transformed and applied to the data that is extracted and loaded.

Business Model Level

Business model level divided into three category and these are:

1. Application program interface level
2. Method level
3. User interface level

Application Program Interface Level

Here the custom or packaged applications' interfaces are used for integration. Developers use the interfaces to access the business processes and information so as to integrate various applications to share business logic and information.

Method Level

The business logic is shared between different applications within the enterprise. The methods of various applications can be accessed without rewriting each method within the respective applications.

User Level

User Interfaces are used to tie different applications together. This process uses windows and menus to get the relevant data that needs to be extracted and moved to other applications and data stores.

Case Study

Healthcare Middleware Solution

Middleware/Data Management Solution to significantly streamline laboratory work flow:

The Client

Our client is a software company that develops value-added solutions in data management for clinical laboratories. Our client is experienced in the most advanced Microsoft and Internet technologies.

Contd...

The Idea

With a very rich history our client built its place on the market on a company culture oriented toward customer satisfaction and the quality and reliability of their solutions. Our customer's extensive client portfolio of large companies has driven the need for numerous interfaces for most laboratory information systems and instruments available on the market. The Middleware/Data Management software enables clinical labs to consolidate test information from various instrument systems and manage data in real-time.

The Challenge

For manufacturers and clinical laboratories, software is the premier cause of postponements of instrument launches. The software that causes most delays is the one that mediates interactions between laboratory instruments and the laboratory information system (LIS) middleware.

In order to speed up instruments go-to-market times, the middleware solutions needed to feature in a highly customizable work flow, an entire range of functionalities like: multi-disciplines, multi-sites, multi-LIS, multi-instruments, multi-users, multi-languages, automatic results validation, delta checking, reflex testing, quality control, results editing, and archiving and restoring.

The solution also needed to comply with the strict regulatory framework in the area of clinical related operations like 21 CFR Part 11, FDA approval or CE marking. On top of this, our client's own standards and certifications implied total quality control over the development process to fully contribute to products destined to become homologated for the U.S. and European markets.

The Solution

OSF Global Services partnered with the client building a mixed team with his own staff, working from the client's own specifications to build tools and framework elements.

The use of open standards and service-oriented architectures (SOA) to build the middleware was selected to embed the digital library information sources in the technical frameworks, thus obtaining an open, reusable middleware that can bridge the complex functionality of the LIS with the instruments interfaces.

Running off a desktop station, the middleware system views, tracks, and manages from one location all laboratory instruments, to enable automatic validation and reporting of normal results, according to rules programmed into the software. With reporting modules such as regulatory provider, audit and auto-validation, quality control interface, and up to 50 instruments driver development, the middleware system developed provides a versatile response to a variety of issues.

Heavy quality control documentation, testing and process maturity procedures were implemented all throughout the development of the project to ensure compliance with all quality standards to integrate with client's own processes and procedures as well as the challenging clinical environment requirements.

Technologies

Microsoft .Net Framework 2.0,

C Sharp,

Dev Express,

Engine reporting, code covering tools

Contd...

The Results

Our client's middleware critically improves the integration of the instrumentation, the automation system, the lab's data management component, and the LIS. The data management system that OSF Global Services contributed to helps streamline laboratory information processes for maximum efficiency interfacing timely, accurate and reliable clinical test results, eliminating LIS bottlenecks, and enabling lab operations in a cost-effective manner.

Questions

1. What are the challenges they face?
2. If you are the manager of this firm what are your suggestion for your client.

Source: http://www.osf-global.com

6.21 Summary

- Enterprise Application Integration is an integration framework composed of a collection of technologies and services which form a middleware to enable integration of systems and applications across the enterprise.
- Enterprise application integration (EAI) is the process of linking such applications within a single organization together in order to simplify and automate business processes to the greatest extent possible, while at the same time avoiding having to make sweeping changes to the existing applications or data structures. In the words of the Gartner Group, EAI is the "unrestricted sharing of data and business processes among any connected application or data sources in the enterprise."

6.22 Keywords

CRM: CRM (Customer Relationship Management) is a comprehensive strategy and process of acquiring, retaining and partnering with selective customers to create superior value for the company and the customer.

e-business: e-business is the use of the Internet and other networks and information technologies to support electronic commerce, enterprise communications and collaboration, and web-enabled business processes both within an internetworked enterprise, and with its customers and business partners.

e-commerce: It is a general concept covering any form of business transaction or information exchange executed using information and communication technologies.

Enterprise Application Integration: Enterprise Application Integration (EAI) is defined as the use of software and computer systems architectural principles to integrate a set of enterprise computer applications.

6.23 Self Assessment

Choose the appropriate answers:

1. MOM stands for:
   (a) Message-oriented Middleware
   (b) Management-oriented Middleware
   (c) Message-operated Middleware
   (d) Message-operational Middleware

2. OSI Model stands for:
   (a) Open System Interchange Model
   (b) Open System Interaction Model
   (c) Open System Interconnection Model
   (d) Open System Interconnection Mode

Fill in the blanks:

3. .... Modeling helps business users and analysts to define how information flows across systems and organizational boundaries through a graphical model and declarative language instead of programming.
4. .... enterprise refers to an organization that can change its business rules in real time to act on new market opportunities and customer demands.
5. .... allows applications to share common business logic.
6. The service oriented Architecture (SOA) acts like an enabler to ....
7. .... are enterprise applications existing either within a single enterprise or across many enterprises.
8. .... communicate with both the web server and the resource server using transaction-oriented application development.
9. Manufacturing's key competitive measure is cost reduction through inventory control and .... processes.
10. The traditional approach to integration is creation of .... interfaces.
11. The .... layer plays a critical role in achieving the virtual application.

6.24 Review Questions

1. What do you mean by enterprise application integration?
2. What are the various reasons of emergence of EAI?
3. Briefly explain the EAI technology in detail.
4. Explain the architecture of enterprise application integration.
5. What do you mean by EAI market segmentation?
6. Describe various advantages of EAI implementation
7. What do you mean by source integration?
8. Explain service oriented application integration.
9. What do you mean by portal oriented application integration?
10. Write short note on "research in source integration".

Answers: Self Assessment

1. (a)	2. (c)
3. Business Process	4. Zero latency
5. Service-Oriented Application Integration (SOAI)	
6. Business Method Integration strategy	7. Back-end applications

Notes

8. Application servers	9. just-in-time
10. point-to-point	11. Business Intelligence

6.25 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 7: Applications Cases of Integration

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Know about EAI
- Describe EAI industry specific implementations
- Describe case studies

Introduction

Enterprise Application Integration (EAI) is a process of data and application integration technologies which focuses on linking transactional applications together, typically in real time. This differentiates it from EII (Enterprise Information Integration), which is focused on presenting a unified view of data. It is common knowledge that EII is viewed as a bi-product of EAI, because, since the application processes are interlinked, the flow of information needs to be channeled for a unified view. EAI tools are also generally focused on point-to-point application communications.

Technologies that can make up an EAI solution include web services, transaction monitors, message brokers, and message queues. Most commonly, EAI vendors discuss messaging and web services.

In today's dynamic environment, applications like supply Chain Management, Customer Relationship Management, Business Intelligence and Integrated Collaboration environments have become indispensable for any organization that depends upon continuous uninterrupted production in a solid and competitive commercial environment. And so therefore the EAI methodologies provide a consistent solution for this. Each of these applications in themselves are a mamoth effort. Enterprise Application Integration (EAI) is a step forward where it links these applications and others in order to realize financial and operational competitive advantages.

7.1 Introduction to EAI

Today's business world is infinitely more complex than it was a long time ago. Modern companies have a large number of applications that take care of running the business. Such diverse applications weren't a problem initially because they were meant to provide stand-alone, independent, and automated functions. The result of this diversity was a collection of stovepipe applications rather than a unified network of linked systems. But now, the companies are realizing the utmost need to integrate these independent data silos to leverage the information stored in them across the various vertical and horizontal domains as well as surmount the ever-increasing costs of building new applications.

And this is where an EAI solution comes into the picture.

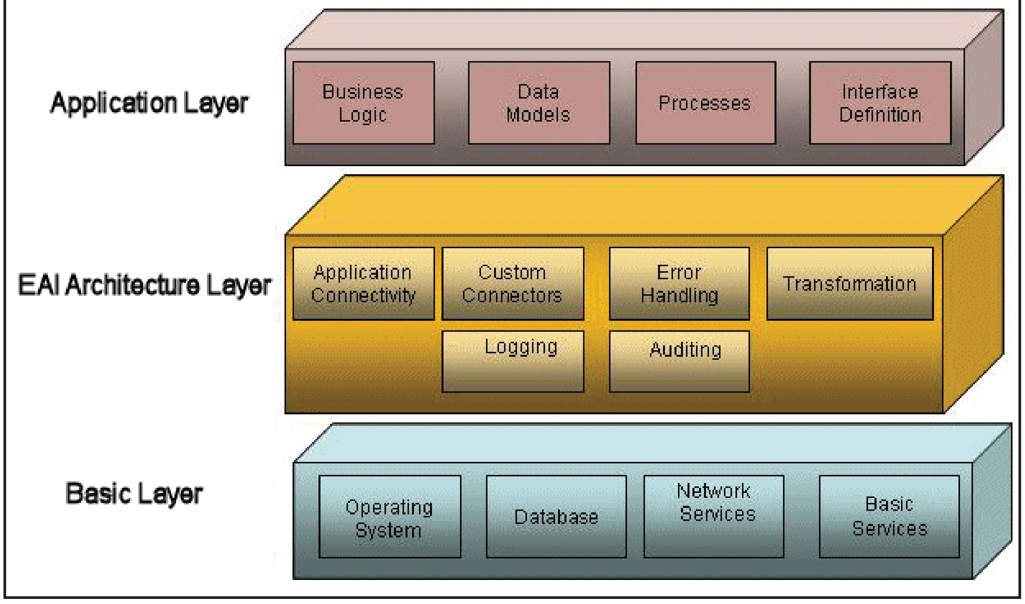
EAI is a collection of processes, software and hardware tools, methodologies, and technologies. When implemented together, they have the aim of consolidating, connecting, and organizing all the businesses computer applications, data, and business processes (both legacy and new) into a seamlessly interfaced framework of system components that allow real-time exchange, management, and easy reformulation of the company's mission-critical information and knowledge. It is an unrestricted sharing of data throughout the networked applications or data sources in an enterprise.

When designing an Enterprise Application Integration (EAI) solution, it is important to recognize that there are different levels of integration, each with its own requirements and considerations. Successful implementation of consistent, scalable, reliable, incremental, cost-effective EAI solutions depends on the standards and methodologies that we define for these levels. It must be determined how we need to share information:

1. Within an application
2. Between applications within an enterprise
3. Between enterprises
4. Directly with customers

Moreover, the solution should be based upon a multi-tiered, non-redundant, coarse-grained architecture that is enforced across the enterprise. This requires the total set of functions involved in the end-to-end integrated solution be provided by distinct layers, each providing unique and non-overlapping services as shown in Figure 7.1.

Figure 7.1: Multi-tiered EAI Architecture



Notes

This architecture provides sufficient high-level consistency for interoperability and a certain degree of local freedom.

Example: The architecture supports semantic diversity (different interpretations of the same data) and permits use of diverse technical tools and techniques. It also provides the basis for organizational responsibility and ownership of each layer. For example, business data persists at the application component layer and not in the middleware/EAI layer. Hence, this data is the responsibility of the application owners and not the middleware solution developers.

7.2 Purpose of EAI

The purposes of EAI are:

1. Ensure that Data (information) in multiple systems is kept consistent. This is also known as EII (Enterprise Information Integration).
2. Business Process are integrated and linked across applications.
3. Business policies or rules from applications are abstracted. So a change in vendor shouldn't warrant rebuilding the application from scratch.
4. It provides a single unified consistent access interface to various applications.

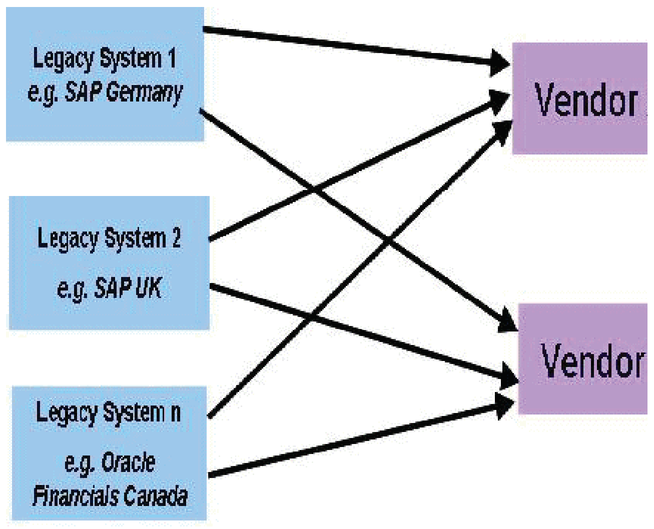
Case Study

Case 1: Business Case Introduction

Royal Wallace, a UK-based transportation company, is a global leader in the rail equipment and servicing industry. Its wide-range of products includes passenger rail vehicles and total transit systems. It also manufactures locomotives, freight cars, bogies, and provides rail-control solutions.

Because of the structure of its business, the Royal Wallace company has a presence in several countries across Europe and in North America, including the USA and Canada. And, each of these country-specific sites has its own dedicated legacy system such as SAP-based systems in UK and Germany, Oracle financials in Canada, and so on (see Figure 1).

Figure 1: As-Is Scenario



Contd...

Notes

As can be seen in Figure 1, in the as-is scenario, these legacy systems are acting as independent data silos with absolutely no kind of data-sharing happening between them. If the German SAP system wants to procure something from any of its vendors, it does so by using some kind of manual process at its end. The same scenario is replicated for all the other systems, too. And, this is leading to a lot of data duplication taking place. For example, if two different systems are using the same vendor (say, Vendor A) for procuring white board markers, this vendor (Vendor A) information is getting stored at both the legacy systems. Also, this kind of scenario makes it impossible to leverage existing information across multiple systems.

Royal Wallace understands that, to survive the competition in today's fast paced economy and to reduce its time-to-market, it needs to put an integration solution in place as soon as possible. The integration model should integrate all the existing legacy systems irrespective of the platforms and operating systems that these systems are based upon. The model should take care of the various data formats that would pass through this integration scenario and apply the necessary transformation, translation, and data validation rules upon them. And last, but definitely not the least, this integration solution should be scalable enough such that any new system could be easily plugged into this solution in the future with minimal changes.

The integration model should provide:

1. Integration between the various legacy systems
2. Automated process for procurement of goods
3. Data transformation and translation logic according to the prescribed rules
4. Persistent storage mechanism for the data
5. Validation of business data to a certain extent
6. Connectivity to various drop zones and databases

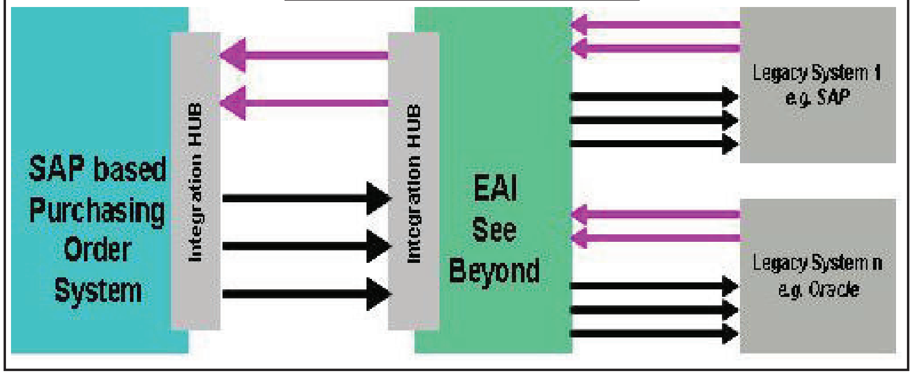
Project Life Cycle

Design and Architecture

As seen in Figure 2, the EAI solution proposed would make use of the integration capabilities of the Seebeyond eGate software to seamlessly integrate the various individual legacy systems of the Royal Wallace company with the SAP-based purchasing order (PO) system.

SeeBeyond eGate Integrator is a fully J2EE certified and Web services-based distributed integration platform. It provides the core integration platform, comprehensive systems connectivity, guaranteed messaging, and robust transformation capabilities.

Figure 2: To-Be Scenario



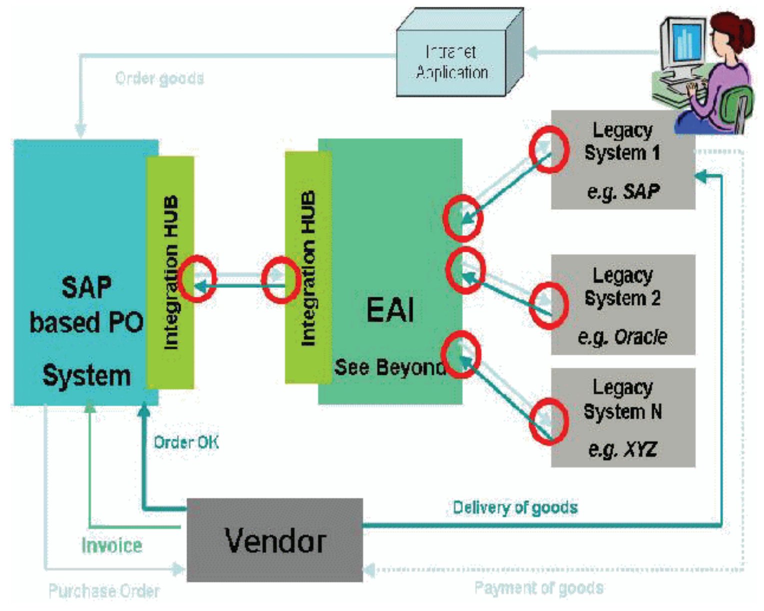
Contd...

In this scenario, the SAP-based PO system acts as a data hub for all the legacy systems such that all the procurement orders at Royal Wallace would be routed through this SAP based PO system. Also, Seebeyond is acting as the integration hub between the SAP-based PO system on one end and the legacy systems at the Royal Wallace company on the other, thereby enabling a bi-directional flow of data.

Process Flow

Figure 3 gives a comprehensive picture of the process-flow through the integration model.

Figure 3: Process Flow



Whenever a Royal Wallace employee needs to procure something, she logs on to an intranet application. The workflow in this application is managed by the SAP-based purchasing order (PO) system. The PO system, in turn, places the order with the vendor on behalf of the employee. The vendor acknowledges the receipt of the order to the PO system and delivers the goods to the concerned legacy system. Once the goods are delivered, the vendor sends an invoice for the same to the SAP PO system. The PO system, in turn, sends this invoice to the appropriate legacy system which then makes the necessary payment to the vendor. The legacy system also sends back a 'payment done' kind of acknowledge back to the PO system. This scenario is replicated for all the other legacy systems, too.

The EAI solution (Seebeyond eGate) is responsible for all the communication between the SAP PO system and the legacy systems. Various interfaces were developed as part of this solution to enable bi-directional flow of data. All the information pertaining to the various legacy systems and the SAP-based PO system (source and target) were captured as part of the functional specifications. These specifications covered the following topics:

1. The platforms, databases, and operating systems for the source and target
2. The various applications constituting these systems and the way they interacted with each other
3. The input and output data formats for the source and target (EBCDIC, ASCII, Binary, and so forth)

Contd...

4. Frequency of data transfer and outage periods
5. System connectivity details such as FTP drop zones
6. The file structures that were being processed, if applicable

The technical specifications covered the following topics:

1. Detailed technical design and architecture of the project
2. All database-related details including the various servers, IP details, and the like
3. Detailed configuration details of the various Seebeyond integration components, including connectors
4. Detailed description of the error flow, including the codes
5. Detailed description of the logging mechanism
6. Input and output data structures and the transformations involved, if any
7. Pseudo-code

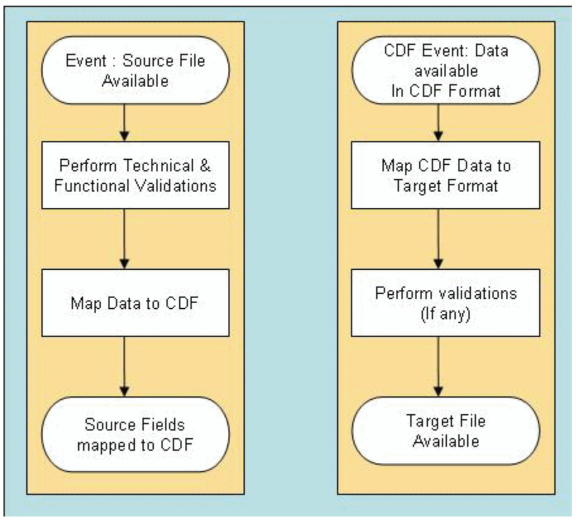
Common Data Format (CDF)

The various systems at Royal Wallace company that were to be incorporated into the integration model were based on different platforms. And, the applications that were running on these systems used to exchange data in various different formats with the outside world. Most of these were proprietary formats that each of these applications had created as per their own requirements over the years; there was no single standard approach that had been followed. Some of them had fixed-length records, others were delimited; some followed ASCII character encoding, others were EBCDIC; and so on. And above all, as most of these formats were in existence over the past several years, the system owners were very reluctant to replace them for a common standard format/template.

So one of the major challenges was to come up with a solution that could somehow transform and translate these multiple data formats passing through the integration model, thereby enabling the various systems to communicate with the SAP-based PO system. Moreover, the solution had to be scalable enough to incorporate any new data formats in the future with minimal changes to the existing scenario.

The solution was to make use of the Common Data Format (CDF), as shown in Figure 4.

Figure 4: Common Data Format


Event : Source File Available
Perform Technical & Functional Validations
Map Data to CDF
Source Fields mapped to CDF
CDF Event: Data available In CDF Format
Map CDF Data to Target Format
Perform validations (If any)
Target File Available

Contd...

Notes

The Common Data Format (CDF) was an event structure that was generated within the Seebeyond eGate environment. Because the various applications deployed on the legacy systems used the integration model to procure goods from vendors, the incoming data to the EAI environment from the various applications had some common data elements irrespective of the data formats that these applications used. For example, all the procurement requests had information about type and quantity of the goods to be procured. All these common elements were captured under multiple nodes in the CDF event structure in Seebeyond. This proved useful in several ways:

The entire data structure for the whole integration model was defined using a single event definition in Seebeyond. Now, all the incoming data into EAI could be parsed and then mapped to the CDF event structure.

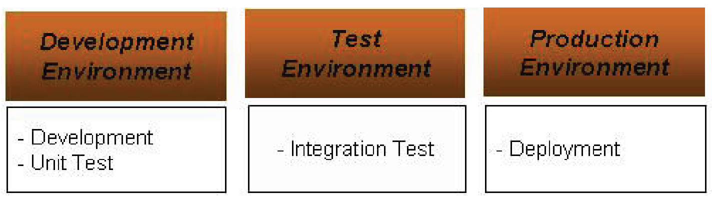
Once the data was available in the CDF, it was very easy to create the event structure as per the target system requirement and then simply map the CDF data into that structure.

Project Build, Test and Deployment

After an elaborate design an architecture phase, the project entered the build phase. It is during this period that the various integration components were developed by making use of the Seebeyond eGate software.

As seen in Figure 5, three different environments were used for this purpose.

Figure 5: Different Environments for Various Phases of the Project



Case Summary

The EAI solution implemented at Royal Wallace addressed all the integration requirements of the company. It provided for an automated procurement process in place of a manual one and seamlessly integrated all the legacy applications at the company with minimum fuss. The changes required in the existing scenario were kept to a minimum. Moreover, because the Common Data Format (CDF) provides a lot of flexibility to the model, any new application can be easily incorporated with a plug-and-play kind of adaptability.

Question

Summarize the case in your own language.

Sources: http://eai.ebizq.net and http://www.eaijournal.com

Notes

Case Study

Case 2: EAI Implementation

We appreciate Softima's contribution as a SI (System Integration) partner to webmethods and also as a reliable Vendor for Parkway Group. We command their participation and cooperation in the overall Implementation of Phase-I and Phase-II. Our experience working with them has been very smooth and we will be more than happy to recommend Softima's as a valuable partner to other clients.

The Client

The Client is one of the largest health care organizations in the world. They own a network of hospitals across the globe with an emphasis in Asia. The IT organization prides itself on being on the cutting edge of all technologies including ERP.

The Challenge

The Client has three major hospitals in Singapore and has different applications for each department. Their IT team had developed several systems and the remaining ones were purchased from different Vendors. A VB application (Interface Engine) was designed to send data from all systems to the SAP Health Care Module which was utilized by all departments as a central system. The need was to integrate various systems like IRIS, Cerner and Merlin.

Scope of the Project

To Replace the Interface Engine with webMethods Integration Platform and to Provide a Common Integration Framework -Re-architect Interface Engine using webMethods Integration Platform by Applying HL7 Health and to integrate them into the SAP HCM system.

First phase of the project involved integrating Laboratory System (Cerner) and Radiology System (IRIS) with SAP Health Care Module. Data format for communication was HCM (for SAP system) and HL7 (for IRIS & Cerner systems). After receiving the data from a system, webMethods parses, validates, logs and delivers the messages to corresponding receiver. Protocols used for this communication were TCP/IP, Directory and SAP RFC's.

Second phase of the project involved integrating Pharmacy (Merlin) with SAP Health Care Module by using webMethods as the Interface Engine. Data format for communication is HCM (for SAP), HL7 and Custom messages (for Merlin System). Protocols used for this integration were FTP, Directory, SAP RFC and SAP BAPI.

Contd...

Notes

Figure 1(a): System Architecture before EAI

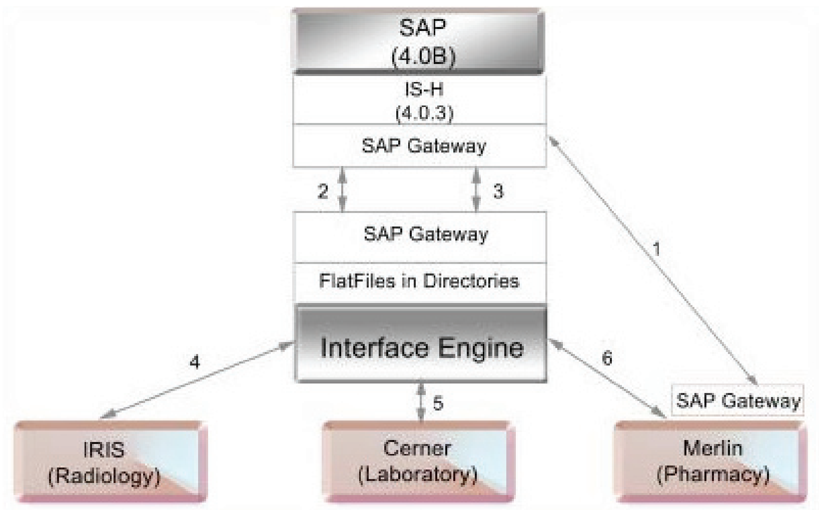
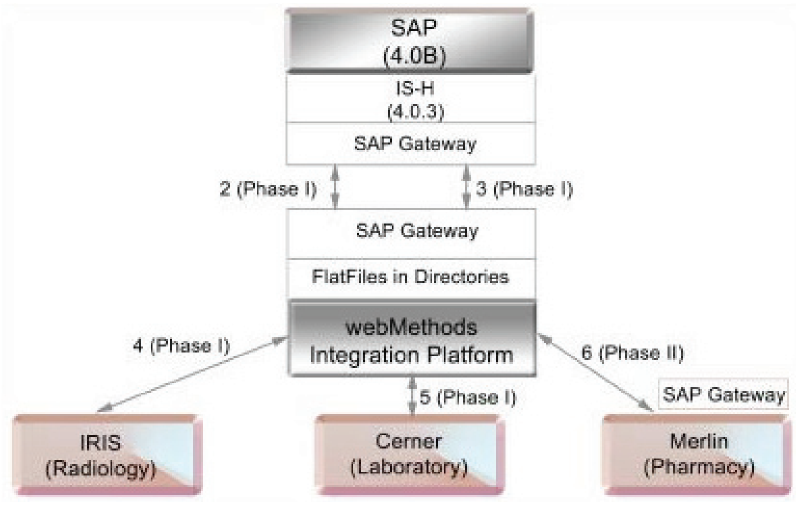


Figure 1(b): System Architecture after EAI



Question

EAI in the system should work properly. Give your suggestion.

Notes

Case Study

Case 3: Application Integration for MTNL

Mahanagar Telephone Nigam Ltd, had developed a portal for online presentation of bills to its customer. The application allows the MTNL customers to view their bills on the screen; the customer can in turn take a print out of the bills and present at any MTNL payment counters. The bills are printed with bar codes and hence the existing application of MTNL can read the bar codes and fetch the bills details online from the servers for accepting the payment. Besides bill viewing and printing; the customer can get all the other details like unpaid bills listing, paid bills listing, STD/ISD details and etc. The services is extremely fast, free and the customer just need to have an e-mail account to get the bills e-mail to them on a regular basis.

MTNL New Delhi needed an EAI tool to integrate seamlessly their existing remote billing server. There are currently 11 remote billing server (CSMS) and depending upon the telephone location, the customer database along with the bills are stored in the respective remote servers. EAI was needed to integrate the portal application with these remote billing servers. Under this all the bills location, fetching, updating the bills status was to be done by the EAI tool. MTNL, New Delhi chose Microsoft BizTalk Server 2002 as the EAI tool and Microsoft SQL Server 2000 Enterprise Edition as database for the portal application. Tender was floated by MTNL and COMM-IT India PVT LTD was given the order for the same. The scope covered in the job were:

1. Supply, Installation, demonstration and commissioning of EAI.
2. Training on the EAI application which also covered BizTalk Server 2002 training

Question

What are the benefits of application integration at MTNL?

Source: http://www.comm-it.in/form/Case_EnterpriseApplicationIntegration.aspx

Case Study

Case 4: Integrating and Automating Composite Applications

Enterprises are increasingly seeking new competitive advantages, and are demanding IT provide new applications in record times to meet those demands, and deliver strategic value.

Enterprise architects are therefore turning to a new breed of applications that use and reuse components developed with multiple technologies - including Web Services, J2EE, .NET, and messaging - across a wide range of platforms and systems. These composite applications are more efficient to develop, and deliver faster results to the business, but entail many new challenges such as where to focus development time, how to integrate legacy applications, and how to maximize limited compute resources.

Questions

Take any organization and apply integration tool in that it should work or not.

Its really helpful for organization or not give your suggestions.

Source: BMC Software

7.3 Summary

- There are many types of EAI software on the market (such as Sun Microsystems SeeBeyond), each approaching the problem of integration from a different angle and presenting a different solution. However, there are four overarching purposes for which EAI software can be used to improve efficiency.
- The integration problems many enterprises face today are due to the fact that until relatively recently there was no expectation that applications should be able to 'talk' to each other.
- Until the advent of networks, computer applications were designed to perform a specific purpose, and were often written in a range of different programming languages and used different data structures than each other, with no thought to integration.

7.4 Keywords

Auditing: Auditing is an evaluation of a person, organization, system, process, enterprise, project or product. Audits are performed to ascertain the validity and reliability of information; also to provide an assessment of a system's internal control.

Database: Database is a set of computer programs that controls the creation, maintenance, and the use of the database in a computer platform or of an organization and its end users.

EAI: Enterprise Application Integration is the term used to describe the integration of the computer applications of an enterprise so as to maximise their utility throughout the enterprise.

Operating System: An operating system (OS) is an interface between hardware and user which is responsible for the management and coordination of activities and the sharing of the resources of the computer that acts as a host for computing applications run on the machine.

7.5 Review Questions

1. What do you mean by enterprise application integration?
2. What are the purposes of EAI uses?
3. Describe the industry specific implementations of EAI.
4. What are the advantages and disadvantages of Integration?

7.6 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Notes

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 8: Data Warehouse Refreshment

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Know data warehouse refreshment
- Explain incremental data extraction
- Describe data cleaning

Introduction

A distinguishing characteristic of data warehouses is the temporal character of warehouse data, i.e., the management of histories over an extended period of time. Historical data is necessary for business trend analysis which can be expressed in terms of analysing the temporal development of real-time data. For the refreshment process, maintaining histories in the DWH means that either periodical snapshots of the corresponding operational data or relevant operational updates are propagated and stored in the warehouse, without overriding previous warehouse states.

Extraction is the operation of extracting data from a source system for further use in a data warehouse environment. This is the first step of the ETL process. After the extraction, this data can be transformed and loaded into the data warehouse.

The source systems for a data warehouse are typically transaction processing applications. For example, one of the source systems for a sales analysis data warehouse might be an order entry system that records all of the current order activities.

Notes

8.1 Data Warehouse Refreshment

The possibility of having "fresh data" in a warehouse is a key factor for success in business applications. In many activities such as in retail, business applications rely on the proper refreshment of their warehouses. For instance, Jahnke mentions in the case of WalMart, the world's most successful retailer. Many of WalMart's large volume suppliers such as Procter & Gamble have direct access to the WalMart data warehouse, so they deliver goods to specific stores as needed. WalMart pays such companies for their products only when they are sold. Procter & Gamble ships 40% of its items in this way eliminating paperwork and sale calls on both sides. It is essential for the supplier to use fresh data in order to establish accurate shipment plans and to know how much money is due from the retailer.

Refreshment Process within the Data Warehouse Lifecycle

The data warehouse can be defined as a hierarchy of data stores which goes from source data to highly aggregated data. Between these tow extremes can be other data stores depending on the requirements of OLAP applications. One of these stores in the Corporate Data Warehouse store (CDW) which groups all aggregated views used for the generation of the data marts. The corporate data store can be complemented by an Operational Data Store (ODS) which groups the base data collected and integrated from the sources. Data extracted from each source can also be stored in different data structures. This hierarchy of data stores is a logical way to represent the data flow between the sources and the data marts. In practice all the intermediate states between the source and the data marts can be represented in the same database.

Distinguish four levels in the construction of the hierarchy of stores. The first level includes three major steps:

1. The extraction of data from the operation data sources
2. Their cleaning with respect to the common rules defined for the data warehouse store.
3. Their possible archiving in the case when integration needs some synchronization between extraction.

Note However that this decomposition is only logical. The extraction step and part of cleaning step can be grouped into the same software component, such as a wrapper or a data migration tool.

When the extraction and cleaning steps are separated data need to be stored in between. This can be done using one storage medium per source or one shared medium for all sources.

The second level is the integration step. This phase is often coupled with rich data transformation capabilities into the same software component which usually performs the loading into the ODS when it exists or into the CDW. The third level concerns the data aggregation for the purpose of cubes construction. Finally the fourth level is a step of cube customization. All these steps can also be grouped into the same software such as multi-database system.

In order to understand which kind of tools the refreshment process needs, it is important to locate it within the global data warehouse lifecycle which is defined by three following phases:

Design Phase

The design phase consists of the definition of user views, auxiliary views, source extractors, data cleaners, data integrators and all others features that guarantee an explicit specification of the data warehouse application. These specifications could be done with respect to abstraction levels (conceptual, logical and physical) ad user perspectives (source view, enterprise view, client views). The result of the design is a set of formal or semiformal specification which constitutes the metadata used by the data warehouse system and applications.

Loading Phase

The loading phase consists of the initial data warehouse instantiation which is the initial computation of the data warehouse content. This initial loading is globally a sequential process of four steps:

1. Preparation
2. Integration
3. High level aggregation
4. Customization

The first step is done for each source and consists of data extraction, data cleaning and possibly data archiving before or after cleaning. The second step consists of data integration which is reconciliation of data originated from heterogeneous sources and derivation of the base relations of the ODS. The third step consists of the computation of aggregated views from base views. In all three steps not just the loading of data but also the loading of indexed is of crucial importance for query and update performance. While the data extracted from eth sources and integrated in the ODS are considered as ground data with very low-level aggregation the data in aggregated views are generally highly summarized using aggregation functions. These aggregated views constitute what is sometimes called the CDS, i.e. the set of materialized views from which data marts are derived. The fourth step consists of the derivation and customization of the user views which define the data marts. Customization refers to various presentations needed by the users for multidimensional data.

Refreshment Phase

The refreshment phase has a data flow similar to the loading phase but, while the loading process is a massive feeding of the data warehouse the refreshment process capture the differential changes that occurred in the sources and propagates them through the hierarchy of data stores. The preparation step extract from each source the data that characterize the changes that have occurred in this source since the last extraction. As for the loading phase, these data are cleaned and possibly archived before their integration. The integration step reconciles the source changes coming from multiple sources and adds them to the ODS. The aggregation step computes incrementally the hierarchy of aggregated views using these changes. The customization step propagates the summarized data to the data marts.

Requirements and Difficulties of Data Warehouse Refreshment

The refreshment of a data warehouse is an important process which determines the effective usability of the data collected and aggregated from the sources. Indeed the quality of data provided to the decision makers depends on the capability of the data warehouse system to propagate the changes made at the data sources in reasonable time. Most of the design decisions are then influenced by re choice of data structures and updating techniques that optimize the refreshment of the data warehouse.

Building an efficient refreshment strategy depends on various parameters related to the following:

1. ***Application requirements***: e.g., data freshness, computation time of queries and views, data accuracy
2. ***Source Constraints***: e.g., availability windows, frequency of change.
3. ***Data Warehouse System limits***: e.g., storage space limit, functional limits.

Most of these parameters may evolve during the data warehouse lifetime, hence leading to frequent reconfiguration of the data warehouse architecture and changes in the refreshment strategies. Consequently data warehouse administrators must be provided with powerful tools that enable them to efficiently redesign data warehouse applications.

For those corporations in which an ODS makes sense, Inmon proposes to distinguish among three classes of ODSs, depending on the speed of refreshment demanded.

1. The first class of ODSs is refreshed within a few seconds after the operational data sources are updated. Very little transformations are performed as the data passes form the operational environment into the ODS. A typical example of such an ODS is given by a banking environment where data sources keep individual accounts of a large multinational customer, and the ODS stores the total balance for this customer.
2. With the second class of ODSs integrated and transformed data are first accumulated and stored into an intermediate data store and then periodically forwarding to the ODS on say an hourly basis. This class usually involves more integration and transformation processing. To illustrate this consider now a bank that stores in the ODS an integrated individual bank account on a weekly basis, including the number of transactions during the week the starting and ending balances the largest and smallest transactions, etc. The daily transactions processed at the operational level are stored and forwarded on an hourly basis. Each change received by the ODS triggers the updating of a composite record o the current week.
3. Finally, the third class of ODSs is strongly asynchronous. Data are extracted from the sources and used to refresh the ODS on a day-or-more basis. As an example of this class, consider an ODS that stores composite customer records computed from different sources. As customer data change very slowly, it is reasonable to refresh ODS in a more infrequent fashion.

Quite similar distinctions also apply for the refreshment of a global data warehouse except that there is usually no counterpart for ODS of the first class. The period for refreshment is considered to be larger for global data warehouses. Nevertheless, different data warehouses demand different speed of refreshment. Besides the speed of the refreshment, which can be determined statically after analyzing the requirements of the information processing application other dynamic parameters may influence the refreshment strategy of the data warehouse. For instance one may consider the volume of changes in the data sources as given by the number of update transactions. Coming back to the previous example of an ODS of the second class, such a parameter may determine dynamically the moment at which the changes accumulated into an intermediate data store should be forwarded to the ODS. Another parameter can be determined by the profile to queries that execute on the data warehouse. Some strategic queries that require to use fresh data may entail the refreshment of the data warehouse for instance using the changes that have been previously logged between then sources and the ODS or the sources and the global data warehouse.

In any case, the refreshment of a data warehouse is considered to be a difficult and critical problem for three main reasons:

1. First the volume of data stored in a warehouse is usually large and is predicted to grow in the near future. Recent inquiries show that 100 GB warehouses are becoming commonplace. Also a study from META Group published in January 1996 reported that 52% of the warehouses surveyed would be 20 GB to 1 TB or larger in 12-18 months. In particular the level of detail required by the business leads to fundamentally new volumes of warehoused data. Further the refreshment process must be propagated along the various levels of data (ODS, CDW and data marts), which enlarges the volume of data must be refreshed.

2. Second the refreshment of warehouse requires the execution of transactional workloads of varying complexity. In fact, the refreshment of warehouses yields different performance challenges depending on its level in the architecture. The refreshment of an ODS involves many transactions that need to access and update a few records. Thus, the performance requirements for refreshment are those of general purpose record-level update processing. The refreshment of a global data warehouse involves heavy load and access transactions. Possibly large volumes of data are periodically loaded in the data warehouse, and once loaded, these data are accessed either for informational processing or for refreshing the local warehouses. Power for loading is now measured in GB per hour and several companies are moving to parallel architectures when possible to increase their processing power for loading and refreshment. The network interconnecting the data sources to the warehouse can also be bottleneck during refreshment and calls for compression techniques for data transmission. Finally, as a third reason the refreshment of local warehouses involves transactions that access many data perform complex calculations to produce highly summarized and aggregated data and update a few records in the local warehouses. This is particularly true for the local data warehouses that usually contain the data cubes manipulated by OLAP applications. Thus, a considerable processing time may be needed to refresh the warehouses. This is a problem because there is always a limited time frame during which the refreshment is expected to happen. Even if this time frame goes up to several hours and does not occur at peak periods it may be challenging to guarantee that the data warehouse will be refreshed within it.

3. Third, the refreshment of a warehouse may be run concurrently with the processing of queries. This may happen because the time frame during which the data warehouse is not queried is either too short or nonexistent.

Task The weather date is stored for different locations in a warehouse. The weather data consists of 'temperature,' 'pressure,' humidity,' and 'wind velocity.' The location is defined in terms of 'latitude,' 'longitude,' altitude' and 'time.' Assume that nation() is a function that returns the name of the country for a given latitude and longitude. Propose a warehousing model for this case.

8.2 Incremental Data Extraction

The way incremental data extraction can be implemented depends on the characteristics of the data sources and also on the desired functionality of the data warehouse system.

Data sources are heterogeneous and can include conventional database systems and nontraditional sources like flat files, XML and HTML documents, knowledge systems and legacy systems. The mechanisms offered by each data source to help the detection of changes are also quite heterogeneous.

It is convenient to associate a wrapper with the data source in order to provide a uniform description of the capabilities of the data sources. Moreover the role of the wrapper in a data warehouse context in enlarged. Its first functionality is to give a description of the data stored by each data source in a common data model. I assume that this common model is a relational data model. This is the typical functionality of a wrapper in a classical wrapper/mediator architecture therefore, I will call it wrapper functionality. The second functionality is to detect the changes of interest that have happened in the underlying data source. This is a specific functionality required by data warehouse architectures in order to support the refreshment of the data warehouse in a incremental way. For this reason I reserve the term change monitoring to refer to this kind of functionality.

Wrapper Functionality

The principal function of the wrapper relative to this functionality is to make the underlying data source appear as having the same data format and model that are used in the data warehouse system. For instance, if the data source is a set of XML document and the data model used in the data warehouse is the relational model, then the wrapper must be defined in such a way that it present the data sources of this type as it they were relational.

The development of wrapper generators has received attention from the research community especially in the case of sources that contain semi-structured data such as HTML or SGML documents. These tools for instance, enable to query the documents using an OQL-base interface.

Another important function that should be implemented by the wrapper is to establish the communication with the underlying data source and allow the transfer of information between the data source and the change monitor component. If the data warehouse system and the data source share the same data model then the function of the wrapper would be just to translate the data format and to support the communication with the data source. For data sources that are relational system and supposing that the data model used in the data warehouse is also relational it is possible to use wrappers that have been developed by software companies such as database vendors or database independent companies. These wrappers also called "middleware", "gateways" or "brokers" have varying capabilities in terms of application programming interface, performance and extensibility.

In the client server database environment several kinds of middleware have already been developed to enable the exchange queries and their associated answers between a client application and a database server, or between database servers in a transparent way. The term "transparent" usually means that the middleware hide the underlying network protocol, the database systems and the database query languages supported by these database systems from the application.

The usual sequence of steps during the interaction of a client application and a database server through a middleware agent is as follows. First, the middleware enables the application to connect and disconnect to the database server. Then, it allows the preparation and execution of requests. A request preparation specifies the request with formal parameters which generally entails its compilation in the server. A prepared request can then be executed by invoking its name and passing its actual parameters. Requests are generally expressed in SQL. Another functionality offered by middleware is the fetching of results which enables a client application to get back all or part of the result of a request. When the results are large, they can be cached on the serve. The transfer of requests and results is often built on a protocol supporting remote procedure calls.

There has been an important effort to standardize the programming interface offered by middleware and the underlying communication protocol. Call Level Interface (CLI) is a standardized API developed by the X/Open standardization committee. It enables a client application to extract data from a relational database server through a standard SQL-based interface. This API is currently supported by several middleware products such as ODBC and IDAPI. The RDA standard communication protocol specifies the messages to be exchanged between clients and servers. Its specialization to SQL requests enables the transport of requests generated by a CLI interface.

Despite these efforts, existing middleware products do not actually offer a standard interface for client-server developments. Some products such as DAL/DAM or SequeLink offer their own API, although some compatibility is sometimes offered with other tools, such as ODBC. Furthermore, database vendors have developed their own middleware. For instance, Oracle proposes several levels of interface, such as Oracle Common Interface (OCI), on top of its client-server protocol named SQL*Net. The OCI offers a set of functions close to the ones of CLI, and enables any client having SQL*NET to connect to an Oracle server using any kind of communication protocol.

Finally the alternative way to provide a transparent access to database servers is to use Internet protocols. In fact it must be noted that the World Wide Web is simply a standard-based client-server architecture.

8.3 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 over fitting 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:

8.3.1 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.

8.3.2 Noisy Data

Noise is a random error or variance in a measured variable. Given a numeric attribute such as, say, price, we can "smooth" out the data by using the following techniques:

Binning Methods

Binning methods smooth a sorted data value by consulting the "neighborhood", or values around it. The sorted values are distributed into a number of 'buckets' or bins. Because binning methods consult the neighborhood of values, they perform local smoothing. The following example illustrates some binning techniques. In this example, the data for price are first sorted and partitioned into equi-depth bins (of depth 3).

1. In smoothing by bin means, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 4, 8, and 15 in Bin 1 is 9. Therefore, each original value in this bin is replaced by the value 9.
2. Similarly, smoothing by bin medians can be employed, in which each bin value is replaced by the bin median.
3. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value. In general, the larger the width, the greater the effect of the smoothing. Alternatively, bins may be equi-width, where the interval range of values in each bin is constant.

Example

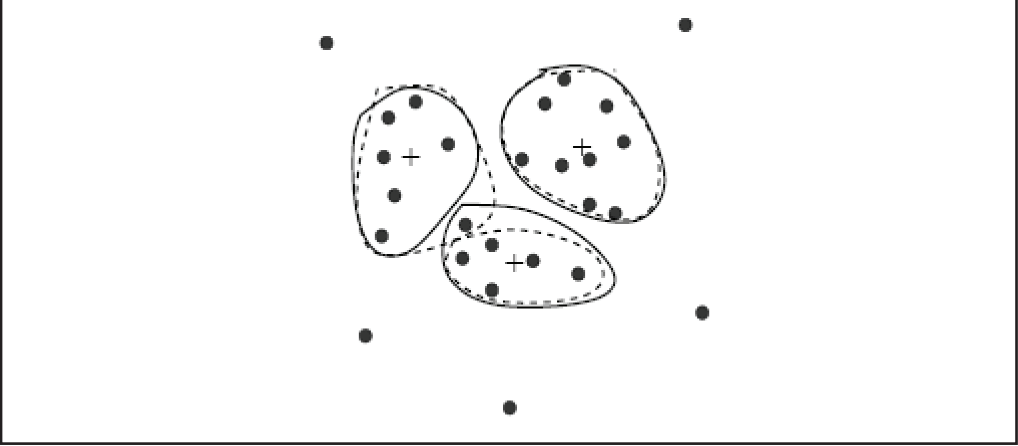
1. Sorted data for price (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34
2. Partition into (equi-width) bins:
   (a) Bin 1: 4, 8, 15
   (b) Bin 2: 21, 21, 24
   (c) Bin 3: 25, 28, 34
3. Smoothing by bin means:
   (a) Bin 1: 9, 9, 9,
   (b) Bin 2: 22, 22, 22
   (c) Bin 3: 29, 29, 29
4. Smoothing by bin boundaries:
   (a) Bin 1: 4, 4, 15
   (b) Bin 2: 21, 21, 24
   (c) Bin 3: 25, 25, 34

Clustering

Outliers may be detected by clustering, where similar values are organized into groups or "clusters". Intuitively, values which fall outside of the set of clusters may be considered outliers (Figure 8.1).

Notes

Figure 8.1: Outliers may be detected by Clustering Analysis



Combined Computer and Human Inspection

Outliers may be identified through a combination of computer and human inspection. In one application, for example, an information-theoretic measure was used to help identify outlier patterns in a handwritten character database for classification. The measure's value reflected the "surprise" content of the predicted character label with respect to the known label. Outlier patterns may be informative (e.g., identifying useful data exceptions, such as different versions of the characters "0" or "7"), or "garbage" (e.g., mislabeled characters). Patterns whose surprise content is above a threshold are output to a list. A human can then sort through the patterns in the list to identify the actual garbage ones.

This is much faster than having to manually search through the entire database. The garbage patterns can then be removed from the (training) database.

Regression

Data can be smoothed by fitting the data to a function, such as with regression. Linear regression involves finding the "best" line to fit two variables, so that one variable can be used to predict the other. Multiple linear regression is an extension of linear regression, where more than two variables are involved and the data are fit to a multidimensional surface. Using regression to find a mathematical equation to fit the data helps smooth out the noise.

Task Discuss for the integration of multiple heterogeneous information sources, many companies in industry prefer the update-driven approach rather than query driven approach.

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.

Notes

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

Case Study

Banco Popular

The Company

Banco Popular, the largest bank in Puerto Rico, with more then 2 million customers, recognizes that effective service requires understanding individual needs and responding to them promptly. That's why, for the past 17 years, it has used a customer information system to keep all its employees abreast of its customer relationships.

The bank has achieved steady growth with this approach, but until recently, success had taken a toll on data system performance. As the bank added more and more customer and account data to the system, sluggish response times and inconsistent, duplicated, and nonstandardized data values made it increasingly difficult for employees to access information.

Open Database is Key to Customer Data System Success

By replacing the old system with a new DB2-based system, the bank has built a solid foundation for staying in tune with its clientele. The customer data system is integrated with all other applications within the bank, and, working in conjunction with data cleansing and reengineering software from Trillium Software, provides a complete, unified view of each customer. A program analyst at Banco Popular said, "Opening a new account once required several operations. Now it is completed in just one menu function. The DB2-based system has improved both customer satisfaction and productivity."

Residing on an IBM platform, the system is the hub of activity at the bank. As many as 50 percent of the 6,000 employees in more then 200 branches depend on it to manage more than 5.7 million personal and business accounts. Every transaction that requires accessing, updating, or collecting information on customers is handled by the customer data system. For example, before opening a new account, customer service associates access the system to research the customer's existing relationships with the bank.

Any new customer and account information is then fed into the system, where it can be used by other divisions of the bank.

Because of the key role the customer data system plays in growing the bank's business, Banco Popular insisted that the database at the heart of its new system be both robust and open. The bank understood it needed a relational database. The main advantage of DB2 among relational databases is that companies can implement any third-party application on top of it and it will meet expectations. Scalability was equally important to Banco Popular. In the past, it had been constrained by a system that was unable to keep up with growing data volumes and number of users. DB2 also gave the bank virtually limitless scalability as well as high performance.

The bank is confident that its needs will be fulfilled for many years.

The Trillium Software System is used to cleanse and standardize each customer record and then to identify and match those records against the database.

Banco Popular took advantage of Trillium's highly customizable business rules to eliminate duplicate records and develop an accurate view of its current customers and their relationships with the bank. This has helped the bank establish meaningful and profitable relationships with its customers.

Contd...

The data cleansing process is also helping reduce marketing costs. Banco Popular will use the Trillium Software System® to enforce standardization and cleanse data. This will be the key to an accurate "householding" process, which is a way of identifying how many account holders live at the same address.

By doing this, the bank can eliminate duplicate mailings to the same household, which makes the bank look much more efficient in its customers' eyes, and saves at least $70,000 in mailing expenses every month. Banco Popular's home-grown address standardization system will soon be replaced by Trillium Software's geocoding solution. This will save the cost of changing and recertifying the system each time the US Postal Service changes its standardization requirements.

DB2 can easily handle customer information systems containing millions of records in multiple languages that are initially cleansed with the Trillium Software System. Not only is Banco Popular expanding, its customers may be represented within complex financial records on the database in either English or Spanish. Trillium Software scales in step with the growing DB2 database and works in numerous languages to provide a global solution for this multinational bank.

8.4 Summary

- DWH refreshment so far has been investigated in the research community mainly in relation to techniques for maintaining materialized views.
- In these approaches, the DWH is considered as a set of materialized views defined over operational data. Thus, the topic of warehouse refreshment is defined as a problem of updating a set of views (the DWH) as a result of modifications of base relations (residing in operational systems). Several issues have been investigated in this context.
- The extraction method you should choose is highly dependent on the source system and also from the business needs in the target data warehouse environment.
- Very often, there is no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems.
- Sometimes even the customer is not allowed to add anything to an out-of-the-box application system.

8.5 Keywords

Corporate Data Store: The corporate data store can be complemented by an Operational Data Store (ODS) which groups the base data collected and integrated from the sources.

Data Cleaning: Data cleaning can be applied to remove noise and correct inconsistencies in the data.

Incremental Data Extraction: Incremental data extraction can be implemented depends on the characteristics of the data sources and also on the desired functionality of the data warehouse system.

The Design Phase: The design phase consists of the definition of user views, auxiliary views, source extractors, data cleaners, data integrators.

8.6 Self Assessment

Fill in the blanks:

1. Recent inquiries show that .... warehouses are becoming commonplace.
2. The refreshment of an .... many transactions that need to access and update a few records.
3. .... are heterogeneous and can include conventional database systems and nontraditional sources like flat files, XML and HTML documents.
4. .... is a standardized API developed by the X/Open standardization committee.
5. Replace all missing attribute values by the same constant, such as a label like ....
6. .... is a random error or variance in a measured variable.
7. .... may be detected by clustering, where similar values are organized into groups or "clusters".
8. Some data inconsistencies may be corrected manually using .... references.
9. The .... computes incrementally the hierarchy of aggregated views using these changes.
10. Power for loading is now measured in .... per hour and several companies are moving to parallel architectures when possible to increase their processing power for loading and refreshment.

8.7 Review Questions

1. Which data you call inconsistent data? Explain with suitable example.
2. Describe data refreshment process in detail.
3. Explain loading phase of data refreshment.
4. What are the major difficulties generally face in data warehouse refreshment?
5. Describe incremental data extraction.
6. "Dirty data can cause confusion for the mining procedure." Explain.
7. "The refreshment of a data warehouse is an important process which determines the effective usability of the data collected and aggregated from the sources." Discuss.
8. "The period for refreshment is considered to be larger for global data warehouses." Why
9. "Outliers may be identified through a combination of computer and human inspection." Explain
10. "Data cleaning can be applied to remove noise and correct inconsistencies in the data". Discuss

Answers: Self Assessment

1. 100 GB	2. ODS involves
3. Data sources	4. Call Level Interface (CLI)
5. Unknown	6. Noise

7. Outliers
8. external
9. aggregation step
10. GB

8.8 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 9: Data Warehouse Refreshment – II

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain update propagation into materialized views
- Know towards a quality-oriented refreshment process
- Describe implementation of the approach

Introduction

Your organization has decided to build a data warehouse. You have defined the business requirements and agreed upon the scope of your application, and created a conceptual design. Now you need to translate your requirements into a system deliverable. To do so, you create the logical and physical design for the data warehouse.

The logical design is more conceptual and abstract than the physical design. In the logical design, you look at the logical relationships among the objects. In the physical design, you look at the most effective way of storing and retrieving the objects as well as handling them from a transportation and backup/recovery perspective.

9.1 Update Propagation into Materialized Views

Typically, data flows from one or more online transaction processing (OLTP) database into a data warehouse on a monthly, weekly, or daily basis. The data is normally processed in a staging file before being added to the data warehouse. Data warehouses commonly range in size from tens of gigabytes to a few terabytes. Usually, the vast majority of the data is stored in a few very large fact tables.

One technique employed in data warehouses to improve performance is the creation of summaries. Summaries are special types of aggregate views that improve query execution times by pre-calculating expensive joins and aggregation operations prior to execution and storing the results in a table in the database.

Example: You can create a table to contain the sums of sales by region and by product.

The summaries or aggregates that are referred to in this book and in literature on data warehousing are created in Oracle Database using a schema object called a *materialized view*. Materialized views can perform a number of roles, such as improving query performance or providing replicated data.

In the past, organizations using summaries spent a significant amount of time and effort creating summaries manually, identifying which summaries to create, indexing the summaries, updating them, and advising their users on which ones to use. The introduction of summary management eased the workload of the database administrator and meant the user no longer needed to be aware of the summaries that had been defined. The database administrator creates one or more materialized views, which are the equivalent of a summary. The end user queries the tables and views at the detail data level. The query rewrite mechanism in the Oracle server automatically rewrites the SQL query to use the summary tables. This mechanism reduces response time for returning results from the query. Materialized views within the data warehouse are transparent to the end user or to the database application.

Although materialized views are usually accessed through the query rewrite mechanism, an end user or database application can construct queries that directly access the materialized views. However, serious consideration should be given to whether users should be allowed to do this because any change to the materialized views will affect the queries that reference them.

Materialized Views for Data Warehouses

In data warehouses, you can use materialized views to pre-compute and store aggregated data such as the sum of sales. Materialized views in these environments are often referred to as summaries, because they store summarized data. They can also be used to pre-compute joins with or without aggregations. A materialized view eliminates the overhead associated with expensive joins and aggregations for a large or important class of queries.

Materialized Views for Distributed Computing

In distributed environments, you can use materialized views to replicate data at distributed sites and to synchronize updates done at those sites with conflict resolution methods. The materialized views as replicas provide local access to data that otherwise would have to be accessed from remote sites. Materialized views are also useful in remote data marts.

Materialized Views for Mobile Computing

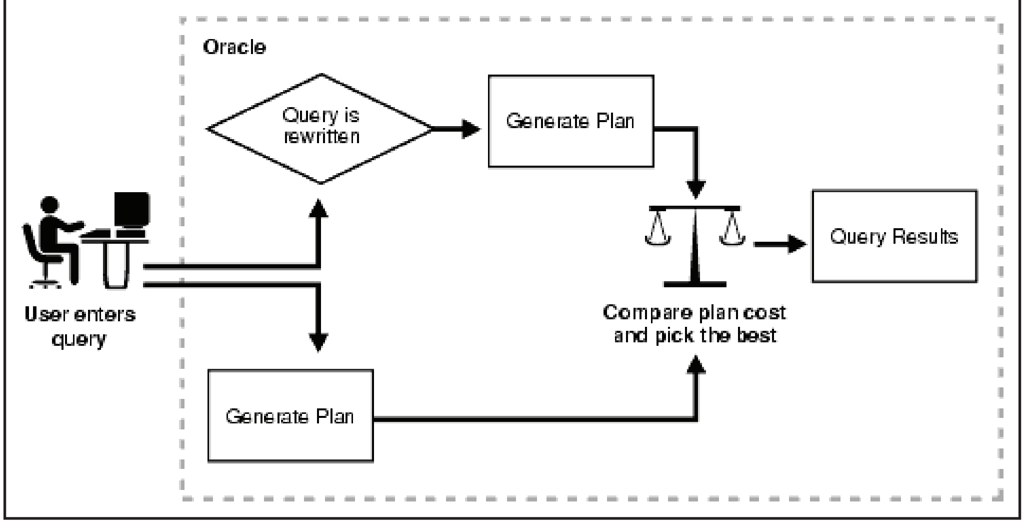
You can also use materialized views to download a subset of data from central servers to mobile clients, with periodic refreshes and updates between clients and the central servers.

The Need for Materialized Views

You can use materialized views to increase the speed of queries on very large databases. Queries to large databases often involve joins between tables, aggregations such as SUM, or both. These operations are expensive in terms of time and processing power. The type of materialized view you create determines how the materialized view is refreshed and used by query rewrite.

Materialized views improve query performance by pre-calculating expensive join and aggregation operations on the database prior to execution and storing the results in the database. The query optimizer automatically recognizes when an existing materialized view can and should be used to satisfy a request. It then transparently rewrites the request to use the materialized view. Queries go directly to the materialized view and not to the underlying detail tables. In general, rewriting queries to use materialized views rather than detail tables improves response. Figure 9.1 illustrates how query rewrite works.

Figure 9.1: Transparent Query Rewrite



When using query rewrite, create materialized views that satisfy the largest number of queries. For example, if you identify 20 queries that are commonly applied to the detail or fact tables, then you might be able to satisfy them with five or six well-written materialized views. A materialized view definition can include any number of aggregations (SUM, COUNT(x), COUNT(*), COUNT(DISTINCT x), AVG, VARIANCE, STDDEV, MIN, and MAX). It can also include any number of joins. If you are unsure of which materialized views to create, Oracle provides the SQLAccess Advisor, which is a set of advisory procedures in the DBMS_ADVISOR package to help in designing and evaluating materialized views for query rewrite.

If a materialized view is to be used by query rewrite, it must be stored in the same database as the detail tables on which it relies. A materialized view can be partitioned, and you can define a materialized view on a partitioned table. You can also define one or more indexes on the materialized view.

Unlike indexes, materialized views can be accessed directly using a SELECT statement. However, it is recommended that you try to avoid writing SQL statements that directly reference the materialized view, because then it is difficult to change them without affecting the application. Instead, let query rewrite transparently rewrite your query to use the materialized view.

Note The techniques shown in this unit illustrate how to use materialized views in data warehouses. Materialized views can also be used by Oracle Replication.

9.2 Types of Materialized Views

The SELECT clause in the materialized view creation statement defines the data that the materialized view is to contain. Only a few restrictions limit what can be specified. Any number of tables can be joined together. However, they cannot be remote tables if you wish to take advantage of query rewrite. Besides tables, other elements such as views, inline views (subqueries in the FROM clause of a SELECT statement), subqueries, and materialized views can all be joined or referenced in the SELECT clause. You cannot, however, define a materialized with a subquery in the select list of the defining query. You can, however, include subqueries elsewhere in the defining query, such as in the WHERE clause.

The types of materialized views are:

1. Materialized Views with Aggregates
2. Materialized Views Containing Only Joins
3. Nested Materialized Views

9.2.1 Materialized Views with Aggregates

In data warehouses, materialized views normally contain aggregates as shown in Example 9.1. For fast refresh to be possible, the SELECT list must contain all of the GROUP BY columns (if present), and there must be a COUNT(*) and a COUNT(column) on any aggregated columns. Also, materialized view logs must be present on all tables referenced in the query that defines the materialized view. The valid aggregate functions are: SUM, COUNT(x), COUNT(*), AVG, VARIANCE, STDDEV, MIN, and MAX, and the expression to be aggregated can be any SQL value expression.

Fast refresh for a materialized view containing joins and aggregates is possible after any type of DML to the base tables (direct load or conventional INSERT, UPDATE, or DELETE). It can be defined to be refreshed ON COMMIT or ON DEMAND. A REFRESH ON COMMIT materialized view will be refreshed automatically when a transaction that does DML to one of the materialized view's detail tables commits. The time taken to complete the commit may be slightly longer than usual when this method is chosen. This is because the refresh operation is performed as part of the commit process. Therefore, this method may not be suitable if many users are concurrently changing the tables upon which the materialized view is based.

Here are some examples of materialized views with aggregates. Note that materialized view logs are only created because this materialized view will be fast refreshed.

Example: Creating a Materialized View

```
CREATE MATERIALIZED VIEW LOG ON products WITH SEQUENCE, ROWID
(prod_id, prod_name, prod_desc, prod_subcategory, prod_subcategory_desc,
prod_category, prod_category_desc, prod_weight_class, prod_unit_of_measure,
 prod_pack_size, supplier_id, prod_status, prod_list_price, prod_min_price)
INCLUDING NEW VALUES;
```

Notes

```
CREATE MATERIALIZED VIEW LOG ON sales
WITH SEQUENCE, ROWID
(prod_id, cust_id, time_id, channel_id, promo_id, quantity_sold, amount_sold)
INCLUDING NEW VALUES;
CREATE MATERIALIZED VIEW product_sales_mv
PCTFREE 0 TABLESPACE demo
STORAGE (INITIAL 8k NEXT 8k PCTINCREASE 0)
BUILD IMMEDIATE
REFRESH FAST
ENABLE QUERY REWRITE
AS SELECT p.prod_name, SUM(s.amount_sold) AS dollar_sales,
COUNT(*) AS cnt, COUNT(s.amount_sold) AS cnt_amt
FROM sales s, products p
WHERE s.prod_id = p.prod_id GROUP BY p.prod_name;
```

This example creates a materialized view product_sales_mv that computes total number and value of sales for a product. It is derived by joining the tables sales and products on the column prod_id. The materialized view is populated with data immediately because the build method is immediate and it is available for use by query rewrite. In this example, the default refresh method is FAST, which is allowed because the appropriate materialized view logs have been created on tables product and sales.

Example: Creating a Materialized View

```
CREATE MATERIALIZED VIEW product_sales_mv
PCTFREE 0 TABLESPACE demo
STORAGE (INITIAL 16k NEXT 16k PCTINCREASE 0)
BUILD DEFERRED
REFRESH COMPLETE ON DEMAND
ENABLE QUERY REWRITE AS
SELECT p.prod_name, SUM(s.amount_sold) AS dollar_sales
FROM sales s, products p WHERE s.prod_id = p.prod_id
GROUP BY p.prod_name;
```

This example creates a materialized view product_sales_mv that computes the sum of sales by prod_name. It is derived by joining the tables sales and products on the column prod_id. The materialized view does not initially contain any data, because the build method is DEFERRED. A complete refresh is required for the first refresh of a build deferred materialized view. When it is refreshed and once populated, this materialized view can be used by query rewrite.

Requirements for using Materialized Views with Aggregates

Table 9.1 Illustrates the aggregate requirements for materialized views.

Table 9.1: Requirements for Materialized Views with Aggregates

If aggregate X is present, aggregate Y is required and aggregate Z is optional		
X	Y	Z
COUNT(expr)	-	-
SUM(expr)	COUNT(expr)	-
AVG(expr)	COUNT(expr)	SUM(expr)
STDDEV(expr)	COUNT(expr) SUM(expr)	SUM(expr * expr)
VARIANCE(expr)	COUNT(expr) SUM(expr)	SUM(expr * expr)

Note COUNT(*) must always be present to guarantee all types of fast refresh. Otherwise, you may be limited to fast refresh after inserts only. Oracle recommends that you include the optional aggregates in column Z in the materialized view in order to obtain the most efficient and accurate fast refresh of the aggregates.

Task Suppose that a data warehouse for a university consists of the following for dimension: Student, course, Semester and Instructor, and two measures count and avg _grade. When at the lowest conceptual level (e.g. for a given student, course, semester and instructor combination), the avg_grade measure stores the actual course grade of the student at the higher conceptual levels, avg_grade stores the average grade for the given combination.

1. Draw a snowflake schema diagram for the data warehouse.
2. Starting with base cuboids, what specific OLAP operations should one perform in order to list the average grade of CS courses for each student.
3. If each dimension has five levels (including all), such as student <major <status< university<all, how many cuboids will this cube contain (including the base and appes cuboids)

9.2.2 Materialized Views Containing Only Joins

Some materialized views contain only joins and no aggregates, such as in example, where a materialized view is created that joins the sales table to the times and customers tables. The advantage of creating this type of materialized view is that expensive joins will be precalculated.

Fast refresh for a materialized view containing only joins is possible after any type of DML to the base tables (direct-path or conventional INSERT, UPDATE, or DELETE).

A materialized view containing only joins can be defined to be refreshed ON COMMIT or ON DEMAND. If it is ON COMMIT, the refresh is performed at commit time of the transaction that does DML on the materialized view's detail table.

If you specify REFRESH FAST, Oracle performs further verification of the query definition to ensure that fast refresh can be performed if any of the detail tables change. These additional checks are:

1. A materialized view log must be present for each detail table and the ROWID column must be present in each materialized view log.
2. The rowids of all the detail tables must appear in the SELECT list of the materialized view query definition.
3. If there are no outer joins, you may have arbitrary selections and joins in the WHERE clause. However, if there are outer joins, the WHERE clause cannot have any selections. Further, if there are outer joins, all the joins must be connected by ANDs and must use the equality (=) operator.
4. If there are outer joins, unique constraints must exist on the join columns of the inner table. For example, if you are joining the fact table and a dimension table and the join is an outer join with the fact table being the outer table, there must exist unique constraints on the join columns of the dimension table.

If some of these restrictions are not met, you can create the materialized view as REFRESH FORCE to take advantage of fast refresh when it is possible. If one of the tables did not meet all of the criteria, but the other tables did, the materialized view would still be fast refreshable with respect to the other tables for which all the criteria are met.

Materialized Join Views FROM Clause Considerations

If the materialized view contains only joins, the ROWID columns for each table (and each instance of a table that occurs multiple times in the FROM list) must be present in the SELECT list of the materialized view.

If the materialized view has remote tables in the FROM clause, all tables in the FROM clause must be located on that same site. Further, ON COMMIT refresh is not supported for materialized view with remote tables. Materialized view logs must be present on the remote site for each detail table of the materialized view and ROWID columns must be present in the SELECT list of the materialized view.

To improve refresh performance, you should create indexes on the materialized view's columns that store the rowids of the fact table.

Example: Materialized View Containing Only Joins

```
CREATE MATERIALIZED VIEW LOG ON sales WITH ROWID;

CREATE MATERIALIZED VIEW LOG ON times WITH ROWID;

CREATE MATERIALIZED VIEW LOG ON customers WITH ROWID;

CREATE MATERIALIZED VIEW detail_sales_mv

PARALLEL BUILD IMMEDIATE

REFRESH FAST AS

SELECT s.rowid “sales_rid”, t.rowid “times_rid”, c.rowid “customers_rid”,

 c.cust_id, c.cust_last_name, s.amount_sold, s.quantity_sold, s.time_id

FROM sales s, times t, customers c

WHERE s.cust_id = c.cust_id(+) AND s.time_id = t.time_id(+);
```

In this example, to perform a fast refresh, UNIQUE constraints should exist on c.cust_id and t.time_id. You should also create indexes on the columns sales_rid, times_rid, and customers_rid, as illustrated in the following. This will improve the refresh performance.

CREATE INDEX mv_ix_salesrid ON detail_sales_mv("sales_rid");

Alternatively, if the previous example did not include the columns times_rid and customers_rid, and if the refresh method was REFRESH FORCE, then this materialized view would be fast refreshable only if the sales table was updated but not if the tables times or customers were updated.

```
CREATE MATERIALIZED VIEW detail_sales_mv
PARALLEL
BUILD IMMEDIATE
REFRESH FORCE AS
SELECT s.rowid "sales_rid", c.cust_id, c.cust_last_name, s.amount_sold,
 s.quantity_sold, s.time_id
FROM sales s, times t, customers c
WHERE s.cust_id = c.cust_id(+) AND s.time_id = t.time_id(+);
```

9.2.3 Nested Materialized Views

A nested materialized view is a materialized view whose definition is based on another materialized view. A nested materialized view can reference other relations in the database in addition to referencing materialized views.

Why use Nested Materialized Views?

In a data warehouse, you typically create many aggregate views on a single join (for example, rollups along different dimensions). Incrementally maintaining these distinct materialized aggregate views can take a long time, because the underlying join has to be performed many times.

Using nested materialized views, you can create multiple single-table materialized views based on a joins-only materialized view and the join is performed just once. In addition, optimizations can be performed for this class of single-table aggregate materialized view and thus refresh is very efficient.

Example: Nested Materialized View

You can create a nested materialized view on materialized views that contain joins only or joins and aggregates. All the underlying objects (materialized views or tables) on which the materialized view is defined must have a materialized view log. All the underlying objects are treated as if they were tables. In addition, you can use all the existing options for materialized views.

Using the tables and their columns from the sh sample schema, the following materialized views illustrate how nested materialized views can be created.

```
CREATE MATERIALIZED VIEW LOG ON sales WITH ROWID;
CREATE MATERIALIZED VIEW LOG ON customers WITH ROWID;
CREATE MATERIALIZED VIEW LOG ON times WITH ROWID;
```

Notes

```
/*create materialized view join_sales_cust_time as fast refreshable at
 COMMIT time */
CREATE MATERIALIZED VIEW join_sales_cust_time
REFRESH FAST ON COMMIT AS
SELECT c.cust_id, c.cust_last_name, s.amount_sold, t.time_id,
 t.day_number_in_week, s.rowid srid, t.rowid trid, c.rowid crid
FROM sales s, customers c, times t
WHERE s.time_id = t.time_id AND s.cust_id = c.cust_id;
```

To create a nested materialized view on the table join_sales_cust_time, you would have to create a materialized view log on the table. Because this will be a single-table aggregate materialized view on join_sales_cust_time, you need to log all the necessary columns and use the INCLUDING NEW VALUES clause.

```
/* create materialized view log on join_sales_cust_time */
CREATE MATERIALIZED VIEW LOG ON join_sales_cust_time
WITH ROWID (cust_last_name, day_number_in_week, amount_sold)
INCLUDING NEW VALUES;
/* create the single-table aggregate materialized view sum_sales_cust_time on
 join_sales_cust_time as fast refreshable at COMMIT time */
CREATE MATERIALIZED VIEW sum_sales_cust_time
REFRESH FAST ON COMMIT AS
SELECT COUNT(*) cnt_all, SUM(amount_sold) sum_sales, COUNT(amount_sold)
 cnt_sales, cust_last_name, day_number_in_week
FROM join_sales_cust_time
GROUP BY cust_last_name, day_number_in_week;
```

Nesting Materialized Views with Joins and Aggregates

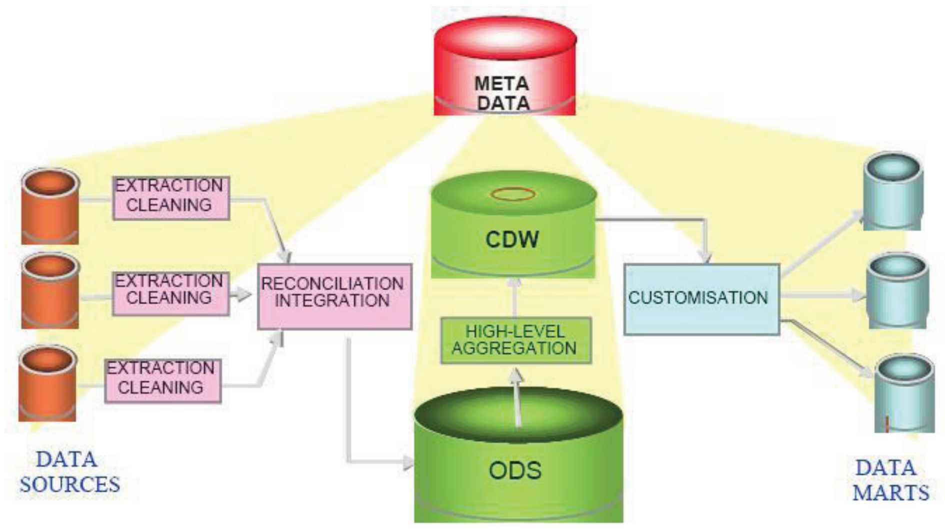
Some types of nested materialized views cannot be fast refreshed. Use EXPLAIN_MVIEW to identify those types of materialized views. You can refresh a tree of nested materialized views in the appropriate dependency order by specifying the nested = TRUE parameter with the DBMS_MVIEW. REFRESH parameter. For example, if you call DBMS_MVIEW.REFRESH ('SUM_SALES_CUST_TIME', nested => TRUE), the REFRESH procedure will first refresh the join_sales_cust_time materialized view, and then refresh the sum_sales_cust_time materialized view.

9.3 Towards a Quality-oriented Refreshment Process

Data warehousing is a new technology which provides software infrastructure for decision support systems and OLAP applications. Data warehouses collect data from heterogeneous and distributed sources. This data is aggregated and then customized with respect to organizational criteria defined by OLAP applications. The data warehouse can be defined as a hierarchy of data stores which goes from source data to the highly aggregated data (data marts). Between these two extreme data stores, we can find different other stores depending on the requirements of OLAP applications. One of these stores is the operational data store which reflects source data in a uniform and clean representation. The corporate data warehouse (CDW) contains highly aggregated data and can be organized into a multidimensional structure. Data extracted from each source can also be stored in intermediate data recipients. Obviously, this hierarchy of data stores is a logical way to represent the data flows which go from the sources to the data marts. All these stores are not necessarily materialized, and if they are, they can just constitute different layers of the same database.

Figure 9.2 shows a typical data warehouse architecture. This is a logical view whose operational implementation receives many different answers in the data warehousing products. Depending on each data source, extraction and cleaning can be done by the same wrapper or by distinct tools. Similarly data reconciliation (also called multi-source cleaning) can be separated from or merged with data integration (multi-sources operations). High level aggregation can be seen as a set of computation techniques ranging from simple statistical functions to advanced data mining algorithms. Customisation techniques may vary from one data mart to another, depending on the way decision makers want to see the elaborated data.

Figure 9.2: Data Warehouse Architecture



The refreshment of a data warehouse is an important process which determines the effective usability of the data collected and aggregated from the sources. Indeed, the quality of data provided to the decision makers depends on the capability of the data warehouse system to convey in a reasonable time, from the sources to the data marts, the changes made at the data sources. Most of the design decisions are then concerned by the choice of data structures and update techniques that optimise the refreshment of the data warehouse.

There is a quiet great confusion in the literature concerning data warehouse refreshment. Indeed, this process is often either reduced to view maintenance problem or confused with the data loading phase. Our purpose in this paper is to show that the data warehouse refreshment is a more complex than the view maintenance problem, and different from the loading process. We define the refreshment process as a workflow whose activities depend on the available products for data extraction, cleaning and integration, and whose triggering events of these activities depend on the application domain and on the required quality in terms of data freshness.

9.3.1 View Maintenance, Data Loading and Data Refreshment

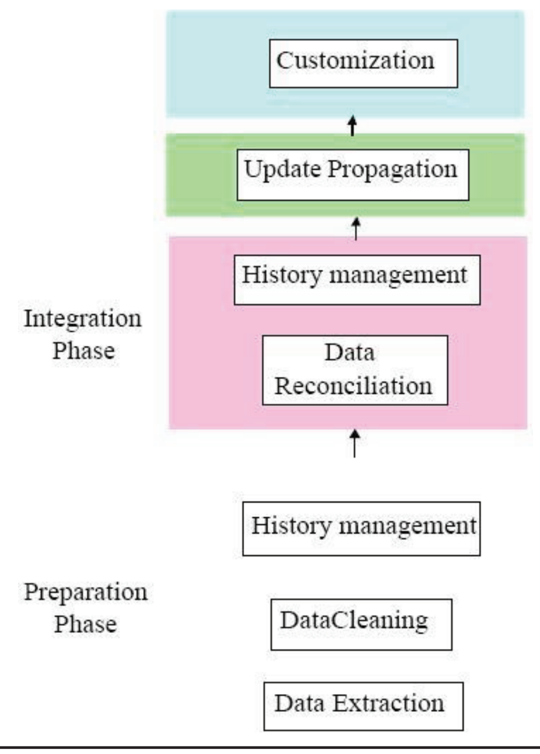
Data refreshment in data warehouses is generally confused with data loading as done during the initial phase or with update propagation through a set of materialized views. Both analogies are wrong. The following paragraphs argument on the differences between data loading and data refreshment, and between view maintenance and data refreshment.

Notes

Data Loading vs. Data Refreshment

The data warehouse loading phase consists in the initial data warehouse instantiation, that is the initial computation of the data warehouse content. This initial loading is globally a sequential process of four steps (Figure 9.3): (i) preparation, (ii) integration, (iii) high level aggregation and (iv) customisation. The first step is done for each source and consists in data extraction, data cleaning and possibly data archiving before or after cleaning. Archiving data in a history can be used both for synchronisation purpose between sources having different access frequencies and for some specific temporal queries. The second step consists in data reconciliation and integration, that is cleaning multi-source cleaning of data originated from heterogeneous sources, and derivation of the base relations (or base views) of the operational data store (ODS). The third step consists in the computation of aggregated views from base views. While the data extracted from the sources and integrated in the ODS is considered as ground data with very low level aggregation, the data in the corporate data warehouse (CDW) is generally highly summarised using aggregation functions. The fourth step consists in the derivation and customisation of the user views which define the data marts. Customisation refers to various presentations needed by the users for multidimensional data.

Figure 9.3: Data Loading Activities



The main feature of the loading phase is that it constitutes the latest stage of the data warehouse design project. Before the end of the data loading, the data warehouse does not yet exist for the users.

Consequently, there is no constraint on the response time. But, in contrast, with respect to the data sources, the loading phase requires more availability.

The data flow which describes the loading phase can serve as a basis to define the refreshment process, but the corresponding workflows are different. The workflow of the refreshment process is dynamic and can evolve with users' needs and with source evolution, while the workflow of the initial loading process is static and defined with respect to current user requirements and current sources.

The difference between the refreshment process and the loading process is mainly in the following. First, the refreshment process may have a complete asynchronism between its different activities (preparation, integration, aggregation and customisation). Second, there may be a high level parallelism within the preparation activity itself, each data source having its own availability window and its own strategy of extraction. The synchronization is done by the integration activity. Another difference lies in the source availability. While the loading phase requires a long period of availability, the refreshment phase should not overload the operational applications which use the data sources. Then, each source provides a specific access frequency and a restricted availability duration. Finally, there are more constraints on response time for the refreshment process than for the loading process. Indeed, with respect to the users, the data warehouse does not exist before the initial loading, so the computation time is included within the design project duration. After the initial loading, the data becomes visible and should satisfy user requirements in terms of data availability, accessibility and freshness.

View Maintenance vs. Data Refreshment

The propagation of changes during the refreshment process is done through a set of independent activities among which we find the maintenance of the views stored in the ODS and CDW levels. The view maintenance phase consists in propagating a certain change raised in a given source over a set of views stored at the ODS or CDW level. Such a phase is a classical materialized view maintenance problem except that, in data warehouses, the changes to propagate into the aggregated views are not exactly those occurred in the sources, but the result of pre-treatments performed by other refreshment activities such as data cleaning and multi-source data reconciliation.

The view maintenance problem has been intensively studied in the database research community. Most of the references focus on the problems raised by the maintenance of a set of materialized (also called concrete) views derived from a set of base relations when the current state of the base relations is modified. The main results concern:

1. ***Self-maintainability:*** Results concerning the self-maintainability are generalized for a set of views : a set of view V is self-maintainable with respect to the changes to the underlying base relations if the changes may be propagated in every views in V without querying the base relations (i.e. the information stored in the concrete views plus the instance of the changes are sufficient to maintain the views).
2. ***Coherent and efficient update propagation:*** Various algorithms are provided to schedule updates propagation through each individual view, taking care of interdependencies between views, which may lead to possible inconsistencies. For this purpose, auxiliary views are often introduced to facilitate update propagation and to enforce self-maintainability.

Results over the self-maintainability of a set of views are of a great interest in the data warehouse context, and it is commonly admitted that the set of views stored in a data warehouse have to be globally selfmaintainable. The rationale behind this recommendation is that the self-maintainability is a strong requirement imposed by the operational sources in order to not overload their regular activity.

Research on data warehouse refreshment has mainly focused on update propagation through materialized views. Many papers have been published on this topic, but a very few is devoted to the whole refreshment process as defined before. We consider view maintenance just as one step of the complete refreshment process. Other steps concern data cleaning, data reconciliation, data customisation, and if needed data archiving. In another hand, extraction and cleaning strategies may vary from one source to another, as well as update propagation which may vary from one user view to another, depending for example on the desired freshness for data. So the data warehouse refreshment process cannot be limited to a view maintenance process.

To summarize the previous discussion, we can say that a refreshment process is a complex system which may be composed of asynchronous and parallel activities that need a certain monitoring. The refreshment process is an event-driven system which evolves frequently, following the evolution of data sources and user requirements. Users, data warehouse administrators and data source administrators may impose specific constraints as, respectively, freshness of data, space limitation of the ODS or CDW, and access frequency to sources. There is no simple and unique refreshment strategy which is suitable for all data warehouse applications, for all data warehouse user, or for the whole data warehouse lifetime.

9.3.2 The Refreshment Process is a Workflow

A workflow is a set of coordinated activities which might be manual or automated activities performed by actors. Workflow concepts have been used in various application domains such as business process modeling, cooperative applications modeling and database transaction modeling. Depending on the application domain, activities and coordination are defined using appropriate specification languages such as statechart diagrams and Petri nets, or active rules. In spite of this diversity of applications and representation, most of the workflow users refer more or less to the concepts and terminology defined by the Workflow Coalition. Workflow systems are supposed to provide high level flexibility to recursively decompose and merge activities, and allow dynamic reorganization of the workflow process. These features are typically useful in the context of data warehouse refreshment as the activities are performed by market products whose functionalities and scope differ from one product to another.

Task Suppose that the data for analysis include the attribute age. The age values for the data tuples are (in creasing order):

13, 15, 16, 19, 20, 21, 22, 22, 25, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 58

1. Use smoothing by bin means to smooth the above data, using a bin depth of 3. Illustrate your steps. Comment on the effect of this technique for the given data.
2. How might you determine outliers in the data?
3. Use Min-max transformation to transform the value 35 for age onto the range [0.0,1.0]

9.4 Implementation of the Approach

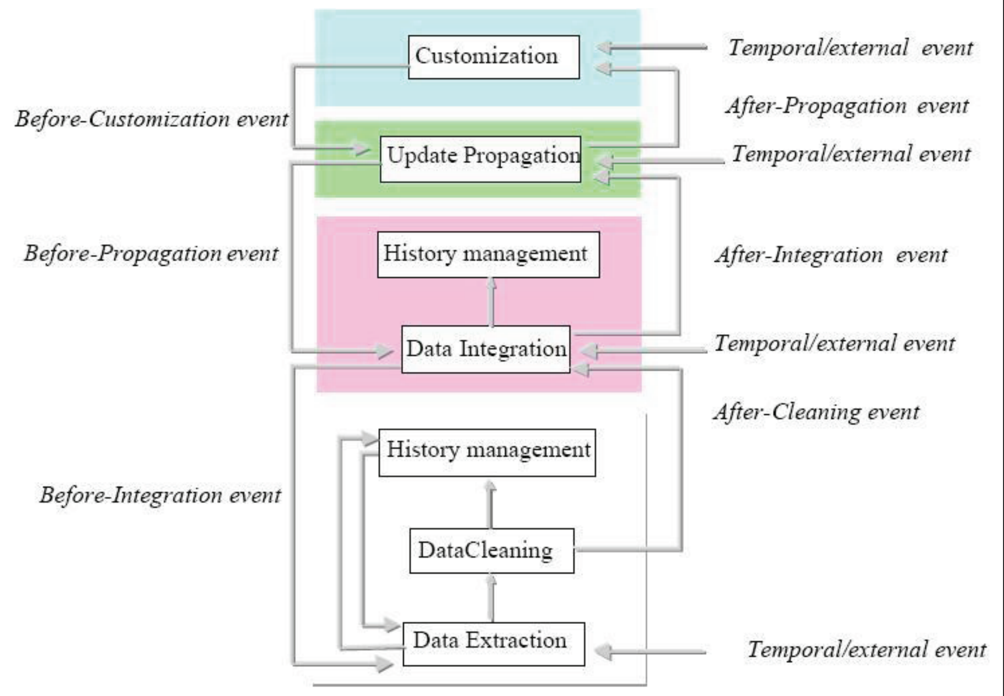
In this section, you show how the refreshment process can be defined as a workflow application. We illustrate the interest of this approach buy the ability to define different scenarios depending on user requirements, source constraints and data warehouse constraints. I show that these scenarios may evolve through the time to fulfill evolution of any of the previous requirements and constraints.

9.4.1 The Workflow of the Refreshment Process

The refreshment process aims to propagate changes raised in the data sources to the data warehouse stores. This propagation is done through a set of independent activities (extraction, cleaning, integration, ...) that can be organized in different ways, depending on the semantics one wants to assign to the refreshment process and on the quality he wants to achieve. The ordering of these activities and the context in which they are executed define this semantics and influence this quality. Ordering and context result from the analysis of view definitions, data source constraints and user requirement in terms of quality factors. In the following subsections, we will describe the refreshment activities and their organization as a workflow. Then we give examples of different workflow scenarios to show how refreshment may be a dynamic and evolving process. Finally, we summarize the different perspectives through which a given refreshment scenario should be considered.

The refreshment process is similar to the loading process in its data flow but, while the loading process is a massive feeding of the data warehouse, the refreshment process captures the differential changes hold in the sources and propagates them through the hierarchy of data stores in the data warehouse. The preparation step extracts from each source the data that characterises the changes that have occurred in this source since the last extraction. As for loading, this data is cleaned and possibly archived before its integration. The integration step reconciliates the source changes coming from multiple sources and adds them to the ODS. The aggregation step recomputes incrementally the hierarchy of aggregated views using these changes. The customisation step propagates the summarized data to the data marts. As well as for the loading phase, this is a logical decomposition whose operational implementation receives many different answers in the data warehouse products. This logical view allows a certain traceability of the refreshment process. Figure 9.4 shows the activities of the refreshment process as well as a sample of the coordinating events.

Figure 9.4: The Generic Workflow for the Refreshment Process



In workflow systems, activities are coordinated by control flows which may be notification of process commitment, emails issued by agents, temporal events, or any other trigger events. In the refreshment process, coordination is done through a wide range of event types.

You can distinguish several event types which may trigger and synchronize the refreshment activities. They might be temporal events, termination events or any other user-defined event. Depending on the refreshment scenario, one can choose an appropriate set of event types which allows to achieve the correct level of synchronization.

Activities of the refreshment workflow are not executed as soon as they are triggered, they may depend on the current state of the input data stores. For example, if the extraction is triggered periodically, it is actually executed only when there are effective changes in the source log file. If the cleaning process is triggered immediately after the extraction process, it is actually executed only if the extraction process has gathered some source changes. Consequently, we can consider that the state of the input data store of each activity may be considered as a condition to effectively execute this activity.

Within the workflow which represents the refreshment process, activities may be of different origins and different semantics, the refreshment strategy is logically considered as independent of what the activities actually do. However, at the operational level, some activities can be merged (e.g., extraction and cleaning), and some others decomposed (e.g. integration). The flexibility claimed for workflow systems should allow to dynamically tailor the refreshment activities and the coordinating events.

There may be another way to represent the workflow and its triggering strategies. Indeed, instead of considering external events such as temporal events or termination events of the different activities, we can consider data changes as events. Hence, each input data store of the refreshment workflow is considered as an event queue that triggers the corresponding activity. However, to be able to represent different refreshment strategies, this approach needs a parametric synchronization mechanism which allows to trigger the activities at the right moment. This can be done by introducing composite events which combine, for example, data change events and temporal events. Another alternative is to put locks on data stores and remove them after an activity or a set of activities decide to commit. In the case of a long term synchronization policy, as it may sometimes happen in some data warehouses, this latter approach is not sufficient.

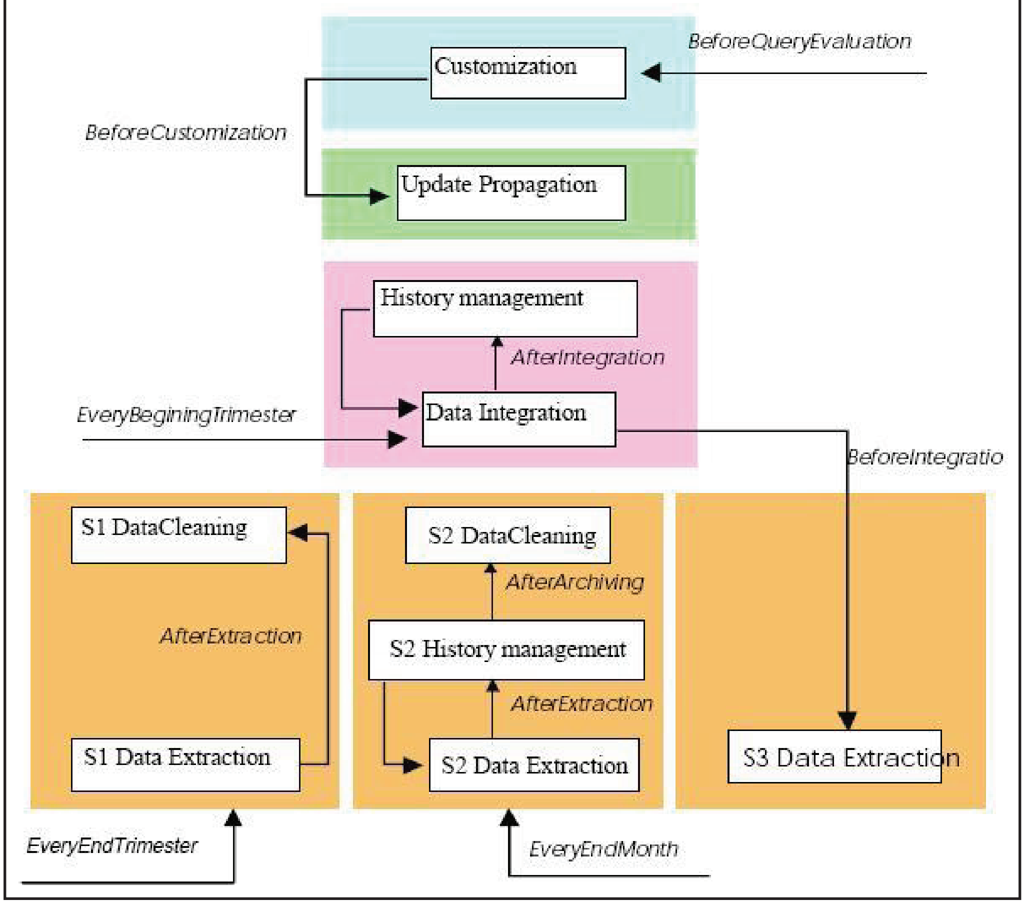
The Workflow Agents

Two main agent types are involved in the refreshment workflow: human agents which define requirements, constraints and strategies, and computer agents which process activities. Among human agents we can distinguish users, the data warehouse administrator, source administrators. Among computer agents, we can mention source management systems, database systems used for the data warehouse and data marts, wrappers and mediators. For simplicity, agents are not represented in the refreshment workflow which concentrates on the activities and their coordination.

9.4.2 Defining Refreshment Scenarios

To illustrate different workflow scenarios, we consider the following example which concern three national Telecom billing sources represented by three relations S1, S2, and S3. Each relation has the same (simplified) schema: (#PC, date, duration, cost). An aggregated view V with schema (avgduration, avg-cost, country) is defined in a data warehouse from these sources as the average duration and cost of a phone call in each of the three country associated with the sources, during the last 6 months. We assume that the construction of the view follows the steps as explained before. During the preparation step, the data of the last six months contained in each source is cleaned (e.g., all cost units are translated in Euros). Then, during the integration phase, a base relation R with schema (date, duration, cost country) is constructed by unioning the data coming from each source and generating an extra attribute (country). Finally, the view is computed using aggregates (Figure 9.5).

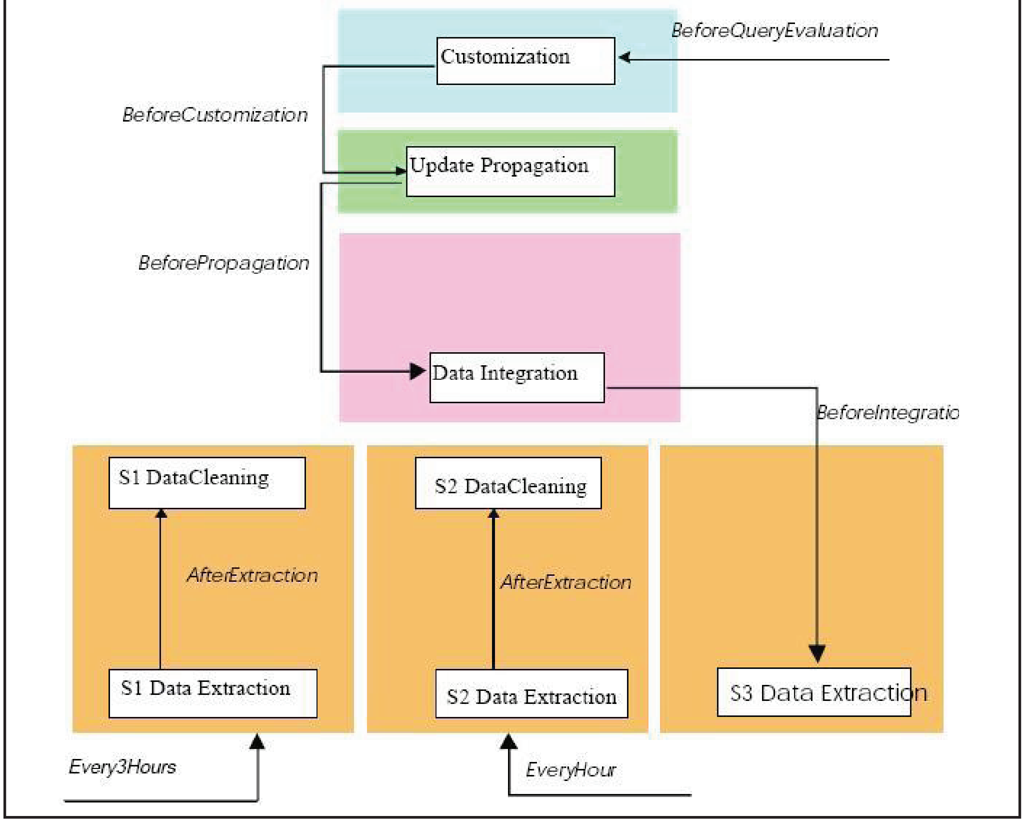
Figure 9.5: First Example of Refreshment Scenario



We can define another refreshment scenario with the same sources and similar views. This scenario mirrors the average duration and cost for each day instead of for the last six months. This leads to change the frequency of extraction, cleaning, integration and propagation. Figure 9.6 gives such a possible scenario. Frequencies of source extractions are those which are allowed by source administrators.

Notes

Figure 9.6: Second Example of Refreshment Scenario



When the refreshment activities are long term activities or when the DWA wants to apply validation procedures between activities, temporal events or activity terminations can be used to synchronize all the refreshment process. In general, the quality requirements may impose a certain synchronization strategy. For example, if users desire high freshness for data, this means that each update in a source should be mirrored as soon as possible to the views. Consequently, this determines the strategy of synchronization: trigger the extraction after each change in a source, trigger the integration, when semantically relevant, after the commit of each data source, propagate changes through views immediately after integration, and customize the user views in data marts.

Refreshment Scheduling

The refreshment process can be viewed through different perspectives:

1. ***Client-driven refreshment*** which describes part of the process which is triggered on demand by the users. This part mainly concern update propagation from the ODS to the aggregated views. The on-demand strategy can be defined for all aggregated views or only for those for which the freshness of data is related to the date of querying.

2. ***Source-driven refreshment*** which defines part of the process which is triggered by changes made in the sources. This part concerns the preparation phase. The independence between sources can be used as a way to define different preparation strategies, depending on the sources. Some sources may be associated with cleaning procedures, others not. Some sources need a history of the extracted data, others not. For some sources, the cleaning is done on the fly during the extraction, for some others after the extraction or on the history

of these changes. The triggering of the extraction may be also different from one source to another. Different events can be defined, such as temporal events (periodic or fixed absolute time), after each change detected on the source, on demand from the integration process.

3. ***ODS-driven refreshment*** which defines part of the process which is automatically monitored by the data warehouse system. This part concerns the integration phase. It may be triggered at a synchronization point defined with respect to the ending of the preparation phase. Integration can be considered as a whole and concerns all the source changes at the same time. In this case, it can be triggered by an external event which might be a temporal event or the ending of the preparation phase of the last source. The integration can also be sequenced with respect to the termination of the preparation phase of each source, that is extraction is integrated as soon as its cleaning is finished. The ODS can also monitor the preparation phase and the aggregation phase by generation the relevant events that triggers activities of these phases.

In the very simple case, one of the two first approaches is used as a single strategy. In a more complex case, there may be as much strategies as the number of sources or high level aggregated views. In between, there may be, for example, four different strategies corresponding to the previous four phases. For some given user views, one can apply the client driven strategy (pull strategy), while for other views one can apply the ODS-driven strategy (push strategy). Similarly, some sources are solicited through a pull strategy while other apply a push strategy.

The strategy to choose depends on the semantic parameters but also on the tools available to perform the refreshment activities (extraction, cleaning, integration). Some extraction tools do also the cleaning in the fly while some integrators propagate immediately changes until the high level views. Then, the generic workflow in Figure 9.4 is a logical view of the refreshment process. It shows the main identified activities and the potential event types which can trigger them.

9.5 Implementation Issues

With respect to the implementation issues, different solutions can be considered. The conceptual definition of the refreshment process by means of a workflow, leads naturally to envision an implementation under the control of a common workflow system in the market, provided that this latter one supplies event types and all features needed by the refreshment scenario. Another solution we have preferred and consists in using active rules which should be executed under a certain operational semantics. The rationale behind our choice is the flexibility and the evolutivity provided by active rules. Indeed the refreshment strategy is not defined once for all; it may evolve with the user needs, which may result in the change of the definition of materialized views or the change of desired quality factors. It may also evolve when the actual values of the quality factors slow down with the evolution of the data warehouse feeding or with the technology used to implement it. Consequently, in order to master the complexity and the evolutivity of the data warehouse, it is important to provide a flexible technology which allows to accommodate this complexity and evolutivity. This is what active rules meant to provide. A prototype has been developed and demonstrated in the context of the DWQ european research project on Data warehouses. However, active rules cannot be considered as an alternative to workflow representation. Workflow is a conceptual view of the refreshment process, while actives rules are operational implementation of the workflow.

Notes

Case Study

BT Groups

A global communications company runs the Trillium Software System® at the heart of its customer-focused operations.

The Company

BT Group is one of Europe's leading providers of telecommunications services. Its principal activities include local, national, and international telecommunications services, higher-value broadband and internet products and services, and IT solutions.

In the UK, BT serves over 20 million business and residential customers with more than 29 million exchange lines, and provides network services to other licensed operators.

Reducing customer dissatisfaction by 25% a year is a key target in BT Group's drive to deliver the highest levels of customer satisfaction. In the 2003 financial year, the Group as a whole achieved a 37% reduction in customer dissatisfaction.

The Challenge

BT's well-established strategy puts customers first and includes ensuring that customers are recognized and properly managed by appropriate account teams who can access all relevant information rapidly. This is made possible behind the scenes by a well-developed and strategic approach to customer information management and data quality.

"BT recognizes that there is a link between the quality and completeness of customer information and good account management, customer service, and effective operations. Customer data is a key business asset. We must manage it with strategic intent," said Nigel Turner, Manager of Information & Knowledge Management Consultancy, BT Exact (BT's research, technology, and IT operations business), which is assisting BT in its information management strategy.

Three initiatives take center stage in BT's customer information strategy: the Name and Address System (NAD), a Customer Relationship Management (CRM) system from Siebel, and the Customer Service System (CSS).

NAD is being built from multiple legacy data sources to provide what is to become a single definitive customer name and address repository. The multimillion-dollar Siebel-based CRM solution commenced rollout in what is destined to become one of the largest Siebel implementations anywhere in the world. CSS provides a complete and accurate view of each customer from 28 million customer records across 29 disparate repositories and 11 mainframes, where many customers have multiple records. The system, now central to customer services, determines the availability of services, the location of a prospect, account management, billing, and fault repair.

A significant challenge in each of these enormous and mission-critical business investments is ensuring the quality of source data in order to deliver output that is accurate, complete, and valuable enough to deliver a strong return on investment (ROI). Garbage in, garbage out applies! BT has needed a process capable of taking tens of millions of customer records from multiple disparate sources, then cleansing and standardizing address formats, then accurately identifying, grouping and linking records that are common to one customer account, household or business. Furthermore, the strategy has had to be capable of building and maintaining customer data quality over time, encompassing both existing data offline and new data entered by call center operatives and others in real time.

Contd...

The Solution

BT first selected the Trillium Software System® to aid data quality and record linking in a central marketing database sourced from millions of customer records. Since then, BT's experiences with the solution have seen the Trillium Software System become the corporate standard for BT enterprise customer data quality.

"We have expanded and extended our use of the Trillium Software System to the extent that it has become our defacto standard for name and address processing across BT," said Turner. "The Trillium Software System is serving BT across tens of millions of customer records, benefiting hundreds of processes, and is central to the effectiveness of BT's customerfocused operations and competitive position."

The Trillium Software System has been built into CSS, NAD, and Siebel and their many source systems (as well as other processes), ensuring the quality of customer names and addresses, enabling source data formats to be standardized, duplicates to be recognized and handled, and fragmented customer records to be intelligently molded into linked information. Matches with external data sources from The Royal Mail, D&B, and others (for name and address and for lifestyle data, for example) can also be made reliably too, in order to supplement BT's own information.

"The Trillium Software System is now much more than an application at BT. It's a key part of the infrastructure," said Turner.

The Results

BT is finding that strong customer data quality is enabling it to get the most from CSS, NAD, Siebel, and other initiatives.

9.6 Summary

- This unit has presented an analysis of the refreshment process in data warehouse applications.
- You have demonstrated, that the refreshment process cannot be limited neither to a view maintenance process nor to a loading process.
- You have shown through a simple example, that the refreshment of a data warehouse can be conceptually viewed as a workflow process.
- You have identified the different tasks of the workflow and shown how they can be organized in different refreshment scenarios, leading to different refreshment semantics.
- You have highlighted design decisions impacting over the refreshment semantics and we have shown how the decisions may be related to some quality factors such as data freshness and to some constraints such as source availability and accessibility.

9.7 Keywords

Data Refreshment: Data refreshment in data warehouses is generally confused with data loading as done during the initial phase or with update propagation through a set of materialized views.

Data Warehousing: Data warehousing is a new technology which provides software infrastructure for decision support systems and OLAP applications.

Materialized View: A materialized view eliminates the overhead associated with expensive joins and aggregations for a large or important class of queries.

Nested Materialized View: A nested materialized view is a materialized view whose definition is based on another materialized view.

9.8 Self Assessment

Choose the appropriate answers:

1. OLTP stands for:
   (a) Offline Transaction Processing
   (b) Online Transaction Processing
   (c) Only Transaction Processing
   (d) Online Transaction Program
2. ODS stands for:
   (a) Operational Data Store
   (b) Organisational Data Store
   (c) Operational Data Source
   (d) Operational Design Store
3. CDW stands for:
   (a) Corporate Design Warehouse
   (b) Computing Data Warehouse
   (c) Common Data Warehouse
   (d) Corporate Data Warehouse

Fill in the blanks:

4. Research on data warehouse refreshment has mainly focused on update propagation through ....
5. A .... is a set of coordinated activities which might be manual or automated activities performed by actors.
6. The .... aims to propagate changes raised in the data sources to the data warehouse stores.
7. The .... reconciliates the source changes coming from multiple sources and adds them to the ODS.
8. Materialized views can perform a number of roles, such as ....
9. The .... creates one or more materialized views, which are the equivalent of a summary.
10. Materialized views are useful in .... data marts.

9.9 Review Questions

1. "A workflow is a set of coordinated activities which might be manual or automated activities performed by actors". Explain
2. "The refreshment process is similar to the loading process in its data flow but, while the loading process is a massive feeding of the data warehouse". Explain
3. Define refreshment scenarios.
4. What do you mean by nested materialized views? Explain with suitable example.

Notes

5. Describe corporate data warehouse.
6. Distinguish between view maintenance vs data refreshment.
7. "Some extraction tools do also the cleaning in the fly while some integrators propagate immediately changes until the high level views." Explain
8. If any user desire high freshness for data, this means that each update in a source should be mirrored as soon as possible to the views. Discuss
9. "Depending on the refreshment scenario, one can choose an appropriate set of event types which allows to achieve the correct level of synchronization." Explain
10. What are the basic uses of data cleaning and data extraction? Explain

Answers: Self Assessment

1.	(b)	2.	(a)
3.	(d)	4.	materialized views
5.	workflow	6.	refreshment process
7.	integration step	8.	improving query performance
9.	database administrator	10.	remote

9.10 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 10: Multi-dimensional Data Models and Aggregation

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain multi-dimensional view of information
- Describe ROLAP data model
- Describe MOLAP data model
- Describe logical and conceptual model for multi-dimensional information

Introduction

Metadata is used throughout Oracle OLAP to define a logical multidimensional model:

1. To describe the source data as multidimensional objects for use by the analytic workspace build tools.
2. To identify the components of logical objects in an analytic workspace for use by the refresh, aggregation, and enablement tools.
3. To describe relational views of analytic workspaces as multidimensional objects for use by OLAP applications.

10.1 Multi-dimensional View of Information

The multi-dimensional data model and view of information depend upon various types of multidimensional data models information and how it is implemented in relational tables and standard form analytic workspaces. It consists of the following topics:

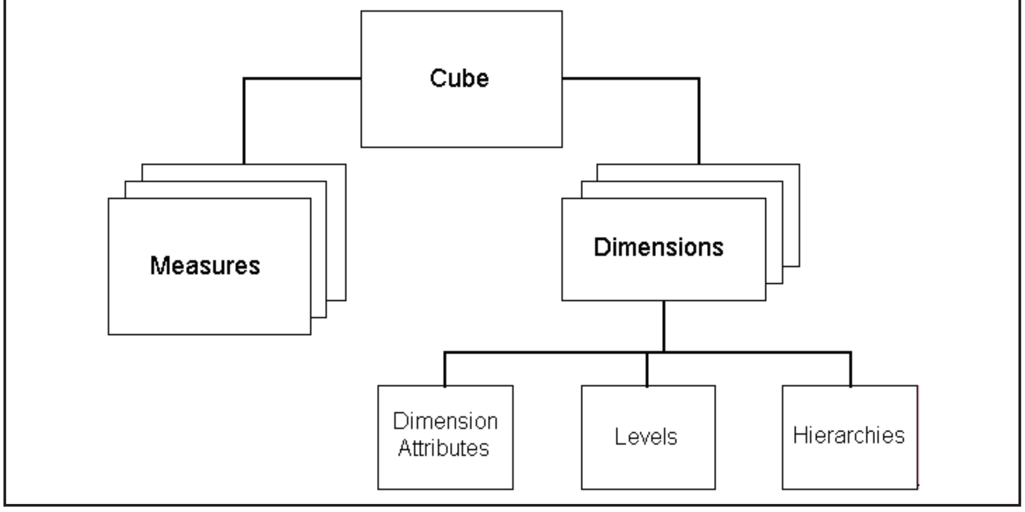
1. The Logical Multi-dimensional Data Model
2. The Relational Implementation of the Model
3. Conceptual Model for Multi-dimensional Information

10.2 The Logical Model for Multi-dimensional Information

Data warehouses and OLAP tools are based on a multi-dimensional data model. It is designed to solve complex queries in real time. The central attraction of the dimensional model of a business is its simplicity which is the fundamental key that allows users to understand databases, and allows software to navigate databases efficiently.

The multi-dimensional data model is composed of logical cubes, measures, dimensions, hierarchies, levels, and attributes. Figure 10.1 shows the relationships among the logical objects.

Figure 10.1: Logical Multi-dimensional Model



Logical Cubes: Logical cubes provide a means of organising measures that have the same shape, that is, they have the exact same dimensions. Measures in the same cube have the same relationships to other logical objects and can easily be analysed and displayed together.

Logical Measures: Measures populate the cells of a logical cube with the facts collected about business operations. Measures are organised by dimensions, which typically include a Time dimension.

Measures are static and consistent while analysts are using them to inform their decisions. They are updated in a batch window at regular intervals: weekly, daily, or periodically throughout the day. Many applications refresh their data by adding periods to the time dimension of a measure, and may also roll off an equal number of the oldest time periods. Each update provides a fixed historical record of a particular business activity for that interval. Other applications do a full rebuild of their data rather than performing incremental updates.

Logical Dimensions: Dimensions contain a set of unique values that identify and categorise data. They form the edges of a logical cube, and thus of the measures within the cube. Because measures are typically multidimensional, a single value in a measure must be qualified by a member of each dimension to be meaningful. For example, the Sales measure has four dimensions: Time, Customer, Product, and Channel. A particular Sales value (43,613.50) only has meaning when it is qualified by a specific time period (Feb-01), a customer (Warren Systems), a product (Portable PCs), and a channel (Catalog).

Logical Hierarchies and Levels: A hierarchy is a way to organise data at different levels of aggregation. In viewing data, analysts use dimension hierarchies to recognise trends at one level, drill down to lower levels to identify reasons for these trends, and roll up to higher levels to see what affect these trends have on a larger sector of the business.

Each level represents a position in the hierarchy. Each level above the base (or most detailed) level contains aggregate values for the levels below it. The members at different levels have a one-to-many parent-child relation. For example, Q1-2002 and Q2-2002 are the children of 2002, thus 2002 is the parent of Q1-2002 and Q2-2002.

Suppose a data warehouse contains snapshots of data taken three times a day, that is, every 8 hours. Analysts might normally prefer to view the data that has been aggregated into days, weeks, quarters, or years. Thus, the Time dimension needs a hierarchy with at least five levels.

Similarly, a sales manager with a particular target for the upcoming year might want to allocate that target amount among the sales representatives in his territory; the allocation requires a dimension hierarchy in which individual sales representatives are the child values of a particular territory.

Hierarchies and levels have a many-to-many relationship. A hierarchy typically contains several levels, and a single level can be included in more than one hierarchy

Logical Attributes: An attribute provides additional information about the data. Some attributes are used for display. For example, you might have a product dimension that uses Stock Keeping Units (SKUs) for dimension members. The SKUs are an excellent way of uniquely identifying thousands of products, but are meaningless to most people if they are used to label the data in a report or graph. You would define attributes for the descriptive labels.

Multi-dimensional Data Storage in Analytic Workspaces

In the logical multidimensional model, a cube represents all measures with the same shape, that is, the exact same dimensions. In a cube shape, each edge represents a dimension. The dimension members are aligned on the edges and divide the cube shape into cells in which data values are stored.

In an analytic workspace, the cube shape also represents the physical storage of multidimensional measures, in contrast with two-dimensional relational tables. An advantage of the cube shape is that it can be rotated: there is no one right way to manipulate or view the data. This is an important part of multidimensional data storage, calculation, and display, because different analysts need to view the data in different ways. For example, if you are the Sales Manager, then you need to look at the data differently from a product manager or a financial analyst.

Assume that a company collects data on sales. The company maintains records that quantify how many of each product was sold in a particular sales region during a specific time period. You can visualise the sales measure as the cube shown in Figure 10.2.

Notes

Figure 10.2: Comparison of Product Sales by City

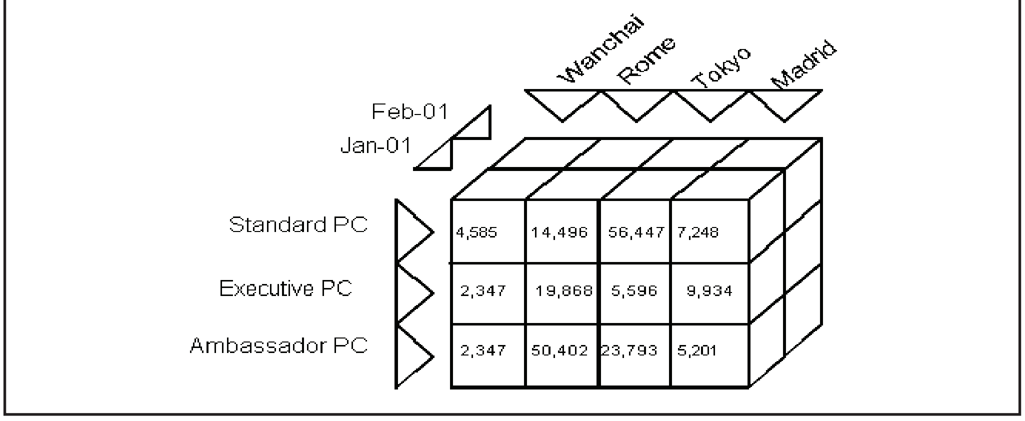
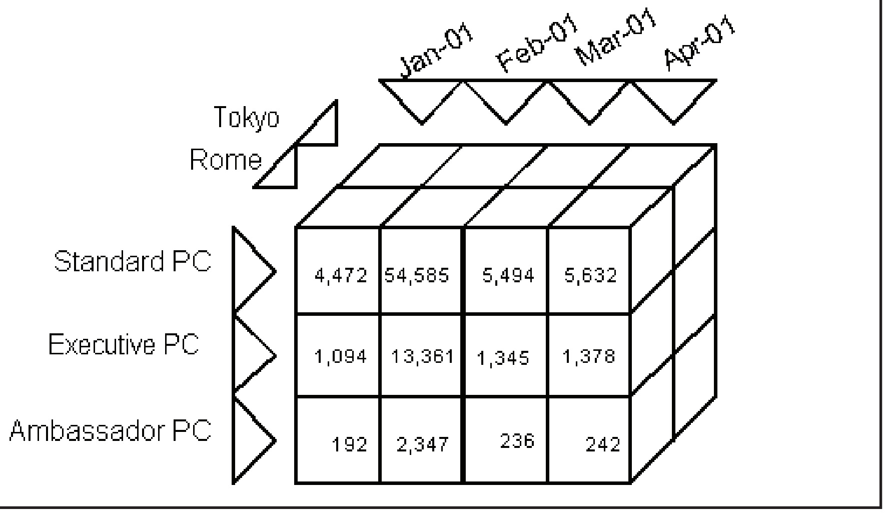


Figure 10.2 compares the sales of various products in different cities for January 2001 (shown) and February 2001 (not shown). This view of the data might be used to identify products that are performing poorly in certain markets. Figure 10.3 shows sales of various products during a four-month period in Rome (shown) and Tokyo (not shown). This view of the data is the basis for trend analysis.

Figure 10.3: Comparison of Product Sales by Month



A cube shape is three dimensional. Of course, measures can have many more than three dimensions, but three dimensions are the maximum number that can be represented pictorially. Additional dimensions are pictured with additional cube shapes.

10.3 Relational Implementation of the Model

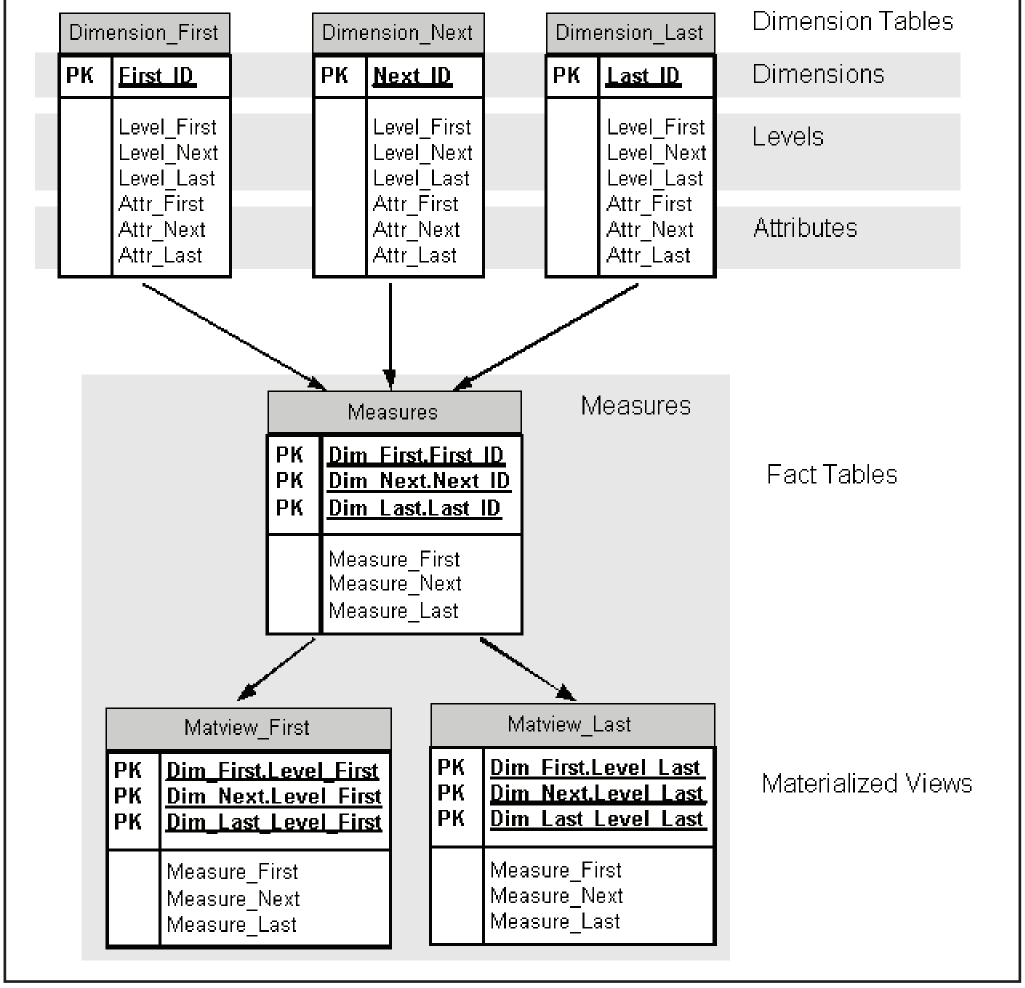
The relational implementation of the multi-dimensional data model is typically a star schema, as shown in Figure 10.4, a snowflake schema or a fact constellation schema.

10.3.1 Star Schema

A star schema is a convention for organising the data into dimension tables, fact tables, and materialised views. Ultimately, all of the data is stored in columns, and metadata is required to identify the columns that function as multidimensional objects.

Notes

Figure 10.4: Star Schema



Dimension Tables

A star schema stores all of the information about a dimension in a single table. Each level of a hierarchy is represented by a column or column set in the dimension table. A dimension object can be used to define the hierarchical relationship between two columns (or column sets) that represent two levels of a hierarchy; without a dimension object, the hierarchical relationships are defined only in metadata. Attributes are stored in columns of the dimension tables.

A snowflake schema normalises the dimension members by storing each level in a separate table.

Fact Tables

Measures are stored in fact tables. Fact tables contain a composite primary key, which is composed of several foreign keys (one for each dimension table) and a column for each measure that uses these dimensions.

Notes

Materialised Views

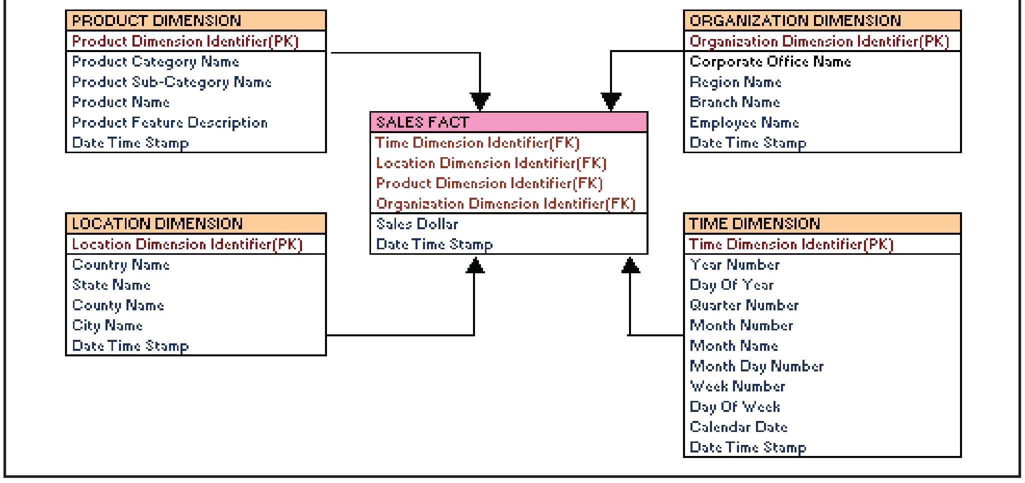
Aggregate data is calculated on the basis of the hierarchical relationships defined in the dimension tables. These aggregates are stored in separate tables, called summary tables or materialised views. Oracle provides extensive support for materialised views, including automatic refresh and query rewrite.

Queries can be written either against a fact table or against a materialised view. If a query is written against the fact table that requires aggregate data for its result set, the query is either redirected by query rewrite to an existing materialised view, or the data is aggregated on the fly.

Each materialised view is specific to a particular combination of levels; in Figure 10.5, only two materialised views are shown of a possible 27 (3 dimensions with 3 levels have 3**3 possible level combinations).

Example: Let, an organisation sells products throughtout the world. The main four major dimensions are product, location, time and organisation.

Figure 10.5: Example of Star Schema



In the example Figure 10.5, sales fact table is connected to dimensions location, product, time and organisation. It shows that data can be sliced across all dimensions and again it is possible for the data to be aggregated across multiple dimensions. "Sales Dollar" in sales fact table can be calculated across all dimensions independently or in a combined manner, which is explained below:

1. Sales Dollar value for a particular product
2. Sales Dollar value for a product in a location
3. Sales Dollar value for a product in a year within a location
4. Sales Dollar value for a product in a year within a location sold or serviced by an employee

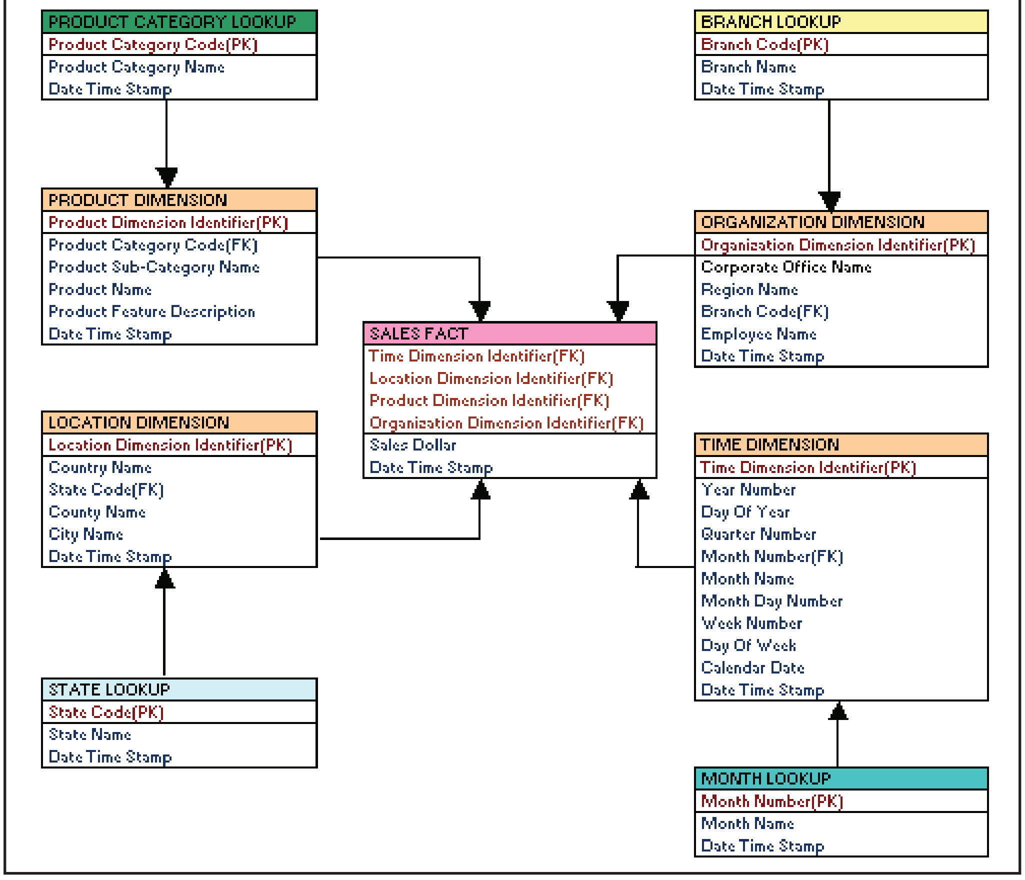
10.3.2 Snowflake Schema

The snowflake schema is a variant of the star schema model, where some dimension tables are normalised, thereby further splitting the data into additional tables. The resulting schema graph forms a shape similar to a snowflake.

The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalised form. Such a table is easy to maintain and also saves storage space because a large dimension table can be extremely large when the dimensional structure is included as columns. Since much of this space is redundant data, creating a normalised structure will reduce the overall space requirement. However, the snowflake structure can reduce the effectiveness of browsing since more joins will be needed to execute a query. Consequently, the system performance may be adversely impacted. Performance benchmarking can be used to determine what is best for your design.

Example: In Snowflake schema, the example diagram shown in Figure 10.6 has 4 dimension tables, 4 lookup tables and 1 fact table. The reason is that hierarchies (category, branch, state, and month) are being broken out of the dimension tables (PRODUCT, ORGANISATION, LOCATION, and TIME) respectively and shown separately. In OLAP, this Snowflake schema approach increases the number of joins and poor performance in retrieval of data.

Figure 10.6: Example of Snowflake Schema



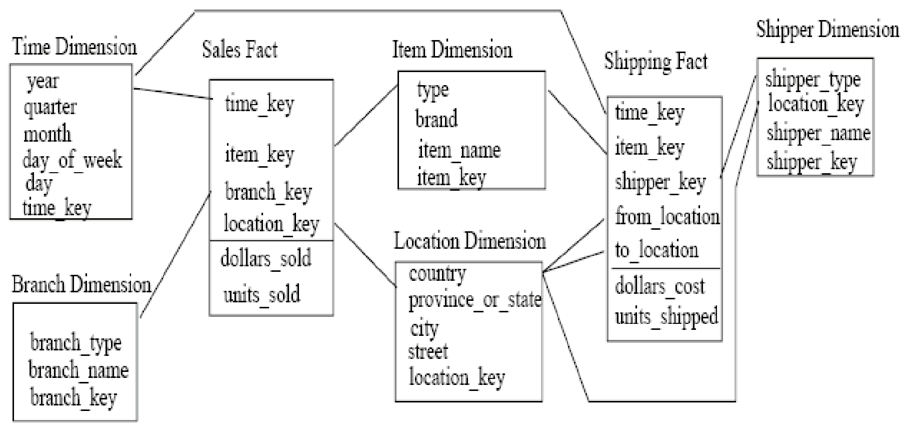
A compromise between the star schema and the snowflake schema is to adopt a mixed schema where only the very large dimension tables are normalised. Normalising large dimension tables saves storage space, while keeping small dimension tables unnormalised may reduce the cost and performance degradation due to joins on multiple dimension tables. Doing both may lead to an overall performance gain. However, careful performance tuning could be required to determine which dimension tables should be normalised and split into multiple tables.

Fact Constellation

Sophisticated applications may require multiple fact tables to share dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a galaxy schema or a fact constellation.

Example: A fact constellation schema of a data warehouse for sales and shipping is shown in the following Figure 10.7.

Figure 10.7: Fact Constellation Schema of a Data Warehouse for Sales and Shipping



Task Discuss the pros and cons of top down and bottom up approaches.

Differences between a Data Warehouse and a Data Mart

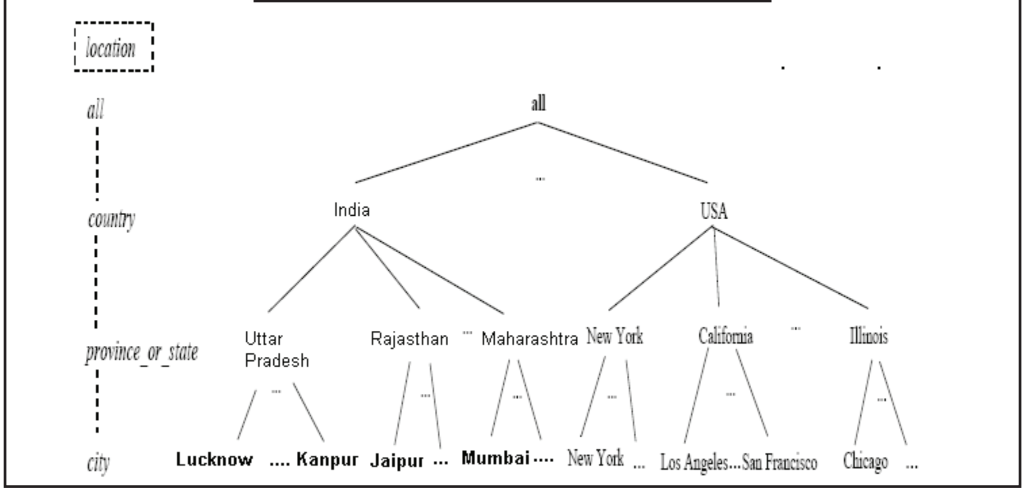
In data warehousing, there is a distinction between a data warehouse and a data mart. A data warehouse collects information about subjects that span the entire organisation, such as customers, items, sales, assets, and personnel, and thus its scope is enterprise-wide. For data warehouses, the fact constellation schema is commonly used since it can model multiple, interrelated subjects. A data mart, on the other hand, is a department subset of the data warehouse that focuses on selected subjects, and thus its scope is department-wide. For data marts, the star or snowflake schema are popular since each are geared towards modeling single subjects

Introducing Concept Hierarchies

A concept hierarchy defines a sequence of mappings from a set of low level concepts to higher level, more general concepts. Consider a concept hierarchy for the dimension location. City values for location include Lucknow, Mumbai, New York, and Chicago. Each city, however, can be mapsped to the province or state to which it belongs. For example, Lucknow can be mapped to Uttar Pradesh, and Chicago to Illinois. The provinces and states can in turn be mapped to the country to which they belong, such as India or the USA. These mappings form a concept hierarchy for the dimension location, mapping a set of low level concepts (i.e., cities) to higher level, more general concepts (i.e., countries). The concept hierarchy described above is illustrated in Figure 10.8.

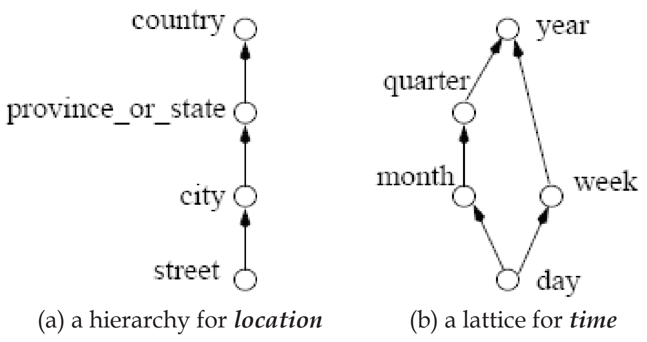
Notes

Figure 10.8: Example of Concept Hierarchy



Many concept hierarchies are implicit within the database schema. For example, suppose that the dimension location is described by the attributes number, street, city, province-or_state, zipcode, and country. These attributes are related by a total order, forming a concept hierarchy such as "street < city < province.or.state < country". This hierarchy is shown in Figure 10.9(a).

Figure 10.9: Hierarchical and Lattice Structures of Attributes in Warehouse Dimensions



Alternatively, the attributes of a dimension may be organised in a partial order, forming a lattice. An example of a partial order for the time dimension based on the attributes day, week, month, quarter, and year is "day < {month <quarter; week} < year". This lattice structure is shown in Figure 10.9(b).

A concept hierarchy that is a total or partial order among attributes in a database schema is called a schema hierarchy. Concept hierarchies that are common to many applications may be predefined in the data mining system, such as the concept hierarchy for time. Data mining systems should provide users with the flexibility to tailor predefined hierarchies according to their particular needs. For example, one may like to define a fiscal year starting on April 1, or an academic year starting on September 1.

OLAP Operations in the Multi-dimensional Data Model

Data Warehouses use On-line Analytical Processing (OLAP) to formulate and execute user queries. OLAP is an SLQ-based methodology that provides aggregate data (measurements) along a set of dimensions, in which each dimension table includes a set of attributes each measure depends on a set of dimensions that provide context for the measure, e.g. for the reseller

Notes

company, the measure is the number of sold units, which are described by the corresponding location, time, and item type all dimensions are assumed to uniquely determine the measure, e.g., for the reseller company, the location, time, producer, and item type provide all necessary information to determine context of a particular number of sold units.

There are five basic OLAP commands that are used to perform data retrieval from a Data warehouse.

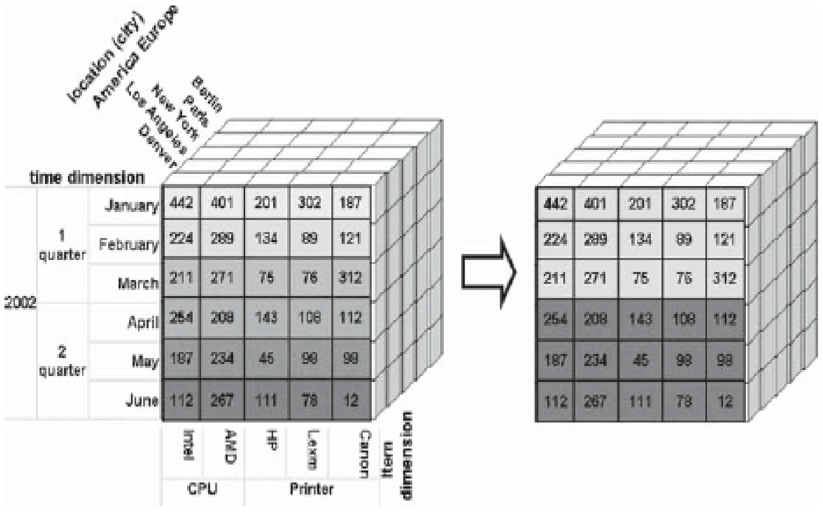
1. ***ROLL UP,*** which is used to navigate to lower levels of details for a given data cube. This command takes the current data cube (object) and performs a GROUP BY on one of the dimensions, e.g., given the total number of sold units by month, it can provide sales summarised by quarter.
2. ***DRILL DOWN,*** which is used to navigate to higher levels of detail. This command is the opposite of ROLL UP, e.g., given the total number of units sold for an entire continent, it can provide sales in the U.S.A.
3. ***SLICE,*** which provides a cut through a given data cube. This command enables users to focus on some specific slice of data inside the cube, e.g., the user may want to look at the data concerning unit sales only in Mumbai.
4. ***DICE,*** which provides just one cell from the cube (the smallest slice), e.g. it can provide data concerning the number of sold Canon printers in May 2002 in Lucknow.
5. ***PIVOT,*** which rotates the cube to change the perspective, e.g., the "time item" perspective may be changed into "time location."

These commands, in terms of their specification and execution, are usually carried out using a point-and-click interface, and therefore we do not describe their syntax. Instead, we give examples for each of the above OLAP commands.

ROLL UP Command

The ROLL UP allows the user to summarise data into a more general level in hierarchy. For instance, if the user currently analyses the number of sold CPU units for each month in the first half of 2002, this command will allows him/her to aggregate this information into the first two quarters. Using ROLL UP, the view

Figure 10.10: Example ROLL UP Command



# sold units		2002					
		January	February	March	April	May	June
CPU	Intel	442	224	211	254	187	112
	AMD	401	289	271	208	234	267

is transformed into

# sold units		2002	
		Quarter 1	Quarter 2
CPU	Intel	877	553
	AMD	961	709

From the perspective of a three-dimensional cuboid, the time _y_ axis is transformed from months to quarters; see the shaded cells in Figure 10.10.

DRILL DOWN Command

The DRILL DOWN command provides a more detailed breakdown of information from lower in the hierarchy. For instance, if the user currently analyses the number of sold CPU and Printer units in Europe and U.S.A., it will allows him/her to find details of sales in specific cities in the U.S.A., i.e., the view.

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	USA	2231	2134	1801	1560	1129
	Europe	1981	2001	1432	1431	1876

is transformed into

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
All	Denver	877	961	410	467	620
	LA	833	574	621	443	213
	NY	521	599	770	650	296

Again, using a data cube representation, the location (z) axis is transformed from summarisation by continents to sales for individual cities; see the shaded cells in Figure 10.11.

SLICE and DICE Command

These commands perform selection and projection of the data cube onto one or more user-specified dimensions.

SLICE allows the user to focus the analysis of the data on a particular perspective from one or more dimensions. For instance, if the user analyses the number of sold CPU and Printer units in all combined locations in the first two quarters of 2002, he/she can ask to see the units in the same time frame in a particular city, say in Los Angeles. The view

Notes

# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
2002	1 quarter	2231	2001	2390	1780	1560
	2 quarter	2321	2341	2403	1851	1621

is transformed into the L.A. table

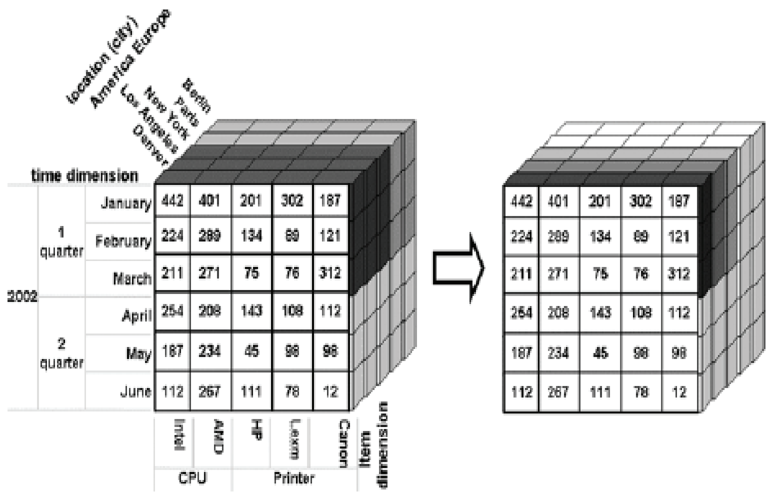
# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
2002	1 quarter	666	601	766	187	730
	2 quarter	1053	759	323	693	501

The DICE command, in contrast to SLICE, requires the user to impose restrictions on all dimensions in a given data cube. An example SLICE command, which provides data about sales only in L.A., and DICE command, which provides data about sales of Canon printers in May 2002 in L.A.

PIVOT Command

PIVOT is used to rotate a given data cube to select a different view. Given that the user currently analyses the sales for particular products in the first quarter of 2002, he/she can shift the focus to see sales in the same quarter, but for different continents instead of for products, i.e., the view.

Figure 10.11: Example of DRILL DOWN Command



# sold units		CPU		Printer		
		Intel	AMD	HP	Lexm	Canon
1 quarter	January	442	401	201	302	187
	February	224	289	134	89	121
	March	211	271	75	76	312

is transformed into

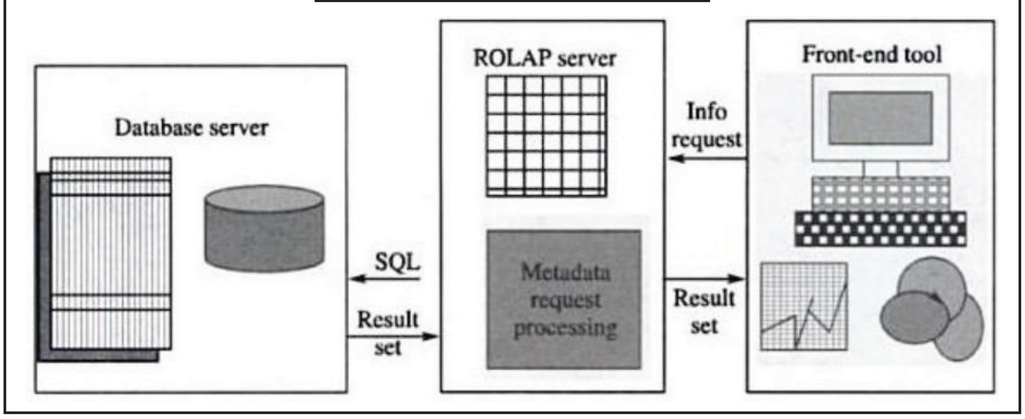
Notes

# sold units		America		Europe		
		Denver	LA	NY	Paris	Berlin
	January	556	321	432	432	341
1 quarter	February	453	564	654	213	231
	March	123	234	345	112	232

10.4 ROLAP Model

Relational OLAP (or ROLAP) is the latest and fastest growing OLAP technology segment in the market. Many vendors have entered the fray in this direction (e.g. Sagent Technology and Micro-strategy). By supporting a dictionary layer of metadata, the RDBMS products have bypassed any requirement for creating a static, multidimensional data structure as shown in Figure 10.12.

Figure 10.12: ROLAP Architecture



This approach enables multiple multidimensional views of two-dimensional relational tables to be created, avoiding structuring data around the desired view. Some products in this segment have supported strong SQL engines to support the complexity of multidimensional analysis. This includes creating multiple SQL statements to handle user requests, being 'RDBMS' aware' and also being capable of generating the SQL statements based in the optimizer of the DBMS engine. While flexibility is the attractive feature of ROLAP, there exist products which require the use of denormalized database design. However, of late there is a noticeable change or realignment in ROLAP technology. Firstly, there is a shift towards pure middle-ware technology so as to simplify the development of multidimensional applications. Secondly, the sharp delineation between ROLAP and other tools and RDBMS products are now eager to provide multidimensional persistent structures with facilities to assist in the administrations of these structures. Notable among vendors of such products are micro-strategy (DSS Agent/DSS server), Platinium/ Perodea Software (Beacon), Information Advantage (AxSys), Informix/Stanford Technology Group (Metacube) and SyBASE (HighGate Project).

Task Discuss OLAP application identify its characteristics.

Notes

10.5 MOLAP Model

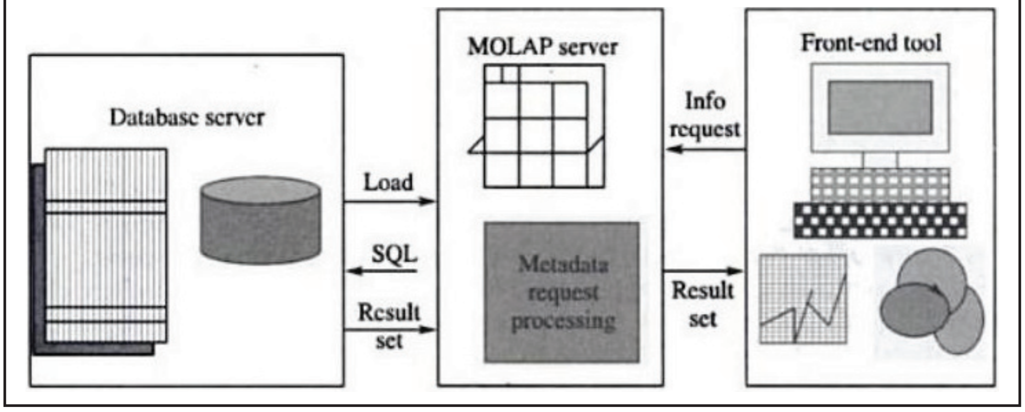
MOLAP-based products organize, navigate and analyze data typically in an aggregated form. They require tight coupling with the applications and they depend upon a multidimensional database (MDDB) system. Efficient implementations store the data in a way similar to the form in which it is utilized by using improved storage techniques so as to minimize storage. Many efficient techniques are used as spare data storage management on disk so as to improve the response time. Some OLAP tools, as Pilot products (Software Analysis Server) introduce 'time' also as an additional dimension for analysis, thereby enabling time 'series' analysis. Some products as Oracle Express Server introduce strong analytical capabilities into the database itself.

Applications requiring iterative and comprehensive time series analysis of trends are well suited for MOLAP technology (e.g. financial analysis and budgeting). Examples include Arbor Software's Essbase. Oracle's Express Server, Pilot Software's Lightship Server, Sinper's TM/1. Planning Science's Gentium and Kenan Technology's Multiway.

Some of the problems faced by users are related to maintaining support to multiple subject areas in an RDBMS. As shown in Figure 10.13, these problems can be solved by the some vendors by maintaining access from MOLAP tools to detailed data in and RDBMS.

This can be very useful for organizations with performance-sensitive multidimensional analysis requirements and that have built or are in the process of building a data warehouse architecture that contains multiple subject areas. An example would be the creation of sales data measured by several dimensions (e.g. product and sales region) to be stored and maintained in a persistent structure. This structure would be provided to reduce the application overhead of performing calculations and building aggregation during initialization. These structures can be automatically refreshed at predetermined intervals established by an administrator.

Figure 10.13: MOLAP Architecture



10.6 Conceptual Models for Multi-dimensional Information

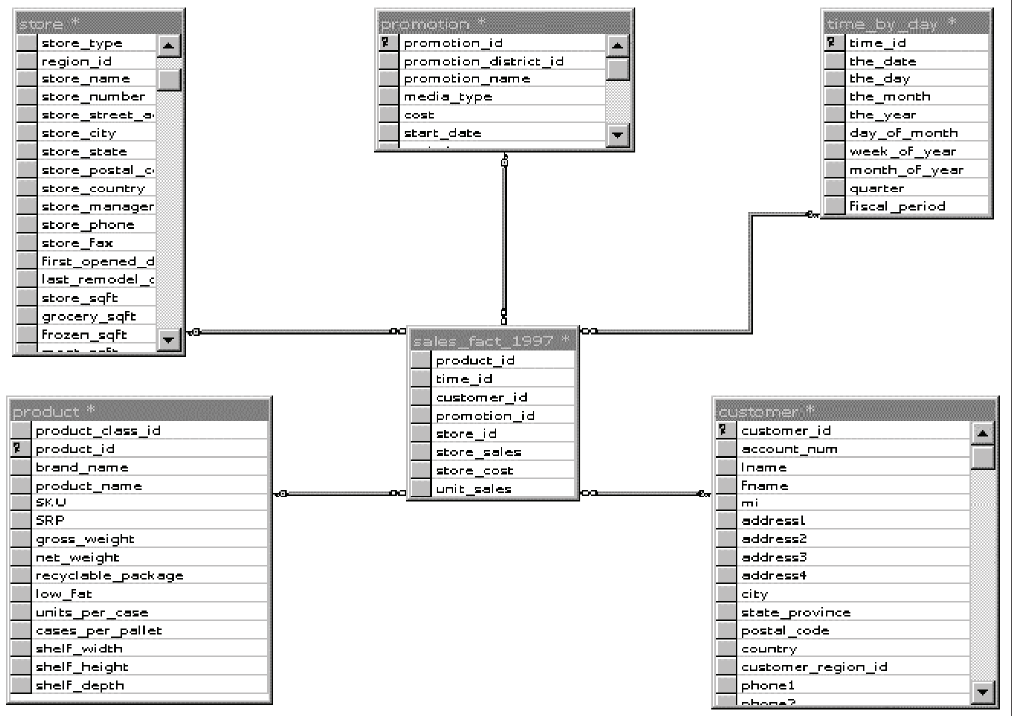
A data model is for a database designer what a box of colors is for a painter: it provides a means for drawing representations of reality. Indeed, it has been claimed that "data modeling is an art", even if the product of this activity has the prosaic name of database scheme.

When a data model allows the designer to devise schemes that are easy to understand and can be used to build a physical database with any actual software system, it is called conceptual. This name comes from the fact that a conceptual model tends to describe concepts of the real world, rather than the modalities for representing them in a computer.

Many conceptual data models exist with different features and expressive powers, mainly depending on the application domain for which they are conceived. As we have said in the Introduction, in the context of data warehousing it was soon realized that traditional conceptual models for database modelling, such as the Entity-Relationship model, do not provide a suitable means to describe the fundamental aspects of such applications. The crucial point is that in designing a data warehouse, there is the need to represent explicitly certain important characteristics of the information contained therein, which are not related to the abstract representation of real world concepts, but rather to the final goal of the data warehouse: supporting data analysis oriented to decision making. More specifically, it is widely recognized that there are at least two specific notions that any conceptual data model for data warehousing should include in some form: the fact (or its usual representation, the data cube) and the dime on. A fact is an entity of an application that is the subject of decision-oriented analysis and is usually represented graphically by means of a data cube. A dimension corresponds to a perspective under which facts can be fruitfully analyzed. Thus, for instance, in a retail business, a fact is a sale and possible dimensions are the location of the sale, the type of product sold, and the time of the sale.

Practitioners usually tend to model these notions using structures that refer to the practical implementation of the application. Indeed, a widespread notation used in this context is the "star schema" (and variants thereof) in which facts and dimensions are simply relational tables connected in a specific way. An example is given in Figure 10.14. Clearly, this low level point of view barely captures the essential aspects of the application. Conversely, in a conceptual model these concepts would be represented in abstract terms which is fundamental for concentration on the basic, multidimensional aspects that can be employed in data analysis, as opposed to getting distracted by the implementation details.

Figure 10.14: An Example of Star Schema



Before tackling in more detail the characteristics of conceptual models for multidimensional applications, it is worth making two general observations. First, we note that in contrast to other application domains, in this context not only at the physical (and logical) but also at the conceptual level, data representation is largely influenced by the way in which final users need to view the information. Second, we recall that conceptual data models are usually used in the preliminary phase of the design process to analyze the application in the best possible way, without implementation "contaminations". There are however further possible uses of multidimensional conceptual representations. First of all, they can be used for documentation purposes, as they are easily understood by non-specialists. They can also be used to describe in abstract terms the content of a data warehousing application already in existence. Finally, a conceptual scheme provides a description of the contents of the data warehouse which, leaving aside the implementation aspects, is useful as a reference for devising complex analytical queries.

Case Study

The Carphone Warehouse

The Carphone Warehouse Calls Trillium Software® in CRM Initiative

The Carphone Warehouse Group plc, known as The Phone House in some countries of operation, was founded in 1989. One of the largest independent retailers of mobile communications in Europe, the group sells mobile phones, phone insurance, network and fixed-line connections through its 1,300 stores, Web site, and direct marketing operations and employs approximately 11,000 people.

A Better Mobile Life

The Carphone Warehouse mission is not just to sell products and services profitably but to offer "A Better Mobile Life" by providing all touch points within the business enough information to give educated advice to customers and deliver customer delight. This customer relationship management (CRM) strategy requires that customer-facing staff know their customers well enough to support a "consultative sell" and that marketing campaigns use segmentation techniques to target only relevant customers.

Single Views

Previously product-centric, The Carphone Warehouse move to a customercentric sales and marketing model presented a challenge. It needed to take fragmented customer information stored across several product-oriented sales and marketing databases and create a new database of "single customer views." These views would then need to be made available to stores, call centers, and marketing in formats suited to their respective process requirements.

A significant component of the Single Customer View project would be the migration of data from three Oracle source databases to a single customer view database (also on Oracle). Informatica's Extract, Transform, and Load (ETL) tool would be used in this process. The Single Customer View database would then support the Customer Dashboard, "One" an at-a-glance resource for call centers and stores and the E.piphany tool used by marketing for campaigns.

The CRM program manager at The Carphone Warehouse knew that before data could be migrated, the source data, fields, format, and content needed to be understood before designing migration mapping rules. He also knew that there could be duplicates and other data quality issues.

"We knew that we had to identify and resolve any significant data quality issues before attempting data migration. We knew that trying to discover and address them manually across some 11 million records would be time-consuming and error-prone. We needed an automated data quality solution," he said.

Contd...

The Carphone Warehouse decided to evaluate data discovery, cleansing, matching, and enhancement solutions. An invitation to tender was issued to six vendors, and quickly narrowed down vendors to two finalists. Each was offered a 2 million-record database of the company's customer records for discovery and cleansing.

Data Quality out of the Box

The Carphone Warehouse chose Trillium Software primarily because in tests it proved to be the most effective in data discovery and cleansing, matching, and enhancement. It supports Informatica and could also easily scale up to their 11 million-record customer database.

The CRM program manager continued, "We especially liked the Trillium Software toolset for out-of-the-box functionality. With tight timescales, we needed to deliver big wins quickly. We were also impressed by sophisticated configuration capabilities that would later enable us to handle the more complex and less obvious data quality issues."

The Carphone Warehouse quickly proceeded to profile source data using the Trillium Software System®. As expected, it discovered a range of issues that needed to be addressed.

Using "One," sales consultants in stores and call centers are generating additional revenue from closing more sales and crosssell opportunities, for example, for insurance products. They are also winning more upsell opportunities such as phone upgrades.

"Beyond the revenue benefits to The Carphone Warehouse, customers experience a higher level of service from us now," said the CRM program manager. "With better data quality, we have a more complete picture of our customers and can target campaigns more effectively. And when a customer makes direct contact, we know a great deal more about them. Overall, with the help of Trillium Software, we are fulfilling our mission to deliver customer delight and a better mobile life."

The company originally estimated that its investment in the Trillium Software System would be recouped in 12 months, but the company's CRM program manager stated it actually paid for itself in less than 6 months.

10.7 Summary

- The data warehouses are supposed to provide storage, functionality and responsiveness to queries beyond the capabilities of today's transaction-oriented databases. Also data warehouses are set to improve the data access performance of databases.
- Traditional databases balance the requirement of data access with the need to ensure integrity of data.
- In present day organizations, users of data are often completely removed from the data sources.
- Many people only need read-access to data, but still need a very rapid access to a larger volume of data than can conveniently by downloaded to the desktop.
- Often such data comes from multiple databases. Because many of the analyses performed are recurrent and predictable, software vendors and systems support staff have begun to design systems to support these functions.
- Currently there comes a necessity for providing decision makers from middle management upward with information at the correct level of detail to support decision-making.
- Data warehousing, online analytical processing (OLAP) and data mining provide this functionality.

10.8 Keywords

Dimensions: Dimensions contain a set of unique values that identify and categories data.

Hierarchy: A hierarchy is a way to organize data at different levels of aggregation.

Logical Cubes: Logical cubes provide a means of organizing measures that have the same shape, that is, they have the exact same dimensions.

OLTP: Databases must often allow the real-time processing of SQL transactions to support e-commerce and other time-critical applications. This type of processing is known as online transaction processing (OLTP).

Star Schema: A star schema is a convention for organizing the data into dimension tables, fact tables, and materialized views.

10.9 Self Assessment

Fill in the blanks:

1. Data warehouses and .... tools are based on a multi-dimensional data model.
2. Measures are organized by dimensions, which typically include a ....
3. Hierarchies and levels have a .... relationship.
4. A cube shape is .... dimensional.
5. A .... stores all of the information about a dimension in a single table.
6. A .... normalizes the dimension members by storing each level in a separate table.
7. Data Warehouses use Online Analytical Processing (OLAP) to formulate and execute ....
8. The .... allows the user to summarise data into a more general level in hierarchy.
9. .... allows the user to focus the analysis of the data on a particular perspective from one or more dimensions.
10. .... products organize, navigate and analyze data typically in an aggregated form.

10.10 Review Questions

1. Describe materialized views with the help of a suitable example.
2. Write short note on snowflake schema.
3. "Concept hierarchies that are common to many applications may be predefined in the data mining system". Explain.
4. "PIVOT is used to rotate a given data cube to select a different view." Discuss
5. Describe OLAP operations in the multi-dimensional data model.
6. Describe PIVOT commands in detail with the help of suitable example.

7. Describe MOLAP model in detail.
8. Explain logical model for multidimensional information.
9. Distinguish between logical measures and logical dimensions.
10. Explain ROLL UP command in detail with the help of suitable example.

Answers: Self Assessment

1.	OLAP	2.	Time dimension
3.	many-to-many	4.	three
5.	star schema	6.	snowflake schema
7.	user queries	8.	ROLL UP
9.	SLICE	10.	MOLAP-based

10.11 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Notes

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Unit 11: Query Processing and Optimization

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain query processing
- Know description and requirements for data warehouse queries
- Describe query processing techniques

Introduction

The Query Language SQL is one of the main reasons of success of RDBMS. A user just needs to specify the query in SQL that is close to the English language and does not need to say how such query is to be evaluated. However, a query needs to be evaluated efficiently by the DBMS. But how is a query-evaluated efficiently? This unit attempts to answer this question. The unit

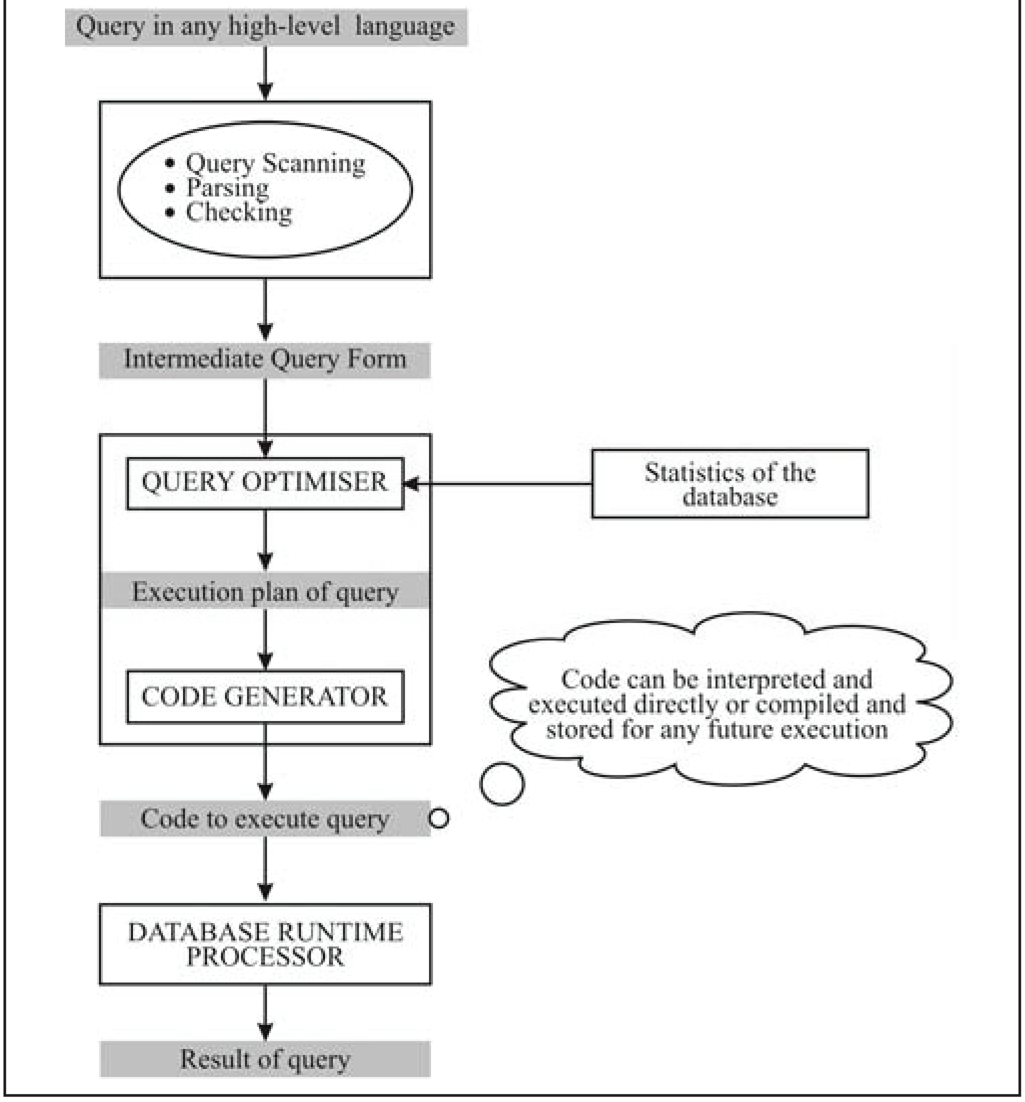
Notes

covers the basic principles of query evaluation, the cost of query evaluation, the evaluation of join queries, etc. in detail. It also provides information about query evaluation plans and the role of storage in query evaluation and optimisation.

11.1 Query Processing: An Overview

Figure 11.1 show the processing of a query.

Figure 11.1: Query Processing


Query in any high-level language
• Query Scanning
• Parsing
• Checking
Intermediate Query Form
QUERY OPTIMISER
Statistics of the database
Execution plan of query
CODE GENERATOR
Code can be interpreted and executed directly or compiled and stored for any future execution
Code to execute query
DATABASE RUNTIME PROCESSOR
Result of query

In the first step Scanning, Parsing, and Validating is done to translate the query into its internal form. This is then further translated into relational algebra (an intermediate query form). Parser checks syntax and verifies relations. The query then is optimised with a query plan, which then is compiled into a code that can be executed by the database runtime processor.

You can define query evaluation as the query-execution engine taking a query-evaluation plan, executing that plan, and returning the answers to the query. The query processing involves the study of the following concepts:

How to measure query costs?

Algorithms for evaluating relational algebraic operations.

How to evaluate a complete expression using algorithms on individual operations?

Optimisation

A relational algebra expression may have many equivalent expressions.

Example: σ (salary < 5000) (πsalary (EMP)) is equivalent to πsalary (σsalary <5000 (EMP)).

Each relational algebraic operation can be evaluated using one of the several different algorithms. Correspondingly, a relational-algebraic expression can be evaluated in many ways.

An expression that specifies detailed evaluation strategy is known as evaluation-plan, for example, you can use an index on salary to find employees with salary < 5000 or we can perform complete relation scan and discard employees with salary ≥ 5000. The basis of selection of any of the scheme will be the cost.

Query Optimisation: Amongst all equivalent plans choose the one with the lowest cost. Cost is estimated using statistical information from the database catalogue, for example, number of tuples in each relation, size of tuples, etc.

Thus, in query optimisation we find an evaluation plan with the lowest cost. The cost estimation is made on the basis of heuristic rules.

11.2 Description and Requirements for Data Warehouse Queries

Most data warehouse queries are designed to follow a star schema and can process hundreds of millions of rows in a single query. By default, the query optimizer detects queries against star schemas and creates efficient query plans for them. One method the optimizer can use to generate an efficient plan is to use bitmap filtering. A bitmap filter uses a compact representation of a set of values from a table in one part of the operator tree to filter rows from a second table in another part of the tree. Essentially, the filter performs a semi-join reduction; that is, only the rows in the second table that qualify for the join to the first table are processed.

In SQL Server 2008, bitmap filtering can be introduced in the query plan after optimization, as in SQL Server 2005, or introduced dynamically by the query optimizer during query plan generation. When the filter is introduced dynamically, it is referred to as an optimized bitmap filter. Optimized bitmap filtering can significantly improve the performance of data warehouse queries that use star schemas by removing non-qualifying rows from the fact table early in the query plan. Without optimized bitmap filtering, all rows in the fact table are processed through some part of the operator tree before the join operation with the dimension tables removes the non-qualifying rows. When optimized bitmap filtering is applied, the non-qualifying rows in the fact table are eliminated immediately.

11.2.1 Understanding Bitmap Filtering

The bitmap filter compares favorably to the bitmap index. A bitmap index is an alternate form for representing row ID (RID) lists in a value-list index using one or more bit vectors indicating which row in a table contains a certain column value. Both can be very effective in removing unnecessary rows from result processing; however, there are important differences between a bitmap filter and a bitmap index. First, bitmap filters are in-memory structures, thus eliminating any index maintenance overhead due to data manipulation language (DML) operations made to the underlying table. In addition, bitmap filters are very small and, unlike existing on-disk indexes that typically depend on the size of the table on which they are built, bitmap filters can be created dynamically with minimal impact on query processing time.

Comparing Bitmap Filtering with Optimized Bitmap Filtering

Bitmap filtering and optimized bitmap filtering are implemented in the query plan by using the bitmap show plan operator. Bitmap filtering is applied only in parallel query plans in which hash or merge joins are used. Optimized bitmap filtering is applicable only to parallel query plans in which hash joins are used. In both cases, the bitmap filter is created on the build input (the dimension table) side of a hash join; however, the actual filtering is typically done within the Parallelism operator, which is on the probe input (the fact table) side of the hash join. When the join is based on an integer column, the filter can be applied directly to the initial table or index scan operation rather than the Parallelism operator. This technique is called in-row optimization.

When bitmap filtering is introduced in the query plan after optimization, query compilation time is reduced; however, the query plans that the optimizer can consider are limited, and cardinality and cost estimates are not taken into account.

Optimized bitmap filters have the following advantages:

1. Filtering from several dimension tables is supported.
2. Multiple filters can be applied to a single operator.
3. Optimized bitmap filters can be applied to more operator types. These include exchange operators such as the Distribute Streams and Repartition Streams operators, table or index scan operators, and filter operators.
4. Filtering is applicable to SELECT statements and the read-only operators used in INSERT, UPDATE, DELETE, and MERGE statements.
5. Filtering is applicable to the creation of indexed views in the operators used to populate the index.
6. The optimizer uses cardinality and cost estimates to determine if optimized bitmap filtering is appropriate.
7. The optimizer can consider more plans.

How Optimized Bitmap Filtering is Implemented?

A bitmap filter is useful only if it is selective. The query optimizer determines when a optimized bitmap filter is selective enough to be useful and to which operators the filter is applied. The optimizer places the optimized bitmap filters on all branches of a star join and uses costing rules to determine whether the plan provides the smallest estimated execution cost. When the optimized bitmap filter is nonselective, the cost estimate is usually too high and the plan is discarded. When considering where to place optimized bitmap filters in the plan, the optimizer looks for hash join variants such as a right-deep stack of hash joins. Joins with dimension tables are implemented to execute the likely most selective join first.

The operator in which the optimized bitmap filter is applied contains a bitmap predicate in the form of PROBE([Opt_Bitmap1001], {[column_name]} [, 'IN ROW']). The bitmap predicate reports on the following information:

1. The bitmap name that corresponds to the name introduced in the Bitmap operator. The prefix 'Opt_' indicates an optimized bitmap filter is used.

2. The column probed against. This is the point from which the filtered data flows through the tree.
3. Whether the bitmap probe uses in-row optimization. When it is, the bitmap probe is invoked with the IN ROW parameter. Otherwise, this parameter is missing.

11.2.2 Optimized Bitmap Filtering Requirements

Optimized bitmap filtering has the following requirements:

1. Fact tables are expected to have at least 100 pages. The optimizer considers smaller tables to be dimension tables.
2. Only inner joins between a fact table and a dimension table are considered.
3. The join predicate between the fact table and dimension table must be a single column join, but does not need to be a primary-key-to-foreign-key relationship. An integer-based column is preferred.
4. Joins with dimensions are only considered when the dimension input cardinalities are smaller than the input cardinality from the fact table.

Task Describe the main activities associated with various design steps of data warehouse?

11.3 Writing your own Queries

When writing your own query, you define the set of choices and conditions that you want to use to retrieve data stored in the Warehouse. You determine such attributes as data elements to be returned (e.g., last name, city, age), conditions for selecting records (e.g., equal to or less than), and sort criteria (the order and priority in which results are to be sorted).

You may want to keep in mind some general questions to guide you in composing a query:

1. What information are you looking for? In what collection does this data reside, and which Business Object universe would be best to use?
2. Bear in mind the level of detail data you need, the time periods concerned, and which source system you would use to verify the data retrieved.
3. Once you have a basic idea of the results you need, consider how the query should be contrained – by time period? account segment(s)? employee or organization names/ codes?
4. What will you do with your results? If you are presenting them to others, you may want to include segment descriptions for those unfamiliar with codes. Also, if you plan to export the data, you may want to include objects which you have used to constrain your query, to better identify the data in the exported file. (For example, although you may have used Accounting_Period as a constraint, it might help to have the period appear as a column in your exported file, so you know what period that data represents.)

11.4 Query Processing Techniques

In this section, introduce to you four current query processing techniques that are used in Data Warehousing queries. This is followed by a description of a cost model that I have adopted and later to be used when I compare the performance of the different query processing techniques.

The performance of these query processing techniques are compared using a disk I/O. Finally, we propose a recommendation for Database Management Systems (DBMSs) to select the most cost-effective query processing techniques based on the cost model

Before you can analyze and interpret customers buying behavior, we have to store all the historical information of customers. This voluminous amount of data is most suitably stored in a data warehouse. The current DBMSs mainly deal with daily operation of business such as Transaction Processing System (TPS) involving reading and writing data. Data warehouse handles the large read only information. The volume of data in data warehouse is increasing rapidly. If a query was to be made on this large volume of data, the response time will, as expected, not be good. The next question that comes to mind will be: How can we achieve a reasonable response time? Response time relies heavily on query processing technique and experiment on new techniques. A new reliable and faster query processing technique will greatly enhance the performance of data warehousing processes..

11.4.1 Relational Model

Suppose we are working with a relation *R* that has m number of tuples {t_1, t_2, … t_m }, and *O* number of attributes {a_1, a_2, …, a_0}, the size *S* of *O* number of attributes {s_1, s_2, …, s_o} and instance of tuples {$i_{1\,1}$, $i_{1\,2}$, $i_{1\,o}$, …, $i_{m\,o}$}. The term's *relation* and *table* are used. Table 11.1 shows the relational model *R*.

Table 11.1: The Relational Model R

a_1	a_2	. . . .	a_4	. . . .
i_{11}	i_{12}	. . . .	i_{14}	. . . .
:	:	:	:	:
:	:	:	:	:
i_{m1}	i_{m2}	:	i_{m4}	:

Assumptions

Let us assume that:

The access time per tuple with 1000 bytes is 0.1 sec.

There are 10,000 tuples in the R relation.

The disk I/O fetches 100 bytes in 0.01 sec.

In a data warehousing environment, the information sources are made up of combined data from different sources. These data are fed into a central repository from the dispersed client locations and form the R relation. In such a situation, usually the respective clients are only interested in their own data sets. Suppose, the particular client wants to display a_1 and a_4. Given this situation, how can we achieve a reasonable response time?

11.4.2 Cost Model

Here, we introduce a cost model to measure the performance of query processing techniques. In general, there are three components to make up the response time of a query. Scanning the data in the table level only will exclude the access time of the index level such as in the case of the full scan technique. Scanning the data in the index level only will exclude the access time of the table level such as in the case of the inverted partitioned indexes. Table 11.2 shows the cost model for respective query processing techniques. To measure the response time of a query, we need to submit what type of query processing techniques as a parameter such as Full Scan, Index techniques or Inverted Partitioned Index. Thus, DBMSs will be able to pick up what components required.

Example:

CostModel (Inverted Partitioned Index) = Access Time of Index Level + Instruction Time (Inverted Partitioned Index)

Table 11.2: Cost Model

CostModel (Type)	Component 1 (Access Time of the table level)	Component 2 (Access Time of Index level)	Component 3 Instruction Time (the algorithms)
Full Scan without an Index Technique	Yes	No	Instruction Time (Full Scan)
Scan with Index Technique	Yes	Yes	Instruction Time (Index Technique)
Inverted Partitioned Index	No	Yes	Instruction Time (Inverted Partitioned Index)

Let us further assume that:

Table 11.3: Instruction Time of Respective Query Processing Algorithms

Query Processing Techniques	Instruction Time (the algorithms)
Full Scan	0.05 sec.
RID Index (Index Technique 1)	0.07 sec.
BitMap Index (Index Technique 2)	1.2 sec.
Inverted Partitioned Index	0.04 sec.

From Table 11.3, we observe that the Instruction Time of BitMap index is the highest because it has most steps than another. To simplify our calculation, given a query, we assume that the found set of R relation is from 10%, 20%, …, 90%, 100%. The selective attributes is/are from 1,2, …, O.

11.4.3 Full Scan Technique

A full scan technique will scan a relation R from the top until the end of the table. The total time that is required to process a query is the total time taken to read the entire table.

Table 11.4: Full Scan Technique In Relation Model R

Start Scanning

a_1	a_2	. . . .	a_4	. . . .
i_{11}	i_{12}	. . . .	i_{14}	. . . .
:	:	:	:	:
:	:	:	:	:
i_{m1}	i_{m2}	:	i_{m4}	:

End Scanning

Notes

The response time of a query using the Full Scan technique:

$$CostModel(m) = (m * att) + (m * itime(fs))$$

Where,

m = Total number of tuples in the relation

att = Access time per tuple

itime = Instruction time of an algorithm

fs = Full scan algorithm

Since the full scan technique is unaffected by the found set of department relation and the number of selective attributes. Therefore;

Response time of a query = (10,000 * 0.1) + (10,000 * 0.05) = 1,500 sec.

The response time of a query remains constant. The average response time for 10% to 100% found set is 1,500 sec.

11.4.4 Scanning with Index Techniques

To access all the tuples in a relation is very costly when there are only small found sets. Since fetching data in the index level is normally 1/10 smaller than table level, therefore, an indexing technique is introduced. For example, to fetch data from the table level is 0.1 sec. and to fetch data from index level is 0.01 sec. Query processing will first process data in the index level and then only will it fetches data from the table level. In the following subsections, we discuss the efficiencies of the RID index technique and BitMap Index Technique in query processing.

Task A database has four transactions. Let min sup = 60% and min_conf = 80%

Tid	date	intms_bought
T100	10/15/99	{K,A,D,B}
T200	10/15/99	{D,A,C,E,B}
T300	10/19/99	{C,A,B,E,}
T400	10/22/99	{B,A,D}

1. Find all frequent item sets using FP_growth and Apriori techniques. Compare the efficiency of the two mining processes.
2. List all the strong association rules (with support and confidence c) matching the following meta rule, where X is a variable representing customers and item i denotes variable representing items (eg "A","B" etc.)

Transaction, buys (X item$_1$) buys (X, item$_2$)=>(X, item$_3$) [s,c]

11.4.5 RID Index Technique

The RID index technique is one of the traditional index techniques used in the Transaction Processing System (TPS). The RID index creates a list of record identification which acts as pointers to records in the table. The total time that is required to process a query is access time of index level and access time of selective table level.

The response time of a query in RID Index Technique is:

CostModel(sc) = (|sc| * att) + (|sc| * ati) + (|sc| * itime(ridi))

Where,

sc =	Selective conditions
|sc| =	The total number of found sets in selective conditions
m =	Total number of tuples in the relation
ati =	Access time per index
att =	Access time per tuple
itime =	Instruction time of an algorithm
ridi =	RID index algorithm

The selective conditions determine the found set of tuples. Refer to Table 11.5. If this is the largest found set, then the total query response time may be more than the full scan technique. Therefore, the resulting cost is important to determine whether the index technique will be selected or not.

Example:

|sc| = found set * total number of tuples = 10% * 10,000 = 1000

Let att = 0.1

Let ati = 0.01

Let itime(rdi) = 0.07

The response time of a query

= |sc| * (att + ati + itime(rdi))

= 1000 * (0.1 + 0.01 + 0.07) = 180 sec.

If the found set is 20%, it will be 20% * 10,000 * 0.18 = 360 sec. and the response times are as depicted in Table 11.5.

Figure 11.5: The Response Time of RID Index in Different Found Sets

Found set	Response time	Found set	Response time
10%	180 sec.	60 %	1080 sec.
20%	360 sec.	70 %	1260 sec.
30%	540 sec.	80 %	1440 sec.
40%	720 sec.	90 %	1620 sec.
50%	900 sec.	100 %	1800 sec.

Based on Table 11.5, the average response time from 10% to 100% found sets is:

(180 + 360 + 540 + 720 + 900 + 1080 + 1260 + 1440 + 1620 + 1800) / 10 = 990 sec..

From Table 11.5, we know that the found sets are between 80 % and 90 %, the responses time are between 1440 sec. and 1620 sec.. Moreover, the response time of full scan without index is 1500 sec. We can therefore use the cost model to derive at the actual percentage of found set.

The response time of a query = |sc| * (att + ati + itime(rid))

Given: The response time of a query = 1500 sec. Where,

att = 0.1

ati = 0.01

itime = 0.07

total number of tuples = 10,000

step 1) 1500 = 10,000 * found set * (0.1 + 0.01 + 0.07)

step 2) 1500 = 1800 * found set

step 3) found set = 1500/1800 = 83.33 %

From the result, we can conclude that the full scan technique outperforms the RID index technique when there is more than or equal to 83.33 % found set in the relation. Therefore, DBMSs shall use full scan technique instead of the RID index technique.

11.4.6 BitMap Index Technique

The BitMap Index technique saves more spaces than the RID index technique that stores bits instead of record identification. The total time that is required to process a query is 1/8 of access time of index level and access time of selective table level.

The response time of a query in BitMap is

$$\text{CostModel(sc)} = 1/8 * ((|sc| * att) + (|sc| * ati)) + (|sc| * itime(bm)))$$

Where,

sc =	Selective conditions
|sc| =	The total number of found sets in selective conditions
m =	Total number of tuples in the relation
att =	Access time per tuple
itime =	Instruction time of an algorithm
ati =	Access time per index
bm =	BitMap index algorithm

The response time of a query seems faster than the RID index technique. However, the instruction time of a BitMap index has more steps than the RID index algorithm.

Example:

Let |sc| = found set * total number of tuples = 10% * 10,000 = 1,000

Let att = 0.1

Let ati = 0.01

Let itime(bm) = 1.2

The response time of a query:

= 1/8 (|sc| * (att + ati + itime(bm)))

= 1/8 (1,000 * (0.1 + 0.01 + 1.2)) = 164 sec..

If the foundset is 20%, then the response time of a query will be:

= 1/8(2,000 * (0.1 + 0.01 + 1.2)) = 382 sec. and the response times are as summarized in Table 11.6.

Notes

Table 11.6: The Response Time of Bitmap Index in Different Foundsets

Found set	Response time	Found set	Response time
10%	164 sec.	60 %	984 sec.
20%	328 sec.	70 %	1148 sec.
30%	492 sec.	80 %	1312 sec.
40%	656 sec.	90 %	1476 sec.
50%	820 sec.	100 %	1640 sec.

Based on Table 11.6, the average response time from 10% to 100% found sets is:

(164 + 328 + 492 + 656 + 820 + 984 + 1148 + 1312 + 1476 + 1640) / 10 = 902 sec.

From the Table 11.6, we found that for the found sets between 90% and 100%, the response times are between 1476 sec. and 1640 sec.. Note that thre response time of full scan without index is 1500 sec. We can now use the cost model to derive the actual percentage of found set.

The response time of a query = 1/8 (|sc| * (att + ati + itime(bm)))

Given: The response time of a query = 1500 sec.

att = 0.1

ati = 0.01

itime = 1.2

total number of tuples = 10,000

step 1) 1,500 = 1/8 * (10,000 * found set (0.1 + 0.01 + 1.2))

step 2) 12,000 = 13,100 * found set

step 3) found set = 12,000 / 13,100 = 91.6 %

From the result, we conclude that the full scan technique outperforms the RID index technique when there is more than or equal to 91.6% found set in the relation. Therefore, DBMSs will use the full scan technique instead of the BitMap index technique.

11.4.7 Inverted Partitioned Index Technique

The technique of Inverted Partitioned processes data in index level only. Therefore, it reduces the time to access data from two levels.

The response time of a query in Inverted Partitioned Index is

$$\text{CostModel}(sc) = (|sc| * ati) + (|sc| * itime(ipi))$$

Where,

sc =	Selective conditions
|sc| =	The total number of found sets in selective conditions
ati =	Access time per index
itime =	Instruction time of an algorithm
ipi =	Inverted Partitioned Index algorithm

Notes

Example:

Let |sc| = found set * total number of tuples = 10% * 10,000 = 1,000

Let ati = 0.01

Let itime(bm) = 0.04

The response time of a query = |sc| * ati * itime(rdi)

= 1,000 * (0.01 + 0.04) = 50 sec..

If the foundset is 20%, it will be 2,000 * (0.01 + 0.04) = 100 sec. and the response time are summarised in Table 11.7.

Table 11.7: The Response Time of BitMap Index in different Found Sets

Found set	Response time	Found set	Response time
10%	50 sec.	60 %	300 sec.
20%	100 sec.	70 %	350 sec.
30%	150 sec.	80 %	400 sec.
40%	200 sec.	90 %	450 sec.
50%	250 sec.	100 %	500 sec.

Based on Table 11.7, the average response time from 10 % to 100% found sets is (50 + 100 + 150 + 200 + 250 + 300 + 350 + 400 + 450 + 500)/ 10 = 275 sec.

11.5 Comparison of the Query Processing Strategies

The Comparison of the Query Processing Strategies is shown in Table 11.8.

Table 11.8: Comparison of the Query Processing Strategies

Query Processing Strategies	Full Scan Technique	RID Index Technique	BitMap Index Technique	Inverted Partition Index Technique
Data Processing Level	Table Level	Index and Table Level	Index and Table Level	Index Level
Average Response time	1500 Sec.	990 sec.	902 sec.	275 sec.
Time required to execution Instructions of Algorithm	0.05 sec.	0.07 sec.	1.2 sec.	0.04 sec.
Small Selective Tuples	Bad	Good	Good	Excellent
Large Selective Tuples	Excellent	Average	Average	Excellent
Small Selected Attributes	Bad	Bad	Bad	Excellent
Select or Retrieve all Attributes from table level	Yes	Yes	Yes	No
Frequently set attributes	No	No	No	No

Notes

Based on the Table 11.8, you learnt that full scan technique and scanning with index techniques are not good for small selected attributes.

Case Study

Oki Data Americas, Inc.

Better Data Extends Value

How can you extend the value of a new software implementation? Follow the example of Oki Data Americas, Inc. A year after the company implemented SAP R/3 business software, it realized low data quality was limiting the value of its new data solution. Oki Data used the Trillium Software System® to consolidate data from legacy systems within its SAP database, increasing both the value of its data solution and the resolution of its customer view.

Oki Data needed to create a single, comprehensive data source by:

1. Identifying and deduplicating customer records in its SAP and legacy systems.
2. Identifying unique customer records in the legacy systems and loading them into the SAP database.
3. Consolidating legacy systems and SAP data to ensure completeness of customer records.

Manually consolidating the data was an unappealing prospect. Oki Data would have had to match records and make data conversions manually. In addition to being time-consuming, manual consolidation would provide virtually no improvement in the quality of name and address data.

It became clear to Oki Data that it needed a data quality solution that could instill higher quality in diverse data from across the enterprise.

Three Months from Concept to Remedy

Oki Data quickly identified Trillium Software as its data quality solution provider. The Trillium Software System met more of Oki Data's 30 solution criteria than other solution candidates.

Oki Data needed a data quality solution that functioned within a heterogeneous environment. The platform-independent architecture of the Trillium Software System ensured compatibility across all past and present systems.

The Trillium Software solution cleansed, standardized, deduplicated and matched Oki Data's US and Canadian customer data. Oki Data particularly liked the software's ability to correct (geocode) addresses according to US postal directories. During the evaluation process, Trillium Software was the only vendor to provide Oki Data with a knowledge transfer that demonstrated the process and results of its data quality process using Oki Data's unique information.

Breadth of functionality was another area in which the Trillium Software System excelled. The solution let Oki Data standardize and correct postal records for US and Canadian customers. Moreover, it could process business data—such as marketing codes—included in customer records.

The systems analyst at Oki Data said, "It is clear that data quality is an important issue, and the Trillium Software System is a good tool for helping us ensure quality data. On-site training and consulting that came with the software solution resulted in faster solution implementation."

Contd...

Oki Data's color and monochrome LED page printers are the hallmark of its technology leadership. Providing results comparable to a laser printer's, Oki Data's LED printers are faster and have fewer moving parts. Over 29 years customers have relied on Oki Data's proven, reliable products. Oki Data's five-year warranty on its LED printhead is a testament to its ability to meet these customer expectations. Oki Data has created a unified customer view.

Rapid ROI

Oki Data has wrought substantial value across the business from its investment in the Trillium Software System. ROI indicators include:

1. Fewer shipping and mailing errors due to verified and corrected addresses
2. Less time spent by users correcting data errors
3. Improved customer service based on more complete customer views

A Bright Future

Oki Data's process has underscored the importance of data quality and impelled the company to consider data quality as an enterprise-wide issue. The Trillium Software System's enterprisewide compatibility and real-time online processing capabilities make it a candidate for further initiatives within Oki Data. One such initiative is online data quality processing in conjunction with Oki Data's e-commerce website.

The data analyst noted, "We keep finding areas beyond the scope of the original project where a Trillium Software application will solve a new and evolving business need. Going forward, we see other potential uses of the Trillium Software System to help us better our data quality.

11.6 Summary

- In this unit you have study query processing and evaluation.
- A query in a DBMS is a very important operation, as it needs to be efficient.
- Query processing involves query parsing, representing query in alternative forms, finding the best plan of evaluation of a query and then actually evaluating it.
- The major query evaluation cost is the disk access time.

11.7 Keywords

Index Scan: Search algorithms that use an index are restricted because the selection condition must be on the search-key of the index.

Indexing: A database index is a data structure that improves the speed of operations on a database table.

Join: Join operation is considered as the real power behind the relational database implementations.

Query Cost: Cost is generally measured as total elapsed time for answering the query.

11.8 Self Assessment

Choose the appropriate answers:

1. RDBMS stands for
    (a) Relative Database Management System
    (b) Relational Database Management System
    (c) Relative Document Management System
    (d) Relational Document Management System
2. DML stands for
    (a) Data Manipulation Language
    (b) Data Multiplication Language
    (c) Dynamic Multipurpose Language
    (d) Data Manipulation Logic

Fill in the blanks:

3. Cost is generally measured as total elapsed time for answering the ....
4. The I/O cost depends on the ....
5. .... is applicable when the selection is an equality comparison on the attribute on which file is ordered.
6. .... search retrieves a single record that satisfies the corresponding equality condition.
7. .... use equivalence rules to systematically generate expressions equivalent to the given expression.
8. .... is applied only in parallel query plans in which hash or merge joins are used.
9. Data warehouse handles the large .... information.
10. The RID index technique is one of the traditional index techniques used in the ....

11.9 Review Questions

1. What do you mean by query processing? Explain.
2. Describe BitMap Index technique description with the help of an example.
3. Explain cost model in detail with any live example.
4. "A new reliable and faster query processing technique will greatly enhance the performance of data warehousing processes". Explain.
5. "Optimized bitmap filtering is applicable only to parallel query plans in which hash joins are used." Discuss.
6. How will you write your own queries? Explain.
7. Describe relational model with the help of suitable example.

Notes

8. Distinguish between full scan technique and RID index technique.
9. How will you calculate cost for simple hash-join? Explain.
10. Explain "A bitmap filter is useful only if it is selective".

Answers: Self Assessment

1. (b)
2. (a)
3. query
4. search criteria
5. Binary search
6. Primary index-scan for equality
7. Query optimizers
8. Bitmap filtering
9. read only
10. Transaction Processing System (TPS)

11.10 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Notes

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 12: Metadata and Warehouse Quality

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe metadata and warehouse quality
- Know metadata management in data warehouse practice
- Explain repository model for the DWQ framework
- Define data warehouse quality

Introduction

Many researchers and practitioners share the understanding that a data warehouse (DW) architecture can be formally understood as layers of materialized views on top of each other. A data warehouse architecture exhibits various layers of data in which data from one layer are derived from data of the lower layer. Data sources, also called stored in open databases, form the lowest layer. They may consist of structured data stored in open database system and legacy systems, or unstructured or semi-structured data stored in files.

12.1 Metadata and Warehouse Quality

To be successful throughout the process, it's important to know what you did right and wrong. In essence, you need "meta data" about what you're doing to accurately measure quality. Successful measurement is the key to warehouse quality, but how do you measure a data warehouse? Wells and Thomann offer a number of different methods, dimensions, and levels for understanding data warehouse measurement. To begin, Thomann describes three "types" of success and their definitions as they relate to data warehousing:

1. ***Economic success:*** The data warehouse has a positive impact on the bottom line.
2. ***Political success:*** People like what you've done. If the data warehouse isn't used, it's obvious that you failed politically.
3. ***Technical success:*** This is the easiest to accomplish. However, don't overwhelm your users with too much technology. Success also means that the chosen technologies are appropriate for the task and are applied correctly.

Since there are three ways to succeed, Wells responds, there are three ways to fail. Quality, he says, can be defined as the degree of excellence of something. "But this is a very subjective measure, and you can make it more subjective through measurement." Three main areas are detailed in the paragraphs below, which can be used to assess the overall quality of a data warehouse.

12.1.1 Business Quality

Directly related to economic success, business quality is the ability of the data warehouse to provide information to those who need it, in order to have a positive impact on the business.

Business quality is made up of business drivers, or concepts that point out a company's strategic plans. So organizations should be concerned with how well the data warehouse helps accomplish these drivers, such as changing economic factors, environmental concerns, and government regulation.

Does the data warehouse align with business strategy, and how well does it support the process of strengthening core competencies and improving competitive position? What about the enablement of business tactics? Does the data warehouse play a tactical role, so that it makes a positive day-to-day difference?

12.1.2 Information Quality

Information doesn't have value if it's not used. Therefore to have information quality, the focus should be on the integration of information into the fabric of business processes, not on data quality itself.

Information quality is the key to political success, which was described above as people actually using the data warehouse. "Some companies don't tell users it's there," Thomann says, "so they may not understand the warehouse or know how to use it." Success in this area means providing awareness, access tools, and the knowledge and skills to use what they're given. For example, could your users easily make the shift from using greenbar reports to a multidimensional data model? Then, assuming they understand the warehouse, can they get to the data easily? Who gets the data? How frequently? How and when is the data used? You may be able to provide 24x7 access, but what if users are at home?

Wells and Thomann believe that information quality also encompasses data quality and performance. This can be a problem area, because everyone is happy if they can get their data overnight. Then they want it in half a day. Then immediately. So these expectations must be managed.

12.1.3 Technical Quality

Technical quality is the ability of the data warehouse to satisfy users' dynamic information needs. Wells describes four important technical quality factors. The first is "reach," or whether the data warehouse can be used by those who are best served by its existence. In today's information-dependent business climate, organizations need to reach beyond the narrow and typical customer base of suppliers, customers, and a few managers.

"Range" is also important. As its name implies, this defines a range of services provided by the data warehouse. In general, these include "What data do I have, and, can I get the data?" For example, Web enablement, such as Hotmail, are services which allow users to get information from wherever they are.

"Manuverability" is the ability of the data warehouse to respond to changes in the business environment. The data warehouse doesn't remain stable, so manuverability becomes particularly important. It is also the single most important factor not given attention today in data warehousing, according to Wells. Manuverability sub-factors include managing:

1. Users and their expectations,
2. Upper management,
3. The overall business,
4. Technology,
5. Data sources, and
6. Technical platform.

Finally, "capability" is an organization's technical capability to build, operate, maintain, and use a data warehouse.

12.2 Matadata Management in Data Warehousing

Although organizations are now successfully deploying data warehousing and decision processing products for providing business users with integrated, accurate and consistent business information, most companies have failed to provide a similar level of integration of the meta data associated with this business information. This result is caused not only by a lack of understanding of the importance of meta data, but also because meta data integration is a complex task. The trend toward the use of packaged analytic applications in data warehousing will make meta data integration even more difficult. Data warehouse customers urgently need to address this problem if the full benefits of data warehousing are to be achieved. In this article we explain why meta data management and standardization is important to the success of a data warehouse and explore industry efforts in this area.

12.2.1 Benefits of Integrated Meta Data

To date in data warehousing most organizations have avoided the issue of meta data management and integration. Many companies, however, are now beginning to realize the importance of meta data in decision processing and to understand that the meta data integration problem cannot be ignored. There are two reasons for this:

1. The use of data warehousing and decision processing often involves a wide range of different products, and creating and maintaining the meta data for these products is time-consuming and error prone. The same piece of meta data (a relational table definition, for example) may have to be defined to several products. This is not only cumbersome, but also makes the job of keeping this meta data consistent and up to date difficult. Also, it is often important in a data warehousing system to track how this meta data changes over time. Automating the meta data management process and enabling the sharing of this so-called technical meta data between products can reduce both costs and errors.
2. Business users need to have a good understanding of what information exists in a data warehouse. They need to understand what the information means from a business viewpoint, how it was derived, from what source systems it comes, when it was created, what pre-built reports and analyses exist for manipulating the information, and so forth. They also may want to subscribe to reports and analyses and have them run, and the results delivered to them, on a regular basis. Easy access to this business meta data enables business users to exploit the value of the information in a data warehouse. Certain types of business meta data can also aid the technical staff examples include the use of a common business model for discussing information requirements with business users and access to existing business intelligence tool business views for analyzing the impact of warehouse design changes.

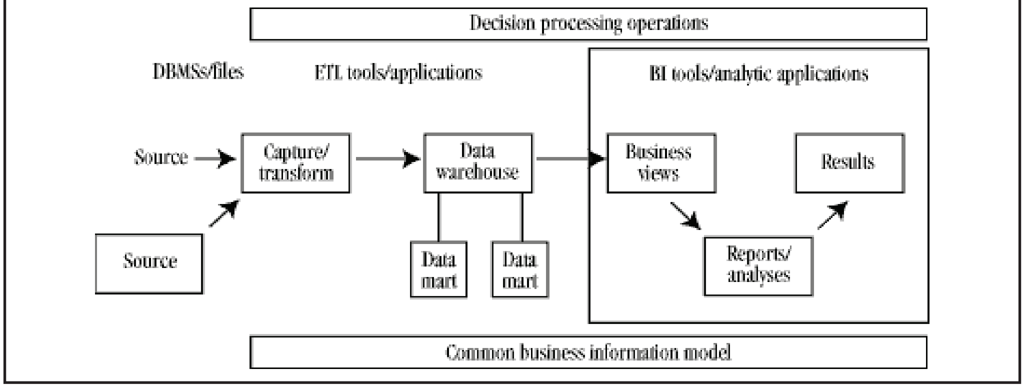
12.2.2 Improved Productivity

The benefit of managing data warehouse technical meta data is similar to those obtained by managing meta data in a transaction processing environment improved developer productivity. Integrated and consistent technical meta data creates a more efficient development environment for the technical staff who are responsible for building and maintaining decision processing systems. One additional benefit in the data warehousing environment is the ability to track how meta data changes over time. The benefits obtained by managing business meta data, on the other hand, are unique to a decision processing environment and are key to exploiting the value of a data warehouse once it has been put into production.

Figure 12.1 shows the flow of meta data through a decision processing system as it moves from source systems, through extract and transformation (ETL) tools to the data warehouse and is used by business intelligence (BI) tools and analytic applications. This flow can be thought of as a meta data value chain. The further along the chain you go, the more business value there is in the meta data. However, business value depends on the integrity of the meta data in the value chain. As meta data is distributed across multiple sources in the value chain, integrity can only be maintained if this distributed meta data is based on a common set of source meta data that is current, complete and accurate. This common set of source meta data is often called the meta data system of record. Another important aspect of the value chain is that business users need to be able to follow the chain backward from the results of decision processing to the initial source of the data on which the results are based.

Notes

Figure 12.1 (a)



There are two items in Figure 12.1(a) that have not been discussed so far. The decision processing operations box in the diagram represents the meta data used in managing the operation of a decision processing system. This includes meta data for tracking extract jobs, business user access to the system, and so forth. The common business model box represents the business information requirements of the organization. This model provides a high-level view, by business subject area, of the information in the warehousing system. It is used to provide common understanding and naming of information across warehouse projects.

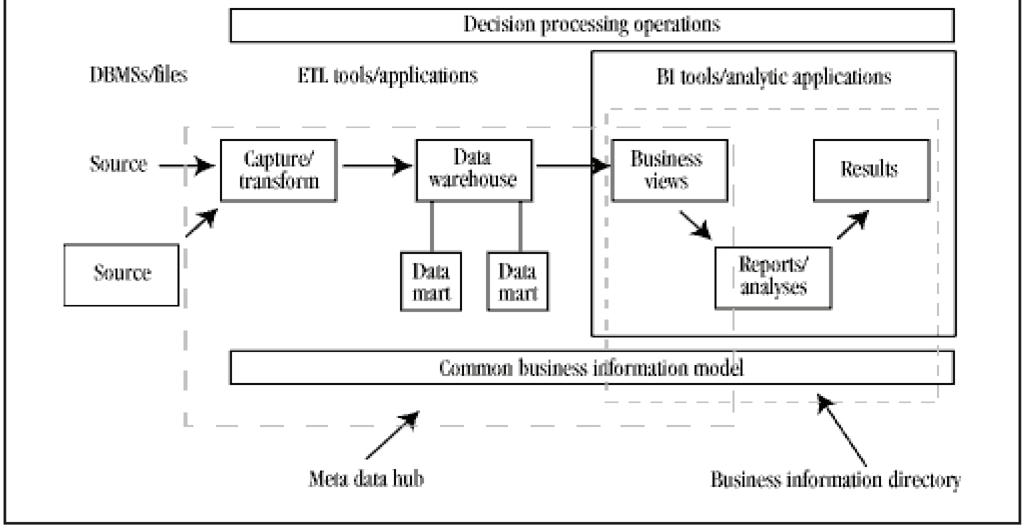
12.2.3 Meta Data Sharing and Interchange

Past attempts by vendors at providing tools for the sharing and interchange of meta data have involved placing the meta data in a central meta data store or repository, providing import/ export utilities and programmatic APIs to this store and creating a common set of meta-models for describing the meta data in the store. In the transaction processing environment, this centralized approach has had mixed success, and there have been many failures. In the decision processing marketplace, vendors are employing a variety of centralized and distributed approaches for meta data management. The techniques used fall into one of three categories:

1. Meta data repositories for meta data sharing and interchange
2. Meta data interchange "standards" defined by vendor coalitions
3. Vendor specific "open" product APIs for meta data interchange.

Given that multiple decision processing meta data approaches and "standards" are likely to prevail, we will, for the foreseeable future, be faced with managing multiple meta data stores, even if those stores are likely to become more open. The industry trend toward building distributed environments involving so-called federated data warehouses, consisting of an enterprise warehouse and/or multiple data marts, will also encourage the creation of multiple meta data stores. The only real solution to meta data management is to provide a facility for managing the flow of meta data between different meta data stores and decision processing products. This capability is provided using a meta data hub for the technical staff and a business information directory for business users Figure 12.1(b).

Figure 12.1(b)



12.2.4 Why two Tools for Meta Data Management?

There are continuing debates in the industry about whether technical and business meta data should be maintained in the same meta data store or kept separate. This debate looks at meta data from the wrong dimension. Meta data should be viewed from the perspective of the person using it not simply from the type of meta data it is. In general, technical staff employ technical meta data during warehouse development and maintenance. They also need access to certain types of business meta data. Examples include the common business model when discussing information requirements with business users and BI tool business views when analyzing the impact of warehouse design changes. Business users, on the other hand, employ business meta data as an aid to finding the business information they require in a warehouse; but they also use high-level technical meta data when trying, for example, to relate decision processing results to the source data used to create the results.

There is a strong argument in favor in of the deployment of the two types of meta data management tools: a GUI- or Web-based tool (a meta data hub) for technical staff when developing and maintaining the data warehouse and a Web- based tool (a business information directory) for business users when employing decision processing tools and applications. The reason for this separation is that the usage, architecture and interfaces for the two types of meta data user are completely different. There does, however, need to be a link between the two types of tool. Users of a business information directory need to be able to drill through from a business information directory to the technical meta data maintained by a meta data hub. To facilitate this drill- through facility, the two types of meta data management tools might employ a common meta data store.

The approaches and products are targeted at technical users and at providing support for one or more requirements of a meta data hub. For this reason, in the remainder of this article, we will focus on the architecture and requirements of a meta data hub. It is important to point out, however, that vendors are also working on meta data management tools for business users. These tools support the concept of a business information directory and are being integrated into a Web-based interface known as an information portal.

Task Discuss the role of Meta data repository in a data warehouse? How does it differ from a catalog in a relational DBMS?

Notes

12.2.5 The Meta Data Hub

The meta data hub is used for managing the interchange and sharing of technical meta data between decision processing products. It is intended for use primarily by technical staff during the development and maintenance of data warehouses. The four main requirements of such a hub are:

1. A meta data hub should support the interchange of meta data between systems and products in a distributed meta data environment. The hub should have a documented and open programmatic object interface (employing COM or CORBA, for example) that enables third-party tools to use the services of the hub. A file transfer mechanism supporting industry recognized file formats (comma delimited file, Meta Data Coalition MDIS, Microsoft XML Interchange Format, for example) should also be provided for meta data interchange.

2. A meta data hub should provide persistent stores for the management and sharing of meta data. Meta data in a store should be maintainable by the object API and file transfer methods outlined above and via supplied GUI and Web client interactive interfaces. An interactive and batch meta data impact analysis and reporting feature is also required. The hub should offer an agent interface that can scan and capture, at user-defined intervals, local products and systems for new or modified meta data for adding to the meta data store. The meta data manager used to maintain meta data in the store should support version and library control features that can create a historical record of meta data changes and support group development. In large distributed environments, the administrator should be able to physically partition the meta data environment across multiple hub servers and meta data stores.

3. The meta data hub should, at a minimum, be able to manage data warehouse information store definitions. Formats supported should include relational tables and columns, and multidimensional measures and dimensions. Another type of meta data that could be handled is information about the data sources used to create data warehouse information and about the transforms applied to this source data before it is loaded in a warehouse. It is recognized, however, that current ETL tools use their own proprietary transformation methods, making it difficult to create a generalized facility for managing this type of meta data. The product should at least provide the ability to document data source and transformation meta data in free-form text format. Ideally, the hub should also document details about the business meta data associated with the common business model discussed earlier and the business views employed by business intelligence tools and analytic applications to access warehouse information.

   The hub should use industry-standard meta data models or supply its own meta-models for the various types of meta data it manages. These meta-models should be fully documented and extensible.

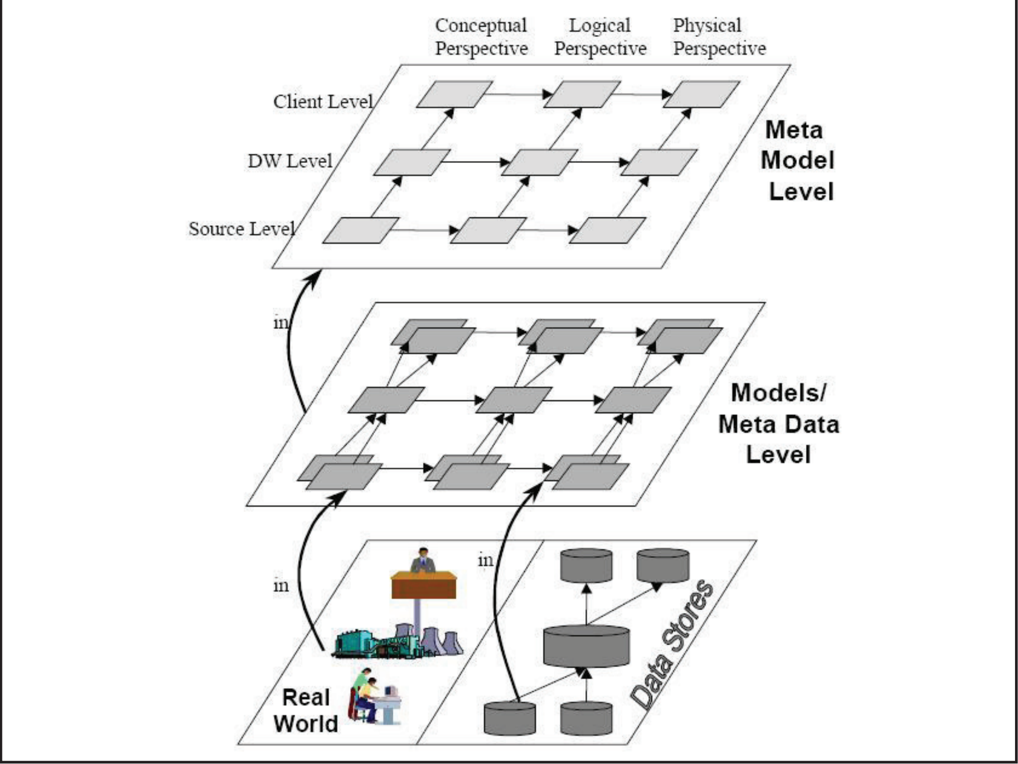
12.3 A Repository Model for the DWQ Framework

In the DWQ (Data Warehouse Quality) project we have advocated the need for enriched metadata facilities for the exploitation of the knowledge collected in a data warehouse.

The proposed categorization of the DW metadata is based on a 3x3 framework, depicted in figure 12.2: you identified three perspectives (conceptual, logical and physical) and three levels (source, data warehouse, client). We made the observation, that the conceptual perspective, which represents the real world of an enterprise, is missing in most data warehousing projects, with the risk of incorrectly representing or interpreting the information found in the data warehouse.

Notes

Figure 12.2: The Data Warehouse Architecture Meta Model



The proposed metamodel (i.e. the topmost layer in Figure 12.2) provides a notation for data warehouse generic entities, such as schema or agent, including the business perspective. Each box shown in Figure 12.2 is decomposed into more detailed data warehouse objects in the metamodel. This metamodel is instantiated with the metadata of the data warehouse (i.e. the second layer in figure 12.2), e.g. relational schema definitions or the description of the conceptual data warehouse model. The lowest layer in Figure 12.2 represents the real world where the actual data reside: in this level the metadata are instantiated with data instances, e.g. the tuples of a relation or the objects of the real world which are represented by the entities of the conceptual model.

12.3.1 Quality Meta Model

Each object in the three levels and perspectives of the architectural framework can be subject to quality measurement. Since quality management plays an important role in data warehouses, we have incorporated it into our metamodeling approach. Thus, the quality model is part of the metadata repository, and quality information is explicitly linked with architectural objects. This way, stakeholders can represent their quality goals explicitly in the metadata repository, while, at the same time, the relationship between the measurable architecture objects and the quality values is retained.

The DWQ quality metamodel is based on the Goal-Question-Metric approach (GQM) of originally developed for software quality management. In GQM, the high-level user requirements are modeled as goals. Quality metrics are values which express some measured property of the object. The relationship between goals and metrics is established through quality questions.

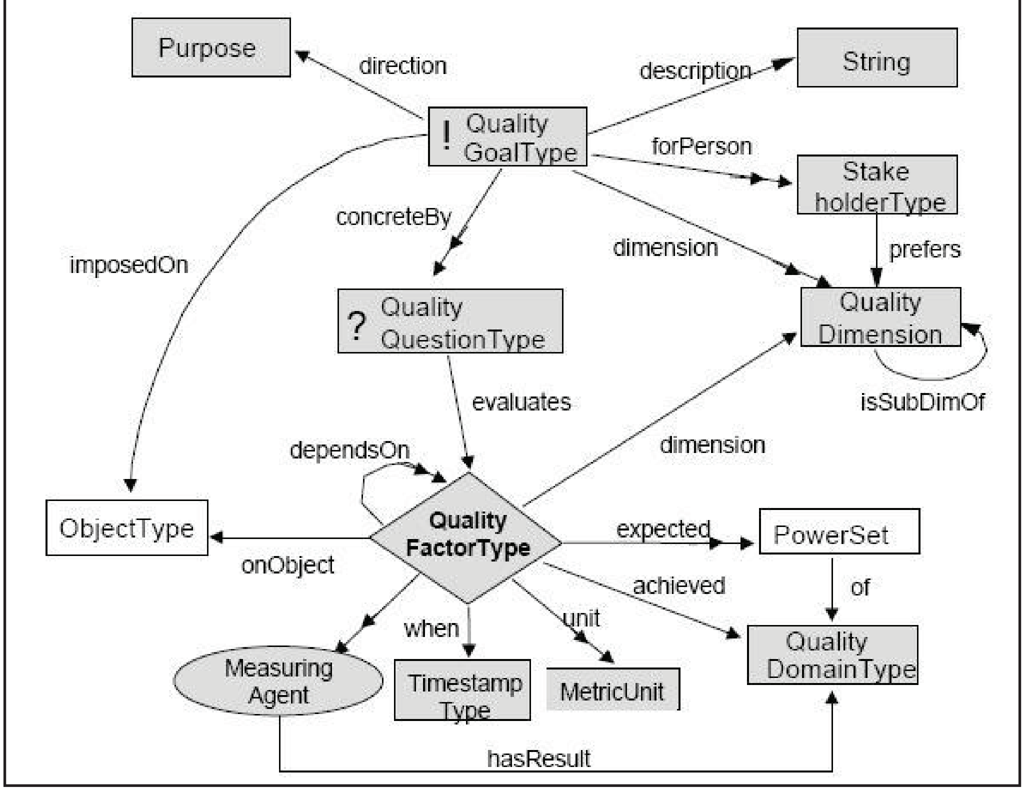
Notes

The main difference in our approach resides in the following points:

1. A clear distinction between subjective quality goals requested by stakeholder and objective quality factors attached to data warehouse objects.
2. Quality goal resolution is based on the evaluation of the composing quality factors, each corresponding to a given quality question.
3. Quality questions are implemented and executed as quality queries on the semantically rich metadata repository.

Figure 12.3 shows the DWQ Quality Model. The class "ObjectType" refers to any meta-object of the DWQframework depicted in the first layer of figure 12.3. A quality goal is an abstract requirement, defined on an object types, and documented by a purpose and the stakeholder interested in. A quality goal roughly expresses natural language requirements like "improve the availability of source s1 until the end of the month in the viewpoint of the DW administrator". Quality dimensions (e.g. "availability") are used to classify quality goals and factors into different categories. Furthermore, quality dimensions are used as a vocabulary to define quality factors and goals; yet each stakeholder might have a different vocabulary and different preferences in the quality dimensions. Moreover, a quality goal is operationally defined by a set of questions to which quality factor values are provided as possible answers. As a result of the goal evaluation process, a set of improvements (e.g. design decisions) can be proposed, in order to achieve the expected quality. A quality factor represents an actual measurement of a quality value, i.e. it relates quality values to measurable objects. A quality factor is a special property or characteristic of the related object with respect to a quality dimension. It also represents the expected range of the quality value, which may be any subset of a quality domain. Dependencies between quality factors are also stored in the repository. Finally, the method of measurement is attached to the quality factor through a measuring agent.

Figure 12.3: DWQ Quality Meta Model



The quality meta-model is not instantiated directly with concrete quality factors and goals, it is instantiated with patterns for quality factors and goals. The use of this intermediate instantiation level enables data warehouse stakeholders to define templates of quality goals and factors. For example, suppose that the analysis phase of a data warehouse project has detected that the availability of the source database is critical to ensure that the daily online transaction processing is not affected by the loading process of the data warehouse. A source administrator might later instantiate this template of a quality goal with the expected availability of his specific source database. Thus, the programmers of the data warehouse loading programs know the time window of the update process.

Based on the meta-model for data warehouse architectures, we have developed a set of quality factor templates which can be used as initial set for data warehouse quality management. The methodology is an adaptation of the Total Quality Management approach and consists of the following steps:

1. Design of object types, quality factors and goals,
2. Evaluation of the quality factors,
3. Analysis of the quality goals and factors and their possible improvements
4. Re-evaluation of a quality goal due to the evolution of data warehouse.

12.3.2 A Quality-oriented Data Warehouse Process Model

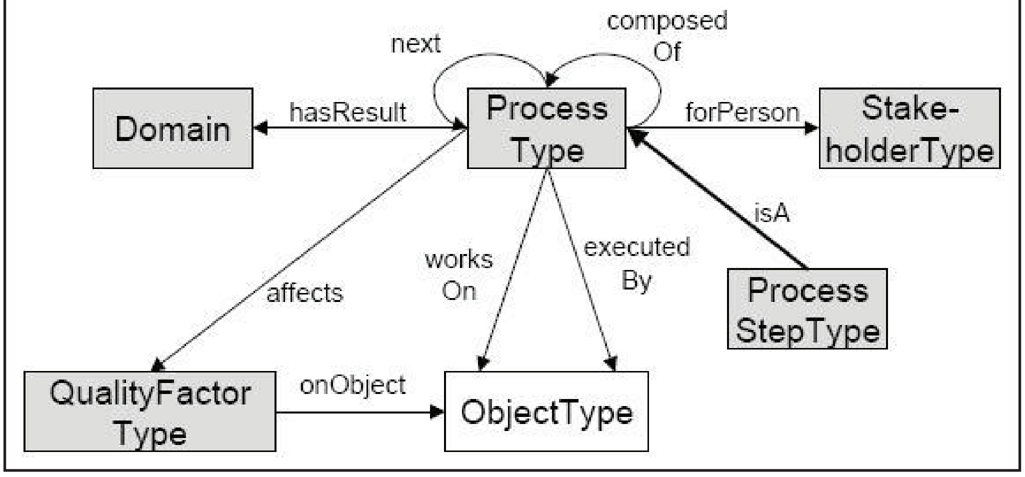
As described in the previous section it is important that all relevant aspects of a data warehouse are represented in the repository. Yet the described architecture and quality model does not represent the workflow which is necessary to build and run a data warehouse, e.g. to integrate data source or to refresh the data warehouse incrementally. Therefore, we have added a data warehouse process model to our meta modeling framework. Our goal is to have a simple process model which captures the most important issues of data warehouses rather than building a huge construction which is difficult to understand and not very useful due to its complexity.

Figure 12.4 shows the meta model for data warehouse processes. A data warehouse process is composed of several processes or process steps which may be further decomposed. Process steps and the processes itself are executed in a specific order which is described by the “next” relation between processes. A process works on an object type, e.g. data loading works on a source data store and a data warehouse data store. The process itself must be executed by some object type, usually an agent which is represented in the physical perspective of the architecture model. The result of a process is some value of a domain, the execution of further processes may depend on this value. For example, the data loading process returns as a result a boolean value representing the completion value of the process, i.e. if it was successful or not. Further process steps like data cleaning are only executed if the previous loading process was successful. The process is linked to a stakeholder which controls or has initiated the process. Moreover, the result of a process is the data which is produced as an outcome of the process, e.g. the tuples of a relation.

Processes affect a quality factor of an object type, e.g. the availability of data source or the accuracy of a data store. It might be useful to store also the expected effect on the quality factor, i.e. if the process improves or decreases the quality factor. However, the achieved effect on the quality factor can only be determined by a new measurement of this factor. A query on the metadata repository can then search for the processes which have improved the quality of a certain object.

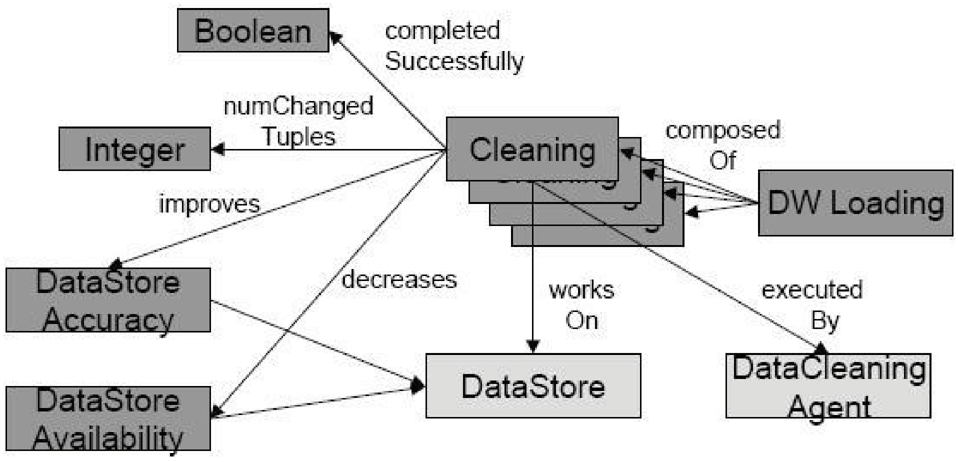
The processes can be subject to quality measurement, too. Yet, the quality of a process is usually determined by the quality of its output. Therefore, we do not go into detail with process quality but quality factors can be attached to processes, too.

Figure 12.4: A Process Model for Data Warehouses



As an example for a data warehouse process we have partially modeled the data warehouse loading process in Figure 12.5. The loading process is composed of several steps, of which one in our example is data cleaning. The data cleaning process step works on a data store, where the data which have to be cleaned reside. It is executed by some data cleaning agent. It affects among others the quality factors accuracy and availability, in the sense that accuracy is hopefully improved and availability is decreased because of locks due to read-write operations on the data store. The data cleaning process may also store some results of its execution in the metadata repository, for example, a boolean value to represent the successful completion of the process and the number of changed tuples in the data store.

Figure 12.5: An Example for a Data Warehouse Process Pattern



The information stored in the repository may be used to find deficiencies in data warehouse. To show the usefulness of this information we use the following query. It returns all data cleaning processes which have decreased the availability of a data store according to the stored measurements. The significance of the query is that it can show that the implementation of data cleaning process has become inefficient.

GenericQueryClass DecreasedAvailability

isA DWCleaningProcess with

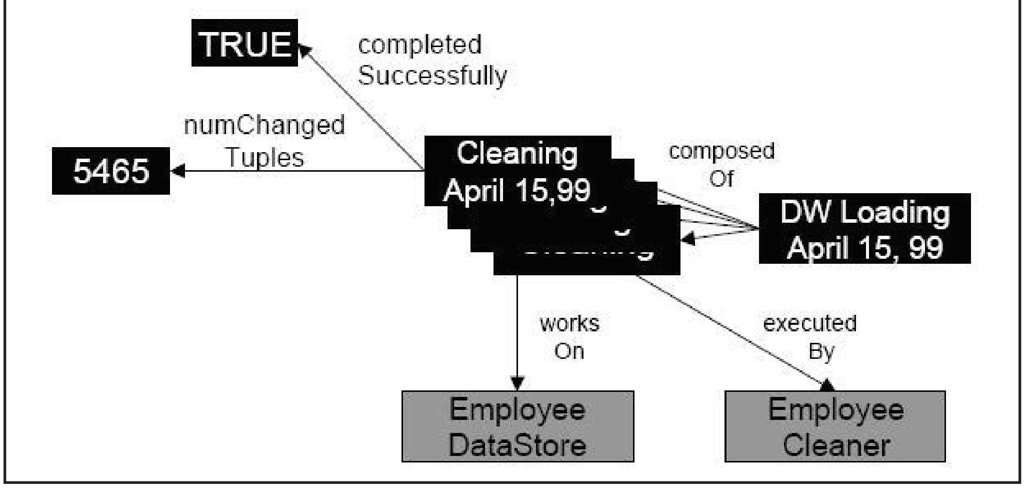
parameter

```
ds : DataStore
constraint c :
$ exists qf1,qf2/DataStoreAvailability
t1,t2,t3/TransactionTime v1,v2/Integer
(qf1 onObject ds) and (qf2 onObject ds) and
(this worksOn ds) and (this executedOn t3) and
(qf1 when t1) and (qf2 when t2) and (t1<t2) and
(t1<t3) and (t3<t2) and (qf1 achieved v1) and
(qf2 achieved v2) and (v1 > v2) $
end
```

The query has a data store as parameter, i.e. the query will return only cleaning processes which are related to the specified data store. The query returns the processes which have worked on the specified data store and which were executed between the measurements of quality factors qf1 and qf2, and the measured value of the newer quality factor is lower than the value of the older quality factor. The query can be formulated in a more generic way to deal with all types of data warehouse processes but for reasons of simplicity and understandability, we have shown this more special variant.

Finally, Figure 12.6 shows the trace of a process at the instance level. The process pattern for DWLoading has been instantiated with a real process, which has been executed on the specified date "April 15, 1999". An instantiation of the links to the quality factors is not necessary, because the information that "data cleaning" affects the accuracy and the availability of a data store is already recorded in the process pattern shown in Figure 12.5.

Figure 12.6: Trace of a Data Warehouse Process



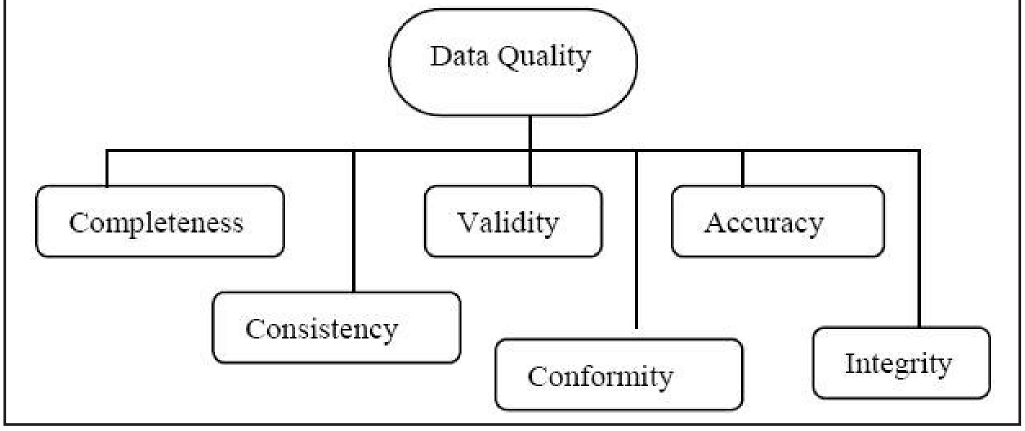
Task Discuss the architecture of data warehouse with a neat diagram and explain each component's functionally in detail.

Notes

12.4 Defining Data Warehouse Quality

The existence of data alone does not ensure that all the management functions and decisions can be smoothly undertaken. The one definition of data quality is that it's about bad data - data that is missing or incorrect or invalid in some context. A broader definition is that data quality is achieved when organization uses data that is timely. Understanding the key data quality dimensions is the first step to data quality improvement. To be process able and interpretable in an effective and efficient manner, data has to satisfy a set of quality criteria. Data satisfying those quality criteria is said to be of high quality. Abundant attempts have been made to define data quality and to identify its dimensions. Dimensions of data quality typically include accuracy, reliability, importance, consistency, precision, timeliness, fineness, understandability, conciseness and usefulness. For our research paper we have under taken the quality criteria by taking 6 key dimensions as depicted below Figure 12.7.

Figure 12.7: Data Quality Criteria

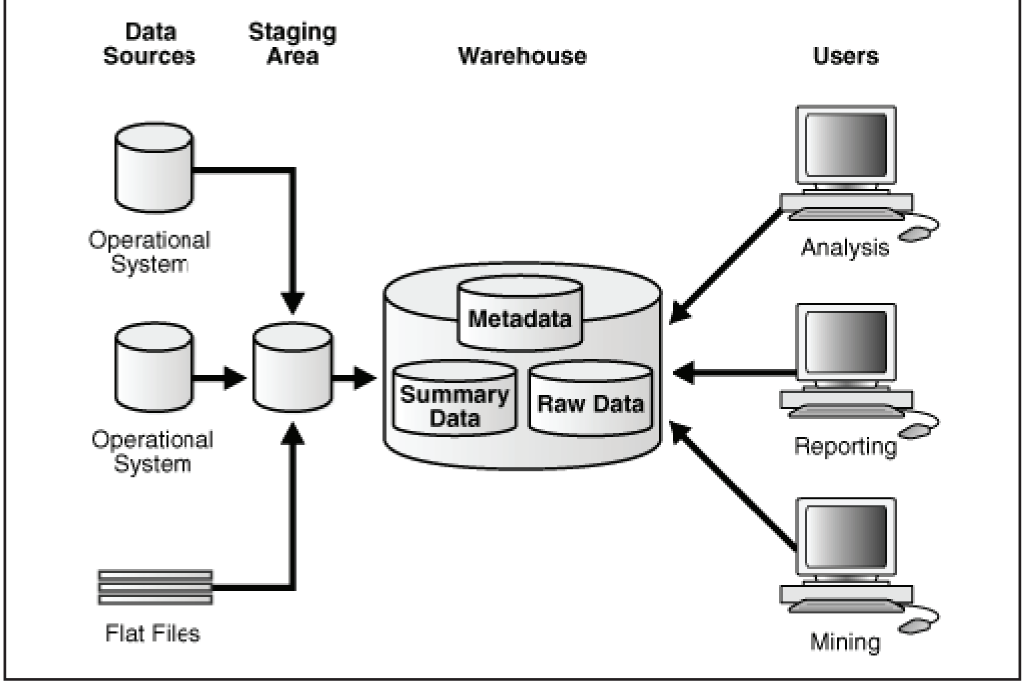


1. ***Completeness:*** Deals with to ensure is all the requisite information available? Are some data values missing, or in an unusable state?
2. ***Consistency:*** Do distinct occurrences of the same data instances agree with each other or provide conflicting information. Are values consistent across data sets?
3. ***Validity:*** refers to the correctness and reasonableness of data.
4. ***Conformity:*** Are there expectations that data values conform to specified formats? If so, do all the values conform to those formats? Maintaining conformance to specific formats is important.
5. ***Accuracy:*** Do data objects accurately represent the "real world" values they are expected to model? Incorrect spellings of product or person names, addresses, and even untimely or not current data can impact operational and analytical applications.
6. ***Integrity:*** What data is missing important relationship linkages? The inability to link related records together may actually introduce duplication across your systems.

Data Warehousing

Data warehouses are one of the foundations of the Decision Support Systems of many IS operations. As defined by the "father of data warehouse", William H. Inmon, a data warehouse is "a collection of Integrated, Subject-Oriented, Non Volatile and Time Variant databases where each unit of data is specific to some period of time. Data Warehouses can contain detailed data, lightly summarized data and highly summarized data, all formatted for analysis and decision support" (Inmon, 1996). In the "Data Warehouse Toolkit", Ralph Kimball gives a more concise definition: "a copy of transaction data specifically structured for query and analysis" (Kimball, 1998). Both definitions stress the data warehouse's analysis focus, and highlight the historical nature of the data found in a data warehouse.

Figure 12.8: Data Warehousing Structure



Stages of Data Warehousing Susceptible to Data Quality Problems

The purpose of paper here is to formulate a descriptive taxonomy of all the issues at all the stages of Data Warehousing. The phases are:

1. Data Source
2. Data Integration and Data Profiling
3. Data Staging and ETL
4. Database Scheme (Modeling)

Quality of data can be compromised depending upon how data is received, entered, integrated, maintained, processed (Extracted, Transformed and Cleansed) and loaded. Data is impacted by numerous processes that bring data into your data environment, most of which affect its quality to some extent. All these phases of data warehousing are responsible for data quality in the data warehouse. Despite all the efforts, there still exists a certain percentage of dirty data. This residual dirty data should be reported, stating the reasons for the failure in data cleansing for the same.

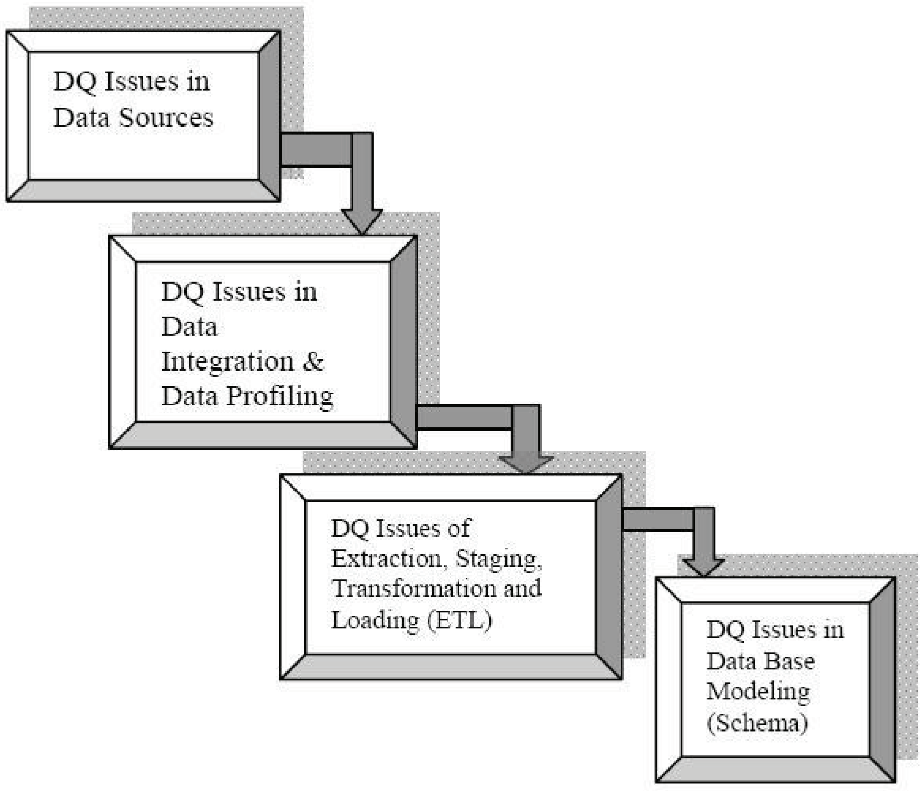
Data quality problems can occur in many different ways. The most common include:

1. Poor data handling procedures and processes.
2. Failure to stick on to data entry and maintenance procedures.
3. Errors in the migration process from one system to another.
4. External and third-party data that may not fit with your company data standards or may otherwise be of unconvinced quality.

Notes

The assumptions undertaken are that data quality issues can arise at any stage of data warehousing viz. in data sources, in data integration & profiling, in data staging, in ETL and database modeling. Following model is depicting the possible stages which are vulnerable of getting data quality problems.

Figure 12.9: Stages of Data Warehouse Susceptible for DQ Problems


DQ Issues in Data Sources
DQ Issues in Data Integration & Data Profiling
DQ Issues of Extraction, Staging, Transformation and Loading (ETL)
DQ Issues in Data Base Modeling (Schema)

Case Study

Trans Union

The Company

Trans Union is a leading information and service provider for direct response marketers. With a national name and address list of over 160 million consumers, Trans Union has one of the most comprehensive data files in the business.

Trans Union understands that organizations need a broad range of information about customers in order to develop one-to-one marketing relationships with prospects. Direct response marketers from a variety of professions rely on the accuracy, quantity and effectiveness of Trans Union products and services.

Competition in the field of consumer information resources is very intense and drives marketing data providers to develop consumer lists faster and with better, deeper and more accurate data. Trans Union recently made a push to reengineer its existing data to find ways of creating new, salient products for its customers.

Contd...

The Challenge

Rapidly processing large quantities of information is a key to Trans Union's success, but that information must also be in an accessible format. "Some of our input was a jumbled mess of data that we couldn't get at," said Trans Union's database construction team leader.

The database team knew it had a wealth of information buried within the comment fields of many records. More specifically, they were looking for information on consumer ownership of large, consumer durables. Most important were boats, recreational vehicles, motor homes and motor vehicles. The challenge, therefore, was to scrutinize the comment field, investigate and parse out the valued data components, and create and populate new fields with data derived from the comment field content.

The targeted quantity of data was large, but not enormous by Trans Union standards: 27 million individual records containing comment fields. Fast turnaround of the data on a Windows NT platform was the goal, because new data is always in demand. Trans Union also didn't want to get into a protracted code-writing exercise that would tax its time and resources. It needed a versatile data-cleansing solution that could be implemented very quickly.

The database team realized that they needed a solution that could scan free-form text, standardize and transform the extracted data, and create new fields that would be populated with intelligence gathered during the cleansing process. The solution would have to be robust enough to handle large volumes of data and simple enough for Trans Union to quickly learn.

Trans Union wanted to develop a standardized set of enterprise business rules for data quality management that could be shared across existing and future platforms. Specifically, the company needed a versatile tool that could reach deep within the complex product data and provide repeatable and reusable business rules for data cleansing.

The Solution

Trans Union chose the Trillium Software System® for its ability to clean and standardize large volumes of generalized data from multiple sources. The Trillium Software System's user-defined transformation and data-filling capabilities, as well as its data element repair facilities, are unique among solutions that operate in multiplatform environments.

It was the Trillium Software System's specific ability to understand, elementize and create a distribution of words and phrases from within floating, free-form text that made it a perfect fit for Trans Union. The company understood that the Trillium Software System could meet the immediate need for data reengineering and fill an expanded role in future projects. Trans Union initially assigned a team of three to the project: a project manager, a programmer and a research analyst. The team's first step was to compile a comprehensive listing of boats, RVs, motor homes and other vehicles made and sold in the previous ten years, and enter them into tables and parameters for data comparisons.

The Results

As a result of its initial data reengineering, Trans Union has created a new suite of products that allows the company to identify owners of specific types of major consumer durables. Trans Union went live with its data-cleansing project within one week of initial training on the system.

"We were able to identify 14 million records—a full 50 percent more than what we had imagined—that had vehicle category information we could append to our customer database," the database construction team leader stated.

Contd...

Notes

In addition, Trans Union was able to identify vehicles by type, classify each vehicle in a standardized, format and create and append new vehicle classifications to the original records.

"Training and learning the Trillium Software System occurred much more easily and quickly than we had imagined. We were able to set up the tables and learn how to tune them within one week, and that allowed us to go live immediately," said the team leader.

Because the implementation was so quick and easy, Trans Union is now considering adding more categories to the mix of products being mined from its consumer database. The company has demonstrated once again that clean data provides a more accurate view of consumers and delivers a more valuable product for its clients.

12.5 Summary

- DWQ provides the framework and modeling formalisms for DW design, development and maintenance tools that can be tuned to particular application domains and levels of quality.
- The potential of such an approach has been demonstrated already by successful commercial usage of the ConceptBase metamodeling tool developed in the COMPULOG project.
- DW development time for a given level of quality will be significantly reduced and adaptation to changing user demands will be facilitated.
- There is a high demand for design tools for distributed databases which is not satisfied by current products.
- DWQ has the potential to satisfy this demand for an increasingly important market segment.
- In addition, a host of quality-enhanced query and update services can be derived from DWQ results.
- Prototypes of specific models and tools developed in the project will be experimented with in various industry and public administration settings, in order to gain reference experiences for industrial uptake of project results.

12.6 Keywords

Data Warehouse Quality: In the DWQ (Data Warehouse Quality) project we have advocated the need for enriched metadata facilities for the exploitation of the knowledge collected in a data warehouse.

Information Quality: Information quality is the key to political success.

Technical Quality: Technical quality is the ability of the data warehouse to satisfy users' dynamic information needs.

The Metadata Hub: The meta data hub is used for managing the interchange and sharing of technical meta data between decision processing products.

12.7 Self Assessment

Choose the appropriate answers:

1. GUI stands for:
   (a) Graphical User Interface
   (b) Graphical User Image
   (c) Graphical User Interchange
   (d) Graphical User Information
2. XML stands for:
   (a) Extension Markup Language
   (b) Extensible Markup Language
   (c) Extensible Makeup Language
   (d) Evaluation Markup Language
3. DWQ stands for:
   (a) Data Warehouse Quantity
   (b) Data object Warehouse Quality
   (c) Data Warehouse Quality
   (d) Database Warehouse Quality

Fill in the blanks:

4. A .... should provide persistent stores for the management and sharing of meta data.
5. .... are values which express some measured property of the object.
6. A quality goal roughly expresses ....
7. A .... represents an actual measurement of a quality value.
8. The process may also store some results of its execution in the metadata repository.
9. The .... is composed of several steps, of which one in our example is data cleaning.
10. A .... on the metadata repository can then search for the processes which have improved the quality of a certain object.

12.8 Review Questions

1. What do you mean by data quality?
2. "Business quality is made up of business drivers, or concepts that point out a company's strategic plans." Discuss.
3. "Information doesn't have value if it's not used." Explain.
4. Describe the use of metadata management in data warehouse.
5. Take a live example and explain metadata hub in detail.
6. Describe quality meta model with the help of suitable example.

7. "The quality meta-model is not instantiated directly with concrete quality factors and goals, it is instantiated with patterns for quality factors and goals." Explain.
8. Explain data warehouse quality in detail.
9. "A quality factor is a special property or characteristic of the related object with respect to a quality dimension." Discuss.
10. Does the data warehouse play a tactical role, so that it makes a positive day-to-day difference? Explain.

Answers: Self Assessment

1.	(a)	2.	(b)
3.	(c)	4.	meta data hub
5.	Quality metrics	6.	natural language requirements
7.	quality factor	8.	data cleaning
9.	loading process	10.	query

12.9 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

. *Online links*

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 13: Metadata and Data Warehouse Quality

CONTENTS

Objectives

After studying this unit, you will be able to:

- Represent and analyse data warehouse quality
- Know quality analysis in data staging

Introduction

Data warehouses are complex systems consisting of many components which store highly aggregated data for decision support. Due to the role of the data warehouses in the daily business work of an enterprise, the requirements for the design and the implementation are dynamic and subjective. Therefore, data warehouse design is a ontinuous process which has to reflect the changing environment of a data warehouse, i.e. the data warehouse must evolve in reaction to the enterprise's evolution. Based on existing meta models for the architecture and quality of a data warehouse, we propose in this paper a data warehouse process model to capture the dynamics of a data warehouse. The evolution of a data warehouse is represented as a special process and the evolution operators are linked to the corresponding architecture components and quality factors they affect. We show the application of our model on schema evolution in data warehouses and its consequences on data warehouse views. The models have been implemented in the metadata repository Concept-Base which can be used to analyze the result of evolution operations and to monitor the quality of a data warehouse.

13.1 Representing and Analyzing Data Warehouse Quality

Data quality (DQ) is an extremely important issue since quality determines the data's usefulness as well as the quality of the decisions based on the data. It has the following dimensions: accuracy, accessibility, relevance, timeliness, and completeness. Data are frequently found to be inaccurate, incomplete, or ambiguous, particularly in large, centralized databases. The economical and social damage from poor-quality data has actually been calculated to have cost organizations billions of dollars, data quality is the cornerstone of effective business intelligence.

Interest in data quality has been known for generations. For example, according to Hasan (2002), treatment of numerical data for quality can be traced to the year 1881. An example of typical data problems, their causes, and possible solutions is provided in Table 13.1.

Table 13.1: Data Problems and Possible Solutions

Problem	Typical Cause	Possible Solutions
Data are not correct	Raw data were entered inaccurately.	Develop a systematic way to ensure the accuracy of raw data. Automate (use scanners or sensors).
	Data derived by an individual were generated carelessly.	Carefully monitor both the data values and the manner in which the data have been generated. Check for compliance with collection rules.
	Data were changed deliberately or accidentally.	Take appropriate security measures
Data are not timely.	The method for generating the data was not rapid enough to meet the need for the data.	Modify the system for generating the data. Move to a client/server system. Automate.
	Raw data were gathered according to a logic or periodicity that was not consistent with the purposes of the analysis.	Develop a system for rescaling or recombining the improperly indexed data. Use intelligent search agents.
Needed data simply do not exit.	Non one ever stored the data needed now	Whether or not it is useful now, store data for future use. Use the Internet to search for similar data. Use experts.
	Required data never existed.	Make an effort to generate the data or to estimate them (use experts). Use neural computing for pattern recognition.

Strong et al., (1997) conducted extensive research on data quality problems. Some of the problems identified are technical ones such as capacity, while others relate to potential computer crimes. The researchers divided these problems into the following four categories and dimensions.

1. ***Intrinsic DQ:*** Accuracy, objectivity, believability, and reputation
2. ***Accessibility DQ:*** Accessibility and access security
3. ***Contextual DQ:*** Relevancy, value added, timeliness, completeness and amount of data.
4. ***Representation DQ:*** Interpretability, ease of understanding, concise representation and consistent representation.

Although business executives recognize the importance of having highquality data, they discover that numerous organizational and technical issues make it difficult to reach this objective. For example, data ownership issues arise from the lack of policies defining responsibility and accountability in managing data. Inconsistent data-quality requirements of various standalone applications create an additional set of problems as organizations try to combine individual applications into integrated enterprise systems. Interorganizational information systems add a new level of complexity to managing data quality. Companies must resolve the issues of administrative authority to ensure that each partner complies with the data-quality standards. The tendency to delegate data-quality responsibilities to the technical teams, as opposed to business users, is another common pitfall that stands in the way of high-quality data.

Different categories of data quality are proposed by Brauer (2001). They are: standardization (for consistency), matching (of data if stored in different places), verification (against the source), and enhancement (adding of data to increase its usefulness). Whichever system is used, once the major variables and relationships in each category are identified, an attempt can be made to find out how to better manage the data.

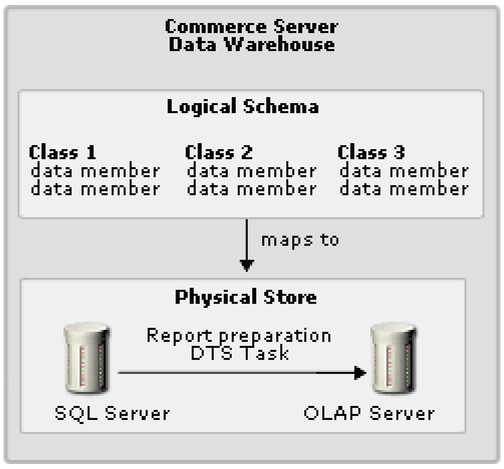
An area of increasing importance is the quality of data that are processed very fast in real time. Many decisions are being made today in such an environment.

13.1.1 Data Warehouse Structure

The Data Warehouse has two main parts:

1. ***Physical store:*** A Microsoft SQL Server database that you can query using SQL queries, and an OLAP database that you can use to run reports.
2. ***Logical schema:*** A conceptual model that maps to the data in the physical store.

In figure below shows the relationship between the physical store and the logical schema.



Physical Store

The physical store for the Data Warehouse includes one database that you can query using SQL queries. The physical store contains all the data that you have imported from different sources.

Commerce Server automatically builds the physical store for the Data Warehouse in both the SQL Server database and in the OLAP database. The Data Warehouse provides the data necessary for all the Commerce Server reports available in the Analysis modules in Business Desk.

There is no need for you to directly modify the physical store for the Data Warehouse. If you need to extend the Data Warehouse, for example, to encompass third-party data, a site developer can programmatically add the fields you need through the logical schema.

Logical Schema

The logical schema provides an understandable view of the data in the Data Warehouse, and supports an efficient import process. For example, a site developer uses the logical schema to modify the location of data stored in the underlying physical tables. When a site developer writes code to add, update, or delete data in the Data Warehouse, the developer interacts with the logical schema. When Commerce Server accesses data in the Data Warehouse, it accesses the data through the logical schema. Only the site developer needs detailed knowledge of the logical schema.

A logical schema includes the following:

1. ***Class***: A logical collection of data members. For example, the RegisteredUser class contains data members describing a registered user.
2. ***Data member***: A structure that stores a piece of data. For example, the E-mail data member of the RegisteredUser class stores the e-mail address for a registered user.
3. ***Relation***: A connection between two classes in a parent-child relationship. This relationship defines the number of instances of each class, and it provides the mechanism for sharing data members between classes. For example, RegisteredUser is a parent to the child class Request. There can be many requests for one registered user.

The logical schema uses classes, data members, relations, and other data structures to map data in the physical store.

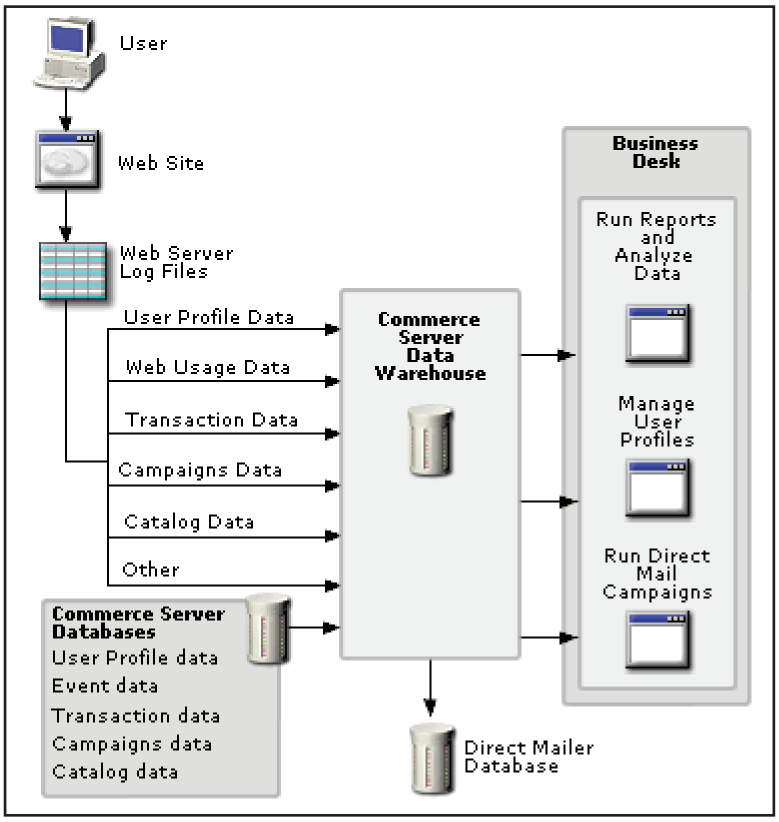
13.1.2 Importing Data to the Data Warehouse

The data that populates the Data Warehouse typically comes from multiple data sources: Web server logs, Commerce Server databases, and other data sources that you specify. The following figure shows the sources for operational data, and how the data might be used to support tasks run from Business Desk.

Because the Data Warehouse is not part of your run-time environment, a system administrator must determine how frequently to import the operational data into the Data Warehouse. For example, you can set up the Data Warehouse so that it automatically imports new data every day or every week. The frequency with which you will need to import data depends on the amount of new data collected every day in your operational data sources. Commerce Server includes custom Data Transformation Service (DTS) tasks that simplify the importing of data into the Data Warehouse. These DTS tasks import data that is used with the reports available from Business Desk.

Even though the operational data can be imported from different types of databases or from storage media that are not databases—all of the data is structured in a consistent manner after it is gathered into the Data Warehouse. For example, you might have one data source in which the first and last name of a user are stored in the same field, and another in which the first and last names are stored in separate fields. When this data is imported into the Data Warehouse, it is automatically structured to be consistent, thus enabling your analysis activities.

Notes



13.1.3 Preparing Data for Analysis with OLAP Server

After data is imported into the Data Warehouse SQL Server database, it must be prepared for analysis so business managers can run reports against it. To prepare data for reporting, the system administrator runs a DTS task that exports a selected subset of data from the SQL Server database to the OLAP database. In the OLAP database, the data is stored in multidimensional cubes.

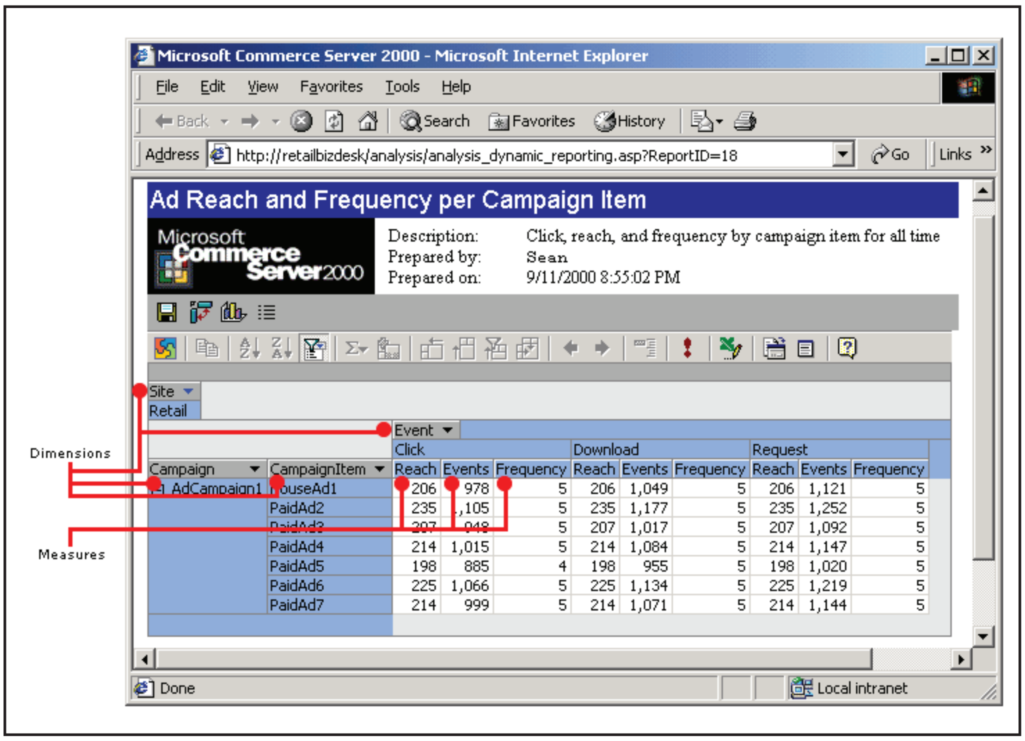
By storing data in OLAP cubes, instead of in relational tables in SQL Server, the Data Warehouse can retrieve data for reporting purposes more quickly. The data can be retrieved from the cubes faster because it is aggregated. That is, data that belongs together is already associated so it is easier to retrieve than searching an entire relational database for the smaller parts. For example, using OLAP server you can run a report that lists users who visit your site based on the time of their visit and on the ASP page that they access first. It would be extremely difficult to run such a report against a large SQL Server database.

In multidimensional cubes, data is grouped in two kinds of structures:

1. ***Measures***: The numeric values that are analyzed.
2. ***Dimensions***: A business entity, such as color, size, product, or time. For example, you would use the color dimension to contrast how many red products and blue products were sold, the size dimension to contrast how many large and small products were sold.

It is the relationship between the dimension (for example, color) and measure (for example, number of products sold) structures that provides the basis for your reports about user activity.

In figure below illustrates the dimensions and measures in a report.


Microsoft Commerce Server 2000 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://retailbizdesk/analysis/analysis_dynamic_reporting.asp?ReportID=18
Go Links
Dimensions
Measures
Done
Local intranet

Ad Reach and Frequency per Campaign Item

Description: Click, reach, and frequency by campaign item for all time
Prepared by: Sean
Prepared on: 9/11/2000 8:55:02 PM

Site: Retail

		Event								
		Click			Download			Request		
Campaign	CampaignItem	Reach	Events	Frequency	Reach	Events	Frequency	Reach	Events	Frequency
AdCampaign1	HouseAd1	206	978	5	206	1,049	5	206	1,121	5
	PaidAd2	235	1,105	5	235	1,177	5	235	1,252	5
	PaidAd3	207	948	5	207	1,017	5	207	1,092	5
	PaidAd4	214	1,015	5	214	1,084	5	214	1,147	5
	PaidAd5	198	885	4	198	955	5	198	1,020	5
	PaidAd6	225	1,066	5	225	1,134	5	225	1,219	5
	PaidAd7	214	999	5	214	1,071	5	214	1,144	5

Task Discuss the various factors behind the measures of data warehouse quality. Why its so important?

13.1.4 Analyzing your Data

To analyze data about user activity on your site, you use the Analysis modules in Business Desk. You can use the Analysis modules to run reports against the Data Warehouse, or to view and analyze Segment models, which identify segments of the user population visiting your site.

Reports

Commerce Server provides two types of reports that you can use to analyze user data:

1. ***Dynamic reports are created at runtime:*** Each time the report is run, it gathers the most recent data in the Data Warehouse. Only the report definition, which remains the same over time, is stored. Commerce Server does not save completed dynamic reports; however, you can export dynamic report results to Microsoft® Excel and store them there. For information about exporting the results of a dynamic report to Microsoft Excel.

2. ***Static reports*** are run immediately upon request, and then stored, with the data, in Completed Reports: The reports appear in a browser window in HTML format. You can export static report results to the List Manager module, and then use the list in a direct mail campaign. You can send these reports to others using e-mail, post them on your Web site, and edit them in other applications. For example, using Microsoft Internet Explorer, you can export the report into a Word document, and then edit it.

Notes

13.2 Quality Analysis in Data Staging

In the data warehousing process, the data staging area is composed of the data staging server application and the data store archive (repository) of the results of extraction, transformation and loading activity. The data staging application server temporarily stores and transforms data extracted from OLTP data sources and the archival repository stores cleaned, transformed records and attributes for later loading into data marts and data warehouses.

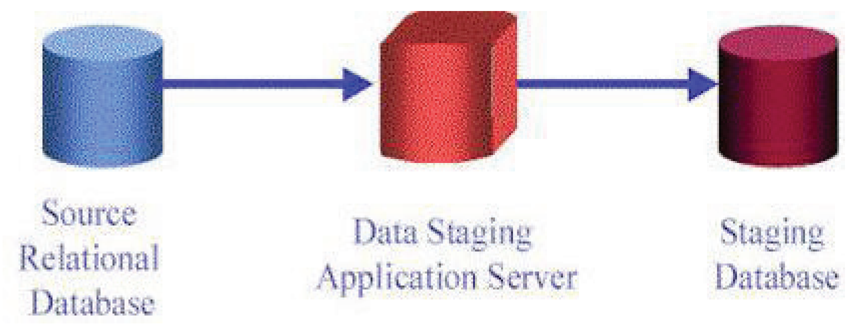
13.2.1 The Data Staging Process

The data staging process imports data either as streams or files, transforms it, produces integrated, cleaned data and stages it for loading into data warehouses, data marts, or Operational Data Stores.

Kimball et.al. distinguish two data staging scenarios:

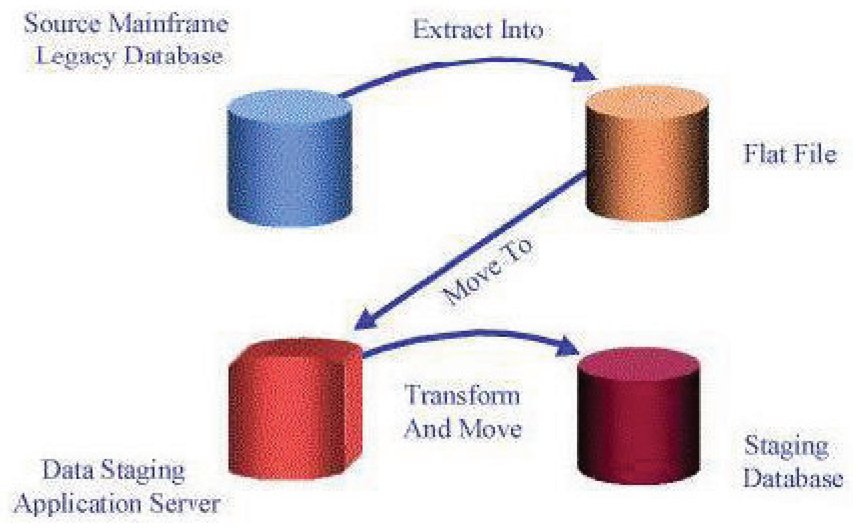
A data staging tool is available, and the data is already in a database. The data flow is set up so that it comes out of the source system, moves through the transformation engine, and into a staging database. The flow is illustrated in Figure 13.1.

Figure 13.1: First Data Staging Scenario



In the second scenario, begin with a mainframe legacy system. Then extract the sought after data into a flat file, move the file to a staging server, transform its contents, and load transformed data into the staging database. Figure 13.2 illustrates this scenario.

Figure 13.2: Second Data Staging Scenario


Source Mainframe Legacy Database
Extract Into
Flat File
Move To
Transform And Move
Data Staging Application Server
Staging Database

The Data Warehouse Staging Area is temporary location where data from source systems is copied. A staging area is mainly required in a Data Warehousing Architecture for timing reasons. In short, all required data must be available before data can be integrated into the Data Warehouse.

Due to varying business cycles, data processing cycles, hardware and network resource limitations and geographical factors, it is not feasible to extract all the data from all Operational databases at exactly the same time.

Example: It might be reasonable to extract sales data on a daily basis, however, daily extracts might not be suitable for financial data that requires a month-end reconciliation process. Similarly, it might be feasible to extract "customer" data from a database in Singapore at noon eastern standard time, but this would not be feasible for "customer" data in a Chicago database.

Data in the Data Warehouse can be either persistent (i.e. remains around for a long period) or transient (i.e. only remains around temporarily).

Not all business require a Data Warehouse Staging Area. For many businesses it is feasible to use ETL to copy data directly from operational databases into the Data Warehouse.

Task Discuss metadata and its importance of source system and Data Staging area.

13.2.2 Pros and Cons of Data Staging

The pros and cons of data staging are:

Pros

1. The staging process is handled in parallel with the transformation process.
2. The disk I/O is reduced in half because the staging table is only written to, as opposed to written to and then extracted from again.

Cons

If the transformation process fails, then the staging process will also stop.

Case Study <u>AT&T</u>

The Company

AT&T, a premier voice, video, and data communications company, was gearing up for the next phase of telecommunications industry deregulation. It recognized that the path to growth was paved with synergistic products and services, such as credit cards, e-commerce offerings and integrated customer service. Moreover, it would need to offer a single support number for all product and service inquiries.

These realizations drove AT&T to regard customers in a new light: not as individual accounts, but as people who buy a family of products or services. To capitalize on the opportunities this presented, AT&T needed to centralize its customer data and consistently identify customers across touch points and accounts. The Integrated Customer View project, with its customer data warehouse, answered these needs.

Contd...

Creating a unified customer view within the warehouse demanded highquality, consistent data. The stakes were high: the potential telecommunications market exceeding 100 million customers meant that even 99 accuracy in customer data would still result in more than a million faulty records.

The Challenge

Data quality wasn't a new idea at the company. Until recently, there wasn't a convenient way for AT&T to reconcile different versions of the same consumer from one product or department to the next. Although the problem was hardly unique to the carrier, the ramifications of duplication and inaccuracy were significant given the size of its marketplace.

Integrating these systems to present a unified customer view was far from straightforward. In the telecommunications industry, the problem is even more challenging. Unlike other businesses, such as consumer banks, AT&T might serve the same customer at multiple addresses and with multiple phone numbers.

Another problem was data volatility: anyone could become a customer at any time. Moreover, if the customer moved to another locale, the source and the content of the data for that particular customer could change, too.

As for data sources, there are more than 1,500 local phone service providers across the U.S. The content, format, and quality of name and address data can vary sharply between one local exchange provider and the next.

The manager of the AT&T Integrated Customer View project explained, "We needed a data cleansing system that could reach for the 'knowledge' in our address base, and perform parsing and matching according to a standardized process."

The Solution

The AT&T consumer division's Integrated Customer View project included a team with roughly a dozen core members, each of whom had years of experience in customer identification and systems development. That core group was supplemented by a larger group of business analysts from across the enterprise.

The team ruled out custom development because of stiff maintenance requirements and rejected out-of-the-box direct mail software packages because they weren't precise enough. Ultimately, the team chose the Trillium Software System® for data identification, standardization, and postal correction. Trillium Software's solution was the only package that could deliver the necessary functionality in both UNIX and mainframe environments.

Multiplatform support was critical because, although the company was committed to client/server migration, legacy systems would likely coexist through the foreseeable future. Initially, the system would consist of two parts: a terabyte-sized master repository residing on an IBM mainframe, and an Oracle-based data warehouse maintained on AT&T UNIX server.

The AT&T project team spent nine months creating and testing data cleansing rules for an initial loading of data into the data warehouse. Because of its unique data requirements, the team developed an automated name and address matching process that resulted in a large number of permutations. According to the AT&T project manager, "The problem was so complex that it was beyond the ability of a single individual to handle."

Contd...

The Results

Preparation was highly efficient. The Trillium Software System made it easy to modularize pattern-matching logic. Its patternmatching techniques gave us the ability to quickly identify that one scenario out of thousands of possible combinations that applied to an individual.

With all preparation completed, the team used the Trillium Software System's batch processing facility to cleanse and load data. The process was completed in half the estimated time due to the adequate preparation and match logic debugging.

AT&T relies on the cleansed data in every customer-based business process and system. As a result, the company knows its customers better and consequently provides better service. The Trillium Software System gives the company a cleansing platform that makes it easier to refine the data cleansing process for optimal results.

13.3 Summary

- We have extended our meta modeling framework for data warehouse architecture and quality by a model for data warehouse processes and have specialized this model to the case of data warehouse evolution. In detail, we have addressed the problem of evolution of data warehouse views.
- The management of the metadata in our repository system.
- ConceptBase allows us to query and analyze the stored metadata for errors and deficiencies.
- In addition, features like client notification and active rules of ConceptBase support the maintenance of the data warehouse components and keep data warehouse users up-to-date on the status of the data warehouse.

13.4 Keywords

Data Quality: Data quality (DQ) is an extremely important issue since quality determines the data's usefulness as well as the quality of the decisions based on the data.

Data Warehouse Staging Area: The Data Warehouse Staging Area is temporary location where data from source systems is copied.

Logical Schema: The logical schema provides an understandable view of the data in the Data Warehouse, and supports an efficient import process.

Physical Store: The physical store for the Data Warehouse includes one database that you can query using SQL queries.

13.5 Self Assessment

Choose the appropriate answers:

1. HTML stands for:
    (a) Hypertext Makeup Language
    (b) Hypertext Markup Language
    (c) Heterogeneous Markup Language
    (d) Homogenous Markup Language

2. ASP stands for:
   (a) Active Server Page
   (b) Active Service Page
   (c) Active Server Paging
   (d) Action Server Page
3. DTS stands for:
   (a) Data Transformation Server
   (b) Database Transformation Service
   (c) Dialogue Transformation Service
   (d) Data Transformation Service

Fill in the blanks:

4. .... is a logical collection of data members.
5. There is no need for you to directly modify the .... for the Data Warehouse.
6. The physical store contains all the data that you have imported from ....
7. The .... uses classes, data members, relations, and other data structures to map data in the physical store.
8. .... includes custom Data Transformation Service (DTS) tasks that simplify the importing of data into the Data Warehouse.
9. A .... is mainly required in a Data Warehousing Architecture for timing reasons.
10. Data in the Data Warehouse can be either persistent or ....

13.6 Review Questions

1. What are the pros and cons of data staging?
2. Describe data preparation for analysis with OLAP server.
3. How will you analyze the data? Discuss.
4. Describe data staging process in detail.
5. What do you mean by data transmission service?
6. Explain a connection between two classes in a parent-child relationship.
7. “Commerce Server automatically builds the physical store for the Data Warehouse in both the SQL Server database and in the OLAP database.” Discuss.
8. What do you mean by logical schemas?
9. “A staging area is mainly required in a Data Warehousing Architecture for timing reasons.” Explain.
10. What is the need of the representation of data warehouse?

Answers: Self Assessment

Notes

1. (b)
2. (a)
3. (d)
4. Class
5. physical store
6. different sources
7. logical schema
8. Commerce Server
9. staging area
10. transient

13.7 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

Notes

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com

Notes

Unit 14: Quality Driven Data Warehouse Design

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe quality driven data warehouse design
- Know interaction between quality factors and data warehouse tasks
- Explain DWQ data warehouse design methodology
- Describe optimizing the materialization of DW views

Introduction

A data warehouse (DW) can be seen as a set of materialized views defined over remote base relations. When a query is posed, it is evaluated locally, using the materialized views, without accessing the original information sources. The DWs are dynamic entities that evolve continuously over time. As time passes, new queries need to be answered by them. Some of these queries can be answered using exclusively the materialized views. In general though new views need to be added to the DW.

Notes

14.1 Quality Driven Data Warehouse Design

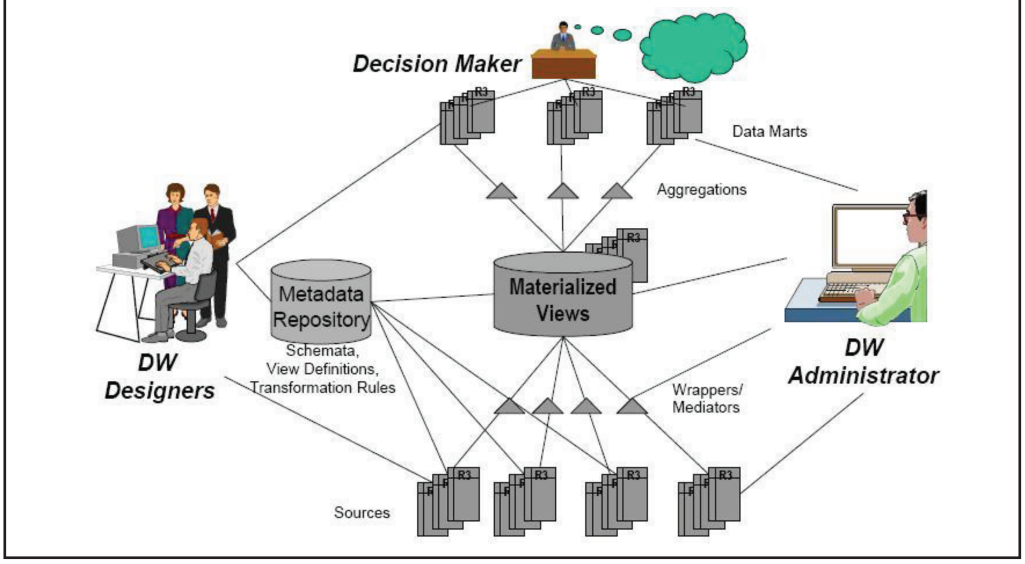
After the basic online transaction processing (OLTP) infrastructure is in place in many organizations, not least through standardized enterprise resource planning tools such as SAP/R3, the focus of interest is now broadening in at least three directions:

1. A broader range of multimedia data sources inside and outside the organization,
2. A broader range of clients with diverse interest and capability profiles as well as situational parameters, and
3. The conversion of the massive experiential data created by transaction processing into knowledge relevant for organizational learning and action.

A wide range of information flow logistics architectures is being proposed under labels such as supply chain management and business-to-business e-commerce. In such architectures, databases can be considered the short- and medium-term intermediate stores of information whereas data warehouses serve for long-term memory, knowledge creation and management. One could also say that a data warehouse is a long-term information broker between the operational part and the reporting/planning part of a company, supporting the Controlling department.

The traditional data warehouse (DW) architecture is shown in Figure 14.1. Physically, a data warehouse system consists of databases (source databases, materialized views in the data warehouse), data transport agents that ship data from one database to another, and a repository which stores metadata about the system and its evolution. In this architecture heterogeneous information sources are first made accessible in a uniform way through extraction mechanisms called wrappers, then mediators take on the task of information integration and conflict resolution. The separation between wrappers and mediators is a considered design decision, reflecting the separation between service wrappers and request brokers in middleware systems such as CORBA.

Figure 14.1: Traditional Data Warehouse Architecture

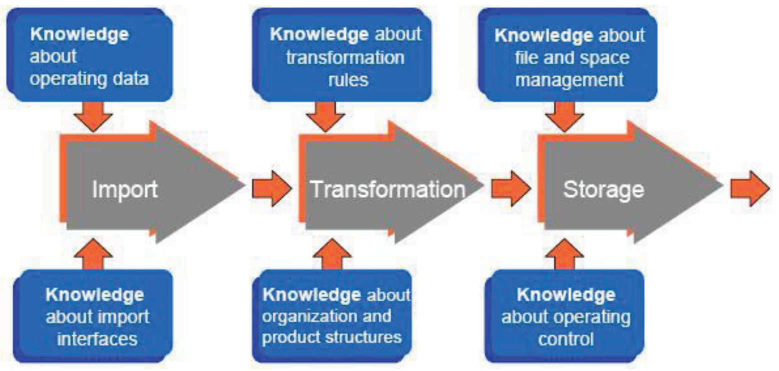


The resulting standardized and integrated data are stored as materialized views in the data warehouse. These base views (often called ODS or operational data store) are usually just slightly aggregated. To customize them for different analyst users, data marts with more aggregated data about specific domains of interest are constructed as second-level caches which are then accessed by data analysis tools ranging from query facilities through spreadsheet tools to full-fledged data mining systems.

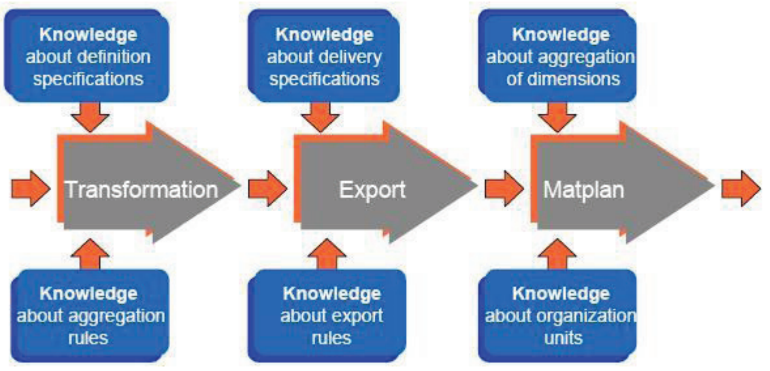
Almost all current research and practice understand a data warehouse architecture as a stepwise information flow from information sources through materialized views towards analyst clients, as shown in Figure 14.1.

The key argument pursued in this paper is that the architecture in Figure 14.1 covers only partially the tasks faced in data warehousing and is therefore unable to even express, let alone support, a large number of important quality problems and management strategies followed in data warehouses. To illustrate this point, consider Figure 14.2 which shows the host of knowledge sources used in the central data warehouse a large German bank has developed for financial controlling.

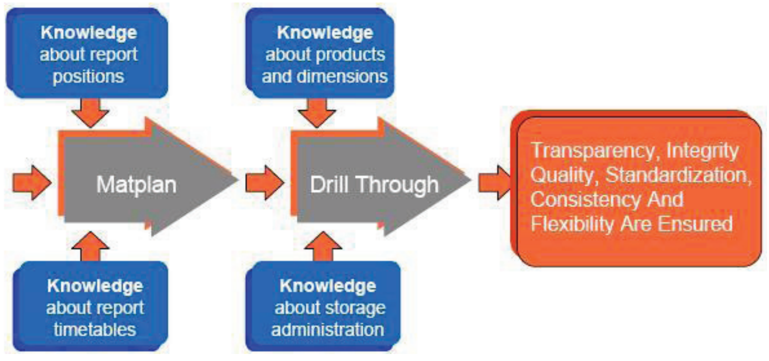
Figure 14.2: Multiple Knowledge Sources Influencing the Design of a Real-World Data Warehouse


Knowledge about operating data
Knowledge about transformation rules
Knowledge about file and space management
Import
Transformation
Storage
Knowledge about import interfaces
Knowledge about organization and product structures
Knowledge about operating control

(a) Loading and refreshing the ODS


Knowledge about definition specifications
Knowledge about delivery specifications
Knowledge about aggregation of dimensions
Transformation
Export
Matplan
Knowledge about aggregation rules
Knowledge about export rules
Knowledge about organization units

(b) Preparing a host of specialized multi-dimensional models


Knowledge about report positions
Knowledge about products and dimensions
Matplan
Drill Through
Transparency, Integrity Quality, Standardization, Consistency And Flexibility Are Ensured
Knowledge about report timetables
Knowledge about storage administration

(c) Personalizing information delivery and access to background information

Despite the fact, that this data warehouse talks "only" about financial figures, there is a host of semantic coherency questions to be solved between the different accounting definitions required by tax laws, stock exchanges, different financial products, and the like. At the same time, there are massive physical data integration problems to be solved by re-calculating ten thousands of multi-dimensional data cubes on a daily basis to have close to zero-latency information for top management. In light of such problems, many architectures discussed in the literature appear somewhat naive.

The key to solving these enormous problems in a flexible and evolvable manner is enriched metadata management, used by different kinds of interacting software components. In the following section, we shall present our approach how to organize this.

14.2 Interaction between Quality Factors and DW Tasks

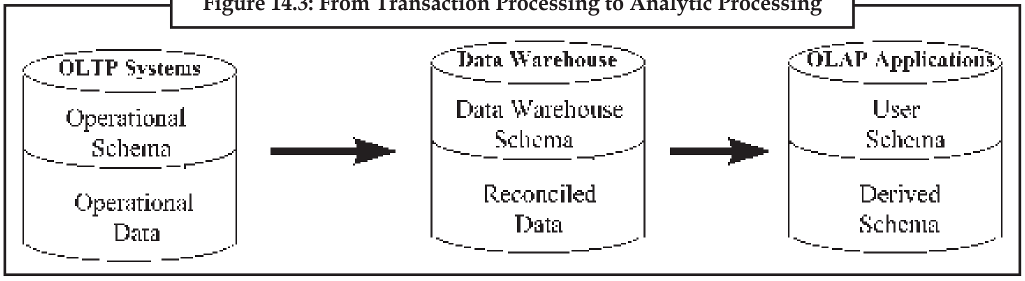
Starting from a definition of the basic DW architecture and the relevant data quality issues, the first project goal is to define the range of design and operational method alternatives for each of the main architecture components and quality factors. Since usually a combination of enabling technologies is required, innovations are envisioned both at the design (e.g., rich meta-data representation and reasoning facilities) as well as at the operational level (e.g., viewing DW contents as views over the underlying information sources, refreshment techniques and optimal handling of views with aggregate functions become important). In a second step, formal models of the DW architecture and services will be developed together with associated tools for consistency checking in the richer model, reuse by subsumption, view materialization strategies, and other components of the data warehousing software. These models and tools will make the knowledge about operational alternatives and their configuration available to the data warehouse designer, in order to allow the dynamic adaptation of the data warehouse structure and quality-of-service to the ever-changing information sources and analysis patterns.

The increased accessibility of information over wide-area networks does not solve the problem to have the right information in the right place at the right time with the right cost.

Data warehousing has become an important strategy to integrate heterogeneous information sources in organizations, and to enable on-line analytic processing. Their development is a consequence of the observation by W. Inmon and E. F. Codd in the early 1990's that operational-level on-line transaction processing (OLTP) and decision support applications (on-line analytic processing or OLAP) cannot co-exist efficiently in the same database environment, mostly due to their very different transaction characteristics.

A DW caches selected data of interest to a customer group, so that access becomes faster, cheaper and more effective. As the long-term buffer between OLTP and OLAP (Figure 14.3), DW's face two essential questions: how to reconcile the stream of incoming data from multiple heterogeneous legacy sources, and how to customize derived data storage to specific OLAP applications. The trade-offs driving the design decisions concerning these two issues change continuously with business needs, therefore design support and change management are of greatest importance if we do not want to run DW projects into dead ends. This is a recognized problem in industry which is not solvable without improved formal foundations.

Figure 14.3: From Transaction Processing to Analytic Processing



Vendors agree that data warehouses cannot be off-the-shelf products but must be designed and optimized with great attention to the customer situation. Traditional database design techniques do not apply since they cannot deal with DW-specific issues such as data source selection, temporal and aggregated data, and controlled redundancy management. Since the wide variety of product and vendor strategies prevents a low-level solution to these design problems at acceptable costs, only an enrichment of metadata services linking heterogeneous implementations is a promising solution. But this requires research in the foundations of data warehouse quality.

The goal of the DWQ project is to develop a semantic foundation that will allow the designers of data warehouses to link the choice of deeper models, richer data structures and rigorous implementation techniques to quality-of-service factors in a systematic manner, thus improving the design, the operation, and most importantly the evolution of data warehouse applications.

DWQ's research objectives address three critical domains where quality factors are of central importance:

1. Enrich the semantics of meta databases with formal models of information quality to enable adaptive and quantitative design optimization of data warehouses;
2. Enrich the semantics of information resource models to enable more incremental change propagation and conflict resolution;
3. Enrich the semantics of data warehouse schema models to enable designers and query optimizers to take explicit advantage of the temporal, spatial and aggregate nature of DW data.

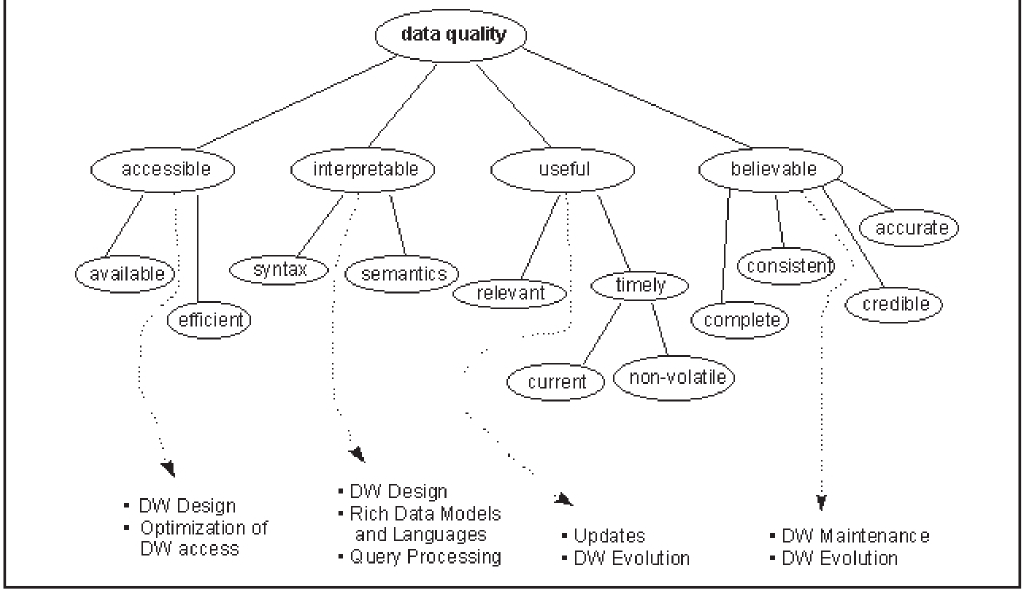
The results will be delivered in the form of publications and supported by a suite of protoptype modules to achieve the following practical objectives:

1. Validating their individual usefulness by linking them with related methods and tools of Software AG, a leading European vendor of DW solutions. The research goal is to demonstrate progress over commercial state-of-the-art, and to give members of the industrial steering committee a competitive advantage by early access to results
2. Demonstrating the interaction of the different contributions in the context of case studies in telecommunications and and environmental protection.

Linking Data Warehousing and Data Quality. DWQ provides assistance to DW designers by linking the main components of a DW reference architecture to a formal model of data quality, as shown in Figure 14.4.

Notes

Figure 14.4: Data Quality Issues and DWQ Research Results



A closer examination of the quality factor hierarchy reveals several relationships between quality parameters and design/operational aspects of DW's. The DWQ project will investigate these relationships in a systematic manner:

1. The simple DW concept itself alleviates the problem of accessibility, by saving its users the effort of searching in a large, poorly structured information space, and avoiding interference of data analysis with operational data processing. However, the issue of delivering the information efficiently, is an important open problem in the light of its differences from traditional query processing. In a DW environment, there is an increased need for fast and dynamic aggregate query processing, indexing of aggregate results, as well as fast update of the DW content after changes are performed to the underlying information sources.

2. It remains difficult for DW customers to interpret the data because the semantics of data description languages for data warehouse schemata is weak, does not take into account domain-specific aspects, and is usually not formally defined and therefore hardly computer-supported. The DWQ project will ensure interpretability by investigating the syntax, semantics, and reasoning efficiency for rich schema languages which (a) give more structure to schemas, and (b) allow the integration of concrete domains (e.g., numerical reasoning, temporal and spatial domains) and aggregate data. This work builds in part on results obtained in the CLN (Computational Logic) II ESPRIT Project.

3. The usefulness of data is hampered because it is hard to adapt the contents of the DW to changing customer needs, and to offer a range of different policies for ensuring adequate timeliness of data at acceptable costs. The DWQ project will develop policies to extend active database concepts such that data caching is optimized for a given transaction load on the DW, and that distributed execution of triggers with user-defined cache refreshment policies becomes possible. This work builds on earlier active database research, e.g. in ACTNET and the STRETCH and IDEA ESPRIT projects.

4. The believability of data is hampered because the DW customer often does not know the credibility of the source and the accuracy of the data. Moreover, schema languages are too weak to ensure completeness and consistency testing. To ensure the quality of individual DW contents, the DWQ project will link rich schema languages to techniques for efficient integrity checking for relational, deductive, and object-oriented databases. Moreover,

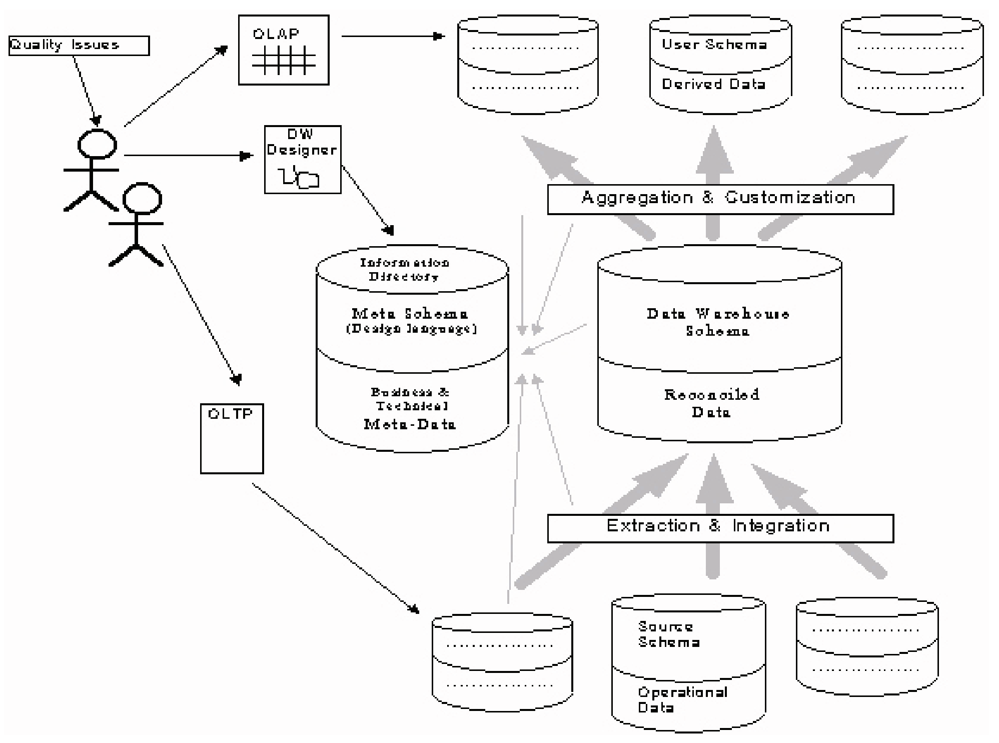
recent techniques from meta modeling and distributed software engineering will help to identify and maintain inter-view relationships. This requires deep integration of AI and database techniques, building on experiences in view integration and maintenance, and meta-level integration of heterogeneous databases.

Task Discuss the differences between no coupling, loose coupling, semi tight coupling and tight coupling architectures for the integration of a data mining system with a Database or data warehouse system.

14.2.1 Expected Results and Innovations

The DWQ project will produce semantic foundations for DW design and evolution linked explicitly to formal quality models, as indicated in the middle of Figure 14.5. These semantic foundations will be made accessible by embedding them in methodological advice and prototype tools. Their usefulness will be validated in the context of Software AG's methodology and tool suite and a number of sample applications.

Figure 14.5: The DWQ Framework



After developing an initial reference model jointly with the industrial committee, the results will be delivered in two stages to enable effective project control and coherence. The first group of results will develop enriched formal meta models for describing the *static architecture* of a DW and demonstrate how these enriched foundations are used in *DW operation*. The corresponding tools include architecture modeling facilities including features for addressing DW-specific issues such as resolution of multiple sources and management of partially aggregated multi-dimensional data, as well as semantics-based methods for query optimization and incremental update propagation.

The second group of results focuses on enhancing these enriched models by tools that support the evolution and optimization of DW applications under changing quality goals. The corresponding tools include: evolution operators which document the link between design decisions and quality factors, reasoning methods which analyze and optimize view definitions with multi-dimensional aggregated data, and allow efficient quality control in bulk data reconciliation from new sources; and quantitative techniques which optimize data source selection, integration strategies, and redundant view materialization with respect to given quality criteria, esp. performance criteria.

14.2.2 Quality Factors and Properties

To carry out data evaluation we firstly need to identify which quality factors to evaluate. The choice of the most appropriate quality factors for a given DIS depends on the user applications and the way the DIS is implemented. Several works study the quality factors that are more relevant for different types of systems. The selection of the appropriate quality factors implies the selection of metrics and the implementation of evaluation algorithms that measure, estimate or bound such quality factors.

In order to calculate quality values corresponding to those factors, the algorithms need input information describing system properties such as, for example, the time an activity needs to execute or a descriptor stating if an activity materializes data or not. These properties can be of two types: (i) descriptions, indicating some feature of the system (costs, delays, policies, strategies, constraints, etc.), or (ii) measures, indicating a quality value corresponding to a quality factor, which can be an actual value acquired from a source, a calculated value obtained executing an evaluation algorithm or an expected value indicating the user desired value for the quality factor. The selection of the adequate properties depends on the quality factors that are relevant for the system and on the calculation processes.

Example: Consider a system where users are interested in the evaluation of response time and freshness. To calculate the response time, it is necessary to know which activities materialize data and the execution cost of the activities that do not materialize data. To calculate the data freshness it is also necessary to know the refreshment frequencies and costs as well as the actual freshness of the data in the sources. Other examples of properties can include execution policies, source constraints and communication delays.

14.3 The DWQ Data Warehouse Design Methodology

Data warehouses support business decisions by collecting, consolidating, and organizing data for reporting and analysis with tools such as online analytical processing (OLAP) and data mining. Although data warehouses are built on relational database technology, the design of a data warehouse database differs substantially from the design of an online transaction processing system (OLTP) database.

14.3.1 Data Warehouses, OLTP, OLAP and Data Mining

A relational database is designed for a specific purpose. Because the purpose of a data warehouse differs from that of an OLTP, the design characteristics of a relational database that supports a data warehouse differ from the design characteristics of an OLTP database.

Notes

Data warehouse database	OLTP database
Designed for analysis of business measures by categories and attributes	Designed for real-time business operations
Optimized for bulk loads and large, complex, unpredictable queries that access many rows per table	Optimized for a common set of transactions, usually adding or retrieving a single row at a time per table
Loaded with consistent, valid data; requires no real time validation	Optimized for validation of incoming data during transactions; uses validation data tables
Supports few concurrent users relative to OLTP	Supports thousands of concurrent users

14.3.2 A Data Warehouse Supports OLTP

A data warehouse supports an OLTP system by providing a place for the OLTP database to offload data as it accumulates, and by providing services that would complicate and degrade OLTP operations if they were performed in the OLTP database.

Without a data warehouse to hold historical information, data is archived to static media such as magnetic tape, or allowed to accumulate in the OLTP database.

If data is simply archived for preservation, it is not available or organized for use by analysts and decision makers. If data is allowed to accumulate in the OLTP so it can be used for analysis, the OLTP database continues to grow in size and requires more indexes to service analytical and report queries. These queries access and process large portions of the continually growing historical data and add a substantial load to the database. The large indexes needed to support these queries also tax the OLTP transactions with additional index maintenance. These queries can also be complicated to develop due to the typically complex OLTP database schema.

A data warehouse offloads the historical data from the OLTP, allowing the OLTP to operate at peak transaction efficiency. High volume analytical and reporting queries are handled by the data warehouse and do not load the OLTP, which does not need additional indexes for their support. As data is moved to the data warehouse, it is also reorganized and consolidated so that analytical queries are simpler and more efficient.

14.3.3 OLAP is a Data Warehouse Tool

Online analytical processing (OLAP) is a technology designed to provide superior performance for ad hoc business intelligence queries. OLAP is designed to operate efficiently with data organized in accordance with the common dimensional model used in data warehouses.

A data warehouse provides a multidimensional view of data in an intuitive model designed to match the types of queries posed by analysts and decision makers. OLAP organizes data warehouse data into multidimensional cubes based on this dimensional model, and then preprocesses these cubes to provide maximum performance for queries that summarize data in various ways. For example, a query that requests the total sales income and quantity sold for a range of products in a specific geographical region for a specific time period can typically be answered in a few seconds or less regardless of how many hundreds of millions of rows of data are stored in the data warehouse database.

OLAP is not designed to store large volumes of text or binary data, nor is it designed to support high volume update transactions. The inherent stability and consistency of historical data in a data warehouse enables OLAP to provide its remarkable performance in rapidly summarizing information for analytical queries.

14.3.4 Data Mining is a Data Warehouse Tool

Data mining is a technology that applies sophisticated and complex algorithms to analyze data and expose interesting information for analysis by decision makers. Whereas OLAP organizes data in a model suited for exploration by analysts, data mining performs analysis on data and provides the results to decision makers. Thus, OLAP supports model-driven analysis and data mining supports data-driven analysis.

Data mining has traditionally operated only on raw data in the data warehouse database or, more commonly, text files of data extracted from the data warehouse database. In SQL Server 2000, Analysis Services provides data mining technology that can analyze data in OLAP cubes, as well as data in the relational data warehouse database. In addition, data mining results can be incorporated into OLAP cubes to further enhance model-driven analysis by providing an additional dimensional viewpoint into the OLAP model. For example, data mining can be used to analyze sales data against customer attributes and create a new cube dimension to assist the analyst in the discovery of the information embedded in the cube data.

14.3.5 Designing a Data Warehouse: Prerequisites

Before embarking on the design of a data warehouse, it is imperative that the architectural goals of the data warehouse be clear and well understood. Because the purpose of a data warehouse is to serve users, it is also critical to understand the various types of users, their needs, and the characteristics of their interactions with the data warehouse.

Data Warehouse Architecture Goals

A data warehouse exists to serve its users analysts and decision makers. A data warehouse must be designed to satisfy the following requirements:

1. Deliver a great user experience user acceptance is the measure of success
2. Function without interfering with OLTP systems
3. Provide a central repository of consistent data
4. Answer complex queries quickly
5. Provide a variety of powerful analytical tools, such as OLAP and data mining

Most successful data warehouses that meet these requirements have these common characteristics:

1. Are based on a dimensional model
2. Contain historical data
3. Include both detailed and summarized data
4. Consolidate disparate data from multiple sources while retaining consistency
5. Focus on a single subject, such as sales, inventory, or finance

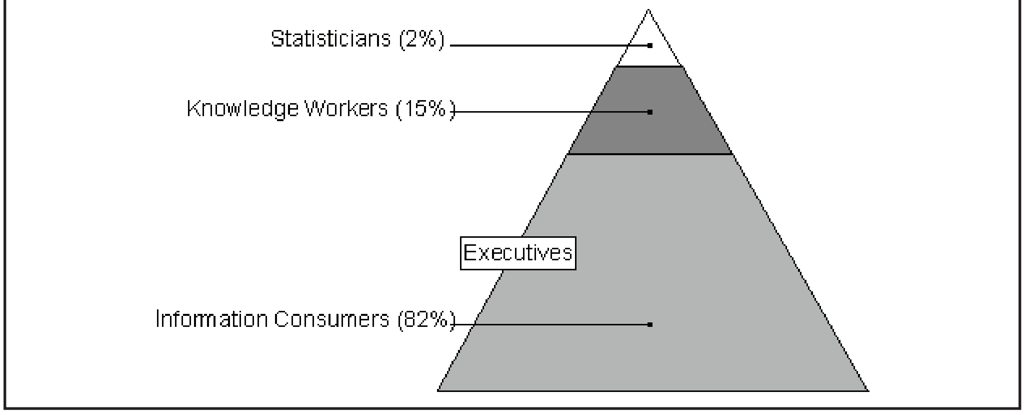
Data warehouses are often quite large. However, size is not an architectural goal it is a characteristic driven by the amount of data needed to serve the users.

14.3.6 Data Warehouse Users

The success of a data warehouse is measured solely by its acceptance by users. Without users, historical data might as well be archived to magnetic tape and stored in the basement. Successful data warehouse design starts with understanding the users and their needs.

Data warehouse users can be divided into four categories: Statisticians, Knowledge Workers, Information Consumers, and Executives. Each type makes up a portion of the user population as illustrated in this Figure 14.6.

Figure 14.6: The User Pyramid



1. ***Statisticians:*** There are typically only a handful of sophisticated analysts Statisticians and operations research types in any organization. Though few in number, they are some of the best users of the data warehouse; those whose work can contribute to closed loop systems that deeply influence the operations and profitability of the company. It is vital that these users come to love the data warehouse. Usually that is not difficult; these people are often very self-sufficient and need only to be pointed to the database and given some simple instructions about how to get to the data and what times of the day are best for performing large queries to retrieve data to analyze using their own sophisticated tools. They can take it from there.

2. ***Knowledge Workers:*** A relatively small number of analysts perform the bulk of new queries and analyses against the data warehouse. These are the users who get the "Designer" or "Analyst" versions of user access tools. They will figure out how to quantify a subject area. After a few iterations, their queries and reports typically get published for the benefit of the Information Consumers. Knowledge Workers are often deeply engaged with the data warehouse design and place the greatest demands on the ongoing data warehouse operations team for training and support.

3. ***Information Consumers:*** Most users of the data warehouse are Information Consumers; they will probably never compose a true ad hoc query. They use static or simple interactive reports that others have developed. It is easy to forget about these users, because they usually interact with the data warehouse only through the work product of others. Do not neglect these users! This group includes a large number of people, and published reports are highly visible. Set up a great communication infrastructure for distributing information widely, and gather feedback from these users to improve the information sites over time.

4. ***Executives:*** Executives are a special case of the Information Consumers group. Few executives actually issue their own queries, but an executive's slightest musing can generate a flurry of activity among the other types of users. A wise data warehouse designer/ implementer/owner will develop a very cool digital dashboard for executives, assuming it is easy and economical to do so. Usually this should follow other data warehouse work, but it never hurts to impress the bosses.

Task Draw & discuss Data warehouse quality reference architecture.

14.4 Optimizing and Materialization of DW Views

Among the techniques adopted in relational implementations of data warehouses to improve query performance, view materialization and indexing are presumably the most effective ones.

Materialized views are physical structures that improve data access time by pre-computing intermediary results. Then, user queries can be efficiently processed by using data stored within views and do not need to access the original data. Nevertheless, the use of materialized views requires additional storage space and entails maintenance overhead when refreshing the data warehouse.

One of the most important issues in data warehouse physical design is to select an appropriate set of materialized views, called a configuration of views, which minimizes total query response time and the cost of maintaining the selected views, given a limited storage space. To achieve this goal, views that are closely related to the workload queries must be materialized.

Materialized views are schema objects that can be used to summarize, precompute, replicate, and distribute data, e.g., to construct a data warehouse.

A materialized view provides indirect access to table data by storing the results of a query in a separate schema object. Unlike an ordinary view, which does not take up any storage space or contain any data.

The motivation for using materialized views is to improve performance, but the overhead associated with materialized view management can become a significant system management problem. When reviewing or evaluating some of the necessary materialized view management activities, consider some of the following:

1. Identifying what materialized views to create initially
2. Indexing the materialized views
3. Ensuring that all materialized views and materialized view indexes are refreshed properly each time the database is updated
4. Checking which materialized views have been used
5. Determining how effective each materialized view has been on workload performance
6. Measuring the space being used by materialized views
7. Determining which new materialized views should be created
8. Determining which existing materialized views should be dropped
9. Archiving old detail and materialized view data that is no longer useful

After the initial effort of creating and populating the data warehouse or data mart, the major administration overhead is the update process, which involves:

1. Periodic extraction of incremental changes from the operational systems
2. Transforming the data
3. Verifying that the incremental changes are correct, consistent, and complete
4. Bulk-loading the data into the warehouse
5. Refreshing indexes and materialized views so that they are consistent with the detail data

Materialized Views and Models

Models, which provide array-based computations in SQL, can be used in materialized views. Because the MODEL clause calculations can be expensive, you may want to use two separate materialized views: one for the model calculations and one for the SELECT ... GROUP BY query. For example, instead of using one, long materialized view, you could create the following materialized views:

```
CREATE MATERIALIZED VIEW my_groupby_mv
REFRESH FAST
ENABLE QUERY REWRITE AS
SELECT country_name country, prod_name prod, calendar_year year,
 SUM(amount_sold) sale, COUNT(amount_sold) cnt, COUNT(*) cntstr
FROM sales, times, customers, countries, products
WHERE sales.time_id = times.time_id AND
 sales.prod_id = products.prod_id AND
 sales.cust_id = customers.cust_id AND
 customers.country_id = countries.country_id
GROUP BY country_name, prod_name, calendar_year;
CREATE MATERIALIZED VIEW my_model_mv
ENABLE QUERY REWRITE AS
SELECT country, prod, year, sale, cnt
FROM my_groupby_mv
MODEL PARTITION BY(country) DIMENSION BY(prod, year)
 MEASURES(sale s) IGNORE NAV
(s['Shorts', 2000] = 0.2 * AVG(s)[CURRENTV(), year BETWEEN 1996 AND 1999],
s['Kids Pajama', 2000] = 0.5 * AVG(s)[CURRENTV(), year BETWEEN 1995 AND 1999],
s['Boys Pajama', 2000] = 0.6 * AVG(s)[CURRENTV(), year BETWEEN 1994 AND 1999],
...
<hundreds of other update rules>);
```

By using two materialized views, you can incrementally maintain the materialized view my_groupby_mv. The materialized view my_model_mv is on a much smaller data set because it is built on my_groupby_mv and can be maintained by a complete refresh.

Materialized views with models can use complete refresh or PCT refresh only, and are available for partial text query rewrite only.

Notes

Case Study

FedEx

Cost Savings All Day Long

How does FedEx make the case for IT spending? Cost savings is a large component. In particular, an innovative Web-based customer service portal, called FedEx InSight, has aligned with significant increases in the use of FedEx services by some of the company's most valuable customers.

With FedEx InSight, business customers view all outgoing, incoming, and third-party shipments online and they prefer interacting with FedEx Insight over other channels. In fact, they like it so much that they forgo lower rates from competitors in order to have access to FedEx online tracking.

Cutting costs while increasing customer loyalty, InSight is considered by FedEx to be a milestone technology. The innovative Web service lets business customers instantly access all their current FedEx cargo information, tailor views, and drill down into freight information including shipping date, weight, contents, expected delivery date, and related shipments. Customers can even opt for email notifications of in-transit events, such as attempted deliveries, delays at customs, etc.

The Perfect Match

InSight works because FedEx can link shipper and receiver data on shipping bills with entries in a database of registered InSight customers. The linking software FedEx chose to support InSight had to be superior in terms of its ability to recognize, interpret, and match customer name and address information. Fast processing speed and flexibility were also top criteria. After a broad and thorough evaluation of vendors in the data quality market, the delivery giant chose Trillium Software®.

The real matching challenge was not with the records for outgoing shippers, who could be easily identified by their account numbers. Linking shipment recipients to customers in the InSight database was far more difficult. It relied on name and address information, which is notoriously fraught with errors, omissions, and other anomalies—especially when entered by individual shippers around the globe. The point of pain was being able to match on addresses, because what FedEx receives on the airbills is not very standardized.

FedEx airbills had another problem: too much information. "For the purpose of matching customers to shipments, the airbills contain a lot of garbage," said FedEx's senior technical analyst. "Information such as parts numbers, stock-keeping units, signature requirements, shipping contents, delivery instructions, country of manufacture, and more obscures the name and address data, making it difficult to interpret that free-form text and correctly identify name and address information."

A Deeper Look at Data

As Trillium Software® demonstrated to FedEx during the sales cycle, no matching software would be successful for FedEx airbills without some intelligent interpretation of free-form text and standardization. Matching is more accurate when it acts on more complete and standardized data.

The Trillium Software System® first investigates data entries word by word—not just line by line—in order to understand maximum data content. Valid content is often "hidden" when it's entered in the wrong field or free-form text fields. Trillium technology reveals this hidden data by identifying individual data elements in each shipping bill, interpreting

Contd...

the real meaning of each element, and ensuring that all valid data elements are part of the matching equation. It then standardizes content into a consistent format.

Beyond Address Correction

In the logistics industry, accuracy is everything; FedEx needed to identify customers reliably based on a variety of data elements, including business names, office suite numbers, and other address elements. Its chosen data quality solution had to identify and distinguish between companies on different floors of an office tower or divisions within a geographically dispersed corporate campus. It had to link customers based on detailed analyses of abbreviations, nicknames, synonyms, personal names, street addresses, and other information.

FedEx knew they needed more than address verification software that only confirmed that an address was internally consistent and correct according to postal authorities. They needed precision matching capabilities that would lie at the heart of InSight. The Trillium Software System had all these capabilities, in addition to usability features that allowed FedEx to tune and test quickly and iteratively the matching process until the match results met the company's stringent requirements.

Split-Second Processing

Speed was another requirement. To efficiently handle the volume of FedEx's daily transactions, the software had to identify and resolve matches at sub-second rates. Only the Trillium Software System could demonstrate this capability and process millions of records per day—as many as 500,000 records per hour.

Surgical Precision

"That we could customize [business] rules and surgically make changes was a big, big winning point," said the senior technical analyst. The Trillium Software System lets FedEx target specific data issues and quickly modify rules to resolve them. The business rules, written in plain text, were understandable, traceable, and repeatable. Because the analyst team could see what the rules were and how they worked, they were more confident about the matching process. "There's a certain amount of confidence you have to have in the product. You have to trust in what it's doing."

Rapid Rollout

FedEx took only about four months to implement its solution fully. Trillium Software professional services helped FedEx get started. After just three days, the senior technical analyst was ready to work on his own: "Once I understood it, it was just a matter of applying that knowledge," he stated.

He also gives Trillium Software Customer Support a lot of credit: "The Trillium Software tech support is just terrific. Most of my support is done through email and someone always gets back to me quickly. If I call, there's no kind of triage. I tell them what language I speak, and then I get to talk to someone."

Award-Winning InSight

FedEx won several e-commerce awards for its innovation, and customers raved about their InSight experiences. In fact, FedEx customers communicated that they would forgo lower shipping rates from competitors, because they prized the ability to track their incoming and outgoing shipments so easily with InSight.

FedEx also realized concrete gains from its investment. Repeatedly, implementation of InSight was shown to align with significant increases in use of FedEx services by some of the company's largest customers.

Contd...

Notes

International Expansion

Based on the success of InSight in the US and Canada, FedEx extended the service to other countries simply by adding global modules. With geographic validation for every country on the globe and in-depth support for more than 60 countries in Europe, Asia-Pacific, and the Americas, the Trillium Software System kept up with FedEx InSight.

14.5 Summary

- We have presented a comprehensive approach how to improve the quality of data warehouses through the enrichment of metadata based on explicit modeling of the relationships between enterprise models, source models, and client interest models.
- Our algorithms, prototypical implementations, and partial validations in a number of real-world applications demonstrate that an almost seamless handling of conceptual, logical, and physical perspectives to data warehousing is feasible, considering a broad range of quality criteria ranging from business-oriented accuracy and actuality to technical systems performance and scalability.

14.6 Keywords

Data Mining: Data mining is a technology that applies sophisticated and complex algorithms to analyze data and expose interesting information for analysis by decision makers.

Data Warehouse: Data warehouses support business decisions by collecting, consolidating, and organizing data for reporting and analysis with tools such as online analytical processing (OLAP) and data mining.

DWQ: DWQ provides assistance to DW designers by linking the main components of the data warehouse reference architecture to a formal model of data quality.

OLAP: Online analytical processing (OLAP) is a technology designed to provide superior performance for ad hoc business intelligence queries.

14.7 Self Assessment

Choose the appropriate answers:

1. DWQ stands for:
   (a) Database Warehouse Quality
   (b) Data Warehouse Quantity
   (c) Data Warehouse Quality
   (d) Document Warehouse Quality
2. OLAP stands for:
   (a) Online Analytical Processing
   (b) Offline Analytical Processing
   (c) Online Analytical Programming
   (d) Online Available Processing

Fill in the blanks:

3. OLAP is not designed to store large volumes of .... or binary data.
4. .... are a special case of the Information Consumers group.
5. .... are physical structures that improve data access time by pre-computing intermediary results
6. .... has become an important strategy to integrate heterogeneous information sources in organizations.
7. Vendors agree that data warehouses cannot be off-the-shelf products but must be designed and optimized with great attention to the .... situation.
8. A .... is designed for a specific purpose.
9. A data warehouse offloads the historical data from the ....
10. A data warehouse provides a multidimensional view of data in an .... designed to match the types of queries posed by analysts and decision makers.

14.8 Review Questions

1. Describe how data mining is a data warehouse tool.
2. Describe the various architectural goals of data warehouse.
3. Write short note on users of data warehouse.
4. Explain with the help of suitable example how will you optimize and materializing of views.
5. What do you mean by quality driven data warehouse?
6. Describe what is the interaction between quality factors and data warehouse tasks?
7. "The DWQ project will produce semantic foundations for DW design and evolution linked explicitly to formal quality models". Explain.
8. Describe various design methodologies of warehouse.
9. "A data warehouse provides a multidimensional view of data in an intuitive model designed to match the types of queries posed by analysts and decision makers." Discuss.
10. "The DWQ project will develop policies to extend active database concepts such that data caching is optimized for a given transaction load on the DW." Discuss.

Answers: Self Assessment

1. (c)	2. (a)
3. text	4. Executives
5. Materialized views	6. Data warehousing
7. customer	8. relational database
9. OLTP	10. intuitive model

Notes

14.9 Further Readings

Books

A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

Alex Berson, *Data Warehousing Data Mining and OLAP*, Tata Mcgraw Hill, 1997

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

Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

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

Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, *Fundamentals of Data Warehouses*, Publisher: Springer

Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

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

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

Sholom M. Weiss and Nitin Indurkhya, "*Predictive Data Mining: A Practical Guide*", Morgan Kaufmann Publishers, 1998.

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

Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

V. Cherkassky and F. Mulier, *Learning From Data,* John Wiley & Sons, 1998.

Online links

www.en.wikipedia.org

www.web-source.net

www.webopedia.com