Rai Technology University

ENGINEERING MINDS

Data Warehousing

Subject: DATA WAREHOUSING **Credits: 4**

SYLLABUS

Basic Concepts of Data Warehousing
Introduction, Meaning and characteristics of Data Warehousing, Online Transaction Processing (OLTP), Data Warehousing Models, Data warehouse architecture & Principles of Data Warehousing Data Mining.

Building a Data Warehouse Project
Structure of the Data warehouse, Data warehousing and Operational Systems, Organizing for building data warehousing, Important considerations – Tighter integration, Empowerment, Willingness Business Considerations: Return on Investment Design Considerations, Technical Consideration, Implementation Consideration, Benefits of Data warehousing.

Managing and Implementing a Data Warehouse Project
Project Management Process, Scope Statement, Work Breakdown Structure and Integration, Initiating a data warehousing project Project Estimation, Analyzing Probability and Risk, Managing Risk: Internal and External, Critical Path Analysis.

Data Mining
What is Data mining (DM)? Definition and description, Relationship and Patterns, KDD vs Data mining, DBMS vs Data mining, Elements and uses of Data Mining, Measuring Data Mining Effectiveness : Accuracy,Speed & Cost Data Information and Knowledge, Data Mining vs. Machine Learning, Data Mining Models. Issues and challenges in DM, DM Applications Areas.

Techniques of Data Mining
Various Techniques of Data Mining Nearest Neighbour and Clustering Techniques, Decision Trees, Discovery of Association Rules, Neural Networks, Genetic Algorithm.

OLAP
Need for OLAP, OLAP vs. OLTP Multidimensional Data Model Multidimensional verses Multirelational OLAP Characteristics of OLAP: FASMI Test (Fast, Analysis Share**,** Multidimensional and Information), Features of OLAP, OLAP Operations Categorization of OLAP Tools: MOLAP, ROLAP

Suggested Readings:

1. Pieter Adriaans, Dolf Zantinge Data Mining, Pearson Education
2. George M. Marakas Modern Data Warehousing, Mining, and Visualization: Core Concepts, Prentice Hall, 1st edition
3. Alex Berson, Stephen J. Smith Data Warehousing, Data Mining, and OLAP (Data Warehousing/Data Management), McGraw-Hill
4. Margaret H. Dunham Data Mining, Prentice Hall, 1st edition,
5. David J. Hand Principles of Data Mining (Adaptive Computation and Machine Learning), Prentice Hall, 1st edition
6. Jiawei Han, Micheline Kamber Data Mining, Prentice Hall, 1st edition
7. Michael J. Corey, Michael Abbey, Ben Taub, Ian Abramson Oracle 8i Data Warehousing McGraw-Hill Osborne Media, 2nd edition

COURSE OVERVIEW

The last few years have seen a growing recognition of information as a key business tool. In general, the current business market dynamics make it abundantly clear that, for any company, information is the very key to survival.

If we look at the evolution of the information processing technologies, we can see that while the first generation of client/server systems brought data to the desktop, not all of this data was easy to understand, unfortunately, and as such, it was not very useful to end users. As a result, a number of new technologies have emerged that are focused on improving the information content of the data to empower the knowledge workers of today and tomorrow. Among these technologies are data warehousing, online analytical processing (OLAP), and data mining.

Therefore, this book is about the need, the value and the technological means of acquiring and using information in the information age.

From that perspective, this book is intended to become the handbook and guide for anybody who's interested in planning, or working on data warehousing and related issues. Meaning and characteristics of Data Warehousing, Data Warehousing Models, Data warehouse architecture & Principles of Data Warehousing, topics related to building a data warehouse project are discussed along with Managing and implementing a data warehouse project. Using these topics as a foundation, this book proceeds to analyze various important concepts related to Data mining, Techniques of data mining, Need for OLAP, OLAP vs. OLTP, Multidimensional data model, Multidimensional verses Multirelational OLAP, OLAP Operations and Categorization of OLAP Tools: MOLAP and ROLAP.

Armed with the knowledge of data warehousing technology, the student continues into a discussion on the principles of business analysis, models and patterns and an in-depth analysis of data mining.

Prerequisite

Knowledge of Database Management Systems

Objective

Ever since the dawn of business data processing, managers have been seeking ways to increase the utility of their information systems. In the past, much of the emphasis has been on automating the transactions that move an organization through the interlocking cycles of sales, production and administration. Whether accepting an order, purchasing raw materials, or paying employees, most organizations process an enormous number of transactions and in so doing gather an even larger amount of data about their business.

Despite all the data they have accumulated, what users really want is information. In conjunction with the increased amount of data, there has been a shift in the primary users of computers, from a limited group of information systems professionals to a much larger group of knowledge workers with expertise in particular business domains, such as finance, marketing, or manufacturing. Data warehousing is a collection of technologies designed to convert heaps of data to usable information. It

does this by consolidating data from diverse transactional systems into a coherent collection of consistent, quality-checked databases used only for informational purposes. Data warehouses are used to support online analytical processing (OLAP).

However, the very size and complexity of data warehouses make it difficult for any user, no matter how knowledgeable in the application of data, to formulate all possible hypotheses that might explain something such as the behavior of a group of customers. How can anyone successfully explore databases containing 100 millions rows of data, each with thousands of attributes?

The newest, hottest technology to address these concerns is data mining. Data mining uses sophisticated statistical analysis and modeling techniques to uncover pattern and relationships hidden in organizational databases – patterns that ordinary methods might miss.

The objective of this book is to have detailed information about Data warehousing, OLAP and data mining. I have brought together these different pieces of data warehousing, OLAP and data mining and have provided an understandable and coherent explanation of how data warehousing as well as data mining works, plus how it can be used from the business perspective. This book will be a useful guide.

DATA WAREHOUSING

CONTENT

CONTENT

LESSON 1
INTRODUCTION TO
DATA WAREHOUSING

Structure

- Objective
- Introduction
- Meaning of Data warehousing
- History of Data warehousing
- Traditional Approaches To Historical Data
- Data from legacy systems
- Extracted information on the Desktop
- Factors, which Lead To Data Warehousing

Objective

The main objective of this lesson is to introduce you with the basic concept and terminology relating to Data Warehousing. By the end of this lesson you will be able to understand:

- Meaning of a Data warehouse
- Evolution of Data warehouse

Introduction

Traditionally, business organizations create billions of bytes of data about all aspects of business everyday, which contain millions of individual facts about their customers, products, operations, and people. However, this data is locked up and is extremely difficult to get at. Only a small fraction of the data that is captured, processed, and stored in the enterprise is actually available to executives and decision makers.

Recently, new concepts and tools have evolved into a new technology that make it possible to provide all the key people within the enterprise with access to whatever level of information needed for the enterprise to survive and prosper in an increasingly competitive world. The term that is used for this new technology is "data warehousing". In this unit I will be discussing about the basic concept and terminology relating to Data Warehousing.

The Lotus was your first test 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.

Meaning of Data Warehousing

Data warehouse potential can be magnify 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 system. It can be loosely defined as any centralized data repository, which can be queried for business benefit, but this will be more clearly defined letter. Data warehouse is a new powerful technique making. It possible to extract archived operational data and over come 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 application and the company's operational activities Johan McIntyre of SAS institute Inc.

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

A data warehouse is a collection of corporate information, derived directly from operational system and some external data sources.

Its specific purpose is to support business decisions, not business ask "What if?" questions. The answer to these questions will ensure your business is proactive, instead of reactive, a necessity in today's information ago.

The industry trend today is moving towards more powerful hardware and software configuration, we now have the ability to process vast volumes of information analytically, which would have been unheard of tenor even five years ago. A business today must we able to use this emerging technology or run the risk if being information under loaded. As you read that correctly - under loaded - the opposite of over loaded. Over-loaded means you are so determine what is important. If you are under loaded, you are information deficient. You cannot cope with decision – making expectation because you do not know where you stand. You are missing critical pieces of information required to make informed decisions.

To illustrate the danger of being information under loaded, consider the children's story of the country mouse is unable to cope with and environment its does not understand.

What is a cat? Is it friend or foe?

Why is the chess in the middle of the floor on the top of a platform with a spring mechanism?

Sensory deprivation and information overload set in. The picture set country mouse cowering in the corner. If is stays there, it will shrivel up and die. The same fate awaits the business that does not respond to or understand the environment around it. The competition will moves in like cultures and exploit all like weaknesses.

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 understating of, information is power. This power equate to a competitive advantage and survival. This unit will discuss building own data warehouse-a repository for storing information your business needs to use if it hopes to survive and thrive in the information age. We will help you

understand what a data warehouse is and what it is not. You will learn what human resources are required, as well as the roles and responsibilities of each player. You will be given an overview of good project management techniques to help ensure the data warehouse initiative dose not fail due the poor project management. You will learn how to physically implements a data warehouse with some new tools currently available to help you mine those vast amounts of information stored with in the warehouse. Without fine running this ability to mine the warehouse, even the most complete warehouse, would be useless.

History of Data Warehousing

Let us first review the historical management schemes of the analysis data and the factors that have led to the evolution of the data warehousing application class.

Traditional Approaches to Historical Data

Throughout the history of systems development, the primary emphasis had been given to the operational systems and the data they process. It was not practical to keep data in the operational systems indefinitely; and only as an afterthought was a structure designed for archiving the data that the operational system has processed. The fundamental requirements of the operational and analysis systems are different: the operational systems need performance, whereas the analysis systems need flexibility and broad scope.

Data from Legacy Systems

Different platforms have been developed with the development of the computer systems over past three decades. In the 1970's, business system development was done on the IBM mainframe computers using tools such as Cobol, CICS, IMS, DB2, etc. With the advent of 1980's computer platforms such as AS/400 and VAX/VMS were developed. In late eighties and early nineties UNIX had become a popular server platform introducing the client/server architecture which remains popular till date.

Despite all the changes in the platforms, architectures, tools, and technologies, a large number of business applications continue to run in the mainframe environment of the 1970's. The most important reason is that over the years these systems have captured the business knowledge and rules that are incredibly difficult to carry to a new platform or application. These systems are, generically called legacy systems. The data stored in such systems ultimately becomes remote and becomes difficult to get at.

Extracted Information on the Desktop

During the past decade the personal computer has become very popular for business analysis. Business Analysts now have many of the tools required to use spreadsheets for analysis and graphic representation. Advanced users will frequently use desktop database programs to store and work with the information extracted from the legacy sources.

The disadvantage of the above is that it leaves the data fragmented and oriented towards very specific needs. Each individual user has obtained only the information that she/he requires. The extracts are unable to address the requirements of multiple users and uses. The time and cost involved in addressing the requirements of only one user are large. Due to the disadvantages faced it led to the development of the new application called **Data Warehousing**

Factors, which Lead To Data Warehousing

Many factors have influenced the quick evolution of the data warehousing discipline. The most important factor has been the advancement in the hardware and software technologies.

Hardware and Software prices: Software and hardware prices have fallen to a great extent. Higher capacity memory chips are available at very low prices.

- **Powerful Preprocessors:** Today's preprocessor are many times powerful than yesterday's mainframes: e.g. Pentium III and Alpha processors
- **Inexpensive disks:** The hard disks of today can store hundreds of gigabytes with their prices falling. The amount of information that can be stored on just a single one-inch high disk drive would have required a roomful of disk drives in 1970's and early eighties.
- **Desktop powerful for analysis tools:** Easy to use GUI interfaces, client/server architecture or multi-tier computing can be done on the desktops as opposed to the mainframe computers of yesterday.
- **Server software:** Server software is inexpensive, powerful, and easy to maintain as compared to that of the past. Example of this is Windows NT that have made setup of powerful systems very easy as well as reduced the cost.

The skyrocketing power of hardware and software, along with the availability of affordable and easy-to-use reporting and analysis tools have played the most important role in evolution of data warehouses.

Emergence of Standard Business Applications

New vendors provide to end-users with popular business application suites. German software vendor SAP AG, Baan, PeopleSoft, and Oracle have come out with suites of software that provide different strengths but have comparable functionality. These application suites provide standard applications that can replace the existing custom developed legacy applications. This has led to the increase in popularity of such applications. Also, the data acquisition from these applications is much simpler than the mainframes.

End-user more Technology Oriented

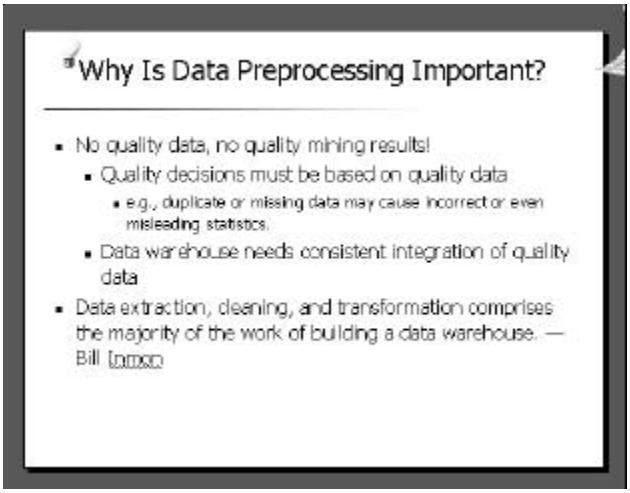
One of the most important results of the massive investment in technology and movement towards the powerful personal computer has been the evolution of a technology-oriented business analyst. Even though the technology-oriented end users are not always beneficial to all projects, this trend certainly has produced a crop of technology-leading business analysts that are becoming essential to today's business. These technology-oriented end users have frequently played an important role in the development and deployment of data warehouses. They have become the core users that are first to demonstrate the initial benefits of data warehouses. These end users are also critical to the development of the data warehouse model: as they become experts with the data warehousing system, they train other users.

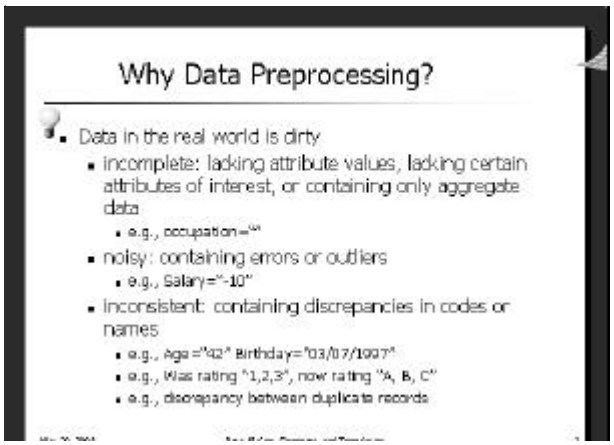
Discussions

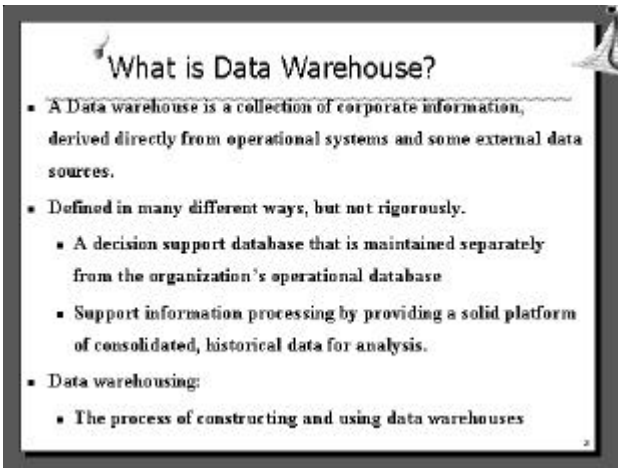
- Write short notes on:
  - Legacy systems
  - Data warehouse
  - Standard Business Applications
- What is a Data warehouse? How does it differ from a database?
- Discuss various factors, which lead to Data Warehousing.
- Briefly discuss the history behind Data warehouse.

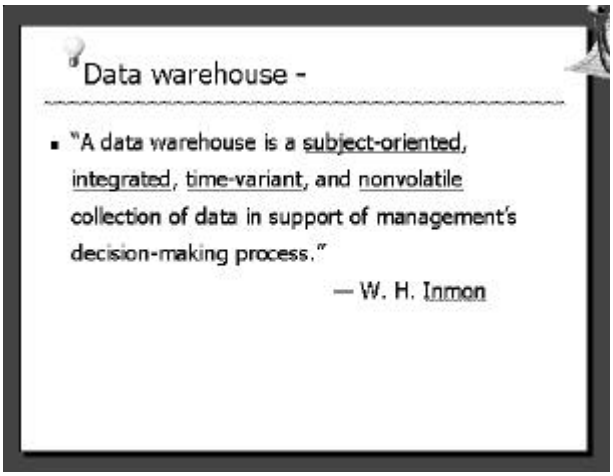
References

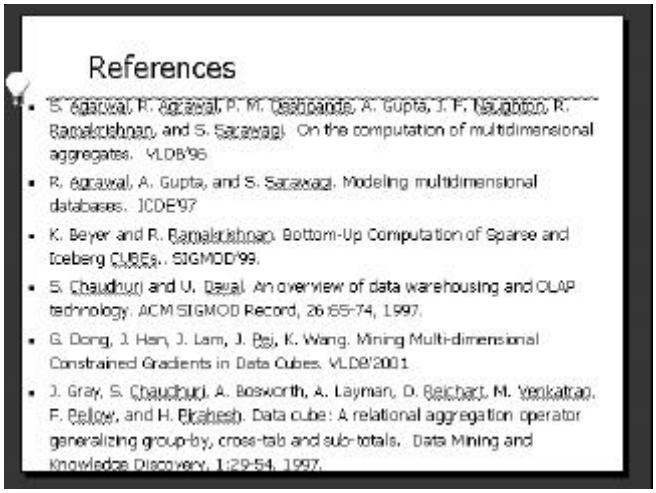
1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. Anahory, Sam, Data warehousing in the real world: a practical guide for building decision support systems, Delhi: Pearson Education Asia, 1997.
3. Berry, Michael J.A. ; Linoff, Gordon, Mastering data mining : the art and science of customer relationship management, New York : John Wiley & Sons, 2000
4. Corey, Michael, Oracle8 data warehousing, New Delhi: Tata McGraw- Hill Publishing, 1998.
5. Elmasri, Ramez, Fundamentals of database systems, 3rd ed. Delhi: Pearson Education Asia, 2000.










References
S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs.. SIGMOD'99.
S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997.
G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB'2001
J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.

Notes

LESSON 2
MEANING AND CHARACTERISTICS OF DATA WAREHOUSING

Structure

- Objective
- Introduction
- Data warehousing
- Operational vs. Informational Systems
- Characteristics of Data warehousing
- Subject oriented
- Integrated
- Time variant
- Non-volatiles

Objective

The objective of this lesson is to explain you the significance and difference between Operational systems and Informational systems. This lesson also includes various characteristics of a Data warehouse.

Introduction

In the previous section, we have discussed about the need of data warehousing and the factors that lead to it. In this section I will explore the technical concepts relating to data warehousing to you.

A company can have data items that are unrelated to each other. Data warehousing is the process of collecting together such data items within a company and placing it in an integrated data store. This integration is over time, geographies, and application platforms. By adding access methods (on-line querying, reporting), this converts a 'dead' data store into a 'dynamic' source of information. In other words, turning a liability into an asset. Some of the definitions of data warehousing are:

"A data warehouse is a single, complete, and consistent store of data obtained from a variety of sources and made available to end users in a way they can understand and use in a business context." (Devlin 1997)

"Data warehousing is a process, not a product, for assembling and managing data from various sources for the purpose of gaining a single, detailed view of part or all of the business." (Gardner 1998)

A Data Warehouse is a capability that provides comprehensive and high integrity data in forms suitable for decision support to end users and decision makers throughout the organization. A data warehouse is managed data situated after and outside the operational systems. A complete definition requires discussion of many key attributes of a data warehouse system Data Warehousing has been the result of the repeated attempts of various researchers and organizations to provide their organizations flexible, effective and efficient means of getting at the valuable sets of data.

Data warehousing evolved with the integration of a number of different technologies and experiences over the last two decades, which have led to the identification of key problems.

Data Warehousing

Because data warehouses have been developed in numerous organizations to meet partic-ular needs, there is no single, canonical definition of the term data warehouse.[1] Profes-sional magazine articles and books in the popular press have elaborated on the meaning in a variety of ways. Vendors have capitalized on the popularity of the term to help mar-ket a variety of related products, and consultants have provided a large variety of services, all under the data-warehousing banner. However, data warehouses are quite distinct from traditional databases in their structure, functioning, performance, and purpose.

Operational vs. Informational Systems

Perhaps the most important concepts that has come out of the Data Warehouse movement is the recognition that there are two fundamentally different types of information systems in all organizations: operational systems and informational systems.

"Operational systems" are just what their name implies, they are the systems that help us run the enterprise operate day-to-day. These are the backbone systems of any enterprise, our "order entry', "inventory", "manufacturing", "payroll" and "accounting" systems. Because of their importance to the organization, operational systems were almost always the first parts of the enterprise to be computerized. Over the years, these operational systems have been extended and rewritten, enhanced and maintained to the point that they are completely integrated into the organization. Indeed, most large organizations around the world today couldn't operate without their operational systems and that data that these systems maintain.

On the other hand, there are other functions that go on within the enterprise that have to do with planning, forecasting and managing the organization. These functions are also critical to the survival of the organization, especially in our current fast paced world. Functions like "marketing planning", "engineering planning" and "financial analysis" also require information systems to support them. But these functions are different from operational ones, and the types of systems and information required are also different. The knowledge-based functions are informational systems.

"Informational systems" have to do with analyzing data and making decisions, often major decisions about how the enterprise will operate, now and in the future. And not only do informational systems have a different focus from operational ones, they often have a different scope. Where operational data needs are normally focused upon a single area, informational data needs often span a number of different areas and need large amounts of related operational data.

In the last few years, Data Warehousing has grown rapidly from a set of related ideas into architecture for data delivery for enterprise end user computing.

They support high-performance demands on an organization's data and information. Several types of applications-OLAP, DSS, and data mining applications-are supported. OLAP (on-line analytical processing) is a term used to describe the analysis of complex data from the data warehouse. In the hands of skilled knowledge workers. OLAP tools use distributed computing capabilities for analyses that require more storage and processing power than can be economically and efficiently located on an individual desktop. DSS (Decision-Support Systems) also known as EIS (Executive Information Systems, not to be confused with enterprise integration systems) support an organization's leading deci-sion makers with higher-level data for complex and important decisions. Data mining is used for knowledge discovery, the pro-cess of searching data for unanticipated new knowledge.

Traditional databases support On-Line Transaction Processing (OLTP), which includes insertions, updates, and deletions, while also supporting information query requirements. Traditional relational databases are optimized to process queries that may touch a small part of the database and transactions that deal with insertions or updates of a few tuples per relation to process. Thus, they cannot be optimized for OLAP, DSS, or data mining. By contrast, data warehouses are designed precisely to support efficient extraction, process-ing, and presentation for analytic and decision-making purposes. In comparison to traditional databases, data warehouses generally contain very large amounts of data from multiple sources that may include databases from different data models and sometimes lies acquired from independent systems and platforms.

A database is a collection of related data and a database system is a database and database software together. A data warehouse is also a collection of information as well as supporting system. However, a clear distinction exists, Traditional databases are transactional: relational, object-oriented, network, or hierarchical. Data warehouses have the distinguishing characteristic that they are mainly intended for decision-support applications. They are optimized for data retrieval, not routine transaction processing.

Characteristics of Data Warehousing

As per W. H. Inmon, author of building the data warehouse and the guru who is ready widely considered to be the originator of the data warehousing concept, there are generally four character that describe a data warehouse:

W. H. Inmon characterized a data warehouse as "a subject-oriented, integrated, nonvola-tile, time-variant collection of data in support of management's decisions." Data ware-houses provide access to data for complex analysis, knowledge discovery, and decision-making.

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 costumer, 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.

Integrated

When data resides in money 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 o and l. When data are moved from the operational environment in to the data warehouse, when data are moved from the operational environment in to the data warehouse, they assume a consistent coding convention e.g. gender data is transformed to "m" and "f".

Time variant

The data warehouse contains a place for storing data that are five to ten years old, or older, to be used for comparisons, trends, and forecasting. These data are not up dated.

Non-volatile

Data are not update or changed in any way once they enter the data warehouse, but are only loaded and accessed.

Data warehouses have the following distinctive characteristics.

- Multidimensional conceptual view.
- Generic dimensionality.
- Unlimited dimensions and aggregation levels.
- Unrestricted cross-dimensional operations.
- Dynamic sparse matrix handling.
- Client-server architecture.
- Multi-user support.
- Accessibility.
- Transparency.
- Intuitive data manipulation.
- Consistent reporting performance.
- Flexible reporting

Because they encompass large volumes of data, data warehouses are generally an order of magnitude (sometimes two orders of magnitude) larger than the source databases. The sheer volume of data (likely to be in terabytes) is an issue that has been dealt with through enterprise-wide data warehouses, virtual data warehouses, and data marts:

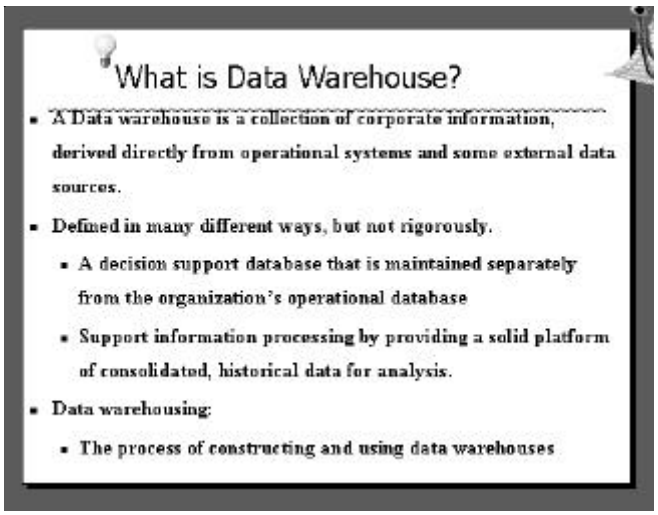
- Enterprise-wide data warehouses are huge projects requiring massive investment of time and resources.
- Virtual data warehouses provide views of operational databases that are materialized for efficient access.
- Data marts generally are targeted to a subset of the organization, such as a dependent, and are more tightly focused.

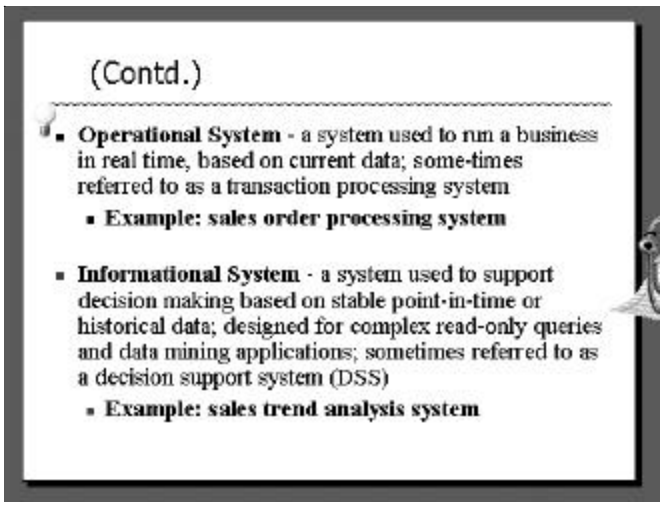
To summarize the above, here are some important points to remember about various characteristics of a Data warehouse:

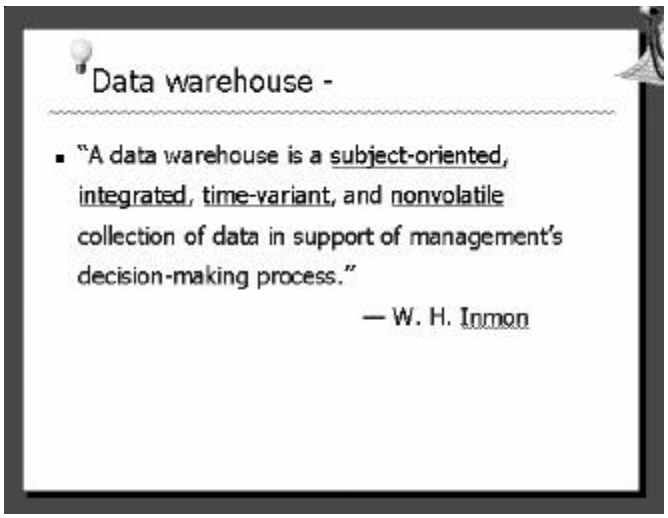
- **Subject-oriented**
  - Organized around major subjects, such as customer, product, sales.

- Focusing on the modeling and analysis of data for decision making, not on daily operations or transaction processing.
- Provide a simple and concise view around particular subject by excluding data that are not useful in the decision support process.

- **Integrated**
  - Constructed by integrating multiple, heterogeneous data sources as relational databases, flat files, on-line transaction records.
  - Providing data cleaning and data integration techniques.
- **Time variant**
  - The time horizon for the data warehouse is significantly longer than that of operational systems.
  - Every key structure in the data warehouse contains an element of time (explicitly or implicitly).
- **Non-volatile**
  - A physically separate store of data transformed from the operational environment.
  - Does not require transaction processing, recovery, and concurrency control mechanisms.
  - Requires only two operations in data accessing: initial loading of data and access of data (no data updates).







Discussions

- Write short notes on:
  - Metadata
  - Operational systems
  - OLAP
  - DSS
  - Informational Systems
- What is the need of a Data warehouse in any organization?
- Discuss various characteristics of a Data warehouse.
- Explain the difference between non-volatile and Subject-oriented data warehouse.

References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
4. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
5. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

Data Warehouse—Subject-Oriented

- Organized around major subjects, such as customer, product, sales.
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing.
- Provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

Data Warehouse—Non-Volatile

- Operational update of data does not occur in the data warehouse environment.
  - Does not require transaction processing, recovery, and concurrency control mechanisms
  - Requires only two operations in data accessing:
    - initial loading of data and access of data.

Data Warehouse—Integrated

- Constructed by integrating multiple, heterogeneous data sources
  - relational databases, flat files, on-line transaction records
- Data cleaning and data integration techniques are applied.
  - Ensure consistency in naming conventions, encoding structures, attribute measures, etc. among different data sources
    - E.g., Hotel price: currency, tax, breakfast covered, etc.
  - When data is moved to the warehouse, it is converted.

References

- S. Sarawal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs.. SIGMOD'99.
- S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997.
- G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB'2001
- J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.

Data Warehouse—Time Variant

- The time horizon for the data warehouse is significantly longer than that of operational systems.
  - Operational database: current value data.
  - Data warehouse data: provide information from a historical perspective (e.g., past 5-10 years)
- These data are not updated.

LESSON 3
ONLINE TRANSACTION PROCESSING

Structure

- Objective
- Introduction
- Data warehousing and OLTP systems
- Similarities and Differences in OLTP and Data Warehousing Processes in Data Warehousing OLTP
- What is OLAP?
- Who uses OLAP and WHY?
- Multi-Dimensional Views
- Benefits of OLAP

Objective

The main objective of this lesson is to introduce you with Online Transaction Processing. You will learn about the importance and advantages of an OLTP system.

Introduction

Relational databases are used in the areas of operations and control with emphasis on transaction processing. Recently relational databases are used for building data warehouses, which stores tactical information (<1year into the future) that answers who and what questions. In contrast OLAP uses MD views of aggregate data to provide access strategic information. OLAP enables users to gain insight to a wide variety of possible views of information and transforms raw data to reflect the enterprise as understood by the user e.g., Analysts, managers and executives.

Data Warehousing and OLTP Systems

A data base which in built for on line transaction processing, OLTP, is generally regarded as inappropriate for warehousing as they have been designed with a different set of need in mind i.e., maximizing transaction capacity and typically having hundreds of table in order not to look out user etc. Data warehouse are interested in query processing as opposed to transaction processing.

OLTP systems cannot be receptacle stored of repositories of facts and historical data for business analysis. They cannot be quickly answer adhoc queries is rapid retrieval is almost impossible. The data is inconsistent and changing, duplicate entries exist, entries can be missing and there is an absence of historical data, which is necessary to analyses trends. Basically OLTP offers large amounts of raw data, which is not easily understood. The data warehouse offers the potential to retrieve and analysis information quickly and easily. Data warehouse do have similarities with OLTP as shown in the table below.

Similarities and Differences in OLTP and Data Warehousing

	OLTP	Data Warehouse
Purpose	Run day-to-day operation	Information retrieval and analysis
Structure	RDBMS	RDBMS
Data Model	Normalized	Multi-dimensional
Access	SQL	SQL plus data analysis extensions
Type of Data	Data that run the business	Data that analyses the business
Condition of Data	Changing incomplete	Historical descriptive

The data warehouse server a different purpose from that of OLTP systems by allowing business analysis queries to be answered as opposed to "simple aggregation" such as 'what is the current account balance for this customer?' Typical data warehouse queries include such things as 'which product line sells best in middle America and how dose this correlate to demographic data?

Processes in Data Warehousing OLTP

The first step in data warehousing is to "insulate" your current operational information, i.e. to preserve the security and integrity of mission- critical OLTP applications, while giving you access to the broadest possible base of data. The resulting database or data warehouse may consume hundred of gigabytes-or even terabytes of disk space, what is required than are capable efficient techniques for storing and retrieving massive amounts of information. Increasingly, large organizations have found that only parallel processing systems offer sufficient bandwidth.

The data warehouse thus retrieves data from a varsity of heterogeneous operational database. The data is than transformed and delivered to the data warehouse/ store based in a selected modal (or mapping definition). The data transformation and movement processes are completed whenever an update to the warehouse data is required so there should some from of automation to manage and carry out these functions. The information that describes the modal metadata is the means by which the end user finds and understands the data in the warehouse and is an important part of the warehouse. The metadata should at the very least contain:

- Structure of the data;
- Algorithm used for summarization;

- Mapping from the operational environment to the data warehouse.

Data cleansing is an important viewpoint of creating an efficient data warehouse of creating an efficient data warehouse in that is the removal of creation aspects Operational data such as low level transaction information which sloe down the query times. The cleansing stage has to be as dynamic as possible to accommodate all types of queries even those, which may require low-level information. Data should be extracted from production sources at regular interval and pooled centrally but the cleansing process has to remove duplication and reconcile differences between various styles of data collection.

Once the data has been cleaned it is than transfer to the data warehouse, which typically is a large database on a high performance box, either SMP Symmetric Multi- Processing or MPP, Massively parallel Processing Number crunching power is another importance aspect of data warehousing because of the complexity involved in processing adhoc queries and because of the vast quantities of data that the organization want to use in the warehouse. A data warehouse can be used in different ways, for example it can be a central store against which the queries are run of it can be used like a data mart, data mart which are small warehouses can be established to provide subsets of the main store and summarized information depending on the requirements of a specific group/ department. The central stores approach generally uses every simple data structures with very little assumptions about the relationships between data where as marts often uses multidimensional data base which can speed up query processing as they can have data structures which reflect the most likely questions.

Many vendors have products that provide on the more of the above data warehouse functions. However, it can take a significant amount of work and specialized programming to provide the interoperability needed between products form. Multiple vendors to enable them to perform the required data warehouse processes a typical implementation usually involves a mixture of procedure forma verity of suppliers.

Another approach to data warehousing is the Parsaye Sandwich paradigm put forward by Dr. Kamran Parsaye , CED of information discovery, Hermosa beach. This paradigm or philosophy encourages acceptance of the probability that the first iteration of data warehousing effort will require considerable revision. The Sandwich paradigm advocates the following approach.

- Pre-mine the data to determine what formats and data are needed to support a data- mining application;
- Build a prototype mini- data warehouse i.e. the, the meat of sandwich most of features envisaged for the and product;
- Revise the strategies as necessary;
- Build the final warehouse.

What is OLAP?

- Relational databases are used in the areas of operations and control with emphasis on transaction processing.
- Recently relational databases are used for building data warehouses, which stores tactical information (< 1 year into the future) that answers who and what questions.
- In contrast OLAP uses Multi-Dimensional (MD) views of aggregate data to provide access strategic information.
- OLAP enables users to gain insight to a wide variety of possible views of information and transforms raw data to reflect the enterprise as understood by the user e.g. Analysts, managers and executives.
- In addition to answering who and what questions OLAPs can answer "what if " and "why".
- Thus OLAP enables strategic decision-making.
- OLAP calculations are more complex than simply summing data.
- However, OLAP and Data Warehouses are complementary
- The data warehouse stores and manages data while the OLAP transforms this data into strategic information.

Who uses OLAP and WHY?

- OLAP applications are used by a variety of the functions of an organisation.
- Finance and accounting:

  Budgeting

  Activity-based costing

  Financial performance analysis

  And financial modelling
- Sales and Marketing

  Sales analysis and forecasting

  Market research analysis

  Promotion analysis

  Customer analysis

  Market and customer segmentation
- Production

  Production planning

  Defect analysis

Thus, OLAP must provide managers with the information they need for effective decision-making. The KPI (key performance indicator) of an OLAP application is to provide just-in-time (JIT) information for effective decision-making. JIT information reflects complex data relationships and is calculated on the fly. Such an approach is only practical if the response times are always short The data model must be flexible and respond to changing business requirements as needed for effective decision making.

In order to achieve this in widely divergent functional areas OLAP applications all require:

MD views of data

Complex calculation capabilities

Time intelligence

Multi-Dimensional Views

- MD views inherently represent actual business models, which normally have more than three dimensions e.g., Sales data is looked at by product, geography, channel and time.

- MD views provide the foundation for analytical processing through flexible access to information.
- MD views must be able to analyse data across any dimension at any level of aggregation with equal functionality and ease and insulate users from the complex query syntax
- What ever the query is they must have consistent response times.
- Users queries should not be inhibited by the complex to form a query or receive an answer to a query.
- The benchmark for OLAP performance investigates a server's ability to provide views based on queries of varying complexity and scope.

  Basic aggregation on some dimensions

  More complex calculations are performed on other dimensions
  - Ratios and averages
  - Variances on sceneries
  - A complex model to compute forecasts
- Consistently quick response times to these queries are imperative to establish a server's ability to provide MD views of information.

Benefits of OLAP

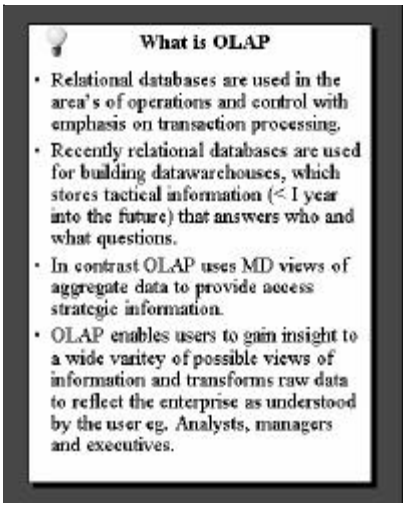
- Increase the productivity of manager's developers and whole organisations.
- Users of OLAP systems become more self-sufficient eg managers no longer depend on IT to make schema changes.
- It allows managers to model problems that would be impossible with less flexible systems
- Users have more control and timely access to relevant strategic information which results in better decision making.(timeliness, accuracy and relevance)
- IT developers also benefit form using OLAP specific software as they can deliver applications to users faster.
- Thus, reducing the application backlog and ensure a better service.
- OLAP further reduces the backlog by making its users self-sufficient to build their own models but yet not relinquishing control over the integrity of the data
- OLAP software reduces the query load and network traffic on OLTP systems and data warehouses.
- Thus, OLAP enables organisations as a whole to respond more quickly to market demands, which often results in increased revenue and profitability. The goal of every organisation.

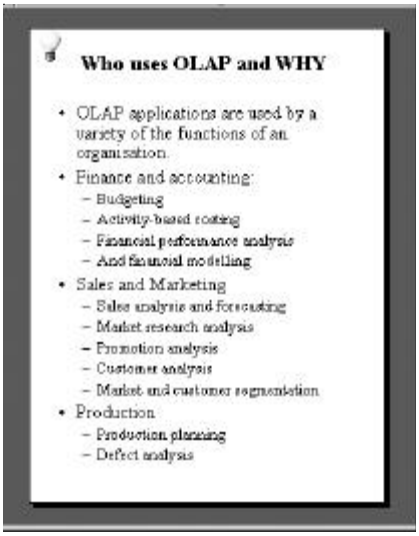
Discussions

- Write short notes on:
  - Multi-Dimensional Views
  - Operational Systems
- What is the significance of an OLTP System?
- Discuss OLTP related processes used in a Data warehouse.
- Explain MD views with an example.
- Identify various benefits of OLTP.
- "OLAP enables organisations as a whole to respond more quickly to market demands, which often results in increased revenue and profitability". Comment.
- Who are the primary users of Online Transaction Processing System?
- "The KPI (key performance indicator) of an OLAP application is to provide just-in-time (JIT) information for effective decision-making". Explain.

References

1. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
2. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
3. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
4. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.





Data warehousing and OLAP systems

	OLAP	Data Warehouse
Purpose	Run day-to-day operations	Information retrieval and analysis
Structure	RDBMS	RDBMS
Data Model	Normalized	Multi-dimensional
Access	SQL	SQL plus data analysis extensions
Types of Data	Data that runs the business	Data that analyses the business
Condition of data	Changing.	Historical, Descriptive.

Benefits of OLAP

- Increase the productivity of managers developers and whole organisations.
- Users of OLAP systems become more self-sufficient eg managers no longer depend on IT to make schema changes.
- It allows managers to model problems that would be impossible with less flexible systems
- Users have more control and timley access to relevant strategic information which results in better decision making.(timeleness, accuracy and relevance)
- IT developers also benefit form using OLAP specific software as they can deliver applications to users faster.

Data Warehouse vs. Operational DBMS

- OLTP (on-line transaction processing)
  - Major task of traditional relational DBMS
  - Day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.
- OLAP (on-line analytical processing)
  - Major task of data warehouse system
  - Data analysis and decision making
- Distinct features (OLTP vs. OLAP):
  - User and system orientation: customer vs. market
  - Data contents: current, detailed vs. historical, consolidated
  - Database design: ER + application vs. star + subject
  - View: current, local vs. evolutionary, integrated
  - Access patterns: update vs. read-only but complex queries

References

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs.. SIGMOD'99.
- S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997.
- G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB'2001
- J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.

Who uses OLAP and Why

- Thus OLAP must provide managers with the information they need for effective decision making.
- The KPI (key performance indicator) of an OLAP application is to provide just-in-time(JIT) information for effective decision making.
- JIT information reflects complex data relationships and is calculated on the fly.
- Such an approach is only practical if the response times are always short
- The data model must be flexible and respond to changing business requirements as needed for effective decision making.

LESSON 4
DATA WAREHOUSING MODELS

Structure

- Introduction
- Objective
- The Date warehouse Model
- Data Modeling for Data Warehouses
- Multidimensional models
- Roll-up display
- A drill-down display
- Multidimensional Schemas
- Star Schema
- Snowflake Schema

Objective

The main objective of this lesson is to make you understand a data warehouse model. It also explains various types of multidimensional models and Schemas.

Introduction

Data warehousing is the process of extracting and transforming operational data into informational data and loading it into a central data store or warehouse. Once the data is loaded it is accessible via desktop query and analysis tools by the decision makers.

The Data Warehouse Model

The data warehouse model is illustrated in the following diagram.

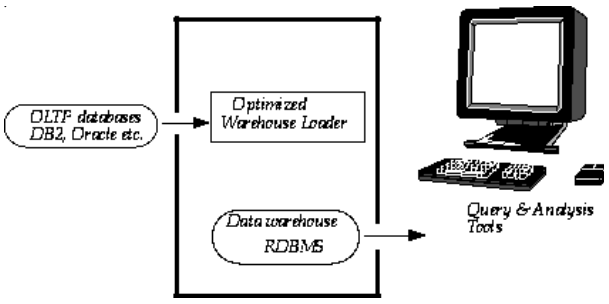


Figure 1: A data warehouse model

The data within the actual warehouse itself has a distinct structure with the emphasis on different levels of summarization as shown in the figure below.

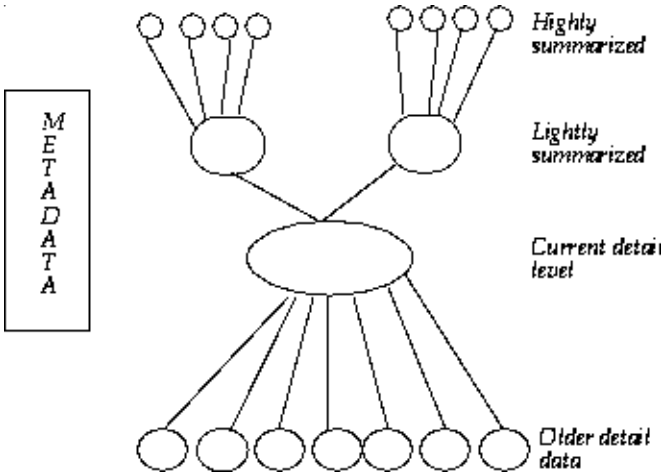


Figure 2: The structure of data inside the data warehouse

The current detail data is central in importance as it:

- Reflects the most recent happenings, which are usually the most interesting;
- It is voluminous as it is stored at the lowest level of granularity;
- it is always (almost) stored on disk storage which is fast to access but expensive and complex to manage.

Older detail data is stored on some form of mass storage, it is infrequently accessed and stored at a level detail consistent with current detailed data.

Lightly summarized data is data distilled from the low level of detail found at the current detailed level and generally is stored on disk storage. When building the data warehouse have to consider what unit of time is summarization done over and also the contents or what attributes the summarized data will contain.

Highly summarized data is compact and easily accessible and can even be found outside the warehouse.

Metadata is the final component of the data warehouse and is really of a different dimension in that it is not the same as data drawn from the operational environment but is used as:

- a directory to help the DSS analyst locate the contents of the data warehouse,
- a guide to the mapping of data as the data is transformed from the operational environment to the data warehouse environment,
- a guide to the algorithms used for summarization between the current detailed data and the lightly summarized data and the lightly summarized data and the highly summarized data, etc.

The basic structure has been described but Bill Inmon fills in the details to make the example come alive as shown in the following diagram.

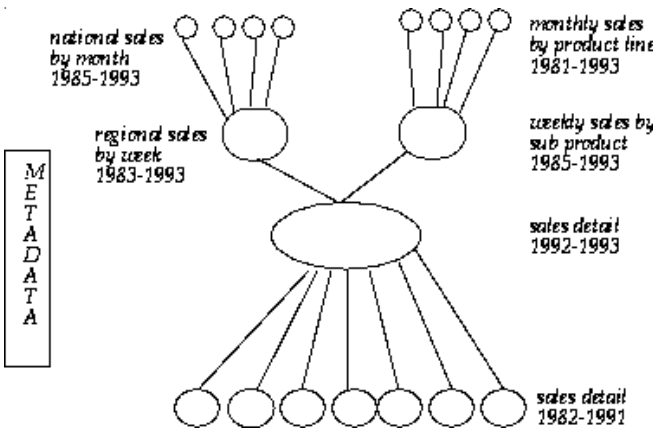


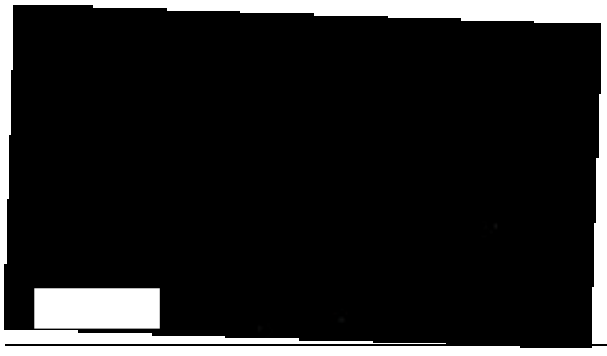
Figure 3: An example of levels of summarization of data inside the data warehouse

The diagram assumes the year is 1993 hence the current detail data is 1992-93. Generally sales data doesn't reach the current level of detail for 24 hours as it waits until it is no longer available to the operational system i.e. it takes 24 hours for it to get to the data warehouse. Sales details are summarized weekly by subproduct and region to produce the lightly summarized detail. Weekly sales are then summarized again to produce the highly summarized data.

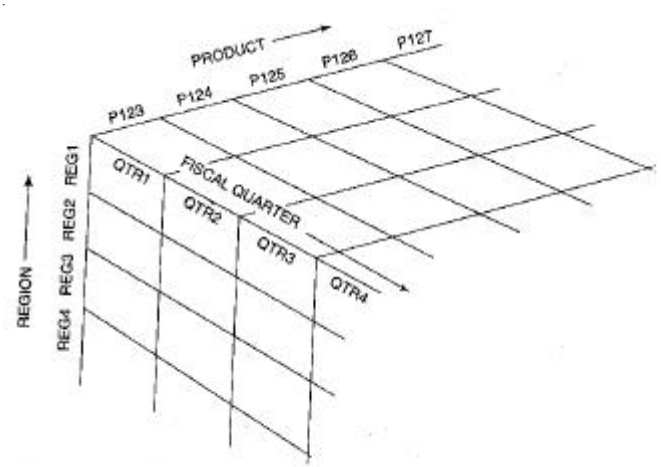
Data Modeling for Data Warehouses

Multidimensional models take advantage of inherent relationships in data to populate data in multidimensional matrices called data cubes. (These may be called hypercube if they have more than three dimensions.) For data that lend themselves to dimensional Formatting, query performance in multidimensional matrices can be much better than in the relational data model. Three examples of dimensions in a corporate data warehouse would be the corporation's fiscal periods, products, and regions.

A standard spreadsheet is a **two-dimensional matrix.** One example would be a spreadsheet of regional sales by product for a particular time period. Products could be shown as rows, with sales revenues for each region comprising the columns. Adding a time dimension, such as an organiza-tion's fiscal quarters, would produce a three-dimensional matrix, which could be repre-sented using a data cube.



In the figure, there is a three-dimensional data cube that organizes product sales data by fiscal quarters and sales regions. Each cell could contain data for a specific prod-uct, specific fiscal quarter, and specific region. By including additional dimensions, a data hypercube could be produced, although more than three dimensions cannot be easily visualized at all or presented graphically. The data can be queried directly in any combination of dimensions, by passing complex database queries. Tools exist for viewing data Data Modeling for Data Warehouses



Pivoted version of the data cube.

According to the user's choice of dimensions. Changing from one dimensional hierarchy -(orientation) to another is easily accomplished in a data cube by a technique called pivoting (also called rotation). In this technique, the data cube can be thought of as rotating to show a different orientation of the axes. For example, you might pivot the data cube to show regional sales revenues as rows, the fiscal quarter revenue totals as columns, and company's products in the third dimension. Hence, this technique is equivalent to having a regional sales table for each product separately, where each table shows quarterly sales for that product region by region.

Multidimensional models lend themselves readily to hierarchical views in what is known **as roll-up display and drill-down display.**

- **Roll-up display** moves up the hierar-chy, grouping into larger units along a dimension (e.g., summing weekly data by quar-ter, or by year). One of the above figures shows a roll-up display that moves from individual products to a coarser grain of product categories.
- A **drill-down display** pro-vides the opposite capability, furnishing a finer-grained view, perhaps disaggregating country sales by region and then regional sales by sub region and also breaking up prod-ucts by styles.

The multidimensional storage model involves two types of tables: dimension tables and fact tables. A dimension table consists of tuples of attributes of the dimension. A fact table can be thought of as having tuples, one per a recorded fact. This fact contains some measured or observed variable(s) and identifies it (them) with pointers to dimension tables. The fact

table contains the data and the dimensions identify each tuple in that data. An example of a fact table that can be viewed from the perspec-tive of multiple dimensional tables.

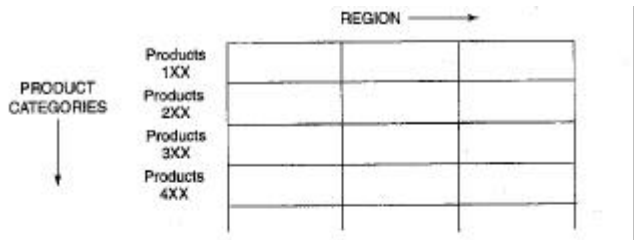


Figure 26.5 The roll-up operation.

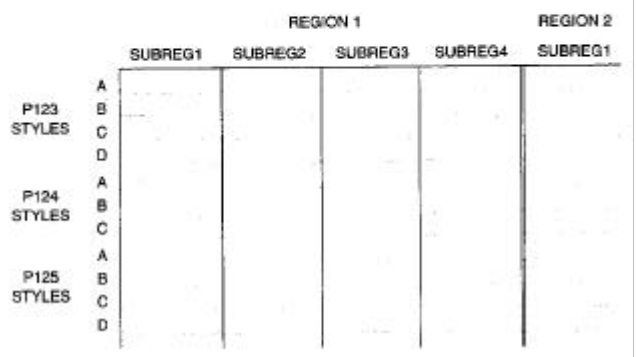


Figure 26.6 The drill-down operation.

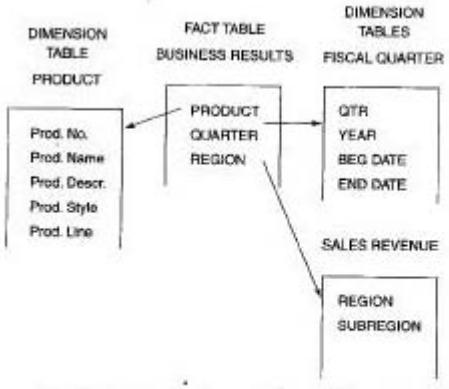


Figure 26.7 A star schema with fact and dimensional tables.

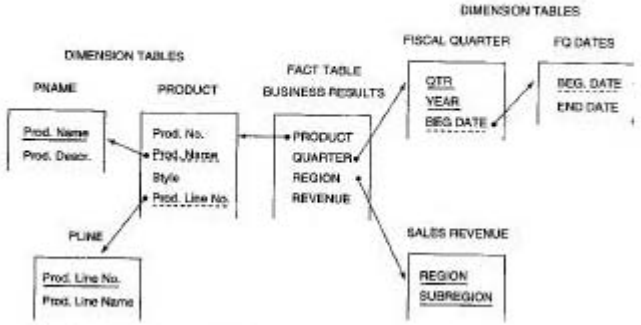


Figure 26.8 A snowflake schema.

Two common multidimensional schemas are the **star schema and the snowflake schema**.

The **star schema** consists of a fact table with a single table for each dimension.

The **snowflake schema** is a variation on the star schema in which the dimensional tables from a star schema are organized into a hierarchy by normalizing them. Some installations are normalizing data warehouses up to the third normal form so that they can access the data warehouse to the finest level of detail. A fact con-stellation is a set of fact tables that share some dimension tables. Following figure shows a fact constellation with two fact tables, business results arid business forecast. These share the dimension table called product. Fact constella-tions limit the possible queries for the ware-house.

Data warehouse storage also utilizes indexing techniques to support high perfor-mance access. A technique called bitmap indexing constructs a bit vector for each value in a domain (column) being indexed.

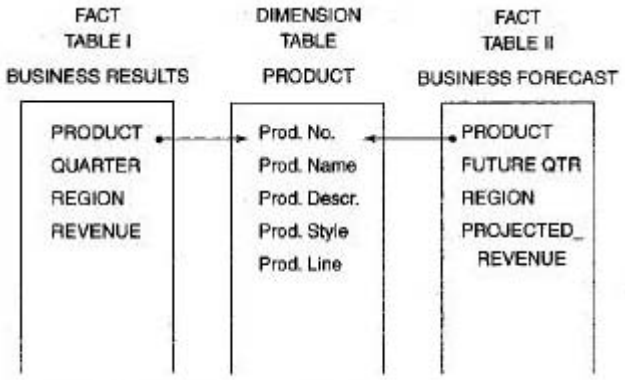


Figure 26.9 A fact constellation.

It works very well for domains of low-cardinality. There is a 1 bit placed in the jth position in the vector if the jth row contains the value being indexed. For example, imagine an inventory of 100,000 cars with a bitmap index on car size. If there are four-car sizes--economy, compact, midsize, and full size-there will be four bit vectors, each containing 100,000 bits (12.5 K) for a total index size of 50K. Bitmap indexing can provide consider-able input/output and storage space advantages in low-cardinality domains. With bit vec-tors a bitmap index can provide dramatic improvements in comparison, aggregation, and join performance. In a star schema, dimensional data can be indexed to tuples in the fact table by join indexing. Join indexes are traditional indexes to maintain relationships between primary key and foreign key values. They relate the values of a dimension of a star schema to rows in the fact table. For example, consider a sales fact table that has city and fiscal quarter as dimensions. If there is a join index on city, for each city the join index maintains the tuple IDs of tuples containing that city. Join indexes may involve multiple dimensions.

Data warehouse storage can facilitate access to summary data by taking further advantage of the nonvolatility of data warehouses and a degree of predictability of the analyses that will be performed using them. Two approaches have been used.(1) smaller tables including summary data such as quarterly sales or revenue by product line, and (2) encoding of level (e.g., weekly, quarterly, annual) into existing tables. By comparison, the overhead of creating and maintaining such aggregations would likely be excessive in a volatile, transaction-oriented database.

Discussions

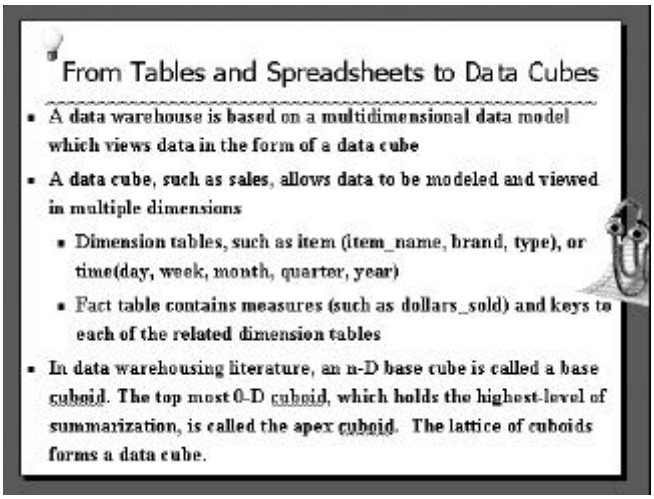
- What are the various kinds of models used in Data warehousing?
- Discuss the following:
  - Roll-up display
  - Drill down operation
  - Star schema
  - Snowflake schema
- Why is the *star schema* called by that name?
- State an advantage of the multidimensional database structure over the relational database structure for data warehousing applications.
- What is one reason you might choose a relational structure over a multidimensional structure for a data warehouse database? .
- Clearly contrast the difference between a fact table and a dimension table.

Exercises

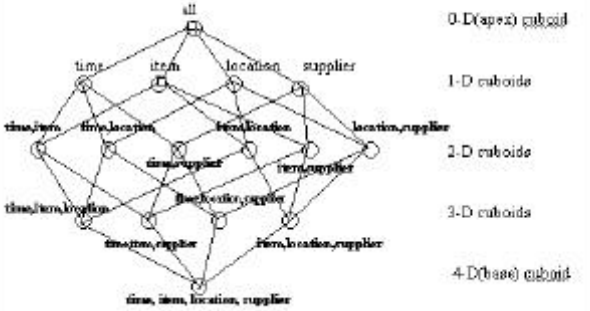
1. Your college or university is designing a data warehouse to enable deans, department chairs, and the registrar's office to optimize course offerings, in terms of which courses are offered, in how many sections, and at what times. The data warehouse planners hope they will be able to do this better after examining historical demand for courses and extrapolating any trends that emerge.
   a. Give three dimension data elements and two fact data elements that could be in the database for this data warehouse. Draw a data cube, for this database.
   b. State two ways in which each of the two fact data elements could be of low quality in some respect.
2. You have decided to prepare a budget for the next 12 months based on your actual expenses for the past 12. You need to get your expense information into what is in effect a data warehouse, which you plan to put into a spreadsheet for easy sorting and analysis.
   a. What are your information sources for this data warehouse?
   b. Describe how you would carry out each of the five steps of data preparation for a data warehouse database, from extraction through summarization. If a particular step does not apply, say so and justify your statement.

References

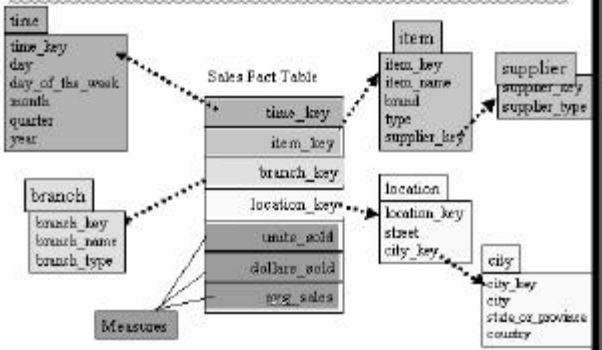
1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
4. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
5. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.



Cube: A Lattice of Cuboids



Example of Snowflake Schema

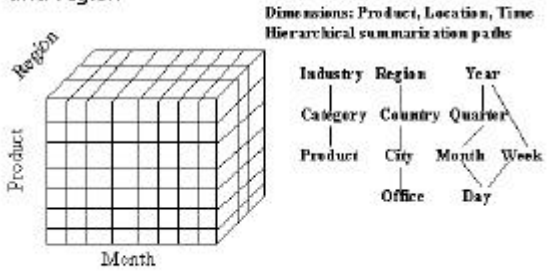


Conceptual Modeling of Data Warehouses

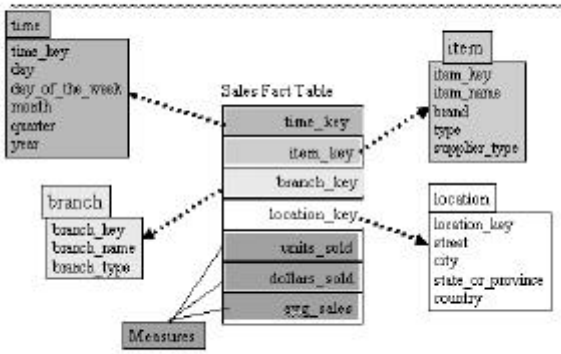
- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

Multidimensional Data

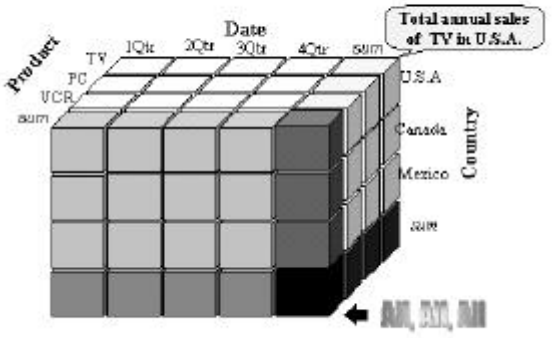
- Sales volume as a function of product, month, and region



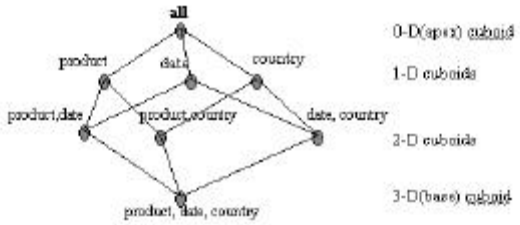
Example of Star Schema



A Sample Data Cube



Cuboids Corresponding to the Cube



References

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs.. SIGMOD'99.
- S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997.
- G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB'2001
- J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.

Notes

LESSON 5
ARCHITECTURE AND PRINCIPLES OF DATA WAREHOUSING

Structure

- Objective
- Introduction
- Structure of a Data warehouse
- Data Warehouse Physical Architectures
- Generic Two-Level
- Expanded Three-Level
- Enterprise data warehouse (EDW)
- Data marts
- Principles of a Data warehousing

Objective

The objective of this lesson is to let you know the basic structure of a Data warehouse. You will also learn about Data warehouse physical architecture and various principles of a Data warehousing.

Introduction

Let me start the lesson with an example, which illustrates the importance and need of a data warehouse. Until several years ago Coca Cola had no idea how many bottles of Coke it produced each day because production data were stored on 24 different computer systems. Then, it began a technique called Data warehousing. One airline spent and wasted over $100 million each year on inefficient mass media advertising campaigns to reach frequent flyers…then it began data warehousing. Several years ago, the rail industry needed 8 working days to deliver a freight quote to a customer. The trucking industry, by contrast, could deliver a freight quote to a customer on the phone instantly, because unlike the rail industry, truckers were using…data warehousing.

A data warehouse is a data base that collects current information, transforms it to ways it can be used by the warehouse owner, transforms that information for clients, and offers portals of access to members of your firm to help them make decisions and future plans.

Data warehousing is the technology trend most often associated with enterprise computing today. The term conjures up images of vast data banks fed from systems all over the globe, with legions of corporate analysts mining them for golden nuggets of information that will make their companies more profitable.

All of the developments in database technology over the past 20 years have culminated in the data warehouse. Entity-relationship modeling, heuristic searches, mass data storage, neural networks, multiprocessing, and natural-language interfaces have all found their niches in the data warehouse. But aside from being a database engineer's dream, what practical benefits does a data warehouse offer the enterprise?

When asked, corporate executives often say that having a data warehouse gives them a competitive advantage, because it gives them a better understanding of their data and a better understanding of their business in relation to their competitors, and it lets them provide better customer service.

So, what exactly is a data warehouse? Should your company have one, and if so, what should it look like?

Structure of a Data Warehouse

Essentially, a data warehouse provides historical data for decision-support applications. Such applications include reporting, online analytical processing (OLAP), executive information systems (EIS), and data mining.

According to W. H. Inmon, the man who originally came up with the term, a data warehouse is a centralized, integrated repository of information. Here, integrated means cleaned up, merged, and redesigned. This may be more or less complicated depending on how many systems feed into a warehouse and how widely they differ in handling similar information.

But most companies already have repositories of information in their production systems and many of them are centralized. Aren't these data warehouses? Not really.

Data warehouses differ from production databases, or online transaction-processing (OLTP) systems, in their purpose and design. An OLTP system is designed and optimized for data entry and updates, whereas a data warehouse is optimized for data retrieval and reporting, and it is usually a read-only system. An OLTP system contains data needed for running the day-to-day operations of a business but a data warehouse contains data used for analyzing the business. The data in an OLPT system is current and highly volatile, which data elements that may be incomplete or unknown at the time of entry. A warehouse contains historical, nonvolatile data that has been adjusted for transactions errors. Finally, since their purposes are so different, OLPT systems and data warehouses use different data-modeling strategies. Redundancy is almost nonexistent in OLTP systems, since redundant data complicates updates. So OLPT systems are highly normalized and are usually based on a relational model. But redundancy is desirable in a data warehouse, since it simplifies user access and enhances performance by minimizing the number of tables that have to be joined. Some data warehouses don't use a relational model at all, preferring a multidimensional design instead.

To discuss data warehouses and distinguish them from transactional databases calls for an appropriate data model. The multidimensional data model is a good fit for OLAP and decision-support technologies. In contrast to multi-databases, which provide access to disjoint and usually heterogeneous databases, a data warehouse is frequently a store of integrated data from multiple sources, processed for storage in a multidimensional model. Unlike most transactional databases, data warehouses typically support time-series and trend analysis, both of which requires more historical data than are generally

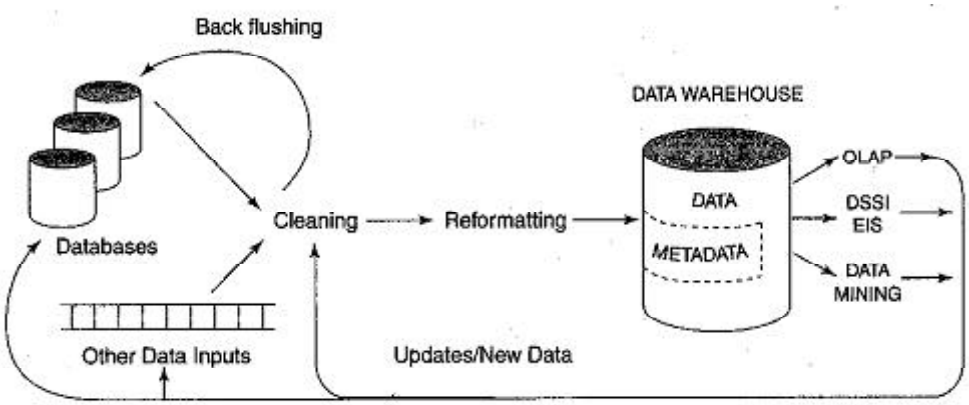
maintained in transactional databases. Compared with transactional databases, data warehouses are nonvolatile. That means that information in the data warehouse changes far less often and may be regarded as non-real-time with periodic updating. In transactional systems, transactions are the unit and are the agent of change a database; by contrast, data warehouse information is much more coarse grained and is refreshed according to a careful choice of refresh policy, usually incremental. Warehouse updates are handled by the warehouse's acquisition component that provides all required preprocessing.

We can also describe data warehousing more generally as "a collection of decision support technologies, aimed at enabling the knowledge worker (executive, manager, ana-lyst) to make, better and faster decisions." The following Figure gives an overview of the conceptual structure of a data warehouse. It shows the entire data warehousing process. This process includes possible cleaning and reformatting of data before it's warehousing. At the end of the process, OLAP, data mining, and DSS may generate new relevant information such as rules; this information is shown in the figure going back into the warehouse figure also shows that data sources may include files.

Data Warehousing, OnLine Analytical Processing (OLAP) and Decision Support Systems - apart from being buzz words of today IT arena - are the expected result of IT systems and current needs. For decades, Information Management Systems have focused exclusively in the gathering and recording into Database Management Systems data that corresponded to everyday simple transactions, from which the name OnLine Transaction Processing (OLTP) comes from.

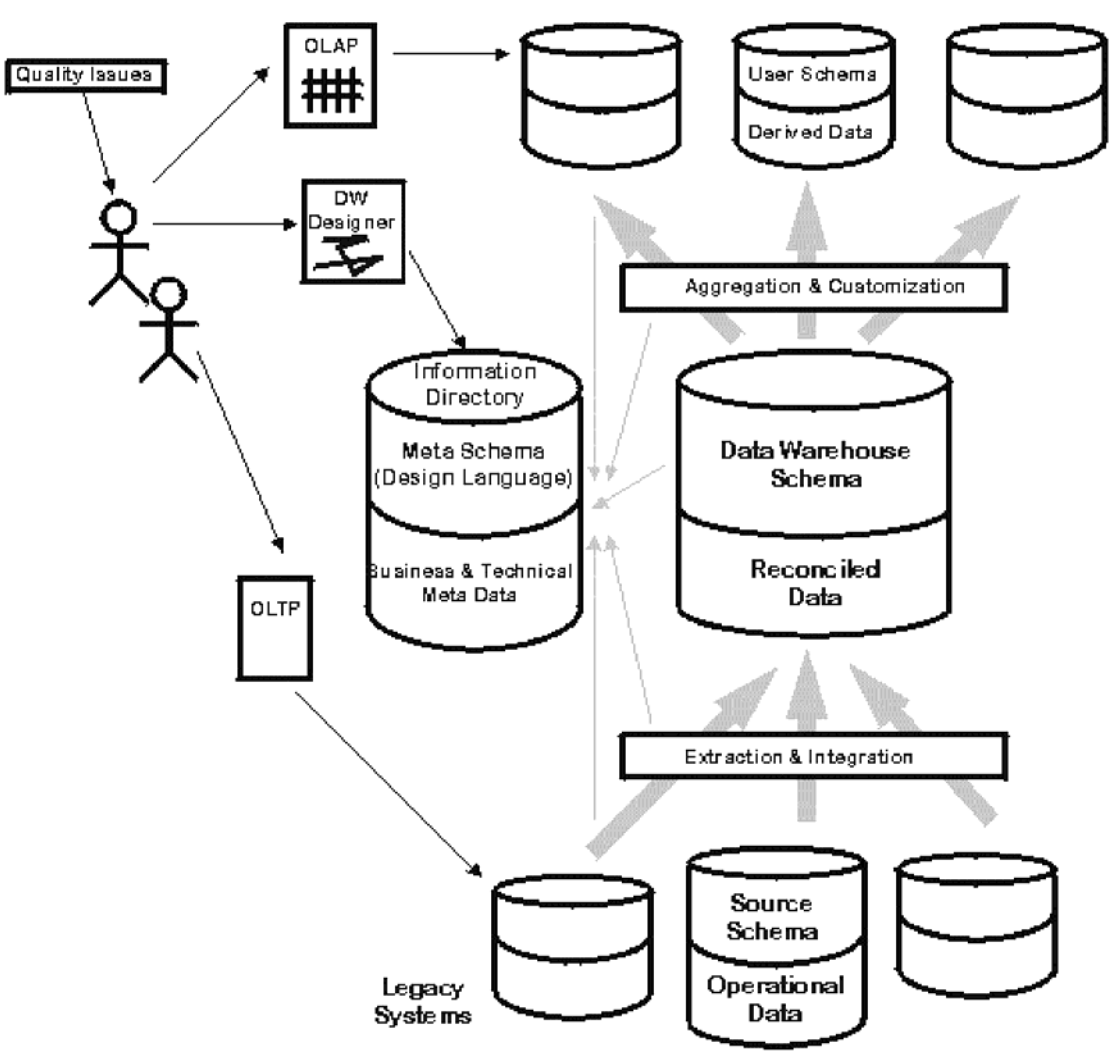
Managers and analysts now need to go steps further from the simple data storing phase and exploit IT systems by posing complex queries and requesting analysis results and decisions that are based on the stored data. Here is where OLAP and Data Warehousing is introduced, bringing into business the necessary system architecture, principles, methodological approach and - finally - tools to assist in the presentation of functional Decision Support Systems.

I.M.F. has been working closely with the academic community - which only recently followed up the progress of the commercial arena that was boosting and pioneering in the area for the past decade - and adopted the architecture and methodology presented in the following picture. This is the result of the ESPRIT funded Basic Research project, "Foundations of Data Warehouse Quality - DWQ".



The overall process of data warehousing.

3. Chaudhuri and Dayal (1997) provide an excellent tutorial on the topic, with this as a starting definition.



Being basically dependent on architecture in concept, a Data Warehouse - or an OLAP system - is designed by applying data warehousing concepts on traditional database systems and using appropriate design tools. Data Warehouses and OLAP applications designed and implemented comply with the adopted methodology by IMF.

The final deployment takes place through the use of specialized data warehouse and OLAP systems, namely MicroStrategy's DSS Series. MicroStrategy Inc. is one of the most prominent and accepted international players on data warehousing systems and tools, offering solutions for every single layer of the DW architecture hierarchy.

Data Warehouse Physical Architectures

- Generic Two-Level
- Expanded Three-Level
- Enterprise data warehouse (EDW) - single source of data for decision making
- Data marts - limited scope; data selected from EDW

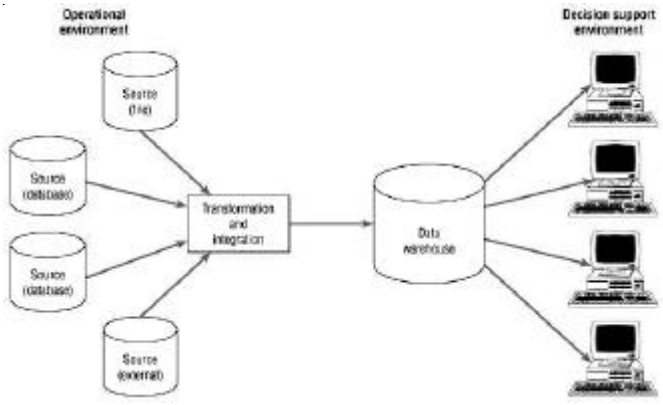


Fig : Generic Two-Level Physical Architecture

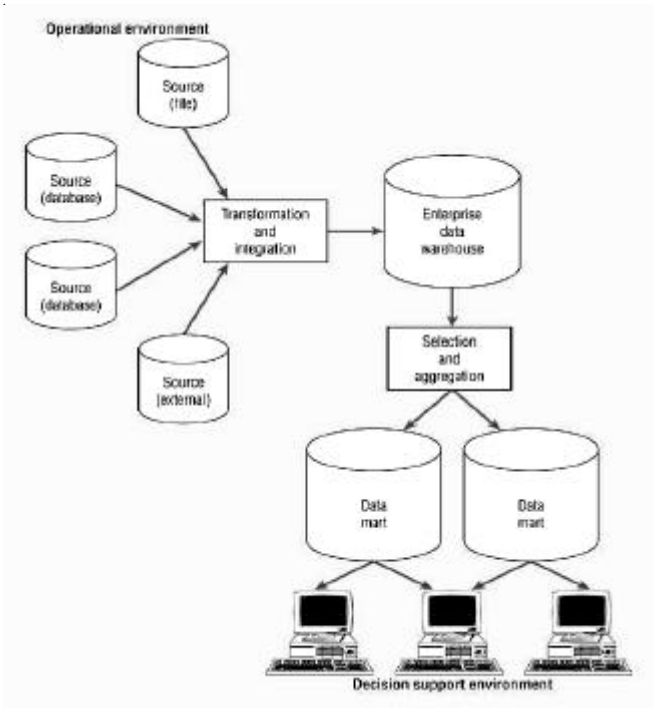


Fig : Expanded Three-Level Physical Architecture

Associated with the three-level physical architecture

- Operational Data

  Stored in the various operational systems throughout the organization

- Reconciled Data

  The data stored in the enterprise data warehouse

  Generally not intended for direct access by end users

- Derived Data

  The data stored in the data marts

  Selected, formatted, and aggregated for end user decision-support applications

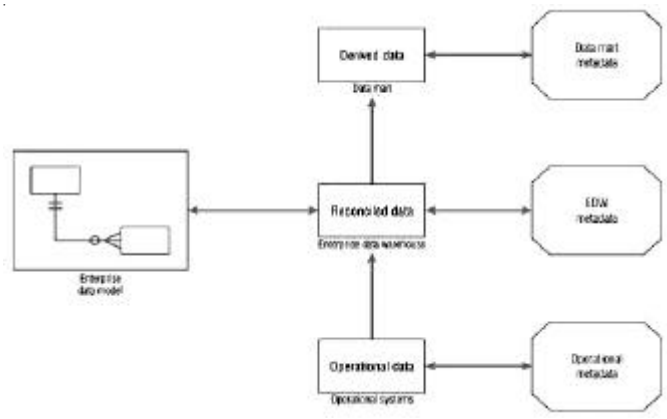


Fig : Three-Layer Data Architecture

Principles of a Data Warehousing

- **Load Performance**

  Data warehouses require increase loading of new data on a periodic basic within narrow time windows; performance on the load process should be measured in hundreds of millions of rows and gigabytes per hour and must not artificially constrain the volume of data business.

- **Load Processing**

  Many steps must be taken to load new or update data into the data warehouse including data conversion, filtering, reformatting, indexing and metadata update.

- **Data Quality Management**

  Fact-based management demands the highest data quality. The warehouse must ensure local consistency, global consistency, and referential integrity despite "dirty" sources and massive database size.

- **Query Performance**

  Fact-based management must not be slowed by the performance of the data warehouse RDBMS; large, complex queries must be complete in seconds not days.

- **Terabyte Scalability**

  Data warehouse sizes are growing at astonishing rates. Today these range from a few to hundreds of gigabytes and terabyte-sized data warehouses.

Discussions

- Write short notes on:
  - Data Quality Management
  - OLAP
  - DSS
  - Data marts
  - Operational data
- Discuss Three-Layer Data Architecture with the help of a diagram.
- What are the various Principles of Data warehouse?
- What is the importance of a data warehouse in any organization? Where it is required?

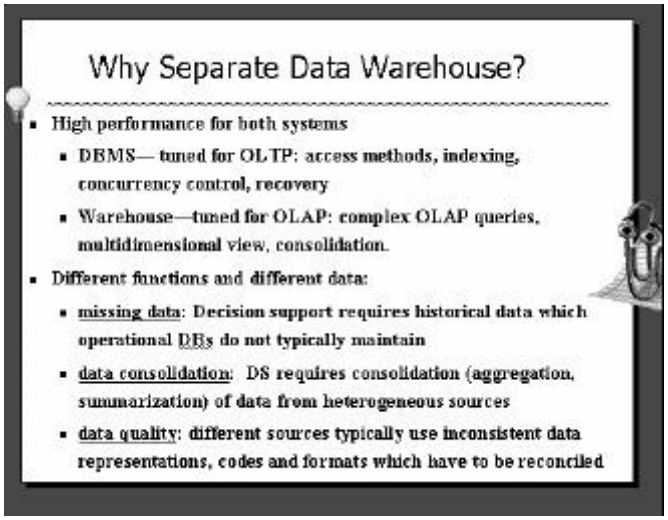
Self Test

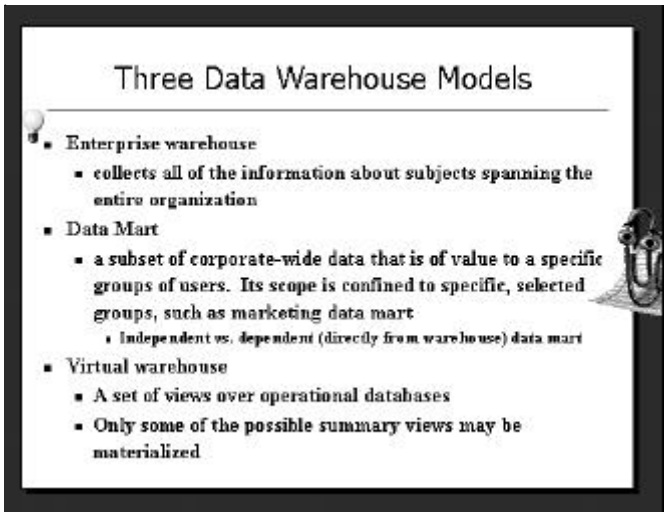
As set of multiple choices is given with every question, Choose the correct answer for the Following questions.

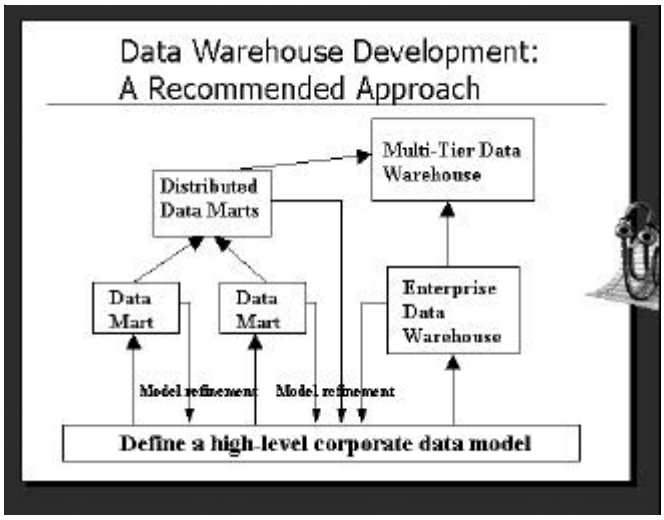
1. Data warehouse cannot deal with
   a. Data analysis
   b. Operational activities
   c. Information extraction
   d. None of these
2. A data warehouse system requires
   a. Only current data
   b. Data for a large period
   c. Only data projections
   d. None of these

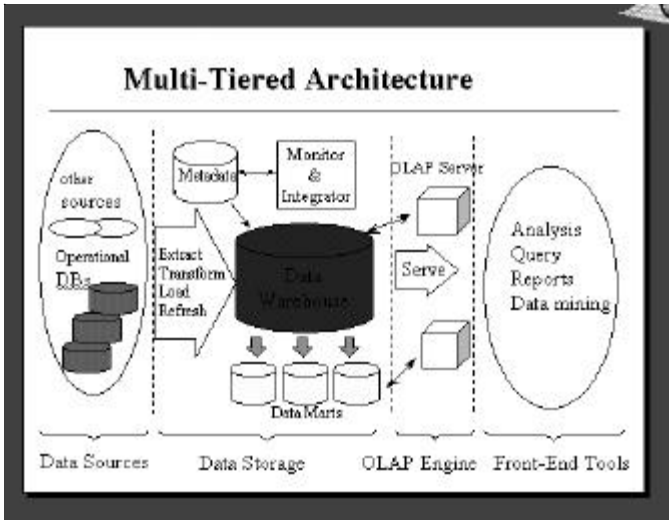
References

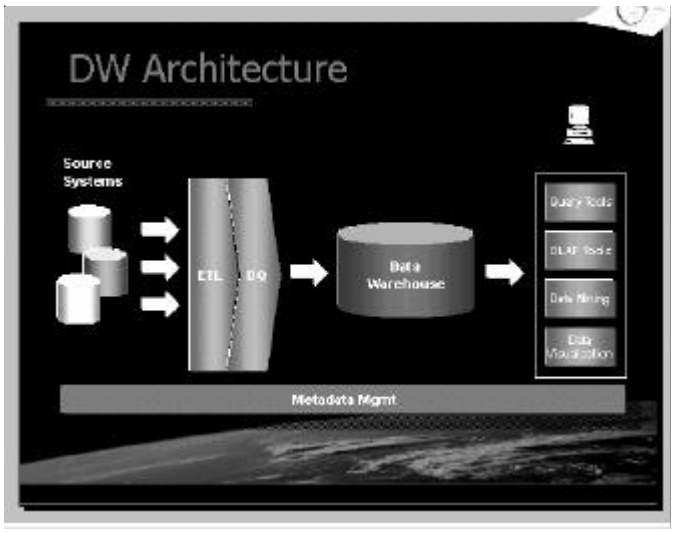
1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
4. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

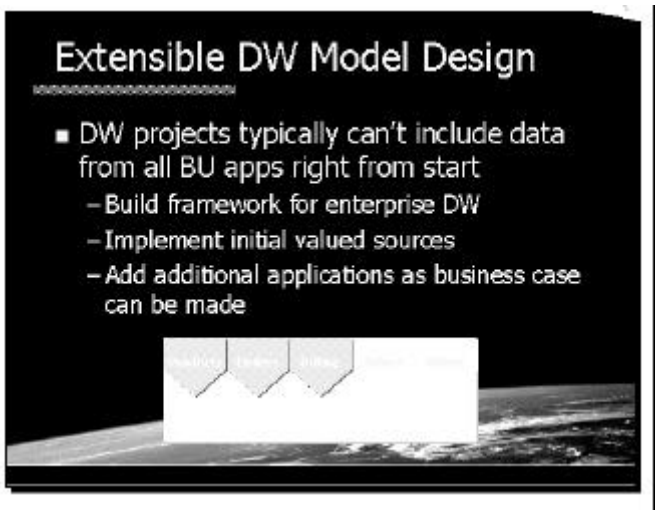




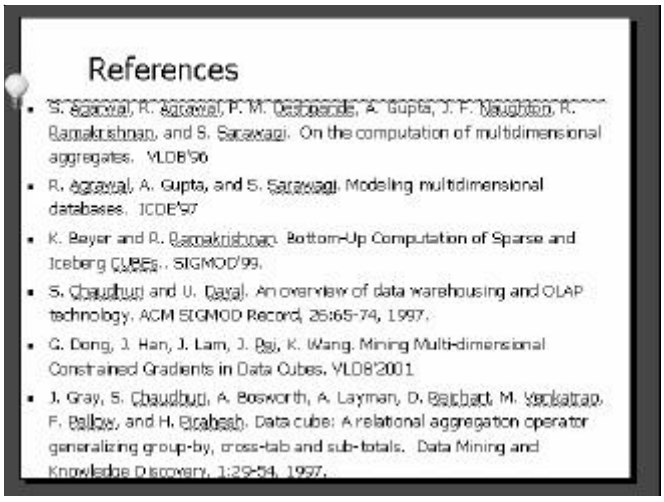







Extensible DW Model Design
DW projects typically can't include data from all BU apps right from start
Build framework for enterprise DW
Implement initial valued sources
Add additional applications as business case can be made


What it is Not
Data warehousing is not a silver bullet
It will not lower your IT costs
It will not allow you to cut resource requirements
It will not fix bad data models, poor system design or data problems


References
S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs. SIGMOD'99.
S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997.
G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB'2001
J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.

Notes

LESSON 6
DATA WAREHOUSING AND OPERATIONAL SYSTEMS

Structure

- Objective
- Introduction
- Operational Systems
- Warehousing" data outside the operational systems
- Integrating data from more than one operational system
- Differences between transaction and analysis processes
- Data is mostly non-volatile
- Data saved for longer periods than in transaction systems
- Logical transformation of operational data
- Structured extensible data model
- Data warehouse model aligns with the business structure
- Transformation of the operational state information

Objective

The aim of this lesson is to explain you the need and importance of an Operational Systems in a Data warehouse.

Introduction

Data warehouse, a collection of data designed to support management decision-making. Data warehouses contain a wide variety of data that present a coherent picture of business conditions at a single point in time.

Development of a data warehouse includes development of systems to extract data from operating systems plus installation of a warehouse database system that provides managers flexible access to the data.

The term data warehousing generally refers to the combination of many different databases across an entire enterprise.

Operational Systems

Up to now, the early database system of the primary purpose was to, meet the needs of operational systems, which are typically transactional in nature. Classic examples of operational systems include

- General Ledgers
- Accounts Payable
- Financial Management
- Order Processing
- Order Entry
- Inventory

Operational systems by nature are primarily concerned with the handling of a single transaction. Look at a banking system, when you, the customer, make a deposit to your checking account, the banking operational system is responsible for recording the transaction to ensure the corresponding debit appears in your account record.

A typical operational system deals with one order, one account, one inventory item. An operational system typically dea1s with predefined events and, due to the nature of these events, requires fast access. Each transaction usually deals with small amounts of data.

Most of the time, the business needs of an operational system do not change much. The application that records the transaction, as well as the application that controls access to the information, that is, there porting side of the- banking business does not change much over time. In this type of system, the information required, when a customer initiates a transaction must be current. Before a bank will allow a withdrawal, it must first be certain of your current balance.

"Warehousing" Data outside the Operational Systems

The primary concept of data warehousing is that the data stored for business analysis can most effectively be accessed by separating it from the data in the operational systems. Many of the reasons for this separation have evolved over the years. In the past, legacy systems archived data onto tapes as it became inactive and many analysis reports ran from these tapes or mirror data sources to minimize the performance impact on the operational systems.

These reasons to separate the operational data from analysis data have not significantly changed with the evolution of the data warehousing systems, except that now they are considered more formally during the data warehouse building process. Advances in technology and changes in the nature of business have made many of the business analysis processes much more complex and sophisticated. In addition to producing standard reports, today's data warehousing systems support very sophisticated online analysis including multi-dimensional analysis.

Integrating Data from more than one Operational System

Data warehousing systems are most successful when data can be combined from more than one operational system. When the data needs to be brought together from more than one source application, it is natural that this integration be done at a place independent of the source applications. Before the evolution of structured data warehouses, analysts in many instances would combine data extracted from more than one operational system into a single spreadsheet or a database. The data warehouse may very effectively combine data from multiple source applications such as sales, marketing, finance, and production. Many large data warehouse architectures allow for the source applications to be integrated into the data warehouse incrementally.

The primary reason for combining data from multiple source applications is the ability to cross-reference data from these

applications. Nearly all data in a typical data warehouse is built around the time dimension. Time is the primary filtering criterion for a very large percentage of all activity against the data warehouse. An analyst may generate queries for a given week, month, quarter, or a year. Another popular query in many data warehousing applications is the review of year-on-year activity. For example, one may compare sales for the first quarter of this year with the sales for first quarter of the prior years. The time dimension in the data warehouse also serves as a fundamental cross-referencing attribute. For example, an analyst may attempt to access the impact of a new marketing campaign run during selected months by reviewing the sales during the same periods. The ability to establish and understand the correlation between activities of different organizational groups within a company is often cited as the single biggest advanced feature of the data warehousing systems.

The data warehouse system can serve not only as an effective platform to merge data from multiple current applications; it can also integrate multiple versions of the same application. For example, an organization may have migrated to a new standard business application that replaces an old mainframe-based, custom-developed legacy application. The data warehouse system can serve as a very powerful and much needed platform to combine the data from the old and the new applications. Designed properly, the data warehouse can allow for year-on-year analysis even though the base operational application has changed.

Differences between Transaction and Analysis Processes

The most important reason for separating data for business analysis from the operational data has always been the potential performance degradation on the operational system that can result from the analysis processes. High performance and quick response time is almost universally critical for operational systems. The loss of efficiency and the costs incurred with slower responses on the predefined transactions are usually easy to calculate and measure. For example, a loss of five seconds of processing time is perhaps negligible in and of itself; but it compounds out to considerably more time and high costs once all the other operations it impacts are brought into the picture. On the other hand, business analysis processes in a data warehouse are difficult to predefine and they rarely need to have rigid response time requirements.

Operational systems are designed for acceptable performance for pre-defined transactions. For an operational system, it is typically possible to identify the mix of business transaction types in a given time frame including the peak loads. It also relatively easy to specify the maximum acceptable response time given a specific load on the system. The cost of a long response time can then be computed by considering factors such as the cost of operators, telecommunication costs, and the cost of any lost business. For example, an order processing system might specify the number of active order takers and the average number of orders for each operational hour. Even the query and reporting transactions against the operational system are most likely to be predefined with predictable volume.

Even though many of the queries and reports that are run against a data warehouse are predefined, it is nearly impossible to accurately predict the activity against a data warehouse. The process of data exploration in a data warehouse takes a business analyst through previously undefined paths. It is also common to have runaway queries in a data warehouse that are triggered by unexpected results or by users' lack of understanding of the data model. Further, many of the analysis processes tend to be all encompassing whereas the operational processes are well segmented. A user may decide to explore detail data while reviewing the results of a report from the summary tables. After finding some interesting sales activity in a particular month, the user may join the activity for this month with the marketing programs that were run during that particular month to further understand the sales. Of course, there would be instances where a user attempts to run a query that will try to build a temporary table that is a Cartesian product of two tables containing a million rows each! While an activity like this would unacceptably degrade an operational system's performance, it is expected and planned for in a data warehousing system.

Data is mostly Non-volatile

Another key attribute of the data in a data warehouse system is that the data is brought to the warehouse after it has become mostly non-volatile. This means that after the data is in the data warehouse, there are no modifications to be made to this information. For example, the order status does not change, the inventory snapshot does not change, and the marketing promotion details do not change. This attribute of the data warehouse has many very important implications for the kind of data that is brought to the data warehouse and the timing of the data transfer.

Let us further review what it means for the data to be non-volatile. In an operational system the data entities go through many attribute changes. For example, an order may go through many statuses before it is completed. Or, a product moving through the assembly line has many processes applied to it. Generally speaking, the data from an operational system is triggered to go to the data warehouse when most of the activity on these business entity data has been completed. This may mean completion of an order or final assembly of an accepted product. Once an order is completed and shipped, it is unlikely to go back to backorder status. Or, once a product is built and accepted, it is unlikely to go back to the first assembly station. Another important example can be the constantly changing data that is transferred to the data warehouse one snapshot at a time. The inventory module in an operational system may change with nearly every transaction; it is impossible to carry all of these changes to the data warehouse. You may determine that a snapshot of inventory carried once every week to the data warehouse is adequate for all analysis. Such snapshot data naturally is non-volatile.

It is important to realize that once data is brought to the data warehouse, it should be modified only on rare occasions. It is very difficult, if not impossible, to maintain dynamic data in the data warehouse. Many data warehousing projects have failed miserably when they attempted to synchronize volatile data between the operational and data warehousing systems.

Data saved for longer periods than in transaction systems

Data from most operational systems is archived after the data becomes inactive. For example, an order may become inactive after a set period from the fulfillment of the order; or a bank account may become inactive after it has been closed for a period of time. The primary reason for archiving the inactive data has been the performance of the operational system. Large amounts of inactive data mixed with operational live data can significantly degrade the performance of a transaction that is only processing the active data. Since the data warehouses are designed to be the archives for the operational data, the data here is saved for a very long period.

In fact, a data warehouse project may start without any specific plan to archive the data off the warehouse. The cost of maintaining the data once it is loaded in the data warehouse is minimal. Most of the significant costs are incurred in data transfer and data scrubbing. Storing data for more than five years is very common for data warehousing systems. There are industry examples were the success of a data warehousing project has encouraged the managers to expand the time horizon of the data stored in the data warehouse. They may start with storing the data for two or three years and then expand to five or more years once the wealth of business knowledge in the data warehouse is discovered. The falling prices of hardware have also encouraged the expansion of successful data warehousing projects.

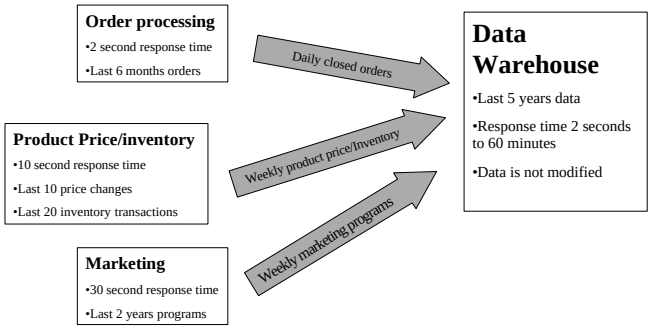


Figure 3. Reasons for moving data outside the operations systems

In short, the separation of operational data from the analysis data is the most fundamental data-warehousing concept. Not only is the data stored in a structured manner outside the operational system, businesses today are allocating considerable resources to build data warehouses at the same time that the operational applications are deployed. Rather than archiving data to a tape as an afterthought of implementing an operational system, data warehousing systems have become the primary interface for operational systems.

Figure 3 highlights the reasons for separation discussed in this section.

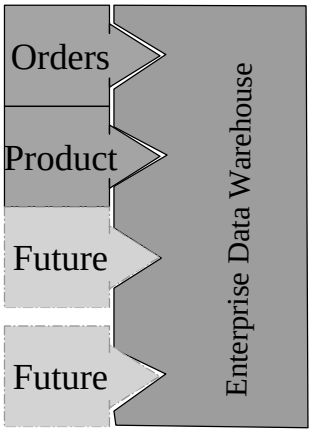
Logical Transformation of Operational Data

This sub-section explores the concepts associated with the data warehouse logical model. The data is logically transformed when it is brought to the data warehouse from the operational systems. The issues associated with the logical transformation of data brought from the operational systems to the data warehouse may require considerable analysis and design effort. The architecture of the data warehouse and the data warehouse model greatly impact the success of the project. This section reviews some of the most fundamental concepts of relational database theory that do not fully apply to data warehousing systems. Even though most data warehouses are deployed on relational database platforms, some basic relational principles are knowingly modified when developing the logical and physical model of the data warehouses.

Structured Extensible Data Model

The data warehouse model outlines the logical and physical structure of the data warehouse. Unlike the archived data of the legacy systems, considerable effort needs to be devoted to the data warehouse modeling. This data modeling effort in the early phases of the data warehousing project can yield significant benefits in the form of an efficient data warehouse that is expandable to accommodate all of the business data from multiple operational applications.

The data modeling process needs to structure the data in the data warehouse independent of the relational data model that may exist in any of the operational systems. As discussed later in this paper, the data warehouse model is likely to be less normalized than an operational system model. Further, the operational systems are likely to have large amounts of overlapping business reference data. Information about current products is likely to be used in varying forms in many of the operational systems. The data warehouse system needs to consolidate all of the reference data. For example, the operational order processing system may maintain the pricing and physical attributes of products whereas the manufacturing floor application may maintain design and formula attributes for the same product. The data warehouse reference table for products would consolidate and maintain all attributes associated with products that are relevant for the analysis processes. Some attributes that are essential to the operational system are likely to be deemed unnecessary for the data warehouse and may not be loaded and maintained in the data warehouse.



•Setup framework for Enterprise data warehouse

•Start with few a most valuable source applications

•Add additional applications as business case can be made

Figure 4. Extensible data warehouse

The data warehouse model needs to be extensible and structured such that the data from different applications can be added as a business case can be made for the data. A data warehouse project in most cases cannot include data from all possible applications right from the start. Many of the successful data warehousing projects have taken an incremental approach to adding data from the operational systems and aligning it with the existing data. They start with the objective of eventually adding most if not all business data to the data warehouse. Keeping this long-term objective in mind, they may begin with one or two operational applications that provide the most fertile data for business analysis. Figure 4 illustrates the extensible architecture of the data warehouse.

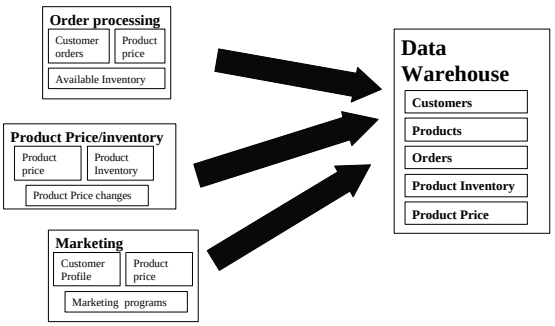
Data Warehouse model aligns with the business structure

A data warehouse logical model aligns with the business structure rather than the data model of any particular application. The entities defined and maintained in the data warehouse parallel the actual business entities such as customers, products, orders, and distributors. Different parts of an organization may have a very narrow view of a business entity such as a customer. For example, a loan service group in a bank may only know about a customer in the context of one or more loans outstanding. Another group in the same bank may know about the same customer in context of a deposit account. The data warehouse view of the customer would transcend the view from a particular part of the business. A customer in the data warehouse would represent a bank customer that has any kind of business with the bank.

The data warehouse would most likely build attributes of a business entity by collecting data from multiple source applications. Consider, for example, the demographic data associated with a bank customer. The retail operational system may provide some attributes such as social security number, address, and phone number. A mortgage system or some purchased database may provide with employment, income, and net worth information.

The structure of the data in any single source application is likely to be inadequate for the data warehouse. The structure in a single application may be influenced by many factors, including:

- **Purchased Applications:** The application data structure may be dictated by an application that was purchased from a software vendor and integrated into the business. The user of the application may have very little or no control over the data model. Some vendor applications have a very generic data model that is designed to accommodate a large number and types of businesses.
- **Legacy Application:** The source application may be a very old mostly homegrown application where the data model has evolved over the years. The database engine in this application may have been changed more than once without anyone taking the time to fully exploit the features of the new engine. There are many legacy applications in existence today where the data model is neither well documented nor understood by anyone currently supporting the application.
- **Platform Limitations:** The source application data model may be restricted by the limitations of the hardware/ software platform or development tools and technologies. A database platform may not support certain logical relationship or there may be physical limitations on the data attributes.



•No data model restrictions of the source application

•Data warehouse model has business entities

Figure 5. Data warehouse entities align with the business structure

Figure 5 illustrates the alignment of data warehouse entities with the business structure. The data warehouse model breaks away from the limitations of the source application data models and builds a flexible model that parallels the business structure. This extensible data model is easy to understand by the business analysts as well as the managers.

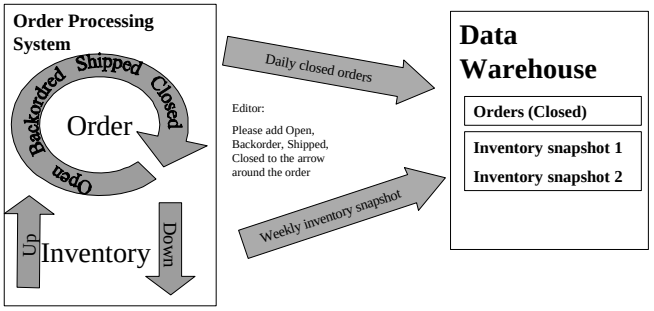
OPERATIONAL SYSTEMS	DATA WAREHOUSE
• PROCESSES THOUSANDS OF MILLIONS OF TRANSACTIONS DAILY.	• PROCESSES ONE TRANSACTION DAILY THAT CONTAINS MILLIONS OF RECORDS.
• Deals with one account at a time.	• Deals with a summary of multiple accounts
• Runs day-to-day operations.	• Creates reports for strategic decision-makers.
• Exists on different machines, dividing university resources.	• Exists on a single machine, providing a centralized resource.
• Provides instantaneous snapshot of an organization's affairs. Data changes from moment to moment.	• Adds layers of snapshots on a regularly scheduled basis. Data changes only when DW Team updates the warehouse.

Transformation of the Operational State Information

It is essential to understand the implications of not being able to maintain the state information of the operational system when the data is moved to the data warehouse. Many of the attributes of entities in the operational system are very dynamic and constantly modified. Many of these dynamic operational system attributes are not carried over to the data warehouse; others are static by the time they are moved to the data warehouse. A data warehouse generally does not contain information about entities that are dynamic and constantly going through state changes.

To understand what it means to lose the operational state information, let us consider the example of an order fulfillment system that tracks the inventory to fill orders. First let us look at the order entity in this operational system. An order may go through many different statuses or states before it is fulfilled or goes to the "closed" status. Other order statuses may indicate that the order is ready to be filled, it is being filled, back ordered, ready to be shipped, etc. This order entity may go through many states that capture the status of the order and the business processes that have been applied to it. It is nearly impossible to carry forward all of attributes associated with these order states to the data warehousing system. The data warehousing system is most likely to have just one final snapshot of this order. Or, as the order is ready to be moved into the data warehouse, the information may be gathered from multiple operational entities such as order and shipping to build the final data warehouse order entity.

Now let us consider the more complicated example of inventory data within this system. The inventory may change with every single transaction. The quantity of a product in the inventory may be reduced by an order fulfillment transaction or this quantity may be increased with receipt of a new shipment of the product. If this order processing system executes ten thousand transactions in a given day, it is likely that the actual inventory in the database will go through just as many states or snapshots during this day. It is impossible to capture this constant change in the database and carry it forward to the data warehouse. This is still one of the most perplexing problems with the data warehousing systems. There are many approaches to solving this problem. You will most likely choose to carry periodical snapshots of the inventory data to the data warehouse. This scenario can apply to a very large portion of the data in the operational systems. The issues associated with this get much more complicated as extended time periods are considered.



•Operational state information is not carried to the data warehouse

•Data is transferred to the data warehouse after all state changes

•Or, data is transferred with period snapshots

Figure 6. Transformation of the operational state information

Figure 6 illustrates how most of the operational state information cannot be carried over the data warehouse system.

De-normalization of Data

Before we consider data model de-normalization in the context of data warehousing, let us quickly review relational database concepts and the normalization process. E. F. Codd developed relational database theory in the late 1960s while he was a researcher at IBM. Many prominent researchers have made significant contributions to this model since its introduction. Today, most of the popular database platforms follow this model closely. A relational database model is a collection of two-dimensional tables consisting of rows and columns. In the relational modeling terminology, the tables, rows, and columns are respectively called relations, attributes, and tuples. The name for relational database model is derived from the term relation for a table. The model further identifies unique keys for all tables and describes the relationship between tables.

Normalization is a relational database modeling process where the relations or tables are progressively decomposed into smaller relations to a point where all attributes in a relation are very tightly coupled with the primary key of the relation. Most data modelers try to achieve the "Third Normal Form" with all

of the relations before they de-normalize for performance or other reasons. The three levels of normalization are briefly described below:

- First Normal Form: A relation is said to be in First Normal Form if it describes a single entity and it contains no arrays or repeating attributes. For example, an order table or relation with multiple line items would not be in First Normal Form because it would have repeating sets of attributes for each line item. The relational theory would call for separate tables for order and line items.
- Second Normal Form: A relation is said to be in Second Normal Form if in addition to the First Normal Form properties, all attributes are fully dependent on the primary key for the relation.
- Third Normal Form: A relation is in Third Normal Form if in addition to Second Normal Form, all non-key attributes are completely independent of each other.

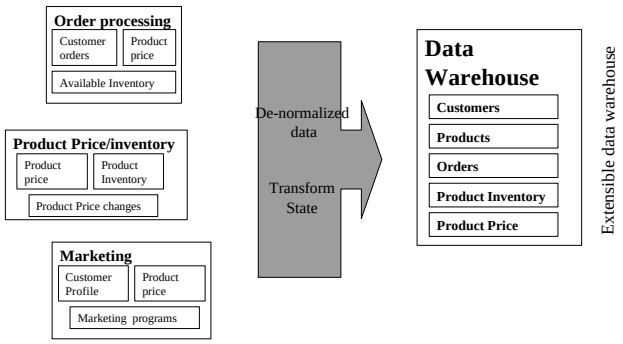
The process of normalization generally breaks a table into many independent tables. While a fully normalized database can yield fantastically flexible model, it generally makes the data model more complex and difficult to follow. Further, a fully normalized data model can perform very inefficiently. A data modeler in an operational system would take normalized logical data model and convert it into a physical data model that is significantly de-normalized. De-normalization reduces the need for database table joins in the queries.

Some of the reasons for de-normalizing the data warehouse model are the same as they would be for an operational system, namely, performance and simplicity. The data normalization in relational databases provides considerable flexibility at the cost of the performance. This performance cost is sharply increased in a data warehousing system because the amount of data involved may be much larger. A three-way join with relatively small tables of an operational system may be acceptable in terms of performance cost, but the join may take unacceptably long time with large tables in the data warehouse system.

Static Relationships in Historical Data

Another reason that de-normalization is an important process in data warehousing modeling is that the relationship between many attributes does not change in this historical data. For example, in an operational system, a product may be part of the product group "A" this month and product group "B" starting next month. In a properly normalized data model, it would be inappropriate to include the product group attribute with an order entity that records an order for this product; only the product ID would be included. The relational theory would call for a join on the order table and product table to determine the product group and any other attributes of this product. This relational theory concept does not apply to a data warehousing system because in a data warehousing system you may be capturing the group that this product belonged to when the order was filled. Even though the product moves to different groups over time, the relationship between the product and the group in context of this particular order is static.

Another important example can be the price of a product. The prices in an operational system may change constantly. Some of these price changes may be carried to the data warehouse with a periodic snapshot of the product price table. In a data warehousing system you would carry the list price of the product when the order is placed with each order regardless of the selling price for this order. The list price of the product may change many times in one year and your product price database snapshot may even manage to capture all these prices. But, it is nearly impossible to determine the historical list price of the product at the time each order is generated if it is not carried to the data warehouse with the order. The relational database theory makes it easy to maintain dynamic relationships between business entities, whereas a data warehouse system captures relationships between business entities at a given time.



- Structured extensible data model
- Data warehouse model aligns with the business structure
- Transformation of the state information
- Data is de-normalized because the relationships are static

Figure 7. Logical transformation of application data

Logical transformation concepts of source application data described here require considerable effort and they are a very important early investment towards development of a successful data warehouse. Figure 7 highlights the logical transformation concepts discussed in this section.

Physical Transformation of Operational Data

Physical transformation of data homogenizes and purifies the data. These data warehousing processes are typically known as "data scrubbing" or "data staging" processes. The "data scrubbing" processes are some of the most labor-intensive and tedious processes in a data warehousing project. Yet, without proper scrubbing, the analytical value of even the clean data can be greatly diminished. Physical transformation includes the use of easy-to-understand standard business terms, and standard values for the data. A complete dictionary associated with the data warehouse can be a very useful tool. During these physical transformation processes the data is sometimes "staged" before it is entered into the data warehouse. The data may be combined from multiple applications during this "staging" step or the integrity of the data may be checked during this process.

The concepts associated with the physical transformation of the data are introduced in this sub-section. Historical data and the current operational application data is likely to have some missing or invalid values. It is important to note that it is essential to manage missing values or incomplete transforma-

tions while moving the data to the data warehousing system. The end user of the data warehouse must have a way to learn about any missing data and the default values used by the transformation processes.

Operational terms transformed into uniform business terms

The terms and names used in the operational systems are transformed into uniform standard business terms by the data warehouse transformation processes. The operational application may use cryptic or difficult to understand terms for a variety of different reasons. The platform software may impose length and format restriction on a term, or purchased application may be using a term that is too generic for your business. The data warehouse needs to consistently use standard business terms that are self-explanatory.

A customer identifier in the operational systems may be called cust, cust_id, or cust_no. Further, different operational applications may use different terms to refer to the same attribute. For example, a customer in the loan organization in a bank may be referred to as a Borrower. You may choose a simple standard business term such as Customer Id in the data warehouse. This term would require little or no explanation even to the novice user of the data warehouse.

Discussions

- Give some examples of Operational systems.
- Explain how the logical transformation of operational data takes place.
- Describe the Differences between transaction and analysis processes
- Explain the Physical transformation of operational data
- Discuss the need of de-normalizing the data.

References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
4. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
5. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

Notes

LESSON 7
BUILDING DATA WAREHOUSING, IMPORTANT CONSIDERATIONS

Structure

- Objective
- Introduction
- Building a Data Warehouse
- Nine decisions in the design of a data warehouse

Objective

The objective of this lesson is to introduce you with the basic concepts behind building a data warehouse.

Introduction

There are several reasons why organizations consider data warehousing a critical need. These drivers for data warehousing can be found in the business climate of a global marketplace, in the changing organizational structures of successful corporations, and in the technology.

Traditional databases support On-Line Transaction Processing (OLTP), which includes insertions, updates, and deletions, while also supporting information query requirements. Traditional relational databases are optimized to process queries that may touch a small part of the database and transactions that deal with insertions or updates of a few tuples per relation to process. Thus, they cannot be optimized for OLAP, DSS, or data mining. By contrast, data warehouses are designed precisely to support efficient extraction, process-ing, and presentation for analytic and decision-making purposes. In comparison to traditional databases, data warehouses generally contain very large amounts of data from multiple sources that may include databases from different data models and sometimes lies acquired from independent systems and platforms.

Building a Data Warehouse

In constructing a data warehouse, builders should take a broad view of the anticipated use of the warehouse. There is no way to anticipate all possible queries or analyses during the design phase. However, the design should specifically support ad-hoc querying, that is, accessing data with any meaningful combination of values for the attributes in the dimension or fact tables. For example; a marketing-intensive consumer-products company -would require different ways of organizing the data warehouse than would a nonprofit charity focused on fund raising. An appropriate schema should be chosen that reflects anticipated usage.

Acquisition of data for the warehouse involves the following steps:

- The data must be extracted from multiple, heterogeneous sources; for example, databases or other data feeds such as those containing financial market data or environmental data.
- Data must be formatted for consistency within the warehouse. Names, meanings, and domains of data from unrelated sources must be reconciled. For instance, subsidiary companies of a large corporation may have different fiscal calendars with quarters ending on different dates, making it difficult to aggregate financial data by quarter. Various credit cards may report their transactions differently, making it difficult to -compute all credit sales. These format inconsistencies must be resolved.
- The data must be cleaned to ensure validity. Data cleaning is an involved and involved and complex process that has been identified as the largest labor-demanding component -data warehouse construction. For input data, cleaning must occur before the data are loaded into the warehouse. There is nothing about cleaning data that is specific to data warehousing and that could not be applied to a host database. However, since input data must be examined and formatted consistently, data warehouse builders should take this opportunity to check for validity and quality. Recognizing erroneous and incomplete data is difficult to automate, and cleaning that requires automatic error correction can be even tougher. Some aspects, such as domain checking, are easily coded into data cleaning routines, but automatic recognition of other data problems can be more challenging. (For example, one might require that City = 'San Francisco' together with State = 'CT' be recognized as an incorrect combination.)

After such problems have been taken care of, similar data from different sources must be coordinated for loading into the warehouse. As data managers in the organization. Discover that their data are being cleaned for input into the warehouse; they will likely want to upgrade their data with the cleaned data. The process of returning cleaned data to the source is called **back flushing**.

- The data must be fitted into the data model of the warehouse. Data from the various sources must be installed in the data model of the warehouse. Data may have to be converted from relational, object-oriented, or legacy databases (network and/or hierarchical) to a multidimensional model.
- The data must be loaded into the warehouse. The sheer volume of data in the warehouse makes loading the data a significant task. Monitoring tools for loads as well as methods to recover from incomplete or incorrect loads are required. With the huge volume of data in the warehouse, incremental updating is usually the only feasible approach. The refresh policy will probably emerge as a compromise that takes into account the answers to the following questions:
- How up-to-date must the data be?
- Can the warehouse go off-line, and for how long?
- What are the data interdependencies?
- What is the storage availability?
- What are the distribution requirements (such as for replication and partitioning)?

- What is the loading time (including cleaning, formatting, copying, transmitting, and overhead such as index rebuilding)?

As we have said, databases must strike a balance between efficiency in transaction processing and supporting query requirements (ad hoc user requests), but a data warehouse is typically optimized for access from a decision maker's, needs.

Data storage in a data warehouse reflects this specialization and involves the following processes:

- Storing the data according to the data model of the warehouse.
- Creating and maintaining, required data structures.
- Creating and maintaining appropriate access paths.
- Providing for time-variant data as new data are added.
- Supporting the updating of warehouse data.
- Refreshing the data.
- Purging data

Although adequate time can be devoted initially to constructing the warehouse, the sheer volume of data in the warehouse generally makes it impossible to simply reload the warehouse in its entirety later on. Alternatives include selective (partial) refreshing of data and separate warehouse versions (requiring, double, storage capacity for the ware-house). When the warehouse uses an incremental data refreshing mechanism, data may need to be periodically purged; for example, a warehouse that maintains data on the pre-vious twelve business quarters may periodically purge its data each year.

Data warehouses must also be designed with full consideration of the environment in which they will reside. Important design considerations include-the following:

- Usage projections.
- The fit of the data model.
- Characteristics of available sources.
- Design of the metadata component.
- Modular component design.
- Design for manageability and change.
- Considerations of distributed and parallel architecture

We discuss each of these in turn. Warehouse design is initially driven by usage pro-jections; that is, by expectations about who will use the warehouse and in what way. Choice of a data model to support this usage is a key initial decision. Usage projections and the characteristics of the warehouse's data sources are both taken into account. Modular design is a practical necessity to allow the warehouse to evolve with. the organization and its information environment. In addition, a well-built data warehouse must be designed for maintainability, enabling the warehouse managers to effectively plan for and manage change while providing optimal support to users.

Metadata is defined as - description of a database including its schema definition. **The metadata repository** is a key data warehouse component. The metadata repository includes both technical and business metadata. The first, technical metadata, covers details of acquisition processing, storage structures, data descriptions, warehouse operations and maintenance, and access support functionality. The second, business metadata, includes the relevant business rules and organizational details supporting the warehouse.

The architecture of the organization's distributed computing environment is a major -determining characteristic for the design of the warehouse. There are two basic distributed architectures: the **distributed warehouse and the federated warehouse**. For a distributed warehouse, all the issues of distributed databases are relevant, for example, replication, partitioning, communications, and consistency concerns. A distributed architecture can provide benefits particularly important to warehouse performance, such as improved load balancing, scalability of performance, and higher availability. A single replicated metadata repository would reside at each distribution site. The idea of the federated warehouse is like that of the federated database: a decentralized confederation of autonomous data warehouses, each with its own metadata repository. Given the magnitude of the challenge inherent to data warehouses, it is likely that such federations will consist of smaller-scale components, such as data marts. Large organizations may choose -to federate data marts rather than build huge data warehouses.

Nine Decisions in the design of a Data Warehouse

The job of a data warehouse designer is a daunting one. Often the newly appointed data warehouse designer is drawn to the job because of the high vis-ibility and importance of the data warehouse function. In effect, management says to the designer: "Take all the enterprise data and make it available to management so that they can answer all their questions and sleep at night. And please do it very quickly, and we're sorry, but we can't add any more staff until the proof of concept is successful."

This responsibility is exciting and very visible, but most designers feel over-whelmed by the sheer enormity of the task. Something real needs to be accom-plished, and fast. Where do you start? Which data should be brought up first? Which management needs are most important? Does the design depend on the details of the most recent interview, or are there some underlying and more constant design guidelines that you can depend on? How do you scope the proj-ect down to something manageable, yet at the same time build an extensible architecture that will gradually let you build a comprehensive data warehouse environment?

These questions are close to causing a crisis in the data warehouse industry. Much of the recent surge in the industry toward "data marts" is a reaction to these very issues. Designers want to do something simple and achievable. No one is willing to sign up for a galactic design that must somehow get every-thing right on the first try. Everyone hopes that in the rush to simplification, the long-term coherence and extendibility of the design will not be compro-mised. Fortunately, a pathway through this design challenge achieves an implementable immediate result, and at the same time it continuously mends the design so that eventually a true enterprise-scale data warehouse is built. The secret is to keep in mind a design methodology, which Ralph Kimball calls the "nine-step method"(see Table 7.1).

As a result of interviewing marketing users, finance users, sales force users, operational users, first- and second-level manage-ment, and senior manage-ment, a picture emerges of what is keeping these people awake at night.

Table 7.1 Nine-Step Method in the Design of a Data Warehouse

1. Choosing the subject matter
2. Deciding what a fact table represents
3. Identifying and conforming the dimensions.
4. Choosing the facts
5. Storing precalculations in the fact table.
6. Rounding out the dimension tables
7. Choosing the duration of the database
8. The need to track slowly changing dimensions
9. Deciding the query priorities and the query modes

Can list and prioritize the primary business issues facing the enterprise. At the same time you should conduct a set of interviews with the legacy systems' DBAs who will reveal which data sources are clean, which contain valid and consistent data, and which will remain supported over the next few years.

Preparing for the design with a proper set of interviews is crucial. Inter-viewing is also one of the hardest things to teach people. I find it helpful to reduce the interviewing process to a tactic and an objective. Crudely put, the tactic is to make the end-users talk about what they do, and the objective is to gain insights that feed the nine design decisions. The tricky part is that the interviewer can't pose the design questions directly to the end users. End users talk about what are important in their business lives. End users are intim-idated by system design questions, and they are quite right when they insist that system design is IT responsibility, not theirs. Thus, the challenge of the data mart designer is to meet the users tar more than half way.

In any event, armed with both the top-down view (what keeps management awake) and the bottom-up view (which data sources are available), the data warehouse designer may follow these steps:

Step 1: Choosing the subject matter of a particular data mart. The first data mart you build should be the one with the most bangs for the buck. It should simultaneously answer the most important business questions and be the most accessible in terms of data extraction. According to Kimball, a great place to start in most enterprises is to build a data mart that consists of cus-tomer invoices or monthly statements. This data source is probably fairly accessible and of fairly high quality. One of Kimball's laws is that the best data source in any enterprise is the record of "how much money they owe us." Unless costs and profitability are easily available before the data mart is even designed, it's best to avoid adding these items to this first data mart. Nothing drags down a data mart implementation faster than a heroic or impossible mission to provide activity-based costing as part of the first deliverable.

Step 2: Deciding exactly what a fact table record represents. This step, according to R. Kimball, seems like a technical detail at this early point, but it is actually the secret to making progress on the design. A table is the large central table in the dimensional designs that has a multipart key. Each component of the multipart key is a foreign key to an individual dimension table. In the example of customer invoices, the "grain" of the fact table is the individual line item on the customer invoice. In other words, a line item on an invoice is a single fact table record, and vice versa. Once the fact table representation is decided, a coherent discussion of what the dimen-sions of the data mart's fact table are can take place.

Step 3: Identifying and conforming the dimensions. The dimensions are the drivers of the data mart. The dimensions are the platforms for brows-ing the allowable constraint values and launching these constraints. The dimensions are the source' of row headers in the user's final reports; they carry the enterprise's vocabulary to the users. A well-architect set of dimensions makes the data mart understandable and easy to use. A poorly presented or incomplete set of dimensions robs the data mart of its usefulness. Dimensions should be chosen with the long-range data warehouse in mind. This choice pre-sents the primary moment at which the data mart architect must disregard the data mart details and consider the longer-range plans. If any dimension occurs in two data marts, they must be exactly the same dimension, or one must be a mathematical subset of the other. Only in this way can two data marts share one or more dimensions in the same application. When a dimen-sion is used in two data marts, this dimension is said to be conformed. Good examples of dimensions that absolutely must be conformed between data marts are the customer and product dimensions in an enterprise. If these dimensions .are allowed drifting out of synchronization between data marts, the overall data warehouse will fail, because the two data marts will not be able to be used together. The requirement to conform dimensions across data marts if very strong. Careful thought must be given to this requirement before the firs: data mart is implemented. The data mart team must figure out what an enter-prise customer ID is and what an enterprise product ID is. If this task is done correctly, successive data marts can be built at different times, on different machines, and by different development teams, and these data marts

Will merge coherently into an overall data warehouse. In particular, if the dimen-sions of two data marts are conformed, it is easy to implement drill across by sending separate queries to the two data marts, and then sort-merging the answer sets on a set of common row headers. The row headers can be made be common only if they are drawn from a conformed dimension common to the two data marts.

With these first three steps correctly implemented, designers can attack last six steps (see Table 7.1). Each step gets easier if the preceding steps have been performed correctly.

Discussions

- Write short notes on:
  - Data cleaning
  - Back flushing
  - Heterogeneous Sources
  - Metadata repository

- Discuss various steps involved in the acquisition of data for the data warehouse.
- List out various processes involved in data storage in a data warehouse.
- What are the important design considerations, which need to be thought of, while designing a data warehouse?
- Explain the difference between distributed warehouse and the federated warehouse.
- What are the nine decisions in the design of a data warehouse?

References

1. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
2. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
3. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
4. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

Notes

LESSON 8
BUILDING DATA WAREHOUSING - 2

Structure

- Objective
- Introduction
- Data warehouse Application
- Approaches used to build a Data Warehousing
- Important considerations
- Tighter Integration
- Empowerment
- Willingness
- Reason for Building a Data Warehousing

Objective

The aim of this lesson is to study about Data warehouse applications and various approaches that are used to build a Data warehouse.

Introduction

It is the professional warehouse team deals with issues and develops solutions, which wi1l best suit the needs of the analytical user community. A process of negotiation and, sometimes of give and take is used to address issues that have common ground between the players in the data warehouse delivery team.

Data Warehouse Application

In application, data warehouse application is different transaction deals with large amounts of data, which is aggregate in nature, a data warehouse application answers questions like

- What is the average deposit by branch?
- Which day of the week is busiest?
- Which customers with high average balances currently are not participating in a checking- plus account)

Because we are dealing with questions, each request is unique. The interface that supports this end user must be flexible by design. You have many different applications accessing the same information.

Each with a particular strength. A data mart is typically a subset of your warehouse with a specific purpose in mind. For example, you might have a financial mart and a marketing mart; each designed to feed information to a specific part of your corporate business community. A key issue in the industry today is which approach should you take when building a Decision Support System?

Approaches used to build a Data Warehouse

We have experiences with two approaches to the build process:

Top-Down Approach, meaning that an organization has developed an enterprise data model, collected enterprise wide business requirements, and decided to build an enterprise data warehouse with subset data marts. In this approach, we need to spend the extra time and build a core data warehouse first, and then use this as the basis to quickly spin off many data marts.

Disadvantage is this approach takes longer to build initially since time has to be spent analyzing data requirements in the full – blown warehouse, identifying the data elements that will be used in numerous marts down the road.

The advantage is that once you go to build the data mart, you already have the warehouse to draw from.

Bottom-Up Approach, implying that the business priorities resulted in developing individual data marts, which are then integrated into the enterprise data warehouse. In this approach we need to build a workgroup specific data mart first. This approach gets data into your users hands quicker but the work it takes to get the information into the data mart may not be reusable when moving the same data into a warehouse or trying to use similar data in different data mart.

Advantage is you gain speed but not portability

We don't care. In fact, we'd like to coin the term "ware mart". The answer to which approach is correct depends on a number of vectors. You can take the approach that gets information in to your user's hands quickest. In our experience, that arguing over the marts and evolving into a warhorse. Results are what count not arguing over the data mart v/s the warehouse approach.

Are they Different?

Well, you can give many reasons your operational system and your data warehouses are not the same. The data need to support operational needs is differently then the data needed to support analytical processing. In fact the data are physically stored quite differently. An operationa1 system is optimized for transactional updates, while a data warehouse system is optimized-for large queries dealing with large data sets. These differences become apparent. When you begin to monitor central processing unit (CPU) usage on a computer that contains a data warehouse v / s CPU usage on a system that contains an operational system.

Important Considerations

- Tighter Integration
- Empowerment
- Willingness

Tighter Integration

The term back end describes the data repository used to support the data warehouse coupled with the software that supports the repository. For example, Oracle 7 Cooperative Server Front end describes the tools used *by* the warehouse end-users to support their decision making activities, With classic operational systems sometimes ad-hoc dialog exist between the front end and back end to support specialties; with the data

warehouse team, this dialog must be ongoing as the warehouse springs to life. Unplanned fixes may be come necessary for the back end to support what the user community daces to do it data warehouse system. This tighter integration between the front end and the back end requires continual communication between the data warehouse project team players. Some tools exist for end user analytic processing that require specific data structures in the warehouse and in some cases specific naming standards, as result sometimes the nature *of* the front end tool drives the design of the data structure in the warehouse.

Empowerment

Users to control their own destiny- no running to the programmers asking for reports programmed to satisfy burning needs. How many times in operational environments are new reports identified, scoped and programmed, only to discover by the tie they are ready, the user has found other sources for the desired output? An inevitable time delay exists between the identification of new requirements and their delivery. This is no one's fault it simple reality.

Willingness

With the data warehouse Resistance often exists to making change, but then a warehouse initiative is rolled out with a set of executives, manager clerks and analysts it will succeed. With continual input and attention to detail the warehouse team spends most of its time working wit the hands -on involvement of the uses. All but the most stubborn users embrace the new technology because it can provide them with answers to their question almost immediately. In shops here the move to a windows-like environment has no yet been made the simultaneous rollout of a point-and- click interface and the placement of power in the hands of the consumer is bound to succeed.

Reason for Building a Data Warehousing Opportunity

Considering a data warehouse development project within every organization, many of the same frustration exist. These common themes are true business opportunity that a data \warehouse can assist in achieving. Whenever one or more of the following is heard, it is a signal that a true business reason exists for building a data warehouse

- We have all the data in our system. We just need access to it to make a better decision for running the business.
- We need an easier way to analyze information than training our business people in SQL.

Self Test

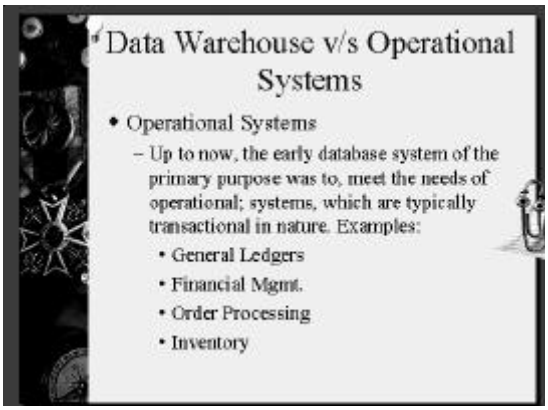
1. Identify the factors which does not facilitate change management
   a. Training
   b. Communication
   c. Rigid structure
   d. None of these
2. Which group is not responsible for successful warehouse ?
   a. Users
   b. Top management
   c. Working management
   d. None of these
3. Which is not a key to developing data warehouse?
   a. Structure of organization
   b. Change management
   c. User requirement
   d. Methodology
4. Which *of* these factors does not track project development?
   a. Performance measures based requirements
   b. Enterprise requirements gathering
   c. Cultural requirements
   d. None of these
5. Which one is not a consideration for building a data warehouse?
   a. Operating efficiency
   b. Improved customer service
   c. Competitive advantage
   d. None of these

Discussion

1. Write short notes on:
   - Tighter Integration
   - Empowerment
   - Willingness
2. Discuss different approaches used to build a Data Warehousing. Which approach is generally used for building a data warehouse?
3. Discuss various applications of a data warehouse.

References

1. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
2. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
3. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.
4. **Alex Berson, Stephen J.Smith,** *Data warehousing, data mining and OLTP:* Tata McGraw- Hill Publishing, 2004



Contd.

- Operational systems(OS) by nature are primarily concerned with the handling of a single transaction.
- A typical OS deals with one order, one account, one inventory item. An OS typically deals with predefined events and, due to the nature of these events, requires fast access. Each transaction usually deals with small amounts of data.
- In this type of system, the info reqd. when a customer initiates a transaction must be current. Before a bank will allow a withdrawal, it must first be certain of your current balance.

Important Considerations

- Tighter Integration
  - The term back end describes the data repository used to support the data warehouse coupled with the software that supports the repository. For e.g.,Oracle 7 Cooperative Server. Front end describes the tools used by the warehouse end-users to support their decision making activities. This tighter integration between front end and back end requires continual communication between the data warehouse project team players

Data Warehouse

- Data Warehouse application is different transaction deals with large amounts of data, which is aggregate in nature.
- Data Warehouse application answers questions like:
  - What is the average deposit by branch?
  - Which day of the week is busiest?
- Data mart is a subset of your warehouse with a specific purpose in mind.
  - E.g., a financial mart and a marketing mart, is designed to feed info to a specific part of your corporate business community.

Empowerment

- Users to control their own destiny – no running to the programmers asking for reports programmed to satisfy burning needs, how many times in operational environments are new reports identified, scoped and programmed. An inevitable time delay exists between the identification of new requirements and their delivery.

Two approaches when building DSS:

1. To spend extra time and build a core data warehouse first, then use this a s the basis to quickly spin off many data marts. This approach takes longer to build initially since time has to be spent analyzing data requirements in the full-blown warehouse, identifying the data elements that will be used in numerous data marts down the road. The adv. Is that once you go to build the data mart, you already have a warehouse to draw from.
2. The other approach is to build a workgroup specific data mart first. This approach gets data into yours user's hand quicker but the work it takes to get then info into the data mart may not be reusable when moving the same data into a warehouse or trying to use similar data in different data mart.You gain speed but not portability.

Willingness

- With the data warehouse resistance often exists to making change, but then a warehouse initiative is rolled out with a set of executives, manager, clerks and analysts it will succeed.

LESSON 9
BUSINESS CONSIDERATIONS: RETURN ON INVESTMENT DESIGN CONSIDERATIONS

Structure

- Objective
- Introduction
- Need of a Data warehouse
- Business Considerations: Return on Investment
- Approach
- Organizational issues
- Design Considerations
- Data content
- Metadata
- Data distribution
- Tools
- Performance considerations

Objective

The objective of this lesson is to learn about the need of a data warehouse. It also includes topics related to various business considerations and performance considerations.

Introduction

The data warehouse is an environment, not a product. It is an architectural construct of information systems that provides users with current and historical decision support information that is hard to access or present in traditional operational data stores. In fact, the data warehouse is a cornerstone of the organization's ability to do effective information processing, which, among other things, can enable and shear the discovery and exploration of important business trends and dependencies that otherwise would have gone unnoticed.

In principle, the data warehouse can meet informational needs of knowledge workers and can provide strategic business opportunities by allowing customers and vendors access to corporate data while maintaining necessary security measures. There are several reasons why organizations consider data warehousing a critical need.

Need of a Data Warehouse

There are several reasons why organizations consider data warehousing a crit-ical need. These drivers for data warehousing can be found in the business cli-mate of a global marketplace, in the changing organizational structures of successful corporations, and in the technology.

From a business perspective, to survive and succeed in today's highly com-petitive global environment, business users demand business answers mainly because

- Decisions need to be made quickly and correctly, using all available data. . Users are business domain experts, not computer professionals.
- The amount of data is doubling every 18 months, which affects response time and the sheer ability to comprehend its content.
- Competition is. Heating up in the areas of business intelligence. And added Information value.

In addition, the necessity for data warehouses has increased as organizations dis-tribute control away from the middle-management layer, which has traditionally provided and screened business information. As users depend more on informa-tion obtained from Information Technology (IT) systems - from critical-success measures to vital business-event-related information - the need to provide an information warehouse for the remaining staff to use becomes more critical.

There are several technology reasons for the existence of data warehousing. First, the data warehouse is designed to address the incompatibility of infor-mational and operational transactional systems. These two classes of informa-tion systems are designed to satisfy different, often incompatible, requirements. At the same time, the IT infrastructure is changing rapidly, and its capabilities are increasing, _s evidenced by the following:

- The price of MIPS (computer processing speed) continues to decline, while
- The power of microprocessors doubles every 2 years.
- The price of digital storage is rapidly dropping.
- Network bandwidth is increasing, while the price of high bandwidth is decreasing.
- The workplace is increasingly heterogeneous with respect to both the hard- ware and software.
- Legacy systems need to, and can, be integrated with new applications.

These business and technology drivers often make building a data warehouse a strategic imperative. This chapter takes a close look at what it takes to build a successful data warehouse.

Business Considerations: Return on Investment

Approach

The information scope of the data warehouse varies with the business require-ments, business priorities, and even magnitude of the problem. The subject-oriented nature of the data warehouse means that the nature of the subject determines the scope (or the coverage) of the warehoused information. Specifically, if the data warehouse is implemented to satisfy a specific subject area (e.g., human resources), such a warehouse is expressly designed to solve busi-ness problems related to personnel. An organization may choose to build another warehouse for its marketing department. These two warehouses could be implemented independently and be completely stand-alone applica-tions, or they could be viewed as compo-

nents of the enterprise, interacting with each other, and using a common enterprise data model. As defined earlier the individual warehouses are known as data marts. Organizations embarking on data warehousing development can chose one of the two approaches:

- The top-down approach, meaning that an organization has developed an enterprise data model, collected enterprise wide business requirements, an..: decided to build an enterprise data warehouse with subset data marts
- The bottom-up approach, implying that the business priorities resulted i.e. developing individual data marts, which are then integrated into the enter-prise data warehouse
- The bottom-up approach is probably more realistic, but the complexity of the integration may become a serious obstacle, and the warehouse designer's should carefully analyze each data mart for integration affinity.

Organizational Issues

Most IS an organization has considerable expertise in developing operational systems. However; the requirements and environments associated with the informational applications of a data warehouse are different. Therefore, an organization will need to employ different development practices than the ones it uses for operational applications.

The IS department will need to bring together data that cuts across a com-pany's operational systems as well as data from outside the company. But users will also need to be involved with a data warehouse implementation since they are closest to the data. In many ways, a data warehouse implemen-tation is not truly a technological issue; rather, it should be more concerned with identifying and establishing information requirements, the data sources to fulfill these requirements, and timeliness.

Design Considerations

To be successful, a data warehouse designer must adopt a holistic approach consider all data warehouse components as parts of a single complex system and take into the account all possible data sources and all known usage requirements. Failing to do so may easily result in a data warehouse design that is skewed toward a particular business requirement, a particular data source, or a selected access tool.

In general, a data warehouse's design point is to consolidate data from mul-tiple, often heterogeneous, sources into a query database. This is also one of the reasons why a data warehouse is rather difficult to build. The main factors include

- Heterogeneity of data sources, which affects data conversion, quality, and time-lines. Use of historical data, which implies that data may be "old". Tendency of databases to grow very large. Another important point concerns the experience and accepted practices. Basi-cally, the reality is that the data warehouse design is different from traditional OLTP. Indeed, the data warehouse is business-driven (not IS-driven, as in OLTP), requires continuous interactions with end users, and is never finished, since both requirements and data sources change. Understanding these points allows developers to avoid a number of pitfalls relevant to data warehouse development, and justifies a new approach to data warehouse design: a business driven, continuous, iterative warehouse engineering approach. In addition to these general considerations, there are several specific points relevant to the data warehouse design.

Data content

One common misconception about data warehouses is that they should not con-tain as much detail-level data as operational systems used to source this data in. In reality, however, while the data in the warehouse is formatted differently from the operational data, it may be just as detailed. Typically, a data warehouse may contain detailed data, but the data is cleaned up and-transformed to fit the ware-house model, and certain transactional attributes of the data are filtered out. These attributes are mostly the ones used for the internal transaction system logic, and they are not meaningful in the context of analysis and decision-making.

The content and structure of the data warehouse are reflected in its data model. The data model is the template that describes how information will be organized within the integrated warehouse framework. It identifies major sub-jects and relationships of the model, including keys, attribute, and attribute groupings. In addition, a designer should always remember that decision sup-port queries, because of their broad scope and analytical intensity, require data models to be optimized to improve query performance. In addition to its effect on query performance, the data model affects data storage requirements and data loading performance.

Additionally, the data model for the data warehouse may be (and quite often is) different from the data models for data marts. The data marts, discussed in the previous chapter, are sourced from the data warehouse, and may contain highly aggregated and summarized data in the form of a specialized demoralized relational schema (star schema) or as a multidimensional data cube. The key point is, however, that in a dependent data mart environment, the data mart data is cleaned up, is transformed, and is consistent with the data warehouse and other data marts sourced from the same warehouse.

Metadata

As already discussed, metadata defines the contents and location of data (data model) in the warehouse, relationships between the operational databases and the data warehouse, and the business views of the warehouse data that are accessible by end-user tools. Metadata is searched by users to find data defini-tions or subject areas. In other words, metadata provides decision-support oriented pointers to warehouse data, and thus provides a logical link between warehouse data and the decision support application. A data warehouse design should ensure that there is mechanisms that populates and maintains the metadata repository, and that all access paths to the data warehouse have metadata as an entry point. To put it another way, the warehouse design should prevent any direct access to the warehouse data (especially updates) if it does not use metadata definitions to gain the access.

Data Distribution

One of the biggest challenges when designing a data warehouse is the data placement and distribution strategy. This follows from the fact that as the data Volumes continue to grow; the database size may rapidly outgrow a single server. Therefore, it becomes necessary to know how the data should be divided across multiple servers, and which users should get access to which types of data. The data placement and distribution design should consider several options, including data distribution by subject area (e.g., human resources, marketing), location (e.g., geographic regions), or time (e.g., current, monthly, Quarterly). The designers should be aware that, while the distribution solves a number of problems, it may also create a few of its own; for example, if the warehouse servers are distributed across multiple locations, a query that spans several servers across the LAN or WAN may flood the network with a large amount of data. Therefore, any distribution strategy should take into account all possible access needs for the warehouse data.

Tools

A number of tools available today are specifically designed to help in the i1hple-mentation of a data warehouse. These tools provide facilities for defining the transformation and cleanup rules, data movement (from operational sources into the warehouse), end-user query, reporting, and data analysis. Each tool takes a slightly different approach to data warehousing and often maintains its own version of the metadata, which is placed in a tool-specific, proprietary meta-data repository. Data warehouse designers have to be careful not to sacrifice the overall design to fit a specific tool. At the same time, the designers have to make sure that all selected tools are compat-ible with the given data warehouse envi-ronment and with each other. That means that all selected tools can use a com-mon metadata repository. Alternatively, the tools should be able to source the metadata from the warehouse data dictionary (if it exists) or from a CASE tool used to design the warehouse database. Another option is to use metadata gate-ways that translate one tool's metadata into another tool's format. If these requirements are not satisfied, the resulting warehouse environment may rapidly become unmanageable, since every modification to the warehouse data model may involve some significant and labor-intensive changes to the meta-data' definitions for every tool in the environment. And then, these changes would have to be verified for consistency and integrity.

Performance Considerations

Although the data warehouse design point does not include sub second response times typical of OLTP systems, it is nevertheless a clear business requirement that an ideal data warehouse environment should support inter-active query processing. In fact, the majority of end-user tools are designed as interactive applications. Therefore, "rapid" query processing is a highly desired feature that should be designed into the data warehouse. Of course, the actual performance levels are business-dependent and vary widely from one environ-ment to another. Unfortunately, it is relatively difficult to predict the perfor-mance of a typical data warehouse. One of the reasons for this is the unpredictable usage pattern against the data. Thus, traditional database design and tuning techniques don't always work in the data warehouse arena. When designing a data warehouse, therefore, the need to clearly understand users informational requirements becomes mandatory. Specifically, knowing how end users need to access various data can help design warehouse data-bases to avoid the majority of the most expensive operations such as multitable scans and joins. For example, one design technique is to populate the ware-house with a number of demoralized views containing summarized, derived, and aggregated data. If done correctly, many end-user queries may execute directly against these views, thus maintain-ing appropriate overall perfor-mance levels.

Discussions

- Write short notes on:
  - Meta data
  - Data distribution
  - Data content
  - CASE tools
  - Data marts
- Discuss various design considerations, which are taken into account while building a data warehouse.
- Explain the need and importance of a data warehouse.
- Describe the organization issues, which are to be considered while building a data warehouse.
- Explain the need of Performance Considerations
- "Organizations embarking on data warehousing development can choose one of the two approaches". Discuss these two approaches in detail.
- Explain the business considerations, which are taken into account while building a data warehouse

References

1. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
2. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
3. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
4. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

LESSON 10
TECHNICAL CONSIDERATION, IMPLEMENTATION CONSIDERATION

Structure

- Objective
- Introduction
- Technical Considerations
- Hardware platforms
- Balanced approach
- Optimal hardware architecture for parallel query scalability
- Data warehouse and DBMS specialization
- Communications infrastructure
- Implementation Considerations
- Access tools

Objective

The purpose of this lesson is to take a close look at what it takes to build a successful data warehouse. Here, I am focusing on the Technical Issues, which are to be considered when designing and implementing a data warehouse.

Introduction

There are several technology reasons for the existence of data warehousing. First, the data warehouse is designed to address the incompatibility of informational and operational transac-tional system. These two classes of informa-tion systems are designed to satisfy different, often incompatible, requirements. At the same time, the IT infrastructure is changing rapidly, and its capabilities are increasing, _s evidenced by the following:

- The price of MIPS (computer processing speed) continues to decline, while
- The power of microprocessors doubles every 2 years.
- The price of digital storage is rapidly dropping.
- Network bandwidth is increasing, while the price of high bandwidth is decreasing.
- The workplace is increasingly heterogeneous with respect to both the hard- ware and software.
- Legacy systems need to, and can, be integrated with new applications.

These business and technology drivers often make building a data warehouse a strategic imperative. This chapter takes a close look at what it takes to build a successful data warehouse.

Technical Considerations

A number of technical issues are to be considered when designing and imple-menting a data warehouse environment. These issues include:

- The hardware platform that would house the data warehouse
- The database management system that supports the warehouse database
- The communications infrastructure that connects the warehouse, data marts,
- Operational systems, and end users
- The hardware platform and software to support the metadata repository
- The systems management framework that enables centralized management and administration of the entire environment.

Let's look at some of these issues in more detail.

Hardware Platforms

Since many data warehouse implementations are developed into already exist-ing environments, many organizations tend to leverage the existing platforms and skill base to build a data warehouse. This section looks at the hardware platform selection from an architectural viewpoint: what platform is best to build a successful data warehouse from the ground up.

An important consideration when choosing a data warehouse server is its capacity for handling the volumes of data required by decision support appli-cations, some of which may require a significant amount of historical (e.g., up to 10 years) data. This capacity requirement can be quite large. For example, in general, disk storage allocated for the warehouse should be 2 to 3 times the size of the data component of the warehouse to accommo-date DSS processing, such as sorting, storing of intermediate results, summarization, join, and format-ting. Often, the platform choice is the choice between a mainframe and non-MVS (UNIX or Windows NT) server.

Of course, a number of arguments can be made for and against each of these choices. For example, a mainframe is based on a proven technology; has large data and throughput capacity; is reliable, available, and serviceable; and may support the legacy databases that are used as sources for the data warehouse. The data warehouse residing on the mainframe is best suited for situations in which large amounts of legacy data need to be stored in the data warehouse. A mainframe system, however, is not as open and flexible as a contemporary client/server system, and is not optimized for ad hoc query processing. A mod-ern server (no mainframe) can also support large data volumes and a large number of flexible GUI-based end-user tools, and can relieve the mainframe from ad hoc query processing. However, in general, non-MVS servers are not as reliable as mainframes, are more difficult to manage and integrate into the existing environment, and may require new skills and even new organizational structures.

From the architectural viewpoint, however, the data warehouse server has to be specialized for the tasks associated with the data warehouse, an main frame can be well suited to be a data warehouse server. Let's look at the hardware features that make a server-whether it is mainframe, UNIX-, or NT-based-an appropriate technical solution for the data warehouse.

To begin with, the data warehouse server has to be able to support large data volumes and complex query processing. In addition, it has to be scalable, since the data warehouse is never finished, as new user requirements, new data sour and more historical data are continuously incorporated into the ware-house, a clear as the user population of the data warehouse continues to grow. Therefore, a clear -requirement for the data warehouse server is the scalable high performance data loading and ad hoc query processing as well as the ability to support databases in a reliable, efficient fashion. Chapter 4 briefly touched on various design points to enable server specialization for scalability in performance throughput, user support, and very large database (VLDB) processing.

Balanced Approach

An important design point when selecting a scalable computing platform is the right balance between all computing component example, between the number of processors in a multiprocessor system an the I/O bandwidth. Remember that the lack of balance in a system inevitably results in a bottleneck!

Typically, when a hardware platform is sized to accommodate the data house, this sizing is frequently focused on the number and size of disks. A typical disk configuration. Includes 2.5 to 3 times the amount of raw important consideration-disk throughput comes from the actual number of disks, and not the total disk space. Thus, the number of disks has direct on data parallelism. To balance the system, it is very important to correct number of processors to efficiently handle all disk I/O operations. If this allocation is not balanced, an expensive data warehouse platform can rapidly become CPU-bound. Indeed, since various processors have widely performance ratings and thus can support a different number of CPU, data warehouse designers should carefully analyze the disk I/O processor capabilities to derive an efficient system configuration. For if it takes a CPU rated at 10 SPECint to efficiently handle one 3-Glry- _ drive, then a single 30 SPECint processor in a multiprocessor system can handle three disk drives. Knowing how much data needs to be processed, should give you an idea of how big the multiprocessor 'system should be. A consider-ation is related to disk controllers. A disk controller can support a -amount of data throughput (e.g., 20 Mbytes/s). Knowing the per-disk through-put ratio and the total number of disks can tell you how many controller given type should be config-ured in the system.

The idea of a balanced approach can (and should) be carefully extended to all system components. The resulting system configuration will easily handle known workloads and provide a balanced and scalable computing platform for future growth.

Optimal hardware architecture for parallel query scalability

An important consideration when selecting a hardware platform for a data warehouse is the -ability. Therefore, a frequent approach to system selection is to take of hardware parallelism that comes in the form of shared-memory symmetric multipro-cessors (SMPs), clusters, and shared-nothing distributed-memory system terns (MPPs). As was shown in Chap. 3, the scalability of these systems can be seriously affected by the system-architecture-induced data skew. This architecture-induced data skew is more severe in the low-density asymmetric connection architectures (e.g., daisy-chained, 2-D and 3-D mesh), and is virtu-ally nonexistent in symmetric connection architectures (e.g., cross-bar switch). Thus, when selecting a hardware platform for a data warehouse, take into account the fact that the system-architecture-induced data skew can over-power even the best data layout for parallel query execution, and can force an expen-sive parallel computing system to process queries serially.

Data warehouse and DBMS Specialization

To reiterate, the two important challenges facing the developers of data ware-houses are the very large size of the databases and the need to process complex ad hoc queries in a relatively short time. Therefore, among the most important requirements for the data warehouse DBMS are performance, throughput, and scalability.

The majority of established RDBMS vendors have imple-mented various degrees of parallelism in their respective products. Although any relational database management system-such as DB2, Oracle, Informix, or Sybase--supports parallel database processing, some of these products have been architect to better suit the specialized requirements of the data warehouse.

In addition to the "traditional" relational DBMSs, there are databases that have been optimized specifically for data warehousing, such as Red Brick Warehouse from Red Brick Systems.

Communications Infrastructure

When planning for a data warehouse, one often-neglected aspect of the archi-tecture is the cost and efforts associated with bringing access to corporate data directly to the desktop. These costs and efforts could be significant, since many large organiza-tions do not have a large user population with direct electronic access to information, and since a typical data warehouse user requires a rela-tively large bandwidth to interact with the data warehouse and retrieve a sig-nificant amount of data for the analysis. This may mean that communications networks have to be expanded, and new hardware and software may have to be purchased.

Implementation Considerations

A data warehouse cannot be simply bought and installed-its implementation requires the integration of many products within a data warehouse. The caveat here is that the necessary customization drives up the cost of implementing a data warehouse. To illustrate the complexity of the data warehouse implemen-tation, let's discuss the logical steps needed to build a data warehouse:

- Collect and analyze business requirements.
- Create a data model and a physical design for the data warehouse.
- Define data sources.
- Choose the database technology and platform for the warehouse.

- Extract the data from the operational databases, transform it, and clean it up.
- And load it into the database.
- Choose database access and reporting tools.
- Choose database connectivity software.
- Choose data analysis and presentation software.
- Update the data warehouse.

When building the warehouse, these steps must be performed within the constraints of the current state of data warehouse technologies. .

Access Tools

Currently, no single tool on the market can handle all possible data warehouse Access needs. Therefore, most implementations rely on a suite of tools. The best way to choose this suite includes the definition of different types of access. The data and selecting the best tool for that kind of access. Examples of access types include

- Simple tabular form reporting
- Ranking.
- Multivariable analysis
- Time series analysis
- Data visualization, graphing, charting, and pivoting. Complex textual search
- Statistical analysis
- Artificial intelligence techniques for testing of hypothesis, trends discovery definition, and validation of data clusters and segments
- Information mapping (i.e., mapping of spatial data in geographic' information systems)
- Ad hoc user-specified queries
- Predefined repeatable queries
- Interactive drill-down reporting and analysis
- Complex queries with multitable joins, multilevel subqueries, and sophisticated search criteria

In addition, certain business requirements often exceed existing tool capabili-ties and may require building sophisticated applications to retrieve and analyze warehouse data. These applications often take the form of custom-developed screens and reports that retrieve frequently used data and format it in a pre-defined standardized way. This approach may be very useful for those data warehouse users who are not yet comfortable with ad hoc queries.

There are a number of query tools on the market today. Many of these tools are designed to easily compose and execute ad hoc queries and build cus-tomized reports with little knowledge of the underlying database technology, SQL, or even the data model (i.e., Impromptu from Cognos, Business Objects, etc.), while others (e.g., Andyne's GQL) provide relatively low-level capabilities for an expert user to develop complex ad hoc queries in a fashion similar to developing SQL queries for relational databases. Business requirements that exceed the capabilities of ad hoc query and reporting tools are fulfilled by dif-ferent classes of tools: OLAP and data mining tools.

Discussions

- Write short notes on:
  - Access tools
  - Hardware platforms
  - Balanced approach
- Discuss Data warehouse and DBMS specialization
- Explain Optimal hardware architecture for parallel query scalability
- Describe the Technical issues, which are to be considered while building a data warehouse.
- Explain the **Implementation Considerations**, which are taken into account while building a data warehouse

References

1. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
2. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
3. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
4. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004
5. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

LESSON 11
BENEFITS OF DATA WAREHOUSING

Structure

- Objective
- Introduction
- Benefits of Data warehousing
- Tangible Benefits
- Intangible Benefits
- Problems with data warehousing
- Criteria for a data warehouse

Objective

The aim of this lesson is to study various benefits provided by a data warehouse. You will also learn about the problems with data warehousing and the criteria for Relational Databases for building a data warehouse

Introduction

Today's data warehouse is a user-accessible database of historical and functional company information fed by a mainframe. Unlike most systems, it's set up according to business rather than computer logic. It allows users to dig and churn through large caverns of important consumer data, looking for relationships and making queries. That process-where users sift through piles of facts and figures to discover trends and patterns that suggest new business opportunities-is called data mining.

All that shines is not gold, however. Data warehouses are not the quick-hit fix that some assume them to be. A company must commit to maintaining a data warehouse, making sure all of the data is accurate and timely.

Benefits and rewards abound for a company that builds and maintains a data warehouse correctly. Cost savings and increases in revenue top the list for hard returns. Add to that an increase in analysis of marketing databases to cross-sell products, less computer storage on the mainframe and the ability to identify and keep the most profitable customers while getting a better picture of who they are, and it's easy to see why data warehousing is spreading faster than a rumor of gold at the old mill.

For example, the telecom industry uses data warehouses to target customers who may want certain phone services rather than doing "blanket" phone and mail campaigns and aggravating customers with unsolicited calls during dinner.

Some of the soft benefits of data warehousing come in the technology's effect on users. When built and used correctly, a warehouse changes users' jobs, granting them faster access to more accurate data and allowing them to give better customer service.

A company must not forget, however, that the goal for any data warehousing project is to lower operating costs and generate revenue—this is an investment, after all, and quantifiable ROI should be expected over time. So if the data warehousing effort at your organization is not striking it rich, you might want to ask your data warehousing experts if they've ever heard of fool's gold.

Benefits of Data Warehousing

Data warehouse usage includes

- Locating the right information
- Presentation of information (reports, graphs). Testing of hypothesis
- Discovery of information
- Sharing the analysis

Using better tools to access data can reduce outdated, historical data. Arise; users can obtain the data when they need it most, often during business. Decision processes, not on a schedule predetermined months earlier by the department and computer operations staff.

Data warehouse architecture can enhance overall availability of business intelligence data, as well as increase the effectiveness and timeliness of business decisions.

Tangible Benefits

Successfully implemented data warehousing can realize some significant tangible benefits. For example, conservatively assuming an improvement in out-of-stock conditions in the retailing business that leads to 1 percent increase in sales can mean a sizable cost benefit (e.g., even for a small retail business with $200 million in annual sales, a conservative 1 percent improvement in salary yield additional annual revenue of $2 million or more). In fact, several retail enterprises claim that data warehouse implementations have improved out-of stock conditions to the extent that sales increases range from 5 to 20 percent. This benefit is in addition to retaining customers who might not have returned if, because of out-of-stock problems, they had to do business with other retailers.

Other examples of tangible benefits of a data warehouse initiative include the following:

- Product inventory turnover is improved.
- Costs of product introduction are decreased with improved selection of target markets.
- More cost-effective decision making is enabled by separating (ad hoc) query
- Processing from running against operational databases.
- Better business intelligence is enabled by increased quality and flexibility of market analysis available through multilevel data structures, which may range from detailed to highly summarize. For example, determining the effectiveness of marketing programs allows the elimination of weaker programs and enhancement of stronger ones.
- Enhanced asset and liability management means that a data warehouse can provide a "big" picture of enterprise wide

purchasing and inventory patterns, and can indicate otherwise unseen credit exposure and opportunities for cost savings.

Intangible Benefits

In addition to the tangible benefits outlined above, a data warehouse provides a number of intangible benefits. Although they are more difficult to quantify, intangible benefits should also be considered when planning for the data ware-house. Examples of intangible benefits are:

1. Improved productivity, by keeping all required data in a single location and eliminating the rekeying of data
2. Reduced redundant processing, support, and software to support over- lapping decision support applications.
3. Enhanced customer relations through improved knowledge of individual requirements and trends, through customization, improved communica-tions, and tailored product offerings
4. Enabling business process reengineering-data warehousing can provide useful insights into the work processes themselves, resulting in developing breakthrough ideas for the reengineering of those processes

Problems with Data Warehousing

One of the problems with data mining software has been the rush of companies to jump on the band wagon as

these companies have slapped 'data warehouse' labels on traditional transaction-processing products, and co-opted the lexicon of the industry in order to be considered players in this fast-growing category.

Chris Erickson, president and CEO of Red Brick (HPCwire, Oct. 13, 1995)

Red Brick Systems have established a criteria for a relational database management system (RDBMS) suitable for data warehousing, and documented 10 specialized requirements for an RDBMS to qualify as a relational data warehouse server, this criteria is listed in the next section.

According to Red Brick, the requirements for data warehouse RDBMSs begin with the loading and preparation of data for query and analysis. If a product fails to meet the criteria at this stage, the rest of the system will be inaccurate, unreliable and unavailable.

Criteria for a Data Warehouse

The criteria for data warehouse RDBMSs are as follows:

- **Load Performance -** Data warehouses require incremental loading of new data on a periodic basis within narrow time windows; performance of the load process should be measured in hundreds of millions of rows and gigabytes per hour and must not artificially constrain the volume of data required by the business.
- **Load Processing -** Many steps must be taken to load new or updated data into the data warehouse including data conversions, filtering, reformatting, integrity checks, physical storage, indexing, and metadata update. These steps must be executed as a single, seamless unit of work.
- **Data Quality Management -** The shift to fact-based management demands the highest data quality. The warehouse must ensure local consistency, global consistency, and referential integrity despite "dirty" sources and massive database size. While loading and preparation are necessary steps, they are not sufficient. Query throughput is the measure of success for a data warehouse application. As more questions are answered, analysts are catalyzed to ask more creative and insightful questions.
- **Query Performance -** Fact-based management and ad-hoc analysis must not be slowed or inhibited by the performance of the data warehouse RDBMS; large, complex queries for key business operations must complete in seconds not days.
- **Terabyte Scalability -** Data warehouse sizes are growing at astonishing rates. Today these range from a few to hundreds of gigabytes, and terabyte-sized data warehouses are a near-term reality. The RDBMS must not have any architectural limitations. It must support modular and parallel management. It must support continued availability in the event of a point failure, and must provide a fundamentally different mechanism for recovery. It must support near-line mass storage devices such as optical disk and Hierarchical Storage Management devices. Lastly, query performance must not be dependent on the size of the database, but rather on the complexity of the query.
- **Mass User Scalability -** Access to warehouse data must no longer be limited to the elite few. The RDBMS server must support hundreds, even thousands, of concurrent users while maintaining acceptable query performance.
- **Networked Data Warehouse -** Data warehouses rarely exist in isolation. Multiple data warehouse systems cooperate in a larger network of data warehouses. The server must include tools that coordinate the movement of subsets of data between warehouses. Users must be able to look at and work with multiple warehouses from a single client workstation. Warehouse managers have to manage and administer a network of warehouses from a single physical location.
- **Warehouse Administration -** The very large scale and time-cyclic nature of the data warehouse demands administrative ease and flexibility. The RDBMS must provide controls for implementing resource limits, chargeback accounting to allocate costs back to users, and query prioritization to address the needs of different user classes and activities. The RDBMS must also provide for workload tracking and tuning so system resources may be optimized for maximum performance and throughput. "The most visible and measurable value of implementing a data warehouse is evidenced in the uninhibited, creative access to data it provides the end user.
- **Integrated Dimensional Analysis -** The power of multidimensional views is widely accepted, and dimensional support must be inherent in the warehouse RDBMS to provide the highest performance for relational OLAP tools. The RDBMS must support fast, easy creation of precomputed summaries common in large data warehouses. It also should provide the maintenance tools to automate the creation of these precomputed aggregates. Dynamic

calculation of aggregates should be consistent with the interactive performance needs.

- **Advanced Query Functionality -** End users require advanced analytic calculations, sequential and comparative analysis, and consistent access to detailed and summarized data. Using SQL in a client/server point-and-click tool environment may sometimes be impractical or even impossible. The RDBMS must provide a complete set of analytic operations including core sequential and statistical operations.

Discussions

- Write short notes on:
  - Query Performance
  - Integrated Dimensional Analysis
  - Load Performance
  - Mass User Scalability
  - Terabyte Scalability
- Discuss in brief Criteria for a data warehouse
- Explain Tangible benefits. Provide suitable examples for explanation.
- Discuss various problems with data warehousing
- Explain Intangible benefits. Provide suitable examples for explanation.
- Discuss various benefits of a Data warehouse.

References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
4. **Corey, Michael,** *Oracle8 data warehousing,* New Delhi: Tata McGraw- Hill Publishing, 1998.
5. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.
6. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004

Notes

LESSON 12
PROJECT MANAGEMENT PROCESS, SCOPE STATEMENT

Structure

- Objective
- Introduction
- Project Management Process
- Scope Statement
- Project planning
- Project scheduling
- Software Project Planning
- Decision Making

Objective

The main objective of this lesson is to introduce you with various topics related to Process Management Process. It also includes the need of Scope statement; project planning, Project Scheduling Software Project Planning and decision-making.

Introduction

Now here we will talk about project management when we haven't yet defined the term "project", For this definition, we for to the source - the project management institute, The Project management institute is a nonprofit professional organization dedicated to advancing the state of the art in the management of projects.

Membership is open to anyone actively engaged or interested in the application practice teaching, and researching of project management principles and techniques.

According to the book. A Guide to the project management body knowledge, published by the project management institute standards committee. A project is a temporary endeavor undertaken to create u unique product or service.

One key point here is temporary endeavor. Any project must have a defined start and end, If you are unclear about what must be done to complete the project, don't start it. This is a sure path to disaster. In addition, if at any point in the project if becomes clear that the end product of the project cannot be delivered then the project should *be* called. This also applies to a data warehouse project. Another key point is to create a unique product or service. You must have a clear idea of what you are building and why it is unique. It is not necessary for you to understand why your product or service is unique from day one. But through the process of developing the project plan the uniqueness of the project. Only with a define end do you have a project that can succeed.

Project Management Process

The project management institute is a nonprofit professional organization dedicated to advancing the state of the art in the management of projects.

A project is a temporary endeavor undertaken to create a unique product or service.

Every project must have a defined start and end. If you are unclear about what must be done to complete the project, don't start it.

If at any point in the project if becomes clear that the end product of the project cannot be delivered then the project should be cancelled. This also applies to a data warehouse project.

One must have a clear idea of what you are building and why it is unique.

It is not necessary to understand why your product or service is unique from day one. But through the process of developing the project plan, the uniqueness of the project. Only with a define end do you have a project that can succeed

In addressing the unique aspect or providing business users with timely access to data amidst constantly changing business conditions, Whether you are embarking on a customer relationship management initiative, a balanced scorecard implementation, a risk management system or other, decision support applications, there is a large body of knowledge about what factors contributed the most to the failures of these type of initiative. We, as an industry: need to leverage this knowledge and learn these lessons so we don't repeat them in our own projects.

A data warehouse is often the foundation for many of decision support initiatives. Many studies have show that a significant reason for the failure of data warehousing

And decision support projects is not failure of the technology, but rather, inappropriate project management including lack of integration, lack of communication and lack of clear linkages to business objectives and to benefits achievement.

The Scope Statement

One of the major proven techniques we can use to help us with this discovery process is called a scope statement. A scope statement is a written document by which you begin to define the job at hand and all the key deliverables. In fact, we feel it is good business practice not to begin work on any project until you have developed a scope statement. No work should begin on, a project until a scope statement has been developed. These are the major elements in the breakdown of a scope statement.

1.Project Title and Description: Every project should have a clear name and description of what you are trying to accomplish.

2.Project Justification: clear description of why this project is being done. What is the goal of the project?

3. Project Key Deliverables: a 1ist of keys items that must be accomplished so this project can be completed. What must be done for us to consider the project done?

4. Project Objective: an additional list of success criteria. These items must be measurable: a good place to put any time, money, or resource constraints.

Think of scope statement as your first stake in the ground. What's important is that a scope statement provides a documented basis for building a common understanding among all the shareholders of the project at hand. It is crucial that you begin to write down and document the project and all its assumptions. If you do not do this, you will get burned. All a stakeholder remembers about a hallway conversation is the deliverable, not the constraints. Many project managers have been burned by making a hallway statement, such as: "If the new network is put into place by July I, feel comfortable saying we can provide you with the legacy data by July 10, not All the stakeholder will remember will remember is the date of July 10, not the constraint associated with it.

A scope stmt provides a documented basis for building a common understanding among all the shareholders of the project at hand.

It is crucial that you begin to write down and document the project and all its assumptions. If you do not do this, you might get into some problem.

Project Planning

Project planning is probably the most time-consuming project management activity. It is a continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available Various different types of plan may be developed to support the main software project plan that is concerned with schedule and budget.

Types of project plan are:

- Quality plan
- Validation plan
- Configuration management plan
- Maintenance plan
- Staff development plan

A project plan is usually of the following structure:

- Introduction
- Project organization
- Risk analysis
- Hardware and software resource requirements
- Work breakdown
- Project schedule
- Monitoring and reporting mechanisms

Three important concepts in project planning are:

- **Activities** in a project should be organised to produce tangible outputs for management to judge progress
- **Milestones** are the end-point of a process activity
- **Deliverables** are project results delivered to customers

The waterfall process allows for the straightforward definition of progress milestones.

During early stages of planning, critical decisions must be made due to estimation results e.g.

- Whether or not to bid on a contract, and if so, for how much
- Whether or not build software, or to purchase existing software that has much of the desired functionality
- Whether or not to subcontract (outsource) some of the software development

The project plan is the actual description of:

- The tasks to be done
- Who is responsible for the tasks
- What order tasks are to be accomplished
- When the tasks will be accomplished, and how long they will take

The Project Plan is updated throughout the project.

Project planning is only part of the overall management of the project. Other project management activities include:

- Risk analysis and management
- Assigning and organizing project personnel
- Project scheduling and tracking
- Configuration management

Project Management (across disciplines) is actually considered a whole separate field from software engineering. The Project Management Institute (http://www.pmi.org) has actually developed an entire Body of Knowledge for project management, called PMBOK. However, there are some aspects of software project management that are unique to the management of software.

There are four dimensions of development:

- People
- Process
- Product
- Technology

The resources a manager should consider during project planning are:

- Human Resources: This includes "overhead", and by far is the most dominant aspect of software costs and effort.
- Software Resources: COTS, in-house developed software and tools, reusable artifacts (e.g. design patterns), historical data (helps with cost/effort estimation)

Note that there is usually not much involving physical resources, except as it relates to overhead.

Project Scheduling

During project scheduling:

- Split the project into tasks and estimate time and resources required to complete each task.
- Organize tasks concurrently to make optimal use of work force
- Minimize task dependencies to avoid delays caused by one task waiting for another to complete

This activity is highly dependent on project managers intuition and experience.

Some scheduling problems are:

- Estimating the difficulty of problems and hence the cost of developing a solution is hard
- Productivity is not proportional to the number of people working on a task
- Adding people to a late project makes it later because of communication overheads
- For further reading on this, look at Frederick Brooks' book "The Mythical Man Month"
- The unexpected always happens; always allow contingency in planning.

Scheduling starts with estimation of task duration:

- Determination of task duration
- Estimation of anticipated problems
- Consideration of unanticipated problems
- Use of previous project experience

Activity charts are used to show task dependencies. They can be used to identify the critical path i.e. the project duration.

Bar charts and activity networks are:

- Graphical notations used to illustrate the project schedule
- Show project breakdown into tasks
- Tasks should not be too small. They should take about a week or two
- Activity charts show task dependencies and the the critical path
- Bar charts show schedule against calendar time
- Used to show activity timelines
- Used to show staff allocations

Software Project Planning

Software project planning is the process of

- Determining what the scope and available resources for a project,
- Estimating the costs, total effort and duration of the project required to develop and deliver the software,
- Determining the feasibility of the project,
- Deciding on whether or not to go forward with project implementation,
- Creating a plan for the successful on-time and within budget delivery of reliable software that meets the client's needs, and
- Updating the plan as necessary throughout the project lifetime.

Planning is primarily the job of the "Project Manager" and its steps are

- Develop Goals
- Identify the end state
- Anticipate problems
- Create alternative courses of action
- Define strategy
- Set policies and procedures

Why do we need to make a project plan?

There are two quotations that I'd like to include here:

- No one plans to fail, they just fail to plan.
- Plan the flight, fly the plan.

Planning is about making a schedule that involves using tools like

- Activity Networks: PERT/CPM
- Gantt Chart
- Tabular Notation

Activity networks show:

- Activities: Task + Duration + Resources
- Milestones: No Duration, No Resources

Critical Path Method

Duration of an activity is calculated with the formula

$$\text{Duration} = (a + 4m + b)/6$$

Where a is the pessimistic, m is the most likely and b is the optimistic estimation of the duration of that activity.

Terminologies relating to timing between activities are:

- ES: early start, soonest an activity can begin
- LS: late start, latest time an activity can begin
- EF: early finish, earliest an activity can finish
- LF: late finish, latest an activity can finish
- Slack: difference between ES and LS, if zero critical path

The scope of the project describes the desired functionality of the software, and what constraints are involved. Therefore, project scope is determined as part of the analysis process by the requirements team. The project manager then uses the scope that has been defined in order to determine project feasibility, and to estimate what costs, total effort and project duration will be involved in engineering the software.

So does that mean that planning should come after requirements? Definitely no. Project planning is actually an iterative process, which starts at the beginning of the project, and continues throughout.

Decision Making

Decision-making during software project planning is simply option analysis. In a project, the manager must achieve a set of essential project goals. He also may have desirable (but not essential) organisational goals. He must plan within organisational constraints. There are usually several different ways of tackling the problem with different results as far as goal achievement is concerned.

Decision-making is usually based on estimated value: by comparing the values of possible alternatives.

A few examples of goals and constraints are:

- **Project goals**: high reliability, maintainability, development within budget, development time <2 years
- **Desirable goals**: staff development, increase organisational profile in application area, reusable software components

- **Organisational constraints**: use of a specific programming language or tool set, limited staff experience in application area.

In decision-making, it is advised to make goals as explicit and (wherever possible) quantified as possible. Then score possible options against each goal, and use a systematic approach to option analysis to select the best option

It always helps to use graphs and visual aids during decision-making. When a single value is used, a list is usually sufficient. When more than one value is considered during decision-making, it is advised to use Polar Graphs for presenting this information.

As a practice, consider the following problem:

You are a project manager and the Rational salesman tells you that if you buy the Rational Suite of tools, it will pay for itself in just a few months.

The software will cost you $9600. You talk to other companies and find that there is a 50% chance that the software will cut defect rework costs by $32000.

There is a 20% chance that there will be no improvement and a 30% chance that negative productivity will increase project costs by $20000.

Should you buy this software package?

Summary

- Good project management is essential for project success
- The intangible nature of software causes problems for management
- Managers have diverse roles but their most significant activities are planning, estimating and scheduling
- Planning and estimating are iterative processes which continue throughout the course of a project
- A project milestone is a predictable state where some formal report of progress is presented to management.
- The primary objective of software project planning is to plan for the successful on-time and within budget delivery of reliable software that meets the client's needs
- Project scope should be determined through requirements analysis and used for planning by the project manager
- The primary resources needed for a software project are the actual developers
- Early on in a project decisions must be made on whether or not to proceed with a project, and whether or not to buy or build the software
- The project tasks, schedule and resource allocation is all part of the Project Plan
- Project planning is only part of the project management process
- Effective management depends on good planning
- Planning and estimating are iterative process
- Milestones should occur regularly
- Option analysis should be carried out to make better decision

Discussions

- Explain the following with related examples:
  - Scope Statement
  - Goals
  - Deliverables
  - Organizational Constraints
  - Decision making
  - Milestones
- What is Project Management Process?
- Describe the need of Project planning. Can a project be completed in time without Project Planning? Explain with an example.
- Explain the significance of Project scheduling
- What is Software Project Planning?
- Describe the contents that are included in a Project Plan. Why do we need to make a project plan?

Reference

1. **Hughes, Bob,** *Software project management,* 2nd ed. New Delhi: Tata McGraw- Hill Publishing, 1999.
2. **Kelkar, S.A.,** *Software project management: a concise study,* New Delhi: Prentice Hall of India, 2002
3. **Meredith, Jack R.; Mantel, Samuel J.,** *Project management: a managerial approach,* New York: John Wiley and Sons, 2002.
4. **Royce, Walker,** *Software project management: a unified framework,* Delhi: Pearson Education Asia, 1998.
5. **Young, Trevor L.,** *Successful project management,* London: Kogan Page, 2002.

LESSON 13
WORK BREAKDOWN STRUCTURE

Structure

- Objective
- Introduction
- Work Breakdown Structure
- How to build a WBS
- To create work breakdown structure
- From WBS to Activity Plan
- Estimating Time

Objective

When you will complete this lesson you should be able to:

- Understand the purpose of a Work Breakdown Structure.
- Construct a WBS

Introduction

Once Project scope stmt is completed, another useful technique is called Work Breakdown Structure.

Work Breakdown Structure is exactly as it sounds a breakdown of all the work that must be done. This includes all deliverables. For e.g.., if you are expected to provide the customer with weekly status reports, this should also be in the structure.

Work Breakdown Structure (WBS)

In order to identify the individual tasks in a project it is useful to create a Work Breakdown Structure. Get the team together and brainstorm all of the tasks in the project, in no particular order. Write them down on sticky notes and put them on a whiteboard. Once everyone has thought of as many tasks as they can arrange the sticky notes into groups under the major areas of activity.

A work breakdown structure (WBS) is a document that shows the work involved in completing your project.

Keep it simple:

- A one-page diagram can convey as much information as a 10-page document.
- As a general rule, each box on your diagram is worth about 80 hours (three weeks) of work.
- About 3-4 levels is plenty of detail for a single WBS diagram. If you need to provide more detail, make another diagram.

Draw a work breakdown structure early in the project. It will help you:

- Clearly explain the project to your sponsor and stakeholders;
- Develop more detailed project plans (schedules, Gantt charts etc) in an organized, structured way.

The work breakdown structure is part of the project plan. The sponsor and critical stakeholders should sign it off before the project begins.

Revise the work breakdown structure whenever design concepts or other key requirements change during the project.

How to build a WBS (a serving suggestion)

1. Identify the required outcome of the project, the overall objective you want to achieve. Write it on a Post-It Note and stick it high on a handy wall.
2. Brainstorm with your project team and sponsor:
   - Identify all the processes, systems and other things you will need to do in the project.
   - Write each component on a separate Post-It Note.
   - Stick the components on the wall, arranged randomly under the first Post-It Note.
   - Every process should have an outcome (otherwise, why follow the process?). Identify which Post-Its represent processes and break them down into their outcomes: products, services, documents or other things you can deliver. Then discard the process Post-Its.
3. Have a break.
4. Over the next few hours (or days, for a large or complex project), return to the wall and move the Post-It Notes around so that they are grouped in a way that makes sense.
   - Every team member should have a chance to do this, either in a group or individually.
   - Everyone has an equal say about what groupings make sense.
   - If an item seems to belong in two or more groups, make enough copies of the Post-It Note to cover all the relevant groups.
5. When the team has finished its sorting process, reconvene the group and agree on titles (labels) for each group of Post-Its.
6. Draw your WBS diagram.

Whether the WBS should be activity-oriented or deliverable-oriented is a subject of much discussion. There are also various approaches to building the WBS for a project. Project management software, when used properly, can be very helpful in developing a WBS, although in early stages of WBS development, plain sticky notes are the best tool (especially in teams).

An example of a work breakdown for painting a room (activity-oriented) is:

- Prepare materials
  - Buy paint
  - Buy a ladder
  - Buy brushes/rollers
  - Buy wallpaper remover

- Prepare room
  - Remove old wallpaper
  - Remove detachable decorations
  - Cover floor with old newspapers
  - Cover electrical outlets/switches with tape
  - Cover furniture with sheets
- Paint the room
- Clean up the room
  - Dispose or store left over paint
  - Clean brushes/rollers
  - Dispose of old newspapers
  - Remove covers

The size of the WBS should generally not exceed 100-200 terminal elements (if more terminal elements seem to be required, use subprojects). The WBS should be up to 3-4 levels deep.

To Create Work Breakdown Structure

Work breakdown structure should be at a level where any stakeholder can understand all the steps it will take to accomplish each task and produce each deliverable.

1. Project Management

1.1 Administrative

1.1.1 Daily Mgmt

1.1.2 Daily Communication

1.1.3 Issue Resolution

1.2 Meeting

1.2.1 Client Meetings

1.2.2 Staff Meetings

2.Technical Editor

2.1 Choosing Technical Editor

2.1.1 Determine skill set

2.1.2 Screen Candidates

2.1.3 Determine appropriate Candidates

2.1.4 Choose Candidates

2.2 Working with Technical Editor

2.2.1 Establish Procedure

2.2.2 Agree on Rule of Engagement.

From WBS to Activity Plan

1. Identify irrelevant tasks
2. Identify resources needed
3. Decide time needed (task time, not lapsed time) for the smallest units of WBS.
4. Identify dependencies (many are already in the WBS, but there are probably also cross-dependencies.) There are two types:
   - Logical dependency: carpet cannot be laid until painting is complete.
   - Resource dependency: if same guy is to paint, he cannot paint two rooms at the same time

Time dependencies:

- End-start (task 1 must end before task 2 can begin)
- Start-start (two tasks must begin at the same time, e. g. simultaneous PR and advertising; forward military movement and logistics train activities.)
- End-end: two tasks must end at the same time. E. G. cooking: several ingredients must be ready at the right moment to combine together.
- Staggered start. Tasks that could possibly be done start-start, but might not actually require it, may be done staggered-start for external convenience-related reasons.

Estimating Time

Lapsed time: end date minus start date. Takes into account weekends, other tasks.

Task time: just the actual days required to do the job.

Hofstadter's law: things take longer than you planned for, even if you took Hofstadter's law into account.

Discussion:

- Write short notes on
  - Activity Plan
  - Dependencies
  - Estimated Time
  - Activity-oriented
- Explain the need and importance of a Work Breakdown structure.
- Construct a WBS for an Information Technology upgrade project. Breakdown the work to at least third level for one of the WBS items. Make notes of questions you had while completing this exercise.
- Create WBS for the following:
  - Hotel Management System
  - Railway Reservation System
  - Hospital Management System

Reference

1. **Carl L. Pritchard**. *Nuts and Bolts Series 1: How to Build a Work Breakdown Structure.*
2. **Project Management Institute**. *Project Management Institute Practice Standard for Work Breakdown Structures.*
3. **Gregory T. Haugan**. *Effective Work Breakdown Structures (The Project Management Essential Library Series).*

More information

1. Noel N Harroff (1995, 2000) "The Work Breakdown Structure". Available at http://www.nnh.com/ev/wbs2.html
2. 4pm.com (undated) "Work Breakdown Structure". Available at http://www.4pm.com/articles/work_breakdown_structure.htm
3. University of Nottingham (November 2002) "Work Breakdown Structure" for building the Compass student portal. Available at http://www.eis.nottingham.ac.uk/compass/workstructure.html

4. NASA Academy of Program and Project Leadership (undated) "Tools: work breakdown structure". Collection of documents available at http://appl.nasa.gov/perf_support/tools/tools_wbs.htm
5. Kim Colenso (2000) "Creating The Work Breakdown Structure". Available at http://www.aisc.com/us/lang_en/library/white_papers/Creating_a_Work_Breakdown_Structure.pdf

Notes

LESSON 14
PROJECT ESTIMATION, ANALYZING PROBABILITY AND RISK

Structure

- Objective
- Introduction
- Project Estimation
- Analyzing probability & Risk

Objective

When you will complete this lesson you should be able to:

- Understand the importance of Project Estimation.
- Study about probability and Risk analysis.

Introduction

To estimate budget and control costs, project managers and their teams must determine what physical resources (people, equipment and materials) and what quantities of those resources are required to complete the project. The nature of project and the organization will affect resource planning. Expert judgment and the availability of alternatives are the only real tools available to assist in resource planning. It is important to have people who have experience and expertise in similar projects and with the organization performing this particular project help determine what resources are necessary.

Accurately planning and estimating software projects is an extremely difficult software management function. Few organizations have established formal estimation processes, despite evidence that suggests organizations without formal estimation are four times more likely to experience cancelled or delayed projects.

Project Estimation

In the 1970s, geologists at Shell were excessively confident when they predicted the presence of oil or gas. They would for example estimate a 40% chance of finding oil, but when ten such wells were actually drilled, only one or two would produce. This overconfidence cost Shell considerable time and money. Shell embarked on a training programme, which enabled the geologists to be more realistic about the accuracy of their predictions. Now, when Shell geologists predict a 40% chance of finding oil, four out of ten are successful.

Software project managers are required to estimate the size of a project. They will usually add a percentage for 'contingency', to allow for their uncertainty. However, if their estimates are overconfident, these 'contingency' amounts may be insufficient, and significant risks may be ignored. Sometimes several such 'contingency' amounts may be multiplied together, but this is a clumsy device, which can lead to absurdly high estimates, while still ignoring significant risks. In some cases, higher management will add additional 'contingency', to allow for the fallibility of the project manager. Game playing around 'contingency' is rife.

This paper proposes that project managers be trained to be more closely aware of the limitations of their own estimation powers, which will lead to project plans that explicitly match the project capabilities and risks, as well as contingency plans that should be more meaningful.

Most organizations have difficulty estimating what it takes to deliver a data warehouse (DW). The time and effort to implement an enterprise DW, let alone a pilot, is highly variable and dependent on a number of factors. The following qualitative discussion is aimed toward estimating a pilot project. It can also be useful in choosing an acceptable pilot that falls within a time schedule that is acceptable to management.

Each of the following points can make a substantial difference in the cost, risk, time and effort of each DW project.

1. **What are the number of internal source systems and database/files?**

Each source system along with its database and files will take additional research, including meetings with those who have knowledge of data. The time to document the results of the research and meeting should also be included in the estimates.

2. **How many business processes are expected for the pilot?**

(Examples: analyze sales, analyze markets, and analyze financial accounts.) a pilot should be limited to just one business process. If management insists on more than one, the time and effort will be proportionally greater.

3. **How many subject areas are expected for the pilot?**

(Examples: customer, supplier/vendor, store/location, product, organizational unit, demographic area of market segment, general ledger account, and promotion/campaign.)

If possible, a pilot should be limited to just one subject area. If management insists on more than one, the time and effort will be proportionally greater.

4. **Will a high-level enterprise model be developed during the pilot?**

Ideally, an enterprise model should have been developed prior to the start of the DW pilot, if the model has not been finished and the pilot requires its completion. The schedule for the pilot must be adjusted.

5. **How many attributes (fields, columns) will be selected for the pilot?**

The more attributes to research understand, clean, integrate and document, the longer the pilot and the greater the effort.

6. **Are the source files well modeled and well documented?**

Documentation is critical to the success of the pilot. Extra time and effort must be included if the source files and databases have not been well documented.

7. Will there be any external data (Lundberg, A. C. Nielsen. Dun and Bradstreet) in the pilot system? Is the external system well documented?

External data is often not well documented and usually does not follow the organization standards. Integrating external data is often difficult and time consuming.

8. Is the external data modeled? (Modeled, up-to-date, accurate, actively being used and comprehensive; a high level accurate and timely mode! Exists: an old, out of date model exists; no model exists)

Without a model, the effort to understand the source external data is significantly greater. It's unlikely that the external data has been modeled, but external data vendors should find the sale of their data easier when they have models that effectively document their products.

9. How much cleaning will the source data require? (Data need no cleaning, minor complexity, transformations, medium/moderate complexity; and very complicated transformation required)

Data cleansing both with and without software tools to aid the process is tedious and time consuming organizations usually overestimate the quality of their data and always underestimate the effort to clean the data.

10. How much integration will be required? (None required, moderate integration required, serious and comprehensive integration required)

An example of integration is pulling customer data together form multiple internal files as well as from external data. The absence of consistent customer identifiers can cause significant work to accurately integrate customer data.

11. What the estimated size of the pilot database?

Data warehouse Axiom #A – Large DW databases (100GB TO 500GB) will always have performance problems. Resolving those problems (living within an update/refresh/backup window, providing acceptable query performance) will always take significant time and effort.

Data Warehouse Axiom #B – A pilot greater than 500 GB is not a pilot; it's a disaster waiting to happen.

12. What is the service level requirement? (Five days/week, eight hours/day; six days/weeks, eighteen hours/day; seven days/week, 24 hours/day)

It is always easier to establish an operational infrastructure as well as a develop the update/refresh/backup scenarios for an 8*% than for a 24*& schedule. It's also easier for operational people and DBAs to maintain a more limited scheduled up time.

13. How frequently will the data be loaded/updated/ refreshed? (Monthly, weekly, daily, hourly)

The more frequent the load/ update/refresh, the greater the performance impact, If real time is every being considered, the requirement is for an operational, not a decision support system.

14. Will it be necessary to synchronize the oper5ational system with the data warehouse?

This is always difficult and will require initial planning, generation procedures and ongoing effort form operations.

15. Will a new hardware platform be required? If so, will to different than the existing platform?

The installation of new hardware always requires planning and execution effort.

If it is to be a new type of platform, operations training and familiarization takes time and efforts, a new operating system requires work by the technical support staff. There are new procedures to follow, new utilities to learn and the shakedown and testing efforts for anything new is always time consuming and riddled with unexpected problems.

16. Will new desktop swill require?

New desktops require installation, testing and possible training of the users of the desktops.

17. Will new network work be required?

If a robust network (one that one handle the additional load form the data warehouse with acceptable performance) is already in place, shaken-out and tested, a significant amount of work and risk will be eliminated.

18. Will network people be available?

If network people are available, it will eliminate the need to recruit or train

19. How many query tools will be chosen?

Each new query tool takes time to train those responsible for support and time to train the end users.

20. Is user management sold on and committed to this project and what is the organization level at which the commitment was made?

If management is not sold on the project, the risk is significantly greater. For the project manager, lack of management commitment means far more difficulty in getting resources (money, involvement) and getting timely answers.

21. Where does the DW project manager's report in the organization?

The higher up the project mangers report, the greater the management commitment, the more visibility and the more the indication that the project is important to the organization.

22. Will the appropriate user be committed and available for the project?

If people important to the project are not committed and available, it will take far longer for the project to complete. User involvement is essential to the success of any DW project.

23. Will knowledgeable application developers (programmers) be available for the migration process?

These programmers need to be available when they are needed unavailability means the project will be delayed.

24. How many trained and experienced programmer/ analysts will be available for system testing?

If these programmer/analysts are not available, the will have to be recruited and / or trained.

25. How many, trained an experienced systems analysts will be assigned to the project full time?

If an adequate number of these trained and experienced system analysts are not available full time, the will have to be recruited and/or trained. A data warehouse project is not a part-time option. It requires the full dedication of team members.

26. Will the DBAs be familiar with the chosen relational database management systems (RDBMS), will they experienced in database design and will they be available for this project full time?

A new RDBMS requires recruitment or training, and there is always a ramp up effort with software as critical as RDBMS. DBAs have different areas of expertise and to all DBAs will have database design skill. Database design is a skill that is mandatory for designing a good DW. The DBA for the DW will be very important and will require a full-time effort.

27. Will technical support people be available for capacity planning, performance monitoring and trouble-shooting?

A Data warehousing of any size will require the involvement of technical support people. Capacity planning is very difficult in this environment as the ultimate size of the DW, user volumes and user access patterns can rarely be anticipated. However, What can be anticipated are performance problems that will require the constant attention of the technical support staff.

28. Will be necessary to get an RFP (Request for Proposal) for any of the data warehouse software tools?

Some organizations require an RFP on software over a specific dollar amount. RFPs take time to write and to evaluate. It's easier to get the information other ways without the need for and RFP.

29. Will a CASE tool be chosen and will the data administrators who use the CASE tool be available and experienced?

A CASE tool should facilitate documentation, communication with the users and provide one of the sources of the DW metadata. Data administrators are usually the folks who use the CASE tools. Their experience with one of the tools will expedite the documentation, communication and meta-data population processes.

30. How many users are expected for the pilot?

The more users, the more training, user support and the more performance issues that can be anticipated. More users means more changes and requests for additional function, and more problems that are uncovered that require resolution.

31. How many queries/day/'user are expected and what is the level of complexity? (simple, moderately complex, very complex)

The higher the volume of queries and the greater their complexity would indicate performance problems more user training required and misunderstanding of the ways to write accurate queries.

32. How comfortable and familiar are the users with desktop computers and with the operating system (Windows, NT, OS/2)? (Very familiar and familiarity) What is the level of sophistication of the users? (Power users, occasional users, little desktop experience)

The users; level of familiarity and sophistication will dictate the amount of training and user support required.

33. Will migration software will be chosen and used for the pilot?

Migration software will require training and time in the learning curve, but should decrease the overall data migration effort.

34. Will a repository be used for the pilot?

A repository of some type is highly recommended of the DW but it will require training, establishing standards for metadata and repository use as well as the effort to populate the repository.

35. Are there any serious security issues? What audit requirements need to be followed of the pilot?

Security and audit require additional procedures (or at the least the implementation of existing procedures) and the time to test and satisfy those procedures.

Analyzing Probability & Risk

In life, most things result in a bell curve. The figure below shows a sample bell curve that measures the likelihood of on-time project delivery. The graph measures the number of projects delivered late, on time, and early.

As you can see in the figure the most likely outcome fails to the center of the curve. This type of graph is skewed in the center; hence, the terminology bell curve is taken from the shape. The following table1 summarizes the data presented in the figure.

What this teaches us is we currently do time estimates incorrectly. That is, trying to predict a single point will never work. The law of averages works against/us.

Table 1: Bell Curve Data Summary

Percentage Of projects	Delivered Days fro, Expected Delivery
25	33 days early or 35 days late
50	25 days early or 25 days late
75	8 days early or 13 late

Table 2 : Three-point Time Estimate Worksheet

Task	Subtasks	Best Case	Most Likely	Worst	
Choosing Technical editor	The Determine Skill set		1.0	3.0	5.0
	Screen candidates		1.0	2.0	3.0
	Choose candidate		0.5	1.0	2.0
Total			2.5	6.0	10.0

We should predict project time estimates like we predict rolling dice. Experience has taught us when a pair of dice is rolled; the most likely number to come up is seven. When you look at alternatives, the odds of a number other than seven coming up are less. You should test a three-point estimate the optimistic view, pessimistic view, and the most likely answer?

Based on those answers. You can determine a time estimate Table 2 shows an example of a three point estimated worksheet.

As you can see from, Table 2 just the task of choosing the technical editor has considerable latitude in possible outcomes, yet each one of these outcomes, yet each one of these outcomes has a chance of becoming reality. Within a given projection many of the tasks would come in on the best case guess and many of the tasks will also come in on the worst case guess. In addition, each one of these outcomes has associated measurable risks.

We recommend you get away fro, single point estimates and move toward three point estimates. By doing this, you will start to get a handle on your true risk. By doing this exercise with your team members you will set everyone thinking about the task and all the associated risks, what if a team member gets sick? What if the computer breaks down what if someone gets pulled away on another tasks? These things do happen and they do affect the project.

You are now also defining the acceptable level of performance. For example, if project team members came in with 25 days to choose a technical editor, we would consider this irresponsible. WE would require a great deal of justification.

Another positive aspect to the three-point estimate is it improves the stakeholder's morale. The customers will begin to feel more comfortable because he or she will have an excellent command on the project. At the same time, when some tasks do fall behind, everyone realizes this should be expected. Because the project takes all outcomes into consideration. You could still come in within the acceptable timelines. Point estimates and improve the level of accuracy.

Discussions

- Write notes on:
  - Integration
  - RFP
  - External Data
- Discuss various points that can make a substantial difference in the cost, risk, time and effort of each DW project.
- Discuss Probability & Risk Analysis

Reference

1. **Hughes, Bob,** *Software project management,* 2nd ed. New Delhi: Tata McGraw- Hill Publishing, 1999.
2. **Kelkar, S.A.,** *Software project management: a concise study,* New Delhi: Prentice Hall of India, 2002
3. **Meredith, Jack R.; Mantel, Samuel J.,** *Project management: a managerial approach,* New York: John Wiley and Sons, 2002.
4. **Royce, Walker,** *Software project management: a unified framework,* Delhi: Pearson Education Asia, 1998.
5. **Young, Trevor L.,** *Successful project management,* London: Kogan Page, 2002.

Notes

LESSON 15
MANAGING RISK: INTERNAL AND EXTERNAL, CRITICAL PATH ANALYSIS

Structure

Objective

Introduction

Risk Analysis

Risk management

Objective

When you have completed this lesson you should be able to:

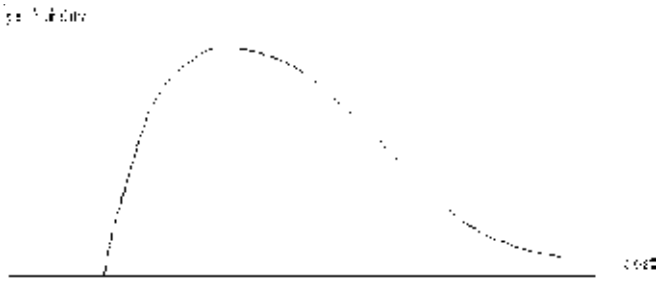
- Understand the importance of Project Risk Management.
- Understand what is Risk.
- Identify various types of Risks.
- Describe Risk Management Process
- Discuss Critical Path Analysis

Introduction

Project Risk Management is the art and science of identifying, assigning and responding to risk throughout the life of a project and in the best interests of meeting project objectives. Risk management can often result in significant improvements in the ultimate success of projects. Risk management can have a positive impact on selecting projects, determining the scope of projects, and developing realistic schedules and cost estimates. It helps stakeholders understand the nature of the project, involves team members in defining strengths and weaknesses, and help to integrate the other project management areas.

Risk Analysis

At a given stage in a project, there is a given level of knowledge and uncertainty about the outcome and cost of the project. The probable cost can typically be expressed as a skewed bell curve, since although there is a minimum cost; there is no maximum cost.



There are several points of particular interest on this curve:

Minimum	The lowest possible cost.
Mode	The most likely cost. This is the highest point on the curve.
Median	The midway cost of n projects. (In other words, n/2 will cost less than the median, and n/2 will cost more.)
Average	The expected cost of n similar projects, divided by n.
Reasonable maximum	The highest possible cost, to a 95% certainty.
Absolute maximum	The highest possible cost, to a 100% certainty.

On this curve, the following sequence holds:
Minimum < mode < median < average < reasonable maximum < absolute maximum

Note the following points:

- The absolute maximum cost may be infinite, although there is an infinitesimal tail. For practical purposes, we can take the reasonable maximum cost. However, the reasonable maximum may be two or three times as great as the average cost.
- Most estimation algorithms aim to calculate the mode. This means that the chance that the estimates will be achieved on a single project is much less than 50%, and the chances that the total estimates will be achieved on a series of projects is even lower. (In other words, you do not gain as much on the roundabouts as you lose on the swings.)
- A tall thin curve represents greater certainty, and a low broad curve represents greater uncertainty.
- Risk itself has a negative value. This is why investors demand a higher return on risky ventures than on 'gilt-edged' securities. Financial number-crunchers use a measure of risk called 'beta'.
- Therefore, any information that reduces risk has a positive value.

This analysis yields the following management points:

- A project has a series of decision points, at each of which the sponsors could choose to cancel.
- Thus at decision point k, the sponsors choose between continuation, which is then expected to cost Rk, and cancellation, which will cost Ck.
- The game is to reduce risk as quickly as possible, and to place decisions at the optimal points.
- This means we have to understand what specific information is relevant to the reduction of risk, plan the project so that this information emerges as early as possible, and to place the decision points immediately after this information is available.

Risk Management

Risk management is concerned with identifying risks and drawing up plans to minimize their effect on a project. A risk is a probability that some adverse circumstance will occur:

- **Project risks** affect schedule or resources
- **Product risks** affect the quality or performance of the software being developed
- **Business risks** affect the organization developing or procuring the software

The risk management process has four stages:

- Risk **identification**: Identify project, product and business risks
- Risk **analysis**: Assess the likelihood and consequences of these risks
- Risk **planning**: Draw up plans to avoid or minimize the effects of the risk
- Risk **monitoring**: Monitor the risks throughout the project

Some risk types that are applicable to software projects are:

- Technology risks
- People risks
- Organizational risks
- Requirements risks
- Estimation risks

Risk Analysis

- Assess probability and seriousness of each risk
- Probability may be very low, low, moderate, high or very high
- Risk effects might be catastrophic, serious, tolerable or insignificant

Risk planning steps are:

1. Consider each risk and develop a strategy to manage that risk
2. Avoidance strategies: The probability that the risk will arise is reduced
3. Minimization strategies: The impact of the risk on the project or product will be reduced
4. Contingency plans: If the risk arises, contingency plans are plans to deal with that risk

To mitigate the risks in a software development project, a management strategy for every identified risk must be developed.

Risk monitoring steps are:

- Assess each identified risks regularly to decide whether or not it is becoming less or more probable
- Also assess whether the effects of the risk have changed
- Each key risk should be discussed at management progress meetings

Managing Risks: Internal & External

When a closer look at risk. When you do get caught this is typically due to one of three situations:

1. Assumptions – you get caught by unvoiced assumptions which were never spelled out.
2. Constraints – you get caught by restricting factors, which were not fully understood.
3. Unknowns – items you could never predict, by they are acts of God or human errors.

The key to risk management Is to do our best to identify the source of all risk and the likelihood of its happening.,

For example when we project plan, we typically do not take work stoppages into account. But if we were working for and airline that was under threat of major strike, we might re evaluate the likelihood of losing valuable project time.

Calculate the cost to the project if the particular risk happens and make decision. You can decide either to accept it, find a way to avoid it or to prevent it. Always look for ways around the obstacles.

Internal and External Risks

Duncan Nevison lists the following types of internal risks:

1. Project Characteristics
   Schedule Bumps
   Cost Hiccups
   Technical Surprises
2. Company politics
   Corporate Strategy Change
   Departmental Politics
3. Project Stakeholders
   Sponsor
   Customer
   Subcontractors
   Project Team

As well as the following external risks;

1. Economy
   - Currency Rate Change
   - Market Shift
   - Competitors Entry OR Exit
   - Immediate Competitive Actions
   - Supplier Change

2. Environment
   - Fire, Famine, Flood
   - Pollution
   - Raw Materials
3. Government
   - Change in Law
   - Change in Regulation

By going through at this risk, you get a sense fo all the influences that may impact your particular project. You should take the time to assess and reassess these. For example; if your project is running severely behind schedule, is there another vendor waiting to try to take the business? If your project is running way over budget, is there a chance, the funding may get cut? We must always be aware of the technology with which we are working. Familiarity with technology is important; we need to know if what we are working with is a new release of the software or a release that has been out for a long time.

Critical Path Analysis

After your determination what must be done and how long it will take, you are ready to start looking for your critical paths and dependencies. These critical paths are yet another form of risk within a project. For example, there a re inherent dependencies among many project activities

A technical editor must be selected before the editing cycle of a chapter can be completed. These are examples of dependency analysis.

Let's say we are working on a project with three unique lists of activities associated with it. Each unique path (A, B, C) of activities is represented based on its dependencies. Each cell represents the number of days that activity would take.

By adding all the rows together, you tell the duration of each path. This is shown in table 3.

Start A represents a part of the project with three steps, which will take a total of eight days. Start B represents a part of the project with three steps, which will take a total of six days. Start C represents a path with two steps, which will take a total of 20 days

Unique Task	Part # 1	Part # 2	Part # 3	Total
Start A	1	3	4	8 days
Start B	1	2	3	6 days
Start C	**15**	**5**		**20 days**

Table 3 : Critical Path Analysis

The critical path is Start C. You must begin this as soon as possible. In fact, this tells us the sooner this project can be done is 20 days. If you do not start the activity that takes 15 days first, it will delay the entire project ending one day for each day you wait.

Self Test

As set of multiple choices given with every question, choose the correct answer for the following question.

1. Which on is not an environmental risk
   a. Technological changes
   b. Legal requirements
   c. Government decisions
   d. Office politics
2. What is a economic risk faced by project
   a. Competitor's exit or entry
   b. Supplier's change
   c. Market shift
   d. All of the above
3. Which of these is not internal risk
   a. Policy change
   b. Department structures
   c. Sponsor
   d. None of these
4. Which of these is an internal risk
   a. Customer expectations
   b. Regulatory changes
   c. Market factors
   d. Monetary
5. The project estimation can get delayed because fo the reason
   a. Wrong assumptions
   b. Constraints
   c. Unknown factors
   d. All of the above
6. Which one is not an environmental risk
   a. Technological changes
   b. Legal requirements
   c. Government decisions
   d. Office politics
7. What is a economic risk faced by project
   a. Competitor's exit or entry
   b. Suppliers change
   c. Market shift
   d. All of the above
8. Which of these is not an internal risk
   a. Policy change
   b. Department Structures
   c. Sponsor
   d. None of these
9. Which of these is an internal risk
   a. Customer expectations
   b. Regulatory changes
   c. Market factors
   d. Monetary

10. The project estimation can get delayed because of the reason
    a. Wrong assumptions
    b. Constraints
    c. Unknown factors
    d. All of the above

Reference

1. **Hughes, Bob,** *Software project management,* 2nd ed. New Delhi: Tata McGraw- Hill Publishing, 1999.
2. **Kelkar, S.A.,** *Software project management: a concise study,* New Delhi: Prentice Hall of India, 2002
3. **Meredith, Jack R.; Mantel, Samuel J.,** *Project management: a managerial approach,* New York: John Wiley and Sons, 2002.
4. **Royce, Walker,** *Software project management: a unified framework,* Delhi: Pearson Education Asia, 1998.
5. **Young, Trevor L.,** *Successful project management,* London: Kogan Page, 2002.

Notes

LESSON 16
DATA MINING CONCEPTS

Structure

- Objective
- Introduction
- Data mining
- Data mining background
- Inductive learning
- Statistics
- Machine Learning

Objective

When you have completed this lesson you should be able to:

- Understand the basic concepts of data mining.
- Discuss data mining background
- Learn various concepts like Inductive learning, Statistics and machine learning

Introduction

Data mining is a discovery process that allows users to understand the substance of and the relationships between, their data. Data mining uncovers patterns and rends in the contents of this information.

I will briefly review the state of the art of this rather extensive field of data mining, which uses techniques from such areas as machine learning, statistics, neural networks, and genetic algorithms. I will highlight the nature of the information that is discov-ered, the types of problems faced in databases, and potential applications.

Data Mining

The past two decades has seen a dramatic increase in the amount of information or data being stored in electronic format. This accumulation of data has taken place at an explosive rate. It has been estimated that the amount of information in the world doubles every 20 months and the size and number of databases are increasing even faster. The increase in use of electronic data gathering devices such as point-of-sale or remote sensing devices has contributed to this explosion of available data. Figure 1 from the Red Brick Company illustrates the data explosion.

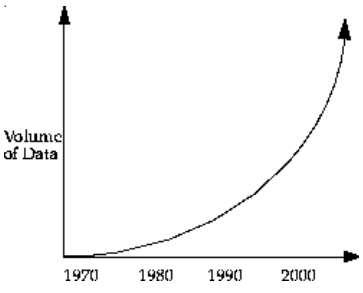


Figure 1: The Growing Base of Data

Data storage became easier as the availability of large amounts of computing power at low cost i.e., the cost of processing power and storage is falling, made data cheap. There was also the introduction of new machine learning methods for knowledge representation based on logic programming etc. in addition to traditional statistical analysis of data. The new methods tend to be computationally intensive hence a demand for more processing power.

Having concentrated so much attention on the accumulation of data the problem was what to do with this valuable resource? It was recognized that information is at the heart of business operations and that decision-makers could make use of the data stored to gain valuable insight into the business. Database Management systems gave access to the data stored but this was only a small part of what could be gained from the data. Traditional on-line transaction processing systems, OLTPs, are good at putting data into databases quickly, safely and efficiently but are not good at delivering meaningful analysis in return. Analyzing data can provide further knowledge about a business by going beyond the data explicitly stored to derive knowledge about the business. This is where Data Mining or Knowledge Discovery in Databases (KDD) has obvious benefits for any enterprise.

The term data mining has been stretched beyond its limits to apply to any form of data analysis. Some of the numerous definitions of Data Mining, or Knowledge Discovery in Databases are:

Data Mining, or Knowledge Discovery in Databases (KDD) as it is also known, is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. This encompasses a number of different technical approaches, such as clustering, data summarization, learning classification rules, finding dependency net works, analyzing changes, and detecting anomalies.

William J Frawley, Gregory Piatetsky-Shapiro and Christopher J Matheus

Data mining is the search for relationships and global patterns that exist in large databases but are 'hidden' among the vast amount of data, such as a relationship between patient data and their medical diagnosis. These relationships represent valuable knowledge about the database and the objects in the database and, if the database is a faithful mirror, of the real world registered by the database.

Marcel Holshemier & Arno Siebes (1994)

The analogy with the mining process is described as:

Data mining refers to "using a variety of techniques to identify nuggets of information or decision-making

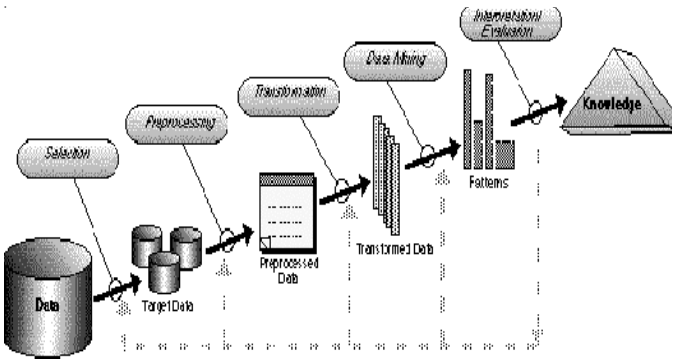
knowledge in bodies of data, and extracting these in such a way that they can be put to use in the areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but as it stands of low value as no direct use can be made of it; it is the hidden information in the data that is useful"

Clementine User Guide, a data mining toolkit

Basically data mining is concerned with the analysis of data and the use of software techniques for finding patterns and regularities in sets of data. It is the computer, which is responsible for finding the patterns by identifying the underlying rules and features in the data. The idea is that it is possible to strike gold in unexpected places as the data mining software extracts patterns not previously discernable or so obvious that no one has noticed them before.

Data mining analysis tends to work from the data up and the best techniques are those developed with an orientation towards large volumes of data, making use of as much of the collected data as possible to arrive at reliable conclusions and decisions. The analysis process starts with a set of data, uses a methodology to develop an optimal representation of the structure of the data during which time knowledge is acquired. Once knowledge has been acquired this can be extended to larger sets of data working on the assumption that the larger data set has a structure similar to the sample data. Again this is analogous to a mining operation where large amounts of low-grade materials are sifted through in order to find something of value.

The following diagram summarizes the some of the stages/ processes identified in data mining and knowledge discovery by Usama Fayyad & Evangelos Simoudis, two of leading exponents of this area.



The phases depicted start with the raw data and finish with the extracted knowledge, which was acquired as a result of the following stages:

- **Selection -** selecting or segmenting the data according to some criteria e.g. all those people who own a car, in this way subsets of the data can be determined.
- **Preprocessing -** this is the data cleansing stage where certain information is removed which is deemed unnecessary and may slow down queries for example unnecessary to note the sex of a patient when studying pregnancy. Also the data is reconfigured to ensure a consistent format as there is a possibility of inconsistent formats because the data is drawn from several sources e.g. sex may recorded as f or m and also as 1 or 0.
- **Transformation -** the data is not merely transferred across but transformed in that overlays may added such as the demographic overlays commonly used in market research. The data is made useable and navigable.
- **Data mining -** this stage is concerned with the extraction of patterns from the data. A pattern can be defined as given a set of facts(data) *F*, a language *L*, and some measure of certainty *C* a pattern is a statement *S* in *L* that describes relationships among a subset *Fs* of *F* with a certainty *c* such that *S* is simpler in some sense than the enumeration of all the facts in *Fs*.
- **Interpretation and evaluation -** the patterns identified by the system are interpreted into knowledge which can then be used to support human decision-making e.g. prediction and classification tasks, summarizing the contents of a database or explaining observed phenomena.

Data Mining Background

Data mining research has drawn on a number of other fields such as inductive learning, machine learning and statistics etc.

Inductive Learning

Induction is the inference of information from data and inductive learning is the model building process where the environment i.e. database is analyzed with a view to finding patterns. Similar objects are grouped in classes and rules formulated whereby it is possible to predict the class of unseen objects. This process of classification identifies classes such that each class has a unique pattern of values, which forms the class description. The nature of the environment is dynamic hence the model must be adaptive i.e. should be able learn.

Generally, it is only possible to use a small number of properties to characterize objects so we make abstractions in that objects, which satisfy the same subset of properties, are mapped to the same internal representation.

Inductive learning where the system infers knowledge itself from observing its environment has two main strategies:

- **Supervised learning** - this is learning from examples where a teacher helps the system construct a model by defining classes and supplying examples of each class. The system has to find a description of each class i.e. the common properties in the examples. Once the description has been formulated the description and the class form a classification rule, which can be used to predict the class of previously unseen objects. This is similar to discriminate analysis as in statistics.
- **Unsupervised learning** - this is learning from observation and discovery. The data mine system is supplied with objects but no classes are defined so it has to observe the examples and recognize patterns (i.e. class description) by itself. This system results in a set of class descriptions, one for each class discovered in the environment. Again this similar to cluster analysis as in statistics.

Induction is therefore the extraction of patterns. The quality of the model produced by inductive learning methods is such that the model could be used to predict the outcome of future situations in other words not only for states encountered but rather for unseen states that could occur. The problem is that most environments have different states, i.e. changes within, and it is not always possible to verify a model by checking it for all possible situations.

Given a set of examples the system can construct multiple models some of which will be simpler than others. The simpler models are more likely to be correct if we adhere to Ockhams razor, which states that if there are multiple explanations for a particular phenomena it makes sense to choose the simplest because it is more likely to capture the nature of the phenomenon.

Statistics

Statistics has a solid theoretical foundation but the results from statistics can be overwhelming and difficult to interpret, as they require user guidance as to where and how to analyze the data. Data mining however allows the expert's knowledge of the data and the advanced analysis techniques of the computer to work together.

Statistical analysis systems such as SAS and SPSS have been used by analysts to detect unusual patterns and explain patterns using statistical models such as linear models. Statistics have a role to play and data mining will not replace such analyses but rather they can act upon more directed analyses based on the results of data mining. For example statistical induction is something like the average rate of failure of machines.

Machine Learning

Machine learning is the automation of a learning process and learning is tantamount to the construction of rules based on observations of environmental states and transitions. This is a broad field, which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc. A learning algorithm takes the data set and its accompanying information as input and returns a statement e.g. a concept representing the results of learning as output. Machine learning examines previous examples and their outcomes and learns how to reproduce these and make generalizations about new cases.

Generally a machine learning system does not use single observations of its environment but an entire finite set called the training set at once. This set contains examples i.e. observations coded in some machine-readable form. The training set is finite hence not all concepts can be learned exactly.

Discussions

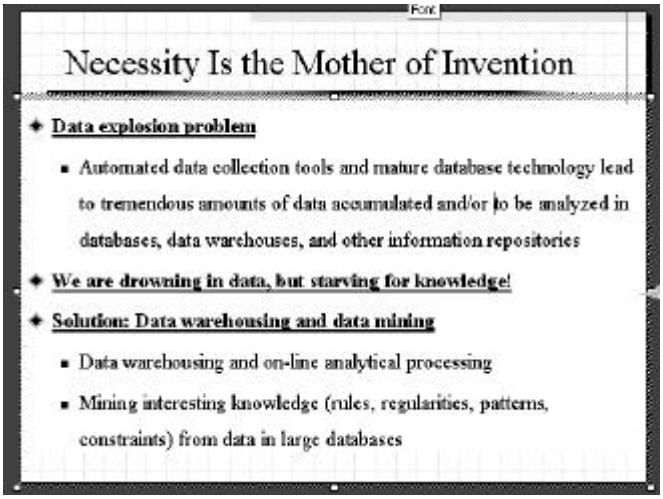
- Write short notes on:
  - KDD
  - Machine Learning
  - Inductive Learning
  - Statistics
- What is Data mining?
- Illustrate various stages of Data mining with the help of a diagram.
- "Inductive learning is a system that infers knowledge itself from observing its environment has two main strategies". Discuss these two main strategies.
- Explain the need of Data mining.
- Explain how Data mining helps in decision-making?

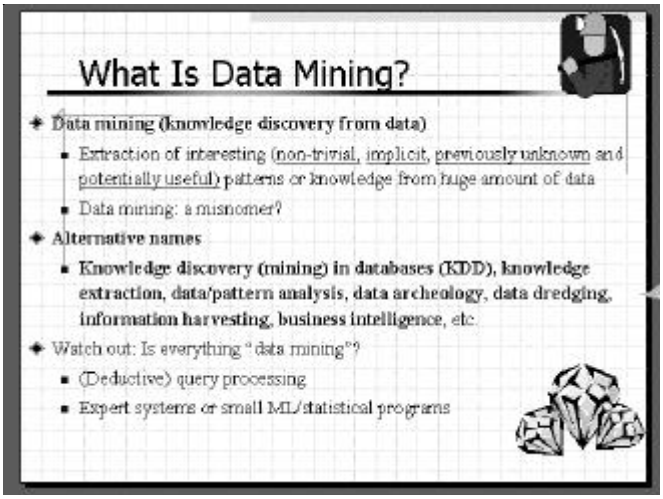
References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004
3. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

Related Websites

- www-db.stanford.edu/~ullman/mining/mining.html
- www.cise.ufl.edu/class/cis6930fa03dm/notes.html
- www.ecestudents.ul.ie/Course_Pages/Btech_ITT/Modules/ET4727/lecturenotes.htm





What is Data Mining

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

Recommended Reference Books

- R. Agrawal, J. Han, and H. Mannila, Readings in Data Mining: A Database Perspective, Morgan Kaufmann (in preparation)
- U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996
- J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001
- D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2001
- T. M. Mitchell, Machine Learning, McGraw Hill, 1997
- G. Piatetsky-Shapiro and W. J. Frawley, Knowledge Discovery in Databases, AAAI/MIT Press, 1991
- S. M. Weiss and N. Indurkhya, Predictive Data Mining, Morgan Kaufmann, 1998

Notes

LESSON 17
DATA MINING CONCEPTS-2

Structure

- Objective
- Introduction
- What is Data Mining?
- Data Mining: Definitions
- KDD vs Data mining,
- Stages of KDD
  - Selection
  - Preprocessing
  - Transformation
  - Data mining
  - Data visualization
  - Interpretation and Evaluation
- Data Mining and Data Warehousing
- Machine Learning vs Data mining
- DBMS vs Data mining
- Data Warehousing
- Statistical Analysis
- Difference between Database management systems (DBMS), Online Analytical Processing (OLAP) and Data Mining

Objective

When you have completed this lesson you should be able to:

- Understand the basic concepts of data mining.
- Study the concept of Knowledge Discovery of Data (KDD)
- Study the relation between KDD and Data mining
- Identify various stages of KDD
- Understand the difference between Data mining and Data warehousing
- Learn various concepts like Inductive learning, Statistics and machine learning
- Understand the difference between data mining and DBMS
- Understand the difference between Database management systems (DBMS), Online Analytical Processing (OLAP) and Data Mining

Introduction

Lets revise the topic done in the previous lesson.

Can you define the term "data mining"?

Here is the answer: "Simply put, data mining is used to discover patterns and relationships in your data in order to help you make better business decisions."
Herb Edelstein, Two Crows

"The non trivial extraction of implicit, previously unknown, and potentially useful information from data"
William J Frawley, Gregory Piatetsky-Shapiro and Christopher J Matheus

Some important points about Data mining:

- Data mining finds valuable information hidden in large volumes of data.
- Data mining is the analysis of data and the use of software techniques for finding patterns and regularities in sets of data.
- The computer is responsible for finding the patterns by identifying the underlying rules and features in the data.
- It is possible to "strike gold" in unexpected places as the data mining software extracts patterns not previously discernible or so obvious that no one has noticed them before.
- Mining analogy:
  - Large volumes of data are sifted in an attempt to find something worthwhile.
  - In a mining operation large amounts of low-grade materials are sifted through in order to find something of value.

What is Data Mining?

Data mining in the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. With the widespread use of databases and the explosive growth in their sizes, organizations are faced with the problem of information overload. The problem of effectively utilizing these massive volumes of data is becoming a major problem for all enterprises. Traditionally, we have been using data for querying a reliable database repository via some well-circumscribed application or canned report-generating utility. While this mode of interaction is satisfactory for a large class of applications, there exist many other applications, which demand exploratory data analyses. We have seen that in a data warehouse, the OLAP engine provides an adequate interface for querying summarized and aggregate information across different dimension-hierarchies. Though such methods are relevant for decision support systems, these lack the exploratory characteristics of querying. The OLAP engine for a data warehouse (and query languages for DBMS) supports query-triggered usage of data, in the sense that the analysis is based on a query posed by a human analyst. On the other hand, data mining techniques support automatic exploration of data. Data mining attempts to source out [patterns and trends in the data and infers rules from these patterns. With these rules the user will be able to support, review and examine decisions in some related business or scientific area. This opens up the possibility of a new way of interacting with databases and data warehouses. Consider, for example, a banking application where the manager wants to know whether there is a specific pattern followed by defaulters. It is hard to formulate a SQL query for

such information. It is generally accepted that if we know the query precisely we can turn to query language to formulate the query. But if we have some vague idea and we do not know the precisely query, then we can resort to data mining techniques.

The evolution of data mining began when business data was first stored in computers, and technologies were generated to allow users to navigate through the data in real time. Data mining takes this evolutionary process beyond retrospective data access and navigation, to prospective and proactive information delivery. This massive data collection, high performance computing and data mining algorithms.

We shall study some definitions of term data mining in the following section.

Data Mining: Definitions

Data mining, the extraction of the hidden predictive information from large databases is a powerful new technology with great potential to analyze important information in the data warehouse. Data mining scours databases for hidden patterns, finding predictive information that experts may miss, as it goes beyond their expectations. When implemented on a high performance client/server or parallel processing computers, data mining tolls can analyze massive databases to deliver answers to questions such as which clients are most likely to respond to the next promotions mailing. There is an increasing desire to use this new technology in the new application domain, and a growing perception that these large passive databases can be made into useful actionable information.

The term 'data mining' refers to the finding of relevant and useful information from databases. Data mining and knowledge discovery in the databases is a new interdisciplinary field, merging ideas from statistics, machine learning, databases and parallel computing. Researchers have defined the term ' data mining' in many ways.

We discuss a few of these definitions below.

1. Data mining or knowledge discovery in databases, as it is also known is the non-trivial extraction of implicit, previously unknown and potentially useful information from the data. This encompasses a number of technical approaches, such as clustering, data summarization, classification, finding dependency networks, analyzing changes, and detecting anomalies.

Though the terms data mining and KDD are used above synonymously, there are debates on the difference and similarity between data mining and knowledge discovery. In the present book, we shall be suing these two terms synonymously. However, we shall also study the aspects in which these two terms are said to be different.

Data retrieval, in its usual sense in database literature, attempts to retrieve data that is stored explicitly in the database and presents it to the user in a way that the user can understand. It does not attempt to extract implicit information. One may argue that if we store 'date-of-birth' as a field in the database and extract 'age' from it, the information received from the database is not explicitly available. But all of us would agree that the information is not 'non-trivial'. On the other hand, if one attempts to as a sort of non-trivial extraction of implicit information. Then, can we say that extracting the average age o f the employees of a department from the employees database (which stores the date-of-birth of every employee) is a data-mining task? The task is surely 'non-trivial' extraction of implicit information. It is needed a type of data mining task, but at a very low level. A higher-level task would, for example, be to find correlations between the average age and average income of individuals in an enterprise.

2. Data mining is the search for the relationships and global patterns that exist in large databases but are hidden among vast amounts of data, such as the relationship between patient data and their medical diagnosis. This relationship represents valuable knowledge about the databases, and the objects in the database, if the database is a faithful mirror of the real world registered by the database.

Consider the employee database and let us assume that we have some tools available with us to determine some relationships between fields, say relationship between age and lunch-patterns. Assume, for example, that we find that most of employees in there thirties like to eat pizzas, burgers or Chinese food during their lunch break. Employees in their forties prefer to carry a home-cooked lunch from their homes. And employees in their fifties take fruits and salads during lunch. If our tool finds this pattern from the database which records the lunch activities of all employees for last few months, then we can term out tool as a data-mining tool. The daily lunch activity of all employees collected over a reasonable period fo time makes the database very vast. Just by examining the database, it is impossible to notice any relationship between age and lunch patterns.

3. Data mining refers to using a variety of techniques to identify nuggets of information or decision-making knowledge in the database and extracting these in such a way that they can be put to use in areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but it has low value and no direct use can be made of it. It is the hidden information in the data that is useful.

Data mining is a process of finding value from volume. In any enterprise, the amount of transactional data generated during its day-to-day operations is massive in volume. Although these transactions record every instance of any activity, it is of little use in decision-making. Data mining attempts to extract smaller pieces of valuable information from this massive database.

4. Discovering relations that connect variables in a database is the subject of data mining. The data mining system self-learns from the previous history of the investigated system, formulating and testing hypothesis about rules which systems obey. When concise and valuable knowledge about the system of interest is discovered, it can and should be interpreted into some decision support system, which helps the manager to make wise and informed business decision.

Data mining is essentially a system that learns from the existing data. One can think of two disciplines, which address such problems- Statistics and Machine Learning. Statistics provide sufficient tools for data analysis and machine learning deals with different learning methodologies. While statistical methods are theory-rich-data-poor, data mining is data-rich-theory-poor

approach. On the other hand machine-learning deals with whole gamut of learning theory, which most often data mining is restricted to areas of learning with partially specified data.

5. Data mining is the process of discovering meaningful, new correlation patterns and trends by sifting through large amount of data stored in repositories, using pattern recognition techniques as well as statistical and mathematical techniques.

One important aspect of data mining is that it scans through a large volume of to discover patterns and correlations between attributes. Thus, though there are techniques like clustering, decision trees, etc., existing in different disciplines, are not readily applicable to data mining, as they are not designed to handle Ii amounts of data. Thus, in order to apply statistical and mathematical tools, we have to modify these techniques to be able efficiently sift through large amounts of data stored in the secondary memory.

KDD vs. Data Minning

Knowledge Discovery in Database (KDD) was formalized in 1989, with reference to the general concept of being broad and high level in the pursuit of seeking knowledge from data. The term data mining was then coined; this high-level application technique is used to present and analyze data for decision-makers.

Data mining is only one of the many steps involved in knowledge discovery in databases. The various Steps in the knowledge discovery process include data selection, data cleaning and preprocessing, data transformation and reduction, data mining algorithm selection and finally the post-processing and the interpretation of the discovered knowledge. The KDD process tends to be highly iterative and interactive. Data mining analysis tends to work up from the data and the best techniques are developed with an orientation towards large volumes of data, making use of as much data as Possible to arrive at reliable conclusions and decisions. The analysis process starts with a set of data, and uses a methodology to develop an optimal representation of the structure of data, during which knowledge is acquired. Once knowledge is acquired, this can be extended to large sets of data on the assumption that the large data set has a structure similar to the simple data set.

Fayyad et al. distinguish between KDD and data mining by giving the following definitions.

Knowledge Discovery in Databases is, "Process of identifying a valid, potentially useful and ultimately understandable structure in data. This process involves selecting or sampling data from a data warehouse, cleaning or preprocessing it, transforming or reducing it (if needed), applying a data-mining component to produce a structure, and then evaluating the derived structure.

Data Mining is a step in the KDD process concerned with the algorithmic means by which patterns or structures are enumerated from the data under acceptable computational efficiency limitations".

Thus, the structures that arc the outcome of the data mining process must meet certain conditions so that these can be considered as knowledge. These conditions are: validity, understandability, utility, novelty and interestingness.

Stages OF KDD

The stages of KDD, starting with the raw data and finishing with the extracted knowledge, are given below.

Selection

This stage is concerned with selecting or segmenting the data that are relevant to some criteria. For example, for credit card customer profiling, we extract the type of transactions for each type of customers and we may not be interested in details of the shop where the transaction takes place.

Preprocessing

Preprocessing is the data cleaning stage where unnecessary information is removed. For example, it is unnecessary to note the sex of a patient when studying pregnancy! When the data is drawn from several sources, it is possible that the same information represented in different sources in different formats. This stage reconfigures the data to ensure a consistent format, as there is a possibility of inconsistent formats.

Transformation

The data is not merely transferred across, but transformed in order to be suitable the task of data mining. In this stage, the data is made usable and navigable.

Data Mining

This stage is concerned with the extraction of patterns from the data.

Interpretation and Evaluation

The patterns obtained in the data mining stage are converted into knowledge, which turn, is used to support decision-making.

Data Visualization

Data visualization makes it possible for the analyst to gain a deeper, more intuitive understanding of the data and as such can work well alongside data mining. Data mining allows the analyst to focus on certain patterns and trends and explore them in depth using visualization. On its own, data visualization can be overwhelmed by the volume of data in a database but in conjunction with data mining can help with exploration.

Data visualization helps users to examine large volumes of data and detect patterns visually. Visual displays of data such as maps, charts and other graphics representations allow data to be presented compactly to the users. A single graphical screen can encode as much information as can a far larger, number of text screens. For example, if a user wants to find out whether the production problems at a plant are correlated to the location of the plants, the problem locations can be encoded in a special color, say red, on a map. The user can then discover locations in which the problems are occurring. He may then form a hypothesis about why problems are occurring in those locations, and may verify the hypothesis against the database.

Data Mining and Data Warehousing

The goal of a data warehouse is to support decision making with data. Data mining can be used in conjunction with a data warehouse to help with certain types of decisions. Data mining can be applied to operational databases with individual transactions. To make data mining more efficient, the data warehouse should have an aggregated' or summarized collec-

tion of data. Data mining helps in extracting meaningful new patterns that cannot be found necessarily by merely querying or processing data or metadata in the data warehouse. Data mining applications should therefore be strongly considered early, during the design of a data warehouse. Also, data mining tools should be designed to facilitate their use in conjunction with data ware-houses. In fact, for very large databases 'running into terabytes of data, successful use of database mining applications will depend, first on the construction of a data warehouse.

Machine Learning vs. Data Mining

- Large Data sets in Data Mining
- Efficiency of Algorithms is important
- Scalability of Algorithms is important
- Real World Data
- Lots of Missing Values
- Pre-existing data - not user generated
- Data not static - prone to updates
- Efficient methods for data retrieval available for use
- Domain Knowledge in the form of integrity constraints available.

Data Mining vs. DBMS

- Example DBMS Reports
  - Last months sales for each service type
  - Sales per service grouped by customer sex or age bracket
  - List of customers who lapsed their policy
- Questions answered using Data Mining
  - What characteristics do customers that lapse their policy have in common and how do they differ from customers who renew their policy?
  - Which motor insurance policy holders would be potential customers for my House Content Insurance policy?

Data Warehouse

- Data Warehouse: centralized data repository, which can be queried for business benefit.
- Data Warehousing makes it possible to
  - Extract archived operational data
  - Overcome inconsistencies between different legacy data formats
  - Integrate data throughout an enterprise, regardless of location, format, or communication requirements
  - Incorporate additional or expert information

Statistical Analysis

- Ill-suited for Nominal and Structured Data Types
- Completely data driven - incorporation of domain knowledge not possible
- Interpretation of results is difficult and daunting
- Requires expert user guidance

Difference between Database management systems (DBMS), Online Analytical Processing (OLAP) and Data Mining

Area	DBMS	OLAP	Data Mining
Task	Extraction of detailed and summary data	Summaries, trends and forecasts	Knowledge discovery of hidden patterns and insights
Type of result	Information	Analysis	Insight and Prediction
Method	Deduction (Ask the question, verify with data)	Multidimensional data modeling, Aggregation, Statistics	Induction (Build the model, apply it to new data, get the result)
Example question	Who purchased mutual funds in the last 3 years?	What is the average income of mutual fund buyers by region by year?	Who will buy a mutual fund in the next 6 months and why?

Discussion

- What is data mining?
- Discuss the role of data mining in data warehousing.
- How a data mining different from KDD?
- Explain the stages of KDD.
- Write an essay on "Data mining: Concepts, Issues and Trends".
- How can you link data mining with a DBMS?
- Explain the difference between a KDD and Data mining.
- Correctly contrast the difference between Database Management System, OLAP and Data mining?

References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004
3. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

Related Websites

- www-db.stanford.edu/~ullman/mining/mining.html
- www.cise.ufl.edu/class/cis6930fa03dm/notes.html
- www.ecestudents.ul.ie/Course_Pages/Btech_ITT/Modules/ET4727/lecturenotes.htm

What Is Data Mining?

- Data mining (knowledge discovery from data)
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data
  - Alternative names
  - Knowledge discovery (mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.
- Watch out: Is everything "data mining"?
  - (Deductive) query processing
  - Expert systems or small ML/statistical programs

Contd.

- The phases depicted start with the raw data and finish with the extracted knowledge which was acquired as a result of the following stages:
  - Selection – selecting or segmenting the data according to some criteria. E.g., all those people who own a car, in this way subsets of data can be determined.
  - Preprocessing – this is the data cleansing stage where certain information is removed which is deemed unnecessary and may slow down queries. E.g., unnecessary to note the sex of a patient when studying pregnancy.
  - Transformation – the data is not merely transferred across but transformed in that overlays may added such as the demographic overlays commonly used in market research.

Contd.

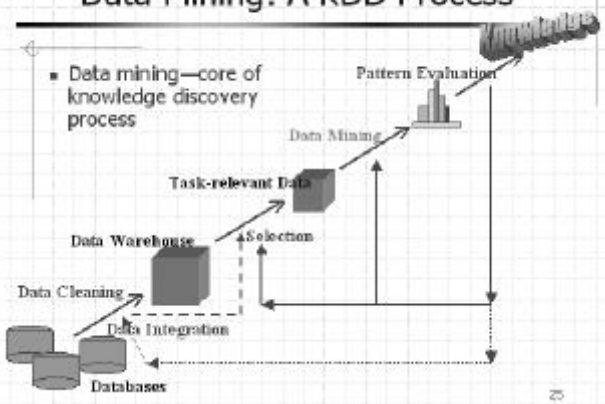
- Data Mining is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase revenue, cut costs or both.
- Data mining software is one of a number of analytical tools for analyzing data.
- It allows users to analyze data from many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

Contd.

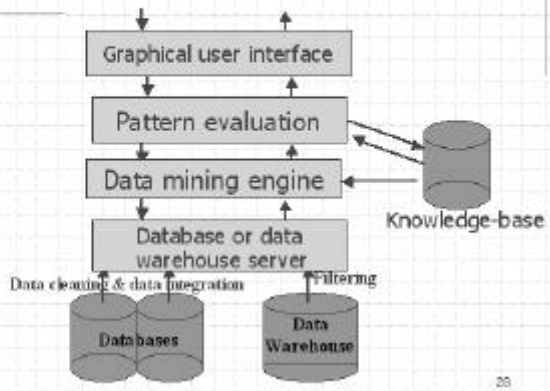
- Data mining – this stage is concerned with the extraction of patterns from the data. A pattern can be defined as given a set of facts F, a language L, and some measure of certainty C a pattern is a statement S in L that describes relationships among a subset Fs of F with a certainty c such that S is simpler in some sense than the enumeration of all the facts in Fs.
- Interpretation and evaluation – the patterns identified by the system are interpreted into knowledge which can then be used to support human decision-making e.g., prediction & classification tasks, summarizing the contents of a database or explaining observed phenomena.

Data Mining: A KDD Process

- Data mining—core of knowledge discovery process



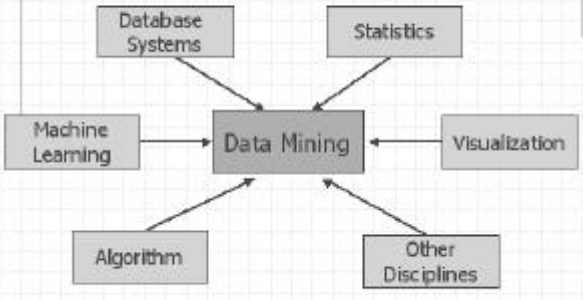
Architecture: Typical Data Mining System



Data Mining vs Machine Learning

- Knowledge discovery in databases (KDD) or data mining and the part of machine learning (ML) dealing with learning from examples overlap in the algorithms used and the problems addressed.
  - Differences:
  - KDD is concerned with finding understandable knowledge, while ML is concerned with improving performance of an agent.
  - KDD is concerned with v.large, real world databases, while ML typically looks at smaller data sets.
  - ML is a broader field which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc. KDD is a part of ML which is concerned with finding understandable knowledge in large set of real world examples.

Data Mining: Confluence of Multiple Disciplines



37

Contd.

- Practical KDD systems are expected to include three interconnected phases:
  - Translation of standard DB info. Into a form suitable for use by learning facilities;
  - Using ML techniques to produce knowledge bases from DB; &
  - Interpreting the knowledge produced to solve user's problems and /or reduce data spaces. Data spaces being the no. of examples.

Recommended Reference Books

- R. Agrawal, J. Han, and H. Mannila, Readings in Data Mining: A Database Perspective, Morgan Kaufmann (in preparation)
- U. M. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996
- U. Fayad, G. Grinstein, and A. Wierse, Information Visualization in Data Mining and Knowledge Discovery, Morgan Kaufmann, 2001
- J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001
- D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2001
- T. M. Mitchell, Machine Learning, McGraw-Hill, 1997
- S. M. Weiss and N. Indurkhya, Predictive Data Mining, Morgan Kaufmann, 1998

Notes

LESSON 18
ELEMENTS AND USES OF DATA MINING

Structure

- Objective
- Introduction
- Elements and uses of Data Mining
- Relationships & Patterns
- Data mining problems/issues
  - Limited Information
  - Noise and missing values
  - Uncertainty
  - Size, updates, and irrelevant fields
- Potential Applications
- Retail/Marketing
- Banking
  - Insurance and Health Care
  - Transportation
  - Medicine
- Data Mining and Data Warehousing

Data Mining as a Part of the Knowledge Discovery Process
Objective
At the end of this lesson you will be able to

- Understand various elements and uses of Data mining
- Study the importance and role of relationships and patterns in data mining
- Study various issues related to Data mining
- Identify potential applications of data mining
- Understand how data mining is a part of KDD

Introduction

After studying the previous lessons, you must have understood the meaning and significance of Data mining. In this lesson, I will explain you about various elements and uses of Data mining. You will study the importance and role of relationships and patterns in data mining. There are various issues related to Data mining, I will discuss all these issues in this lesson. You will also study various potential applications of data mining. Apart from the above topics, I will also tell you how data mining is a part of KDD.

Elements and uses of Data Mining

Data mining consists of five major elements:

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

Relationships & Patterns

- Discovering relationships is key to successful marketing.
- In operational or data warehouse system, the data architect and design personnel have meticulously defined entities and relationships.
- An entity is a set of information containing fact about a related set of data. The discovery process in a data mining exercise sheds light on relationships hidden deep down in many layers of corporate data.
- The benefits of pattern discovery to a business add real value to a data mining exercise. No one can accurately predict that person X is going to perform activity in close proximity with activity Z.
- Using data mining techniques & systematic analysis on warehouse data, however this prediction can be backed up by the detection of patterns behavior.
- Patterns are closely related to habit; in other words, the likelihood of an activity being performed in closely proximity to another activity is discovered in the midst of identifying a pattern.
- Some operational systems in the midst of satisfying daily business requirements create vast amount of data.
- The data is complex and the relationships between elements are not easily found by the naked eye.
- We need Data mining Softwares to do these tasks.
  - Insightful Miner
  - XAffinity

Data Mining Problems/Issues

Data mining systems rely on databases to supply the raw data for input and this raises problems in that databases tend be dynamic, incomplete, noisy, and large. Other problems arise as a result of the adequacy and relevance of the information stored.

Limited Information

A database is often designed for purposes different from data mining and sometimes the properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Inconclusive data causes problems because if some attributes essential to knowledge about the application domain are not present in the data it may be impossible to discover significant knowledge about a given domain. For example cannot diagnose malaria from a patient database if that database does not contain the patients' red blood cell count.

Noise and missing values

Databases are usually contaminated by errors so it cannot be assumed that the data they contain is entirely correct. Attributes, which rely on subjective or measurement judgments, can give rise to errors such that some examples may even be mis-classified. Error in either the values of attributes or class information are known as noise. Obviously where possible it is desirable to eliminate noise from the classification information as this affects the overall accuracy of the generated rules.

Missing data can be treated by discovery systems in a number of ways such as;

- simply disregard missing values
- omit the corresponding records
- infer missing values from known values
- treat missing data as a special value to be included additionally in the attribute domain
- or average over the missing values using Bayesian techniques.

Noisy data in the sense of being imprecise is characteristic of all data collection and typically fit a regular statistical distribution such as Gaussian while wrong values are data entry errors. Statistical methods can treat problems of noisy data, and separate different types of noise.

Uncertainty

Uncertainty refers to the severity of the error and the degree of noise in the data. Data precision is an important consideration in a discovery system.

Size, Updates, and Irrelevant Fields

Databases tend to be large and dynamic in that their contents are ever-changing as information is added, modified or removed. The problem with this from the data mining perspective is how to ensure that the rules are up-to-date and consistent with the most current information. Also the learning system has to be time-sensitive as some data values vary over time and the discovery system is affected by the 'timeliness' of the data.

Another issue is the relevance or irrelevance of the fields in the database to the current focus of discovery for example post codes are fundamental to any studies trying to establish a geographical connection to an item of interest such as the sales of a product.

Potential Applications

Data mining has many and varied fields of application some of which are listed below.

Retail/Marketing

- Identify buying patterns from customers
- Find associations among customer demographic characteristics
- Predict response to mailing campaigns
- Market basket analysis

Banking

- Detect patterns of fraudulent credit card use
- Identify 'loyal' customers
- Predict customers likely to change their credit card affiliation
- Determine credit card spending by customer groups
- Find hidden correlations between different financial indicators
- Identify stock trading rules from historical market data

Insurance and Health Care

- Claims analysis - i.e which medical procedures are claimed together
- Predict which customers will buy new policies
- Identify behaviour patterns of risky customers
- Identify fraudulent behaviour

Transportation

- Determine the distribution schedules among outlets
- Analyse loading patterns

Medicine

- Characterise patient behaviour to predict office visits
- Identify successful medical therapies for different illnesses

Data Mining and Data Warehousing

The goal of a data warehouse is to support decision making with data. Data mining can be used in conjunction with a data warehouse to help with certain types of decisions. Data mining can be applied to operational databases with individual transactions. To make data mining more efficient, the data warehouse should have an aggregated' or summarized collection of data. Data mining helps in extracting meaningful new patterns that cannot be found necessarily by merely querying or processing data or metadata in the data warehouse. Data mining applications should therefore be strongly considered early, during the design of a data warehouse. Also, data mining tools should be designed to facilitate their use in conjunction with data ware-houses. In fact, for very large databases 'running into terabytes of data, successful use of database mining applications will depend, first on the construction of a data warehouse.

Data Mining as a Part of the Knowledge Discovery Process

Knowledge Discovery in Databases, frequently abbreviated as KDD, typically encompasses more than data mining. The knowledge discovery process comprises six phases:[6] data selection, data cleansing, enrichment, data transformation or encoding, data mining, and the reporting and display of the discovered information.

As an example, consider a transaction database maintained by a specialty consumer goods retailer. Suppose the client data includes a customer name, zip code, phone num-ber, date of purchase, item code, price, quantity, and total amount. A variety of new knowledge can be discovered by KDD processing on this client database. During data selec-tion, data about specific items or categories of items, or from stores in a specific region or area of the country, may be selected. The data cleansing process then may correct invalid zip codes or eliminate records with incorrect phone prefixes. Enrichment typically enhances the data with additional sources of information. For example,

given the client names and phone numbers, the store may purchase other data about age, income, and credit rating and append them to each record. Data transformation and encoding may be done to reduce the amount of data. For instance, item codes may be grouped in terms of product categories into audio, video, supplies, electronic gadgets, camera, accessories, and ;0 on. Zip codes may be aggregated into geographic regions; incomes may be divided into Len ranges, and so on.

Goals of Data Mining and Knowledge Discovery

Broadly speaking, the goals of data mining fall into the following classes: Prediction, identification, classification, and opti-mization.

- **Prediction** - Data mining can show how certain attributes within the data will behave in the future. Examples of predictive data mining include the analysis of buy-ing transactions to predict what consumers will buy under certain discounts, how much sales volume a store would generate in a given period, and whether deleting a product line would yield more profits. In such applications, business logic is used cou-pled with data mining. In a scientific context, certain seismic wave patterns may pre-dict an earthquake with high probability.
- **Identification** - Data patterns can be used to identify the existence of an item, an event, or an activity. For example, intruders trying to break a system may be identified by the programs executed, files accessed, and CPU time per session. In biological applica-tions, existence of a gene may be identified by certain sequences of nucleotide symbols in the DNA sequence. The area known as authentication is a form of identification .It ascertains whether a user is indeed a' specific' user or one from an authorized class; it involves a comparison of parameters or images or signals against a database.
- **Classification -** Data mining can partition the data so that different classes or categories can be identified based on combinations of parameters. For example, customer in a supermarket can be categorized into discount-seeking shoppers, shoppers in a rush, loyal regular shoppers, and infrequent shoppers. This classification may be in different analyses of customer buying transactions as a post-mining activity times classification based on common domain knowledge is used as an input to decompose the mining problem and make it simpler. For instance, health foods, party foods, or school lunch foods are distinct categories in the supermarket business. It -makes sense to analyze relationships within and across categories as separate prob-lems. Such, categorization may be used to encode the data appropriately before subjecting it to further data mining.
- **Optimization** - One eventual goal of data mining may be to optimize the use off limited resources such as time, space, money, or materials and to maximize output variables such as sales or profits under a given set of constraints. As such, this goal of mining resembles the objective function used in operations research problems deals with optimization under constraints.

The term data mining is currently used in a very broad sense. In some situation includes statistical analysis and constrained optimization as well as machine. There is no sharp line separating data mining from these disciplines. It is beyond our scope, therefore, to discuss in detail the entire range of applica-tions that make up this body of work.

Types of Knowledge Discovered during Data Mining

The term "knowledge" is broadly interpreted as involving some degree of intelligence. Knowledge is often class inductive and deductive. Data mining addresses inductive knowledge. Knowledge can be represented in many forms in an unstruc-tured sense; it can be represented by rules, or prepositional logic. In a structured form, it may be represented in decision trees, semantic networks, neural works, or hierarchies of classes or frames. The knowledge discovered during data mining can be described in five ways, as follows.

1. **Association. Rules-**These rules correlate the presence of a set of items another range of values for another set of variables. Examples: (1) When a _ retail shopper buys a handbag, she is likely to buy shoes. (2) An X-ray image maintaining characteristics a and b is likely to also exhibit characteristic c.
2. **Classification hierarchies-**The goal is to work from an existing set of even a transaction to create a hierarchy of classes. Examples: (1) A population may be -divided into five ranges of credit worth -iness based on a history of previous co transactions. (2) A model may be developed for the factors that determine desirability of location of a. store-on a 1-10 scale. (3) Mutual funds may be classified based on performance data using characteristics such as growth, income, and stability.
3. **Sequential patterns-**A sequence of actions or events is sought. Example: If a patient underwent cardiac bypass surgery for blocked arteries and an aneurysm and later developed high blood urea within a year of surgery, he or she is likely to suffer from kidney failure within the next 18 months. Detection of sequential pat-terns is equivalent to detecting association among events with certain temporal relationships.
4. **Patterns within time series-**Similarities can be detected within positions of the time series. Three examples follow with the stock market price data as a time series: (1) Stocks of a utility company ABC Power and a financial company XYZ Securities show the same pattern during 1998 in terms of closing stock price. (2) Two products show the same selling pattern in summer but a different one in win-ter. (3) A pattern in solar magnetic wind may be used to predict changes in earth atmospheric conditions
5. **Categorization and segmentation-:** A given population of events or items can be partitioned (segmented) into sets of "similar" elements. Examples: (1) An entire population of treatment data on a disease may be divided into groups based on the similarity of side effects produced. (2) The adult popu1ation in the United States may be categorized into five groups from "most likely to buy" to "least likely to buy" a new product. (3) The web accesses made by a collection of users against a set of documents (say, in a

digital library) may be analyzed in. terms of the key-words of documents to reveal clusters or categories of Users.

For most applications, the desired knowledge is a combination of the above types. We expand on each of the above knowledge types in the following subsections.

Discussions

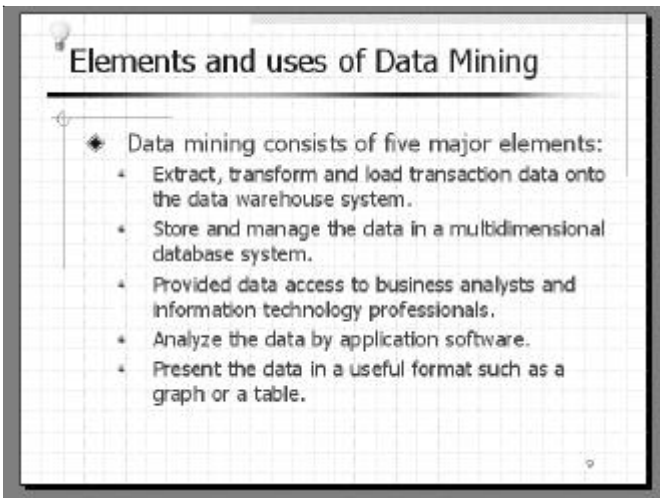
- Write short notes on:
  - **Association Rules**
- Discuss various elements and uses of data mining.
- Explain the significance of relationship and pattern in data mining.
- Give some examples of Operational systems.

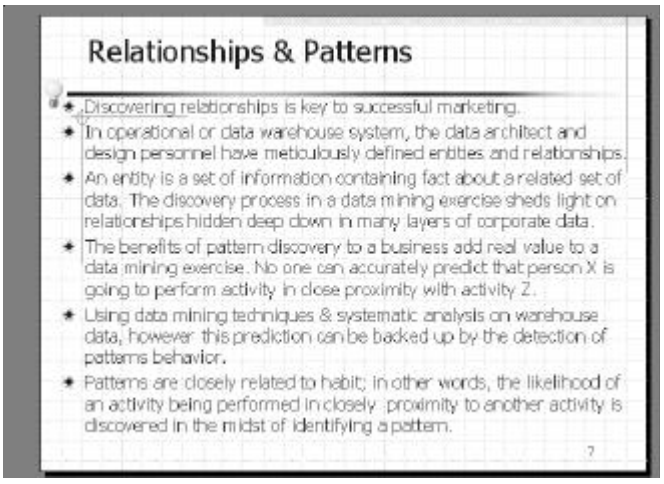
References

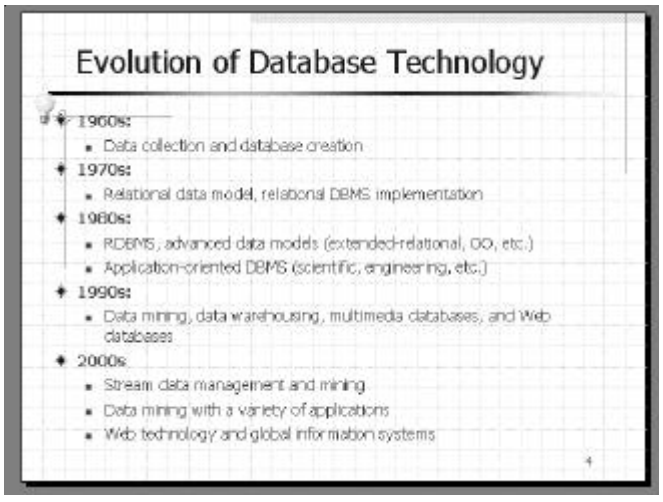
1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Anahory, Sam,** *Data warehousing in the real world: a practical guide for building decision support systems,* Delhi: Pearson Education Asia, 1997.
3. **Berry, Michael J.A. ; Linoff, Gordon,** *Mastering data mining : the art and science of customer relationship management,* New York : John Wiley & Sons, 2000
4. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.
5. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004

Related Websites

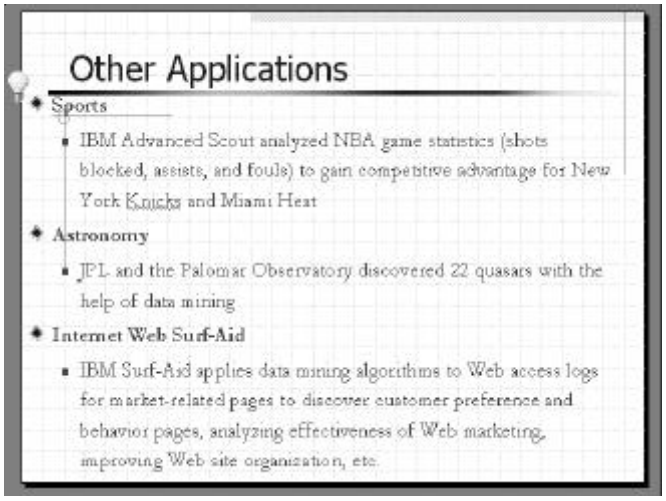
- www-db.stanford.edu/~ullman/mining/mining.html
- www.cise.ufl.edu/class/cis6930fa03dm/notes.html
- www.ecestudents.ul.ie/Course_Pages/Btech_ITT/Modules/ET4727/lecturenotes.htm








Why Data Mining?—Potential Applications
Data analysis and decision support
Market analysis and management
Target marketing, customer relationship management (CRM), market basket analysis, cross selling, market segmentation
Risk analysis and management
Forecasting, customer retention, improved underwriting, quality control, competitive analysis
Fraud detection and detection of unusual patterns (outliers)
Other Applications
Text mining (news group, email, documents) and Web mining
Stream data mining
DNA and bio-data analysis
39


Other Applications
Sports
IBM Advanced Scout analyzed NBA game statistics (shots blocked, assists, and fouls) to gain competitive advantage for New York Knicks and Miami Heat
Astronomy
JPL and the Palomar Observatory discovered 22 quasars with the help of data mining
Internet Web Surf-Aid
IBM Surf-Aid applies data mining algorithms to Web access logs for market-related pages to discover customer preference and behavior pages, analyzing effectiveness of Web marketing, improving Web site organization, etc.

Notes

LESSON 19
DATA INFORMATION AND KNOWLEDGE

Structure

- Objective
- Introduction
- Data, Information and Knowledge
- Information
- Knowledge
- Data warehouse
- What can Data Mining Do?
- How Does Data Mining Work?
- Data Mining in a Nutshell
- Differences between Data Mining and Machine Learning

Objective

At the end of this lesson you will be able to

- Understand the meaning and difference of Data, Information and knowledge
- Study the need of data mining
- Understand the working of data mining
- Study the difference between Data Mining and Machine Learning

Introduction

In the previous lesson you have studied various elements and uses of data mining. In this lesson, I will explain you the difference and significance of Data, information and knowledge. Further, you will also study about the need and working of data mining.

Data, Information and Knowledge

Data are any facts, numbers, or text that can be processed by a computer. Today, organizations are accumulating vast and growing amounts of data in different formats and different databases. This includes

- Operational or transactional data such as sales, cost, inventory, payroll, accounting.
- Non-operational data such as industry sales, forecast data, and macro economic data.
- Meta data-data about the data itself, such as logical database design or data dictionary definitions

Information

The patterns, associations, or relationships among all this data can provide information. For example, analysis of retail point of sale transaction data can yield information on which product are selling and when.

Meta data-data about the data itself, such as logical database design or data dictionary definitions

Knowledge

Information can be converted into knowledge about historical patterns and future trends. For example summary information on retail supermarket sales can be analyzed in light of promotional efforts to provide knowledge of consumer buying behavior. Thus, a manufacturer or retailer could determine which items are most susceptible to promotional efforts.

Data Warehouse

Dramatic advances in data capture, processing power, data transmission and storage capabilities are enabling organization to integration their various databases into data warehouses. Data warehousing is defined as a process of centralized data management and retrieval. Data warehousing, like data mining, is a relatively new term although the concept itself has been around for years. Data warehousing represents an ideal vision of maintaining a central repository of all organizational data. Centralization of data is needed to maximize user access and analysis. Dramatic technological advances are making this vision a reality for many companies. And, equally dramatic advances in data analysis software is what support data mining.

What can Data Mining do?

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

With data mining, a retailer could use point of sale records of customer purchase to send targeted promotions based on an individual's purchase history. By mining demographic data from comment or warranty cards, the retailer could develop products and promotions to appeal to specific customer segments. For example, Blockbuster Entertainment mines its video rental history database to recommend rentals to individual customers. American Express can suggest products to its cardholders based on analysis of their monthly expenditures.

Wal-Mart is pioneering massive data mining to transform its supplier relationships. Captures point of sale transactions from over 2,900 stores in 6 countries and continuously transmits this data to its massive 7.5 terabyte tera data warehouse. Wal-Mart allows more than 3,500 suppliers, to access data on their products and perform data analyses. These suppliers use this data to identify customer-buying patterns at the store display level. They use this information to mange local store inventory and identify new merchandising opportunities. In 1995, Wal-mart computers processed over 1 million complex queries.

The national Basketball Association (NBA) is exploring a data mining application that can be used in conjunction with image recordings of basketball games. The Advanced Scout software analyzes the movement of players to help coaches orchestrate players and strategies. For example and analysis of the play by play sheet of the game played between the NEW YORK Knick and the Cleveland Cavaliers on January 6, 1995 reveals that when mark price played the guard position, john Williams attempted four jump shots and made each one! Advanced scout not only finds this pattern, but also explains that it is interesting because it differs considerably form the average shooting percentage of 49.30%, for the cavaliers during that game.

By using the NBA universal clock, a coach can automatically bring up the video clips showing each of the jump shots attempted by Williams with price on the floor, without needing to comb through hours of video footage. Those clips show very successful picks and roll pay in which price draws the knick's defense, and then finds Williams for and open jumps shot.

How does Data Mining Work?

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

Classes: Stored data is used to locate data in predetermined groups. For example, a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order. This information could be used to increase traffic by having daili specials

Clusters: Data items are grounded according to logical relationship or consumer preferences; For example, data can be mined to identify market segments or consumer affinities.

Association: Data can be mined to identify associations. The beer diaper example is an example of associative mining.

Sequential patterns: Data is mined to anticipate behavior patterns and trends. For example, an outdoor equipment retailer could predict the likelihood of a backpack being purchased based on a consumer's purchase of sleeping bags and hiking shoes.

Data Mining in a Nutshell

The past two decades has seen dramatic increases in the amount of information or data being stored in electronic format. This accumulation of data has taken palace at an explosive rate. It has been estimated that the amount of information in the world doubles every 20 months and number of databases are increasing even faster. The increase in use of electronic data gathering devices such as point-of-sale or remote sensing devices has contributed to this explosion of available data. Figure 1 form the red brick company illustrates the data explosion.

Data storage becomes easier as the availabilities of large amount computing power at low cost i.e., the cost of processing power and storage is falling, made data cheap. There was also a introduction of few machine learning method for statistical analysis of data. The new method tend to be computationally intensive hence a demand for more processing power.

Having concentrated so much attention of the accumulation of data the problem was what to do with this valuable resource? It was recognize that information is at the heart of business operations and that decision-markets could make use of the data stored to gain valuable insight into the business. Database Management systems gave access to the data stored but this was only a small part of what would be gained from the data. Traditional on-line transaction processing system, OLTP, Sare goods at putting data into database quickly, safely and efficiently but are not good at delivering meaningful analysis in return. Analyzing data can provide further knowledge about a business by going beyond the data explicitly stored to derive knowledge about the business. This is where data mining or knowledge discovery in data base (KDD) has obvious for any enterprise.

The term data mining has been stretched beyond its limits to apply to any form of data analysis. Some of the numerous definitions of data mining, or knowledge discovery in database are:

Data mining or knowledge discovery in database (KDD) as it is also known is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. This encompasses a number of different technical approaches, such as clustering, data summarization learning classification rules finding dependency net works analyzing changes and detecting anomalies.

Willam J frawley, Gregory piatetsky – Shapiro and christopher j matheus Data ming is the search of relationships and global patterns that exit in large database but are hidden among the vast amount of data such as a relationship between patient data and their medical diagnosis. These relationships represent valuable knowledge about the database and the object in the database and if the database is a faithful mirror; of the real world registered by the database.

Data mining analysis tends to work from the data up and the best techniques are those developed with an orientation towards large volumes of data, making use of as much of the collected data as possible to arrive at reliable conclusions and decisions. The analysis process starts with a set of data, uses a methodology to develop an optimal representation of the structure of the data during which time knowledge id acquired. Once knowledge has been acquired this can be extended to larger data set has a structure similar to the sample data. Again this is analogous to a mining operation where large amounts of low-grade materials are sifted through in order to find something of value.

Differences between Data Mining and Machine Learning

Knowledge Discovery in Databases (KDD) or Data Mining, and the part of Machine Learning (ML) dealing with learning from examples overlap in the algorithms used and the problems addressed.

The main differences are:

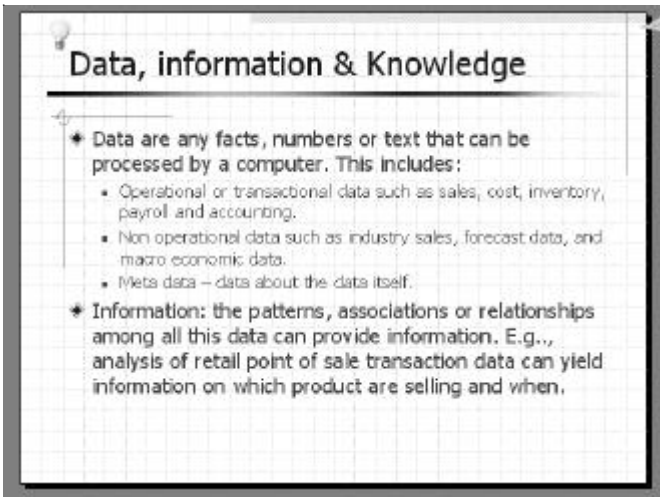
- KDD is concerned with finding understandable knowledge, while ML is concerned with improving performance of an agent. So training a neural network to balance a pole is part of ML, but not of KDD. However, there are efforts to extract knowledge from neural networks, which are very relevant for KDD.
- KDD is concerned with very large, real-world databases, while ML typically (but not always) looks at smaller data sets. So efficiency questions are much more important for KDD.
- ML is a broader field, which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc.

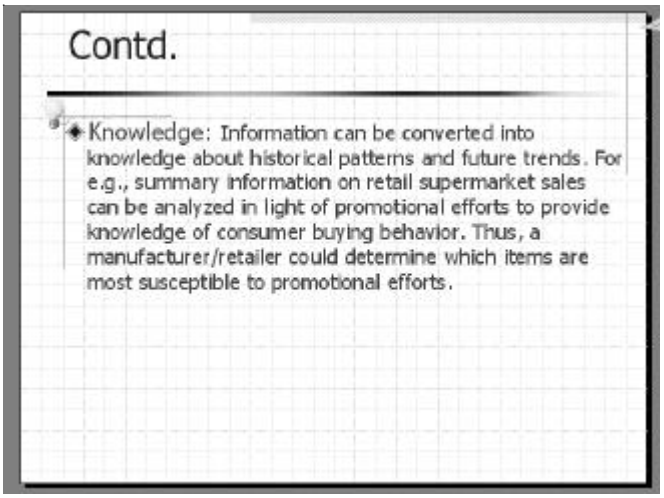
KDD is that part of ML which is concerned with finding understandable knowledge in large sets of real-world examples. When integrating machine-learning techniques into database systems to implement KDD some of the databases require:

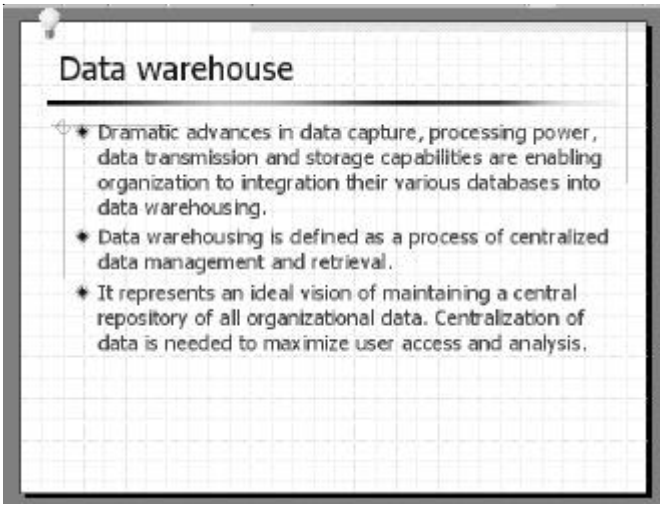
- More efficient learning algorithms because realistic databases are normally very large and noisy. It is usual that the database is often designed for purposes different from data mining and so properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Databases are usually contaminated by errors so the data-mining algorithm has to cope with noise whereas ML has laboratory type examples i.e. as near perfect as possible.
- More expressive representations for both data, e.g. tuples in relational databases, which represent instances of a problem domain, and knowledge, e.g. rules in a rule-based system, which can be used to solve users' problems in the domain, and the semantic information contained in the relational schemata.

Practical KDD systems are expected to include three interconnected phases

- Translation of standard database information into a form suitable for use by learning facilities;
- Using machine learning techniques to produce knowledge bases from databases; and
- Interpreting the knowledge produced to solve users' problems and/or reduce data spaces. Data spaces being the number of examples.







References

1. **Adriaans, Pieter,** *Data mining,* Delhi: Pearson Education Asia, 1996.
2. **Berson, Smith,** *Data warehousing, Data Mining, and OLAP,* New Delhi: Tata McGraw- Hill Publishing, 2004
3. **Elmasri, Ramez,** *Fundamentals of database systems,* 3rd ed. Delhi: Pearson Education Asia, 2000.

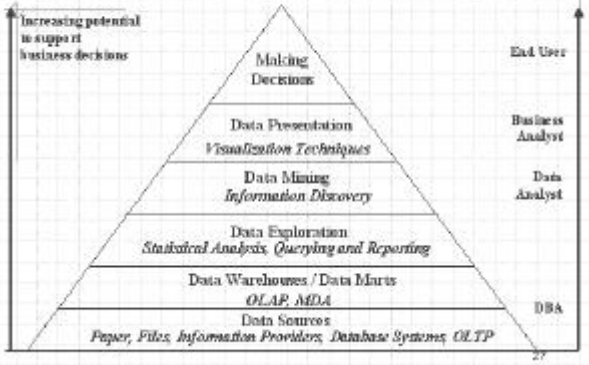
Related Websites

- www-db.stanford.edu/~ullman/mining/mining.html
- www.cise.ufl.edu/class/cis6930fa03dm/notes.html
- www.ecestudents.ul.ie/Course_Pages/Btech_ITT/Modules/ET4727/lecturenotes.htm
- http://web.utk.edu/~peilingw/is584/ln.pdf

What can Data Mining Do?

- Data Mining is primarily used today by companies with a strong consumer focus retail, financial communication, and marketing organizations.
- It enables these companies to determine relationships among "internal" factors such as price, product positioning or staff skill & external factors like economic indicators, competition and customer demographics.
- With data mining, a retailer could use point of sale records of customer purchase to send targeted promotions based on an individual's purchase history.

Data Mining and Business Intelligence



How does data mining work?

- Data Mining software analyzes relationships & patterns in stored transaction data based on open ended user queries. E.g., Statistical, machine learning and neural networks.Generally, any of four types of relationships are sought:
  - Classes : Stored data is used to locate data in predetermined groups. For e.g..a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order.This info. Could be used to increase traffic by having daily specials.

Market Analysis and Management

- Where does the data come from?
  - Credit card transactions, loyalty cards, discount coupons, customer complaint calls, plus (public) lifestyle studies
- Target marketing
  - Find clusters of "model" customers who share the same characteristics: interest, income level, spending habits, etc.
  - Determine customer purchasing patterns over time
- Cross-market analysis
  - Associations/co-relations between product sales, & prediction based on such association
- Customer profiling
  - What types of customers buy what products (clustering or classification)
- Customer requirement analysis
  - identifying the best products for different customers
  - predict what factors will attract new customers
- Provision of summary information
  - multidimensional summary reports
  - statistical summary information (data central tendency and variation)

Contd.

- Clusters : data items are grounded according to logical relationship or consumer preferences, for e.g., data can be mined to identify market segments or consumer affinities.
- Association – Data can be mined to identify associations.
- Sequential patterns – data is mind to anticipate behavior patterns and trends. E.g., an outdoor equipment retailer could predict the likelihood of a backpack being purchased based on a consumer purchase of sleeping bags and hiking shoes.

Recommended Reference Books

- R. Agrawal, J. Han, and H. Mannila, Readings in Data Mining: A Database Perspective, Morgan Kaufmann (in preparation)
- U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996
- J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2001
- D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2001
- T. M. Mitchell, Machine Learning, McGraw Hill, 1997
- G. Piatetsky-Shapiro and W. J. Frawley. Knowledge Discovery in Databases. AAAI/MIT Press, 1991
- S. M. Weiss and N. Indurkhya, Predictive Data Mining, Morgan Kaufmann, 1998

LESSON 20
DATA MINING MODELS

Structure

- Objective
- Introduction
- Data mining
- Data Mining Models
- Verification Model
- Discovery Model
- Data warehousing

Objective

At the end of this lesson you will be able to

- Reviewing the concept of Data mining
- Study various types of data mining models
- Understand the difference between Verification and Discovery model.

Introduction

In the previous lesson, I have explained you the difference and significance of Data, Information and Knowledge. You have also studied about the need and working of data mining. In this lesson, I will explain you various types of Data Mining Models.

Data Mining

Data mining, the extraction of the hidden predictive information from large databases is a powerful new technology with great potential to analyze important information in the data warehouse. Data mining scours databases for hidden patterns, finding predictive information that experts may miss, as it goes beyond their expectations. When implemented on a high performance client/server or parallel processing computers, data mining tolls can analyze massive databases to deliver answers to questions such as which clients are most likely to respond to the next promotions mailing. There is an increasing desire to use this new technology in the new application domain, and a growing perception that these large passive databases can be made into useful actionable information.

The term 'data mining' refers to the finding of relevant and useful information from databases. Data mining and knowledge discovery in the databases is a new interdisciplinary field, merging ideas from statistics, machine learning, databases and parallel computing. Researchers have defined the term ' data mining' in many ways.

1. Data mining or knowledge discovery in databases, as it is also known is the non-trivial extraction of implicit, previously unknown and potentially useful information from the data. This encompasses a number of technical approaches, such as clustering, data summarization, classification, finding dependency networks, analyzing changes, and detecting anomalies.

Though the terms data mining and KDD are used above synonymously, there are debates on the difference and similarity between data mining and knowledge discovery. In the present book, we shall be suing these two terms synonymously. However, we shall also study the aspects in which these two terms are said to be different.

Data retrieval, in its usual sense in database literature, attempts to retrieve data that is stored explicitly in the database and presents it to the user in a way that the user can understand. It does not attempt to extract implicit information. One may argue that if we store 'date-of-birth' as a field in the database and extract 'age' from it, the information received from the database is not explicitly available. But all of us would agree that the information is not 'non-trivial'. On the other hand, if one attempts to as a sort of non-trivial extraction of implicit information. Then, can we say that extracting the average age o f the employees of a department from the employee's database (which stores the date-of-birth of every employee) is a data-mining task? The task is surely 'non-trivial' extraction of implicit information. It is needed a type of data mining task, but at a very low level. A higher-level task would, for example, be to find correlations between the average age and average income of individuals in an enterprise.

2. Data mining is the search for the relationships and global patterns that exist in large databases but are hidden among vast amounts of data, such as the relationship between patient data and their medical diagnosis. This relationship represents valuable knowledge about the databases, and the objects in the database, if the database is a faithful mirror of the real world registered by the database.

Consider the employee database and let us assume that we have some tools available with us to determine some relationships between fields, say relationship between age and lunch-patterns. Assume, for example, that we find that most of employees in there thirties like to eat pizzas, burgers or Chinese food during their lunch break. Employees in there forties prefer to carry a home-cooked lunch from their homes. And employees in there fifties take fruits and salads during lunch. If our tool finds this pattern from the database which records the lunch activities of all employees for last few months, then we can term out tool as a data-mining tool. The daily lunch activity of all employees collected over a reasonable period fo time makes the database very vast. Just by examining the database, it is impossible to notice any relationship between age and lunch patterns.

3. Data mining refers to using a variety of techniques to identify nuggets of information or decision-making knowledge in the database and extracting these in such a way that they can be put to use in areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but it has low value and no direct use can be

made of it. It is the hidden information in the data that is useful.

Data mining is a process of finding value from volume. In any enterprise, the amount of transactional data generated during its day-to-day operations is massive in volume. Although these transactions record every instance of any activity, it is of little use in decision-making. Data mining attempts to extract smaller pieces of valuable information from this massive database.

4. Discovering relations that connect variables in a database is the subject of data mining. The data mining system self-learns from the previous history of the investigated system, formulating and testing hypothesis about rules which systems obey. When concise and valuable knowledge about the system of interest is discovered, it can and should be interpreted into some decision support system, which helps the manager to make wise and informed business decision.

Data mining is essentially a system that learns from the existing data. One can think of two disciplines, which address such problems- Statistics and Machine Learning. Statistics provide sufficient tools for data analysis and machine learning deals with different learning methodologies. While statistical methods are theory-rich-data-poor, data mining is data-rich-theory-poor approach. On the other hand machine-learning deals with whole gamut of learning theory, which most often data mining is restricted to areas of learning with partially specified data.

5. Data mining is the process of discovering meaningful, new correlation patterns and trends by sifting through large amount of data stored in repositories, using pattern recognition techniques as well as statistical and mathematical techniques.

One important aspect of data mining is that it scans through a large volume of to discover patterns and correlations between attributes. Thus, though there are techniques like clustering, decision trees, etc., existing in different disciplines, are not readily applicable to data mining, as they are not designed to handle Ii amounts of data. Thus, in order to apply statistical and mathematical tools, we have to modify these techniques to be able efficiently sift through large amounts of data stored in the secondary memory.

Data Mining Models

IBM have identified two types of model or modes of operation which may be used to unearth information of interest to the user.

Verification Model

The Verification model takes a hypothesis from the user and tests the validity of it against the data. The emphasis is with the user who is responsible for formulating the hypothesis and issuing the query on the data to affirm or negate the hypothesis.

In a marketing division for example with a limited budget for a mailing campaign to launch a new product it is important to identify the section of the population most likely to buy the new product. The user formulates an hypothesis to identify potential customers and the characteristics they share. Historical data about customer purchase and demographic information can then be queried to reveal comparable purchases and the characteristics shared by those purchasers, which in turn can be used to target a mailing campaign. The whole operation can be refined by 'drilling down' so that the hypothesis reduces the 'set' returned each time until the required limit is reached.

The problem with this model is the fact that no new information is created in the retrieval process but rather the queries will always return records to verify or negate the hypothesis. The search process here is iterative in that the output is reviewed, a new set of questions or hypothesis formulated to refine the search and the whole process repeated. The user is discovering the facts about the data using a variety of techniques such as queries, multidimensional analysis and visualization to guide the exploration of the data being inspected.

Discovery Model

The discovery model differs in its emphasis in that it is the system automatically discovering important information hidden in the data. The data is sifted in search of frequently occurring patterns, trends and generalizations about the data without intervention or guidance from the user. The discovery or data mining tools aim to reveal a large number of facts about the data in as short a time as possible.

An example of such a model is a bank database, which is mined to discover the many groups of customers to target for a mailing campaign. The data is searched with no hypothesis in mind other than for the system to group the customers according to the common characteristics found.

The typical discovery driven tasks are

- Discovery of association rules
- Discovery of classification rules
- Clustering

These tasks are of an exploratory nature and cannot be directly handed over to currently available database technology. We shall concentrate on these tasks now.

Discovery of Association Rules

An association rule is an expression of the form XÞY, where X and Y are the sets of items. The intuitive meaning of such a rule is that the transaction of the database which contains X tends to contain Y. Given a database, the goal is to discover all the rules that have the support and confidence greater than or equal to the minimum support and confidence, respectively.

Let $L = \{l_1, l_2, \ldots, l_m\}$ be a set of literals called items. Let D, the database, be a set of transactions, where each transactions, where each transaction T is a set of items. T is a set of items. T supports an item x, if x is in T. T is said to support a subset of items X, if T supports each item x in X. XÞY holds with confidence c, if c% of the transactions in D that support X also support. Y. the rule X Þ Y has support is the transaction set D if s% of the transactions in D support XOY. Support means how often X and Y occur together as a percentage of the total transactions. Confidence measures how much a particular item is dependent on another.

Thus, the association with a very high support and confidence is a pattern that occurs often in the database that should be obvious to the end user. Patterns with extremely low support and confidence should be regarded as of no significance. Only patterns with a combination of intermediate values of confi-

dence and support provide the user with interesting and previously unknown information. We shall study the techniques to discover association rules in the following chapters.

Discovery of Classification Rules

Classification involves finding rules that partition the data into disjoint groups. The input for the classification is the training data set, whose class labels are already known. Classification analyzes the training data set and constructs a model based on the class label, and aims to assign a class label to the future unlabelled records. Since the class field is known, this type of classification is known as supervised learning. A set of classifies future data and develops a better understanding of each class in the database. We can term this as supervised learning too.

There are several classification discovery models. They are: the decision trees, neural networks, genetic algorithms and the statistical models like linear/ geometric discriminates. The applications include the credit card analysis, banking, medical applications and the like. Consider the following example.

The domestic flights in our country were at one time only operated by Indian Airlines. Recently, many other private airlines began their operations for domestic travel. Some of the customers of Airlines started flying with these private airlines and, as a result, Indian Airlines lost these customers. Let us assume that Indian Airlines want to understand why some customers remain loyal while others leave. Ultimately, the airline wants to predict which customers it is most likely to lose to its competitors. Their aim to build a model based on the historical data of loyal customers versus customers who have left. This becomes a classifications problem. It is supervised learning task, as the historical data becomes the training set, which is used to trading the model. The decision tree is the most popular classification technique. We shall discuss different methods of decision tree construction in the forthcoming lessons.

Clustering

Clustering is a method of grouping data into different groups, so that the data in each group share similar trends and patterns. Clustering constitutes a major class of data mining algorithms. The algorithm attempts to automatically partition the data space into a set of regions or clusters, to which the examples in the table are assigned, either deterministically or probability-wise. The goal of process is to identify all sets of similar examples in the data, in some optimal fashion.

Clustering according to similarity is a concept which appears in many disciplines. If a measure of similarity is available, then there are a number of techniques for forming clusters. Another approach is to build set functions that measure some particular property of groups. This latter approach achieves what is known as optimal partitioning.

The objectives of clustering are:

- To uncover natural groupings
- To initiate hypothesis about the data
- To find consistent and valid organization of the data.

A retailer may want to know where similarities in his customer base, so that he can create and understand different groups. He can use the existing database of the different customers or, more specifically, different transactions collected over a period of time. The clustering methods will help him in identifying different categories of customers. During the discovery process, the differences between data sets can be discovered in order to separate them into different groups, and similarity between data sets can be used to group similar data together. We shall discuss in detail about using the clustering algorithm for data mining tasks in further lessons.

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 benefit 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,

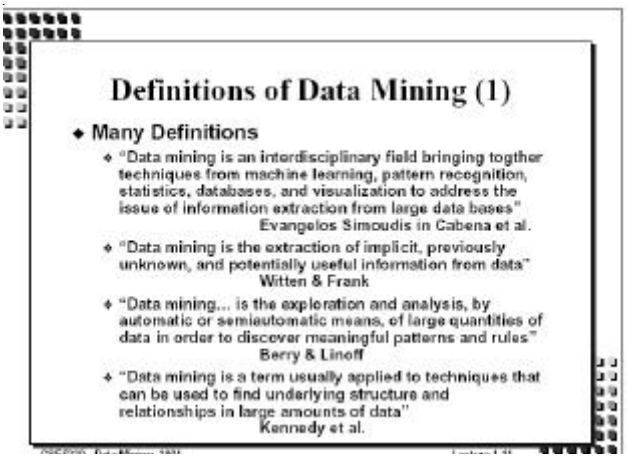
The logical link between what the managers see in their decision support EIS applications and the company's operational activities

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.

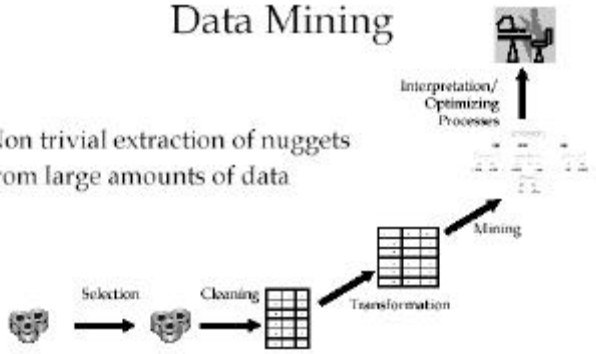
Discussion

- Explain the relation between a data warehouse and data mining
- What are the various kinds of Data mining models?
- "Data warehousing is a new powerful technique making it possible to extract archived operational data and overcome inconsistencies between different legacy data formats". Comment.
- Explain the problems associated with the Verification Model.
- "Data mining is essentially a system that learns from the existing data". Illustrate with examples.



Data Mining

Non trivial extraction of nuggets from large amounts of data



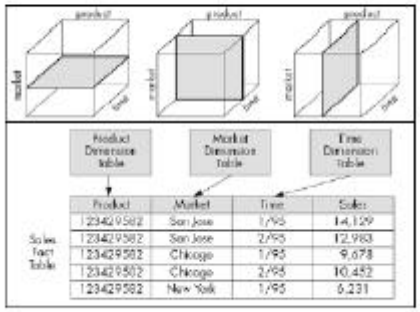
Data Mining is …

- Finding groups of people with similar hobbies

- Are chances of getting cancer higher if you live near a power line?

Data Mining is not …

- Generating multidimensional cubes of a relational table



Source: Multidimensional OLAP vs. Relational OLAP by Colin White

Data Mining Tasks

- Prediction methods
  - Use some variables to predict unknown or future values of the same or other variables
- Description methods
  - Find human interpretable patterns that describe data

From Fayyad, et al., Advances in Knowledge Discovery and Data Mining, 1996

Data Mining is not …

- Searching for a phone number in a phone book

- Searching for keywords on Google

Data Mining Models

- IBM have identified 2-types model or modes of operation which maybe used to unearth info. Of interest to the user.
- Verification Model : takes an hypothesis from the user & tests the validity of it against the data. The emphasis is with the user who is responsible for formulating the hypothesis & issuing the query on the data to affirm or negate the hypothesis.

  The problem with this model is the fact that no new info. Is created in the retrieval process but rather the queries will always return records to verify /negate the hypothesis. The search process here is iterative in that the o/p is reviewed, a new set of question /hypothesis formulated to refine the search & the whole process repeated.

Contd.

- Discovery Model : differs in its emphasis in that it is the system automatically discovering important facts hidden in the data. The data is shifted in search of frequently occurring patterns, trends & generalizations about the data w/o intervention or guidance from the user.
- The discovery or data mining tools aim to reveal a large number of facts about the data in as short a time as possible.

Data Mining and the Data Warehouse

- Organizations realized that they had large amounts of data stored (especially of transactions) but it was not easily accessible
- The data warehouse provides a convenient data source for data mining. Some data cleaning has usually occurred. It exists independently of the operational systems
  - Data is retrieved rather than updated
  - Indexed for efficient retrieval
  - Data will often cover 5 to 10 years
- A data warehouse is not a pre-requisite for data mining

Definitions of Data Mining (1)

- Many Definitions
  - "Data mining is an interdisciplinary field bringing togther techniques from machine learning, pattern recognition, statistics, databases, and visualization to address the issue of information extraction from large data bases" Evangelos Simoudis in Cabena et al.
  - "Data mining is the extraction of implicit, previously unknown, and potentially useful information from data" Witten & Frank
  - "Data mining... is the exploration and analysis, by automatic or semiautomatic means, of large quantities of data in order to discover meaningful patterns and rules" Berry & Linoff
  - "Data mining is a term usually applied to techniques that can be used to find underlying structure and relationships in large amounts of data" Kennedy et al.

Recommended Reference Books

- R. Agrawal, J. Han, and H. Mannila, Readings in Data Mining: A Database Perspective, Morgan Kaufmann (in preparation)
- U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996
- J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001
- D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2001
- T. M. Mitchell, Machine Learning, McGraw Hill, 1997
- G. Piatetsky-Shapiro and W. J. Frawley, Knowledge Discovery in Databases, AAAI/MIT Press, 1991
- S. M. Weiss and N. Indurkhya, Predictive Data Mining, Morgan Kaufmann, 1998

Definitions of Data Mining (2)

- Use of analytical tools to *discover knowledge* in a collection of data
  - The knowledge takes the form of patterns, relationships and facts which would *not otherwise be immediately apparent*
- These analytical tools may be drawn from a number of disciplines, which include:
  - machine learning
  - pattern recognition
  - statistics
  - artificial intelligence
  - human-computer interaction
  - information visualization
  - and many more...

LESSON 21
ISSUES AND CHALLENGES IN DM, DM APPLICATIONS AREAS

Structure

- Objective
- Introduction.
- Data mining problems/issues
- Limited Information
- Noise and missing values
- Uncertainty
- Size, updates, and irrelevant fields
- Other mining problems
  - Sequence Mining
  - Web Mining
- Text Mining
- Spatial Data Mining
- Data mining Application Areas

Objective

At the end of this lesson you will be able to

- Understand various issues related to data mining
- Learn difference between Sequence mining, Web mining, Text and Spatial data mining.
- Study in detail about different application areas of Data mining.

Introduction

In the previous lesson you have studied various types of Data Mining Models. In this lesson you will learn about various issues and problems related to Data mining. Here, you will also study about Sequence mining, Web mining, Text mining and spatial data mining.

Data Mining Problems/Issues

Data mining systems rely on databases to supply the raw data for input and this raises problems in that databases tend be dynamic, incomplete, noisy, and large. Other problems arise as a result of the adequacy and relevance of the information stored.

1. Limited Information

A database is often designed for purposes different from data mining and sometimes the properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Inconclusive data causes problems because if some attributes essential to knowledge about the application domain are not present in the data it may be impossible to discover significant knowledge about a given domain. For example cannot diagnose malaria from a patient database if that database does not contain the patient's red blood cell count.

2. Noise and missing values

Databases are usually contaminated by errors so it cannot be assumed that the data they contain is entirely correct. Attributes which rely on subjective or measurement judgements can give rise to errors such that some examples may even be mis-classified. Error in either the values of attributes or class information are known as noise. Obviously where possible it is desirable to eliminate noise from the classification information as this affects the overall accuracy of the generated rules.

Missing data can be treated by discovery systems in a number of ways such as;

- Simply disregard missing values
- Omit the corresponding records
- Infer missing values from known values
- Treat missing data as a special value to be included additionally in the attribute domain
- Or average over the missing values using Bayesian techniques.

Noisy data in the sense of being imprecise is characteristic of all data collection and typically fit a regular statistical distribution such as Gaussian while wrong values are data entry errors. Statistical methods can treat problems of noisy data, and separate different types of noise.

3. Uncertainty

Uncertainty refers to the severity of the error and the degree of noise in the data. Data precision is an important consideration in a discovery system.

4. Size, updates, and irrelevant fields

Databases tend to be large and dynamic in that their contents are ever-changing as information is added, modified or removed. The problem with this from the data mining perspective is how to ensure that the rules are up-to-date and consistent with the most current information. Also the learning system has to be time-sensitive as some data values vary over time and the discovery system is affected by the 'timeliness' of the data.

Another issue is the relevance or irrelevance of the fields in the database to the current focus of discovery for example post codes are fundamental to any studies trying to establish a geographical connection to an item of interest such as the sales of a product.

Other Mining Problems

We observed that a data mining system could either be a portion of a data warehousing system or a stand-alone system. Data for data mining need not always be enterprise -related data residing on a relational database. Data sources are very diverse and appear in varied form. It can be textual data, image data, CAD data, Map data, ECG data or the much talked about Genome data. Some data are structured and some are unstructured. Data mining remains an important tool, irrespective of

the forms or sources of data. We shall study the Data mining problems for different types of data.

Sequence Mining

Sequence mining is concerned with mining sequence data. It may be noted that in the, discovery of association rules, we are interested in finding associations between items irrespective of their order of occurrence. For example, we may be interested in the association between the purchase of a particular brand of soft drinks and the occurrence of stomach upsets. But it is more relevant to identify whether there is some pattern in the stomach upsets which occurs after the purchase of the soft drink. Then one is inclined to infer that the soft drink causes stomach upsets. On the other hand, if it is more likely that the purchase of the soft drink follows the occurrence of the stomach upset, then it is probable that the soft drink provides some sort of relief to the user. Thus, the discovery of temporal sequences of events concerns causal relationships among the events in a sequence. Another application of this domain concerns **drug misuse**. Drug misuse can occur unwittingly, when a patient is prescribed two or more interacting drugs within a given time period of each other. Drugs that interact undesirably are recorded along with the time frame as a pattern that can b located within the patient records. The rules that describe such instances of drug misuse are then successfully inducted based on medical records.

Another related area which falls into the larger domain of temporal data mining is trend discovery. One characteristic of sequence-pattern discovery in comparison with trend discovery is the lack of shapes, since the causal impact of a series of events, cannot be shaped.

Web Mining

With the huge amount of information available online, the World Wide Web is a ferti1e area for data mining research. Web mining research is at the crossroads of research from several research communities, such as database, information retrieval, and within AI, especially the sub areas of machine learning and natural language processing. Web mining is the use of data mining techniques to automatically discover and extract information from web documents and services. This area of research is so huge today partly due to the interests of various research commuI1ities, the tremendous growth of information sources available on the web and the recent interest in e-commerce. This phenomenon often creates confusion when we ask what constitutes web mining. Web mining can be broken down into following subtasks:

1. Resource finding: retrieving documents intended for the web.
2. Information selection and preprocessing: automatically selecting and preprocessing specific information from resources retrieved from the web.
3. Generalization: to automatically discover general patterns at individual web site as well as across multiple sites.
4. Analysis: validation and/or interpretation of the mined patterns.

Text Mining

The term text mining or KDT (Knowledge Discovery in Text) was first proposed by Feldman and Dagan in 1996. They suggest that text documents be structured by means of information extraction, text categorization, or applying NLP techniques as a preprocessing step before performing any kind of KDTs. Presently the term text mining, is being used to cover many applications such as text categorization, exploratory data analysis, text clustering, finding patterns in text databases, finding sequential patterns in texts, IE (Information Extraction), empirical computational linguistic tasks, and association discovery.

Spatial Data Mining

Spatial data mining is the branch of data mining that deals with spatial (location) data. The immense explosion in geographically-referenced data occasioned by developments in IT, digital mapping, remote sensing, and the global diffusion of GIS, places demands on developing data driven inductive approaches to spatial analysis and modeling. Spatial data mining is regarded as a special type of data mining that seeks to perform similar generic functions as conventional data mining tools, but modified to take into account the special features of spatial information.

For example, we may wish to discover some association among patterns of residential colonies and topographical features. A typical spatial association may look like: 'The residential land pockets are dense in a plain region and rocky areas are thinly populated"; or, "The economically affluent citizens reside in hilly, secluded areas whereas the middle income group residents prefer having their houses near the market".

Data mining Application Areas

The discipline of data mining is driven in part by new applications, which require new capabilities that are not currently being supplied by today's technology. These new applications can be naturally divided into three broad categories [Grossman, 1999].

A. Business And E-Commerce Data

This is a major source category of data for data mining applications. Back-office front-office, and network applications produce large amounts of data about business processes. Using this data for effective decision-making remains a fundamental challenge.

Business Transactions

Modern business processes are consolidating with millions of customers and billions of their transactions. Business enterprises require necessary information for their effective functioning in today's competitive world. For example, they would like know: "Is this transaction fraudulent?"; "Which customer is likely to migrate?", and "What product is this customer most likely to buy next?'.

Electronic Commerce

Not only does electronic commerce produce large data sets in which the analysis marketing patterns and risk patterns is critical but, it is also important to do this near-real time, in order to meet the demands of online transactions.

B. Scientific, Engineering And Health Care Data

Scientific data and metadata tend to be more complex in structure than business data. In addition, scientists and engineers are making increasing use of simulation and systems with application domain knowledge.

Genomic Data

Genomic sequencing and mapping efforts have produced a number of databases, which are accessible on the web. In addition, there are also a wide variety of other online databases. Finding relationships between these data sources is another fundamental challenge for data mining. .

Sensor Data

Remote sensing data is another source of voluminous data. Remote sensing satellites and a variety of other sensors produce large amounts of geo-referenced data. A fundamental challenge is to understand the relationships, including causal relationships, amongst this data.

Simulation Data

Simulation is now accepted as an important mode of science, supplementing theory and experiment. Today, not only do experiments produce huge data sets, but so do simulations. Data mining and, more generally, data intensive computing is proving to be a critical link between theory; simulation, and experiment.

Health Care Data

Hospitals, health care organizations, insurance companies, and the concerned government agencies accumulate large collections of data about patients and health care-related details. Understanding relationships in this data is critical for a Wide variety of problems ranging from determining what procedures and clinical protocols are most effective, to how best deliver health care to the maximum number of people.

Web Data

The data on the web is growing not only in volume but also in complexity. Web data now includes not only text, audio and video material, but also streaming data and numerical data.

Multimedia Documents

Today's technology for retrieving multimedia items on the web is far from satisfactory. On the other hand, an increasingly large number of matters are on the web and the number of users is also growing explosively. It is becoming harder to extract meaningful information from the archives of multimedia data as the volume grows.

Data Web

Today, the web is primarily oriented toward documents and their multimedia extensions. HTML has proved itself to be a simple, yet powerful, language for supporting this. Tomorrow, the potential exists for the web to prove equally important for working with data. The Extensible Markup Language (XML) is an emerging language for working with data in networked environments. As this infrastructure grows, data mining is expected to be a critical enabling technology for the emerging data web.

Data Mining Applications-Case Studies

There is a wide range of well-established business applications for data mining. These include customer attrition, profiling, promotion forecasting, product cross-selling, fraud detection, targeted marketing, propensity analysis, credit scoring, risk analysis, etc. We shall now discuss a few mock case studies and areas of DM applications.

Housing Loan Prepayment Prediction

A home-finance loan actually has an average life-span of only 7 to 10 years, due to prepayment. Prepayment means that the loan is paid off early, rather than at the end of say, 25 years. People prepay loans when they refinance or when they sell their home. The financial return that a home-finance institution derives from a loan depends on its life-span. Therefore, it is necessary for the financial institutions to be able to predict the life-spans of their loans. Rule discovery techniques can be used to accurately predict the aggregate number of loan prepayments in a given quarter (or, in a year), as a function of prevailing interest rates, borrower characteristics, and account data. This information can be used to fine-tune loan parameters such as interest rates, points, and fees, in order to maximize profits.

Mortgage Loan Delinquency Prediction

Loan defaults usually entail expenses and losses for the banks and other lending institutions. Data mining techniques can be used to predict whether or not a loan would go delinquent within the succeeding 12 months, based on historical data, on account information, borrower demographics, and economic indicators. The rules can be used to estimate and finetune loan loss reserves and to gain some business insight into the characteristics and circumstances of delinquent loans. This will also help in deciding the funds that should be kept aside to handle bad loans.

Crime Detection

Crime detection is another area one might immediately associate with data mining. Le us consider a specific case: to find patterns in 'bogus official' burglaries.

A typical example of this kind of crime is when someone turns up at the door pretending to be from the water board, electricity board, telephone department or gas company. Whilst they distract the householder, their partners will search the premises and steal cash and items of value. Victims of this sort of crime tend to be the elderly. These cases have no obvious leads, and data mining techniques may help in providing some unexpected connections to known perpetrators.

In order to apply data mining techniques, let us assume that each case is filed electronically, and contains descriptive information about the thieves. It also contains a description of their *modus operandi*. We can use any of the clustering techniques to examine a situation where a group of similar physical descriptions coincide with a group of similar *modus operandi*. If there is a good match here, and the perpetrators are known for one or more of the offences, then each of the unsolved cases could have well been committed by the same people.

By matching unsolved cases with known perpetrators, it would be possible to clear up old cases and determine patterns of behavior. Alternatively, if the criminal is unknown but a large

cluster of cases seem to point to the same offenders, then these frequent offenders can be subjected to careful examination.

Store-Level Fruits Purchasing Prediction

A super market chain called 'Fruit World' sells fruits of different types and it purchases these fruits from the wholesale suppliers on a day-to-day basis. The problem is to analyze fruit-buying patterns, using large volumes of data captured at the 'basket' level. Because fruits have a short shelf life, it is important that accurate store-level purchasing predictions should be made to ensure optimum freshness and availability. The situation is inherently complicated by the 'domino' effect. For example, when one variety of mangoes is sold out, then sales are transferred to another variety. With help of data mining techniques, a thorough understanding of purchasing trends enables a better availability of fruits and greater customer satisfaction.

Other Application Area

Risk Analysis

Given a set of current customers and an assessment of their risk-worthiness, develop descriptions for various classes. Use these descriptions to classify a new customer into one of the risk categories.

Targeted Marketing

Given a database of potential customers and how they have responded to a solicitation, develop a model of customers most likely to respond positively, and use the model for more focused new customer solicitation. Other applications are to identify buying patterns from customers; to find associations among customer demographic characteristics, and to predict the response to mailing campaigns.

Retail/Marketing

- Identify buying patterns from customers
- Find associations among customer demographic characteristics
- Predict response to mailing campaigns
- Market basket analysis

Customer Retention

Given a database of past customers and their behavior prior to attrition, deve1op a model of customers most likely to leave. Use the model for determining the course of action for these customers.

Portfolio Management

Given a particular financial 'asset, predict the return on investment to determine the inclusion of the asset in a folio or not.

Brand Loyalty

Given a customer and the product he/she uses, predict whether the customer will switch brands.

Banking

The application areas in banking are:

- Detecting patterns of fraudulent credit card use
- Identifying 'loyal' customers
- Predicting customers likely to change their credit card affiliation
- Determine credit card spending by' customer groups
- Finding hidden correlations between different financial indicators
- Identifying stock trading rules from historical market data

Insurance and Health Care

- Claims analysis - i.e., which medical procedures are claimed together
- Predict which customers will buy new policies
- Identify behavior patterns of risky customers
- Identify fraudulent behavior

Transportation

- Determine the distribution schedules among outlets
- Analyze loading patterns

Medicine

- Characterize patient behavior to predict office visits
- Identify successful medical therapies for different illnesses

Discussion

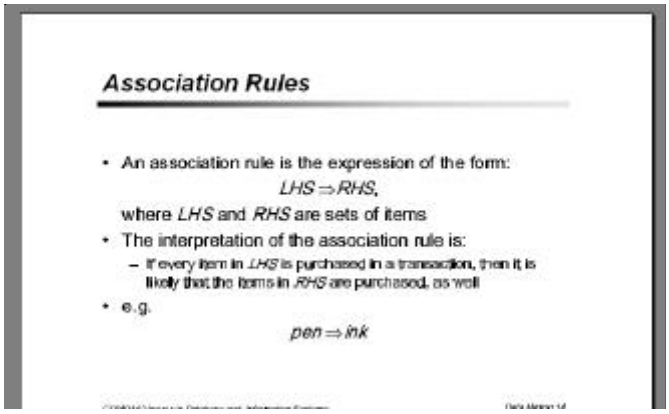
- Discuss different data mining tasks.
- What is spatial data mining?
- What is sequence mining?
- What is web mining?
- What is text mining?
- Discuss the applications of data mining in the banking industry.
- Discuss the applications of data mining in customer relationship management.
- How is data mining relevant to scientific data?
- How is data mining relevant for web-based computing?
- Discuss the application of data mining in science data.

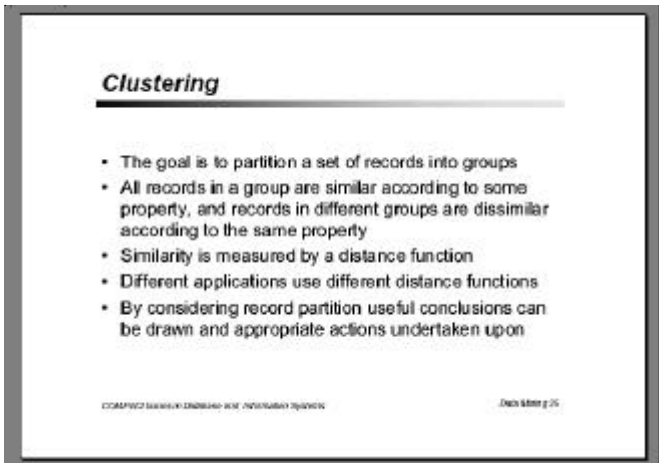
Bibliography

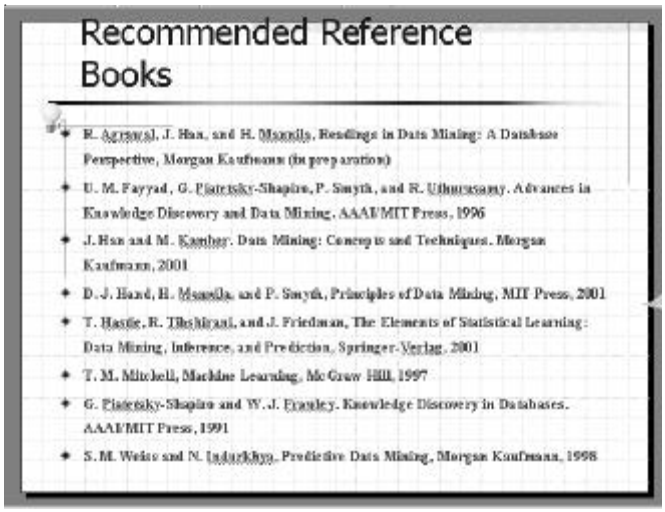
- **Agrawal R, Gupta A., and Sarawagi S.,** *Modeling multidimensional databases.* ICDE, 1997.
- **Anahory S., and Murray D.** *Data warehousing in the Real World: Apracfical guide for building decision support systems.* Addison Wesley Longman, 1997.
- **Barbara D. (ed.)** *Special Issue on Mining of Large Datasets; IEEE Data Engineering Bulletin,* 21 (1), 1998
- **Brachman R., Khabaza T., Kloesgen W., Shapiro G.P., and Simoudis E.,** Industrial applications of data mining and knowledge discovery, *Communication of ACM,-* 1926.
- **Fayyad U.M., Piatetsky-Shapiro G., Smyth P., Uthurusamy R. (Eds.):** *Advances in Knowledge Discovery and Data Mining.* Menlo Park, CA: AAAI Press/ The MIT Press, 1996
- **Fayyad U.M., Uthurusamy R. (eds.):** Special issue on data mining. *CommunicatJon of ACM,1996* .
- **Grossman R., Kasif S.,Moore R., Rocke D. and Ullmann J.** *Data Mining Research: Opportunities and Challenges, A Report.* www..ncdni.uic.edu/M3D-final-report.htm., Jan 1999.

- **Heckerman D.,** Bayesian networks for data, mining. *Data mining and knowledge Discovery,* 1997.
- **Imielinski T., Virmani A.,** Association Rules ... and What's Next? Towards Second Generation Data Mining Systems; In *Proceedings of the 2nd East=European ADBIS Conference.,* pp. 6-25,1998.
- **Mannila H.**, Methods and Problems in Data Mining; In *Proceedings of the 6'h International Ccmference on Database Theory* 1997 ,.Springer LNCS 1186.
- **Nestorov S., and Tsur S.** Integrating data miniqg with: relational DBMS: A tightly coupled approach, "www-db.stanford.edu/people/evitov.html; 1998.
- **Piatetsky-Shapiro G., and Frawley W.** (ed.): *Knowledge Discovery in Databases,* MIT Press, Cambridge, Ma, 1991
- **Sarawagi S.,** *et al.* Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications; In *Proceedings of the ACM SIGMOD International Conference on Management of Data* 1998,343-354.
- **Vinnani A.** *Second Generation Data Mining: Concepts and implementation ; Ph.D.* Thesis, Rutgers University, Apri11998.

Some slides on important Topics:






Recommended Reference Books
R. Agrawal, J. Han, and H. Mannila, Readings in Data Mining: A Database Perspective, Morgan Kaufmann (in preparation)
U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996
J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2001
D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2001
T. M. Mitchell, Machine Learning, McGraw Hill, 1997
G. Piatetsky-Shapiro and W. J. Frawley, Knowledge Discovery in Databases, AAAI/MIT Press, 1991
S. M. Weiss and N. Indurkhya, Predictive Data Mining, Morgan Kaufmann, 1998

Notes

LESSON 22
VARIOUS TECHNIQUES OF DATA MINING NEAREST NEIGHBOR AND CLUSTERING TECHNIQUES

Structure

- Objective
- Introduction.
- Types of Knowledge Discovered during Data Mining
  - Association Rules
  - Classification hierarchies
  - Sequential patterns
  - Patterns within time series
  - Categorization and segmentation
- Comparing the Technologies
- Clustering And Nearest-Neighbor Prediction Technique
- Where to Use Clustering and Nearest-Neighbor Prediction
- Clustering for clarity
- Nearest neighbor for prediction
- There is no best way to cluster
- How are tradeoffs made when determining which records fall into which clusters?
- What is the difference between clustering nearest-neighbor prediction?
- Cluster Analysis: Overview

Objective

At the end of this lesson you will be able to

- Understand various techniques used in Data mining
- Study about Nearest Neighbor and clustering techniques
- Understand about Cluster Analysis.

Introduction

In this lesson you will study about various techniques used in Data mining. You will study in detail about Nearest Neighbor and clustering techniques. I will also cover Cluster Analysis in this lesson.

Types of Knowledge Discovered during Data Mining

The term "knowledge" is broadly interpreted as involving some degree of intelligence. Knowledge is often class inductive and deductive. Data mining addresses inductive knowledge. Knowledge can be represented in many forms in an unstructured sense; it can be represented by rules, or prepositional logic. In a structured form, it may be represented in decision trees, semantic networks, neural works, or hierarchies of classes or frames.

The knowledge discovered during data mining can be described in five ways, as follows.

1. **Association Rules-**These rules correlate the presence of a set of items another range of values for another set of variables. Examples: (1) When a _ retail shopper buys a handbag, she is likely to buy shoes. (2) An X-ray image maintaining characteristics a and b is likely to also exhibit characteristic c.
2. **Classification hierarchies-**The goal is to work from an existing set of even a transaction to create a hierarchy of classes. Examples: (1) A population may be -divided into five ranges of credit worthiness based on a history of previous co transactions. (2) A model may be developed for the factors that determine desirability of location of a. store-on a 1-10 scale. (3) Mutual funds may be classified based on performance data using characteristics such as growth, income, and stability.
3. **Sequential patterns -** A sequence of actions or events is sought. Example: If a patient underwent cardiac bypass surgery for blocked arteries and an aneurysm and later developed high blood urea within a year of surgery, he or she is likely to suffer from kidney failure within the next 18 months. Detection of sequential pat-terns is equivalent to detecting association among events with certain temporal relationships.
4. **Patterns within time series -** Similarities can be detected within positions of the time series. Three examples follow with the stock market price data as a time series: (1) Stocks of a utility company ABC Power and a financial company XYZ Securities show the same pattern during 1998 in terms of closing stock price. (2) Two products show the same selling pattern in summer but a different one in win-ter. (3) A pattern in solar magnetic wind may be used to predict changes in earth atmospheric conditions
5. **Categorization and segmentation -** A given population of events or items can be partitioned (segmented) into sets of "similar" elements. Examples: (1) An entire population of treatment data on a disease may be divided into groups based on the similarity of side effects produced. (2) The adult popu1ation in the United States may be categorized into five groups from "most likely to buy" to "least likely to buy" a new product. (3) The web accesses made by a collection of users against a set of documents (say, in a digital library) may be analyzed in. terms of the key-words of documents to reveal clusters or categories of Users.

Comparing the Technologies

Most of the data mining technologies that are out there today are relatively new to the business community, and there are a lot of them (each technique usually is accompanied by a plethora of new companies and products). Given this state of affairs, one of the most important questions being asked right after "What is data mining?" is "Which technique(s) do I choose for my particular business problem?" The answer is, of course, not a simple one. There are inher-ent strengths and weaknesses of the different approaches, but most of the weaknesses can be overcome. How the technology is implemented into the data-

mining product can make all the difference in how easy the product is to use independent of how complex the underlying technology is.

The confusion over which data mining technology to use is further exacerbated by the data mining companies themselves who will often lead one to believe that their product is deploying a brand-new technology that is vastly superior to any other technology currently developed. Unfortunately this is rarely the case, and as we show in the chapters on modeling and comparing the technologies, it requires a great' deal of discipline and good experimental method to fairly compare different data mining methods. More often than not, this discipline is not used when evaluating many of the newest technologies. Thus the claims of improved accuracy that are often made are not always defensible.

To appear to be different from the rest, many of the products that arrive on the market are packaged in a way so as to mask the inner workings of the data mining algorithm Many data mining companies emphasize the newness and the black-box nature of their technology. There will, in fact, be data mining offerings that seek to combine every possible new technology into their product in the belief that more is better. In fact, more technology is usually just more confusing and makes it more difficult to make a fair comparison between offer-ings. When these techniques are understood and their similarities researched, one will find that many techniques that appeared to initially be different when they were not well understood are, in fact, quite similar. For that reason the data mining technologies that are introduced in this book are the basics from which the thousands of subtle variations are made. If you can understand these technologies and where they can be used, you will probably understand better than 99 percent of all the techniques and products that are currently available.

To help compare the different technologies and make the business user a lit-tle more savvy in how to choose a technology, we have introduced a high-level system of scorecards for each data mining technique described in this book. These scorecards can be used by the reader as a first-pass high-level look at what the strengths and weaknesses are for each of the different techniques. Along with the scorecard will be a more detailed description of how the scores were arrived! It, and if the score is low, what possible changes or workarounds could be made in the technique to improve the situation.

Clustering And Nearest-Neighbor Prediction Technique

Clustering and the nearest-neighbor prediction technique are among the oldest techniques used in data mining. Most people have an intuition that they understand what clustering is – namely that like records are grouped or clustered together and put into the same grouping. Nearest neighbor is a prediction technique that is quite similar to clustering; its essence is that in order to determine what a prediction value is in one record, the user should look for records with similar predictor values in the historical database and use the prediction value from the record that is "nearest" to the unknown record.

Where to Use Clustering and Nearest-Neighbor Prediction

Clustering and nearest-neighbor prediction are used in a wide variety of appli-cations, ranging from personal bankruptcy prediction to computer recognition-on of a person's handwriting. People who may not even realize that they are doing any kind of clustering also use these methods every day. For instance, we may group certain types of foods or automobiles together (e.g., high-fat foods, U.S. manufactured cars).

Clustering for Clarity

Clustering is a method in which like records are grouped together. Usually this is done to give the end user a high-level view of what is going on in the database.

Clustering is a data mining technique that is directed toward the goals of identification and classification. Clustering to identify a finite set of categories or clusters to which each data object (tuple) can be mapped. The categories may be disjoint or overlapping and may sometimes be organized into trees. For example, one might form categories of customers into the form d" tree and then map each customer to one or more of the categories. A closely related problem is that of estimating multivariate probability density functions of all variables that - could be attributes in a relation or from different relations.

Clustering for Outlier Analysis

Sometimes clustering is performed not so much to keep records together as to make it easier to see when one record sticks out from the rest. For instance

Most wine distributors selling inexpensive wine in Missouri and that ship a certain volume of product produce a certain level of profit. A cluster of stores can be formed with these characteristics. One store stands out, however, as producing significantly lower profit. On closer examination it turns out that the distributor was delivering product to but not collecting payment from one of its customers.

A sale on men's suits is being held in all branches of a department store for southern California. All stores except one with these characteristics have seen at least a 100 percent jump in revenue since the start of the sale.

TABLE 20.2 Some Commercially Available Cluster Tags

Name	Income	Age	Education	Vendor
Blue Blood Estates	High	35–54	College	Claritas PRIZM*
Shotguns and Pickup	Middle	35–64	High school	Claritas PRIZM*
Southside City	Low	Mix	Grade school	Claritas PRIZM*
Living off the Land	Middle-low	Families with school-age children	Low	Equifax MicroVision*
University USA	Very low	Young-mix	Medium–high	Equifax MicroVision*
Sunset Years	Medium	Seniors	Medium	Equifax MicroVision*

* All trademarks.

Nearest Neighbor for Prediction

One essential element underlying the concept of clustering is that one particu-lar object (whether cars, food, or customers) can be closer to another object than can some third object. It is interesting that most people have an innate sense of ordering placed on a variety of different objects.

Most people would agree that an apple is closer to an orange than it is to a tomato and that a Toyota Corolla is closer to a Honda Civic than to a Porsche. This sense of ordering on many different objects helps us place them in time and space and to

make sense of the world. It is what allows us to build clus-ters-both in databases on computers and in our daily lives. This definition of nearness that seems to be ubiquitous also allows us to make predictions. The nearest-neighbor prediction algorithm, simply stated, is

Objects that are "near" *to* each other will have similar prediction values as well. Thus, if you know the prediction value of one of the objects, you can predict it for its nearest neighbors.

One of the classic places where nearest neighbor has been used for prediction has been in text retrieval. The problem to be solved in text retrieval is one in which end users define a document (e.g., a Wall Street Journal article, a techni-cal conference paper) that is interesting to them and they solicit the system to "find more documents like this one," effectively defining a target of "this is the interesting document" or "this is not interesting." The prediction problem is that only a very few of the documents in the database actually have values for this prediction field (viz., only the documents that the reader has had a chance to look at so far). The nearest-neighbor technique is used to find other documents that share important characteristics with those documents that have been marked as interesting. AB with almost all prediction algorithms, nearest neigh-bor can be used for a wide variety of places. Its successful use depends mostly on the pre-formatting of the data, so that nearness can be calculated, and where individual records can be defined. In the text-retrieval example this was not too difficult-the objects were documents. This is not always as easy as it is for text retrieval. Consider what it might be like in a time series problem-say, for pre-dicting the stock market. In this case the input data is just a long series of stock prices over time without any particular record that could be considered to be an object. The value to be predicted is just the next value of the stock price.

This problem is solved for both nearest-neighbor techniques and for some other types of prediction algorithms by creating training records, taking, for instance, 10 consecutive stock prices and using the first 9 as predictor values and the 10th as the prediction value. Doing things this way, if you had 100 data points in your time series, you could create at least 10 different training records.

You could create even more training records than 10 by creating a new record starting at every data point. For instance, you could take the first 10 Data points in your time series and create a record. Then you could take the 10 consecutive data points starting at the second data point, then the 10 consecu-tive data points starting at the third data point. Even though some of the data points would overlap from one record to the next, the prediction value would always be different. In our example of 100 initial data points, 90 different training records could be created this way, as opposed to the 10 training records created via the other method.

There is no best way to cluster

This example, although simple, points up some important questions about clus-tering. For instance, is it possible to say whether the first clustering that was per-formed above (by financial status) was better or worse than the second clustering (by age and eye color)? Probably not, since the clusters were constructed for no particular purpose except to note similarities between some of the records and that the view of the database could be somewhat simplified by using clusters. But even the differences that were created by the two different clustering were driven by slightly different motivations (financial vs. romantic). In general, the reasons for clustering are just this ill defined because clusters are used more often than not for exploration and summarization and nut _s much as for prediction.

How are tradeoffs made when determining which records fall into which clusters?

Note that for the first clustering example, there was a pretty simple rule by which the records could be broken up into clusters-namely, by income.

TABLE : A Simple Clustering of the Example Database

ID.	Name	Prediction	Age	Balance ($)	Income	Eyes	Gender
3	Betty	No	47	16,543	High	Brown	F
5	Carla	Yes	21	2,300	High	Blue	F
6	Carl	No	27	5,400	High	Brown	M
8	Don	Yes	46	0	High	Blue	M
1	Amy	No	62	0	Medium	Brown	F
2	AI	No	53	1,800	Medium	Green	M
4	Bob	Yes	32	45	Medium	Green	M
7	Donna	Yes	50	165	Low	Blue	F
9	Edna	Yes	27 .	500	Low	Blue	F
10	Ed	No	68	1,200	Low	Blue	M

What is the difference between clustering nearest-neighbor prediction?

The main distinction between clustering and the nearest-neighbor technique is that clustering is what is called an *unsupervised learning* technique and near-est neighbor is generally used for prediction or a *supervised learning* technique. Unsupervised learning techniques are unsupervised in the sense that when they are run, there is no particular reason for the creation of the models the way there is for supervised learning techniques that are trying to perform predic-tion. In prediction, the patterns that are found in the database and presented in the model are always the most important patterns in the database for perform-ing some particular prediction. In clustering there is no particular sense of why certain records are near to each other or why they all fall into the same cluster. Some of the differences between clustering and nearest-neighbor prediction are summarized in Table 20.7.

How is the space for clustering and nearest neighbor defined?

For clustering, the n-dimensional space is usually defined by assigning one predictor to each dimension. For the nearest-neighbor algorithm, predictors are also mapped to dimensions, but then those dimensions are literally stretched or compressed according to how important the particular predictor is in making the prediction. The stretching of a dimension effectively makes

that dimension (and hence predictor) more important than the others in calculating the distance.

For instance, 'if you were a mountain climber and someone told you that you were 2 mi from your destination, the distance would be the same whether it were 1 mi north and' 1 mi up the face of the mountain or 2 mi north on level ground, but clearly the former route is much different from the latter. The dis-tance traveled straight upward is the most important in figuring out how long it will really take to get to the destination, and you would probably like to con-sider this "dimension" to be more important than the others. In fact, you, as a mountain climber, could "weight" the importance of the vertical dimension in calculating some new distance by reasoning that every mile upward is equiva-lent to 10 mi on level ground.

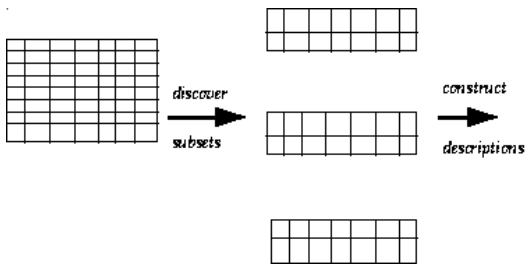
If you used this rule of thumb to weight the importance of one dimension over the other, it would be clear that in one case you were much "farther away" from your destination (limit) than in the second (2 mi). In the net section we'll show how the nearest neighbor algorithm uses the distance measure that similarly weights the important dimensions more heavily when calculating a distance.

Nearest Neighbor	Clustering
Used for prediction as well as consolidation.	Used mostly for consolidating data into a high-level view and general grouping of records into like behaviors.
Space is defined by the problem to be solved (Supervised Learning).	Space is defined as default n-dimensional space, or is defined by the user, or is a predefined space driven by past experience (Unsupervised learning).
Generally, only uses distance metrics to determine nearness.	Can use other metrics besides distance to determine nearness of two records – for example, linking points together.

Cluster Analysis: Overview

In an unsupervised learning environment the system has to discover its own classes and one way in which it does this is to cluster the data in the database as shown in the following diagram. The first step is to discover subsets of related objects and then find descriptions e.g., D1, D2, D3 etc. which describe each of these subsets.

Figure 5: Discovering clusters and descriptions in a database



Clustering and segmentation basically partition the database so that each partition or group is similar according to some criteria or metric. Clustering according to similarity is a concept, which appears in many disciplines. If a measure of similarity is available there are a number of techniques for forming clusters. Membership of groups can be based on the level of similarity between members and from this the rules of membership can be defined. Another approach is to build set functions that measure some property of partitions ie groups or subsets as functions of some parameter of the partition. This latter approach achieves what is known as optimal partitioning.

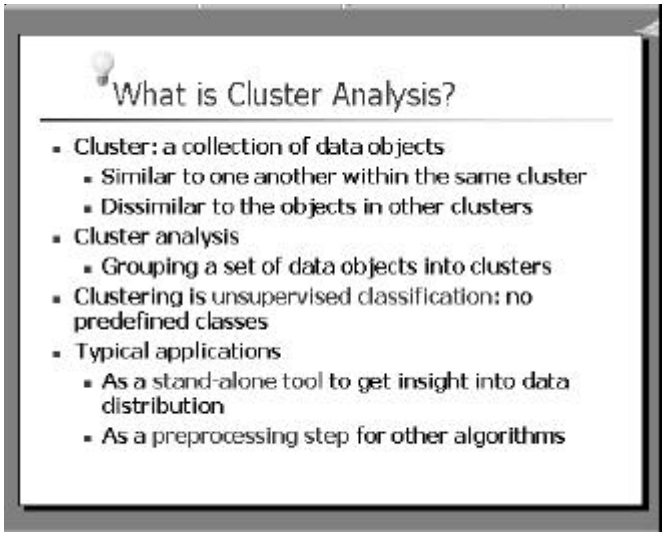
Many data mining applications make use of clustering according to similarity for example to segment a client/customer base. Clustering according to optimization of set functions is used in data analysis e.g. when setting insurance tariffs the customers can be segmented according to a number of parameters and the optimal tariff segmentation achieved.

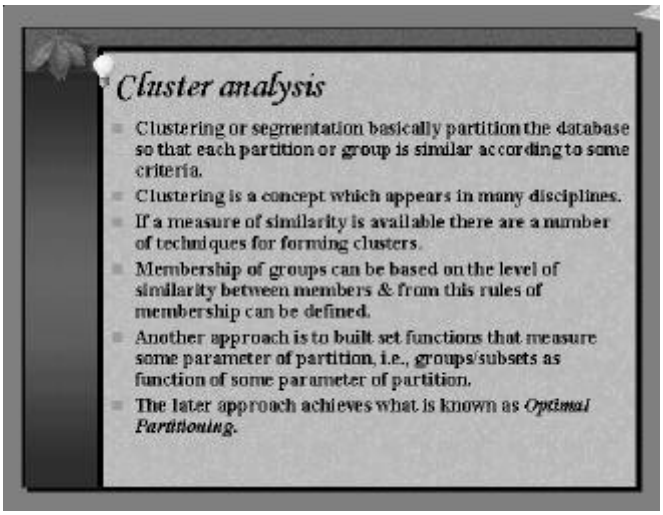
Clustering/segmentation in databases are the processes of separating a data set into components that reflect a consistent pattern of behavior. Once the patterns have been established they can then be used to "deconstruct" data into more under-standable subsets and also they provide sub-groups of a population for further analysis or action, which is important when dealing with very large databases. For example a database could be used for profile generation for target marketing where previous response to mailing campaigns can be used to generate a profile of people who responded and this can be used to predict response and filter mailing lists to achieve the best response.

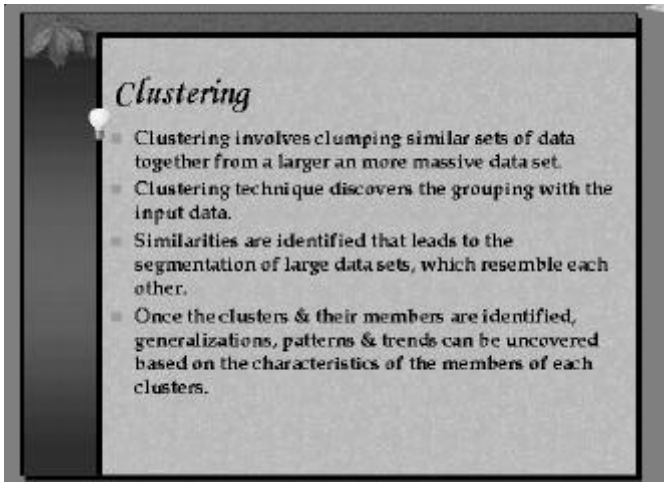
Discussion

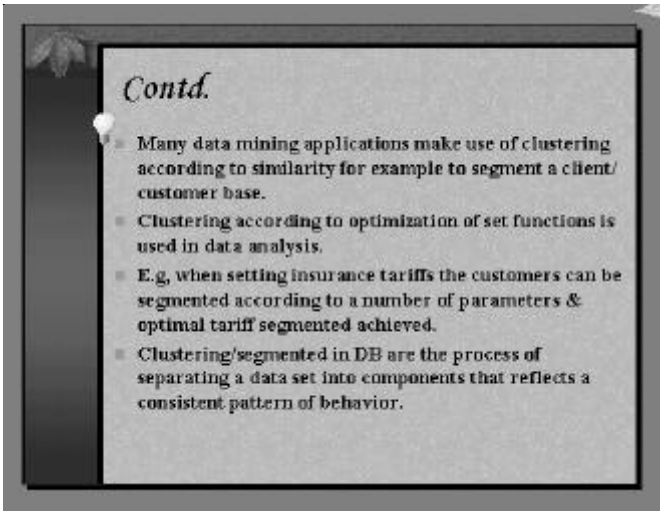
1. Write short notes on:
   - Clustering
   - Sequential patterns
   - Segmentation
   - Association rules
   - Classification hierarchies
2. Explain Cluster analysis.
3. Correctly contrast the difference between supervised and unsupervised learning.
4. Discuss in brief, where Clustering and Nearest-Neighbor Prediction are used?
5. How is the space for clustering and nearest neighbor defined? Explain.
6. What is the difference between clustering nearest-neighbor prediction?
7. How are tradeoffs made when determining which records fall into which clusters?
8. Explain the following:
   - Clustering
   - Nearest-Neighbor

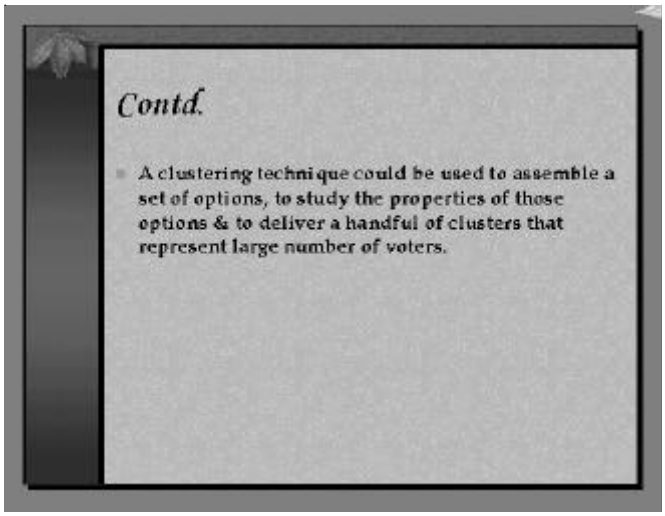
Some important slides

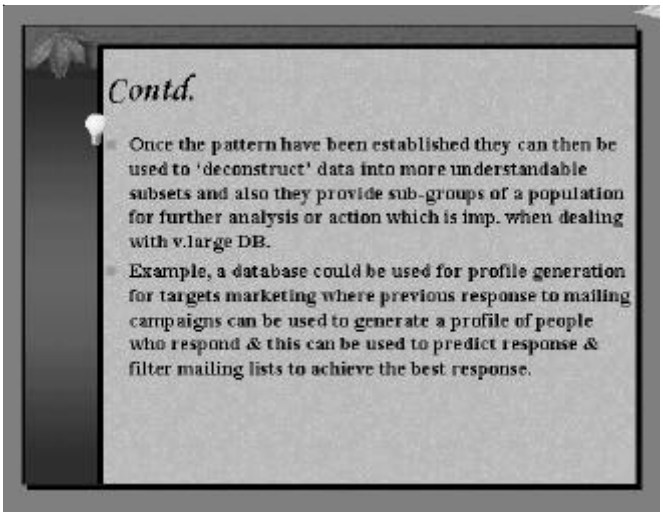












LESSON 23
DECISION TREES

Structure

- Objective
- Introduction
- What is a Decision Tree?
- Advantages and Shortcomings of Decision Tree Classifications:
- Where to Use Decision Trees?
- Tree Construction Principle
- The Generic Algorithm
- Guillotine Cut
- Overfit

Objective

At the end of this lesson you will be able to understand Decision trees, as techniques for data mining.

Introduction

The classification of large data sets is an important problem in data mining. The classification problem can be simply stated as follows. For a database with a number of records and for a set of classes such that each record belongs to one of the given classes, the problem of classification is to decide the class to which a given record belongs. But there is much more to this than just simply classifying. The classification problem is also concerned with generating a description or a model for each class from the given data set. Here, we are concerned with a type of classification called supervised classification. In supervised classification, we have a training data set of records and for each record of this set; the respective class to which it belongs is also known. Using the training set, the classification process attempts to generate the descriptions of the classes, and these descriptions help to classify the unknown records. In addition to the training set, we can also have a test data set, which is used to determine the effectiveness of a classification. There are several approaches to supervised classifications. *Decision trees* (essentially, *Classification trees)* are especially attractive in the data-mining environment as they represent rules. Rules can readily be expressed in natural language and are easily comprehensible. Rules can also be easily mapped to a database access language, like SQL.

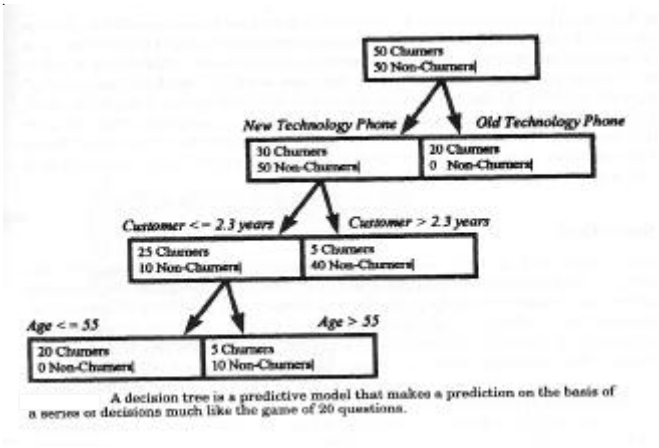
This lesson is concerned with decision trees as techniques for data mining. Though the decision-tree method is a well-known technique in statistics and machine learning, these algorithms are not suitable for data mining purposes. The specific require-ments that should be taken into consideration while designing any decision tree construction algorithms for data mining are that:

a. The method should be efficient in order to handle a very large-sized database,

b. The method should be able to handle categorical attributes.

What is a Decision Tree?

A decision tree is a predictive model that, as its name implies, can be viewed as a tree. Specifically, each branch of the tree is a classification question, and the leaves of the tree are partitions of the data set with their classification. For instance, if we were going to classify customers who churn (don't renew their phone contracts) in the cellular telephone industry, a decision tree might look something like that found in following figure.



A decision tree is a predictive model that makes a prediction on the basis of a series of decisions much like the game of 20 questions.

You may notice some interesting things about the tree.

- It divides the data on each branch point without losing any of the data (the number of total records in a given parent node is equal to the sum of the records contained in its two children).
- The number of churners and no churners is conserved as you move up or down the tree.
- It is pretty easy to understand how the model is being built (in contrast to
- The models from neural networks or from standard statistics).
- It would also be pretty easy to use this model if you actually had to target
- Those customers who are likely to churn with a targeted marketing offer.
- You may also build some intuitions about your customer base, for example,
- Customers who have been with you for a couple of years and have up-to-date
- Cellular phones and are pretty loyal

From a business perspective, decision trees can be viewed as creating a seg-mentation of the original data set (each segment would be one of the leaves of the tree). Segmentation of

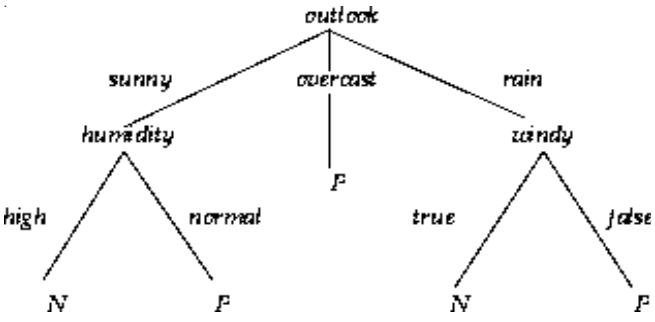
customers, products, and sales regions is something that marketing managers have been doing for many years. In the past this seg-mentation has been performed in order to get a high-level view of a large amount of data-with no particular reason for creating the segmentation except that the records within each segmentation were somewhat similar to each other.

In this case the segmentation is done for a particular reason-namely, for the prediction of some important piece of information. The records that fall within each segment fall there because they have similarity with respect to the infor-mation being predicted-not just that they are similar-without "similar-ity" being well defined. These predictive segments that are derived from the deci-sion tree also come with a description of the characteristics that define the pre-dictive segment. Thus the decision trees and the algorithms that create them may be complex, but the results can be presented in an easy-to-understand way that can be quite useful to the business user.

Decision Trees

Decision trees are simple knowledge representation and they classify examples to a finite number of classes, the nodes are labeled with attribute names, the edges are labeled with possible values for this attribute and the leaves labeled 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.



Decision Tree Structure

In order to have a clear idea of a decision tree, I have explained it with the following examples:

Example 23.1

Let us consider the following data sets-the training data set (see Table 23.1) and the test data set (see Table 23.2). The data set has five attributes.

Table 23.1 Training Data Set

OUTLOOK	TEMP(F)	HUMIDITY(%)	WINDY	CLASS
sunny	79	90	true	no play
sunny	56	70	false	play
sunny	79	75	true	play
sunny	60	90	true	no play
overcast	88	88	false	no play
overcast	63	75	true	play
overcast	88	95	false	play
ram	78	60	false	play
ram	66	70	false	no play
ram	68	60	true	no play

There is a special attribute: the attribute *class* is the class label. The attributes, *temp* (temperature) and *humidity* are numerical attributes and the other attributes are categorical, that is, they cannot be ordered. Based on the training data set, we want to find a set of rules to know what values of *outlook, temperature, humidity* and *wind,* determine whether or not to play golf. Figure 23.1 gives a sample decision tree for illustration.

In the above tree (Figure 23.1), we have five leaf nodes. In a decision' tree, each leaf node represents a rule. We have the following rules corresponding to the tree given in Figure 23.1.

RULE 1 If it is sunny and the humidity is not above 75%, then play.

RULE 2 If it is sunny and the humidity is above 75%, then do not play.

RULE 3 If it is overcast, then play.

RULE 4 If it is rainy and not windy, then play.

RULE 5 If it is rainy and windy. then don't play.

Please note that this may not be the best set of rules that can _be derived from the given set of training data.

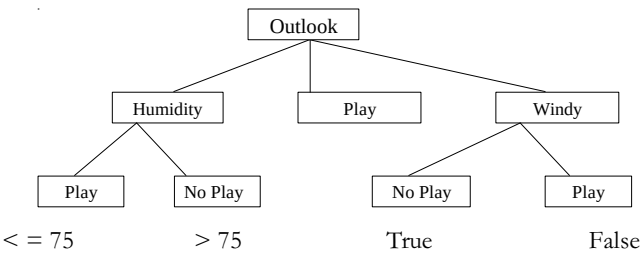


Figure 23.1 A Decision Tree

The classification of an unknown input vector is done by traversing the tree from the root node to a leaf node. A record enters the tree at the root node. At the root, a test is applied to determine which child node the record will encounter next. This process is repeated until the record arrives at a leaf node. All the records that end up at a given leaf of the tree are classified in the same way. There is a unique path from the root to each leaf. The path is a rule, which is used to classify the records.

In the above tree, we can carry out the classification for an unknown record as follows. Let us assume" for the record, that we know the values of the first four attributes (but we do not know the value of class attribute) as

outlook= rain; temp = 70; humidity = 65; and windy= true.

We start from the root node to check the value of the attribute associated at the root node. This attribute is the splitting attribute at this node. Please note that for a decision tree, at every node there is an attribute: associated with the node called the splitting attribute. In our example, outlook is the splitting attribute at root. Since for the given record, outlook = rain, we move to the right-most child node of the root. At this node, the splitting attribute is windy and we find that for the record we want classify, windy = true. Hence, we move to the left child node to conclude that the class label is "no play".

Note that every path from root node to leaf nodes represents a rule. It may be noted that many different leaves of the tree may refer to the same class labels, but each leaf refers to a different rule.

The accuracy of the classifier is determined by the percentage of the test data set that is correctly classified. Consider the following test data set (Table 23.2).

Table 23.2 Test Data Set

OUTLOOK	TEMP(F)	HUMIDITY(%)	WINDY	CLASS
sunny	79	90	true	play
sun)1Y	56	70	false	play
sunny	79	75	true	no play
sunny	50	90	true	no play
overcast	88	88	false	no play
overcast	63	75	true	Play
overcast	88	95	false	Play
ram	78	60	false	play
ram	66	70	false	no play
rain	68	60	true	play

We can see that for Rule 1 there are two records of the test data set satisfying outlook= sunny and humidity s 75, and only one of these is correctly classified as play. Thus, the accuracy of this rule is 0.5 (or 50%). Similarly, the accuracy of Rule 2 is also 0'.5 (or 50%). The accuracy of Rule 3 is 0.66.

Example 23.2

At this stage, let us consider another example to illustrate the concept of categorical attributes. Consider the following training data set (Table 23.3). There are three attributes, namely, age, pin code and class. The attribute class is used for class label.

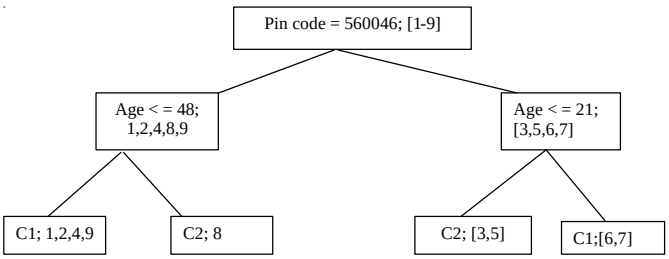
Table 23.3 Another Example

ID	AGE	PINCODE	CLASS
1	30	5600046	Cl
2	25	5600046	Cl
3	21	5600023	C2
4'	43	5600046	Cl
5	18	5600023	C2
6	33	5600023	Cl
7	29	5600023	Cl
8	55	5600046	C2
9	48	5600046	Cl

The attribute *age* is a numeric attribute, whereas pincode is a categorical one. Though the domain of pincode is numeric, no ordering can be defined among pincode values. You cannot derive any useful information if one pin-code is greater than another pincode. Figure 23.2 gives a decision tree for this training data. The splitting attribute at the root is pincode and the splitting criterion here is pincode = 500 046. Similarly, for the left child node, the splitting criterion is age £ 48 (the splitting attribute is age). Although the right child node has the same attribute as the splitting attribute, the splitting criterion is different.

Most decision tree building algorithms begin by trying to find the test, which does the best job of splitting the records among the desired categories. At each succeeding level of the tree, the subsets created by the preceding split are themselves split according to whatever rule works best for them. The tree continues to grow until it is no longer possible to find better ways to split up incoming records, or when all the records are in one class.

Figure 23.2 A Decision Tree



In Figure 23.2, we see that at the root level we have 9 records. The associated splitting criterion is pincode = 500 046. As a result, we split the records into two subsets, Records 1, 2, 4, 8 and 9 are to the left child node and the remaining to the right node. This process is repeated at every node.

A decision tree construction process is concerned with identifying the splitting attributes and splitting criteria at every level of the tree. There are several alternatives and the main aim of the

decision tree construction process is to generate simple, comprehensible rules with high accuracy.

Some rules are apparently better than others. In the above example, we see that Rule 3 is simpler than Rule 1. The measure of simplicity is the number of antecedents of the rule. It may happen that another decision tree may yield a rule like: "if the temperature lies between 70° F and 80° F, and the humidity is between 75% and 90%, and it is not windy, and it is sunny, then play". Naturally, we would prefer a rule like Rule 1 to this rule. That is why simplicity is sought after.

Sometimes the classification efficiency of the tree can be improved by revising the Tree through some processes like pruning and grafting. These processes are activated after the decision tree is built.

Advantages and Shortcomings of Decision Tree Classifications:

The major strength~ of the decision tree methods are the following:

- Decision trees are able to generate understandable rules,
- They are able to handle both numerical and the categorical attributes, and
- They provide a clear indication of which fields are most important for prediction or classification.

Some of the weaknesses of the decision trees are:

- Some decision trees can only deal with binary-valued target classes. Others are able to assign records to an arbitrary number of classes, but are error-prone when the number of training examples per class gets small. This can happen rather quickly in a tree with many levels and/or many branches per node.
- The process of growing a decision tree is computationally expensive. At each node, each candidate splitting field is examined before its best split can be found.

Where to use Decision Trees?

Decision trees are a form of data mining technology that has been around in a form very similar to the technology of today for almost 20 years now, and early versions of the algorithms date back till the 1960s. Often these techniques were originally developed for statisticians to automate the process of determining which fields in their database were actually useful or correlated with the particular problem that they were trying to understand. Partly because of this history, decision tree algorithms tend to automate the entire process of hypothesis generation and then validation much more completely and in a much more integrated way than any other data mining techniques. They are also particularly adept at handling raw data with little or no preprocessing. Perhaps also because they were originally developed to mimic the way an analyst interactively performs data mining, they provide a simple-to-understand predictive model based on rules (such as "90 percent of the time credit card customers of less than 3 months who max out their credit limits are going to default on their credit card loans").

Because decision trees score so highly on so many of the critical features of data mining, they can be used in a wide variety of business problems for both exploration and prediction. They have been used for problems ranging from credit card attrition prediction to time series prediction of the exchange rate of different international currencies. There are also some problems where decision trees will not do as well. Some very simple problems in which the prediction is just a simple multiple of the predictor can be solved much more quickly and easily by linear regression. Usually the models to be built and the interactions to be detected are much more complex in real-world problems, and this is where decision trees excel.

Tree Construction Principle

After having understood the basic features of decision trees, we shall now focus on the methods of building such trees from a given training data set. Based on the foregoing discussion, I shall formally define few concepts for your study.

Definition 23.1: Splitting Attribute

With every node of the decision tree, there is an associated attribute whose values determine the partitioning of the data set when the node is expanded.

Definition 23.2: Splitting Criterion

The qualifying condition on the splitting attribute for data set splitting at a node, is called the splitting criterion at that node. For a numeric attribute, the criterion can be an equation or an inequality. For a categorical attribute, it is a membership condition on a subset of values.

All the decision tree construction methods are based on the principle of recursively' partitioning the data set till homogeneity is achieved. We shall study this common principle later and discuss in detail the features of different algorithms individually. The construction of the decision tree involves the following three main phases.

- Construction phase - The initial decision tree is constructed in this phase, based on the entire training data set. It requires recursively partitioning the training set, into two, or more, sub-partitions using splitting criteria, until a stopping criteria is met.
- Pruning phase - The tree constructed in the previous phase may not result in the best possible set of rules due to overfitting (explained below). The pruning phase removes some of the lower branches and nodes to improve its performance.
- Processing the pruned tree - to improve understandability.

Though these three phases are common to most of the well-known algorithms, some of them attempt to integrate the first two phases into a single process.

The Generic Algorithm

Most of the existing algorithms of the construction phase use Hunt's method as the basic principle in this phase. Let the training data set be T with class-labels {C1, C2, .., Ck}, The tree is built by repeatedly partitioning the training data, using some criterion like the goodness of the split. The process is continued till all the records in a partition belong to the same class.

- T is homogeneous T contains cases all belonging to a single class Cj. The decision tree for T is a leaf identifying class Cj.
- T is not homogeneous T contains cases that belong to a mixture of classes. A test is chosen, based on a single attribute, that has one or more mutually exclusive outcomes

{Ol, O2, ..., On} .T is partitioned into the subsets T1, T2, T3, ..., Tn where T1; contains all those cases in T that have the outcome OJ of the chosen test. The decision tree for T consists of a decision node identifying the test, and one branch for each possible outcome. The same tree building method is applied recursively to each subset of training cases. Most often, n is chosen to be 2 and hence, the algorithm generates a binary decision tree.

- T is trivial T contains no cases. The decision tree T is a leaf, but the class to be associated with the leaf must be determined from information other than T.

The generic algorithm of decision tree construction outlines the common principle of all algorithms. Nevertheless, the following aspects should be taken into account while studying any specific algorithm. In one sense, the following are three major difficulties, which arise when one uses a decision tree in a real-life situation.

Guillotine Cut

Most decision tree algorithms examine only a single attribute at a time. As mentioned in the earlier paragraph, normally the splitting is done for a single attribute at any stage and if the attribute is numeric, then the splitting test is an inequality. Geometrically, each splitting can be viewed as a plane parallel to one of the axes. Thus, splitting one single attribute leads to rectangular classification boxes that may not correspond too well with the actual distribution of records in the decision space. We call this the guillotine cut phenomenon. The test is of the form (X> z) or (X < z), which is called a guillotine cut, since it creates a guillotine cut subdivision of the Cartesian space of the ranges of attributes.

However, the guillotine cut approach has a serious problem if a pair of attributes are correlated. For example, let us consider two numeric attributes, height (in meters) and weight (in Kilograms). Obviously, these attributes have a strong correlation. Thus, whenever there exists a correlation between variables, a decision tree with the splitting criteria on a single attribute is not accurate. Therefore, some researchers propose an oblique decision tree that uses a splitting criteria involving more than one attribute.

Over Fit

Decision trees are built from the available data. However, the training data set may not be a proper representative of the real-life situation and may also contain noise. In an attempt to build a tree from a noisy training data set, we may grow a decision tree just deeply enough to perfectly classify the training data set.

Definition 23.3 Overfit

A decision tree T is said to over fit the training data if there exists some other tree T' which is a simplification of T, such that T has smaller error than T' over the training set but T' has a smaller error than T over the entire distribution of instances.

Overfitting can lead to difficulties when there is noise in the training data, or when the number of training examples is too small. Specifically, if there is no conflicting instances in the training data set, the error of the fully built tree is zero, while the true error is likely to be bigger. There are many disadvantages of an overfitted decision tree:

a. Overfitted models are incorrect

b. Overfitted decision trees require more space and more computational resources.

c. Overfitted models require the collection of unnecessary features

d. They are more difficult to comprehend.

The pruning phase helps in handling the overfitting problem. The decision tree is pruned back by removing the subtree rooted at a node and replacing it by a leaf node, using some criterion. Several pruning algorithms are reported in literature.

In the next lesson we will study about Decision Tress Construction Algorithms and the working of decision trees.

Exercises

1. What is a decision tree? Illustrate with an example.
2. Describe the essential features in a decision tree. How is it useful to classify data?
3. What is a classification problem? What is supervised classification? How is a decision tree useful in classification?
4. Explain where to use Decision Trees?
5. What are the disadvantages of the decision tree over other classification techniques?
6. What are advantages and disadvantages of the decision tree approach over other approaches of data mining?
7. What are the three phases of construction of a decision tree? Describe the importance of each of the phases.
8. The 103 generates a
   a. Binary decision tree
   b. A decision tree with as many branches as there are distinct values of the attribute
   c. A tree with a variable number of branches, not related to the domain of the attributes
   d. A tree with an exponential number of branches.

Suggested Readings

1. Pieter Adriaans, Dolf Zantinge Data Mining, Pearson Education, 1996
2. George M. Marakas Modern Data Warehousing, Mining, and Visualization: Core Concepts, Prentice Hall, 1st edition, 2002
3. Alex Berson, Stephen J. Smith Data Warehousing, Data Mining, and OLAP (Data Warehousing/Data Management), McGraw-Hill, 1997
4. Margaret H. Dunham Data Mining, Prentice Hall, 1st edition, 2002
5. David J. Hand Principles of Data Mining (Adaptive Computation and Machine Learning), Prentice Hall, 1st edition, 2002
6. Jiawei Han, Micheline Kamber Data Mining, Prentice Hall, 1st edition, 2002
7. Michael J. Corey, Michael Abbey, Ben Taub, Ian Abramson Oracle 8i Data Warehousing McGraw-Hill Osborne Media, 2nd edition, 2001

LESSON 24
DECISION TREES - 2

Structure

- Objective
- Introduction
- Best Split
- Decision Tree Construction Algorithms
- CART
- ID3
- C4.5
- CHAID
- When does the tree stop growing?
- Why would a decision tree algorithm prevent the tree from growing if there weren't enough data?
- Decision trees aren't necessarily finished after they are fully grown
- Are the splits at each level of the tree always binary yes/no splits?
- Picking the best predictors
- How decision trees works?

Objective

The objective of this lesson is to introduce you with decision tree construction algorithms along with the working of decision trees.

Introduction

In this lesson, I will explain you various kinds of decision tree construction algorithms like CART, ID3, C4.5 and CHAID. You will study about the working of decision trees in detail.

Best Split

We have noticed that there are several alternatives to choose from for the splitting attribute and the splitting criterion. But in order to build an optimal decision tree, it is necessary to select those corresponding to the best possible split. The main operations during the tree building are

1. Evaluation of splits for each attribute and the selection of the best split; determination of the splitting attribute,
2. Determination of the splitting condition on the selected splitting attribute, and
3. Partitioning the data using the best split.

The complexity lies in determining the best split for each attribute. The splitting also depends on the domain of the attribute being numerical or categorical. The generic algorithm for the construction of decision trees assumes that the method to decide the splitting attribute at a node and the splitting criteria are known. The desirable feature of splitting is that it should do the best job of splitting at the given stage.

The first task is to decide which of the independent attributes makes the best splitter. The best split is defined as one that does the best job of separating the records into groups, where a single class predominates. To choose the best splitter at a node, we consider each independent attribute in turn.

Assuming that an attribute takes on multiple values, we sort it and then, using some evaluation function as the measure of goodness, evaluate each split. We compare the effectiveness of the split provided by the best splitter from each attribute. The winner is chosen as the splitter for the root node. How does one know which split is better than the other? We shall discuss below two different evaluation functions to determine the splitting attributes and the splitting criteria.

Decision Tree Construction Algorithms

A number of algorithms for inducing decision trees have been proposed over the years. However, they differ among themselves in the methods employed for selecting splitting attributes and splitting conditions. In the following few sections, we shall study some of the major methods of decision tree constructions.

CART

CART (Classification And Regression Tree) is one of the popular methods of building decision trees in the machine learning community. CART builds a binary decision tree by splitting the records at each node, according to a function of a single attribute. CART uses the gini index for determining the best split. CART follows the above principle of constructing the decision tree. We outline the method for the sake of completeness

The initial split produces two nodes, each of which we now attempt to split in the same manner as the root node. Once again, we examine all the input fields to find the candidate splitters. If no split can be found that significantly decreases the diversity of a given node, we label it as a leaf node. Eventually, only leaf nodes remain and we have grown the full decision tree. The full tree may generally not be the tree that does the best job of classifying a new set of records, because of overfitting.

At the end of the tree-growing process, every record of the training set has been assigned to some leaf of the full decision tree. Each leaf can now be assigned a class and an error rate. The error rate of a leaf node is the percentage of incorrect classification at that node. The error rate of an entire decision tree is a weighted sum of the error rates of all the leaves. Each leaf's contribution to the total is the error rate at that leaf multiplied by the probability that a record will end up in there.

ID3

Quinlan introduced the ID3, Iterative Dichotomizer 3, for constructing the decision trees from data. In ID3, each node corresponds to a splitting attribute and each arc is a possible value of that attribute. At each node the splitting attribute is selected to be the most informative among the attributes not

yet considered in the path from the root. Entropy is used to measure how informative is a node. This algorithm uses the criterion of information gain to determine the goodness of a split. The attribute with the greatest information gain is taken as the splitting attribute, and the data set is split for all distinct values of the attribute.

C4.5

C4.5 is an extension of ID3 that accounts for unavailable values, continuous attribute value ranges, pruning of decision trees and rule derivation. In building a decision tree, we can deal with training sets that have records with unknown attribute values by evaluating the gain, or the gain ratio, for an attribute by considering only those records where those attribute values are available. We can classify records that have unknown attribute values by estimating the probability of the various possible results. Unlike CART, which generates a binary decision tree, C4.5 produces trees with variable branches per node. When a discrete variable is chosen as the splitting attribute in C4.5, there will be one branch for each value of the attribute.

CHAID

CHAID, proposed by Kass in 1980, is a derivative of AID (Automatic Interaction Detection), proposed by Hartigan in 1975. CHAID attempts to stop growing the tree before overfitting occurs, whereas the above algorithms generate a fully grown tree and then carry out pruning as post-processing step. In that sense, CHAID avoids the pruning phase.

In the standard manner, the decision tree is constructed by partitioning the data set into two or more subsets, based on the values of one of the non-class attributes. After the data set is partitioned according to the chosen attributes, each subset is considered for further partitioning using the same algorithm. Each subset is partitioned without regard to any other subset. This process is repeated for each subset until some stopping criterion is met. In CHAID, the number of subsets in a partition can range from two up to the number of distinct values of the splitting attribute. In this regard, CHAID differs from CART, which always forms binary splits, and from ID3 or C4.5, which form a branch for every distinct value.

The splitting attribute is chosen as the one that is most significantly associated with the dependent attributes according to a chi-squared test of independence in a contingency table (a cross-tabulation of the non-class and class attribute). The main stopping criterion used by such methods is the p-value from this chi-squared test. A small p-value indicates that the observed association between the splitting attribute and the dependent variable is unlikely to have occurred solely as the result of sampling variability.

If a splitting attribute has more than two possible values, then there may be a very large number of ways to partition the data set based on these values. A combinatorial search algorithm can be used to find a partition that has a small p-value for the chi-squared test. The p-values for each chi-squared test are adjusted for the multiplicity of partitions. A Bonferroni adjustment is used for the p-values computed from the contingency tables, relating the predictors to the dependent variable. The adjustment is conditional on the number of branches (compound categories) in the partition, and thus does not take into account the fact that different numbers of branches are considered.

When does the tree stop growing?

If the decision tree algorithm just continued like this, it could conceivably cre-ate more and more questions and branches in the tree so that eventually there was only one record in the segment. To let the tree grow to this size is compu-tationally expensive and also unnecessary. Most decision tree algorithms stop growing the tree when one of three criteria are met:

1. The segment contains only one record or some algorithmically defined min-imum number of records, (Clearly, there is no way to break a smile-record segment into two smaller segments, and segments with very few records are not likely to be very helpful in the final prediction since the predictions that they are making won't be based on sufficient historical data.)
2. The segment is completely organized into just one prediction value. There is no reason to continue further segmentation since this data is now com-pletely organized (the tree has achieved its goal).
3. The improvement in organization is not sufficient to warrant making the split. For instance, if the starting segment were 90 percent churners and the resulting segments from the best possible question were 90.001 percent churners and 89.999 percent churners, then not much progress would have been or could be made by continuing to build the tree.

Why would a decision tree algorithm prevent the tree from growing if there weren't enough data?

Consider the following example of a segment that we might want to split further because it has only two examples. Assume that it has been created out of a much larger customer database by selecting only those customers aged 27 with blue eyes and with salaries ranging between $80,000 and $81,000.

In this case all the possible questions that could be asked about the two cus-tomers turn out to have the same value (age, eye color, salary) except for name.

TABLE: Decision Tree Algorithm Segment>

Name	Age	Eyes	Salary ($)	Churned?
Steve	27	Blue	80,000	Yes
Alex	27	Blue	80,000	No

* This segment cannot be split further except by using the predictor "name."

Decision trees aren't necessarily finished after they are fully grown

After the tree has been grown to a certain size (depending on the particular stop-ping criteria used in the algorithm), the CART algorithm has still more work to do. The algorithm then checks to see if the model has been over fit to the data. It does this in several ways using a cross-validation approach or a test set valida-tion approach-basically using the same mind-numbingly simple approach it used to find the best questions in the first place: trying many different simpler versions of the tree on a held-aside test set. The algorithm as the best model selects the tree that does the best on the held-aside data. The

nice thing about CART is that this testing and selection is all an integral part of the algo-rithm as opposed to the after-the-fact approach that other techniques use.

Are the splits at each level of the tree always binary yes/no splits?

There are several different methods of building decision trees, some of which can make splits on multiple values at time-for instance, eye color: green, blue, and brown. But recognize that any tree that can do binary splits can effectively partition the data in the same way by just building two levels of the tree: the first, which splits brown and blue from green; and the second, which splits apart the brown and blue split. Either way, the minimum number of questions you need to ask is two.

Table - Two Possible Splits for Eight Records with Calculation of Entropy for Each Split Shown*

Two Possible Splits for Eight Records with Calculation of Entropy for Each Split Shown*

Candidate	Left split	Right split	Left entropy	Right entropy
Split A	++++−	+−−−−	− ⅘ lg (⅘) + − ⅕ lg (⅕) = 0.72	− ⅕ lg (⅕) + − ⅘ lg (⅘) = 0.72
Split B	+++++−−−	−	− ⅝ lg (⅝) + − ⅜ lg (⅜) = 0.99	− ⅛ lg (⅛) + − ⅞ lg (⅞) = 0

* The positive and negative values for the prediction target are represented by plus and minus signs, respectively.

How the Decision Tree Works

In the late 1970s J. Ross Quinlan introduced a decision tree algorithm named ID3. This was one of the first decision tree algorithms though it was built solidly on previous work on inference systems and concept learning systems from that decade and the preceding decade. Initially ID3 was used for tasks such as learn-ing good game-playing strategies for chess end games. Since then ID3 has been applied to a wide variety of problems in both academia and industry and has been modi-fied, improved, and borrowed from many times over.

ID3 picks predictors and their splitting values on the basic of the gain in information that the split or splits provide. Gain represents the difference between the amount of information that is needed to correctly make a predic-tion both before and after the split has been made (if the amount of informa-tion required is much lower after the split is made, then that split has decreased the disorder of the original single segment) and is defined as the dif-ference between the entropy of the original segment and the accumulated entropies of the resulting split segments. Entropy is a well-defined measure of the disorder or information found in data.

The entropies of the child segments are accumulated by weighting their con-tribution to the entire entropy of the split according to the number of records they contain. For instance, which of the two splits shown in Table 18.7 would you think decreased the entropy the most and thus would provide the largest gain?

Split A is actually a much better split than B because it separates out more of the data despite the fact that split B creates a new segment that is perfectly homogeneous (0 entropy). The problem is that this perfect zero-entropy seg-ment has only one record in it and splitting off one record at a time will not cre-ate a very useful decision tree. The small number of records in each segment (Let 1) is unlikely to provide useful repeatable patterns. The calculation (met-ric) that we use to determine which split is chosen should make the correct choice in this case and others like it. The metric needs to take into account two main effects:

- How much has the disorder been lowered in the new segments?
- How should the disorder in each segment be weighted?

The entropy measure can easily be applied to each of the new segments as eas-ily as it was applied to the parent segment to answer the first question, but the second criterion is a bit harder. Should all segments that result from a split be treated equally? This question needs to be answered in the example above where the split has produced a perfect new segment but with little real value because of its size. If we just took the average entropy for the new segments, we would choose split B since in that case the average of 0.99 and 0.0 is around 0.5 We can also do this calculation for split A and come up with an average entropy of 0.72 for the new segments.

If, on the other hand, we weighted the contribution of each new segment with respect to the size of the segment (and consequently how much of the database that segment ex-plained), we would get a quite different measure of the disor-der across the two new segments. In this case the weighted entropy of the two segments for split A is the same as before but the weighted entropy of split B is quite a bit higher. (See the following Table)

Since the name of this game is to reduce entropy to as little as possible, we are faced with two different choices of which is the best split. If we average the entropies of the new segments, we would pick split B; if we took into account the number of records that are covered by each split, we would pick split A.

ID3 uses the weighted entropy approach as it has been found, in general, to produce better predictions than just averaging the entropy. Part of the reason for this may be that, as .we have seen from the modeling chapter, that the more data that is used in a prediction, the more likely the prediction is to be correct and the more likely the model is to match the true underlying causal reasons and processes that are actually at work in forming the prediction values.

Weighting the Entropy Values for Two Possible Shifts*

Candidate	Left split	Right split	Average entropy	Weighted entropy
Split A	++++−	+−−−−	0.72 = (0.72 + 0.72)/2	0.72 = (⅘) * 0.72 + (⅕) * 0.72
Split B	+++++−−−−	−	0.50 = (0.99 + 0)/2	0.89 = (⅝) * 0.99 + (⅛) * 0.0

* The better of the two possible splits can be chosen when the entropies are weighted by the size of the resulting segments.

State of the Industry

The current offerings in decision tree software emphasize different important aspects and use of the algorithm. The different emphases are usually driven because of differences in the targeted user and the types of problems being solved. There are four main categories of products:

- Business-those that emphasize ease of use for the business users
- Performance-those that emphasize the overall performance and database size
    - Exploratory-those that emphasize ease of use for the analyst
    - Research-those tailored specifically for detailed research or academic experimentation

Tools such as Pilot Software's Discovery Server fall into the category of busi-ness use. The Pilot Discovery Server (trade-mark) provides easy-to-use graphi-cal tools to help the business user express their modeling problem and also provides applications such as the Pilot Segment Viewer and Pilot Profit Chart (both trademarks) to allows business end users to visualize the model and per-form simple profitless models on different targeted marketing applications. A tool that falls into the performance category would be Thinking Machines Cor-poration's Star Tree tool, which implements CART on MPP and SMP computer hardware and has been optimized for large databases and difficult-to-solve problems. Angoss' Knowledge Seeker (trademark) tool, on the other hand, is targeted mostly at the PC user but provides more control to the analyst to spec-ify different parameters that control the underlying CHAID algorithm, if desired. Salford Systems' CART product provides even more control over the underlying algorithms but provides only limited GUI or applications support; however, it is useful to researchers and business analysts who want in-depth analysis and control over their model creation.

Exercises

1. What are advantages and disadvantages of the decision tree approach over other approaches of data mining?
2. Describe the ID3 algorithm of the decision tree construction. Why is it unsuitable for data mining applications?
3. Consider the following examples

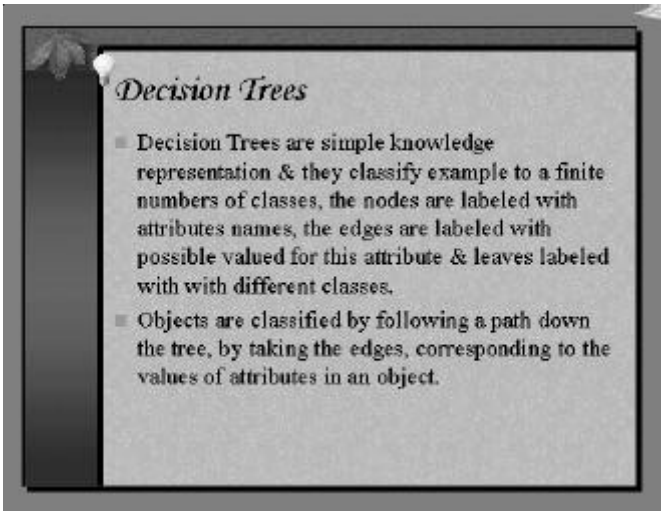
MOTOR	WHEELS	DOORS	SIZE	TYPE	CLASS
NO	2	0	small	cycle	bicycle
NO	3	0	small	cycle	tricycle
YES	2	0	small	cycle	motorcycle
YES	4	2	small	automobile	Sports car
YES	4	3	medium	automobile	minivan
YES	4	4	medium	automobile	sedan
YES	4	4	large	automobile	sumo

Use this example to illustrate the working of different algo-rithms.

4. Overfitting is an inherent characteristic of decision tree and its occurrence depends on the construction process and not on the training data set. True or False?
5. Pruning is essentially to avoid overfitting. True or False?
6. Bootstrapping is carried out in the main memory. True or False?
7. Write short notes on:
    - C4.5
    - CHAID
    - ID3
    - CART

Suggested Readings

1. **Pieter Adriaans, Dolf Zantinge** *Data Mining,* Pearson Education, 1996
2. **George M. Marakas** *Modern Data Warehousing, Mining, and Visualization: Core Concepts*, Prentice Hall, 1st edition, 2002
3. **Alex Berson, Stephen J. Smith** *Data Warehousing, Data Mining, and OLAP (Data Warehousing/Data Management),* McGraw-Hill, 1997
4. **Margaret H. Dunham** *Data Mining,* Prentice Hall, 1st edition, 2002
5. **David J. Hand** *Principles of Data Mining (Adaptive Computation and Machine Learning),* Prentice Hall, 1st edition, 2002
6. **Jiawei Han, Micheline Kamber** *Data Mining,* Prentice Hall, 1st edition, 2002
7. **Michael J. Corey, Michael Abbey, Ben Taub, Ian Abramson** *Oracle 8i Data Warehousing* McGraw-Hill Osborne Media, 2nd edition, 2001



LESSON 25
NEURAL NETWORKS

Structure

- Objective
- Introduction
- What is a Neural Network?
- Learning in NN
- Unsupervised Learning
- Data Mining using NN: A Case Study

Objective

The aim of this lesson is to introduce you with the concept of Neural Networks. It also includes various topics, which explains how the method of neural network is helpful in extracting the knowledge from the warehouse.

Introduction

Data mining is essentially a task of learning from data and hence, any known technique which attempts to learn from data can, in principle, be applied for data mining purposes. In general, data mining algorithms aim at minimizing I/O operations of disk-resident data, whereas conventional algorithms are more concerned about time and space complexities, accuracy and convergence. Besides the techniques discussed in the earlier lessons, a few other techniques hold promise of being suitable for data mining purposes. These are Neural Networks (NN), Genetic Algorithms (GA) and Support Vector Machines (SVM). The intention of this chapter is to briefly present you the underlying concepts of these subjects and demonstrate their applicability to data mining. We envisage that in the coming years these techniques are going to be important areas of data mining techniques.

Neural Networks

The first question that comes to the mind is, what is this Neural Network?

When data mining algorithms are discussed these days, people are usually talking about either decision trees or neural networks. Of the two, neural net-works have probably been of greater interest through the formative stages of data mining technology. As we will see, neural networks do have disadvantages that can be limiting in their ease of use and ease of deployment, but they do also have some significant advantages. Foremost among these advantages are their highly accurate predictive models, which can be applied across a large number of different types of problems.

To be more precise, the term neural network might be defined as an "artifi-cial" neural network. True neural networks are biological systems [also known as (a.k.a.) brains] that detect patterns, make predictions, and learn. The artifi-cial ones are computer programs implementing sophisticated pattern detection and machine learning algorithms on a computer to build predictive models from large historical databases. Artificial neural networks derive their name from their historical development, which started off with the premise that machines could be made to "think" if scientists found ways to mimic the structure and functioning of the human brain on the computer. Thus historically neural networks grew out of the community of artificial intelligence rather than from the discipline of statistics. Although scientists are still far from understanding the human brain, let alone mimicking it, neural networks that run on computers can do some of the things that people can do.

lt is difficult to say exactly when the first "neural network" on a computer was built. During World War II a seminal paper was published by McCulloch and Pitts which first outlined the idea that simple processing units (like the individual neurons in the human brain) could be connected together in large \networks to create a system that could solve difficult problems and display behavior that' was much more complex than the simple pieces that made it up. Since that time much progress has been made in finding ways to apply artifi cial neural networks to real-world prediction problems and improving the per-formance of the algorithm in general. In many respects the greatest breakthrough in neural networks in recent years have been in their application to more mundane real-world problems such as customer response prediction or fraud detection rather than the loftier goals that were originally set out for the techniques such as overall human learning and computer speech and image understanding.

Don't neural networks learn to make better predictions?

Because of the origins of the techniques and because of some of their early suc-cesses, the techniques have enjoyed a great deal of interest. To understand how neural networks can detect patterns in a database, an analogy is often made that they "learn" to detect these patterns and make better predictions, similar to the way human beings do. This view is encouraged by the way the historical training data is often supplied the network-one record (example) at a time.

Networks do "learn" in a very real sense, but under the hood, the algo-rithms and techniques that are being deployed are not truly different from the techniques found in statistics or other data mining algorithms. It is, for instance, unfair to assume that neural networks could outperform other tech-niques because they "learn" and improve over time while the other techniques remain static. The other techniques, in fact, "learn" from historical examples in exactly the same way, but often the examples (historical records) to learn from are processed all at once in a more efficient manner than are neural networks, which often modify their model one record at a time.

Are neural networks easy to use?

A common claim for neural networks is that they are automated to a degree where the user does not need to know that much about how they work, or about predictive modeling or even the

database in order to use them. The implicit claim is also that most neural networks can be unleashed on your data straight out of the box without the need to, rearrange or modify the data very much to begin with.

Just the opposite is often true. Many important design decisions need to be made in order to effectively use a neural network, such as

- How should the nodes in the network be connected?
- How many neurons like processing units should be used?
- When should "training" be stopped in order to avoid over fitting?

There are also many important steps required for preprocessing the data that goes into a neural network-most often there is a requirement to normalize numeric data between 0.0 and 1.0, and categorical predictors may need to be broken up into virtual predictors that are 0 or 1 for each value of the original categorical predictor. And, as always, understanding what the data in your database means and a clear definition of the business problem to be solved are essential to ensuring eventual success. The bottom line is that neural networks provide no shortcuts.

Business Scorecard

Neural networks are very powerful predictive modeling techniques, but some of the power comes at the expense of ease of use and ease of deployment. As we will see in this chapter, neural networks create very complex models that are almost always impossible to fully understand, even by experts. The model itself is represented by numeric values in a complex calculation that requires all the predictor values to be in the form of a number. The output of the neural net-work is also numeric and needs to be translated if the actual prediction value is categorical (e.g., predicting the demand for blue, white, or black jeans for a clothing manufacturer requires that the predictor values blue, black and white for the predictor color be converted to numbers). Because of the complexity of these techniques, much effort has been expended in trying to increase the clar-ity with which the model can be understood by the end user. These efforts are still in their infancy but are of tremendous importance since most data mining techniques including neural networks are being deployed against real business problems where significant investments are made on the basis of the predic-tions from the models (e.g., consider trusting the predictive model from a neural network that dictates which one million customers will receive a $1 mailing).

These shortcomings in understanding the meaning of the neural network model have been successfully addressed in two ways:

1. The neural network is packaged up into a complete solution such as fraud prediction. This allows the neural network to be carefully crafted for one particular application, and once it has been proven successful, it can be used over and over again without requiring a deep understanding of how it works.
2. The neural network is packaged up with expert consulting services. Here trusted experts who have a track record of success deploy the neural network. The experts either are able to explain the models or trust that the models do work.

The first tactic has seemed to work quite well because when the technique is used for a well-defined problem, many of the difficulties in preprocessing the data can be automated (because the data structures have been seen before) and interpretation of the model is less of an issue since entire industries begin to use the technology successfully and a level of trust is created. Several ven-dors have deployed this strategy (e.g., HNC's Falcon system for credit card fraud prediction and Advanced Software Applications' Model MAX package for direct marketing).

Packaging up neural networks with expert consultants is also a viable strat-egy that avoids many of the pitfalls of using neural networks, but it can be quite expensive because it is human-intensive. One of the great promises of data mining is, after all, the automation of the predictive modeling process. These neural network-consulting teams are little different from the analytical departments many companies already have in house. Since there is not a great difference in the overall predictive accuracy of neural networks over standard statistical techniques, the main difference becomes the replacement of the sta-tistical expert with the neural network expert. Either with statistics or neural network experts, the value of putting easy-to-use tools into the hands of the business end user is still not achieved.

Neural networks rate high for accurate mod-els that provide good return on investment but rate low in terms of automation and clarity, making them more difficult to deploy across the enterprise.

Where to use Neural Networks

Neural networks are used in a wide variety of applications. They have been used in all facets of business from detecting the fraudulent use of credit cards and credit risk prediction to increasing the hit rate of targeted mailings. They also have a long history of application in other areas such as the military for the automated driving of an unmanned vehicle at 30 mph on paved roads to biological simulations such as learning the correct pronunciation of English words from written text.

Neural Networks for Clustering

Neural networks of various kinds can be used for clustering and prototype cre-ation. The Kohonen network described in this chapter is probably the most common network used for clustering and segmentation of the database. Typi-cally the networks are used in a unsupervised learning mode to create the clus-ters. The clusters are created by forcing the system to compress the data by cre-ating prototypes or by algorithms that steer the system toward creating clus-ters that compete against each other for the records that they contain, thus ensuring that the clusters overlap as little as possible.

Business Score Card for Neural Networks

Data mining measure	Description
Automation	Neural networks are often repre sented as automated data mining techniques. While they are very powerful at building predictive models, they do require significant data preprocessing and a good understanding and definition of the prediction target. Usually

normalizing predictor values between 0.0 and 1.0 and converting categorical to numeric values is required. The networks themselves also require the setting of numerous parameters that determine how the neural network is to be constructed (e.g., the number of hidden nodes). There can be significant differences in performance due to small differences in the neural network setup or the way the data is preformatted.

Clarity

The bane of neural networks is often the clarity with which the user can see and understand the results that are being presented. To some degree the complexity of the neural network models goes hand in hand with their power to create accurate predictions. This shortcoming in clarity is recognized by the neural network vendors, and they have tried to provide powerful techniques to better visualize the neural networks and to possibly

ROI

provide understandable rulers prototypes to help explain the models. Neural networks do provide powerful predictive models and theoretically are more general than other data mining and standard statistical techniques. In practice, however, the gains in accuracy over other techniques are often quite small and can be dwarfed by some of the costs because of careless construction or use of the model by nonexperts. The models can also be quite time-consuming to build.

Neural Networks for Feature Extraction

One of the important problems in all data mining is determining which predic-tors are the most relevant and the most important in building models that are most accurate at prediction. These predictors may be used by themselves or in conjunction with other predictors to form "features." A simple example of a fea-ture in problems that neural networks are working on is the feature of a verti-cal line in a computer image. The predictors, or raw input data, are just the colored pixels (picture elements) that make up the picture. Recognizing that the predictors (pixels) can be organized in such a way as to create lines, and then using the line as the input predictor, can prove to dramatically improve the accuracy of the model and decrease the time to create it.

Some features such "as lines in computer images are things that humans are already pretty good at detecting; in other problem domains it is more difficult to recognize the features. One novel way that neural networks have been used to detect features is to exploit the idea that features are a form of a compres-sion of the training database. For instance, you could describe an image to a friend by rattling off the color and intensity of each pixel on every point in the picture, or you could describe it at a higher level in terms of lines and circles or maybe even at a higher level of features such as trees and mountains. In either case your friend eventually gets all the information needed to know what the picture looks like, but certainly describing it in terms of high-level features requires much less communication of information than the "paint by numbers" approach of describing the color on each square millimeter of the image.

If we think of features in this way, as an efficient way to communicate our data, then neural networks can be used to automatically extract them. Using just five hidden nodes uses the neural network shown in Fig. 25.1 to extract features by requiring the network to learn to re-create the input data at the output nodes. Consider that if you were allowed 100 hidden nodes, then re-creating the data for the network would be rather trivial-involving simply pass-ing the input node value directly through the corresponding hidden node and on to the output node. But as there are fewer and fewer hidden nodes, that information has to be passed through the hidden layer in a more and more efficient manner since there are less hidden nodes to help pass along the information.

To accomplish this, the neural network tries to have the hidden nodes extract features from the input nodes that efficiently describe the record rep-resented at the input layer. This forced "squeezing" of the data through the narrow hidden layer forces the neural network to extract only those predictors and combinations of predictors that are best at re-creating the input record. The link weights used to create the inputs to the hidden nodes are effectively cre-ating features that are combinations of the input node values.

Applications Score Card

Table 25.1 show the applications scorecard for neural net-works with respect to how well they perform for a variety of basic underlying applications. Neural networks have been used for just about every type of supervised and unsupervised learning application. Because the underlying model is a complex mathematical equation, the generation of rules and the effi-cient detection of links in the database is a stretch for neural networks. Also, because of the large number of different words in text-based applications (high dimensionality), neural networks are seldom used for text retrieval. They do provide some sense of confidence in the degree of the prediction so that outliers who do not match the existing model can be detected.

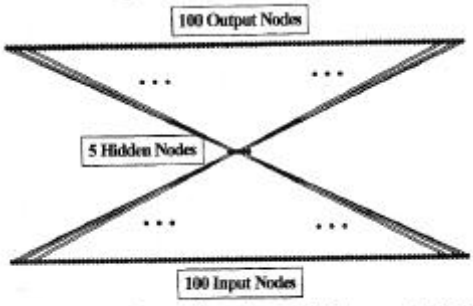


Fig: 25.1 Neural networks can be used for data compression and feature extraction.

TABLE 25.1 Applications Score Card for Neural Networks

Problem type	Description
Clusters	Although neural networks were originally conceived to mimic neural function in the brain and then used for a variety of prediction and classification tasks, they have also been found useful for clustering. Almost coincidentally, the self-organizing nature of the brain when mimicked in an artificial neural network results in the clustering of records from a database.
Links	Neural networks can be used to determine links and patterns in the database, although to be efficient, neural architectures very different from the standard single hidden layer need to be used. To do this efficiently, a network would generally have as many input nodes as output nodes and each node would represent an individual object that could be linked together.
Outliers	The general structure of the neural network is not designed for outlier detection in the way that nearest-neighbor classification techniques are, but they can be used for outlier detection by simply building the predictive model and seeing which record's actual values correspond to the predicted values. Any large disparity between the actual and predicted could well be an outlier.
Rules	Neural networks do not generate rules either for classification or explanation. Some new techniques are now being developed that would create rules after the fact to try to help explain the neural network, but these are additions to the basic neural network architec ture.
Sequences	Because of their strengths in performing predictions for numeric prediction values and regression in general, neural networks are often used to do sequence prediction (like predicting the stock market). Generally a significant amount of preprocessing of the data needs to be performed to convert the time series data into something useful to the neural network.
Text	Because of the large number of possible input nodes (number of different words used in a given language), neural networks are seldom used for text retrieval. They have been used at a higher level to create a network that learns the relationships between documents.

The General Idea

What does a neural network look like?

A neural network is loosely based on concepts of how the human brain is orga-nized and how it learns. There are two main structures of consequence in the neural network:

1. The node-which loosely corresponds to the neuron in the human brain
2. The link-which loosely corresponds to the connections between neurons

(Axons, dendrites, and synapses) in the human brain

Figure 25.2 is a drawing of a simple neural network. The round circles repre-sent the nodes, and the connecting lines represent the links. The neural net-work functions by accepting predictor values at the left and performing calculations on those values to produce new values in the node at the far right. The value at this node represents the prediction from the neural network model. In this case the network takes in values for predictors for age and income and predicts whether the person will default on a bank loan.

How does a neural net make a prediction?

In order to make a prediction, the neural network accepts the values for the predictors on what are called the input nodes. These become the values for those nodes; these values are then multiplied by values that are stored in the links (sometimes called weights and in some ways similar to the weights that are applied to predictors in the nearest-neighbor method). These values are then added together at the node at the far right (the output node), a special threshold function is applied, and the resulting number is the prediction. In this case, if the resulting number is 0, the record is considered to be a good

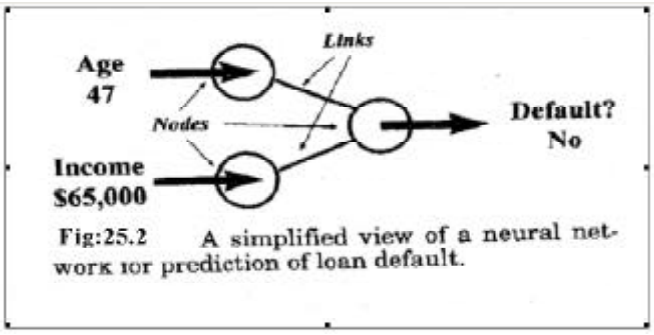


Fig:25.2 A simplified view of a neural network for prediction of loan default.

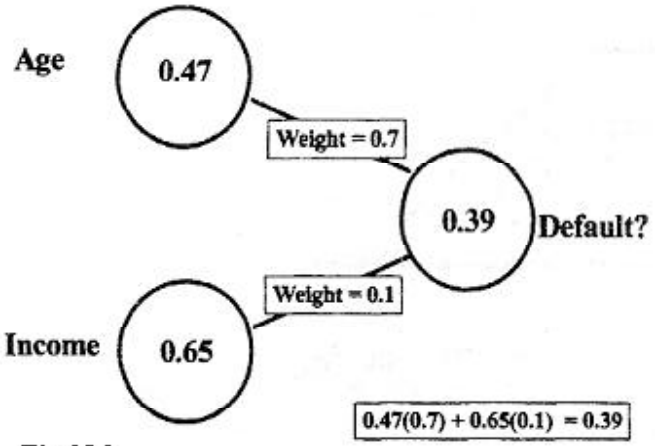


Fig:25.3 The normalized input values are multiplied by the link weights and added together at the output.

Figure 25.3, The normalized input values are multiplied by the link weights and added together at the output.

Credit risk (no default); if the number is 1; the record is considered to be a bad credit risk (likely default).

A simplified version of the calculations depicted in Fig. 25.2 might look like Fig. 25.3. Here the value of age of 47 is normalized to fall between 0.0 and 1.0 and has the value 0.47, and the income is normalized to the value 0.65. This simplified neural network makes the prediction of no default for a 47-year-old making $65,000. The links are weighted at 0.7 and 0.1, and the resulting value after multiplying the node values by the link weights is 0.39. The network has been trained to learn that an output value of 1.0 indicates default and that 0.0 indicate no default. The output value calculated here (0.39) is closer to 0.0 than to 1.0, so the record is assigned a no default prediction.

How is the neural network model created?

The neural network model is created by presenting it with many examples of the predictor values from records in the training set (in this example age and income are used) and the prediction value from those same records. By comparing the correct answer obtained from the training record and the pre-dicted answer from the neural network, it is possible to slowly change the behavior of the neural network by changing the values of the link weights. In some ways this is like having a grade school teacher ask questions of her student (a.k.a. the neural network) and if the answer is wrong, to verbally correct the student. The greater the error, the harsher the verbal correction; thus large errors are given greater attention at correction than are small errors.

For the actual neural network, it is the weights of the links that actually con-trol the prediction value for a given record. Thus the particular model that is being found by the neural network is, in fact, fully specified by the weights and the architectural structure of the network. For this reason it is the link weights that are modified each time an error is made.

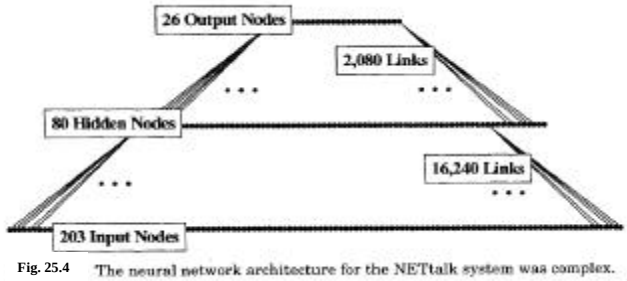


Fig. 25.4 The neural network architecture for the NETtalk system was complex.

How complex can the neural network model become?

The models shown in Figs. 25.2 and 25.3 have been designed to be as simple as possible in order to make them understand-able. In practice no networks are as simple as these. Figure 25.4 shows a network with many more links and many more nodes. This was the architecture of a neural network system called NET talk, which learned how to pronounce written English words. This draw-ing shows only some of the nodes and links. Each node in the network was con-nected to every node in the level above it and below it, resulting in 18,629 link weights that needed to be learned in the network. Note that this network also now has a row of nodes in between the input nodes and the output nodes. These are called "hidden nodes" or the "hidden layer" because the values of these nodes are not visible to the end user in the way that the output nodes are (which contain the prediction) and the input nodes (which just contain the pre-dictor values). There are even more complex neural network architectures that have more than one hidden layer. In practice one hidden layer seems to suffice, however.

Notes

LESSON 26
NEURAL NETWORKS

Structure

- Objective
- What Is A Neural Network?
- Hidden nodes are like trusted advisors to the output nodes
- Design decisions in architecting a neural network
- How does the neural network resemble the human brain?
- Applications Of -Neural Networks
- Data Mining Using NN: A Case Study

Objective

The main objective of this lesson is to introduce you the principles of neural computing.

What is a Neural Network?

Neural networks are a different paradigm for computing, which draws its inspiration from neuroscience. The human brain consists of a network of neurons, each of which is made up of a number of nerve fibres called dendrites, connected to the cell body where the cell nucleus is located. The axon is a long, single fibre that originates from the cell body and branches near its end into a number of strands. At the ends of these strands are the transmitting ends of the synapses that connect to other biological neurons through the receiving ends of the synapses found on the dendrites as well as the cell body of biological neurons. A single axon typically makes thousands of synapses with other neurons. The transmission process is a complex chemical process, which effectively increases or decreases the electrical potential within the cell body of the receiving neuron. When this electrical potential reaches a threshold value (action potential), it enters its excitatory state and is said to *fire*. It is the connectivity of the neuron that gives these simple 'devices' their real power.

Artificial neurons (or processing elements, PE) are highly simplified models of biological neurons. As in biological neurons, an artificial neuron has a number of inputs, a cell body (most often consisting of the summing node and the transfer function), and an output, which can be connected to a number of other artificial neurons. Artificial neural networks are densely interconnected networks of PEs, together with a rule (learning rule) to adjust the strength of the connections between the units in response to externally supplied data.

The evolution of neural networks as a new computational model originates from the pioneering work of McCulloch and Pitts in 1943. They suggested a simple model of a neuron that connoted the weighted sum of the inputs to the neuron and an output of 1 or 0, according to whether the sum was over a threshold value or not. A 0 output would correspond to the inhibitory state of the neuron, while a 1 output would correspond to the excitatory state of the neuron. Consider a simple example illustrated below.

The network has 2 binary inputs, I_0 and I_1 and one binary output *Y*. W_0 and W_1 are the connection strengths of input 1 and input 2, respectively. Thus, the total input received at the processing unit is given by

$W_0I_0 + W_1I_1 - W_b$,

where

W_b is the threshold (in another notational convention, it is viewed as the bias). The output Y takes on the value 1, if $W_0I_0 + W_1I_1 - W_b$, > 0 and, otherwise, it is 0 if

W_0I_0 + Will - Wb £, O.

But the model, known as perceptron, was far from a true model of a biological neuron as, for a start, the biological neuron's output is a continuous function rather than a step function. This model also has a limited computational capability as it represents only a linear-separation. For two classes of inputs, which are linearly separable, we can find the weights such that the network returns 1 as output for one class and 0 for another class.

There have been many improvements on this simple model and many architectures have been presented in recently. As a first step, the threshold function or the step function is replaced by other more general, continuous functions called *activation.*

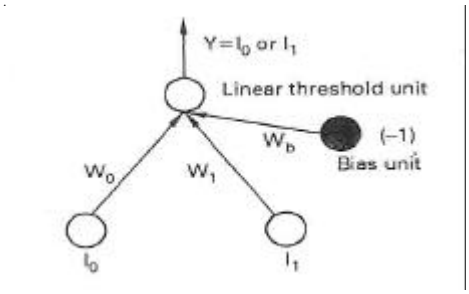


Figure 26.1 a simple perceptron

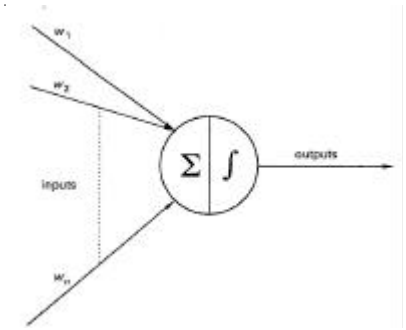


Figure 26.2 A Typical Artificial Neuron with Activation Function

functions. Figure 26.2 illustrates the structure of a node (PE) with an activation function. For this particular node, n weighted inputs (denoted W;, i = 1,..., n) are combined via a combination function that often consists of a simple summation. A transfer function then calculates a corresponding value, the result yielding a single output, usually between 0 and 1. Together, the combination function and the transfer function make up the activation function of the node.

Three common transfer functions are the sigmoid, linear and hyperbolic functions. The sigmoid function (also known as the logistic function) is very widely used and it produces values between 0 and 1 for any input from the combination function. The sigmoid function is given by (the subscript *n* identifies a PE):

$$Y = \frac{1}{1 + \exp(-\varsigma(\sum_i W_{in} X_{in}))}$$

Note that the function is strictly positive and defined for all values of the input. When plotted, the graph takes on a sigmoid shape, with an inflection point at (0, .5) in the Cartesian plane. The graph (Figure 26.3) plots the different values of S as the input varies from -10 to 10.

Individual nodes are linked together in different ways to create neural networks. In a feed-forward network, the connections between layers are unidirectional from input to output. We discuss below two different architectures of the feed-forward network, Multi-Layer Perceptron and Radial-Basis Function.

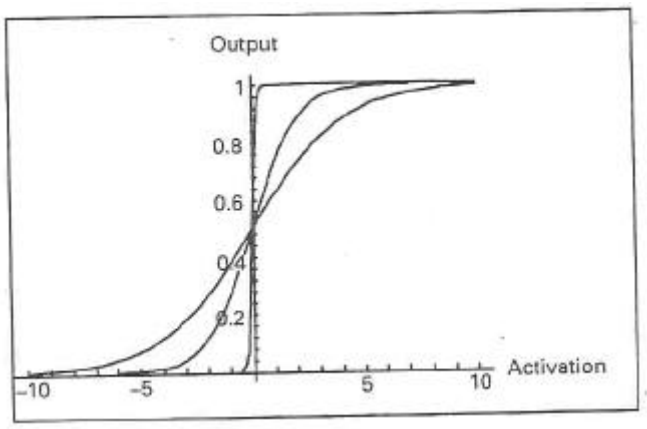


Figure 26.3 Sigmoid Functions

Hidden nodes are like trusted advisors to the output nodes

The meanings of the input nodes and the output nodes are usually pretty well understood-and are usually defined by the end user with respect to the par-ticular problem to be solved and the nature and structure of the database. The hidden nodes, however, do not have a predefined meaning and are determined by the neural network as it trains. This poses two problems:

1. It is difficult to trust the prediction of the neural network if the meaning of these nodes is not well understood.
2. Since the prediction is made at the output layer and the difference between the prediction and the actual value is calculated there, how is this error cor-rection fed back through the hidden layers to modify the link weights. Those connect them?

The meaning of these hidden nodes is not necessarily well understood but sometimes after the fact they can be studied to see when they are active (have larger numeric values) and when they are not and derive some meaning from them. In some of the earlier neural networks were used to learn the family trees of two different families-one was Italian and one was English and the network was trained to take as inputs either two people and return their rela-tionship (father, aunt, sister, etc.) or given one person and a relationship to return the. Other person. After training, the units in one of the hidden layers were exam med to see If there was any discernible explanation as to their role in the prediction. Several of the nodes did seem to have specific and under-standable purposes. One, for instance, seemed to break up the input records (people) into either Italian or English descent, another unit encoded for which generation a person belonged to, and another encoded for the branch of the family that the person came from. The neutral network to aid in predictor automatically extracted each of these nodes.

Any interpretation of the meaning of the hidden nodes needs to be done after the fact-after the network has been trained, and it is not always possible to determine a logical description for the particular function for the hidden nodes. The second problem with the hidden nodes is perhaps more serious (if it hadn't been solved, neural networks wouldn't work). Luckily it has been solved.

The learning procedure for the neural network has been defined to work for the weights in the links connecting the hidden layer. A good analogy of how this works would be a military operation in some war where there are many layers of command with a general ultimately responsible for making the decisions on where to advance and where to retreat. Sev-eral lieutenant generals probably advise the general, and several major generals, in turn, probably advise each lieutenant general. This hierarchy continues downward through colonels and privates at the bottom of the hierarchy.

This is not too far from the structure of a neural network with several hid-den layers and one output node. You can think of the inputs coming from the hidden nodes as advice. The link weight corresponds to the trust that generals have in their advisors. Some trusted advisors have very high weights and some advisors may not be trusted and, in fact, have negative weights. The other part of the advice from the advisors has to do with how competent the particular

Advisor is for a given situation. The general may have a trusted advisor, but if that advisor has no expertise in aerial invasion and the situation in question involves the air force, this advisor may be very well trusted but the advisor per-sonally may not have any strong opinion one way or another.

In this analogy the link weight of a neural network to an output unit is like the trust or confidence that commanders have in their advisors and the actual node value represents how strong an opinion this particular advisor has about this particular situation. To make a decision, the general considers how trustworthy and valuable the advice is and how knowledgeable and confident all the advisors are in making their suggestions; then, taking all this into account, the general makes the decision to advance or retreat.

In the same way, the output node will make a decision (a prediction) by tak-ing into account all the input from its advisors (the nodes connected to it). In the case of the neural network multiplying the link weight by the output value of the node and summing these values across all nodes reach this decision. If the prediction is incorrect, the nodes that had the most influence on making the decision have their weights modified so that the wrong prediction is less likely to be made the next time.

This learning in the neural network is very similar to what happens when the general makes the wrong decision. The confidence that the general has in all those advisors who gave the wrong recommendation is decreased-and all the more so for those advisors who were very confident and vocal in they're recommendations. On the other hand, any advisors who were making the correct recommendation but whose input was not taken as seriously would be taken more seriously the next time. Likewise, any advisors who were reprimanded for giving the wrong advice to the general would then go back to their own advisors and determine which of them should have been trusted less and whom should have been listened to more closely in rendering the advice or recommendation to the general. The changes generals should make in listening to their advisors to avoid the same bad decision in the future are shown in Table 26.1.

This feedback can continue in this way down throughout the organizational each level, giving increased emphasis to those advisors who had advised cor-rectly and decreased emphasis to those who had advised incorrectly. In this way the entire organization becomes better and better at supporting the general in making the correct decision more of the time.

A very similar method of training takes place in the neural network. It is called back propagation and refers to the propagation of the error backward from the output nodes (where the error is easy to determine as the difference between the actual prediction value from the training database and the pre-

TABLE 26.1 Neural Network Nodes.

General's trust	Advisor's Recommendation	Advisor's Confidence	Change to General's trust
High	Good	High	Great increase
High	Good	Low	Increase
High	Bad	High	Great decrease
High	Bad	Low	Decrease
Low	Good	High	Increase
Low	Good	Low	Small increase
Low	Bad	High	Decrease
Low	Bad	Low	Small decrease

*The link weights in a neural network are analogous to the confidence that gen-erals might have. In there trusted advisors.

diction from the neural network) through the hidden layers and to the input layers. At each level the link weights between the layers are updated so as to decrease the chance of making the same mistake again.

Design decisions in architecting a neural network

Neural networks are often touted as self-learning automated techniques that simplify the analysis process. The truth is that there still are many decisions to be made by the end user in designing the neural network even before training begins. If these decisions are not made wisely, the neural network will likely come up with a sub optimal model. Some of the decisions that need to be made include

- How will predictor values be transformed for the input nodes? Will normal-ization be sufficient? How will categoricals be entered?
- How will the output of the neural network be interpreted?
- How many hidden layers will there be?
- How will the nodes be connected? Will every node be connected to every other node, or will nodes just be connected between layers?
- How many nodes will there be in the hidden layer? (This can have an impor-tant influence on whether the predictive model is over fit to the training database.)
- How long should the network be trained for? (This also has an impact on whether the model over fits the data.)

Depending on the tool that is being used, these decisions may be (explicit, where the user must set some parameter value, or they may be decided for the user because the particular neural

network is being used for a specific type of prob-lem (like fraud detection).

Different types of Neural Networks

There are literally hundreds of variations on the back propagation feed forward neural networks that have been briefly described here. One involves changing the architecture of the neural network to include recurrent connections where the output from the output layer is connected back as input into the hidden layer. These recurrent nets are sometimes used for sequence prediction, in which the previous outputs from the network need to be stored someplace and then fed back into the network to provide context for the current prediction. Recurrent networks have also been used for decreasing the amount of time that it takes to train the neural network. Another twist on the neural net theme is to change the way that the network learns. Back propagation effec-tively utilizes a search technique called gradient descent to search for possible improvement in the link weights to reduce the error. There ever, many other ways of doing search in a high-dimensional space effectively utilizes a search technique called gradient descent to search for the best possible improvement in the link weights to reduce the error. There are, how-ever, many other ways of doing search in a high-dimensional space (each link weight corresponds to a dimension), including Newton's methods and conju-gate gradient as well as simulating the physics of cooling metals in a process called simulated annealing or in simulating the search process that goes on in biological evolution and using genetic algorithms to optimize the weights of the

Neural networks. It has even been suggested that creating a large number of neural networks with randomly weighted links and picking the one with the lowest error rate would be the best learning procedure.

Despite all these choices, the back propagation learning procedure is the most commonly used. It is well understood, is relatively simple, and seems to work in a large number of problem domains. There are, however, two other neural network architectures that are used relatively often. Kohonen feature maps are often used for unsupervised learning and clustering, and radial-basis-function networks are used for supervised learning and in some ways represent a hybrid between nearest-neighbor and neural network classifications.

Kohonen feature Maps

Kohonen feature maps were developed in the 1970s and were created late certain human brain functions. Today they are used mostly to unsupervised learning and clustering.

Kohonen networks are feed forward neural networks generally with den layer. The networks contain only an input layer and an output h the nodes in the output layer compete among themselves to display the strongest activation to a given record, what is sometimes called a "win all" strategy.

Behaviors 01 the real neurons were 'taken into effect-namely, that physical locality of the neurons seems to play an important role in the behavior learning of neurons. The specific features of real neurons were:

- Nearby neurons seem to compound the activation of each other.
- Distant neurons seemed to inhibit each other
- The tasks assigned to other neurons.

Much of this early research came from the desire to simulate the way that vision worked in the brain. For instance, some of the early physiological showed that surgically rotating a section of a frog's eye ball so that it was upside down would result in the frog jumping up for food that was al below the frog's body. This led to the belief that the neurons had certainty enable roles that were dependent on the physical location of the neuron. Koho-nen networks were developed to accommodate these physiological features by a very simple learning algorithm:

1. Layout the output nodes of the network on a two-dimensional grid with no hidden layer.
2. Fully connect the input nodes to the output nodes.
3. Connect the output nodes so that nearby nodes would strengthen each other and distant nodes would weaken each other.
4. Start with random weights on the links.
5. Train by determining which output node responded most strongly to the Current record being input.
6. Change the weights to that highest-responding node to enable it to respond even more strongly in the future. This is also known as Hebbian Learning.
7. Normalize the link weights so that they add up to some constant amount; thus, increasing one weight decreases some other.
8. Continue training until some form of global organization is formed on the two-dimensional output grid (where there are clear winning nodes for each input and, in general, local neighborhoods of nodes are activated).

When these networks were run, in order to simulate the real-world visual system it became obvious that the organization that was automatically being con-structed on the data was also very useful for '3egmenting and clustering the train-ing database: Each output node represented a cluster and nearby clusters were nearby in the two-dimensional output layer. Each record in the database would fall into one and only one cluster (the most active output node), but the other clusters in which it might also fit would be shown and likely to be next to the best matching cluster. Figure 26.4 shows the general form of a Kohonen network.

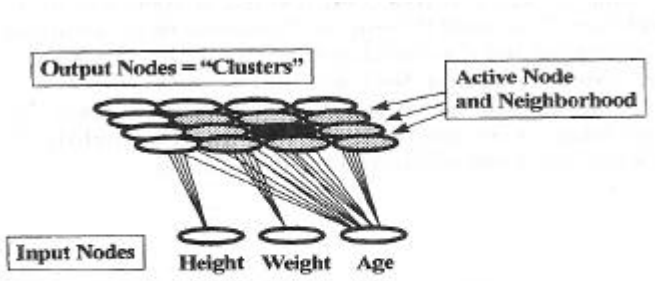


Fig:26.4 The Kohonen network arranges the output nodes in a two-dimensional grid to simulate the positive reinforcement of neighboring neurons in brains.

How does the neural network resemble the human brain?

Since the inception of the idea of neural networks, the ultimate goal for these techniques has been to have them recreated human thought and learning. This has once again proved to be a difficult task-despite the power of these new techniques and the similarities of their architecture to that of the human brain. Many of the things that people take for granted are difficult for neural networks-such as avoiding over fitting and working with real-world data without a lot of preprocessing required. There have also been some exciting successes.

The human brain is still much more powerful

With successes like NET talk and ALVINN and some of the commercial suc-cesses of neural networks for fraud prediction and targeted marketing, it 1.1; tempting to claim that neural networks are making progress toward "think-ing," but it is difficult to judge just how close we are. Some real facts that we can look at are to contrast the human brain as a computer to the neural net-work implemented on the computer.

Today it would not be possible to create an artificial neural network that even had as many neurons in it as the human brain, let alone all the process-ing required for the complex calculations that go on inside the brain. The cur-rent estimates are that there are 100 billion neurons in the average person (roughly 20 times the number of people on earth). Each single neuron can receive input from up to as many as 100,000 synapses or connections to other neurons, and overall there are 10,000 trillion synapses.

To get an idea of the size of these numbers, consider that if you had a 10-Tbyte data warehouse (the largest warehouse in existence today), and you were able to store all of the complex-ity of a synapse in only a single byte of data within that warehouse, you would still require 1000 of these warehouses just to store the synapse information. This doesn't include the data required for the neu-rons or all the computer processing power required to actually run this simu-lated brain. The bottom line is that we're still a factor of 1000 away from even storing the required data to simulate a brain. If storage densities on disks and other media keep increasing and the prices continue to decrease, this problem may well be solved. Nonetheless, there is much more work to be done in under-standing how real brains function.

Applications of Neural Networks

These days, neural networks are used in a very large number of applications. We list here some of those relevant to our study. Neural networks are being used in

- **Investment analysis:**

  To predict the movement of stocks, currencies etc., from previous data. There, they are replacing earlier simpler linear models.

- **Monitoring:**

  Networks have been used to monitor the state of aircraft engines. By monitoring vibration levels and sound, an early warning of engine problems can be given.

- **Marketing:**

  Neural networks have been used to improve marketing mailshots. One technique is to run a test mailshot, and look at the pattern of returns from this. The idea is to find a predictive mapping from the data known about the clients to how they have responded. This mapping is then used to direct further mailshots.

Data Mining Using NN: A Case Study

In this section, I will outline a case study to you to illustrate the potential application of NN for Data mining. This case study is taken from [Shalvi, 1996].

Knowledge Extraction Through Data Mining

Kohonen, self-organizing maps (SOMs) are used to cluster a specific medical data set containing information about the patients' drugs, topographies (body locations) and morphologies (physiological abnormalities); these categories can be identified as the three input subspaces. Data mining techniques are used to collapse the subspaces into a form suitable for network classification. The goal is to acquire medical knowledge which may lead to tool formation, automation and to assist medical decisions regarding population. The data is organized as three hierarchical trees, identified as Drugs, Topography and Morphology. The most significant portion of the morphology tree is displayed in Figure 26.5. Before presenting the data to the neural network, certain preprocessing should be done. Standard techniques can be employed to clean erroneous and redundant data.

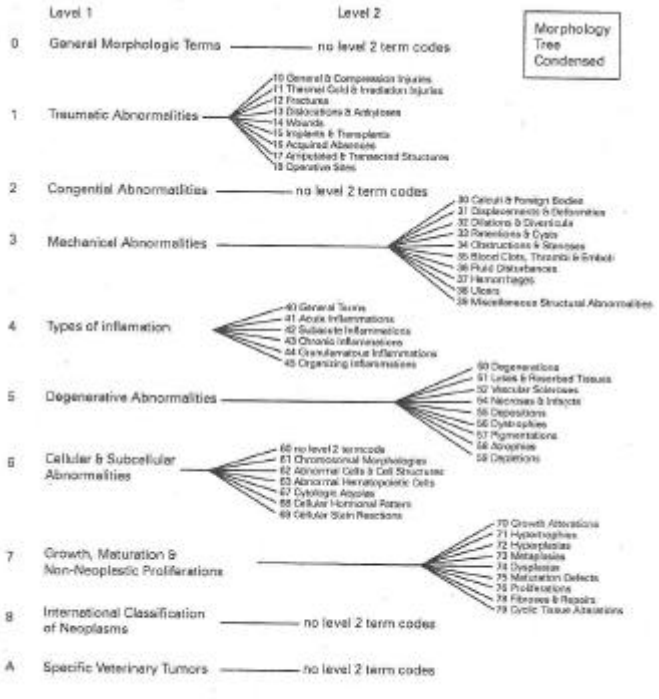


Fig:26.5 Morphology Tree

The data is processed at the root level of each tree-fourteen root level drugs, sixteen root level topographies and ten root level morphologies. By constraining all the data to the root level, the degree of differentiation has been greatly reduced from thousands to just 40. Each tuple is converted into a bipolar format. Thus, each tuple is a 40-dimensional bipolar array-a value of either 1 or -1 depending on whether any data existed for the leaves of that root node.

The Kohonen self-organizing map (SOM) was chosen to organize the data, in order to make use of a spatially ordered, 2-dimensional map of arbitrary granularity. An *n* x *n* SOM was implemented for several values *ofn.* We describe below only the case with *n*= 1 O. The input layer consists of 40 input nodes; the training set consists of 2081 tuples; and the training period was of 30 epochs. The learning coefficient S is initialized to 0.06. After approximately 7Y2 epochs, S is halved to 0.03. After another 7Y2 epochs, it is halved again to 0.015. For the final set of7Y2 epochs, it is halved again to become 0.0075.

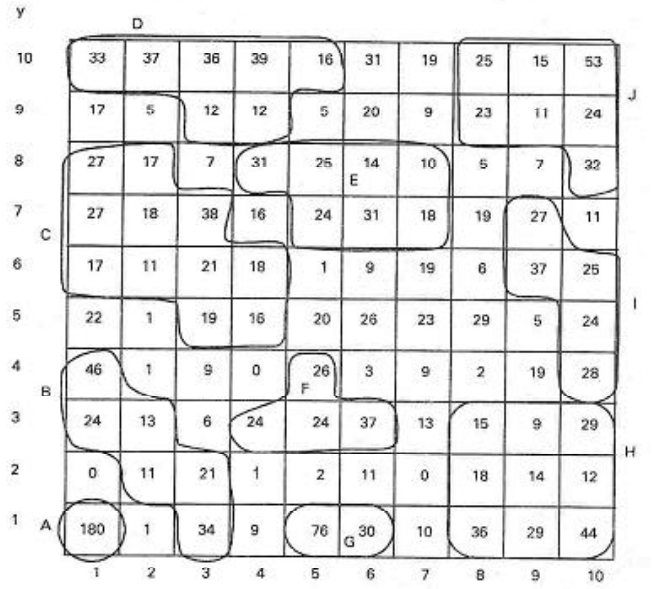


Fig:26.6 Population clusters

After the network is trained it is used for one final pass through the input data set, in which the weights are not adjusted. This provides the final classification of each input data tuple into a single node in the l0 x 10 grid. The output is taken from the coordinate layer as an *(x, y)* pair. The output of the SOM is a population distribution of tuples with spatial significance (Figure 26.6). This grid displays the number of tuples that were classified into each Kohorien layer node (square) during testing; for example, Square (1,1) contains 180 tuples.

Upon examination of the raw data within these clusters, one finds similarities between the tuples which are indicative of medical relationships or dependencies. Numerous hypotheses can be made regarding these relationships, many of which were not known *a priori. The* SOM groups together tuples in each square according to their similarity. The only level at which the SOM can detect similarities between tuples is at the root level of each of the three subspace trees, since this is the level of differentiation presented to the SOM's input. For example, every one of the tuples in square (1,1) contains root level data only for Drug 6, Topography 6 and Morphology 5. The tuple at square (2,1) contains these three root level nodes as well as Drug 7, a difference slight enough for the network to distinguish the tuple by classifying it one square away from (1,1). All 34 tuples in square (3,1) contain Drug 6, Topography D and Morphology 5; but only 29 of the 34 tuples contain Topography 6. Clearly, the difference between square (3,1) and square (1,1) is greater than the difference between square (2,1) and square (1,1).

Exercises

1. Describe the principle of neural computing and discuss its suitability to data mining.
2. Discuss various application areas of Neural Networks.
3. Explain in brief different types of neural networks.
4. "Hidden nodes are like trusted advisors to the output nodes". Discuss.
5. Explain in brief Kohonen feature maps.

Notes

LESSON 27
ASSOCIATION RULES AND GENETIC ALGORITHM

Structure

- Objective
- Association Rules
- Basic Algorithms for Finding Association Rules
- Association Rules among Hierarchies
- Negative Associations
- Additional Considerations for Association Rules
- Genetic Algorithms (GA)
- Crossover
- Mutation
- Problem-Dependent Parameters
- Encoding
- The Evaluation Step
- Data Mining Using GA

Objective

The objective of this lesson is to introduce you with data mining techniques like association rules and genetic algorithm.

Association Rules

One of the major technologies in data mining involves the discovery of association rules. The database is regarded as a collection of transactions, each involving. Set of items. A common example is that of market-basket data. Here the market basket corresponds to what a consumer buys in a supermarket during one visit. Consider four such transactions in a random sample:

Transaction-id	Time	Items- Brought
101	6:35	milk, bread, juice
792	7:38	milk, juice
1130	8:05	milk, eggs
1735	8:40	bread, cookies, coffee

An **association rule** is of the form X=>Y, where X = {x_1, x_2... x_n}, and Y = {y1, y2...y_m} are sets, of items, with x_i and Y_j being distinct items for all i and j. This association states that if a customer buys X, he or she is also likely to buy Y. In general, any association rule has the form LHS (left-hand side) Þ RHS (right-hand side), where LHS and RHS are sets of items. Association rules should supply both support and confidence.

The support for the rule LHSÞRHS is the percentage of transactions that hold all of the items in the union, the set LHS U RHS. If the support is low, it implies that there is no overwhelming evidence that items in LHS U RHS occur together, because the union happens in only a small fraction of transactions. The rule MilkÞJuice has 50% support, while Bread Þ Juice has only 25% support. Another term for support is preva-lence of the rule.

To compute confidence we consider all transactions that include items in LHS. The confidence for the association rule LHS ÞRHS is the percentage (fraction) of such transactions that also include RHS. Another term for confidence is strength of the rule.

For MilkÞJuice, the confidence is 66.7% (meaning that, of three transactions in which milk occurs, two contain juice) and breadÞjuice has 50% confidence (meaning that one of two transactions containing bread also contains juice.)

As we can see, support and confidence do not necessarily go hand in hand. The goal of mining association rules, then, is to generate all possible rules that exceed some mini-mum user-specified support and confidence thresholds. The problem is thus decomposed into two subproblems:

1. Generate all item sets that have a support that exceeds the threshold. These sets of items are called large itemsets. Note that large here means large support.
2. For each large item set, all the rules that have a minimum confidence are gener-ated as follows: for a large itemset X and Y C X, let Z = X – Y; then if support (X) /support (Z) Þ minimum confidence, the rule Z ÞY (Le., X - Y Þ Y) is a valid rule. [Note: In the previous sentence, Y C X reads, "Y is a subset of X."]

Generating rules by using all large itemsets and their supports is relatively straightforward. However, discovering all large itemsets together with the value for their support is a major problem if the cardinality of the set of items is very high. A typical supermarket has thousands of items. The number of distinct itemsets is 2^m, where m is the number c£ items, and counting support for all possible itemsets becomes very computation-intensive.

To reduce the combinatorial search space, algorithms for finding association rule have the following properties:

- A subset of a large itemset must also be large (i.e., each subset of a large itemset: L exceeds the minimum required support).
- Conversely, an extension of a small itemset is also small (implying that it does 00: have enough support).

The second property helps in discarding an itemset from further consideration J extension, if it is found to be small.

Basic Algorithms for Finding Association Rules

The current algorithms that find large itemsets are designed to work as follows:

1. Test the support for itemsets of length 1, called l-itemsets, by scanning the database. Discard those that do not meet minimum required support.
2. Extend the large l-itemsets into 2-itemsets by appending one item each time, to generate all candidate itemsets of length two. Test the support for all candidate itemsets by scanning the database and eliminate those 2-itemsets that do not meet the minimum support.
3. Repeat the above steps at step k; the previously found (k - 1) itemsets are extended into k-itemsets and tested for minimum support.

The process is repeated until no large item sets can be found. However, the naive ver-sion of this algorithm is a combinato-rial nightmare. Several algorithms have been pro-posed to mine the association rules. They vary mainly in terms of how the candidate itemsets are generated, and how the supports for the candidate item sets are counted.

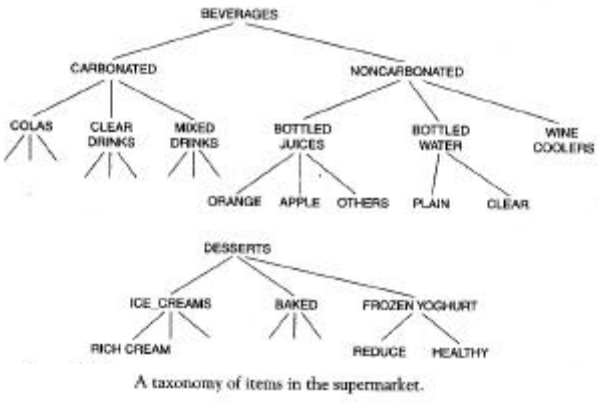
Some algorithms use such data structures as bitmaps and has trees to keep informa-tion about item sets. Several algorithms have been proposed that use multiple scans of the database because the potential number of item sets, 2^m, can be too large to set up counters during a single scan. We have proposed an algorithm called the **Partition algorithm** Summarized below.

If we are given a database with a small number of potential large item sets, say, a few thousand, then the support for all of them can be tested in one scan by using a partition-ing tech-nique. Partitioning divides the database into no overlapping partitions; these are individually considered as separate data-bases and all large item sets for that partition are generated in one pass. At the end of pass one, we thus generate a list of large item sets from each partition. When these lists are merged, they contain some false positives. That is, some of the itemsets that are large in one partition may not qualify in several other par-titions and hence may not exceed the minimum support when the original database is considered. Note that there are no false negatives, i.e., no large itemsets will be missed. The union of all large item sets identified in pass one is input to pass two as the candidate itemsets, and their actual support is measured for the entire database. At the end of phase two, all actual large itemsets are identified. Partitions are chosen in such a way that each partition can be accommodated in main memory and a partition is read only once in each phase. The Partition algo-rithm lends itself to parallel implementation, for efficiency.

Fur-ther improvements to this algorithm have been suggested.

Association Rules among Hierarchies

There are certain types of associations that are particularly interesting for a special reason. These associations occur among hierarchies of items. Typically, it is possible to divide items among disjoint hierarchies based on the nature of the domain. For example, foods in a supermarket, items in a department store, or articles in a sports shop can be categorized into classes and subclasses that give rise to hierarchies. Which shows the taxonomy of items in a supermarket. The figure shows two hierarchies—beverages and desserts, respectively. The entire groups may not produce associations of the form beverages p Þ desserts, or dessertsÞbeverages.



A taxonomy of items in the supermarket.

However, associations of the type Healthy-brand frozen yogurt Þ bottled water, or Richcream-brand ice cream Þ wine cooler may produce enough confidence and sup-port to be valid association rules of interest.

Therefore, if the application area has a natural classification of the itemsets into hier-archies, discovering associations within the hierarchies is of no particular interest. The ones of specific interest are associations across hierarchies. They may occur among item groupings at-different levels.

Negative Associations

The problem of discovering a negative association is harder than that of discovering a positive association. A negative association is of the following type: "60% of customers who buy potato chips do not buy bottled water." (Here, the 60% refers to the confidence for the negative association rule.) In a database with 10,000 items, there are 2^{10000} possible combina-tions of items, a majority of which do not appear even once in the database. If the absence of a certain item combination is taken to mean a negative association, then we potentially have millions and millions of negative ass0ci-ation rules with RHSs that are of no interest at all. The problem, then, is to find only interesting negative rules. In general, we are interested in cases in which two specific sea of items appear very rarely in the same transaction.

This poses two problems: For a total item inventory of 10,000 items, the probability of any two being bough together is (1/ 10,000) * (1/10,000) = 10^{-8}. If we find the actual support for these two occurring together to be zero, that does not' represent a significant departure from expectation and hence is not an interesting (negative) association.

The other problem is more serious. We are looking for item combinations with very low support, and there are millions and millions with low or even zero support. For example, a data set of 10 million transactions has most of the 2.5 billion pairwise combinations of 10,000 items missing. This would generate billions of useless rules.

Therefore, to make negative association rules interesting we must use prior knowledge about the itemsets. One approach is to use hierarchies. Suppose we use the hierar-chies of soft drinks and chips shown in Fig 27.1. A strong positive associa-tion has been shown between soft drinks and chips. If we find a large support for the fact that when customers buy pays chips

they predominantly buy Topsy and not Joke and not Wakeup that would be interesting. This is so because we would Normally expect that if there is a strong association between Days and Topsy, there should also be such a strong association between Days and Joke or Days and Wakeup.

In the frozen yogurt and bottled water groupings in Fig 27.1, suppose the Reduce versus Healthy brand division is 80-20 and the Plain and Clear brands division 60-40 among respective categories. This would give a joint probability of Reduce frozen yogurt.

Being purchased with Plain bottled water as 48% among the transactions containing a frozen yogurt and bottled water. If this support, however, is found to be only 20%, that would indicate a significant negative association among Reduce yogurt and Plain bottled water; again, that would be interesting.

The problem of finding negative association is important in the above situations given the domain knowledge in the form of item generalization hierarchies (that is, the beverage given and desserts hierarchies shown in Fig 27.1), the existing positive associations (such as between the frozen-yogurt and bottled water group's), and the distribution of items (such as the name brands within related groups). Recent work has been reported by the database group at Georgia Tech in this context (see bibliographic notes). The Scope of dis-covery of negative associations is limited in terms of knowing the item hierarchies and dis-tributions. Exponential growth of negative associations remains a challenge.

Additional Considerations for Association Rules

For very large datasets, one way to improve efficiency is by sampling. If a representative sample can be found that truly repre-sents the properties of the original data, then most of the rules can be found. The problem then reduces to one of devising a proper sampling procedure. This process has the potential danger of discovering some false positives (large item sets that are not truly large) as well as hawing false negatives by missing some large itemsets and corresponding association rules.

Fig: 27.1 Simple hierarchies of soft drinks and chips

Mining association rules in real-life databases is further complicated by the following factors.

The cardinality of itemsets in most situations is extremely large, and the volume of transactions is very high as well. Some operational databases in retailing and commu-nication industries collect tens of millions of transactions per day.
Transactions show variability in such factors as geographic location and seasons, making sampling difficult. Item classifications exist along multiple dimensions. Hence, driving the discovery process with domain knowledge, particularly for negative rules, is extremely difficult. Quality of data is variable; significant problems exist with missing, erroneous, con-flicting, as well as redundant data in many industries.

Association rules can be generalized for data mining purposes although the notion of itemsets was used above to discover association rules, almost any data in the standard relational form with a number of attributes can be used. For example, consider blood-test data with attributes like hemoglobin, red blood cell count, white blood cell count, blood--sugar, urea, age of patient, and so on. Each of the attributes can be divided into ranges, and the presence of an attribute with a value can be considered equivalent to an item. Thus, if the hemoglobin attribute is divided into ranges 0-5, 6-7, 8-9, 10-12, 13-14, and above 14, then we can. Consider them as items HI, H2... H7. Then a specific hemoglo-bin value for a patient corresponds to one of these seven items being present. The mutual exclusion among these hemoglobin items can be used to some advantage in the scanning for large itemsets. This way of dividing variable values into ranges allows us to apply the association-rule machinery to any database for mining purposes. The ranges have to be determined from domain knowledge such as the relative importance of each of the hemo-globin values.

Genetic Algorithm

Genetic algorithms (GA), first proposed by Holland in 1975, are a class of computational models that mimic natural evolution to solve problems in a wide variety of domains. Genetic algorithms are particularly suitable for solving complex optimization problems and for applications that require adaptive problem-solving strategies. Genetic algorithms are search algorithms based on the mechanics of natural genetics, i.e., operations existing in nature. They combine a Darwinian 'survival of the fittest approach' with a structured, yet randomized, information exchange. The advantage is that they can search complex and large amount of spaces efficiently and locate near-optimal solutions pretty rapidly. GAs was developed in the early 1970s by John Holland at the University of Michigan (Adaptation *in Natural and Artificial Systems,* 1975).

A genetic algorithm operates on a set of individual elements (the population) and there is a set of biologically inspired operators that can change these individuals. According to the evolutionary theory_ only the more suited individuals in the population are likely to survive and to generate offspring, thus transmitting their biological heredity to new generations.

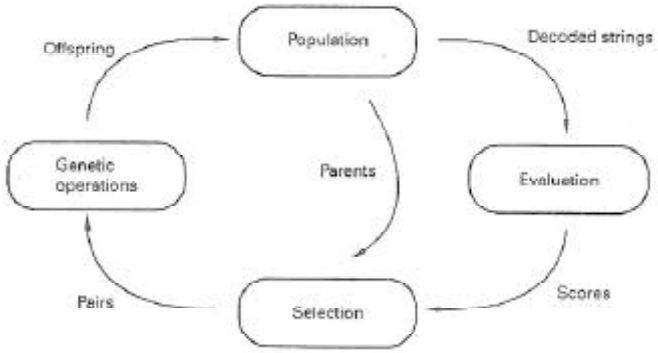
In computing terms, genetic algorithms map strings of numbers to each potential solution. Each solution becomes an "individual in the population, and each string becomes a representation of an individual. There should be a way to derive each individual from its string representation. The genetic algorithm then manipulates the most promising strings in its search for an improved solution. The algorithm operates through a simple cycle:

- Creation of a population of strings.
- Evaluation of each string.
- Selection of the best strings.
- Genetic manipulation to create a new population of strings.

Figure 27.2 shows how these four stages interconnect. Each cycle produces a new generation of possible solutions (individuals) for a given problem. At the first stage, a population of possible solutions is created as a starting point. Each individual

in this population is enc6ded into a string (the chromosome) to be manipulated by the genetic operators. In the next stage, the individuals are evaluated, first the individual is created from its string description (its chromosome), then its performance in relation to the target response is evaluated. This determines how fit this individual is in relation to the others. in the population. Based on each individual's fitness, a selection mechanism chooses the best pairs for the genetic manipulation process. The selection policy is responsible to ensure the survival of the fittest individuals.

The manipulation process enables the genetic operators to produce a new population of individuals, the offspring, by manipulating the genetic information possessed by the pairs chosen to reproduce. This information is stored in the strings (Chromosomes) that describe the individuals. Two operators are used: *Crossover* and *Mutation.* The offspring generated by this process take the place of the older population and the cycle is repeated until a desired level of fitness is attained, or a determined number of cycles is reached.



27.2 The "Reproduction" Cycle

Crossover

Crossover is one of the genetic operators used to recombine the population's genetic material. It takes two chromosomes and swaps part of their genetic information to produce new chromosomes. As Figure 27.3 shows, after the crossover point has been randomly chosen, portions of the parent's chromosome (strings). Parent 1 and Parent 2 are combined to produce the new offspring, Son.

Genetic Algorithms In detail

Genetic algorithms (GAs) are a class of randomized search processes capable of adaptive and robust search over a wide range of search space to pologies. Modeled after the adaptive emergence of biological species from evolutionary mechanisms, and introduced by Holland GAs have been successfully applied in such diverse. Fields such as image analysis, scheduling, and engineering design.

Genetic algorithms extend the idea from human genetics of the four-letter alphabet loosed on the A, C, T, G nucleotides) of the human DNA code. The construction of a genetic algorithm involves devising an alphabet that encodes the solutions to the deci-sion problem in terms of strings of that alphabet. Strings are equivalent to individuals. A fitness function defines which solutions, can survive and which cannot. The ways in which solutions can be combined are patterned after the crossover operation of cutting and combining strings from a father and a mother. An initial population of well-varied population is provided, and a game of evolution is played in which mutations occur among strings. They combine to produce a new generation of individuals the fittest indi-viduals survive and mutate until a family of successful solutions develops.

The solutions produced by genetic algorithms (GAs) are distinguished from most other search techniques by the following characteristics:

- A GA search uses a set of solutions during each generation rather than a single solu-tion.
- The search in the string-space represents a much larger parallel search in the space of encoded solutions. .
- The memory of the search done is represented solely by the set of solutions available for a generation.
- A genetic algorithm is a randomized algorithm since search mechanisms use 'probabi-listic operators.
- While progressing from one generation to next, a GA finds near-optimal balance between knowledge acquisition and exploitation by manipulating encoded solutions.

Genetic algorithms are used for problem solving and clustering problems. Their ability to solve problems in parallel provides a powerful tool for data mining. The draw-backs of GAs include the large overproduction, of individual solutions, the random char-acter of the searching process, and the high demand on computer processing. In general, substantial computing power is required to achieve anything of significance with genetic algorithms.

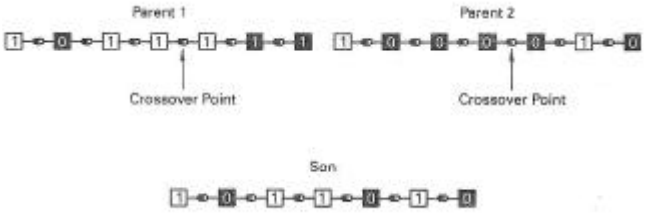


Figure 27.3 Crossover

The selection process associated with the recombination made by the crossover, assures that special genetic structures, called building blocks, are retained for future generations. These building blocks represent the fittest genetic structures in the population.

Mutation

The mutation operator introduces new genetic structures in the population by randomly changing some of its building blocks. Since the modification is totally random and thus not related to any previous genetic structures present in the population, it creates different structures related to other sections of the search space. As shown in Figure 27.4, the mutation is implemented by occasionally altering a random bit from a chromosome (string). The figure shows the operator being applied to the fifth element of the chromosome.

A number of other operators, apart from crossover and mutation, have been introduced since the basic model was proposed. They are usually versions of the recombination and genetic alterations processes adapted to the constraints of a particular problem. Examples of other operators are: inversion, dominance and genetic edge recombination.

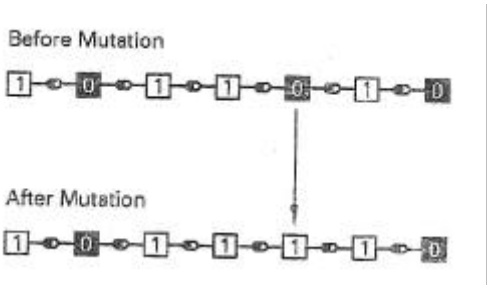


Figure 27.4 Mutation

Problem-Dependent Parameters

This description of the GA' s computational model reviews the steps needed to create the algorithm. However, a real implementation takes into account a number of problem-dependent parameters. For instance, the offspring produced by genetic manipulation (the next population to be evaluated) can either replace the whole population (generational approach) or just its less fit members (steady-state approach). The problem constraints will dictate the best option. Other parameters to be adjusted are the population size, crossover and mutation rates, evaluation method, and convergence criteria.

Encoding

Critical to the algorithm's performance is the choice of underlying encoding for the solution of the optimization problem (the individuals or the population). Traditionally, binary encoding has being used because they are easy to implement. The crossover and mutation operators described earlier are specific to binary encoding. When symbols other than 1 or 0 are used, the crossover and mutation operators must be tailored accordingly.

The Evaluation Step

The evaluation step in the cycle, shown in Figure 27.2, is more closely related to the actual application the algorithm is trying to optimize. It takes the strings representing the individuals of the population and, from them, creates the actual individuals to be tested the way the individuals are coded as strings will depend on what parameters one is tying to optimize and the actual structure of possible solutions (individuals). After the actual individuals have been created, they have to be tested and scored. These two tasks again are closely related to the actual system being optimized. The testing depends on what characteristics should be optimized and the scoring. The production of a single value representing the fitness of an individual depends on the relative importance of each different characteristic value obtained during testing.

Data Mining using GA

The application of the genetic algorithm in the context of data mining is generally for the tasks of hypothesis testing and refinement, where the user poses some hypothesis and the system first evaluates the hypothesis and then seeks to refine it. Hypothesis refinement is achieved by "seeding" the system with the hypothesis and then allowing some or all parts of it to vary. One can use a variety of evaluation functions to determine the fitness of a candidate refinement. The important aspect of the GA application is the encoding of the hypothesis and the evaluation function for fitness.

Another way to use GA for data mining is to design hybrid techniques by blending one of the known techniques with GA. For example, it is possible to use the genetic algorithm for optimal decision tree induction. By randomly generating different samples, we can build many decision trees using any of the traditional techniques. But we are not sure of the optimal tree. At this stage, the GA is very useful in deciding on the optimal tree and optimal splitting attributes. The genetic algorithm evolves a population of biases for the decision tree induction algorithm. We can use a two-tiered search strategy. On the bottom tier, the traditional greedy strategy is performed through the space of the decision trees. On the top tier, one can have a genetic search in a space of biases. The attribute selection parameters are used as biases, which are used to modify the behavior of the first tier search. In other words, the GA controls the preference for one type of decision tree over another.

An individual (a bit string) represents a bias and is evaluated by using testing data subsets. The "fitness" of the individual is the average cost of classification of the decision tree. In the next generation, the population is replaced with new individuals. The new individuals are generated from the previous generation, using mutation and crossover. The fittest individuals in the first generation have the most offspring in the second generation. After a fixed number of generations, the algorithm halts and its output is the decision tree the determined by the fittest individual.

Exercises

1. Write short notes on
   a. Mutation
   b. Negative Associations
   c. Partition algorithm
2. Discuss the importance of association rules.
3. Explain the basic Algorithms for Finding Association Rules.
4. Discuss the importance of crossover in Genetic algorithm.
5. Explain Association Rules among Hierarchies with example.
6. Describe the principle of Genetic algorithm and discuss its suitability to data mining.
7. Discuss the salient features of the genetic algorithm. How can a data-mining problem be an optimization problem? How do you use GA for such cases?

Reference

- Goldberg D.E., *Genetic Algorithm in Search, Optimization and Machine Learning,* Addison-Wesley Publishing Company, 1989.
- Holland J.H. *Adaptation in Natural and Artificial System* (2nd ed.), Prentice Hall, 1992.
- Marmelstein R., and Lamont G. Pattern Classification using a Hybrid Genetic Program-Decision Tree Approach. In *Proceedings of the Jed Annual Genetic Programming Conference,* 223-231, 1998.
- McCallum R., and Spackman K. Using genetic algorithms to learn disjunctive rules from examples. In *Proceedings of the 7th International Conference on Machine Learning,* 149-152, 1990.
- Ryan M.D., and Rayward-Smith V.J. The evolution of decision trees. In *Proceedings of the Third Annual Genetic Programming Conference,* 350-358, 1998.
- Syswerda G. In *First Workshop on the Foundations of Genetic Algorithms and Classification Systems,* Morgan Kaufmann, 1990.

Notes

LESSON 28
ONLINE ANALYTICAL PROCESSING, NEED FOR OLAP MULTIDIMENSIONAL DATA MODEL

Structure

- Objective
- Introduction
- On-line Analytical processing
- What is Multidimensional (MD) data and when does it become OLAP?
- OLAP Example
- What is OLAP?
- Who uses OLAP and WHY?
- Multi-Dimensional Views
- Complex Calculation capabilities
- Time intelligence

Objective

At the end of this lesson you will be able to

- Understand the significance of OLAP in Data mining
- Study about Multi-Dimensional Views, Complex Calculation capabilities, and Time intelligence

Introduction

This lesson focuses on the need of Online Analytical Processing. Solving modern business problems such as market analysis and financial forecasting requires query-centric database schemas that are array-oriented and multidimensional in nature. These business problems are characterized by the need to retrieve large numbers of records from very large data sets and summarize them on the fly. The multidimensional nature of the problems it is designed to address is the key driver for OLAP.

In this lesson i will cover all the important aspects of OLAP.

On Line Analytical Processing

A major issue in information processing is how to process larger and larger databases, containing increasingly complex data, without sacrificing response time. The client/server architecture gives organizations the opportunity to deploy specialized servers, which are optimized for handling specific data management problems. Until recently, organizations have tried to target relational database management systems (RDBMSs) for the complete spectrum of database applications. It is however apparent that there are major categories of database applications which are not suitably serviced by relational database systems. Oracle, for example, has built a totally new Media Server for handling multimedia applications. Sybase uses an object-oriented DBMS (OODBMS) in its Gain Momentum product, which is designed to handle complex data such as images and audio. Another category of applications is that of on-line analytical processing (OLAP). OLAP was a term coined by E F Codd (1993) and was defined by him as;

The dynamic synthesis, analysis and consolidation of large volumes of multidimensional data

Codd has developed rules or requirements for an OLAP system;

- Multidimensional conceptual view
- Transparency
- Accessibility
- Consistent reporting performance
- Client/server architecture
- Generic dimensionality
- Dynamic sparse matrix handling
- Multi-user support
- Unrestricted cross dimensional operations
- Intuitative data manipulation
- Flexible reporting
- Unlimited dimensions and aggregation levels

An alternative definition of OLAP has been supplied by Nigel Pendse who unlike Codd does not mix technology prescriptions with application requirements. Pendse defines OLAP as, **Fast Analysis of Shared Multidimensional Information**, which means:

Fast in that users should get a response in seconds and so doesn't lose their chain of thought;

Analysis in that the system can provide analysis functions in an intuitative manner and that the functions should supply business logic and statistical analysis relevant to the users application;

Shared from the point of view of supporting multiple users concurrently;

Multidimensional as a main requirement so that the system supplies a multidimensional conceptual view of the data including support for multiple hierarchies;

Information is the data and the derived information required by the user application.

One question that arises is,

What is Multidimensional (MD) data and when does it become OLAP?

It is essentially a way to build associations between dissimilar pieces of information using predefined business rules about the information you are using. Kirk Cruikshank of Arbor Software has identified three components to OLAP, in an issue of UNIX News on data warehousing;

- A multidimensional database must be able to express complex business calculations very easily. The data must be referenced and mathematics defined. In a relational system there is no relation between line items, which makes it very difficult to express business mathematics.

- Intuitive navigation in order to 'roam around' data, which requires mining hierarchies.
- Instant response i.e. the need to give the user the information as quickly as possible.

Dimensional databases are not without problem as they are not suited to storing all types of data such as lists for example customer addresses and purchase orders etc. Relational systems are also superior in security, backup and replication services as these tend not to be available at the same level in dimensional systems. The advantages of a dimensional system are the freedom they offer in that the user is free to explore the data and receive the type of report they want without being restricted to a set format.

OLAP Example

An example OLAP database may be comprised of sales data which has been aggregated by region, product type, and sales channel. A typical OLAP query might access a multi-gigabyte/ multi-year sales database in order to find all product sales in each region for each product type. After reviewing the results, an analyst might further refine the query to find sales volume for each sales channel within region/product classifications. As a last step the analyst might want to perform year-to-year or quarter-to-quarter comparisons for each sales channel. This whole process must be carried out on-line with rapid response time so that the analysis process is undisturbed. OLAP queries can be characterized as on-line transactions which:

- Access very large amounts of data, e.g. several years of sales data.
- Analyze the relationships between many types of business elements e.g. sales, products, regions, and channels.
- Involve aggregated data e.g. sales volumes, budgeted dollars and dollars spent.
- Compare aggregated data over hierarchical time periods e.g. monthly, quarterly, and yearly.
- Present data in different perspectives e.g. sales by region vs. sales by channels by product within each region.
- Involve complex calculations between data elements e.g. expected profit as calculated as a function of sales revenue for each type of sales channel in a particular region.
- Are able to respond quickly to user requests so that users can pursue an analytical thought process without being stymied by the system.

What is OLAP?

- Relational databases are used in the areas of operations and control with emphasis on transaction processing.
- Recently relational databases are used for building data warehouses, which stores tactical information (< 1 year into the future) that answers who and what questions.
- In contrast OLAP uses Multi-Dimensional (MD) views of aggregate data to provide access strategic information.
- OLAP enables users to gain insight to a wide variety of possible views of information and transforms raw data to reflect the enterprise as understood by the user e.g. Analysts, managers and executives.
- In addition to answering who and what questions OLAPs can answer "what if " and "why".
- Thus OLAP enables strategic decision-making.
- OLAP calculations are more complex than simply summing data.
- However, OLAP and Data Warehouses are complementary
- The data warehouse stores and manages data while the OLAP transforms this data into strategic information.

Who uses OLAP and WHY?

- OLAP applications are used by a variety of the functions of an organisation.
- Finance and accounting:
  - Budgeting
  - Activity-based costing
  - Financial performance analysis
  - And financial modelling
- Sales and Marketing
  - Sales analysis and forecasting
  - Market research analysis
  - Promotion analysis
  - Customer analysis
  - Market and customer segmentation
- Production
  - Production planning
  - Defect analysis

Thus, OLAP must provide managers with the information they need for effective decision-making. The KPI (key performance indicator) of an OLAP application is to provide just-in-time (JIT) information for effective decision-making. JIT information reflects complex data relationships and is calculated on the fly. Such an approach is only practical if the response times are always short The data model must be flexible and respond to changing business requirements as needed for effective decision making.

In order to achieve this in widely divergent functional areas OLAP applications all require:

- MD views of data
- Complex calculation capabilities
- Time intelligence

Multi-Dimensional Views

- MD views inherently represent actual business models, which normally have more than three dimensions e.g., Sales data is looked at by product, geography, channel and time.
- MD views provide the foundation for analytical processing through flexible access to information.
- MD views must be able to analyse data across any dimension at any level of aggregation with equal functionality and ease and insulate users from the complex query syntax
- What ever the query is they must have consistent response times.

- Users queries should not be inhibited by the complex to form a query or receive an answer to a query.
- The benchmark for OLAP performance investigates a servers ability to provide views based on queries of varying complexity and scope.
    - Basic aggregation on some dimensions
    - More complex calculations are performed on other dimensions
- Ratios and averages
- Variances on sceneries
- A complex model to compute forecasts
- Consistently quick response times to these queries are imperative to establish a server's ability to provide MD views of information.

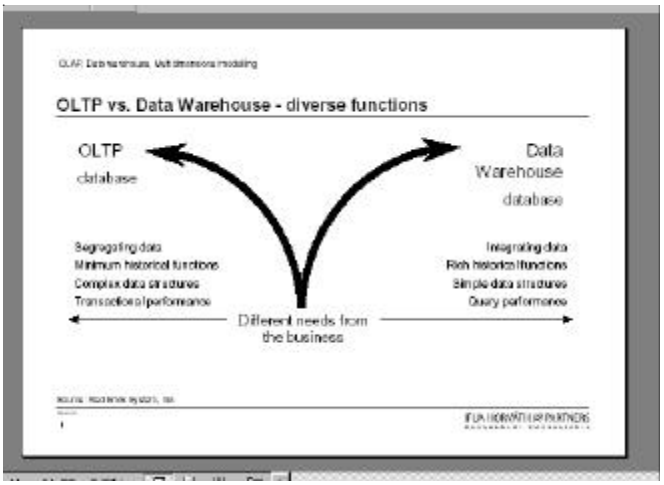
Complex Calculations

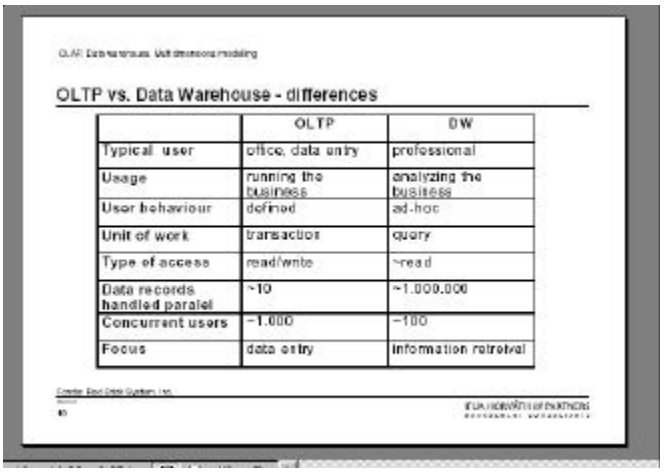
- The ability to perform complex calculations is a critical test for an OLAP database.
- Complex calculations involve more than aggregation along a hierarchy or simple data roll-ups, they also include percentage of the total share calculations and allocations utilising hierarchies from a top-down perspective.
- Further calculations include:
    - Algebraic equations for KPI
    - Trend algorithms for sales forecasting
    - Modelling complex relationships to represent real world situations
- OLAP software must provide powerful yet concise computational methods.
- The method for implementing computational methods must be clear and non-procedural
- Its obvious why such methods must be clear but they must also be non-procedural otherwise changes can not be done in a timely manner and thus eliminate access to JIT information.
- In essence OLTP systems are judged on their ability to collect and manage data while OLAP systems are judged on their ability to make information from data. Such abilities involves the use of both simple and complex calculations

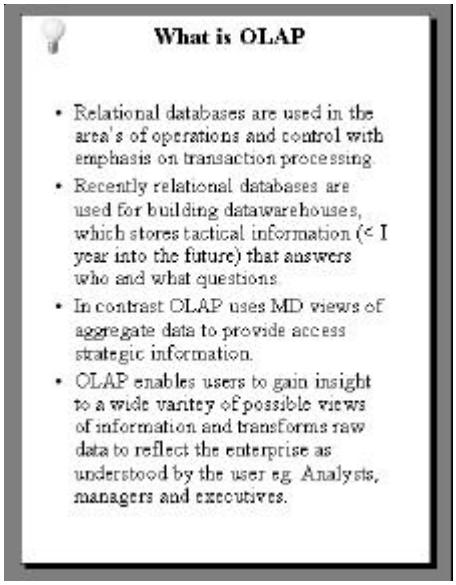
Time Intelligence

- Time is important for most analytical applications and is a unique dimension in that it is sequential in character. Thus true OLAP systems understand the sequential nature of time.
- The time hierarchy can be used in a different way to other hierarchies eg sales for june or sales for the first 5 months of 2000.
- Concepts such as year to date must be easily defined
- OLAP must also understand the concept of balances over time. Eg in some cases, for employees, an average is used while in other cases an ending balance is used.

The OLAP performance benchmark contains how time is used in OLAP applications. Eg the forecast calculation uses this year's vs. last year's knowledge, year-to-date knowledge factors.







What is OLAP

- In addtion to answering who and what questions OLAPs can answer "what if " and "why".
- Thus OLAP enables strategic decision making.
- OLAP calculations are more complex than simply summing data.
- However, OLAP and Data Warehouses are complementary
- The data warehouse stores and manages data while the OLAP transforms this data into strategic information.

Who uses OLAP and Why

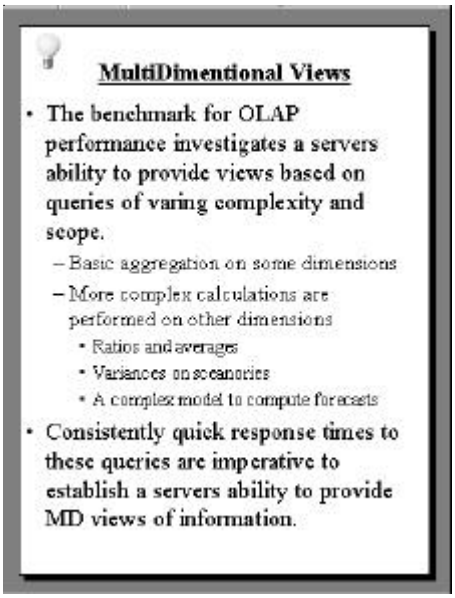
- Thus OLAP must provide managers with the information they need for effective decision making.
- The KPI (key performance indicator) of an OLAP application is to provide just-in-time(JIT) information for effective decision making.
- JIT information reflects complex data relationships and is calculated on the fly.
- Such an approach is only practical if the response times are always short
- The data model must be flexible and respond to changing business requirements as needed for effective decision making.

Who uses OLAP and WHY

- OLAP applications are used by a variety of the functions of an organisation.
- **Finance and accounting:**
  - Budgeting
  - Activity-based costing
  - Financial performance analysis
  - And financial modelling
- **Sales and Marketing**
  - Sales analysis and forecasting
  - Market research analysis
  - Promotion analysis
  - Customer analysis
  - Market and customer segmentation
- **Production**
  - Production planning
  - Defect analysis

MultiDimentional Views

- MD views inherently represent actual bsuiness models which normally has more than three dimensions eg. Sales data is looked at by product, geography, channel and time.
- MD views provide the foundation for analytical processing through flexible access to information.
- MD views must be able to analyse data across any dimension at any level of aggregation with equal functionality and ease and insulate users from the complex query syntax
- What ever the query is they must have consistent response times .
- Users queries should not be inhibited by the complex to form a query or receive an answer to a query.



Exercise

1. Write short notes on:
   - Multidimensional Views
   - Time Intelligence
   - Complex Calculations
2. What do you understand by Online Analytical Processing (OLAP)? Explain the need of OLAP.
3. Who uses OLAP and why?
4. Correctly contrast the difference between OLAP and Data warehouse.
5. Discuss various applications of OLAP.

Notes

LESSON 29
OLAP VS. OLTP, CHARACTERISTICS OF OLAP

Structure

- Objective
- Definitions of OLAP
- Comparison of OLAP and OLTP
- Characteristics of OLAP: FASMI
- Basic Features of OLAP
- Special features

Objective

At the end of this lesson you will be able to

- Understand the significance of OLAP.
- Compare between OLAP and OLTP
- Learn about various characteristics of OLAP.

Definitions of OLAP

In a white paper entitled 'Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate', E.F. Codd established 12 rules to define an OLAP system. In the same paper he listed three characteristics of an OLAP system. Dr. Codd later added 6 additional features of an OLAP system to his original twelve rules.

Three significant characteristics of an OLAP system

- Dynamic Data Analysis

This refers to time series analysis of data as opposed to static data analysis, which does not allow for manipulation across time. In an OLAP system historical data must be able to be manipulated over multiple data dimensions. This allows the analysts to identify trends in the business.

- Four Enterprise Data Models
- The Categorical data model describes what has gone on before by comparing historical values stored in the relational database. The Exegetical data model reflects what has previously occurred to bring about the state, which the categorical model reflects. The Contemplative data model supports exploration of 'what-if' scenarios. The Formulaic data model indicates which values or behaviors across multiple dimensions must be introduced into the model to affect a specific outcome.

Comparison of OLAP and OLTP

OLAP applications are quite different from On-line Transaction Processing (OLTP) applications, which consist of a large number of relatively simple transactions. The transactions usually retrieve and update a small number of records that are contained in several distinct tables. The relationships between the tables are generally simple.

A typical customer order entry OLTP transaction might retrieve all of the data relating to a specific customer and then insert a new order for the customer. Information is selected from the customer, customer order, and detail line tables. Each row in each table contains a customer identification number, which is used to relate the rows from the different tables. The relationships between the records are simple and only a few records are actually retrieved or updated by a single transaction.

The difference between OLAP and OLTP has been summarized as, OLTP servers handle mission-critical production data accessed through simple queries; while OLAP servers handle management-critical data accessed through an iterative analytical investigation. Both OLAP and OLTP have specialized requirements and therefore require special optimized servers for the two types of processing.

OLAP database servers use multidimensional structures to store data and relationships between data. Multidimensional structures can be best visualized as cubes of data, and cubes within cubes of data. Each side of the cube is considered a dimension.

Each dimension represents a different category such as product type, region, sales channel, and time. Each cell within the multidimensional structure contains aggregated data relating elements along each of the dimensions. For example, a single cell may contain the total sales for a given product in a region for a specific sales channel in a single month. Multidimensional databases are a compact and easy to understand vehicle for visualizing and manipulating data elements that have many inter relationships.

OLAP database servers support common analytical operations including: consolidation, drill-down, and "slicing and dicing".

- **Consolidation** - involves the aggregation of data such as simple roll-ups or complex expressions involving inter-related data. For example, sales offices can be rolled-up to districts and districts rolled-up to regions.
- **Drill-Down** - OLAP data servers can also go in the reverse direction and automatically display detail data, which comprises consolidated data. This is called drill-downs. Consolidation and drill-down are an inherent property of OLAP servers.
- "**Slicing and Dicing**" - Slicing and dicing refers to the ability to look at the database from different viewpoints. One slice of the sales database might show all sales of product type within regions. Another slice might show all sales by sales channel within each product type. Slicing and dicing is often performed along a time axis in order to analyse trends and find patterns.

OLAP servers have the means for storing multidimensional data in a compressed form. This is accomplished by dynamically selecting physical storage arrangements and compression techniques that maximize space utilization. Dense data (i.e., data exists for a high percentage of dimension cells) are stored separately from sparse data (i.e., a significant percentage of cells are empty). For example, a given sales channel may only sell a

few products, so the cells that relate sales channels to products will be mostly empty and therefore sparse. By optimizing space utilization, OLAP servers can minimize physical storage requirements, thus making it possible to analyse exceptionally large amounts of data. It also makes it possible to load more data into computer memory, which helps to significantly improve performance by minimizing physical disk I/O.

In conclusion OLAP servers logically organize data in multiple dimensions, which allows users to quickly, and easily analyze complex data relationships. The database itself is physically organized in such a way that related data can be rapidly retrieved across multiple dimensions. OLAP servers are very efficient when storing and processing multidimensional data. RDBMSs have been developed and optimized to handle OLTP applications. Relational database designs concentrate on reliability and transaction processing speed, instead of decision support need. The different types of server can therefore benefit a broad range of data management applications.

Characteristics of OLAP: FASMI

Fast – means that the system targeted to deliver most responses to user within about five second, with the simplest analysis taking no more than one second and very few taking more than 20 seconds.

Analysis – means that the system can cope with any business logic and statistical analysis that it relevant for the application and the user, keep it easy enough for the target user. Although some pre programming may be needed we do not think it acceptable if all application definitions have to be allow the user to define new adhoc calculations as part of the analysis and to report on the data in any desired way, without having to program so we exclude products (like Oracle Discoverer) that do not allow the user to define new adhoc calculation as part of the analysis and to report on the data in any desired product that do not allow adequate end user oriented calculation flexibility.

Share – means that the system implements all the security requirements for confidentiality and, if multiple write access is needed, concurrent update location at an appropriated level not all applications need users to write data back, but for the growing number that do, the system should be able to handle multiple updates in a timely, secure manner.

Multidimensional – is the key requirement. OLAP system must provide a multidimensional conceptual view of the data, including full support for hierarchies, as this is certainly the most logical way to analyze business and organizations.

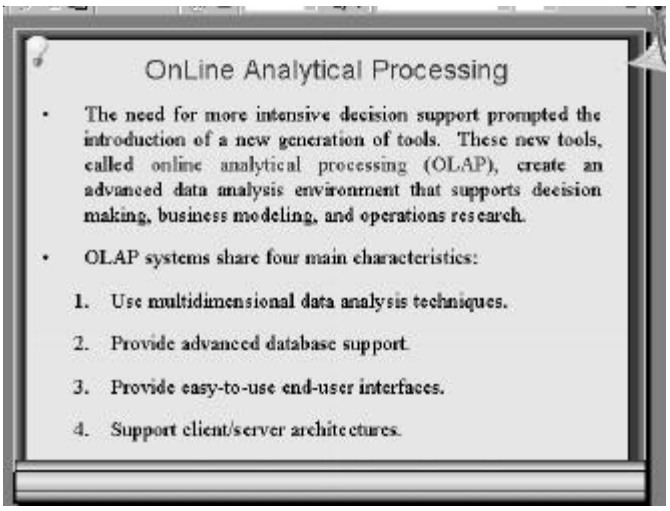
Information – are all of the data and derived information needed? Wherever it is and however much is relevant for the application. We are measuring the capacity of various products in terms of how much input data they can handle, not how many gigabytes they take to store it.

Basic Features of OLAP

- Multidimensional Conceptual view: We believe this to be the central core of OLAP
- Initiative data manipulation: Dr. Codd prefers data manipulation to be done through direct action on cells in the view w/o resource to menus of multiple actions.
- Accessibility: OLAP as a mediator, Dr. Codd essentially describes OLAP engines as middleware, sitting heterogeneous data sources & OLAP front-end.
- Batch Extraction vs. Interpretive: this rule effectively requires that products offer both their own staging database for OLAP data as well as offering live access to external data.
- OLAP analysis models: Dr. Codd requires that OLAP products should support all four-analysis models that describes in his white paper model
- Client server architecture: Dr. Codd requires not only that the product should be client/server but that the server component of an OLAP product should be sufficiently intelligent that various client can be attached with minimum effort and programming for integration.
- Transparency: full compliance means that a user should be able to get full value from an OLAP engine and not even be aware of where the data ultimately comes from. To do this products must allow live excess to heterogeneous data sources from a full function spreadsheet add-in, with the OLAP server in between.
- Multi-user support: Dr. Codd recognizes that OLAP applications are not all read-only, & says that, to be regarded as strategic, OLAP tools must provide concurrent access, integrity & security.

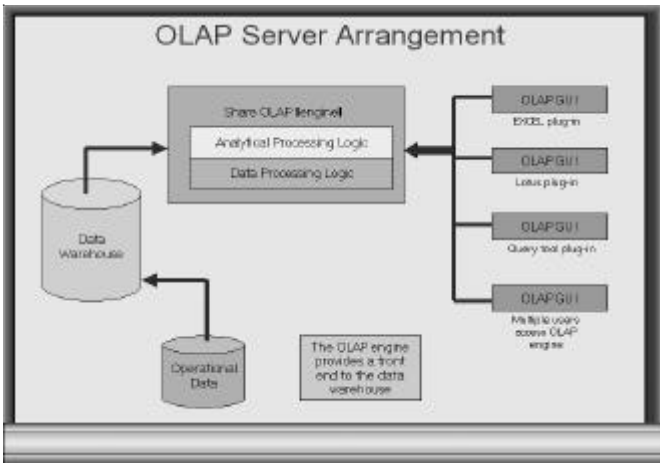
Special features

- Treatment of non-normalize data: this refers to the integration between an OLAP engine and denormalized source data.
- Storing OLAP Result: keeping them separate from source data. This is really an implementation rather than a product issue. Dr. Codd is endorsing the widely held view that read-write OLAP applications should not be implemented directly on live transaction data, and OLAP data changes should be kept distinct from transaction data.
- Extraction of missing value: all missing value are to be distinguished from zero values.
- Treatment of Missing values: all missing values to be ignored by the OLAP analyzer regardless of their source.



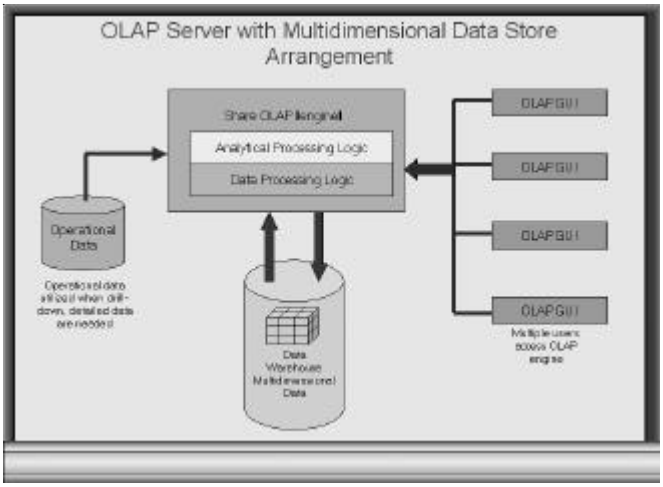
Multidimensional Data Analysis Techniques

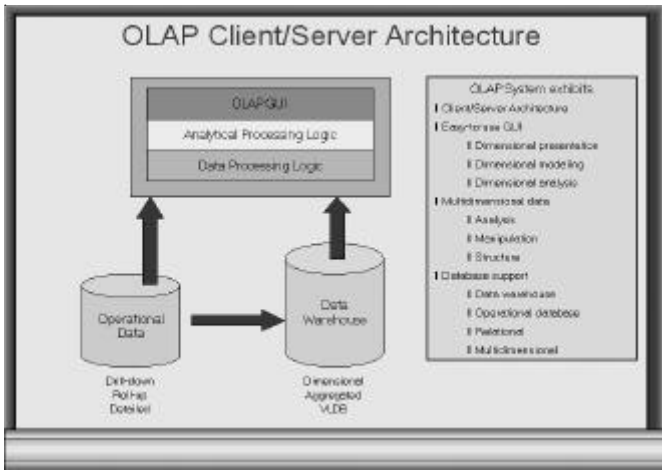
- The most distinct characteristic of OLAP tools is their capacity for multidimensional analysis. In multidimensional analysis, data are processed and viewed as part of a multidimensional structure. This view of data analysis is particularly attractive to business decision makers because they tend to view business data as data that are related to other business data.



Multidimensional analysis techniques are augmented by:

- **Advanced data presentation functions: 3D graphics, pivot tables, crosstabs, data rotation, three-dimensional cubes, and so on.**
- **Advanced data aggregation, consolidation, and classification functions that all the business data analyst to create multiple data aggregation levels, slice and dice, and drill down and roll up data across different dimensions and aggregation levels. For example aggregating data across the time dimension (by day, week, month, quarter, year) allows the analyst to drill down and roll up across time dimensions.**
- **Advanced computational functions: business-oriented variables (market share, period comparisons, sales margins), financial and accounting ratios (profitability, overhead, cost allocations, returns, etc.).**
- **Advanced data modeling functions: support for "what-if" scenarios, variable assessment, linear programming, variable contributions to outcome, etc.**





Exercise

1. Write short notes on
   - Client Server Architecture
   - Slicing and Dicing
   - Drill down
2. Correctly contrast and Compare OLAP and OLTP with example.
3. What is FASMI? Explain in brief.
4. Explain various Basic Features of OLAP
5. Discuss the importance of Multidimensional View in OLAP. Explain with an example.

LESSON 30
MULTIDIMENSIONAL VERSES MULTIRELATIONAL OLAP, FEATURES OF OLAP

Structure

- Objective
- Introduction
- Multidimensional Data Model
- Multidimensional versus Multirelational OLAP
- OLAP Guidelines

Objective

At the end of this lesson you will be able to

- Study in detail about Multidimensional Data Model
- Understand the difference between Multidimensional verses Multirelational OLAP.
- Identify various OLAP Guidelines

Introduction

OLAP is an application architecture, not intrinsically a data warehouse or a database management system (DBMS). Whether it uti-lizes a data warehouse or not, OLAP is becoming an architecture that an increasing number of enterprises are implementing to support analytical applications. The majority of OLAP applications are deployed in a "stovepipe" fashion, using specialized MDDBMS technology, a narrow set of data, and, often, a prefabricated application-user interface. AB we look at OLAP trends, we can see that the architectures have clearly defined layers and that delin-eation exists between the application and the DBMS.

Solving modern business problems such as market analysis and financial fore-casting requires query-centric database schemas that are array-oriented and multidimensional in nature. These business problems are characterized by the need to retrieve large numbers of records from very large data sets (hundreds of gigabytes and even terabytes) and summarize them on the fly. The multi-dimensional nature of the problems it is designed to address is the key driver for OLAP.

The result set may look like a multidimensional spreadsheet (hence. the term multidimensional). Although all the necessary data can be represented in a relational database and accessed via SQL, the two-dimensional relational model of data and the Structured Query Language (SQL) have some serious limitations for such complex real-world problems. For example, a query may translate into a number of complex SQL statements, each of which may involve full table scan, multiple joins, aggregations and sorting, and large temporary tables for storing intermediate results. The resulting query may require signif-icant computing resources that may not be available at all times and even then may take a long time to complete. Another drawback of SQL is its weakness in handling time series data and complex mathematical functions. Time series calculations such as a 3-month moving average or net present value calcula-tions typically require extensions to ANSI SQL rarely found in commercial products.

Response time and SQL functionality are not the only problems. OLAP is a continuous, iterative, and preferably interactive process. An analyst may drill down into the data to see, for example, how an individual salesperson's per-formance affects monthly revenue numbers. At the same time, the drill-down procedure may help the analyst discover certain patterns in sales of given products. This discovery can force another set of questions of similar or greater complexity. Technically, all these analytical questions can be answered by a large number of rather complex queries against a set of detailed and presum-marized data views. In reality, however, even if the analyst could quickly and accurately formulate SQL statements of this complexity, the response time and resource consumption problems would still persist, and the analyst's produc-tivity would be seriously impacted.

Multidimensional Data Model

The multidimensional nature of business questions is reflected in the fact that, for example, marketing managers are no longer satisfied by asking simple one-dimensional questions such as "How much revenue did the new product gener-ate?" Instead, they ask questions such as "How much revenue did the new product generate by month, in the northeastern division, broken down by user demographic, by sales office, relative to the previous version of the product, compared the with plan?"- A six-dimensional question. One way to look at the multidimensional data model is to view it as a cube (see Fig. 30.1). The table on the left contains detailed sales data by product, market, and time. The cube on the right associates sales numbers (units sold) with dimensions-product type, market, and time-with the UNIT variables organized as *cells* in an *array*. This cube can be expanded to include another array-price-which can be associated with all or only some dimensions (for example, the unit price of a product mayor may not change with time, or from city to city). The cube sup-ports matrix arithmetic that allows the cube to present the dollar sales array simply by performing a *single* matrix operation on all cells of the array {dollar sales = units * price}.

The response time of the multidimensional query still depends on how many cells have to be added on the fly. The caveat here is that, as the number of dimensions increases, the number of the cube's cells increases exponentially. On the other hand, the majority of multidimensional queries deal with sum-marized, high-level data. Therefore, the solution to building an efficient multi-dimensional database is to preaggregate (consolidate) all logical subtotals and totals along all dimensions. This preaggregation is especially valuable since typical dimensions are hierarchical in nature. For example, the TIME dimen-sion may contain hierarchies for years, quarters, months, weeks, and days;

GEOGRAPHY may contain country, state, city, etc. Having the predefined hier-archy within dimensions allows for logical pre aggregation and, conversely, allows for a logical drill-down-from the product group to individual products, from annual sales to weekly sales, and so on.

Another way to reduce the size of the cube is to properly handle *sparse* data. Often, not every cell has a meaning across all dimensions (many marketing databases may have more than 95 percent of all cells empty or containing 0). Another kind of sparse data is created when many cells contain duplicate data (Le., if the cube contains a PRICE dimension, the same price may apply to all markets and all quarters for the year). The ability of a multidimensional data-base to skip empty or repetitive cells can greatly reduce the size of the cube and the amount of processing.

Dimensional hierarchy, sparse data management, and pre aggregation are the keys, since they can significantly reduce the size of the database and the need to calculate values. Such a design obviates the need for multi table joins and provides quick and direct access to the arrays of answers, thus signifi-cantly speeding up execution of the multidimensional queries.

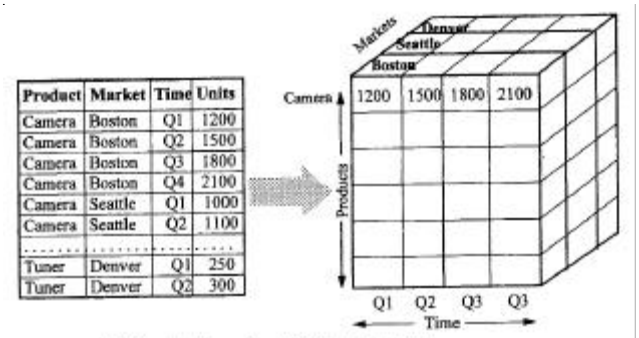


Fig :30.1 Relational tables and multidimensional cubes.

Multidimensional versus Multirelational OLAP

These relational implementations of multidimensional database systems are sometimes referred to as *multirelational* database systems. To achieve the required speed, these products use the star or snowflake schemas-specially optimized and denormalized data models that involve data restructuring and aggregation. (The snowflake schema is an extension of the star schema that supports multiple fact tables and joins between them.)

One benefit of the star schema approach is reduced complexity in the data model, which increases data "legibility," making it eas-ier for users to pose business questions of OLAP nature. Data warehouse queries can be answered up to 10 times faster because of improved navigations.

Two types of database activity:

1. OLTP: On-Line Transaction Processing
   - Short transactions, both queries and updates

   (e.g., update account balance, enroll in course)
   - Queries are simple

   (e.g., find account balance, find grade in course)
   - Updates are frequent

   (e.g., concert tickets, seat reservations, shopping carts)
2. OLAP: On-Line Analytical Processing
   - Long transactions, usually complex queries

   (e.g., all statistics about all sales, grouped by dept and month)
   - "Data mining" operations
   - Infrequent updates

OLAP Guidelines

The data that is presented through any OLAP access route should be identical to that used in operational systems. The values achieved through 'drilling down' on the OLAP side should match the data accessed through an OLTP system.

12 Rules satisfied by an OLAP system

1. Multi-Dimensional Conceptual View

   This is a key feature of OLAP. OLAP databases should support multi-dimensional view of the data allowing for 'slice and dice' operations as well as pivoting and rotating the cube of data. This is achieved by limiting the values of dimensions and by changing the orders of the dimensions when viewing the data.
2. Transparency

   Users should have no need to know they are looking at an OLAP database. The users should be focused only upon the tool used to analysis the data, not the data storage.
3. Accessibility

   OLAP engines should act like middleware, sitting between data sources and an OLAP front end. This is usually achieved by keeping summary data in an OLAP database and detailed data in a relational database.
4. Consistent Reporting Performance

   Changing the number of dimensions or the number of aggregation levels should not significantly change reporting performance.
5. Client-Server Architecture

   OLAP tools should be capable of being deployed in a client-server environment. Multiple clients should be able to access the server with minimum effort.
6. Generic Dimensionality

   Each dimension must be equivalent in both its structure and operational capabilities. Data structures, formulae, and reporting formats should not be biased toward any data dimension.
7. Dynamic Sparse Matrix Handling

   A multi-dimensional database may have many cells that have no appropriate data. These null values should be stored in a way that does not adversely affect performance and minimizes space used.
8. Multi-User Support

   OLAP applications should support concurrent access while maintaining data integrity.

9. Unrestricted Cross-Dimensional Operations

   All forms of calculations should be allowed across all dimensions.

10. Intuitive Data Manipulation

    The users should be able to directly manipulate the data without interference from the user interface.

11. Flexible Reporting

    The user should be able to retrieve any view of data required and present it in any way that they require.

12. Unlimited Dimensions and Aggregation Levels

    There should be no limit to the number of dimensions or aggregation levels.

Six additional features of an OLAP system

1. Batch Extraction vs. Interpretive

   OLAP systems should offer both their own multi-dimensional database as well as live access to external data. This describes a hybrid system where users can transparently reach through to detail data.

2. OLAP Analysis Models

   OLAP products should support all four data analysis models described above (Categorical, Exegetical, Contemplative, and Formulaic)

3. Treatment of Non-Normalized Data

   OLAP systems should not allow the user to alter de-normalized data stored in feeder systems. Another interpretation is that the user should not be allowed to alter data in calculated cells within the OLAP database.

4. Storing OLAP Results: keeping them separate from Source Data

   Read-write OLAP applications should not be implemented directly on live transaction data and OLAP data changes should be kept distinct from transaction data.

5. Extraction of Missing Values

   Missing values should be treated as Null values by the OLAP database instead of zeros.

6. Treatment of Missing Values

An OLAP analyzer should ignore missing values.

Many people take issue with the rules put forth by Dr. Codd. Unlike his rules for relational databases, these rules are not based upon mathematical principles. Because a software company, Arbor Software Corporation, sponsored his paper, some members of the OLAP community feel that his rules are too subjective. Nigel Pendise of the OLAP Report has offered an alternate definition of OLAP. This definition is based upon the phrase Fast Analysis of Shared Multidimensional Information (FASMI).

- **Fast**

  The system should deliver most responses to users within a few seconds. Long delays may interfere with the ad hoc analysis.

- **Analysis**

  The system should be able to cope with any business logic and statistical analysis that is relevant for the application.

- **Shared**

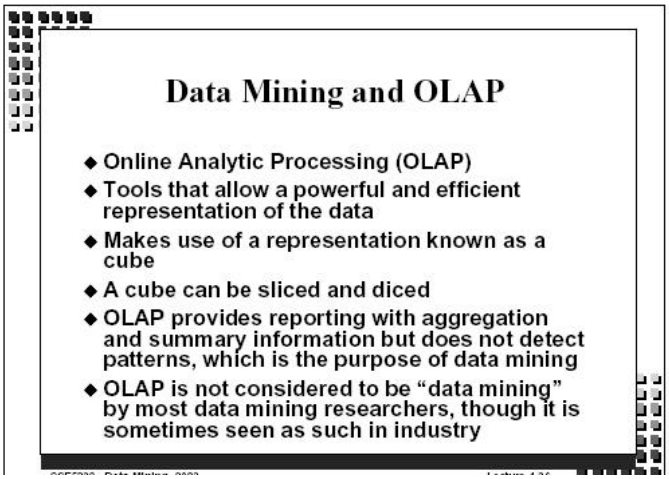
  The system implements all the security requirements for confidentiality. Also, if multiple write access is needed, the system provides concurrent update locking at an appropriate level.

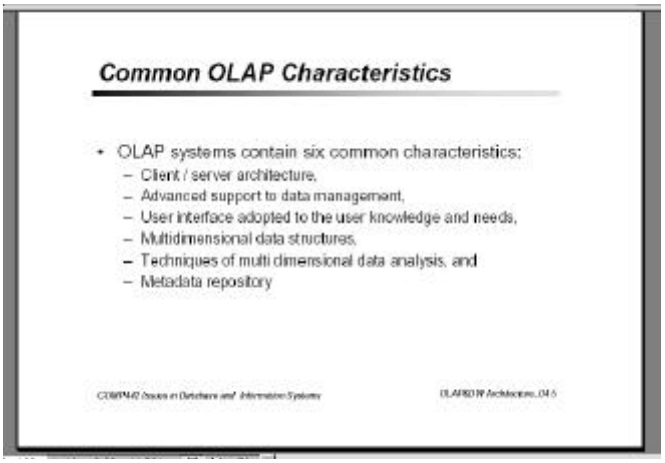
- **Multidimensional**

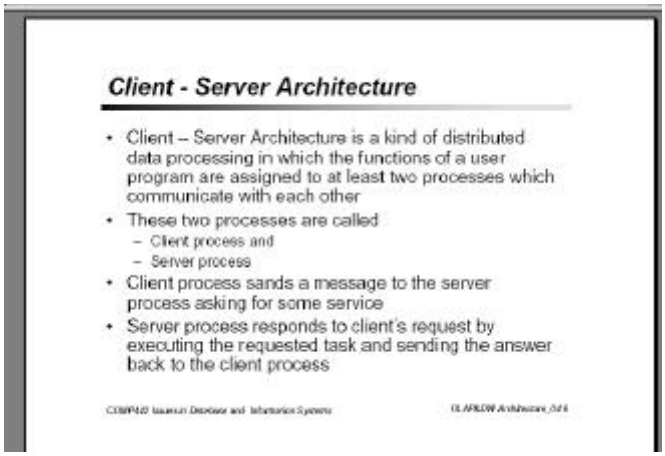
  This is the key requirement. The system should provide a multidimensional conceptual view of the data, including support for hierarchies and multiple hierarchies.

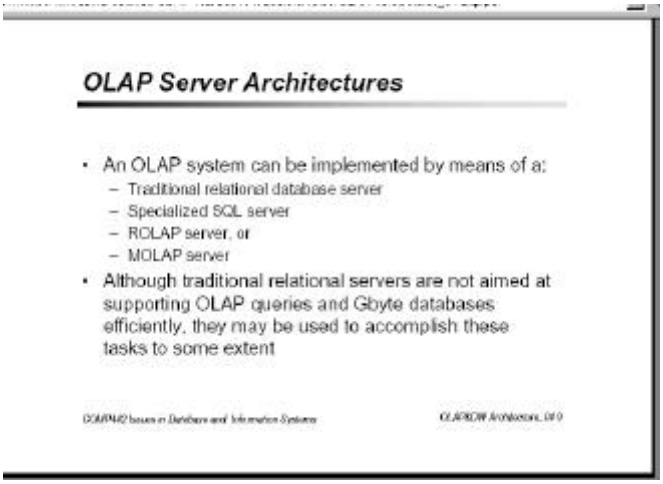
- **Information**

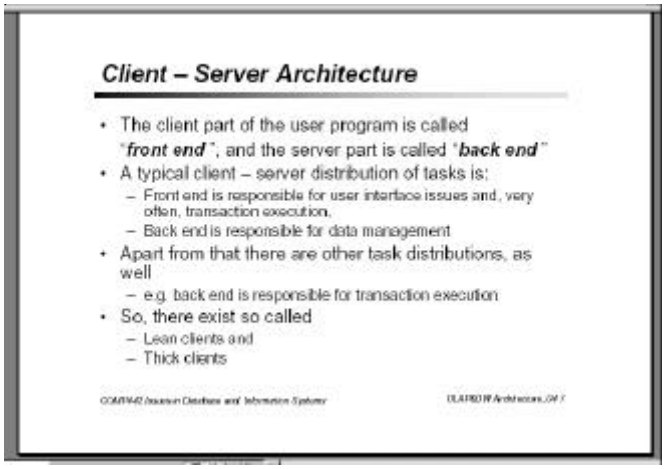
  The system should be able to hold all data needed by the applications. Data sparsity should be handled in an efficient manner.

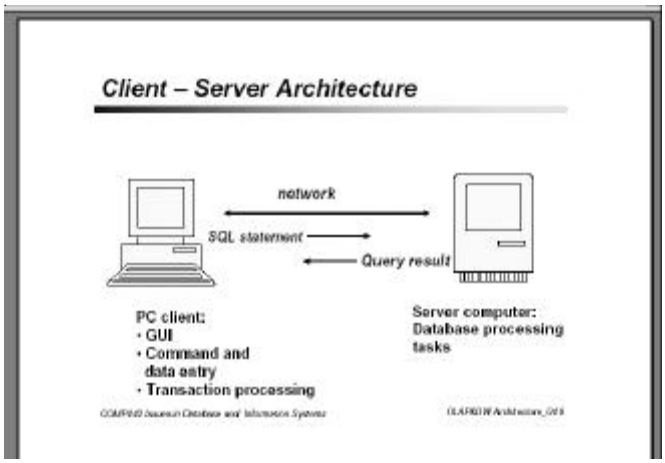












Exercise

1. Write short notes on:
   - OLTP
   - Consistent Reporting Performance
   - OLAP
   - FASMI
2. Illustrate with the help of a diagram the Client-Server Architecture in brief.
3. Explain the importance of Multidimensional Data Model in OLAP.
4. Correctly contrast the difference between Multidimensional versus Multirelational OLAP.
5. Discuss in brief OLAP guidelines suggested by C.J.Codd.

Notes

LESSON 31
OLAP OPERATIONS

Structure

- Objective
- Introduction
- OLAP Operations
- Lattice of cubes, slice and dice operations
- Relational representation of the data cube
- Database management systems (DBMS), Online Analytical Processing (OLAP) and Data Mining
- Example of DBMS, OLAP and Data Mining: Weather data

Objective

The main objective of this lesson is to introduce you with various OLAP Operations

Introduction

In today's fast-paced, information-driven economy, organizations heavily rely on real-time business information to make accurate decisions. The number of individuals within an enterprise who have a need to perform more sophisticated analysis is growing. With their ever-increasing requirements for data manipulating tools, end users cannot be already satisfied with flat grids and a fixed set of parameters for query execution.

OLAP is the best technology that empowers users with complete ease in manipulating their data. The very moment you replace your common grid with an OLAP interface users will be able independently to perform various ad-hoc queries, arbitrarily filter data, rotate a table, drill down, get desired summaries, and rank. From users' standpoint, the information system equipped with OLAP-tool gains a new quality; helps not only get information but also summarize and analyze it.

From the developer's point of view, OLAP is an elegant way to avoid thankless and tedious programming of multiple on-line and printed reports.

OLAP Operations

Assume we want to change the level that we selected for the temperature hierarchy to the intermediate level (hot, mild, cool). To do this we have to group columns and add up the values according to the concept hierarchy. This operation is called **roll-up**, and in this particular case it produces the following cube.

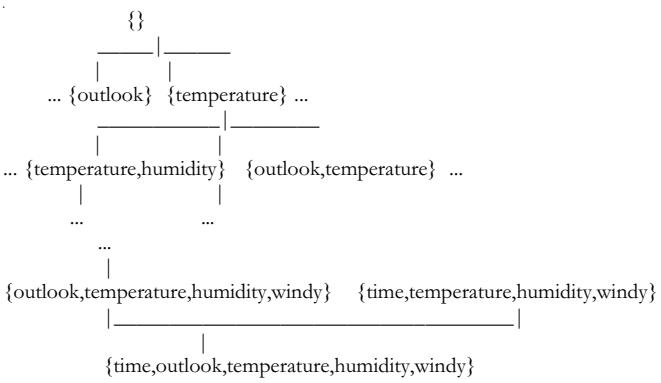
	cool	**mild**	**hot**
week 1	2	1	1
week 2	1	3	1

In other words, climbing up the concept hierarchy produces roll-up's. Inversely, climbing down the concept hierarchy expands the table and is called **drill-down**. For example, the drill down of the above data cube over the time dimension produces the following:

	cool	**mild**	**hot**
day 1	0	0	0
day 2	0	0	0
day 3	0	0	1
day 4	0	1	0
day 5	1	0	0
day 6	0	0	0
day 7	1	0	0
day 8	0	0	0
day 9	1	0	0
day 10	0	1	0
day 11	0	1	0
day 12	0	1	0
day 13	0	0	1
day 14	0	0	0

Lattice of Cubes, Slice and Dice Operations

The number of dimensions defines the total number of data cubes that can be created. Actually this is the number of elements in the **power set** of the set of attributes. Generally if we have a set of **N attributes**, the power set of this set will have $\mathbf{2^N}$ **elements**. The elements of the power set form a **lattice**. This is an algebraic structure that can be generated by applying intersection to all subsets of the given set. It has a **bottom element** - the set itself and a **top element** - the empty set. Here is a part of the lattice of cubes for the weather data cube.



In the above terms the selection of dimensions actually means selection of a cube, i.e. an element of the above lattice.

There are two other OLAP operations that are related to the selection of a cube - slice and dice. **Slice** performs a selection on one dimension of the given cube, thus resulting in a subcube. For example, if we make the selection **(temperature=cool)** we will reduce the dimensions of the cube from two to one, resulting in just a single column from the table above.

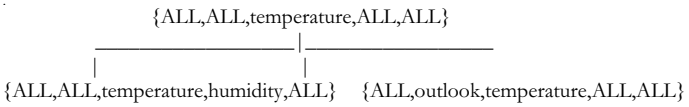
	cool
day 1	0
day 2	0
day 3	0
day 4	0
day 5	1
day 6	0
day 7	1
day 8	0
day 9	1
day 10	0
day 11	0
day 12	0
day 13	0
day 14	0

The **dice** operation works similarly and performs a selection on two or more dimensions. For example, applying the selection **(time = day 3 OR time = day 4) AND (temperature = cool OR temperature = hot)** to the original cube we get the following subcube (still two-dimensional):

	cool	**hot**
day 3	0	1
day 4	0	0

Relational Representation of the Data Cube

The use of the lattice of cubes and concept hierarchies gives us a great flexibility to represent and manipulate data cubes. However, a still open question is how to implement all this. An interesting approach to this based on a simple extension of standard relational representation used in DBMS is proposed by Jim Gray and collaborators. The basic idea is to use the value **ALL** as a legitimate value in the relational tables. Thus, **ALL** will represent the set of all values aggregated over the corresponding dimension. By using **ALL** we can also represent the lattice of cubes, where instead of dropping a dimension when intersecting two subsets, we will replace it with **ALL**. Then all cubes will have the same number of dimensions, where their values will be extended with the val,ue **ALL**. For example, a part of the above shown lattice will now look like this:



Using this technique the whole data cube can be represented as a single relational table as follows (we use higher levels in the concept hierarchies and omit some rows for brevity):

Time	Outlook	Temperature	Humidity	Windy	Play
week 1	sunny	cool	normal	true	0
week 1	sunny	cool	normal	false	0
week 1	sunny	cool	normal	ALL	0
week 1	sunny	cool	high	true	0
week 1	sunny	cool	high	false	0
week 1	sunny	cool	high	ALL	0
week 1	sunny	cool	ALL	true	0
week 1	sunny	cool	ALL	false	0
week 1	sunny	cool	ALL	ALL	0
week 1	sunny	mild	normal	true	0
...	...	...	...	...	...
week 1	overcast	ALL	ALL	ALL	2
week 1	ALL	ALL	ALL	ALL	4
week 2	sunny	cool	normal	true	0
week 2	sunny	cool	normal	false	1
week 2	sunny	cool	normal	ALL	1
week 2	sunny	cool	high	true	0
...	...	...	...	...	...
ALL	ALL	ALL	high	ALL	3
ALL	ALL	ALL	ALL	true	3
ALL	ALL	ALL	ALL	false	6
ALL	ALL	ALL	ALL	ALL	9

The above table allows us to use an unified approach to implement all OLAP operations - they all can me implemented just by selecting proper rows. For example, the following cube, can be extracted from the table by selecting the rows that match the pattern (*, ALL, *, ALL, ALL), where * matches all legitimate values for the corresponding dimension except for ALL.

	cool	mild	hot
week 1	2	1	1
week 2	1	3	1

Database Management Systems (DBMS), Online Analytical Processing (OLAP) and Data Mining

Area	DBMS	OLAP	Data Mining
Task	Extraction of detailed and summary data	Summaries, trends and forecasts	Knowledge discovery of hidden patterns and insights
Type of result	Information	Analysis	Insight and Prediction
Method	Deduction (Ask the question, verify with data)	Multidimensional data modeling, Aggregation, Statistics	Induction (Build the model, apply it to new data, get the result)
Example question	Who purchased mutual funds in the last 3 years?	What is the average income of mutual fund buyers by region by year?	Who will buy a mutual fund in the next 6 months and why?

Example of DBMS, OLAP and Data Mining: Weather Data

Assume we have made a record of the weather conditions during a two-week period, along with the decisions of a tennis player whether or not to play tennis on each particular day. Thus we have generated **tuples** (or examples, instances) consisting of values of four **independent variables** (outlook, temperature, humidity, windy) and one **dependent variable** (play). See the textbook for a detailed description.

DBMS

Consider our data stored in a relational table as follows:

Day	Outlook	Temperature	Humidity	Windy	Play
1	Sunny	85	85	false	no
2	Sunny	80	90	true	no
3	overcast	83	86	false	yes
4	Rainy	70	96	false	yes
5	Rainy	68	80	false	yes
6	Rainy	65	70	true	no
7	overcast	64	65	true	yes
8	Sunny	72	95	false	no
9	Sunny	69	70	false	yes
10	Rainy	75	80	false	yes
11	Sunny	75	70	true	yes
12	overcast	72	90	true	yes
13	overcast	81	75	false	yes
14	Rainy	71	91	true	no

By querying a DBMS containing the above table we may answer questions like:

- What was the temperature in the sunny days? {85, 80, 72, 69, 75}
- Which days the humidity was less than 75? {6, 7, 9, 11}
- Which days the temperature was greater than 70? {1, 2, 3, 8, 10, 11, 12, 13, 14}
- Which days the temperature was greater than 70 and the humidity was less than 75? The intersection of the above two: {11}

OLAP

Using OLAP we can create a **Multidimensional Model** of our data (**Data Cube**). For example using the dimensions: **time**, **outlook** and **play** we can create the following model.

9 / 5	sunny	rainy	overcast
Week 1	0 / 2	2 / 1	2 / 0
Week 2	2 / 1	1 / 1	2 / 0

Obviously here **time** represents the days grouped in weeks (week 1 - days 1, 2, 3, 4, 5, 6, 7; week 2 - days 8, 9, 10, 11, 12, 13, 14) over the vertical axis. The outlook is shown along the horizontal axis and the third dimension **play** is shown in each individual cell as a pair of values corresponding to the two values along this dimension - **yes / no**. Thus in the upper left corner of the cube we have the total over all weeks and all outlook values.

By observing the data cube we can easily identify some important properties of the data, find regularities or patterns. For example, the third column clearly shows that if the outlook is overcast the play attribute is always yes. This may be put as a rule:

if outlook = overcast then play = yes

We may now apply "Drill-down" to our data cube over the time dimension. This assumes the existence of a **concept hierarchy** for this attribute. We can show this as a horizontal tree as follows:

- time
  - week 1
    - day 1
    - day 2
    - day 3
    - day 4
    - day 5
    - day 6
    - day 7
  - week 2
    - day 8
    - day 9
    - day 10
    - day 11
    - day 12
    - day 13

- day 14
- day 15

The drill-down operation is based on climbing down the concept hierarchy, so that we get the following data cube:

9 / 5	Sunny	Rainy	Overcast
1	0 / 1	0 / 0	0 / 0
2	0 / 1	0 / 0	0 / 0
3	0 / 0	0 / 0	1 / 0
4	0 / 0	1 / 0	0 / 0
5	0 / 0	1 / 0	0 / 0
6	0 / 0	0 / 1	0 / 0
7	0 / 0	0 / 0	1 / 0
8	0 / 1	0 / 0	0 / 0
9	1 / 0	0 / 0	0 / 0
10	0 / 0	1 / 0	0 / 0
11	1 / 0	0 / 0	0 / 0
12	0 / 0	0 / 0	1 / 0
13	0 / 0	0 / 0	1 / 0
14	0 / 0	0 / 1	0 / 0

The reverse of drill-down (called roll-up) applied to this data cube results in the previous cube with two values (week 1 and week 2) along the time dimension.

Data Mining

By applying various Data Mining techniques we can find associations and regularities in our data, extract knowledge in the forms of rules, decision trees etc., or just predict the value of the dependent variable (play) in new situations (tuples). Here are some examples (all produced by Weka):

Mining Association Rules

To find associations in our data we first **discretize the numeric attributes** (a part of the data **pre-processing stage** in data mining). Thus we group the temperature values in three intervals (hot, mild, cool) and humidity values in two (high, normal) and substitute the values in data with the corresponding names. Then we apply the **Apriori algorithm** and get the following association rules:

1. Humidity=normal windy=false 4 ==> play=yes (4, 1)
2. Temperature=cool 4 ==> humidity=normal (4, 1)
3. Outlook=overcast 4 ==> play=yes (4, 1)
4. Temperature=cool play=yes 3 ==> humidity=normal(3, 1)
5. Outlook=rainy windy=false 3 ==> play=yes (3, 1)
6. Outlook=rainy play=yes 3 ==> windy=false (3, 1)
7. Outlook=sunny humidity=high 3 ==> play=no (3, 1)
8. Outlook=sunny play=no 3 ==> humidity=high (3, 1)
9. Temperature=cool windy=false 2 ==> humidity=normal play=yes (2, 1)
10. Temperature=cool humidity=normal windy=false 2 ==> play=yes (2, 1)

These rules show some attribute values sets (the so called **item sets**) that appear frequently in the data. The numbers after each rule show the **support** (the number of occurrences of the item set in the data) and the **confidence** (accuracy) of the rule. Interestingly, rule 3 is the same as the one that we produced by observing the data cube.

Classification by Decision Trees and Rules

Using the **ID3** algorithm we can produce the following decision tree (shown as a horizontal tree):

- outlook = sunny
  - humidity = high: no
  - humidity = normal: yes
- outlook = overcast: yes
- outlook = rainy
  - windy = true: no
  - windy = false: yes

The decision tree consists of decision nodes that test the values of their corresponding attribute. Each value of this attribute leads to a subtree and so on, until the leaves of the tree are reached. They determine the value of the dependent variable. Using a decision tree we can classify new tuples (not used to generate the tree). For example, according to the above tree the tuple {sunny, mild, normal, false} will be classified under play=yes.

A decision trees can be represented as a set of rules, where each rule represents a path through the tree from the root to a leaf. Other Data Mining techniques can produce rules directly. For example the **Prism** algorithm available in Weka generates the following rules.

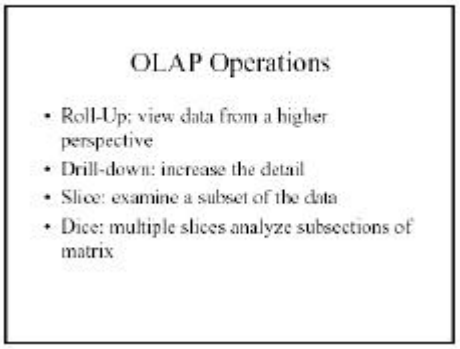
If outlook = overcast then yes
If humidity = normal and windy = false then yes
If temperature = mild and humidity = normal then yes
If outlook = rainy and windy = false then yes
If outlook = sunny and humidity = high then no
If outlook = rainy and windy = true then no

Prediction Methods

Data Mining offers techniques to predict the value of the dependent variable directly without first generating a model. One of the most popular approaches for this purpose is based of statistical methods. It uses the Bayes rule to predict the probability of each value of the dependent variable given the values of the independent variables. For example, applying Bayes to the new tuple discussed above we get:

P(play=yes | outlook=sunny, temperature=mild, humidity=normal, windy=false) = 0.8
P(play=no | outlook=sunny, temperature=mild, humidity=normal, windy=false) = 0.2

Then obviously the predicted value must be "yes".



Exercise

1. Write short notes on:
   - o Relational representation of the data cube
   - o Mining Association Rules
   - o Slice and dice operations
2. Explain in brief various OLAP Operations.
3. Differentiate between Database management systems (DBMS), Online Analytical Processing (OLAP) and Data Mining
4. Explain the difference between DBMS, OLAP and Data Mining with related example.

Notes

LESSON 32
CATEGORIZATION OF OLAP TOOLS CONCEPTS USED IN MOLAP/ ROLAP

Summary

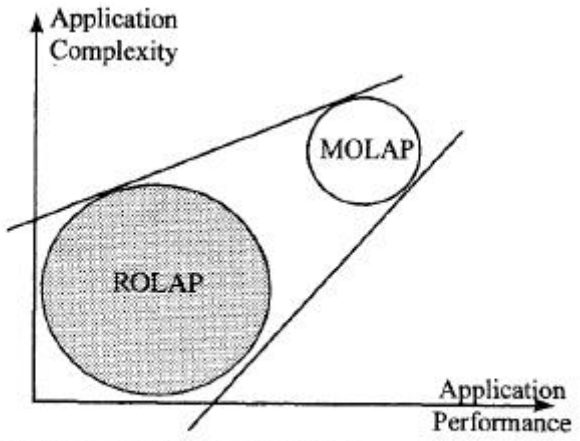
- Objective
- Categorization of OLAP Tools
- MOLAP
- ROLAP
- Managed query environment (MQE)
- Cognos PowerPlay
- Pilot Software
- OLAP Tools and the Internet

Objective

The objective of this lesson is to introduce you with various OLAP Tools

Categorization of OLAP Tools

On-line analytical processing (OLAP) tools are based on the concepts of multi-dimensional databases and allow a sophisti-cated user to analyze the data using elaborate, multidimensional, complex views. Typical business applica-tions for these tools include product performance and profitability, effective-ness of a sales program or a marketing campaign, sales forecasting, and capacity planning. These tools assume that the data is organized in a multidi-mensional model, which is supported by a special multidimensional database (MDDB) or by a relational database designed to enable multidimensional prop-erties (e.g., star schema,). A chart comparing capabilities of these two classes of OLAP tools is shown in Fig. 32.1.



The area of the circles indicate the data size

Fig 32.1 OLAP style comparison.

MOLAP

Traditionally, these products utilized specialized data structures [i.e., multi-dimensional database management systems (MDDBMSs)] to organize, navi-gate, and analyze data, typically in an aggregated form, and traditionally required a tight coupling with the application layer and presentation layer. There recently has been a quick movement by MOLAP vendors to segregate the OLAP through the use of published application programming interfaces (APIs). Still, there remains the need to store the data in a way similar to the way in which it will be utilized, to enhance the performance and provide a degree of predictability for complex analysis queries. Data structures use array technology and, in most cases, provide improved storage techniques to minimize the disk space requirements through sparse data management. This architecture enables excellent performance when the data is utilized as designed, and predict-able application response times for applications address-ing a narrow breadth of data for a specific DSS requirement. In addition, some products treat time as a special dimension (e.g., Pilot Software's Analysis Server), enhancing their ability to perform time series analysis. Other products provide strong analytical capabilities (e.g., Oracle's Express Server) built into the database.

Applications requiring iterative and comprehensive time series analysis of trends are well suited for MOLAP technology (e.g., financial analysis and bud-geting). Examples include Arbor Software's Essbase, Oracle's Express Server, Pilot Software's Lightship Server, Sinper's TM/l, Planning Sciences' Gentium, and Kenan Technology's Multiway.

Several challenges face users considering the implementation of applications with MOLAP products. First, there are limitations in the ability of data struc-tures to support multiple subject areas of data (a common trait of many strate-gic DSS applica-tions) and the detail data required by many analysis applications. This has begun to be addressed in some products, utilizing rudi-mentary "reach through" mechanisms that enable the MOLAP tools to access detail data maintained in an RDBMS (as shown in Fig. 32.2). There are also limitations in the way data can be navigated and analyzed, because the data is structured around the navigation and analysis requirements known at the time the data structures are built. When the navigation or dimension requirements change, the data structures may need to be physically reorganized to optimally support the new requirements. This problem is similar in nature to the older hierarchical and network DBMSs (e.g., IMS, IDMS), where different sets of data had to be created for each application that used the data in a manner different from the way the data was originally maintained. Finally, MOLAP products require a different set of skills and tools for the database administrator to build and maintain the database, thus increasing the cost and complexity of support.

To address this particular issue, some vendors significantly enhanced their reach-through capabilities. These hybrid solutions have as their primary char-acteristic the integration of

specialized multidimensional data storage with RDBMS technology, providing users with a facility that tightly "couples" the multidimensional data structures (MDDSs) with data maintained in an RDBMS (see Fig. 32.2, left). This allows the MDDSs to dynamically obtain detail data maintained in an RDBMS, when the application reaches the bottom of the multidimensional cells during drill-down analysis.

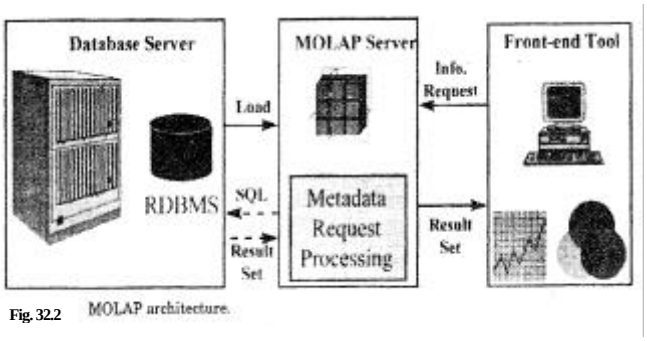


Fig. 32.2 MOLAP architecture.

This may deliver the best of both worlds, MOLAP and ROLAP. This approach 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 calcula-tions and building aggregations during application initialization. These struc-tures can be automatically refreshed at predetermined intervals established by an administrator.

ROLAP

This segment constitutes the fastest-growing style of OLAP technology, with new vendors (e.g., Sagent Technology) entering the market at an accelerating pace. Products in this group have been engineered from the beginning to support RDBMS products directly through a dictionary layer of metadata, bypassing any requirement for creating a static multidimensional data structure (see Fig. 32.3). This enables multiple multidimensional views of the two-dimensional relational tables to be created without the need to structure the data around the desired view. Finally, some of the products in this segment have developed strong SQL- generation engines to support the complexity of multidimensional analysis. This includes the creation of multiple SQL statements to handle user requests, being "RDBMS-aware," and providing the capability to generate the SQL based on the optimizer of the DBMS engine. While flexibility is an attractive feature of ROLAP products, there are products in this segment that recommend, or require, the use of highly denormalized database designs (e.g., star schema).

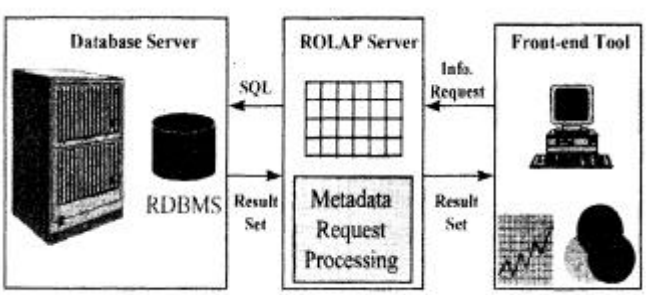


Fig.32.3 ROLAP architecture.

The ROLAP tools are undergoing some technology realignment. This shift in technology emphasis is coming in two forms. First is the movement toward pure middleware technology that provides facilities to simplify development of multidimensional applications. Second, there continues further blurring of the lines that delineate ROLAP and hybrid-OLAP products. Vendors of ROLAP tools and RDBMS products look to provide an option to create multi-dimensional, persistent structures, with facilities to assist in the administration of these structures. Examples include Information Advantage (Axsys), MicroStrategy (DSS AgentIDSS Server), Platinum/Prodea Software (Beacon), Informix/Stanford Technology Group (Metacube), and Sybase (HighGate Project).

Managed Query Environment (MQE)

This style of OLAP, which is beginning to see increased activity, provides users with the ability to perform limited analysis capability, either directly against RDBMS products, or by leveraging an intermediate MOLAP server (see Fig. 32.4). Some products (e.g., Andyne'sPablo) that have a heritage in ad hoc query have developed features to provide "datacube" and "slice and dice" analysis capabilities. This is achieved by first developing a query to select data from the DBMS, which then delivers the requested data to the desktop, where it is placed into a datacube. This datacube can be stored and maintained locally, to reduce the overhead required to create the struCture each time the query is executed. Once the data is in the datacube, users can perform multidimen-sional analysis (i.e., slice, dice, and pivot operations) against it. Alternatively, these tools can work with MOLAP servers, and the data from the relational DBMS can be delivered to the MOLAP server, and from there to the desktop.

The simplicity of the installation and administration of such products makes them particularly attractive to organizations looking to provide seasoned users with more sophisticated analysis capabilities, without the significant cost and maintenance of more complex products. With all the ease of installation and administration that accompanies the desktop OLAP products, most of these tools require the datacube to be built and maintained on the desktop or a sep-arate server.

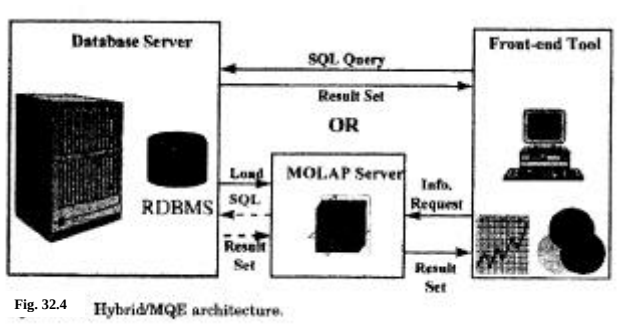


Fig. 32.4 Hybrid/MQE architecture.

With metadata definitions that assist users in retrieving the correct set of data that makes up the datacube, this method causes a plethora of data redundancy and strain to most network infrastructures that support many users. Although this mechanism allows for the flexibility of each user to build a custom datacube, the lack of data consistency among users, and the rel-atively small amount of data that can be efficiently maintained are significant challenges facing tool administrators.

Examples include Cognos Software's PowerPlay, Andyne Software's Pablo, Business Objects' Mercury Project, Dimensional Insight's CrossTarget, and Speedware's Media.

OLAP tools provide an intuitive way to view corporate data. These tools aggre-gate data along common business subjects or dimensions and then let users navigate through the hierarchies, and dimensions with the click of a mouse but-ton. Users can drill down, across, or up levels in each dimension or pivot and swap out dimensions to change their view of the data.

Some tools, such as Arbor Software Corp.'s Essbase and Oracle's Express, pre aggregate data in special multidimensional databases. Other tools work directly against relational data and aggregate data on the fly, such as Micro-Strategy, Inc.'s DSS Agent or Information Advantage, Inc.'s DecisionSuite. Some tools process OLAP data on the desktop instead of a server. Desktop OLAP tools include Cognos' PowerPlay, Brio Technology, Inc.'s HrioQuery, Plan-ning Sciences, Inc.'s Gentium, and Andyne's Pablo. Many of the differences between OLAP tools are fading. Vendors are rearchitecting their products to give users greater control over the tradeoff between flexibility and perfor-mance that is inherent in OLAP tools. Many vendors are rewriting pieces of their products in Java.

Database vendors eventually might be the largest OLAP providers. The lead-ing database vendors plan to incorporate OLAP functionality in their database kernels. Oracle, Informix Software, Inc., and-most recently-Microsoft have taken the first step toward this end by acquiring OLAP vendors **(IRI** Software, Stanford Technology Group, and Panorama, respectively.)

Red Brick Systellls' Red Brick Warehouse has always supported SQL exten-sions that perform simple OLAP functions, such as rank, sorts, and moving averages. Red Brick Warehouse 5.0 also supports data mining algorithms.

Cognos PowerPlay

PowerPlay from Cognos is a mature and popular software tool for multidimen-sional analysis of corporate data. PowerPlay can be characterized as an MQE tool that can leverage corporate investment in the relational database technology to provide multidimensional access to enterprise data, at the same time proving robustness, scalability, and administrative control.

Cognos PowerPlay is an open OLAP solution that can interoperate with a wide variety of third-party software tools, databases, and applications. The analytical data used by PowerPlay is stored in multidimensional data sets called *PowerCubes*. Cognos' client/server architecture allows for the Power-Cubes to be stored on the Cognos universal client or on a server. PowerPlay offers a single universal client for OLAP servers that supports PowerCubes located locally, on the LAN, or (optionally) inside popular relational databases. In addition to the fast installation and deployment capabilities, PowerPlay pro-vides a high level of usability with a familiar Windows interface, high perfor-mance, scalability, and relatively low cost of ownership.

Specifically, starting with version 5, Cognos PowerPlay client offers

- Support for enterprise-size data sets (PowerCubes) of 20+ million records,100,000 categories, and 100 measures
- A drill-through capability for queries from Cognos Impromptu
- Powerful 3-D charting capabilities with background and rotation control for advanced users
- Scatter charts that let users show data across two measures, allowing easy comparison of budget to actual values
- Linked displays that give users multiple views of the same data in a report
- Full support for OLE2 Automation, as both a client and a server
- Formatting features for financial reports: brackets for negative numbers,
- Single and double underlining, and reverse sign for expenses
- Faster and easier ranking of data
- A "home" button that automatically resets the dimension line to the top level
- Unlimited undo levels and customizable toolbars
- An enhanced PowerPlay Portfolio that lets users build graphical, interactive, EIS-type briefing books from PowerPlay reports; Impromptu reports; word processing, spreadsheet, or presentation documents; or any other documents or reports
- A 32-bit architecture for Windows NT, Windows 95, and Windows 3.1
- Access to third-party OLAP tools including direct native access to Arbor's Essbase and Oracle's Express multidimensional databases
- PowerCube creation and access within in existing relational databases such as ORACLE, SYBASE, or Microsoft SQL Server right inside the data warehouse.
- PowerCube creation scheduled for off-peak processing, or sequential to other processes
- Advanced security control by dimension, category, and measure-on the client, the server, or both

- Remote analysis where users pull subsets of information from the server down to the client
- Complete integration with relational database security and data manage-ment features
- An open API through OLE Automation, allowing both server- and client--based PowerCubes to be accessed by Visual Basic applications, spreadsheets, and other third-party tools and applications

PowerPlay Administrator. As was mentioned above, Power Play's capabilities include drill-to-detail using Impromptu. Also, cubes can be built using data from multiple data sources. For the administrators who are responsible for cre-ating multidimensional cubes, new capabilities allow them to populate these PowerCubes inside popular relational databases, and to do the processing off the desktop and on UNIX servers. To provide a robust administration capabil-ities, Cognos offers a companion tool-PowerPlay Administrator, which is available in Database and Server editions.

In Power Play Administrator Database edition, the administrator would con-tinue to model the cube and run the population of the cube process (called *Transform)* on the client platform. The advantage is that data from multiple sources can now be used to generate a PointerCube for the client, and the actual PowerCube can be inside a relational database. This means that exist-ing database management tools and the database administrator can be used to manage the business data, and a single delivery mechanism can be employed for both application and OLAP processing. A sophisticated security model is provided which in effect creates a "master" cube to service a variety of users. This is defined and controlled through the Authenticator, also included with PowerPlay.

The Administrator Server edition of PowerPlay lets users process the popu-lation of the cube on a UNIX platform. An administrator uses client Trans-former to create a model, and moves it to the UNIX server using the supplied software component called *PowerGrid.* The server Transformer, once triggered, will create the PowerCube, and the only prerequisite is that all data sources be accessible. Once completed, the resulting PowerCube (or PointerCube if the multidimensional database is placed inside an RDBMS) is copied or trans-ferred to the client platform for subsequent PowerPlay analysis by the user. The Authenticator can be used to establish user classes and access security, and can also be used to redirect cube access since all database passwords and locations can be known to the Authenticator.

PowerPlay supports clients on Windows 3.1, Windows 95, and Windows NT. Administrator Database and Server editions execute on HP/UX, IBM AIX, and Sun Solaris, and support PowerCubes in ORACLE 7, SYBASE SQL Server, and Microsoft SQL Server.

Pilot Software

Pilot Software offers the Pilot Decision Support Suite of tools from a high speed multidimensional database (MOLAP), Data Warehouse integration (ROLAP), data mining, and a diverse set of customizable business applications targeted after sales and marketing professionals. The following products are at the core of Pilot Software's offering:

Pilot Analysis Server. A full-function multidimensional database with high-speed consolidation, graphical user interface (Pilot Model Builder), and expert-level interface. .The latest version includes relational integration of the multidimensional model with relational data stores, thus allowing the user the choice between high-speed access of a multidimensional database or on-the-fly (ROLAP) access of detail data stored directly in the data ware-house or data mart.

Pilot Link. A database connectivity tool that includes ODBC connectivity and high-speed connectivity via specialized drivers to the most popular rela-tional database platforms. A graphical user interface allows the user seam-less and easy access to a wide variety of distributed databases.

Pilot Designer. An application design environment specifically created to enable rapid development of OLAP applications.

Pilot Desktop. A collection of applications that allow the end user easy nav-igation and visualization of the multidimensional database.

Pilot Sales and Marketing Analysis; Library. A collection of sophisticated applications designed for the Sales and Marketing business end user (includ-ing 80/20 Pareto analysis, time-based ranking, BCG quadrant analysis, trendline, and statistical forecasting). The applications can be modified and tailored to meet individual needs fGr particular customers.

Pilot Discovery Server. A predictive data mining tool that embeds directly into the relational database and does not require the user to copy or trans-form the data. The data mining results are stored with metadata into the data warehouse as a predictive segmentation and are embedded into the multidimensional model as a dimension. The discovery server contains a graphical user interface called Pilot Discovery Server Launch, which eases building data mining models.

Pilot Marketing Intelligence Library. Currently there are two applications for exposing the value of the data mining results. One allows the user to graphically view the predictive segmentation and rules that describe the prediction; the other allows for profi1lloss and return on investment analysis for the data mining results.

Pilot Internet Publisher. A tool that easily allows users to access their Pilot multidimensional database via browsers on the Internet or intranets.

Some of the distinguishing features of Pilot's product offering include the over-all complete solution from powerful OLAP engine and data mining engine to their customizable business applications. Within their OLAP offering, these are some of the key features:

Time intelligence. The Pilot Analysis Server has a number of features to support time as a special dimension. Among these are the ability to process data on the fly to convert from the native periodicity (e.g., the data was col-lected weekly) to the periodicity preferred by the customer viewing the data (e.g., view the data monthly). This feature is accomplished via special optimized structures within the multidimensional database.

Embedded data mining. The Pilot Analysis Server is the first product to integrate predictive data mining (as it is described in this book) with the multidimensional database model. This allows the user to benefit from not only the predictive power of data mining but also the descriptive and ana-lytical power of multidimensional navigation.

Multidimensional database compression. In addition to the compression of sparsity (the removal of cells in the multidi-mensional database which have no value), Pilot Analysis Server also has special code for compressing data values over time. A new feature called a "dynamic dimension" allows some dimensions to be calculated on the fly when they are attributes of an exist-ing dimension. This allows the database to be much smaller and still provide fast access. Dynamic variables, which are also calculated on the fly, are also available to further decrease the total size of the database and thus also decrease the time for consolidation of the database.

Relational Integration. Pilot allows for a seamless integration of both MOLAP and ROLAP to provide the user with either the speed of MOLAP or the more space-efficient ROLAP. The users interface with the system by defining the multidimen-sional model or view that they prefer, and the sys-tem self-optimizes the queries into precalculated MOLAP storage or directly in the relation data store depending on the usage pattern and the time/space tradeoff preferences of the end user.

OLAP Tools and the Internet

The two most pervasive themes in computing have been the Internet/WWW and data warehousing. From a marketing perspective, a marriage of these two giant technologies is a natural and unavoidable event. The reason for this trend is simple; the compelling advantages in using the Web for access are _magnified even further in a data warehouse. Indeed:

The Internet is a virtually free resource, which provides a universal connec-tivity within and between companies. The Web eases complex administrative tasks of managing distributed envi-ronments.

The Web allows companies to store and manage both data and applications on servers that can be centrally managed, main-tained, and updated, thus eliminating problems with software and data currency.

For these and other reasons, the Web is a perfect medium for decision support. Let's look at the general features of the Web-enabled data access.

The first-generation Web sites used a static distribution model, in which clients access static HTML pages via Web browsers. In this model, the deci-sion support reports were stored as HTML documents and delivered to users on request. Clearly, this model has some serious deficiencies, including inability to provide Web clients with interactive analytical capabilities such as drill-down.

The second-generation Web sites support interactive database queries by uti-lizing a multitiered architecture in which a Web client submits a query in the form of HTML-encoded request to a Web server, which in turn transforms the request for structured data into a CGI (Common Gateway Interface) script, or a script written to a proprietary Web-server API (Le., Netscape Server API, or NSAPI). The gateway submits SQL queries to the database, receives the results, translates them into HTML, and sends the pages to the requester (see Fig. 32.5). Requests for the unstructured data (e.g., images, other HTML documents, etc.) can be sent directly to the unstructured data store.

The emerging third-generation Web sites replace HTML gateways with Web--based application servers. These servers can download Java applets or ActiveX applications that execute on clients, or interact with corresponding applets running on servers-servlets. The third-generation Web servers pro-vide users with all the capabilities of existing decision-support applications without requiring them to load any client software except a Web browser.

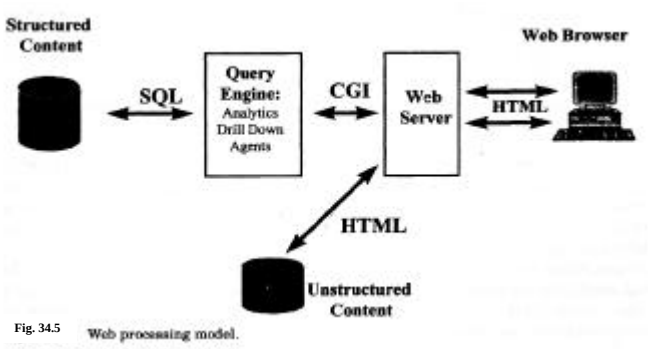


Fig. 34.5 Web processing model.

Not surprisingly, vendors of decision support applications, especially query, reporting, and OLAF tools, are rapidly convert-ing their tools to work on the Web. Vendor approaches for deploying tools on the Web include

HTML publishing. This approach involves transforming an output of a query into the HTML page that can be downloaded into a browser. This approach does not support interactive access to data or reports.

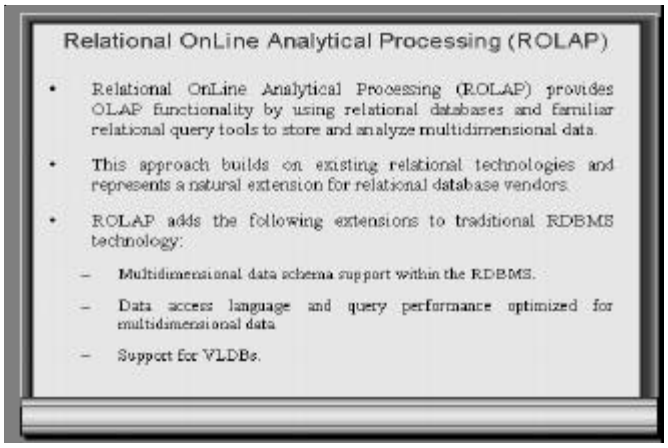
Helper applications. In this approach, a tool is configured as a helper appli-cation that resides within a browser. This is the case of a "fat" client, in which, once the data is downloaded, users can take advantage of all capabil-ities of the tool to analyze data. However, maintaining these helper applica-tions becomes another task for system administrators.

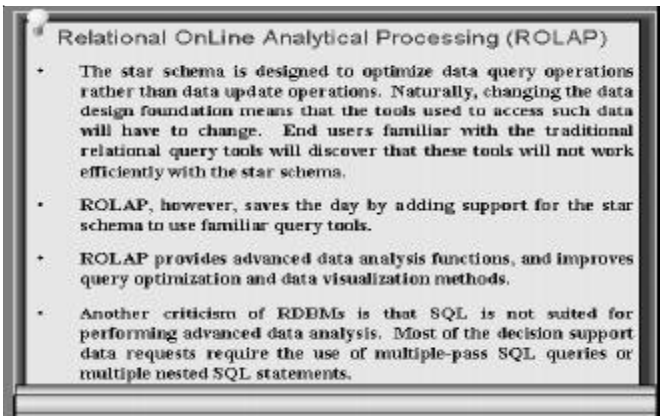
Plug-ins. A variation on the previous approach, plug-ins are helper appli-cations that are downloaded from the Web server prior to their initial use. Since the plug-ins are downloaded from the server, their normal administra-tion and installation tasks are significantly reduced. However, typically plug-ins are browser-specific, and may not run on all platforms or with all browsers. Also, as browsers get updated, this plug-ins may have to be upgraded as well, creating additional administration workload.

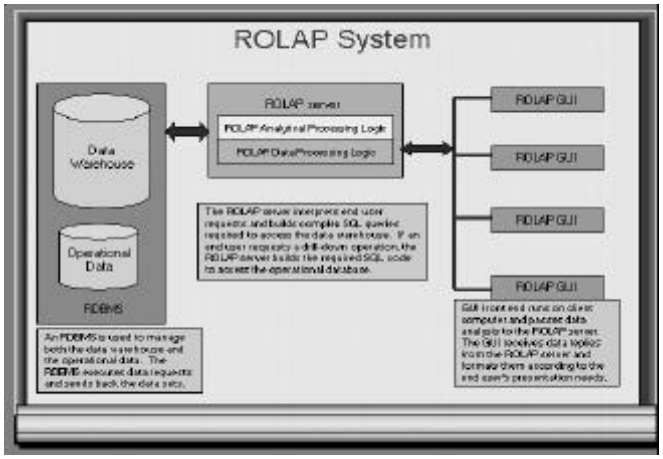
Server-centric components. In this approach the vendor rebuilds a desktop tool as a server component, or creates a new server component that can be integrated with the Web via a Web gateway (e.g., CGI or NSAFI scripts).

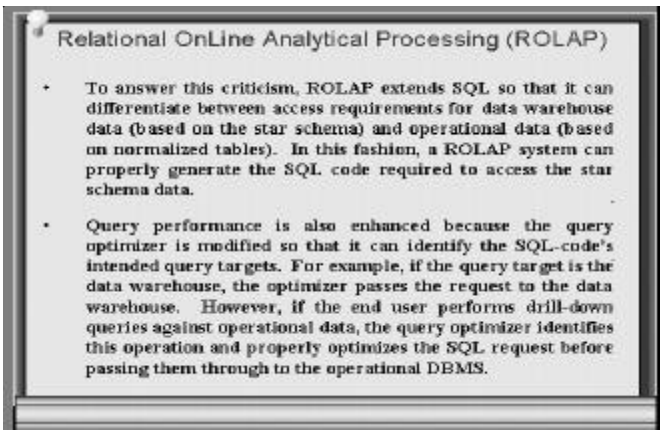
Java and ActiveX applications. This approach is for a vendor to redevelop all or portions of its tool in Java or ActiveX. The result is a true "thin" client model. There are advantages and disadvantages to both, but this approach appears to be one of the most promising and flexible.

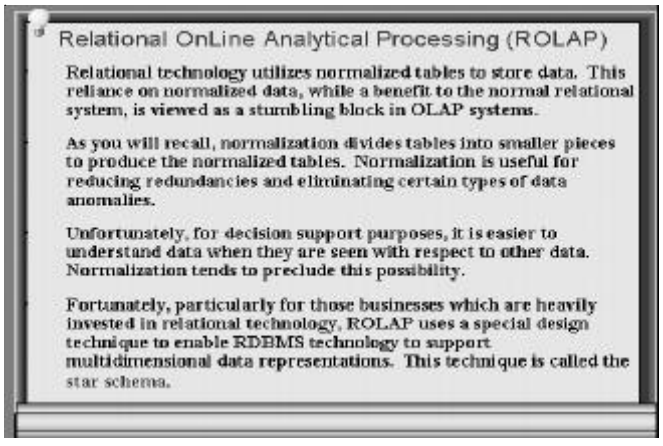
The remainder of this section looks at several OLAF tools from a perspective of Internet/Web implementations.

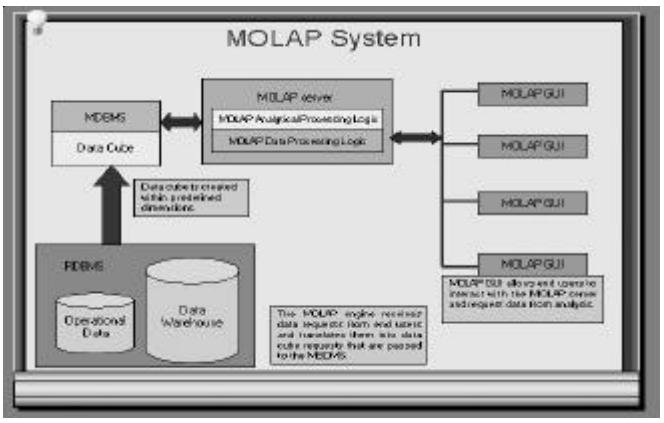






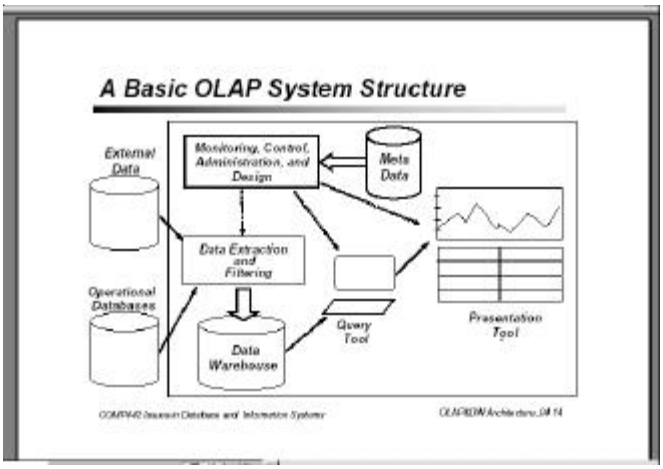


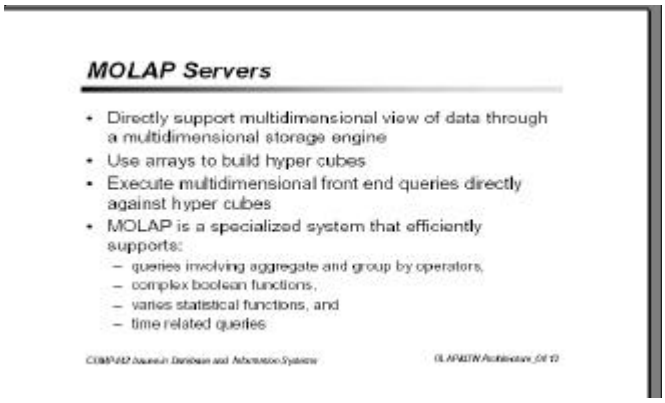


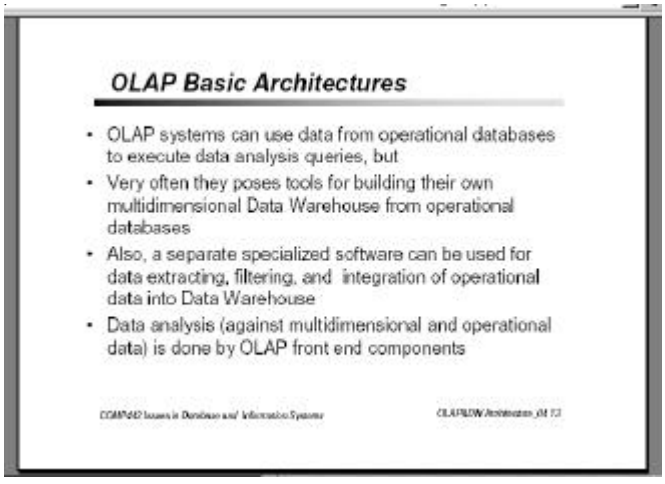


Relational vs. Multidimensional OLAP

Characteristic	ROLAP	MOLAP
Schema	Uses star schema. Additional dimensions added dynamically	Uses data cubes Additional dimensions require re-creation of the data cube
Database Size	Medium to large	Small to medium
Architecture	Client/server Standards based Open	Client/server Proprietary
Access	Supports ad hoc requests Unlimited dimensions	Limited to pre-defined dimensions
Resources	High	Very high
Flexibility	High	Low
Scalability	High	Low
Speed	Good with small data data sets; average for medium to large data sets	Faster for small to medium data sets; average for large data sets.







Exercise

1. Write short notes on:
   - Managed Query environment
   - MDDB
2. Explain the following:
   - ROLAP
   - MOLAP
3. Discuss the difference between Relational OLAP and Multidimensional OLAP.
4. Explain the basic architecture of OLAP with the help of a diagram.
5. What are the various OLAP tools available? Explain any one of them.

Notes

Rai Technology University

ENGINEERING MINDS

Rai Technology University Campus

Dhodballapur Nelmangala Road, SH -74, Off Highway 207, Dhodballapur Taluk, Bangalore - 561204

E-mail: info@raitechuniversity.in | Web: www.raitechuniversity.in