Master of Business Administration

(Open and Distance Learning Mode)

Semester – II

Introduction to Business Analytics

Centre for Distance and Online Education (CDOE)

DEVI AHILYA VISHWAVIDYALAYA, INDORE

"A+" Grade Accredited by NAAC

IET Campus, Khandwa Road, Indore - 452001

www.cdoedavv.ac.in

www.dde.dauniv.ac.in

Program Coordinator

Dr. Manishkant Arya
Centre for Distance and Online Education (CDOE)
Devi Ahilya Vishwavidyalaya, Indore – 452001

Content Design Committee

Dr. Sangita Jain
Institute of Management Studies
Devi Ahilya Vishwavidyalaya, Indore – 452001

Dr. Yamini Karmarkar
Institute of Management Studies
Devi Ahilya Vishwavidyalaya, Indore – 452001

Dr. Geeta Neema
International Institute of Professional Studies
Devi Ahilya Vishwavidyalaya, Indore – 452001

Dr. Manishkant Arya
Centre for Distance and Online Education (CDOE)
Devi Ahilya Vishwavidyalaya, Indore - 452001

Language Editors

Dr. Arti Sharan
Institute of Engineering & Technology
Devi Ahilya Vishwavidyalaya, Indore – 452001

Dr. Ruchi Singh
Institute of Engineering & Technology
Devi Ahilya Vishwavidyalaya, Indore – 452001

SLM Author(s)

Mr. Soumya Roy
MBA
EMRC, Devi Ahilya Vishwavidyalaya, Indore – 452001

Ms. Neha Jain Dugeria
MBA
EMRC, Devi Ahilya Vishwavidyalaya, Indore – 452001

Edition : 2022 (Restricted Circulation)
Published by : Centre for Distance and Online Education (CDOE), Devi Ahilya Vishwavidyalaya
Printed at : University Press, Devi Ahilya Vishwavidyalaya, Indore – 452001

Introduction
to
Business Analytics

Table of Content

Unit 1: Concept of Data, Information, & Knowledge

1.7 Check Your Progress

1.0 Introduction

Anything that is recorded is data. Observations and facts are data. Anecdotes and opinions are also data, of a different kind. Data can be numbers, like the record of daily weather, or daily sales. Data can be alphanumeric, such as the names of employees and customers.

Definitions of information depend on the way in which the term "data" is defined. The major point of difference is whether information can be produced by an automated process and how this information, which is also digital, recorded and can be transmitted, differs from data.

What is knowledge? When someone memorises information this is often referred to as 'rote-learning' or 'learning by heart. We can then say that they have acquired some knowledge. Another form of knowledge is produced as a result of understanding information that has been given to us and using that information to gain knowledge of how to solve problems.

1.1 Unit Objective

Discuss the concept of data, information, and knowledge in detail.

1.2 Concept of Data

Anything that is recorded is data. Observations and facts are data. Anecdotes and opinions are also data, of a different kind. Data can be numbers, like the record of daily weather, or daily sales. Data can be alphanumeric, such as the names of employees and customers.

1. Data could come from any number of sources. It could come from operational records inside an organisation, and it can come from records compiled by industry bodies and government agencies. Data could come from individuals telling stories from memory and from people's interactions in social contexts. Data could come from machines reporting their own status or from web usage logs.
2. Data can come in many ways. It may come as paper reports. It may come as a file stored on a computer. It may be words spoken over the phone. It may be e-mail or chat on the Internet. It may come as movies, songs, DVDs, and so on.

3. There is also data about data. It is called metadata. For example, people regularly upload videos on YouTube. The format of the video file (whether it was a high-def file or lower resolution) is metadata. The information about the time of uploading is metadata. The account from which it was uploaded is also metadata. The record of downloads of the video is also metadata.

1.2.1 Definitions of data

"Data: A representation of facts, concepts or instructions in a formalised manner suitable for communication, interpretation, or processing by humans or by automatic means." (Hicks [1993: 668] quoted by Checkland and Holwell [1998])

Three aspects of data can be identified. These correspond with the three ontologies, realism, nominalism and socially constructed reality and the corresponding beliefs about physical and social reality (objective, subjective or intersubjective).

As a result, they emphasise the different possible roles of data, namely, to

- Record objective facts which will be understood in exactly the same way by everyone;
- Record absolutely any type of concept, with no guarantees as to its accuracy or validity, which will be interpreted in all sorts of different ways by individuals;
- Use agreed structures and conventions for representing information, recording it and transmitting it, all in order to communicate it.

The objective view tends to assume that all data processing will be automated. The subjective view is very different in that it emphasises that if data are processed using a computer, the output is still only more highly structured or reformatted data. The intersubjective view allows for the possibility that data may be processed either by computer or directly by a person.

1.2.2 The objective point of view of data

The objective view makes the following assumptions about data.

- They are factual, resulting from the recording of measurable events, or objects.
- They record particular instances of reality.
- Introna [1992: 2.42] takes a purely objective view of data, proclaiming them to be

"Aperspectual, ahistorical, acontextual".

- They are explicit as they are in a fixed, recorded form.
- Hence, they can be communicated digitally.
- Modern society generates enormous amounts of data that record details of individual events and objects.
- This objective point of view accepts that the data are validated in the sense that they must be measured and recorded accurately. Certain logic checks as to the reasonableness of the data can be done to try to determine whether the data capturing instruments or processes have failed. Data can be shown to be true if they correspond to reality.

The definitions that follow are examples of those that refer only to the objective characteristics of data:

- "Data represent unstructured facts." (Avison and Fitzgerald [1995: 12] quoted by Checkland and Holwell [1998])
- "Data: Facts collected from observations or recordings about events, objects or people." (Clare and Loucopoulos [1987: 2] quoted by Checkland and Holwell [1998])
- "Data: The raw material of organizational life; it consists of disconnected numbers, words, symbols and syllables relating to the events and processes of the business." (Martin and Powell [1992: 10] quoted by Checkland and Holwell [1998])

1.2.3 The subjective point of view of data

On the other hand, the subjective view makes the following assumptions about data.

- The data are not necessarily true or accurate as not all errors can be detected automatically and not everyone will necessarily agree that they are a true representation of a particular fact.
- Some data record subjective opinions, not facts. If data can represent opinions and concepts, they are not truly objective.
- Data represent information and are the only way we can make information explicit.
- Nothing but data can be communicated digitally or in any other way. Only data are transmitted, be it by means of a telecommunications medium, in printed form, or directly without using any technology.

- Data have absolutely no meaning. They acquire meaning only when appropriated by a human recipient.

The definitions that follow are examples of those that include subjective aspects. These definitions include characteristics which have been highlighted as being objective or intersubjective as well. "Data: Natural language: facts given, from which others may be deduced, inferred. Info. Processing and computer science: signs or symbols, especially for transmission in communication systems and for processing in computer systems, usually but not always representing information, agreed facts or assumed knowledge; and represented using agreed characters, codes, syntax and structure." (Maddison [1989: 168] quoted by Checkland and Holwell [1998]).

1.2.4 The intersubjective point of view

- The purpose of data is to permit communication.
- Information exists before data. Some version of that information can be retrieved from the data.
- Data must be recorded in a formalised structure and knowledge of this structure must be shared as prior shared meaning. They cannot be totally unstructured or no one would ever be able to process them, but they can be reorganised into more complex structures during subsequent processing. The structure will result from language syntax and semantics if the data are in the form of text, or in the case of numeric or symbolic data, will depend on the predesigned layout of database records, forms or even the position of the data on a physical object. (We know something about what a number plate denotes from its position on a car even if the format is unfamiliar.)
- They are represented using agreed characters, codes, syntax and structure. A predetermined, agreed way of coding and decoding must be associated with this representation. A stream of bits is not data unless someone has the key by means of which it can be decoded.
- The fact that data are both recorded and have some structure makes them potentially useful - they are in a form suitable for subsequent interpretation and processing. Other information can be inferred and deduced from them and they can be associated with other data. They have potential meaning.

- They have an implied context and history. If the data have been captured or a procedure exists to capture them, a purpose has already been recognised.

Example definitions: "Data: Facts, concepts or derivatives in a form that can be communicated and interpreted." (Galland, [1982: 57] quoted by Checkland and Holwell [1998]) "Data are formalized representations of information, making it possible to process or communicate that information." [Dahlbom & Mathiassen, 1995: 26]

1.2.5 Types of Data

Data can be of different types.

1. Data could be an unordered collection of values. For example, a retailer sells shirts in red, blue, and green colours. There is no intrinsic ordering among these colour values. One can hardly argue that any one colour is higher or lower than the other. This is called nominal (means names) data.
2. Data could be ordered values like small, medium and large. For example, the sizes of shirts could be extra-small, small, medium, and large. There is clarity that medium is bigger than small, and large is bigger than medium. But the differences may not be equal. This is called ordinal (ordered) data.
3. Another type of data has discrete numeric values defined in a certain range, with the assumption of equal distance between the values. Customer satisfaction scores may be ranked on a 10-point scale with 1 being the lowest and 10 being the highest. This requires the respondent to carefully calibrate the entire range as objectively as possible and place his own measurement on that scale. This is called interval (equal intervals) data.
4. The highest level of numeric data is ratio data which can take on any numeric value. The weights and heights of all employees would be exact numeric values. The price of a shirt will also take any numeric value. It is called ratio (any fraction) data.
5. There is another kind of data that does not lend itself to much mathematical analysis, at least not directly. Such data needs to be first structured and then analyzed. This includes data like audio, video, and graphs files, often called BLOBs (Binary Large Objects). These kinds of data lend themselves to different forms of analysis and mining. Songs can be described as happy or sad, fast-paced or slow, and so on. They may contain sentiment and intention, but these are not quantitatively precise.

The precision of analysis increases as data becomes more numeric. Ratio data could be subjected to rigorous mathematical analysis. For example, precise weather data about temperature, pressure, and humidity can be used to create rigorous mathematical models that can accurately predict future weather.

Data may be publicly available and shareable, or it may be marked private. Traditionally, the law allows the right to privacy concerning one's personal data. There is a big debate on whether the personal data shared in social media conversations is private or can be used for commercial purposes.

Datafication is a new term that means that almost every phenomenon is now being observed and stored. More devices are connected to the Internet. More people are constantly connected to "the grid," by their phone network or the Internet, and so on. Every click on the web, and every movement of the mobile devices, is being recorded. Machines are generating data. The "Internet of things" is growing faster than the Internet of people. All of this is generating an exponentially growing volume of data, at high velocity. Kryder's law predicts that the density and capability of hard drive storage media will double every 18 months. As storage costs keep coming down at a rapid rate, there is a greater incentive to record and store more events and activities at a higher resolution. Data is getting stored in a more detailed resolution, and many more variables are being captured and stored.

1.3 Data Operations

1.3.1 Database

A database is a modelled collection of data that is accessible in many ways. A data model can be designed to integrate the operational data of the organization. The data model abstracts the key entities involved in an action and their relationships. Most databases today follow the relational data model and its variants. Each data modelling technique imposes rigorous rules and constraints to ensure the integrity and consistency of data over time.

Take the example of a sales organization. A data model for managing customer orders will

involve data about customers, orders, products, and their interrelationships. The relationship between the customers and orders would be such that one customer can place many orders, but one order will be placed by one and only one customer. It is called a one-to-many relationship. The relationship between orders and products is a little more complex. One order may contain many products. And one product may be contained in many different orders. This is called a many-to-many relationship. Different types of relationships can be modelled in a database.

Databases have grown tremendously over time. They have grown in complexity in terms of the number of objects and their properties being recorded. They have also grown in the quantity of data being stored. A decade ago, a terabyte-sized database was considered big. Today databases are in petabytes and exabytes. Video and other media files have greatly contributed to the growth of databases. E-commerce and other web-based activities also generate huge amounts of data. Data generated through social media has also generated large databases. The e-mail archives, including attached documents of organizations, are in similar large sizes.

Many database management software systems (DBMSs) are available to help store and manage this data. These include commercial systems, such as Oracle and DB2 systems. There are also open-source, free DBMS, such as MySQL and Postgres. These DBMSs help process and store millions of transactions worth of data every second.

Here is a simple database of the sales of movies worldwide for a retail organization. It shows sales transactions of movies over three quarters. Using such a file, data can be added, accessed, and updated as needed.

Movies Transactions Database				
Order #	**Date sold**	**Product name**	**Location**	**Amount**
1	April 2015	Monty Python	US	$9
2	May 2015	Gone With the Wind	US	$15
3	June 2015	Monty Python	India	$9
4	June 2015	Monty Python	UK	$12
5	July 2015	Matrix	US	$12
6	July 2015	Monty Python	US	$12
7	July 2015	Gone With the Wind	US	$15
8	Aug 2015	Matrix	US	$12
9	Sept 2015	Matrix	India	$12
10	Sept 2015	Monty Python	US	$9
11	Sept 2015	Gone With the Wind	US	$15

1.3.2 Data Warehouse

A data warehouse is an organized store of data from all over the organization, specially designed to help make management decisions. Data can be extracted from an operational database to answer a particular set of queries. This data, combined with other data, can be rolled up to a consistent granularity and uploaded to a separate data store called the data warehouse. Therefore, the data warehouse is a simpler version of the operational database, with

the purpose of addressing reporting and decision-making needs only. The data in the warehouse cumulatively grow as more operational data becomes available and is extracted and appended to the data warehouse. Unlike in the operational database, the data values in the warehouse are not updated. To create a simple data warehouse for the movie sales data, assume a simple objective of tracking sales of movies and making decisions about managing inventory. In creating this data warehouse, all the sales transaction data will be extracted from the operational data files. The data will be rolled up for all combinations of the time periods and product numbers. Thus, there will be one row for every combination of time period and product. The resulting data warehouse will look like the table that follows.

Movies Sales Data Warehouse

Row#	Qtr sold	Product name	Amount
1	Q2	Gone With the Wind	$15
2	Q2	Monty Python	$30
3	Q3	Gone With the Wind	$30
4	Q3	Matrix	$36
5	Q3	Monty Python	$30
6	Q4	Gone With the Wind	$15
7	Q4	Monty Python	$18

The data in the data warehouse is in much less detail than in the transaction database. The data warehouse could have been designed at a lower or higher level of detail, or granularity. If the data warehouse were designed on a monthly level, instead of a quarterly level, there would be many more rows of data. When the number of transactions approaches millions and higher, with dozens of attributes in each transaction, the data warehouse can be large and rich with

potential insights. One can then mine the data (slice and dice) in many different ways and discover unique meaningful patterns. Aggregating the data helps improve the speed of analysis. A separate data warehouse allows analysis to go on separately in parallel, without burdening the operational database systems (Table 1.1).

Function	Database	Data Warehouse
Purpose	Data stored in databases can be used for many purposes including day-to-day operations	Data stored in DW is cleansed data useful for reporting and analysis
Granularity	Highly granular data including all activity and transaction details	Lower granularity data; rolled up to certain key dimensions of interest
Complexity	Highly complex with dozens or hundreds of data files, linked through common data fields	Typically organized around a large fact tables, and many lookup tables
Size	Database grows with growing volumes of activity and transactions. Old completed transactions are deleted to reduce size.	Grows as data from operational databases is rolled-up and appended every day. Data is retained for long-term trend analyses
Architectural choices	Relational, and object-oriented, databases	Star schema, or Snowflake schema
Data Access mechanisms	Primarily through high level languages such as SQL. Traditional programming access DB through Open DataBase Connectivity (ODBC) interfaces	Accessed through SQL; SQL output is forwarded to reporting tools and data visualization tools

Table 1.1: Comparing Database systems with Data Warehousing systems

1.3.3 Data Mining

Data Mining is the art and science of discovering useful innovative patterns from data. There is a wide variety of patterns that can be found in the data. There are many techniques, simple or complex, that help with finding patterns. In this example, a simple data analysis technique can be applied to the data in the data warehouse above. A simple cross-tabulation of results by quarter and products will reveal some easily visible patterns.

Movies Sales by Quarters – Cross-tabulation				
Qtr/Product	**Gone With the Wind**	**Matrix**	**Monty Python**	**Total Sales Amount**
Q2	$15	0	$30	$45
Q3	$30	$36	$30	$96
Q4	$15	0	$18	$33
Total Sales Amount	$60	$36	$78	$174

Based on the cross-tabulation above, one can readily answer some product sales questions, like:

1. What is the best-selling movie by revenue? – Monty Python.
2. What is the best quarter by revenue this year? – Q3
3. Any other patterns? – Matrix movie sells only in Q3 (seasonal item).

These simple insights can help plan marketing promotions and manage the inventory of various movies.

If a cross-tabulation was designed to include customer location data, one could answer other questions, such as

1. What is the best-selling geography? – US
2. What is the worst selling geography? – UK
3. Any other patterns? – Monty Python sells globally, while Gone with the Wind sells

only in the US.

If the data mining was done at the monthly level of data, it would be easy to miss the seasonality of the movies. However, one would have observed that September is the highest selling month.

The previous example shows that many differences and patterns can be noticed by analyzing data in different ways. However, some insights are more important than others. The value of the insight depends upon the problem being solved. The insight that there are more sales of a product in a certain quarter helps a manager plan what products to focus on. In this case, the store manager should stock up on Matrix in Quarter 3 (Q3). Similarly, knowing which quarter has the highest overall sales allows for different resource decisions in that quarter. In this case, if Q3 is bringing more than half of total sales, this requires greater attention to the e-commerce website in the third quarter.

Data mining should be done to solve high-priority, high-value problems. Much effort is required to gather data, clean and organize it, mine it with many techniques, interpret the results, and find the right insight. It is important that there be a largely expected payoff from finding the insight. One should select the right data (and ignore the rest), organize it into a nice and imaginative framework that brings relevant data together, and then apply data mining techniques to deduce the right insight.

A retail company may use data mining techniques to determine which new product categories to add to which of their stores; how to increase sales of existing products; which new locations to open stores in; how to segment the customers for more effective communication; and so on.

Data can be analyzed at multiple levels of granularity and could lead to a large number of interesting combinations of data and interesting patterns. Some of the patterns may be more meaningful than others. Such highly granular data is often used, especially in finance and high-tech areas, so that one can gain even the slightest edge over the competition.

Here are brief descriptions of some of the most important data mining techniques used to

generate insights from data.

Decision Trees: They help classify populations into classes. It is said that 70% of all data mining work is about classification solutions; and that 70% of all classification work uses decision trees. Thus, decision trees are the most popular and important data mining technique. There are many popular algorithms to make decision trees. They differ in terms of their mechanisms and each technique works well for different situations. It is possible to try multiple decision-tree algorithms on a data set and compare the predictive accuracy of each tree.

Regression: This is a well-understood technique from the field of statistics. The goal is to find a best-fitting curve through the many data points. The best fitting curve is that which minimizes the (error) distance between the actual data points and the values predicted by the curve. Regression models can be projected into the future for prediction and forecasting purposes.

Artificial Neural Networks: Originating in the field of artificial intelligence and machine learning, ANNs are multi-layer non-linear information processing models that learn from past data and predict future values. These models predict well, leading to their popularity. The model's parameters may not be very intuitive. Thus, neural networks are opaque like a black box. These systems also require a large amount of past data to adequately train the system.

Cluster analysis: This is an important data mining technique for dividing and conquering large data sets. The data set is divided into a certain number of clusters, by discerning similarities and dissimilarities within the data. There is no one right answer for the number of clusters in the data. The user needs to make a decision by looking at how well the number of clusters chosen fits the data. This is most commonly used for market segmentation. Unlike decision trees and regression, there is no one right answer for cluster analysis.

Association Rule Mining: Also called Market Basket Analysis when used in the retail industry, these techniques look for associations between data values. An analysis of items frequently found together in a market basket can help cross-sell products, and also create product bundles.

1.3.4 Data Visualization

As data and insights grow in number, a new requirement is the ability of the executives and decision-makers to absorb this information in real-time. There is a limit to human comprehension and visualization capacity. That is a good reason to prioritize and manage with fewer but key variables that relate directly to the Key Result Areas (KRAs) of a role. Here are a few considerations when presenting using data:

1. Present the conclusions and not just report the data.
2. Choose wisely from a palette of graphs to suit the data.
3. Organize the results to make the central point stand out.
4. Ensure that the visuals accurately reflect the numbers. Inappropriate visuals can create misinterpretations and misunderstandings.
5. Make the presentation unique, imaginative and memorable.

Executive dashboards are designed to provide information on select few variables for every executive. They use graphs, dials, and lists to show the status of important parameters. These dashboards also have a drill-down capability to enable a root-cause analysis of exception situations (Figure 1.3).

Figure 1.3: Sample Executive Dashboard

Data visualization has been an interesting problem across disciplines. Many dimensions of data

can be effectively displayed on a two-dimensional surface to give a rich and more insightful description of the totality of the story.

The classic presentation of the story of Napoleon's march to Russia in 1812, by French cartographer Joseph Minard, is shown in Figure 1.4. It covers about six dimensions. Time is on the horizontal axis. The geographical coordinates and rivers are mapped. The thickness of the bar shows the number of troops at any point in time that is mapped. One colour is used for the onward march and another for the retreat. The weather temperature at each time is shown in the line graph at the bottom.

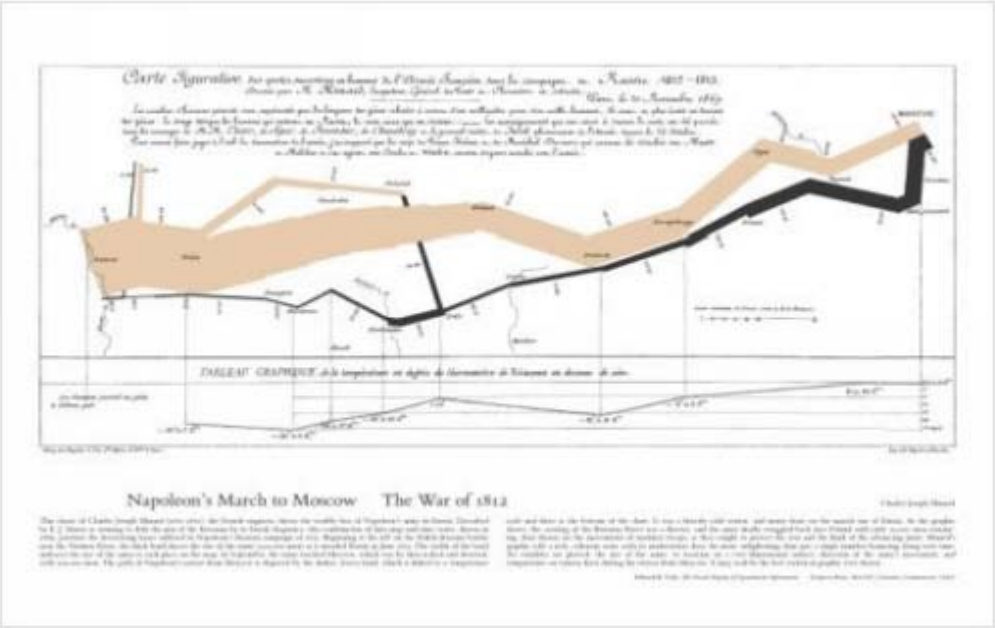


Figure 1.4: Sample Data Visualization

1.4 Information

Definitions of information depend on the way in which the term "data" is defined. The major point of difference is whether information can be produced by an automated process and how this information, which is also digital, recorded and can be transmitted, differs from data.

1.4.1 The objective point of view

- Information is output from a computer program.
- The systems analyst decides what output will be useful. This output remains useful and meaningful regardless of who the recipient is.

- The processing which produces the information includes summarising in order to reduce the volume of data.
- Data may be associated with other data, which may be obtained from different sources, to produce the information.
- The processing (classifying, linking, summarising, sorting, presentation) adds value in the form of potential meaning. The less structured data are less useful and less meaningful than the more structured information.

In the definition that follows "a meaning" seems to imply that the meaning is fixed and not open to interpretation and, therefore, this definition refers only to the objective characteristics of information:

"Information has a meaning ... (it) comes from selecting data, summarizing it and presenting it in such a way that it is useful to the recipient." (Avison and Fitzgerald [1995:12] quoted by Checkland and Holwell [1998])

1.4.2 The subjective point of view

- Data become information only once they have been appropriated by the human recipient. Hence, the output from any computer program is still data.
- The added value of information (compared with data) results from the recipient appropriating the new data, interpreting them and placing them in context by combining them with existing personal information.
- Some authors consider data to be information only if they are used by the recipient in making a decision.
- Data become information only if they include something previously unknown to the recipient.
- Introna considers the information to be historical, contextual and perceptual [Introna, 1992: 2.42]. It is moulded by life experience (Orleans) to provide understanding.

The definitions that follow are examples of those that include subjective aspects. These definitions also include some characteristics which are considered to be objective or intersubjective.

"Information usually implies data that is organized and meaningful to the person receiving it.

Data is therefore raw material that is transformed into information by data processing. Information can be defined in terms of its surprise value. It tells the recipient something he did not know." (Davis et al [1985: 30] quoted by Introna [1992])

"Information: (1) Data that has been transformed into a meaningful and useful form for specific human beings. (2) The meaning that a human assigns to data by means of the known conventions used in its representations." [Lay et al, 1993: 535]

"Information is that which results when some human mental activity (observation, analysis) is successfully applied to data to reveal its meaning or significance." (Galland [1982: 127] quoted by Checkland and Holwell [1998])

"Information is the particular instances of reality as experience, perceived or understood by an individual in a specific context." [Introna, 1992:2.37] "... information comes into being as the receiver appropriates the data and gives it meaning" [Introna, 1992: 2.39]

1.4.3 The intersubjective point of view

- Aspects of shared meaning and discourse (validity claims [Braaten, 1991: 14]) are characteristic of this point of view.
- The recipient has participated in the systems analysis and hence has influenced the process and has said what output was likely to be meaningful to him and others using the system.
- More advanced, database-oriented, systems allow the user to formulate queries and interact directly with the data in the database. Hence, there is a more dynamic process where the user's judgement is combined with the power of the technology.
- Information "has the meaning" which can be communicated versus "is meaning" from the objective point of view.
- The information must be put into some context "... in order to understand something, we already need a preliminary understanding of it" [Dahlbom & Mathiassen, 1995: 32]. This preliminary understanding must be shared in order for a new shared understanding to result.

preconceptions plus information = interpretation => knowledge

"To produce the information we have to interpret what we experience and make explicit what we know." [Dahlbom & Mathiassen, 1995: 26]

1.4.4 Information: Discussions

There are a number of levels of information which are used in different ways and carry different kinds of meaning. The fact that these are all commonly referred to as information can be confusing.

1.4.4.1 Information - lean information

The first sort of information is data (or capta) which has been processed electronically (summarised, sorted, classified, analysed or simply collated and associated with other appropriate data), reformatted and made available to someone who is expected to find it meaningful and useful (and may act on it). The processing associated with the production of this type of information is mechanistic and procedural and can be described by an algorithm. The processing is done deliberately in order to add value. Since this processing was not done by a person, in Checkland and Holwell's terms [1998] This is still capta. In Habermas' terms, this type of information is the manifestation of functional rationality (administrative systems). The intention is, therefore, that:

- The information will be used to initiate action or decisions within a clearly delimited scope of function and time or should be available when action and decisions are required. Its purpose is unambiguous.
- Hence, it generally has a short useful life.
- It has a limited readership for whom it is meaningful and who are authorised to access it. The person who creates the specification of the program that will produce this type of information specifically intends to produce information that the reader will react to and has a relatively stringently defined target group in mind who is expected to be very familiar with the context and share the frame of reference.
- The information is necessary either because it is not already available or to reduce uncertainty by confirming facts.

- Provided that the context is explained, this information is not ambiguous. For example, the trade deficit may be given in a recognised format, for a specified currency, for a specified country, on a certain date, calculated according to a procedure for which a description and explanation are available.
- It is specific, not universal, and hence acontextual, ahistorical and not perpetual leading to local knowledge (functional rationality) rather than wisdom or universal knowledge (theoretical rationality).

1.4.4.2 Information - rich information

What then is the type of information that is not the output from a computer process, such as information in newspapers and textbooks - a text which is intended to inform? Is the difference between Information Systems and Information Science which of these two types of information the subject tries to make available or accessible? This 'information' has been carefully put in context (explained) and a human has done this processing but it is in artefact form (the information is printed on paper or displayed on a computer screen) and is a commodity. In Checkland and Holwell's terms, this is still capta, as it is external to the human mind. In Habermas' terms, the intentional behaviour which produces this type of 'information' may be strategic or communicative action and the communicative rationality would be authentic self-expression referring to the aesthetic sphere. It might be instrumental if it consists largely of instructions such as a manual or recipe book.

This type of information, which will be called Information, has the following characteristics in comparison with Information.

- It is not as likely to initiate immediate actions or decisions as Information but is intended to initiate thought and to influence the reader by altering his lifeworld.
- It is produced deliberately for a purpose but this purpose has a less well-defined scope and the expected outcomes are less specific.
- It has a wider readership. (The newspaper is not prepared for one specific reader, it is prepared with the intention to inform.)
- It is relevant for a longer period.
- It is less structured than information produced by a computer and has more complete sentences and fewer tables.

- It handles more of the type of information which is "soft" and cannot easily be expressed in terms that are exact or precise. Hence, readers need to interpret the information to a greater extent than in the first type. It is for this reason that it is likely to contain more redundant information and that the context is usually explained in greater detail - except where it is assumed that the context is already known, in which case the information may be very difficult for "an outsider" to understand.
- It handles more complex information, where more explanation is required. (Although this might not be considered to be discourse because it is not interactive or a dialogue, it can certainly present one or more sides of an argument and try, by force of the better argument, to convince the reader of a particular "truth".)
- It is sometimes specific but may also attempt to explain universal "truths".
- It is perceptual in the sense that most complex information, formulated by a person, includes that person's perspective, slant, or bias.

The two types of information are related: Both are intended to present facts (and in the case of Information, may include opinions) in a form which will probably (hopefully) communicate meaning to the recipient. In Information, ambiguity is avoided. In Information, ambiguity may be inevitable and is sometimes courted. Information is "lean". Information is "rich". The more subtle and equivocal a text is, the richer it may be considered to be. It is easier to refer to both lean and rich information as information even before it is appropriated and hence, for the moment, neither Introna's nor Checkland and Holwell's terminology is used.

Both Information and Information are processed data in a sense. In one case, a computer has done predetermined calculations and processing to produce the information. In the second case, a person has either intuitively or deliberately processed the information and combined it with his or her unique world view, possibly analysed, synthesised or evaluated it but definitely expressed it in an individual way. As with all aspects of technology, an appropriate fit between the task and the technology is important although it is not the only factor involved in adopting a specific technology. Rich information corresponds largely with "thick information". "In fact in one area - so-called thick information - management technology can be dangerously limiting. As defined by Henry Mintzberg of McGill University, thick information is irrational, subjective, intuitive knowledge that transcends what can be categorized on an MIS report."

[Davidow & Malone, 1992 : 170]

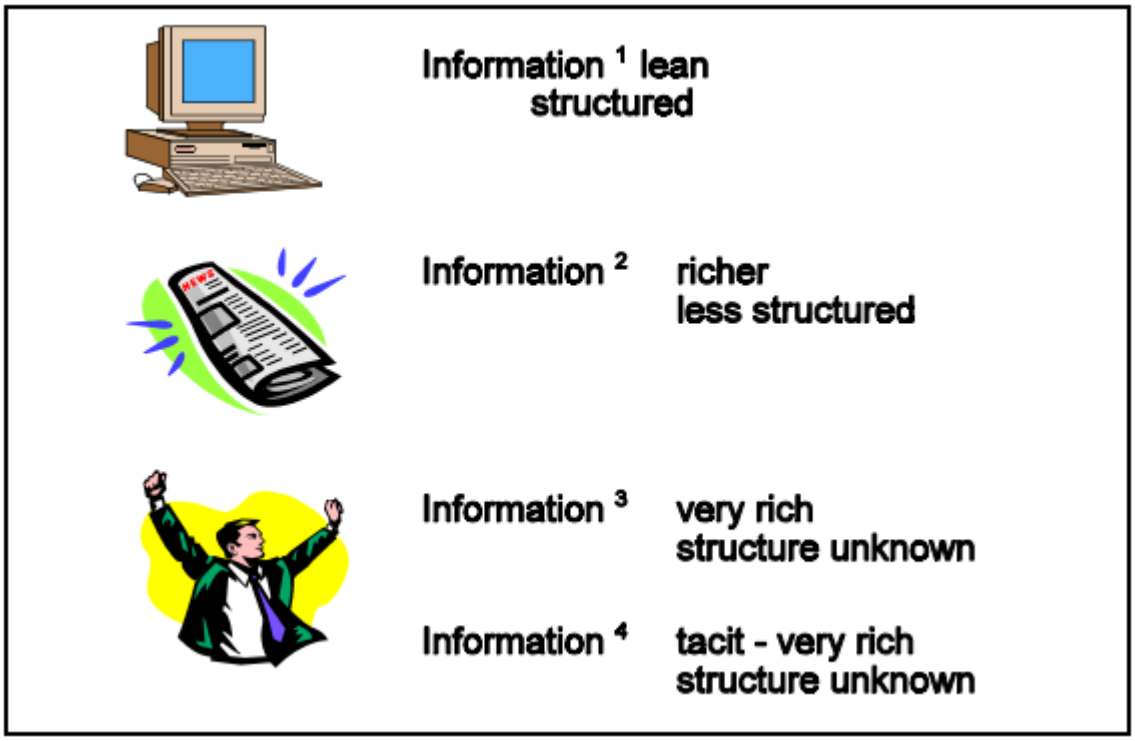
1.4.4.3 Information - appropriated information

When a human acquires information and processes it mentally, some form of learning occurs. The user or acquirer of information relates it to existing knowledge and information and the third type of information results. During this process meaning is attributed to the information. It is possible to assimilate isolated facts but it is easier to remember information that can be related to an existing mental landscape (or a worldview, or mental model). Information is personal and mentally stored and has meaning. This is the richest information. It is impossible to express it exactly and completely. Hence Information cannot be transferred. It may change from day to day. It probably does not correspond entirely with anyone else's version. How it is structured is unknown. The creation of Information is the process of learning. Further learning can occur without further input of Information, as the learner restructures the information and associates it with other personal beliefs, values and knowledge. This is the 'true information' according to Checkland and Holwell [1998] and Introna [1992]. Introna [1992] explains the process of developing understanding in terms of the hermeneutic circle. Interpretations of text follow a cycle in which a new element is interpreted alone and then in terms of the larger text. This results in a re-interpretation of the meaning of the text as a whole. The new element may then be re-interpreted in terms of the new understanding of the complete text. Thus, the individual starts off with certain prior knowledge or prejudices in order to form an initial understanding of a text. This new information is then related to the larger context of the learner's lifeworld (traditions, economic and social situation) and he arrives at a new understanding of the lifeworld. New understanding now influences the more particular interpretation of the text and this will be modified. Once again the modified information will be related to the broader context, and once again this might be adjusted to reach a consistent view. This process will continue until the new information and the larger picture no longer have any inconsistencies.

1.4.4.4 Information - tacit information

Certain types of information are more difficult than others to put into words. Tacit information may be difficult to verbalise. In fact, some skills, such as balancing while riding a bicycle, do not seem to have words that describe them. Hence, this information is usually imparted by

demonstrating it and by the recipient learning by trial and error or practising the skill.



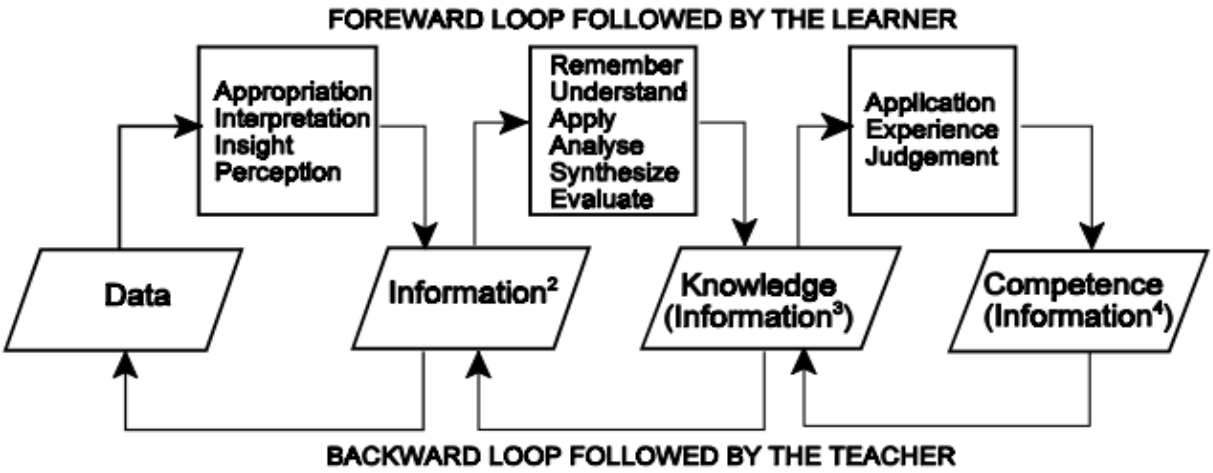


Figure: Adapted from De Villiers (1955:83) Model of the teaching and learning process

Another type of interpersonal communication that is difficult to achieve in formalised exchanges of information is "brainstorming", (problem-solving where a small team or only two people exchange ideas). These may often be verbalised as fragments of conversation which do not seem to make a lot of sense when transcribed and where leaps of logic are particularly fruitful. The term "on the same wavelength" seems to imply the idea that the participants have

evoked already existing, matching, world views. Hence, they need fewer words to refer to a concept that they share and may naturally relate concepts that they both consider appropriate. Eventually, they will arrive at a conclusion or solution that they both find satisfactory.

This creative problem solving is not considered by this author to be tacit. It stems rather from having a compatible background and language of practice (a large degree of existing shared meaning) but is also stimulated by fast response and even interruption so that concepts are moulded before "they set". Introna [1998] is concerned with the fact that cooperation needs this type of intimate and rich communication and that this will be impossible to achieve in technologically supported environments. This seems to point to Information, which is tacit.

1.4.4.5 Making Information explicit

When a person expresses Information as words, music or anything else, it becomes Information. It loses part of its context, its meaning is once again only potential, it may not be well expressed and hence be prone to misinterpretation. This is explained by Dahlbom and Mathiassen [1995: 33] and illustrated in Figure 3.2, based on a diagram in De Villiers [1995: 83]. Mingers [1995], using Dretske's terminology, refers to this as making analogue information digital. Codified information, that is, Information is by definition already explicit and is, therefore, easy to copy and has little natural protection. Tacit information, because it is very difficult to capture and make explicit can be lost permanently when, for example, an experienced member of staff retires [Chesbrough & Teece, 1996].

1.5 Knowledge

What is knowledge? When someone memorises information this is often referred to as 'rote-learning' or 'learning by heart. We can then say that they have acquired some knowledge. Another form of knowledge is produced as a result of understanding information that has been given to us and using that information to gain knowledge of how to solve problems.

Knowledge can therefore be:

- acquiring and remembering a set of facts, or
- the use of information to solve problems.

The first type is often called explicit knowledge. This is the knowledge that can be easily passed on to others. Most forms of explicit knowledge can be stored in certain media. The information contained in encyclopedias and textbooks is a good example of explicit knowledge.

The second type is called tacit knowledge. It is the kind of knowledge that is difficult to pass on to another person just by writing it down. For example, saying that Paris is the capital of France is explicit knowledge that can be written down, passed on, and understood by someone else. However, the ability to speak a foreign language, bake bread, program a computer or use complicated machinery requires additional pieces of knowledge (such as that gained through experience) that are not always known explicitly and are difficult to pass on to other users.

If we put Knowledge into an equation it would look like this: Information + application or use = Knowledge.

1.5.1 Philosophies of knowledge acquisition

There are a variety of theories associated with education. These include theories of the development of the child (for example, Piaget), ways of classifying activities associated with learning (Bloom's taxonomy) and theories that set out to explain how people learn. Learning models can be classified as being behavioural or cognitive and can in turn be associated with the different beliefs about physical and social reality (that is, objective, subjective or intersubjective). An example of a behaviourist learning model is Objectivism. Constructivism and sociocultural are both cognitive models but emphasise the subjective and intersubjective points of view.

1.5.1.1 Behaviourism

Behaviourist models of learning follow the ideas that originated with Pavlov and were continued by Skinner. The central idea is that learning occurs as a result of the learner recognising a stimulus and responding in a predictable way. The use of positive reinforcement (rewards or praise) when the required behaviour is exhibited encourages the learner to react in this way whenever the stimulus is encountered. This theory is built on the supposition that there is an objective world that everyone will perceive in the same way and that everyone can be taught to react in the same way. Hence, teaching and learning involve a transfer of

knowledge which will result in the learner eventually obtaining an exact copy of the knowledge of the teacher. The teacher is in control of the process. This theory assumes that "... the purpose of the mind is to act as a mirror of reality rather than as an interpreter of reality. " [Leidner & Jarvenpaa, 1995]

Amongst these models are those referred to as Objectivist [Leidner and Jarvenpaa, 1995]. In the learning of Mathematics, Platonism and Logicism view reality as being objective [Matthee, 1998]. The objectivist approach is the one mostly used at the tertiary level. Lectures are a good example of teaching according to such a philosophy. This approach assumes that students learn best in locations, such as classrooms, which are removed from outside distractions and that intensive study for relatively short periods is most effective, such as hour-long lecture periods. Leidner and Jarvenpaa [1995] suggest that this approach is most suitable for factual or procedural based learning.

Variations of the basic Stimulus-Response theory exist. Thorndike proposed three primary laws which determine whether learning takes place [De Villiers, 1995]. Firstly, the learner must be ready (presumably this includes physical maturity as well as motivation). Secondly, the association must be built and strengthened as a result of practice and repetition. Finally, the outcome must be rewarding.

1.5.1.2 Constructivism

The Constructivist philosophy, as explained by Alavi [1994], says that learners must be actively involved in the learning process. The learners acquire, generate, analyse, manipulate and structure information in order to construct knowledge. Information forms an essential basis for learning. The new Information2 must be interpreted, elaborated on and related to other Information. Thus, a relatively subjective view of reality is adopted as each person's reality is different. This approach is learner-centred with students controlling the pace and formulating their own questions. Nevertheless, most constructivists acknowledge the existence of an objective world and believe that learning involves a hermeneutic process in which new information is interpreted by the learner. The interpretation is coloured by his existing prejudices but is then compared with the different interpretations offered by others (teachers, classmates or reference books). The learner will gradually or incrementally adjust his personal

interpretation until, ideally, the subjective reality for this particular piece of knowledge is very close to the universal objective reality. "Gadamer's point would seem to be that anything new can only be understood in terms of what one already knows. The first step in understanding is based on what one already knows or the tradition within which one finds oneself. If the process of understanding stops after the first step then, yes, it is subjective. But if one continually opens oneself to the text and continually re-evaluates one's understanding against the text, one will be able to complete the meaning through the process of understanding." [Introna, 1992:2.23]

Piaget is considered to be primarily a supporter of constructivism as he emphasises the role of the learner in discovering knowledge and not the teacher.

Radical constructivists take a more extreme view as they believe that there is no shared understanding and that each individual discovers or constructs his own view of the world, building unique schemas. They sometimes go as far as to say that knowledge cannot be taught or learned from books [Thomas, 2000:87].

Even if radical constructivism is not accepted, constructivists recognise that different people have different learning styles and should be encouraged to consciously recognise this and hence improve their learning effectiveness and efficiency. The following characteristics of constructive learning are derived from those given by Simons ([in Duffy et al, 1993] cited by De Villiers [1995:79]).

- The learner must be actively involved.
- The learner will interpret the new information in the context of existing information.
- The resulting knowledge will, therefore, extend existing knowledge as additional meaning is constructed.
- The learner must be aware of the goals towards which he is working.
- The learner must ensure that he is still on course and progressing towards the goal.
- The learner must be conscious of his way of learning.

There is no real reason why these requirements cannot be achieved in a lecture but it is the learner who must actively participate during the lecture. The lecturer cannot ensure that this learning occurs. However, this approach is well suited for topics where relationships can be determined, multiple representations compared and a real-world context explored. It is not

suitable for acquiring a fixed set of preordained, factual knowledge [Leidner & Jarvenpaa, 1995].

1.5.1.3 The sociocognitive learning theory of Piaget

The sociocognitive model is an example of constructivism. Piaget's writings illustrate how children naturally acquire skills and knowledge without having to be taught them. Piaget believed that all humans develop cognitive abilities in a more or less fixed sequence and at predictable times in their lives. Piaget did not set out to develop a theory specifically relevant to education or teaching but his work concerning the stages of intellectual development, as well as his explanation of what kinds of intellectual structures can be formed at the various stages and how these are developed, is extremely relevant in this context.

Piaget identified two functional invariants which apply to all stages of intellectual development, namely adaptation and organisation. These concepts are closely linked to one another. Organisation refers to the intellectual structures or schemas (or schemata) that the learner develops and makes use of as strategies for solving problems. These strategies are reasonably specific (for example, schemas for doing addition and schemas for reading maps). More complex schemas are created by combining existing schemas. Hence, arithmetic can be combined with map reading to find the shortest route between two places.

"Thus as intellectual development proceeds, the individual's schemas become more complex, differentiated and capable of greater generalisation to situations yet always organised and integrated." [McNally, 1977:7]

The second functional invariant, adaptation, involves two activities, namely assimilation and accommodation. During assimilation data are input from the environment and interpreted in terms of current cognitive structures. The new data may not fit the existing schema exactly and accommodation occurs in order to adapt the schema and in so doing reconcile the most recent experience and the previous experience to an acceptable degree. In this way, equilibrium is reached but this is a temporary condition as the human mind is constantly assimilating new data and accommodating existing schemata. In fact, this intellectual curiosity and adaptation of intellectual structures is a fundamental characteristic of mankind. It is important to note the

difference between the schemas and the facts or content that a person remembers. The same facts will be interpreted or even recollected differently if the schema being applied has been revised. Schemas are not typically forgotten whereas facts may be. Schemas correspond to some extent with tacit knowledge and may be difficult for the learner to explicate.

A description of the stages of intellectual development is unnecessary in this context since, at the tertiary education level, all students can be expected to have reached the final stage, that of formal operational functioning, in which the individual can handle abstract concepts. At this stage, the learner can use hypothetic-deductive reasoning and has a uniform logical system which can be used systematically to isolate variables and hence determine relationships between them. The learner is capable of understanding assertions or propositions independently of concrete examples and can examine these critically. When new concrete facts are assimilated, the learner can be expected to look at the circumstances from a broad perspective and associate information from other sources obtained at other times. Hence, the learner can identify logical relationships and can integrate concepts, seeing how various factors and their relationships interact.

Since the learner at this stage does not need to work from concrete examples in order to understand concepts, use of language, either spoken or written is used more in reasoning and learning than prior to this phase. (The student is capable of interpreting explicit, lean Information2 into rich Information3 or tacit Information4 .) Nevertheless, active participation by the learner in the learning process is still required even though it is not necessary that the learner is given as much practice in manipulating the concrete examples. Richmond [1970: 94] says, " ... learning at any age needs contact with concrete reality. Piaget expresses this as follows: 'The subject must be active, must transform things, and find the structure of his own actions on the objects.' Piaget Rediscovered, p. 4)"

Richmond [1970:108] says specifically that, even in the period of formal operations, "... [t]he needs for contact with the concrete remains in order that potential generalizations may be modified." McNally [1977: 74] confirms this, "... he is clearly in favour of true activity methods, ..."

Piaget makes the distinction between figurative and operative aspects of knowing. Figurative knowing is associated mostly with the perception or image of the object and hence is fairly passive. It is the more superficial knowledge of the static, material characteristics of objects. The operative aspect is closely associated with meaning and is significantly influenced by the existing intellectual structures (schemas) and will probably result in the adaptation of these structures. This type of knowing is more active, in that the learner assimilates the new data. It is encouraged by providing the learner with opportunities to actively interact with the environment. The learner discovers concepts by means of this interaction followed by reflection, leading to the formal abstraction of relationships [McNally, 1977: 101]. Figurative knowing is, therefore, likely to involve memorising, while operative knowing involves understanding.

The social interaction of the individual is an essential part of learning. Richmond [1970: 95] quotes Piaget as follows, " '... without interchange of thought and co-operation with others the individual would never come to group his operations into a coherent whole ...' (Intelligence, p. 163.)" and specifically advocates group work.

McNally [1977: 87] says "The facilitative effects of interpersonal interaction on intellectual development are important for all levels of thought but become particularly important in adolescence with the development of formal thinking."

One aspect of intellectual development, from babyhood, involves the learner's concept of self, of others and ultimately of groups and interaction. Initially, the infant has no real concept of either himself as an entity or others. In time he realises that objects have permanence, the objects can be people and eventually, at about the age of 6 or 7, that he himself is also an object. An extremely important stage is reached when the child is able to imagine things from someone else's perspective, read social signs and master the concepts of social acceptance and rules. This has implications for learning as it is only by recognising a difference in opinions and perspectives that a learner will be motivated to accommodate them in his existing schema. Hence, we can trace a progression from subjective to the objective to intersubjective, although at all times the concept of self is present. This final stage of intersubjective understanding is probably never entirely mastered and is linked to emotional maturity as well as intellectual

maturity. It is this skill that is extended and exercised during collaborative learning and virtual teamwork that is the subject of the research in this thesis.

Piaget's concept of the stages of intellectual development has immense importance for what material is presented to learners at specific times during their education. Material which is too advanced, because relevant schemas are not available, will simply be ignored. Material that is too simple will cause boredom and the learner will lose interest. Hence, the material must be moderately novel and above all the learner must be able to relate it to his previous experience and current cognitive structures [McNally, 1977: 11-12].

1.5.1.4 The sociocultural learning theory of Vygotsky

Socioculturalists believe that the ideal of subjective interpretations of reality being as close as possible to a universal objective reality is in fact not desirable. They emphasise the fact that a learner will only readily accept and understand concepts that he can relate to his own environment, culture and history. Each individual will also have a unique interpretation of reality which reflects his unique lifeworld. Vygotsky emphasises a social origin for learning. Thought is a form of "inner dialogue" modelled on the interaction between people [Thomas, 2000]. Learning is seen as a social process that involves human beings in communication with one another. Hence, he advises that teams should be made up of more advanced learners and less advanced learners so that the learners can learn from one another. This is, therefore, a model that fits in with an intersubjective view of social and physical reality. Despite the fact that both sociocultural learning theory and constructivism are both cognitive models, they differ with respect to how closely the subjective interpretation should coincide with objective reality.

1.5.2 Bloom's taxonomy

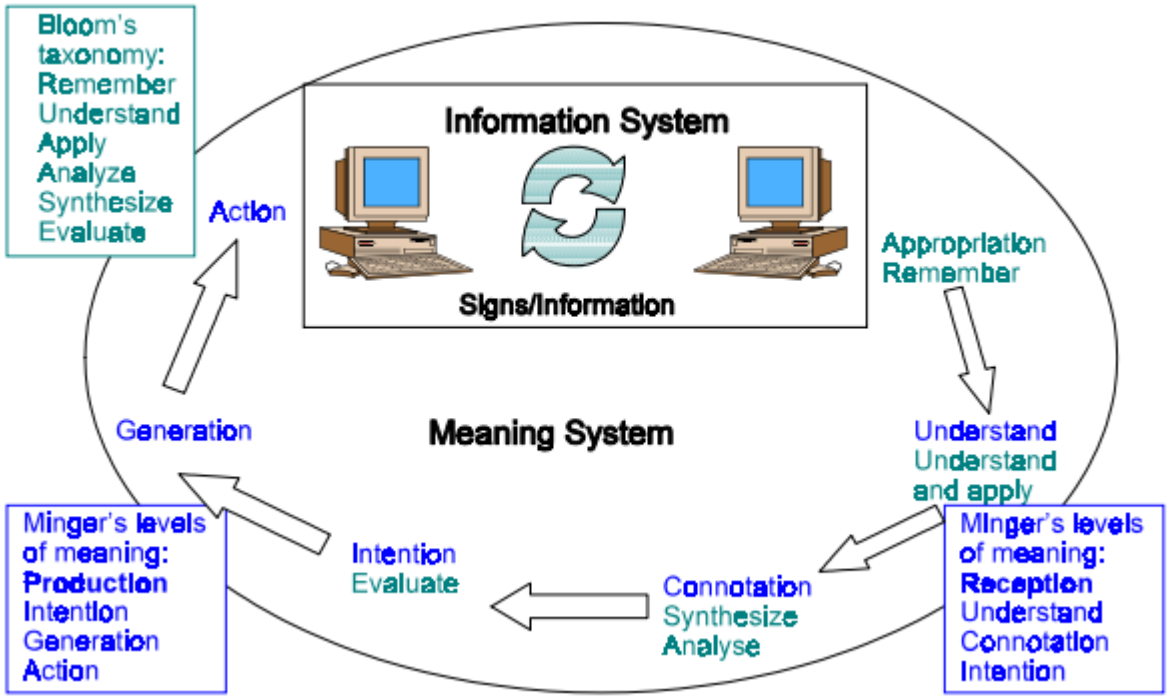
Bloom's taxonomy of educational objectives [1956] is widely accepted and consists of two groups of objectives. The first is knowledge and the second consists of intellectual abilities and skills. He defines knowledge as "... recall of specifics and universals, the recall of methods and processes, or the recall of a pattern, structure or setting." [Bloom, 1956: 201]. This includes knowledge of terminology, specific facts, conventions, trends and sequences, classifications and categories, criteria, methodology, principles and generalisations, theories and structures. Thus, this level includes the recall of highly complex and abstract concepts. Since the use of "knowledge" to identify this category of educational objective is somewhat misleading, I will

refer instead to "recall". The next group of objectives are those in which meaning is applied or constructed. "The abilities and skills objectives emphasize the mental processes of organizing material to achieve a particular process." [Bloom, 1956: 204] The abilities and skills in this group are understanding (comprehension), application, synthesis, analysis, and evaluation. Knowing the rules of a game such as chess is at the lowest level, memorisation. Being able to actually play the game is at the next level. Being able to play well, by foreseeing the possible results of a move which may only be significant several moves ahead and thus planning strategies by planning several moves, would demonstrate the ability to synthesize information and analyse it. Choosing between different strategies shows an ability to evaluate.

The process preceding learning is the subconscious process of appropriation where information is decoded and part of it is selected for attention and further processing. Learning theories, such as that of Bloom, can be used in analysing the levels of constructing meaning which follow appropriation. The first level of learning involves remembering facts, at least in short-term memory. Meaningless data can be remembered in the short term but they exist as a collection of unrelated facts which are difficult to recall. The new information must be related to existing information in long-term memory in order for it to be accessible in future. Even if someone can recall an observed fact that seemed completely meaningless when it was observed and there is no explanation as to why that fact was remembered, it will be remembered in some context. This shows that some interpretation and association (some development of meaning) precedes any learning.

This appropriation is followed by a level of learning during which the learner applies the facts. Analysis involves further interpretation of the information. Synthesis involves associating it with the previously acquired information. As was pointed out above, there is a certain amount of inherited information accessed by means of links between the new information and existing information. ("Nested in" is the term used by Mingers [1995] which he attributes to Dretske.) The highest level of Bloom's taxonomy is evaluation. Here inconsistencies or contradictions between the new information and existing knowledge are recognised and an attempt is made to reconcile them. During the reconciliation, the original, lifeworld knowledge may be modified or the incoming information may be queried (practical discourse) until the inconsistencies can be resolved. This critical process is an essential process of learning and understanding.

During any learning, aspects of more than one of these learning processes will be required and there is a need to interpret information and associate it with existing information during all of them.



1.6 Unit Summary

Anything that is recorded is data. Observations and facts are data. Anecdotes and opinions are also data, of a different kind. Data can be numbers, like the record of daily weather, or daily sales. Data can be alphanumeric, such as the names of employees and customers.

Definitions of information depend on the way in which the term "data" is defined. The major point of difference is whether information can be produced by an automated process and how this information, which is also digital, recorded and can be transmitted, differs from data.

What is knowledge? When someone memorises information this is often referred to as 'rote-learning' or 'learning by heart. We can then say that they have acquired some knowledge. Another form of knowledge is produced as a result of understanding information that has been given to us and using that information to gain knowledge of how to solve problems.

1.7 Check Your Progress

1) Discuss all three concepts of Data, Information, and Knowledge in detail.

Unit 2: Business Analytics: Meaning

2.0 Unit Introduction

Business analytics is believed to be a huge boon for organizations since it helps offer timely insights over the competition, helps optimize business processes, and helps generate growth and innovation opportunities. As organizations embark on their business analytics initiatives, many strategic questions, such as how to operationalize business analytics in order to drive the most value, arise. Recent Information Systems (IS) literature have focused on explaining the role of business analytics and the need for business analytics.

2.1 Unit Objective

This intends to introduce the learners to:
Business Analytics: Meaning
A Brief History of Data Analysis
Data Scientist vs. Data Engineer vs. Business Analyst
Career in Business Analytics
Data Science: Application
Roles & Responsibilities of a Data Scientist

2.2 Business Analytics: Meaning

Business analytics refers to the generation and use of knowledge and intelligence to apply data-based decision-making to support an organization's strategic and tactical business objectives

(Goes, 2014; Stubbs, 2011).

Business analytics includes "decision management, content analytics, planning and forecasting, discovery and exploration, business intelligence, predictive analytics, data and content management, stream computing, data warehousing, information integration and governance" (IBM, 2013, p. 4).

Business analytics has been the hot topic of interest for researchers and practitioners alike due to the rapid pace at which economic and social transactions are moving online, enhanced algorithms that help better understand the structure and content of human discourse, ready availability of large-scale data sets, relatively inexpensive access to computational capacity, a proliferation of user-friendly analytical software, and the ability to conduct large scale experiments on social phenomena (Agarwal & Dhar 2014).

IBM estimates that the market for data analytics is estimated to be $187 billion by the end of the year 2015 (IBM, 2013). Although business analytics promises enhanced organizational performance and profitability, improved decision-making processes, better alignment of resources and strategies, increased speed of decision-making, enhanced competitive advantage, and reduced risks (Computerworld, 2009; Goodnight, 2015; Harvard Business Review Analytics Report, 2012), implementation success is far from assured.

Business analytics (BA) is a set of disciplines and technologies for solving business problems using data analysis, statistical models and other quantitative methods. It involves an iterative, methodical exploration of an organization's data, with an emphasis on statistical analysis, to drive decision-making.

Data-driven companies treat their data as a business asset and actively look for ways to turn it into a competitive advantage. Success with business analytics depends on data quality, skilled analysts who understand the technologies and the business, and a commitment to using data to gain insights that inform business decisions.

How business analytics works:

Before any data analysis takes place, BA starts with several foundational processes:

- Determine the business goal of the analysis.
- Select an analysis methodology.
- Get business data to support the analysis, often from various systems and sources.
- Cleanse and integrate data into a single repository, such as a data warehouse or data mart.

The initial analysis is typically performed on a smaller sample data set of data. Analytics tools range from spreadsheets with statistical functions to complex data mining and predictive modelling applications. Patterns and relationships in the raw data are revealed. Then new questions are asked, and the analytic process iterates until the business goal is met.
Deployment of predictive models involves a statistical process known as scoring and uses records typically located in a database. Scores help enterprises make more informed, real-time decisions within applications and business processes.

BA also supports tactical decision-making in response to unforeseen events. Often the decision-making is automated using artificial intelligence to support real-time responses.
Types of business analytics
Different types of business analytics include the following:

- descriptive analytics, which tracks key performance indicators (KPIs) to understand the present state of a business;
- predictive analytics, which analyzes trend data to assess the likelihood of future outcomes; and
- prescriptive analytics, which uses past performance to generate recommendations for handling similar situations in the future.

Some schools of thought also include a fourth approach, diagnostic analytics, which is like descriptive analytics. It analyzes the state of a business and diagnoses why certain events or outcomes happened.

BUSINESS ANALYTICS IS MAKING DECISIONS SUBJECT TO UNCERTAINTY
Decisions are made in every area of directing a business. What is the long-term strategy of the

company? Does the firm need to hire? How are cash flow problems handled? Under what conditions does the company obtain financial backing for capital development? How should the company's products and services be marketed? How should the company conduct Research and Development? Will customer demand continue? What is the lost opportunity if the company fails to meet customer demand? How is the capacity to be managed? Does the company build, lease, subcontract, expand its present facility, relocate, and automate? Will competitors also increase capacity? Thus, managers must continuously make decisions about investments, products, resources, suppliers, financing, marketing methods, and many other items. Suppose you are a brand manager in your company's snack foods division. After a number of successful years in the market, your snack product has matured and sales have begun to decline. A brand extension has been proposed. How do you decide if this is the right course of action? Or perhaps you work for an electronics superstore chain that wants to expand. You've been assigned the task of choosing where to locate new stores. Your team is generating lots of ideas for sites. How do you know when to call a halt to the research and select from among the various alternatives? These and many other decisions are determined every day in offices around the globe. The difficulty in making such decisions arises from the uncertainty about the consequences of the decisions. Risk, the likelihood of incurring negative consequences, such as a loss, is inevitably involved in undertaking these decisions. Thus risk assessment and risk management are critical to effective management. Stephen Jay Gould said that "Misunderstanding of probability may be the greatest of all impediments to scientific literacy." The aim of this book is to give students—through simulation—a better understanding of probability and how to make decisions in the face of the pervasive uncertainty we all face.

2.3 A Brief History of Data Analysis

In order to understand what Data Analysis can and cannot do today, it is worth looking at how things have got to where they are today. But before we start, it is worth considering the term Data Analysis, although it has no SI definition. From my perspective at least it is the process of gaining a deeper understanding of things and garnering useful information from data. That data is simply the result of record keeping of various sorts.

Record keeping in Stone Age times would not have been trivial, and it is probably not too surprising that there are not many examples extant today. The opportunities for humour

abound, as we think of how cavemen would have had to do things that we do today. It is no surprise that The Flintstones was the most popular TV cartoon franchise for more than three decades, and even now is only beaten by The Simpsons. Apart from the hard work of creating records, Fred would have had to manually sort through a pile of heavy stone tablets in order to produce the sort of information that the simplest computer applications can produce today.

And as is so often the case, people pondered these difficulties and worked on solutions, and in due course, there was a giant technological leap – clay tablets were used instead of stone. This technology meant that records were much easier to produce, but performing any sort of analysis of the stored data would have again involved a hugely manual process of physically sorting through the saved clay tablets.

No ancient novels, unfortunately, but definitely records of what kings did – curiously, there is often more than a suggestion of fiction in the way that their exploits have been recorded! And then apart from the boring old laws, there are records of business transactions, and although it may pains us to consider it, tax details have been recorded.

There is no halting progress, and in due course, paper made from papyrus reeds burst onto the scene about 5,000 – 6-000 years ago, initially in scroll form, but then getting bound into book form.

A huge number of issues with the creation and storage of records were eliminated with the advent of paper. But analysis of that data would still have been far from straightforward.

Other concepts, such as that of zero came into place over thousands of years. Zero itself was only invented about 1400 years ago in India and took some 500 years to reach the West (Pythagoras was able to devise his theorem without having a functional zero). A few prior civilisations did have a sort of zero in use, but not rigorous enough to be a number of equal importance to other numbers. There are more than a few folks who attribute many of the great advances that have been made in the last 1400 years to this invention.

Various counting systems were used by different civilisations, vestiges of which still remain with us. For instance, we can thank the Sumerians for their use of 60, which means that we split hours and minutes into sixtieths.

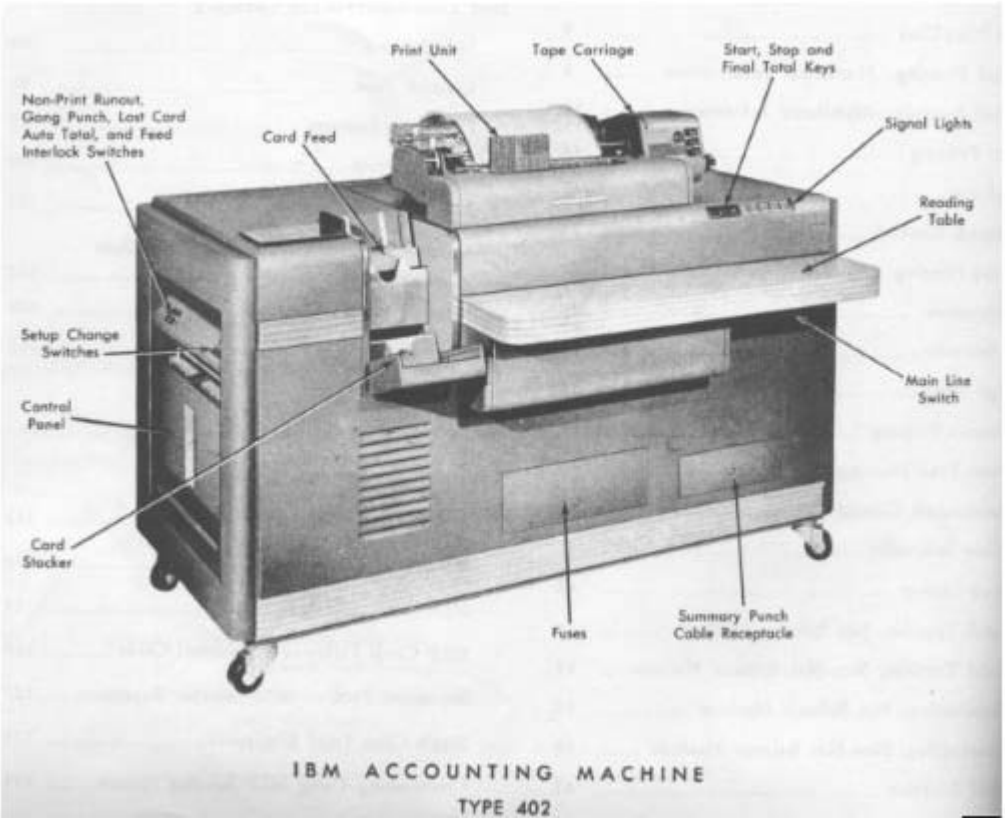
But in the West we settled on the decimal system of numbers (borrowed from other civilisations), and having rounded this off nicely with a proper zero, we had a robust system of numbers ready for some real work. Accounting got a solid base with the invention in Croatia of double entry bookkeeping almost exactly 600 years ago.
A clerk at a bench with a fat ledger and a quill is how accounts were done for hundreds of years. The analysis would have been very cumbersome, and this would not have been helped by the

use of a non-decimal monetary system. A decimal currency gives a huge boost to an analysis by making calculations so much easier.

Despite any claims by my children to the contrary, I have not seen in practice any of the methods of recording data mentioned so far. I am acquainted with the terms like 'groat', 'farthing' and 'sovereign', but although I was born and spent my early childhood in a country that used pounds, shillings and pence, I have not used these in calculations, as decimalisation arrived there shortly after I started school, some years before the British currency was decimalised.

The earlier part of the last century saw the rise of electro-mechanical accounting machines. These machines pretty much sought to duplicate a manual ledger, with a card being the direct equivalent of a ledger page. The cards had a magnetic stripe down the side on which some key bits of information were stored, depending on the application. There was always a counter to keep track of which line was to be printed next, and a running total for the card, or sometimes more than one running total. Totals for the ledger could be calculated by running all the cards through the machine. A deeper analysis than this was not easy. To get sales by category, you might need to manually sort all the cards, and add up all the cards within each stock category, although some of the top-end machines could do it a little easier than this – so long as you had simple stock categories!

These machines were a huge improvement on the totally manual methods that had been in use previously, but still had inherent problems, and analysis had to be planned when the program was written, with no flexibility. Even when everything worked well, it was still cumbersome.



Then computers took over the job of storing business transactions. I was first involved in the computer industry a few years after the IBM PC had been launched, but before they were routinely networked. At that time where I was, the old mechanical accounting machines were still being used in a few places, but most companies had already changed across to proper computers.

I developed accounting systems for networked computers, using a variety of languages including Cobol and Basic. As networking PCs became the norm with the advent of Windows 3.1, they became a valuable platform for even the smallest of organisations. But there were issues. On the system that I was developing on at the time, there was only one flavour of Cobol, and one flavour of Basic. The data was stored using a separate ISAM (Indexed Storage Access Method), which meant that the data could be accessed by either Cobol or Basic, or whatever other language you chose to use on this system. However, over in the PC-world, different varieties of Cobol were not compatible, and you could be fairly certain that the code that was written in one variety could not be used in another variety without some fairly significant changes, and the data that was stored by one variety could not be accessed by another.

Networked computers were mostly proprietary and their networks did not talk to each other. There were tricks that could be done to pass data between systems, and I was involved in the development of a system that produced an output 80 bytes wide. This output was than fed down the phone lines into another system that had been told to listen for a card reader on that line, and so was able to import the data.
I remember the excitement when ODBC (Open Database Connectivity) was announced – okay, so not everybody felt any excitement, only some techies. It sounded as though any development language would be able to talk to any data if you had the right drivers. Unfortunately, the early reality did not quite match the initial hype, but what was important was that different computer companies were talking, and were working together to develop standards that would facilitate the interoperability of various systems.

What also became clear at this time was that a server would look after the data, and networked intelligent devices would handle the UI (User Interface). The traditional model up to this point had been a server that did everything, and dumb terminals that had only enough processing power to talk to the server. With the advent of PCs, it almost seemed as though all processing might be done on local devices. But the importance of sharing data between these local devices highlighted the need for a network, and within the network, it made sense for one device to be responsible for being available all the time and keeping the data safe, so we quickly got back to the necessity of having a server.

In the 1980's, I was regularly producing what were called at the time Management Reports. These were reports that were user-customisable (in a small way) that provided a variety of analyses of the data in their accounting systems. The term Decision Support Systems has also come into vogue and waned, and for some years now the accepted description has been Business Intelligence. One reason for the change in name is that there was a huge amount of change going on, and companies wanted to give an indication that they were offering something new that took advantage of the new developments. The rate of change these days is significantly reduced, although change is still the only constant. I have no doubt at all that a new term will replace BI in the near future.

The company for whom I worked at the time had also supplied a new system to Toys R'Us, and

produced a marketing report about their success with this. With the aid of analysis from this new accounts package, Toys R'Us had been able to identify a toy that was just starting to become very popular, and they had been able to order stock so that they never ran out as the craze peaked, and then when the popularity was fading, they had been able to identify this early and cool their ordering so that they did not end up with piles of unsold stock. This would still be a good outcome these days, but it would not be that remarkable. At that time it was pretty much ground-breaking.

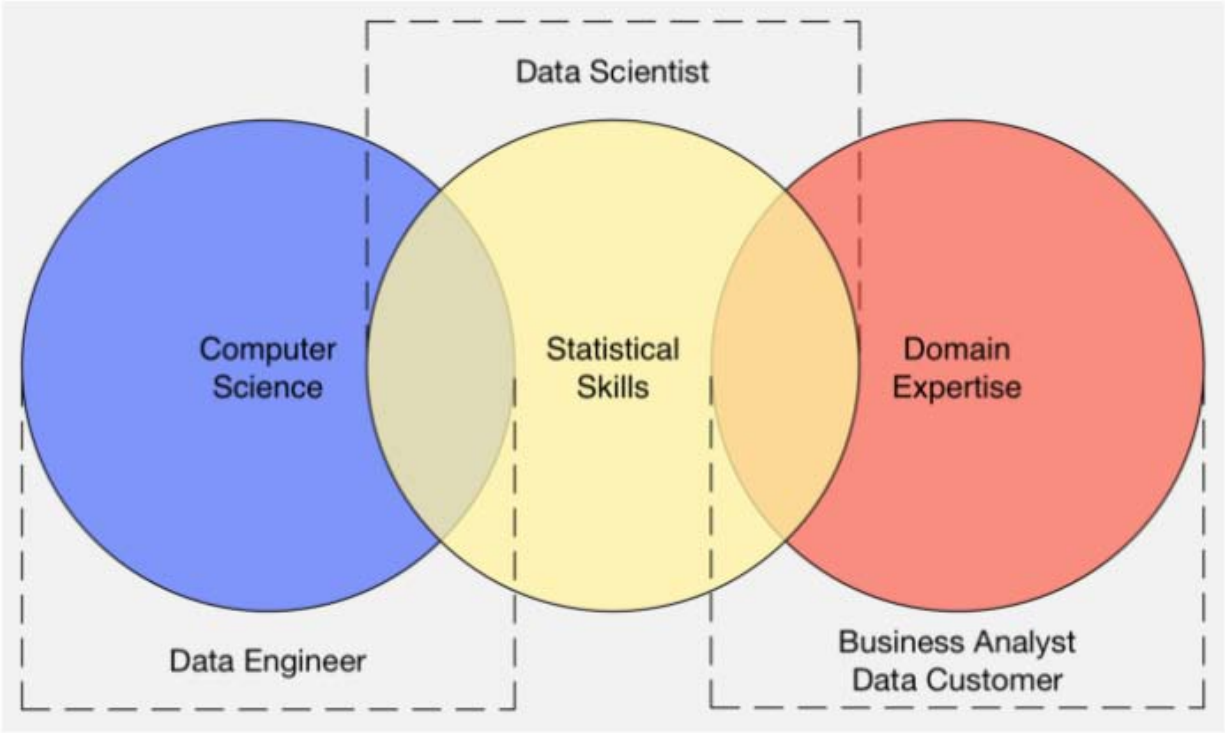
At a very basic level, the architecture of a typical BI system that I might create today remains unchanged over several decades – extracts from the Live Database are made, and imported into a Reporting Database. Additional data may be imported to add to this imported Live data, such as market share data that is purchased from suppliers of such data. These days, the term Big Data is spoken about a great deal, and may mean that data from social media is also extracted to add to the Reporting Database.

Once all the data that is required has been accumulated, then some sort of calculation process is run on the data. Having done all this, analysis may be performed. In my experience, part of the calculation process is likely to involve populating an OLAP cube. An OLAP cube cannot deliver anything that an OLTP database cannot produce, so it might seem at first glance that a cube is redundant. In practice, an OLAP cube is able to produce answers in a fraction of the time that an OLTP database could produce those answers. One major reason for this is that an OLAP cube does a number of aggregations when it prepares itself for use, so that when it is asked complex questions, it probably has a number of parts of the answer already calculated, and only has a small amount of work to do in finishing things off.

2.4 Data Scientist vs. Data Engineer vs. Business Analyst

Data scientists use their advanced statistical skills to help improve the models the data engineers implement and to put proper statistical rigour on the data discovery and analysis the customer is asking for. Data Scientist is the one who analyses and interprets complex digital data. While there are several ways to get into a data scientist's role, the most seamless one is by acquiring enough experience and learning the various data scientist skills. These skills include advanced statistical analyses, a complete understanding of machine learning, data conditioning etc.

Data Engineer either acquires a master's degree in a data-related field or gather a good amount of experience as Data Analyst. A Data Engineer needs to have a strong technical background with the ability to create and integrate APIs. They also need to understand data pipelining and performance optimization.



Business/Data Analyst: Most entry-level professionals interested in getting into a data-related job start off as Data analysts. Qualifying for this role is as simple as it gets. All you need is a bachelor's degree and good statistical knowledge. Strong technical skills would be a plus and can give you an edge over most other applicants. Other than this, companies expect you to understand data handling, modeling and reporting techniques along with a strong understanding of the business.

Data Analyst vs Data Engineer vs Data Scientist Skill Sets		
Data Analyst	**Data Engineer**	**Data Scientist**
Data Warehousing	Data Warehousing & ETL	Statistical & Analytical skills
Adobe & Google Analytics	Advanced programming knowledge	Data Mining
Programming knowledge	Hadoop-based Analytics	Machine Learning & Deep learning principles
Scripting & Statistical skills	In-depth knowledge of SQL/ database	In-depth programming knowledge (SAS/R/ Python coding)
Reporting & data visualization	Data architecture & pipelining	Hadoop-based analytics
SQL/ database knowledge	Machine learning concept knowledge	Data optimization
Spread-Sheet knowledge	Scripting, reporting & data visualization	Decision making and soft skills

2.5 Career in Business Analytics

Business Analytics jobs come in all kinds of flavors.

- **Data Architect**: Data architects are senior-level professionals who use extensive designing tools to develop databases for organizations. They ensure that all information, such as financial or marketing records, is accessible to all employees in an organization. Data architects must also be problem solvers, who innovate new ideas and change along with the evolving technology.
- **Data Analyst:** Every organization collects a large amount of data, and the job of data analysts is to collect and process this data to get useful information, which helps in improving business strategy and making better business decisions. In today's business world, data analysts play a vital role in helping businesses operate effectively with great decision-making skills.
- **Chief Data Officer**: A chief data officer is a senior-level job role where one has to look after the organization's data, draw valuable insights from it, and work with analytics and data science departments. A chief data officer also works to increase business performance by tapping into new and innovative sources of data. To be a good chief data officer, one needs to have relevant skills, such as mathematics and statisticians, plus a specialized master's degree, and excellent expertise in the IT industry.

- **Market Research Analyst:** The job of a market research analyst is to gather complex reports, surveys, feedback, and opinion polls in order to understand market opportunities to help an organization deliver its products and services in the best way possible. A market research analyst works on customer needs and satisfaction. A market research analyst should also have good knowledge of monitoring and forecasting market trends.
- **Data Scientist**: Data scientists are responsible for extracting meaningful information from raw, unstructured data to improve organizational business strategy and operations by making use of various statistical methods and algorithms. Data scientist has been featured as one of the "25 Best Jobs in America" for a couple of years in Glassdoor.
- **Statistician**: Statisticians compile, analyze, and understand quantitative information from surveys and experiments and help organizations reach operational standards. Statisticians apply mathematical and statistical methods to solve real-world problems in almost all fields including business, engineering, healthcare, etc.
- **Information Security Analyst**: Information security analysts secure data and monitor IT systems and networks to protect organizations and ensure cybersecurity. The main responsibility of these professionals includes finding the weaknesses in a system and coming up with new ways to strengthen the network. To become an information security analyst, one must have a bachelor's degree related to information technology, along with several years of relevant experience and any of the cyber security courses.

All organizations need business analysts to make meaningful decisions and enhance growth. Business analytics is a vast field with a number of career opportunities. We hope this blog has helped you in finding your desired career path in the field of business analytics.

AREAS	EMPLOYERS	STRATEGIES
BUSINESS ANALYTICS Data Collection/Data Mining Experiment Design Data Analysis Decision Analysis and Modeling Predictive Analytics Customer Loyalty and Selection Programs Marketing Strategy Development Fraud Detection Applied Statistics Process Optimization Operations Research/Management Manufacturing Design Supply Chain Management Information Technology Database Administration Program/Project Management Consulting	Nearly all industries have need for business analytics including: Retail, online retail Software and technology Telecommunications Financial services and banking Insurance Manufacturing Consumer products Transportation Consulting Entertainment Hospitality Healthcare Government/Public sector Nonprofit organizations	Seek broad exposure to business principles while honing statistics and quantitative skills. Gain relevant experience through an internship in an industry of interest. Develop excellent information technology, database management, and programming skills. Learn to use relevant software or tools such as Apache Hadoop, SQL, and SPSS. Earn industry certifications, e.g. SAS and Google, when possible. Learn to work effectively on interdisciplinary teams and how to communicate data intensive information to colleagues. Hone presentation skills. Develop strong analytical skills and a logical approach to problem solving. Get involved in campus organizations and seek leadership roles. Conduct informational interviews with professionals to learn about various industries or functional areas because business analytics professionals can fit into a wide array of positions. Consider earning a master's degree to qualify for advanced opportunities. Stay abreast of industry developments through professional societies and websites dedicated to business analytics, data mining, information technology, or other relevant topics.

2.6 Data Science: Application

Data science is the science of extracting knowledge from data. In other words, it is a science of drawing out hidden patterns amongst data using statistical and mathematical techniques. It employs techniques and theories drawn from many fields from the broad areas of mathematics, statistics, and information technology including machine learning, data engineering, probability models, statistical learning, pattern recognition and learning, etc. Data Scientist works on massive datasets for weather predictions, oil drillings, earthquake prediction, financial frauds, terrorist networks and activities, global economic impacts, sensor logs, social media analytics, customer churn, collaborative filtering(prediction about interest on users), regression analysis, etc. Data science is multi-disciplinary. Refer to the Figure given ahead.

Retail Analytics

Retail analytics is an umbrella term that comprises various elements which assist with decision-making in the retail business. Typically, this includes data collection and storage (data warehousing), data analysis that involves some statistical or predictive modeling, and decision-making. Traditionally, the analysis of data was limited to monitoring and visualizing some key performance indicators (KPIs) retrospectively. One may use the term business intelligence to refer to the gamut of activities that underlie intelligent business decision-making. However, typically this term is used to refer to the collection and presentation of historical information in

an easyto-understand manner, via reports, dashboards, scorecards, etc. The term advanced analytics is typically reserved for when predictive modeling is applied to data via statistical methods or machine learning. Our focus in this chapter will be on the later, advanced analytics, methodologies that can significantly assist in the decisionmaking process in retail. To understand the role analytics plays in retail, it is useful to break down the business decisions taken in retail into the following categories: consumer, product, workforce, and advertising.

Marketing analytics

Marketing analyticscan help firms realize the true potential of data and explore meaningful insights. Marketing analytics can be defined as a "high technologyenabled and marketing science model-supported approach to harness the true values of the customer, market, and firm level data to enhance the effect of marketing strategies" (Kumar and Sharma 2017; Lilien 2011). Basically, marketing analytics is the creation and use of data to measure and optimize marketing decisions. Marketing analytics comprises tools and processes that can inform and evaluate marketing decisions right from product/service development to sales (Farris et al. 2010; Venkatesan et al. 2014). According to CMO survey, spending on analytics will increase from 4.6% to 22% in the next 3 years.1 This shows the increasing importance of analytics in the field of marketing. Top marketers no longer rely on just intuition or past experience to make decisions. They want to make decisions based on data. But in the same survey, "Lack of processes or tools to measure success through analytics" and "Lack of people who can link to marketing practice" have been cited as the top two factors that prevent marketers from using advanced marketing analytic tools in the real world. This chapter is a step toward closing these two gaps. We present tools that both inform and measure the success of marketing activities and strategies. We also hope that this will help current and potential marketers get a good grasp of marketing analytics and how it can be used in practice (Lilien et al. 2013; Kumar et al. 2015). Since analytics tools are vast in numbers and since it is not feasible to explain the A–Z of every tool here, we select a few commonly used tools and attempt to give a comprehensive understanding of these tools. Interested readers can look at the references in this chapter to get more advanced and in-depth understanding of the discussed tools. The processes and tools discussed in this chapter will help in various aspects of marketing such as target marketing and segmentation, price and promotion, customer valuation, resource allocation, response analysis, demand assessment, and new product development. These can be applied at the following

levels: • Firm: At this level, tools are applied to the firm as a whole. Instead of focusing on a particular product or brand, these can be used to decide and evaluate firm strategies. For example, data envelopment analysis (DEA) can be used for all the units (i.e., finance, marketing, HR, operation, etc.) within a firm to find the most efficient units and allocate resources accordingly. • Brand/product: At the brand/product level, tools are applied to decide and evaluate strategies for a particular brand/product. For example, conjoint analysis can be conducted to find the product features preferred by customers or response analysis can be conducted to find how a particular brand advertisement will be received by the market. • Customer: Tools applied at customer level provide insights that help in segmenting and targeting customers. For example, customer lifetime value is a forward-looking customer metric that helps assess the value provided by customers to the firm (Fig. 19.1).

Data analytics in finance

Data analytics in finance is a part of quantitative finance. Quantitative finance primarily consists of three sectors in finance—asset management, banking, and insurance. Across these three sectors, there are four tightly connected functions in which quantitative finance is used—valuation, risk management, portfolio management, and performance analysis. Data analytics in finance supports these four sequential building blocks of quantitative finance, especially the first three— valuation, risk management, and portfolio management. Quantitative finance can be dichotomized into two branches or subsets having mild overlaps. The first is the risk-neutral world or the Q-world, and the second, wherein data analytics is used extensively, is the risk-averse world or the P-world. Quant professionals in the Q-world are called the Q-quants, and those of the P-world are called P-quants. Before we delve into the methodology of data analysis in finance which we structure as a three-stage process in this chapter, we briefly highlight the

processes, methodologies, challenges, and goals of these two quantworlds and also look at the history and origins of these two dichotomized worlds of quantitative finance.

Social Media and Web Analytics

Social media has created new opportunities to both consumers and companies. It has become one of the major drivers of consumer revolution. Companies can analyze data available from the web and social media to get valuable insights into what consumers want. Social media and web analytics can help companies measure the impact of their advertising and the effect of mode of message delivery on the consumers. Companies can also turn to social media analytics to learn more about their consumers. This chapter looks into various aspects of social media and web analytics.

Healthcare Analytics

Ancient understanding of biology, physiology, and medicine was built upon observations of how the body reacted to external stimuli. This indirect approach of documenting and studying the body's reactions was available long before the body's internal mechanisms were understood. While medical advances since that time have been truly astounding, nothing has changed the central fact that the study of medicine and the related study of healthcare must begin with careful observation, followed by the collection, consideration, and analysis of the data drawn from those observations. This age-old approach remains the key to current scientific method and practice.

Pricing Analytics

One of the most important decisions a firm has to take is the pricing of its products. At its simplest, this amounts to stating a number (the price) for a single product. But it is often a lot more complicated than that. Various pricing mechanisms such as dynamic pricing, promotions, bundling, volume discounts, segmentation, bidding, and name-your-own-price are usually deployed to increase revenues, and this chapter is devoted to the study of such mechanisms. Pricing and revenue optimization is known by different names in different domains, such as revenue management (RM), yield management, and pricing analytics. One formal definition of revenue management is the study of how a firm should set and update pricing and product availability decisions across its various selling channels in order to maximize its profitability.

There are several key phrases in this definition: Firms should not only set but also update prices; thus, price setting should be dynamic and depend on many factors such as competition, availability of inventory, and updated demand forecasts. Firms not only set prices but also make product availability decisions; in other words, firms can stop offering certain products at a given price (such as the closing of low-fare seats on airlines) or offer only certain assortments in certain channels. Firms might offer different products at different prices across selling channels—the online price for certain products might be lower than the retail price! The application of pricing and revenue management analytics in business management began in the 1970s. Airline operators like British Airways (then British Overseas Airways Corp.) and American Airlines began to offer differentiated fares for essentially the same tickets. The pioneer of this technique, called yield management, was Bob Crandall. Crandall, who eventually became chief executive of American Airlines, spearheaded a revolution in airline ticket pricing, but its impact would be felt across industries. Hotel chains, such as Marriott International, and parcelling services, like United Parcel Service, have used it to great effect. These techniques have only become more refined in the decades since. The advent of big data has revolutionized the degree to which analytics can predict patterns of customer demand, helping companies adapt to trends more quickly than ever. Retail chains such as Walmart collect petabytes of data daily, while mobile applications like Uber rely on big data to provide the framework for their business model.

Supply Chain Analytics

A supply chain consists of all activities that create value in the form of goods and services by transforming inputs into outputs. From a firm's perspective, such activities include buying raw materials from suppliers (buy), converting raw materials into finished goods (make), and moving and delivering goods and services to customers (delivery). The twin goals of supply chain management are to improve cost efficiency and customer satisfaction. Improved cost efficiency can lead to a lower price (increases market share) and/or a better margin (improves profitability). Better customer satisfaction, through improved service levels such as quicker delivery and/or higher stock availability, improves relationships with customers, which in turn may also lead to an increase in market share. However, these twin goals have the potential to affect each other conversely. Improving customer satisfaction often requires a higher cost; likewise, cost reduction may lower customer satisfaction. Thus, it is a challenge to achieve both

goals simultaneously. Despite the challenge, however, those companies that were able to achieve them successfully (e.g., Walmart,Amazon, Apple, and Samsung) enjoyed a sustainable and long-term advantage over their competition (Simchi-Levi et al. 2008; Sanders 2014; Rafique et al. 2014).

2.7 Roles & Responsibilities of a Data Scientist

- Management: The Data Scientist plays an insignificant managerial role where he supports the construction of the base of futuristic and technical abilities within the Data and Analytics field in order to assist various planned and continuing data analytics projects.
- Analytics: The Data Scientist represents a scientific role where he plans, implements, and assesses high-level statistical models and strategies for application in the business's most complex issues. The Data Scientist develops econometric and statistical models for various problems including projections, classification, clustering, pattern analysis, sampling, simulations, and so forth.
- Strategy/Design: The Data Scientist performs a vital role in the advancement of innovative strategies to understand the business's consumer trends and management as well as ways to solve difficult business problems, for instance, the optimization of product fulfillment and entire profit.
- Collaboration: The role of the Data Scientist is not a solitary role and in this position, he collaborates with superior data scientists to communicate obstacles and findings to relevant stakeholders in an effort to enhance drive business performance and decision-making.
- Knowledge: The Data Scientist also takes leadership to explore different technologies and tools with the vision of creating innovative data-driven insights for the business at the most agile pace feasible. In this situation, the Data Scientist also uses initiative in assessing and utilizing new and enhanced data science methods for the business, which he delivers to senior management of approval.
- Other Duties: A Data Scientist also performs related tasks and tasks as assigned by the Senior Data Scientist, Head of Data Science, Chief Data Officer, or the Employer.

2.8 Unit Summary

Business analytics refers to the generation and use of knowledge and intelligence to apply data-based decision-making to support an organization's strategic and tactical business objectives

(Goes, 2014; Stubbs, 2011).

Unit 3: Introduction

Structure

3.0 Introduction

A database management system (DBMS) is referred to as a collection of related data. It usually also includes a set of programs to access the relevant data. The primary aim of DBMS is to store and retrieve the relevant information in an efficient and convenient manner. The database systems are designed to manage a large amount of information. Defining structures for data storage and mechanisms for data retrieval, both are included in database management. Additionally, DBMS also ensures the security of information stored. For example, a telephone directory is the simplest example of a database.

The DBMS is a general-purpose software system that facilitates the processes of *defining*, *constructing*, and *manipulating* databases for various applications. *Defining* a database involves specifying the data types, structures, and constraints for the data to be stored in the database. *Constructing* the database is the process of storing the data itself on some storage medium that is controlled by the DBMS. *Manipulating* a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the real world, and generating reports from the data. This unit focuses on the significance of databases, various merits and demerits of DBMS, and basic database system applications.

3.1 Unit Objectives

After completing this unit, the reader will be able to:

- Discuss the significance of the database.
- Understand the fundamentals of database system applications.
- Illustrate the advantages and disadvantages of DBMS.

3.2 Significance of Database

Indeed the importance of the database system has increased in recent years with significant developments in hardware capability, hardware capacity, and communications, including the emergence of the Internet, electronic commerce, business intelligence, mobile communications, and grid computing. We are already aware that a ***database*** is a collection of relevant data. ***Database management system*** (DBMS) is the software that manages and controls access to the database. A ***database application*** is a program that interacts with the database during its execution. The ***database system*** is a collection of application programs that interact with the database along with the DBMS and the database itself.

DBMS provides the following facilities:

- DBMS allows users to define the database, usually through a ***Data Definition Language (DDL)*** by specifying the data types, structures, and constraints on the data to be stored in the database.
- DBMS allows users to insert, update, delete, and retrieve data from the database, usually through ***Data Manipulation Language (DML)***. All the data and data descriptions have a central repository that allows the DML to provide a general inquiry facility to this data, called a ***query language.*** In a query language, the user has to work with a fixed set of queries, causing major software management problems. The most common query language is the ***Structured Query Language (SQL)***.
- DBMS provides controlled access to the database.
- When we analyze the information needs of an organization, we attempt to identify ***entities, attributes, and relationships.*** An ***entity*** is a distinct object (a person, place, thing, concept, or event) in the organization that is to be represented in the database. An ***attribute*** is a property that describes some aspect of the object that we wish to record, and a ***relationship*** is an association between entities.

3.2.1 Purpose of Database Systems

Database systems arose in response to early methods of computerized management of

commercial data. As an example of such methods, typical of the 1960s, consider part of a university organization that, among other data, keeps the information about all instructors, students, departments, and course offerings. One way to keep the information on a computer is to store it in operating system files. To allow users to manipulate the information, the system has a number of application programs that manipulate the files, including programs to:

- Add new students, instructors, and courses.
- Register students for courses and generate class rosters.
- Assign grades to students, compute grade point averages (GPA) , and generate transcripts.

New application programs are added to the system as the need arises. For example, suppose that a university decides to create a new major (say, computer science). As a result, the university creates a new department and creates new permanent files (or adds information to existing files) to record information about all the instructors in the department, students in that major, course offerings, degree requirements, etc. The university may have to write new application programs to deal with rules specific to the new major. New application programs may also have to be written to handle new rules in the university. Thus, as time goes by, the system acquires more files and more application programs.

This typical ***file-processing system*** is supported by a conventional operating system. The system stores permanent records in various files, and it needs different application programs to extract records from, and add records to, the appropriate files. Before database management systems (DBMSs) were introduced, organizations usually stored information in such systems.

Keeping organizational information in a file-processing system has a number of major disadvantages:

- **Data redundancy and inconsistency**

  Since different programmers create the files and application programs over a long period, the various files are likely to have different formats and the programs may be written in several programming languages. Moreover, the same information may be duplicated in several places (files). For example, the address and telephone number of a particular customer may appear in a file that consists of savings-account records and in a file that consists of checking-account records. This redundancy leads to higher storage and access cost. In addition, it may lead to data inconsistency; that is, the various copies of the same data may no longer agree. For example, a changed customer address may be

reflected in savings-account records but not elsewhere in the system. Difficulty in accessing data.

- **Difficulty in accessing data**

  Suppose that one of the bank officers needs to find out the names of all customers who live within a particular postal-code area. The officer asks the data-processing department to generate such a list. Because the designers of the original system did not anticipate this request, there is no application program on hand to meet it. There is, however, an application program to generate the list of all customers. The bank officer has now two choices: either obtain the list of all customers and extract the needed information manually or ask a system programmer to write the necessary application program. Both alternatives are obviously unsatisfactory. Suppose that such a program is written, and that, several days later, the same officer needs to trim that list to include only those customers who have an account balance of $10,000 or more. As expected, a program to generate such a list does not exist. Again, the officer has the preceding two options, neither of which is satisfactory. The point here is that conventional file-processing environments do not allow needed data to be retrieved in a convenient and efficient manner. More responsive data-retrieval systems are required for general use.

- **Data isolation**

  Because data is scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

- **Integrity problems**

  The data values stored in the database must satisfy certain types of *consistency constraints*. For example, the balance of a bank account may never fall below a prescribed amount (say, $25). Developers enforce these constraints in the system by adding appropriate code in the various application programs. However, when new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items from different files.

- **Atomicity problems**

  A computer system, like any other mechanical or electrical device, is subject to failure. In many applications, it is crucial that, if a failure occurs, the data be restored to the consistent state that existed prior to the failure. Consider a program to transfer $50 from account A to account B. If a system failure occurs during the execution of the

program, it is possible that the $50 was removed from account A but was not credited to account B, resulting in an inconsistent database state. Clearly, it is essential to database consistency that either both the credit and debit occur, or that neither occur. That is, the funds transfer must be atomic—it must happen in its entirety or not at all. It is difficult to ensure atomicity in a conventional file-processing system.

- **Concurrent-access anomalies**

  For the sake of the overall performance of the system and faster response, many systems allow multiple users to update the data simultaneously. In such an environment, the interaction of concurrent updates may result in inconsistent data. Consider bank account A, containing $500. If two customers withdraw funds (say $50 and $100 respectively) from account A at about the same time, the result of the concurrent executions may leave the account in an incorrect (or inconsistent) state. Suppose that the programs executing on behalf of each withdrawal read the old balance, reduce that value by the amount being withdrawn, and write the result back. If the two programs run concurrently, they may both read the value $500, and write back $450 and $400, respectively. Depending on which one writes the value last, the account may contain either $450 or $400, rather than the correct value of $350. To guard against this possibility, the system must maintain some form of supervision. But supervision is difficult to provide because data may be accessed by many different application programs that have not been coordinated previously.

- **Security problems**

  Not every user of the database system should be able to access all the data. For example, in a banking system, payroll personnel need to see only that part of the database that has information about the various bank employees. They do not need access to information about customer accounts. But, since application programs are added to the system in an ad hoc manner, enforcing such security constraints is difficult. These difficulties, among others, prompted the development of database systems. In what follows, we shall see the concepts and algorithms that enable database systems to solve the problems with file-processing systems. In most of this book, we use a bank enterprise as a running example of a typical data-processing application found in a corporation.

3.2.2 Components of DBMS Environment

Five major components in the DBMS environment can be identified: hardware, software, data, procedures, and people, as illustrated in Figure 1.1.

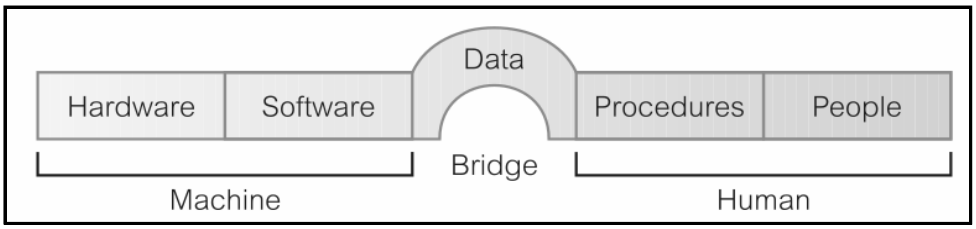


Figure 1.1 Components of DBMS environment

(**Source-** Database Systems: A Practical Approach to Design, Implementation, and Management, Thomas Connolly & Carolyn Begg, Pearson, 6th Edition, Chapter- 1, Page No.- 66)

Hardware

The DBMS and the applications require hardware to run. The hardware can range from a single personal computer to a single mainframe or a network of computers. The particular hardware depends on the organization's requirements and the DBMS used. Some DBMSs run only on particular hardware or operating systems, while others run on a wide variety of hardware and operating systems. A DBMS requires a minimum amount of main memory and disk space to run, but this minimum configuration may not necessarily give acceptable performance.

Software

The software component comprises the DBMS software itself and the application programs, together with the operating system, including network software if the DBMS is being used over a network. Typically, application programs are written in a third-generation programming language (3GL), such as C, C++, C#, Java, Visual Basic, COBOL, Fortran, Ada, or Pascal, or a fourth-generation language (4GL), such as SQL, embedded in a third-generation language. The target DBMS may have its own fourth-generation tools that allow rapid development of applications through the provision of non-procedural query languages, reports generators, forms generators, graphics generators, and application generators. The use of fourth-generation tools can improve productivity significantly and produce programs that are easier to maintain.

Data

The most important component of the DBMS environment is Data. In figure 1.1, we can observe that the data acts as a bridge between the machine components and human

components.

Procedures

Procedures are the instructions and rules that govern the design and use of the database. The documented procedures on how to use or run the system are required for the users of the system. These may consist of instructions on how to: Log on to the DBMS, Use a particular DBMS facility or application program, Start and stop the DBMS, Make backup copies of the database, Handle hardware or software failures. This may include procedures on how to identify the failed component, how to fix the failed component, and following the repair of the fault, how to recover the database, Change the structure of a table, reorganize the database across multiple disks, improve performance, or archive data to secondary storage.

People

The final component is the people or the users involved with the system. We can identify four distinct types of people who participate in the DBMS environment: data and database administrators, database designers, application developers, and end-users.

- Data and Database Administrators: The database and the DBMS are corporate resources that must be managed like any other resource. Data and database administration are the roles generally associated with the management and control of a DBMS and its data. *The Data Administrator (DA)* is responsible for the management of the data resource, including database planning; development and maintenance of standards, policies, and procedures; and conceptual/logical database design. The DA consults with and advises senior managers, ensuring that the direction of database development will ultimately support corporate objectives. The *Database Administrator (DBA)* is responsible for the physical realization of the database, including physical database design and implementation, security and integrity control, maintenance of the operational system, and ensuring satisfactory performance of the applications for users. The role of the DBA is more technically oriented than the role of the DA, requiring detailed knowledge of the target DBMS and the system environment. In some organizations there is no distinction between these two roles; in others, the importance of the corporate resources is reflected in the allocation of teams of staff dedicated to each of these roles.
- Database Designers: In large database design projects, we can distinguish between two types of designers: logical database designers and physical database designers. The

logical database designer is concerned with identifying the data (that is, the entities and attributes), the relationships between the data, and the constraints on the data that is to be stored in the database. The logical database designer must have a thorough and complete understanding of the organization's data and any constraints on this data (the constraints are sometimes called business rules). These constraints describe the main characteristics of the data as viewed by the organization. The *physical database designer* decides how the logical database design is to be physically realized. This involves mapping the logical database design into a set of tables and integrity constraints; selecting specific storage structures and access methods for the data to achieve a good performance; designing any security measures required on the data. The physical database designer must be capable of selecting a suitable storage strategy that takes account of usage. Whereas conceptual and logical database design is concerned with the *what*, physical database design is concerned with the *how*.

- Application Developers: Once the database has been implemented, the application programs that provide the required functionality for the end-users must be implemented. This is the responsibility of the application developers. Typically, the application developers work from a specification produced by systems analysts. Each program contains statements that request the DBMS to perform some operation on the database, which includes retrieving data, inserting, updating, and deleting data. The programs may be written in a third-generation or fourth-generation programming language.
- End-Users: The end-users are the "clients" of the database, which has been designed and implemented and is being maintained to serve their information needs.

3.3 Database System Applications

Databases form an essential part of every enterprise today, storing not only types of information that are common to most enterprises but also information that is specific to the category of the enterprise.

The Internet revolution of the late 1990s sharply increased direct user access to databases. Organizations converted many of their phone interfaces to databases into Web interfaces and made a variety of services and information available online. For instance, when you access an online bookstore and browse a book or music collection, you are accessing data stored in a

database. When you enter an order online, your order is stored in a database. When you access a bank website and retrieve your bank balance and transaction information, the information is retrieved from the bank's database system. When you access a Web site, information about you may be retrieved from a database to select which advertisements you should see. Furthermore, data about your Web accesses may be stored in a database.

Some applications are discussed below:

Telecommunication

For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

Airlines

For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.

Enterprise Information

- **Sales:** For the customer, product, and purchase information.
- **Accounting:** For payments, receipts, account balances, assets, and other accounting information.
- **Human resources:** For information about employees, salaries, payroll taxes, and benefits, and for the generation of paychecks.
- **Manufacturing:** For the management of the supply chain and for tracking the production of items in factories, inventories of items in warehouses and stores, and orders for items.
- **Online retailers:** For sales data noted above plus online order tracking, generation of recommendation lists, and maintenance of online product evaluations.
- *Banking and Finance*
- **Banking:** For customer information, accounts, loans, and banking transactions.
- **Credit card transactions:** For purchases on credit cards and generation of monthly statements.
- **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.

Universities

For student information, course registrations, and grades (in addition to standard enterprise

information such as human resources and accounting).

The application of a database system can be depicted in the following example for using the internet. Many of the sites on the Internet are driven by database applications. For example, you may visit an online bookstore that allows you to browse and buy books, such as Amazon.com. The bookstore allows you to browse books in different categories, such as computing or management, or by author name. In either case, there is a database on the organization's Web server that consists of book details, availability, shipping information, stock levels, and order history. Book details include book titles, ISBNs, authors, prices, sales histories, publishers, reviews, and detailed descriptions. The database allows books to be cross-referenced: for example, a book may be listed under several categories, such as computing, programming languages, bestsellers, and recommended titles. The cross-referencing also allows Amazon to give you information on other books that are typically ordered along with the title you are interested in. You can provide your credit card details to purchase one or more books online. Amazon.com personalizes its service for customers who return to its site by keeping a record of all previous transactions, including items purchased, shipping, and credit card details. When you return to the site, you might be greeted by name and presented with a list of recommended titles based on previous purchases.

3.4 Advantages of DBMS

The database management system has promising potential advantages.

- **Control of data redundancy:** The traditional file-based systems waste space by storing the same information in more than one file. In contrast, the database approach attempts to eliminate the redundancy by integrating the files so that multiple copies of the same data are not stored. However, the database approach does not eliminate redundancy entirely but controls the amount of redundancy inherent in the database. Sometimes it is necessary to duplicate key data items to model relationships; at other times, it is desirable to duplicate some data items to improve performance.
- **Data consistency:** By eliminating or controlling redundancy, we reduce the risk of inconsistencies occurring. If a data item is stored only once in the database, any update to its value has to be performed only once and the new value is available immediately to all users. If a data item is stored more than once and the system is aware of this, the system can ensure that all copies of the item are kept consistent. Unfortunately, many of today's DBMSs do not automatically insure this type of consistency.

- **More information from the same data:** With the integration of the operational data, it may be possible for the organization to derive additional information from the same data.
- **Sharing of data:** Typically, files are owned by the people or departments that use them. On the other hand, the database belongs to the entire organization and can be shared by all authorized users. In this way, more users share more of the data. Furthermore, new applications can build on the existing data in the database and add only data that is not currently stored, rather than having to define all data requirements again. The new applications can also rely on the functions provided by the DBMS, such as data definition and manipulation, and concurrency and recovery control, rather than having to provide these functions themselves.
- **Improved data integrity:** Database integrity refers to the validity and consistency of stored data. Integrity is usually expressed in terms of constraints, which are consistency rules that the database is not permitted to violate. Constraints may apply to data items within a single record or to relationships between records. The integration allows the DBA to define integrity constraints, and the DBMS to enforce them.
- **Improved security:** Database security is the protection of the database from unauthorized users. Without suitable security measures, integration makes the data more vulnerable than file-based systems. However, integration allows the DBA to define database security, and the DBMS to enforce it. This security may take the form of usernames and passwords to identify people authorized to use the database. The access that an authorized user is allowed on the data may be restricted by the operation type (retrieval, insert, update, delete). For example, the DBA has access to all the data in the database; a branch manager may have access to all data that relates to his or her branch office, and a sales assistant may have access to all data relating to properties but no access to sensitive data such as staff salary details.
- **Enforcement of standards:** Again, integration allows the DBA to define and the DBMS to enforce the necessary standards. These may include departmental, organizational, national, or international standards for such things as data formats to facilitate the exchange of data between systems, naming conventions, documentation standards, update procedures, and access rules.
- **The economy of scale:** Combining all the organization's operational data into one

database and creating a set of applications that work on this one source of data can result in cost savings. In this case, the budget that would normally be allocated to each department for the development and maintenance of its file-based system can be combined, possibly resulting in a lower total cost, leading to an economy of scale. The combined budget can be used to buy a system configuration that is more suited to the organization's needs. This may consist of one large, powerful computer or a network of smaller computers.

- **Balance of conflicting requirements:** Each user or department has needs that may be in conflict with the needs of other users. Because the database is under the control of the DBA, the DBA can make decisions about the design and operational use of the database that provide the best use of resources for the organization as a whole. These decisions will provide optimal performance for important applications, possibly at the expense of less-critical ones.
- **Improved data accessibility and responsiveness:** Again, as a result of integration, data that crosses departmental boundaries is directly accessible to the end-users. This provides a system with potentially much more functionality that can, for example, be used to provide better services to the end-user or the organization's clients. Many DBMSs provide query languages or report writers that allow users to ask ad hoc questions and to obtain the required information almost immediately at their terminal, without requiring a programmer to write some software to extract this information from the database.
- **Increased productivity:** The DBMS provides many of the standard functions that the programmer would normally have to write in a file-based application. At a basic level, the DBMS provides all the low-level file-handling routines that are typical in application programs. The provision of these functions allows the programmer to concentrate on the specific functionality required by the users without having to worry about low-level implementation details. Many DBMSs also provide a fourth-generation environment, consisting of tools to simplify the development of database applications. This results in increased programmer productivity and reduced development time (with associated cost savings).
- **Improved maintenance through data independence:** In file-based systems, the descriptions of the data and the logic for accessing the data are built into each

application program, making the programs dependent on the data. A change to the structure of the data—such as making an address 41 characters instead of 40 characters, or a change to the way the data is stored on disk—can require substantial alterations to the programs that are affected by the change. In contrast, a DBMS separates the data descriptions from the applications, thereby making applications immune to changes in the data descriptions. This is known as ***data independence***. The provision of data independence simplifies database application maintenance.

- **Increased concurrency:** In some file-based systems, if two or more users are allowed to access the same file simultaneously, it is possible that the accesses will interfere with each other, resulting in loss of information or even loss of integrity. Many DBMSs manage concurrent database access and ensure that such problems cannot occur.
- **Improved backup and recovery services:** Many file-based systems place the responsibility on the user to provide measures to protect the data from failures to the computer system or application program. This may involve performing a nightly backup of the data. In the event of a failure during the next day, the backup is restored and the work that has taken place since this backup is lost and has to be re-entered. In contrast, modern DBMSs provide facilities to minimize the amount of processing that is lost following a failure.

3.5 Disadvantages of DBMS

The disadvantages of the database approach are given below:

- **Complexity:** The provision of the functionality that we expect of a good DBMS makes the DBMS an extremely complex piece of software. Database designers and developers, data and database administrators, and end-users must understand this functionality to take full advantage of it. Failure to understand the system can lead to bad design decisions, which can have serious consequences for an organization.
- **Size:** The complexity and breadth of functionality make the DBMS an extremely large piece of software, occupying many megabytes of disk space and requiring substantial amounts of memory to run efficiently.
- **Cost of DBMSs:** The cost of DBMSs varies significantly, depending on the environment and functionality provided. For example, a single-user DBMS for a personal computer may only cost $100. However, a large mainframe multi-user DBMS servicing hundreds of users can be extremely expensive, perhaps $100,000 or even

$1,000,000. There is also the recurrent annual maintenance cost, which is typically a percentage of the list price.

- **Additional hardware costs:** The disk storage requirements for the DBMS and the database may necessitate the purchase of additional storage space. Furthermore, to achieve the required performance, it may be necessary to purchase a larger machine, perhaps even a machine dedicated to running the DBMS. The procurement of additional hardware results in further expenditure.
- **Cost of conversion:** In some situations, the cost of the DBMS and extra hardware may be relatively small compared with the cost of converting existing applications to run on the new DBMS and hardware. This cost also includes the cost of training staff to use these new systems, and possibly the employment of specialist staff to help with the conversion and running of the systems. This cost is one of the main reasons why some organizations feel tied to their current systems and cannot switch to more modern database technology. The term legacy system is sometimes used to refer to an older, and usually inferior, system.
- **Performance:** Typically, a file-based system is written for a specific application, such as invoicing. As a result, performance is generally very good. However, the DBMS is written to be more general, to cater to many applications rather than just one. The result is that some applications may not run as fast as they used to.
- **The greater impact of a failure:** The centralization of resources increases the vulnerability of the system. Because all users and applications rely on the availability of the DBMS, the failure of certain components can bring operations to a halt.

3.6 Summary

- A database is a shared collection of logically related data and a description of this data, designed to meet the information needs of an organization.
- A DBMS is a software system that enables users to define, create, maintain, and control access to the database.
- An application program is a computer program that interacts with the database by issuing an appropriate request (typically a SQL statement) to the DBMS.
- The DBMS environment consists of hardware (the computer), software (the DBMS, operating system, and applications programs), data, procedures, and people. The people

include data and database administrators, database designers, application developers, and end-users.

- Some prominent database system applications can be seen in the field of telecommunication, airlines, banking & financial services.
- Some advantages of the database approach include control of data redundancy, data consistency, sharing of data, and improved security and integrity. Some disadvantages include complexity, cost, reduced performance, and higher impact of a failure.

3.7 Key Terms

- **Data Definition Language (DDL):** DDL is the language provided by the DBMS to allow users to define the database.
- **Data Manipulation Language (DML):** DML is the language provided by the DBMS to allow users to insert, update, delete, and retrieve data from the database.
- **Structured Query Language (SQL):** It is a typical programming language used to operate databases, especially relational databases.
- **Entity:** An entity is a distinct object (a person, place, thing, concept, or event) in the organization that is to be represented in the database.
- **Attribute:** An attribute is a property that describes some aspect of the object that is to be recorded.
- **Data Redundancy:** It refers to the repetition of the same data at two different places in a common database.

3.8 Check Your Progress

Short- Answer Type

Q1) Define Database management system.

Q2) _____ is used to insert, update, delete, and retrieve data from the database.

Q3) Full form of SQL:

a) Structured Query Language
b) Semi Query Language
c) Structured Question Language
d) Series Query Language

Q4) DDL is the language provided by the DBMS to allow users to define the database. True/ False?

Q5) The DBMS is a general-purpose software system that facilitates the processes of:

a) Defining b) constructing c) manipulating d) All of the above

Long- Answer Type

Q1) State at least five advantages and disadvantages of DBMS.

Q2) Explain the purpose and significance of DBMS.

Q3) Define the following:

a) Database b) DDL c) DML

Q4) List different types of database users.

Q5) Write a short note on components of the DBMS environment.

References

- *Database Systems: A Practical Approach to Design, Implementation, and Management*, Thomas Connolly & Carolyn Begg, Pearson, 6th Edition.

Database System Concepts, Henry F Korth, Abraham Silberschatz, and S. Sudharshan, 6th Edition, McGraw Hill, 2011.

Unit 4: Different types of databases

Structure

4.0 Introduction

The classification of database systems is based on several criteria. The first important criterion is the data model on which DBMS is based. *The relational data model* is the most widely used data model for many commercial DBMSs. Various older applications run on the database systems based on *hierarchical and network data models*. Another data model that was limited to some commercial systems is the *object data model*. The *relational DBMSs* are evolving continuously, and have been incorporating many of the concepts that were developed in object databases. This has led to a new class of DBMSs called *object-relational DBMSs*. Hence DBMSs can be categorized on the basis of data models: relational, object, object-relational, hierarchical, network, and other.

The second criterion used to classify DBMSs is the number of users supported by the system. Single-user systems support only one user at a time and are mostly used with personal computers. Multiuser systems, which include the majority of DBMSs, support multiple users concurrently.

A third criterion is the number of sites over which the database is distributed. A DBMS is

centralized if the data is stored at a single computer site. A centralized DBMS can support multiple users, but the DBMS and the database themselves reside totally at a single computer site. A *distributed* DBMS (DDBMS) can have the actual database and DBMS software distributed over many sites, connected by a computer network. Homogeneous DDBMSs use the same DBMS software at multiple sites. A recent trend is to develop software to access several autonomous pre-existing databases stored under heterogeneous llBMSs. This leads to a federated DBMS (or multi-database system), in which the participating DBMSs are loosely coupled and have a degree of local autonomy. Many DBMSs use client-server architecture. This unit describes various data models and the basic types of databases, like relational, distributed, and centralized databases.

4.1 Unit Objectives

After completing this unit, the reader will be able to:

- Learn about the different types of data models.
- Describe the classification of databases.
- Illustrate the features of relational, centralized, and distributed databases.

4.2 Data Models

The data model is an integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization. A model is a representation of real-world objects and events and their associations. It is an abstraction that concentrates on the essential, inherent aspects of an organization and ignores the accidental properties. A data model represents the organization itself. It should provide the basic concepts and notations that will allow database designers and end-users to communicate unambiguously and accurately their understanding of the organizational data.

A data model can be thought of as comprising three components:

1. A structural part, consisting of a set of rules according to which databases can be constructed.
2. A manipulative part, defining the types of operations that are allowed on the data (this includes the operations that are used for updating or retrieving data from the database and for changing the structure of the database).

3. A set of integrity constraints, which ensures that the data is accurate.

   The purpose of a data model is to represent data and to make the data understandable. If it does this, then it can be easily used to design a database. There have been many data models proposed in the literature. They fall into three broad categories: *object-based, record-based, and physical data models.* The first two are used to describe data at the conceptual and external levels, the third is used to describe data at the internal level.

4.2.1 Object-based Data Models

Object-based data models use concepts such as entities, attributes, and relationships. An entity is a distinct object (a person, place, thing, concept, event) in the organization that is to be represented in the database. An attribute is a property that describes some aspect of the object that we wish to record, and a relationship is an association between entities. Some of the more common types of the object-based data model are:

- **Entity-Relationship (ER):**

  The entity-relationship (ER) data model uses a collection of basic objects, called entities, and relationships among these objects. An entity is a "thing" or "object" in the real world that is distinguishable from other objects. The entity-relationship model is widely used in database design.

- **Object-oriented:**

  Object-oriented programming (especially in Java, C++, or C#) has become the dominant software development methodology. This led to the development of an object-oriented data model that can be seen as extending the E-R model with notions of encapsulation, methods (functions), and object identity. The object-relational data model combines features of the object-oriented data model and relational data model.

The ER model has emerged as one of the main techniques for database design and forms the basis for the database design methodology used in this book. The object-oriented data model extends the definition of an entity to include not only the attributes that describe the state of the object but also the actions that are associated with the object, that is, its behavior. The object is said to encapsulate both state and behavior.

4.2.2 Record-based Data Models

In a record-based model, the database consists of a number of fixed-format records, possibly of differing types. Each record type defines a fixed number of fields, typically of a fixed length. There are three principal types of record-based logical data model: *the relational data model, the*

network data model, and the hierarchical data model. The hierarchical and network data models were developed almost a decade before the relational data model, so their links to traditional file processing concepts are more evident.

Relational data model

The relational data model is based on the concept of mathematical relations. In the relational model, data and relationships are represented as tables, each of which has a number of columns with a unique name. Figure 2.1 is a sample instance of a relational schema for part of a construction company, showing branch and staff details.

Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Figure 4.1 A sample instance of a relational schema

For example, it shows that employee John White is a manager with a salary of £30,000, who works at branch (branchNo) B005, which, from the first table, is at 22 Deer Rd in London. It is important to note that there is a relationship between Staff and Branch: a branch office has staff. However, there is no explicit link between these two tables; it is only by knowing that the attribute branchNo in the Staff relation is the same as the branchNo of the Branch relation that we can establish that a relationship exists.

Network data model

In the network model, data is represented as collections of records, and relationships are represented by *sets*. Compared with the relational model, relationships are explicitly modeled by

the sets, which become pointers in the implementation. The records are organized as generalized graph structures with records appearing as *nodes* (also called *segments*) and sets as *edges* in the graph. Figure 2.2 illustrates an instance of a network schema for the same data set presented in Figure 2.1.

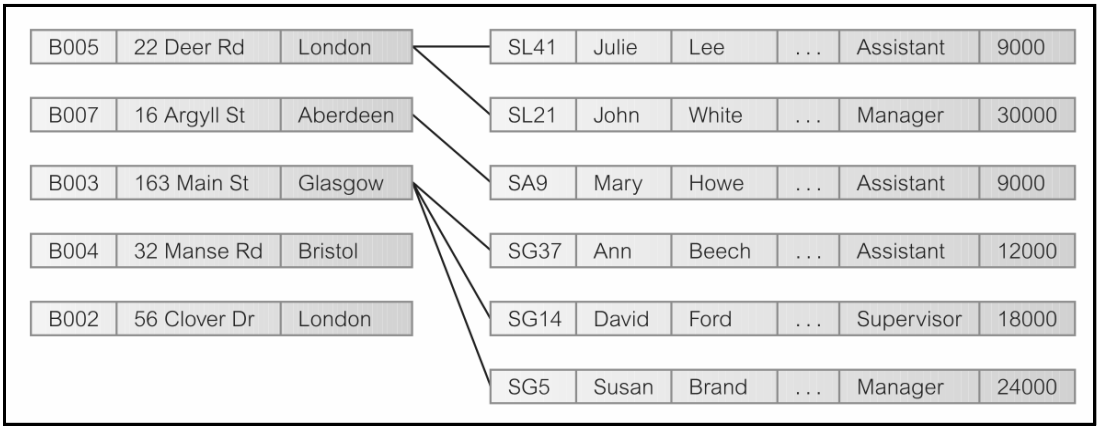


Figure 4.2 A sample instance of a network schema

Hierarchical data model

The hierarchical model is a restricted type of network model. Again, data is represented as collections of *records*, and relationships are represented by *sets*. However, the hierarchical model allows a node to have only one parent. A hierarchical model can be represented as a tree graph, with records appearing as nodes (also called segments) and sets as edges. Figure 2.3 illustrates an instance of a hierarchical schema for the same data set presented in Figure 2.1.

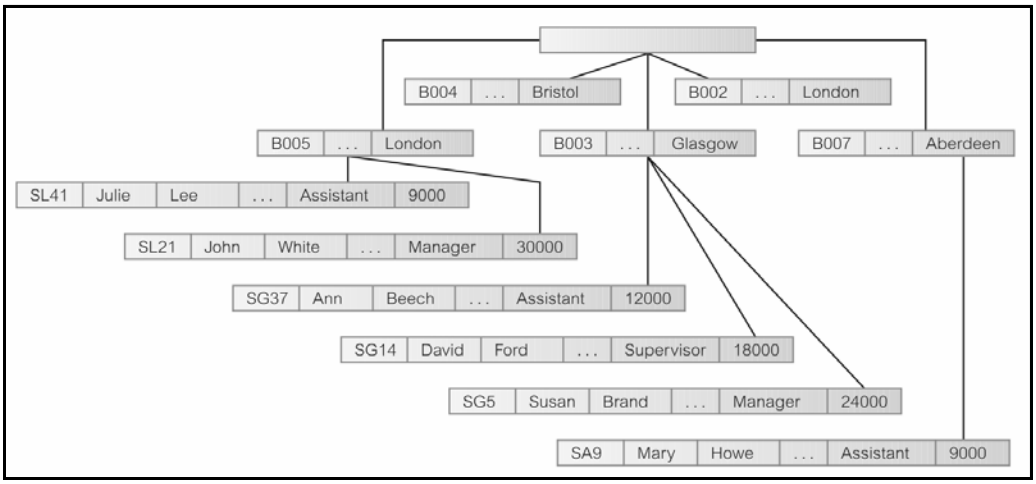


Figure4.3 A sample instance of a hierarchical schema

Record-based (logical) data models are used to specify the overall structure of the database and a higher-level description of the implementation. Their main drawback is that they do not provide adequate facilities for explicitly specifying constraints on the data, whereas the object-

based data models lack the means of logical structure specification but provide more semantic substance by allowing the user to specify constraints on the data.

The majority of modern commercial systems are based on the relational paradigm, whereas the early database systems were based on either the network or hierarchical data models. The latter two models require the user to have knowledge of the physical database being accessed, whereas the former provides a substantial amount of data independence. Hence, relational systems adopt a declarative approach to database processing (that is, they specify what data is to be retrieved), but network and hierarchical systems adopt a navigational approach (that is, they specify how the data is to be retrieved).

4.2.3 Physical Data Models

Physical data models describe how data is stored in the computer, representing information such as record structures, record orderings, and access paths. There are not as many physical data models as logical data models; the most common ones are the unifying model and the frame memory.

4.2.4 Conceptual Modeling

From an examination of the three-level architecture, we see that the conceptual schema is the heart of the database. It supports all the external views and is, in turn, supported by the internal schema. However, the internal schema is merely the physical implementation of the conceptual schema. The conceptual schema should be a complete and accurate representation of the data requirements of the enterprise (business organizations). If this is not the case, some information about the enterprise will be missing or incorrectly represented and we will have difficulty fully implementing one or more of the external views.

Conceptual modeling or conceptual database design is the process of constructing a model of the information used in an enterprise that is independent of implementation details, such as the target DBMS, application programs, programming languages, or any other physical considerations. This model is called a conceptual data model. Conceptual models are also referred to as "logical models" in the literature. However, the conceptual model is independent of all implementation details, whereas the logical model assumes knowledge of the underlying data model of the target DBMS.

4.3 Relational Databases

A data model is a collection of conceptual tools for describing data, data relationships, data

semantics, and consistency constraints. It is based on the relational model and uses a collection of tables to represent both data and the relationships among those data. It also includes a DML and DDL.

The relational model is an example of a record-based model. Record-based models are so named because the database is structured in fixed-format records of several types. Each table contains records of a particular type. Each record type defines a fixed number of fields or attributes. The columns of the table correspond to the attributes of the record type.

A relational database consists of a collection of ***tables***, each of which is assigned a unique name. For example, consider the instructor table of Figure 2.4 (a), which stores information about instructors. The table has four column headers: ID, name, dept name, and salary. Each row of this table records information about an instructor, consisting of the instructor's ID, name, dept_name, and salary. Similarly, the course table of Figure 2.4 (b) stores information about courses, consisting of a course_id, title, dept_name, and credits, for each course. Note that each instructor is identified by the value of the column ID, while each course is identified by the value of the column course_id. Figure 2.4 (c) shows a third table, prereq, which stores the prerequisite courses for each course. The table has two columns, course_id, and prereq_id. Each row consists of a pair of course identifiers such that the second course is a prerequisite for the first course.

Thus, a row in the prereq table indicates that two courses are related in the sense that one course is a prerequisite for the other. As another example, we consider the table instructor, a row in the table can be thought of as representing the relationship between a specified ID and the corresponding values for name, dept_name, and salary values.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

(a)

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

(b)

course_id	*prereq_id*
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

(c)

Figure 4.4 (a) The instructor relation (b) The course relation (c) The prereq relation

In general, a row in a table represents a relationship among a set of values. Since a table is a collection of such relationships, there is a close correspondence between the concept of the table and the mathematical concept of relation, from which the relational data model takes its name. In mathematical terminology, a ***tuple*** is simply a sequence (or list) of values. A relationship between n values is represented mathematically by an ***n-tuple*** of values, i.e., a tuple with n values, which corresponds to a row in a table. The order in which tuples appear in a relation is irrelevant since relation is a set of tuples.

Thus, in the relational model, the term ***relation*** is used to refer to a table, while the term ***tuple*** is used to refer to a row. Similarly, the term ***attribute*** refers to a column of a table. From Figure 2.4 (a), we can see that the relation instructor has four attributes: ID, name, dept name, and salary. For each attribute of relation, there is a set of permitted values, called the ***domain*** of that attribute. Thus, the domain of the salary attribute of the instructor relation is the set of all possible salary values, while the domain of the name attribute is the set of all possible instructor names.

We require that, for all relations *r*, the domains of all attributes of *r* be atomic. A domain is ***atomic*** if elements of the domain are considered to be indivisible units. For example, suppose the table instructor in figure 2.4 (a) had an attribute phone_number, which can store a set of phone numbers corresponding to the instructor. Then the domain of phone_number would not be atomic, since an element of the domain is a set of phone numbers, and it has subparts, namely the individual phone numbers in the set. The ***null*** value is a special value that signifies that the value is unknown or does not exist.

4.4 Distributed Databases

A major motivation behind the development of database systems is the desire to integrate the operational data of an organization and to provide controlled access to the data. Although we may think that integration and controlled access implies centralization, this is not the intention. In fact, the development of computer networks promotes a decentralized mode of work. This decentralized approach mirrors the organizational structure of many companies, which are logically distributed into divisions, departments, projects, and so on, and physically distributed into offices, plants, or factories, where each unit maintains its own operational data. The development of a distributed DBMS that reflects this organizational structure makes the data in all units accessible, and stores data proximate to the location where it is most frequently used, should improve the ability to share the data and should improve the efficiency with which we can access the data.

A distributed database is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network. ***Distributed DBMS*** is the software system that permits the management of the distributed database and makes the distribution transparent to users.

A distributed database management system (DDBMS) consists of a single logical database that is split into a number of ***fragments***. Each fragment is stored on one or more computers (replicas) under the control of a separate DBMS, with the computers connected by a communications network. Each site is capable of independently processing user requests that require access to local data (that is, each site has some degree of local autonomy) and is also capable of processing data stored on other computers in the network.

Users access the distributed database via ***applications***. Applications are classified as those that do not require data from other sites (local applications) and those that do require data from other sites (global applications). We require DDBMS to have at least one global application. A DDBMS, therefore, has the following characteristics:

- a collection of logically related shared data;
- data split into a number of fragments;
- fragments may be replicated;
- fragments/replicas are allocated to sites;
- sites are linked by a communications network;
- data at each site is under the control of a DBMS;
- DBMS at each site can handle local applications, autonomously;

- each DBMS participates in at least one global application.

It is not necessary for every site in the system to have its own local database, as illustrated by the topology of the DDBMS shown in Figure 2.5. From the definition of the DDBMS, the system is expected to make the distribution ***transparent*** (invisible) to the user. Thus, the fact that a distributed database is split into fragments that can be stored on different computers and perhaps replicated should be hidden from the user. The objective of transparency is to make the distributed system appear like a centralized system. This is sometimes referred to as the fundamental principle of distributed DBMSs. This requirement provides significant functionality for the end-user but, unfortunately, creates many additional problems that have to be handled by the DDBMS.

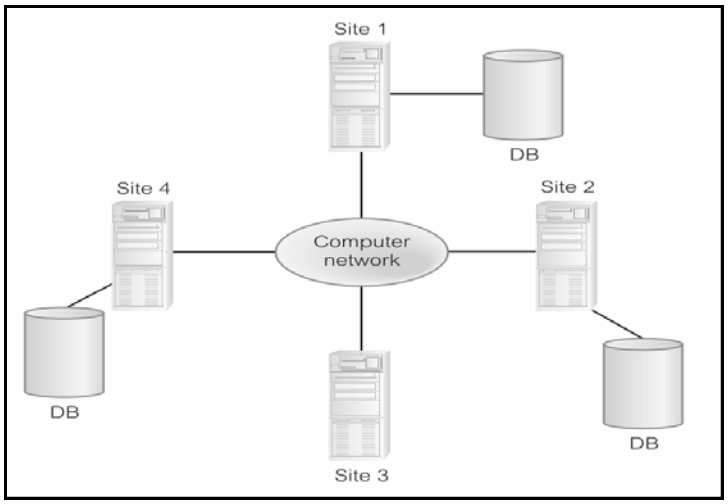


Figure 4.5 Distributed database management system (DDBMS)

Distributed processing is a centralized database that can be accessed over a computer network. It is important to make a distinction between a distributed DBMS and distributed processing. The key point with the definition of a distributed DBMS is that the system consists of data that is physically distributed across a number of sites in the network. If the data is centralized, even though other users may be accessing the data over the network, we do not consider this to be a distributed DBMS simply distributed processing. We illustrate the topology of distributed processing in Figure 2.6.

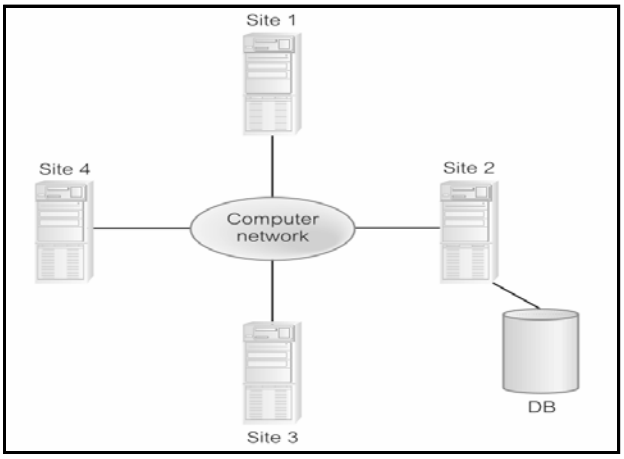


Figure 4.6 Distributed Processing

4.5 Centralized Databases

Centralized database systems are those that run on a single computer system and do not interact with other computer systems. Such database systems span a range from single-user database systems running on personal computers to high-performance database systems running on high-end server systems.

A modern, general-purpose computer system consists of one to a few processors and a number of device controllers that are connected through a common bus that provides access to shared memory as shown in figure 2.7. The processors have local cache memories that store local copies of parts of the memory, to speed up access to data. Each processor may have several independent cores, each of which can execute a separate instruction stream. Each device controller is in charge of a specific type of device (for example, a disk drive, an audio device, or a video display). The processors and the device controllers can execute concurrently, competing for memory access. Cache memory reduces the contention for memory access since it reduces the number of times that the processor needs to access the shared memory.

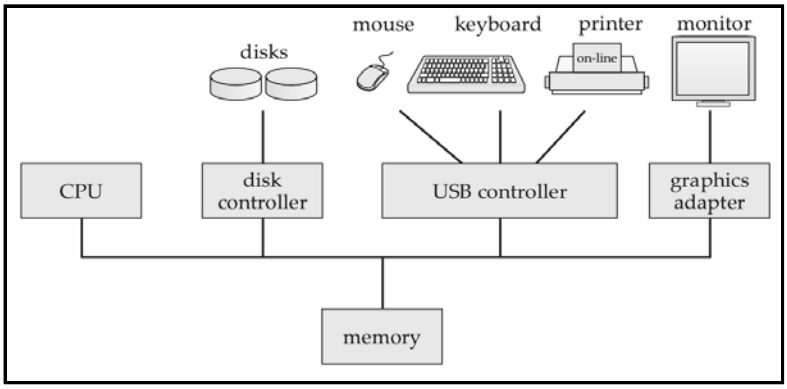


Figure 4.7 Centralized Computer System

Computers can be used in two distinct ways: as ***single-user systems*** and as ***multiuser systems.*** Personal computers and workstations fall into the first category. A typical single-user system is a desktop unit used by a single person, usually with only one processor and one or two hard disks, and usually only one person using the machine at a time. A typical multiuser system, on the other hand, has more disks and more memory and may have multiple processors. It serves a large number of users who are connected to the system remotely.

Database systems designed for use by single users usually do not provide many of the facilities that a multiuser database provides. In particular, they may not support concurrency control, which is not required when only a single user can generate updates. Provisions for crash recovery in such systems are either absent or primitive—for example, they may consist of simply making a backup of the database before any update. In contrast, database systems designed for multiuser systems support the full transactional features that we have studied earlier.

Although most general-purpose computer systems in use today have multiple processors, they have ***coarse-granularity parallelism***, with only a few processors (about two to four, typically), all sharing the main memory. Databases running on such machines usually do not attempt to partition a single query among the processors; instead, they run each query on a single processor, allowing multiple queries to run concurrently. Thus, such systems support a higher throughput; that is, they allow a greater number of transactions to run per second, although individual transactions do not run any faster.

Databases designed for single-processor machines already provide multitasking, allowing multiple processes to run on the same processor in a time-shared manner, giving a view to the user of multiple processes running in parallel. Thus,coarse-granularity parallel machines logically appear to be identical to single-processor machines, and database systems designed for time-shared machines can be easily adapted to run on them.

In contrast, machines with ***fine-granularity parallelism*** have a large number of processors, and database systems running on such machines attempt to parallelize single tasks (queries, for example) submitted by users. Parallelism is emerging as a critical issue in the future design of database systems. Whereas today those computer systems with multicore processors have only a few cores, future processors will have large numbers of cores. As a result, parallel database systems, which once were specialized systems running on specially designed hardware, will become the norm.

Client-Server Systems

As personal computers became faster, more powerful, and cheaper, there was a shift away from the centralized system architecture. Personal computers supplanted terminals connected to centralized systems. Correspondingly, personal computers assumed the user-interface functionality that used to be handled directly by the centralized systems. As a result, centralized systems today act as *server systems* that satisfy requests generated by *client systems*. Figure 2.8 shows the general structure of a client-server system.

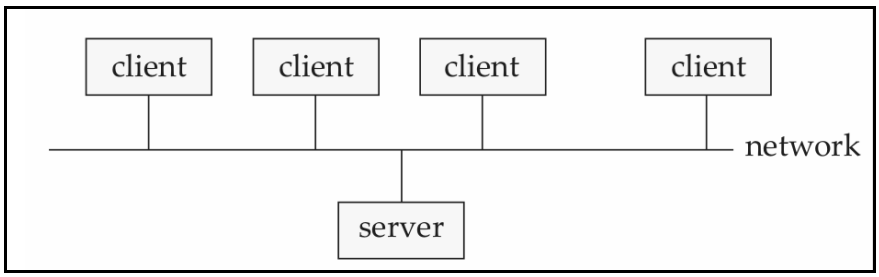


Figure 4.8 General structure of a client-server system

he functionality provided by database systems can be broadly divided into two parts- the front end and the back end. The back end manages access structures, query evaluation and optimization, concurrency control, and recovery. The front end of a database system consists of tools such as the SQL user interface, forms interfaces, report generation tools, and data mining and analysis tools. The interface between the front end and the back end is through SQL, or through an application program. Certain application programs, such as spreadsheets and statistical-analysis packages, use the client–server interface directly to access data from a back-end server. In effect, they provide front ends specialized for particular tasks.

Some transaction-processing systems provide a *transactional remote procedure call* interface to connect clients with a server. These calls appear like ordinary procedure calls to the programmer, but all the remote procedure calls from a client are enclosed in a single transaction at the server end. Thus, if the transaction aborts, the server can undo the effects of the individual remote procedure calls.

4.6 Difference between Centralized and Distributed Databases

We have already learned that a centralized database is basically a type of database that is stored, located as well as maintained at a single location only. This type of database is modified and managed from that location itself. This location is thus mainly any database system or a centralized computer system. The centralized location is accessed via an internet connection

(LAN, WAN, etc). This centralized database is mainly used by institutions or organizations.
On the other hand, a distributed database is basically a type of database which consists of multiple databases that are connected with each other and are spread across different physical locations. The data that is stored on various physical locations can thus be managed independently of other physical locations. The communication between databases at different physical locations is thus done by a computer network. A comparison between centralized and distributed databases is illustrated in Table 4.1.

Table 4.1 Comparison between Centralized and Distributed database

Centralized database	Distributed database
It is a database that is stored, located as well as maintained at a single location only.	It is a database that consists of multiple databases that are connected with each other and are spread across different physical locations.
This database provides a uniform and complete view to the user.	Since it is spread across different locations thus it is difficult to provide a uniform view to the user.
The users cannot access the database in case of database failure occurs.	In a distributed database if one database fails users have access to other databases.
The data access time in the case of multiple users is more in a centralized database.	The data access time in the case of multiple users is less in a distributed database.
The management, modification, and backup of this database are easier as the entire data is present at the same location.	The management, modification, and backup of this database are very difficult as it is spread across different physical locations.
This database has more data consistency in comparison to distributed databases.	This database may have some data replications thus data consistency is less.
A centralized database is less costly.	Distributed databases are very expensive.

4.7 Summary

- The data model is an integrated collection of concepts for describing and manipulating

data, relationships between data, and constraints on the data in an organization.

- Object-based data models use concepts such as entities, attributes, and relationships. In a record-based model, the database consists of a number of fixed-format records, possibly of differing types. Physical data models describe how data is stored in the computer, representing information such as record structures, record orderings, and access paths.
- A relational database is based on the relational model and uses a collection of tables to represent both data and the relationships among those data.
- A distributed database is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.
- Centralized database systems are those that run on a single computer system and do not interact with other computer systems. Such database systems span a range from single-user database systems running on personal computers to high-performance database systems running on high-end server systems.

4.8 Key Terms

- **Conceptual Modeling:** It is also known as conceptual database design. It is the process of constructing a model of the information used in an enterprise that is independent of implementation details.
- **Tuple:** Tuples are used to store multiple items in a single variable.
- **Domain:** For each attribute of relation, there is a set of permitted values, called the domain of that attribute.
- **Granularity:** In parallel computing, granularity (or grain size) of a task is a measure of the amount of work (or computation) that is performed by that task.
- **Coarse-grained parallelism:** In coarse-grained parallelism, a program is split into large tasks. Due to this, a large amount of computation takes place in processors.
- **Fine-grained parallelism:** In fine-grained parallelism, a program is broken down into a large number of small tasks. These tasks are assigned individually to many processors.

4.9 Check Your Progress

Short- Answer Type

Q1) Which of the following is an example of an Object-based logical model?

a) Relational model b) Hierarchical model

c) Network model d) Entity-Relationship model

Q2) In DBMS, the term used to represent the real world concept or object is known as _____.

Q3) Object-based data models use concepts such as entities, attributes, and relationships. True/ False?

Q4) A relational database consists of a collection of _____.

Q5) A domain is atomic if elements of the domain are considered to be _____ units.

Long- Answer Type

Q1) Describe the three types of data models briefly.

Q2) Differentiate between centralized and distributed databases.

Q3) What is a relational database? Also, explain its types.

Q4) Explain the client-server system in detail.

Q5) Discuss the Entity-Relational data model.

References

- *Database Systems: A Practical Approach to Design, Implementation, and Management*, Thomas Connolly & Carolyn Begg, Pearson, 6th Edition.

Database System Concepts, Henry F Korth, Abraham Silberschatz, and S. Sudharshan, 6th Edition, McGraw Hill, 2011.

Unit 5: Introduction to Relational Databases

5.0 Introduction

In the preceding units we have primarily discussed three data models, the Entity-Relationship (ER) model, the relational data model, and the object-oriented data model. We discussed how all these data models have been thoroughly developed in terms of the following features:

- Modeling constructs for developing schemas for database applications.
- Constraints facilities for expressing certain types of relationships and constraints on the data as determined by application semantics.
- Operations and language facilities to manipulate the database.

Out of these three models, the ER model has been primarily employed in CASE tools that are used for database and software design, whereas the other two models have been used as the basis for commercial DBMSs. This unit discusses the emerging class of commercial DBMSs that are called object-relational or enhanced relational systems, and some of the conceptual foundations for these systems. These systems are often called object-relational DBMSs (ORDBMSs). They have emerged as a way of enhancing the capabilities of relational DBMSs (RDBMSs) with some of the features that appeared in object-DBMSs (ODBMSs).

With the arrival of the third generation of database management systems, namely ***Object-Relational Database Management Systems (ORDBMSs)*** and ***Object- Oriented Database Management Systems (OODBMSs)***, the two disciplines have been combined to allow the concurrent modeling of both data and the processes acting upon the data.

The main forces behind the development of extended ORDBMSs stem from the inability of the legacy DBMSs and the basic relational data model as well as the earlier RDBMSs to meet the

challenges of new applications. These are primarily in areas that involve a variety of types of data, for example, text in computer-aided desktop publishing; images in satellite imaging or weather forecasting; complex non-conventional data in engineering designs, in the biological genome information, and in architectural drawings; time series data in history of stock market transactions or sales histories; and spatial and geographic data in maps, air/ water pollution data, and traffic data. Hence, there is a clear need to design databases that can develop, manipulate, and maintain the complex objects arising from such applications.

5.1 Unit Objectives

After completing this unit, the reader will be able to:

- Illustrate the basic concept of Relational Database.
- Learn about the Relational database designing.
- Discuss the integrity constraints in RDBMS.
- Gain knowledge about the standards of OODBMS.

5.2 Basic Concept of Relational Databases

The relational model is today the primary data model for commercial data processing applications. It attained its primary position because of its simplicity, which eases the job of the programmer, compared to earlier data models such as the network model or the hierarchical model.

The relational model represents the database as a collection of relations. Informally, each relation resembles a table of values or, to some extent, a "flat" file of records. When a relation is thought of as a table of values, each row in the table represents a collection of related data values. In the relational model, each row in the table represents a fact that typically corresponds to a real-world entity or relationship. The table name and column names are used to help in interpreting the meaning of the values in each row.

In the formal relational model terminology, a row is called a ***tuple***, a column header is called an ***attribute***, and the table is called a ***relation***. The data type describing the types of values that can appear in each column is called a ***domain***.

An RDBMS requires only that the database be perceived by the user as tables. Note, however, that this perception applies only to the logical structure of the database: that is, the external and conceptual levels of the ANSI-SPARC architecture. It does not apply to the physical

structure of the database, which can be implemented using a variety of storage structures. In the relational model, relations are used to hold information about the objects to be represented in the database. A relation is represented as a two- dimensional table in which the rows of the table correspond to individual records and the table columns correspond to attributes. Attributes can appear in any order and the relation will still be the same relation, and therefore will convey the same meaning.

Domains are an extremely powerful feature of the relational model. Every attribute in a relation is defined on a domain. Domains may be distinct for each attribute, or two or more attributes may be defined on the same domain. The domain concept is important, because it allows the user to define in a central place the meaning and source of values that attributes can hold. As a result, more information is available to the system when it undertakes the execution of a relational operation, and operations that are semantically incorrect can be avoided.

A relation has the following properties:

- The relation has a name that is distinct from all other relation names in the relational schema.
- Each cell of the relation contains exactly one atomic (single) value.
- Each attribute has a distinct name.
- The values of an attribute are all from the same domain.
- Each tuple is distinct; there are no duplicate tuples;
- The order of attributes has no significance;
- The order of tuples has no significance, theoretically. (However, in practice, the order may affect the efficiency of accessing tuples.)

5.3 Relational Database Design

The task of creating a database application is a complex one, involving design of the database schema, design of the programs that access and update the data, and design of a security scheme to control access to data. The needs of the users play a central role in the design process.

The design of a complete database application environment that meets the needs of the enterprise being modeled requires attention to a broad set of issues. These additional aspects of the expected use of the database influence a variety of design choices at the physical, logical, and view levels.

Phases of Database Designing

A high-level data model serves the database designer by providing a conceptual framework in which to specify, in a systematic fashion, the data requirements of the database users, and a database structure that fulfills these requirements.

1. The initial phase of database design is to characterize fully the data needs of the prospective database users. The database designer needs to interact extensively with domain experts and users to carry out this task. The outcome of this phase is a specification of user requirements. While there are techniques for diagrammatically representing user requirements, in this unit we restrict ourselves to textual descriptions of user requirements.
2. Next, the designer chooses a data model and, by applying the concepts of the chosen data model, translates these requirements into a conceptual schema of the database. The schema developed at this conceptual-design phase provides a detailed overview of the enterprise. The entity-relationship model, which we have studied in previous units, is typically used to represent the conceptual design. Stated in terms of the entity-relationship model, the conceptual schema specifies the entities that are represented in the database, the attributes of the entities, the relationships among the entities, and constraints on the entities and relationships. Typically, the conceptual-design phase results in the creation of an entity-relationship diagram that provides a graphic representation of the schema.
3. A fully developed conceptual schema also indicates the functional requirements of the enterprise. In a specification of functional requirements, users describe the kinds of operations (or transactions) that will be performed on the data. Example operations include modifying or updating data, searching for and retrieving specific data, and deleting data. At this stage of conceptual design, the designer can review the schema to ensure it meets functional requirements.
4. The process of moving from an abstract data model to the implementation of the database proceeds in two final design phases.
   - In the logical-design phase, the designer maps the high-level conceptual schema onto the implementation data model of the database system that will be used. The implementation data model is typically the relational data model, and this step typically consists of mapping the conceptual schema defined using the entity-relationship model

into a relation schema.

- Finally, the designer uses the resulting system-specific database schema in the subsequent physical-design phase, in which the physical features of the database are specified.

The physical schema of a database can be changed relatively easily after an application has been built. However, changes to the logical schema are usually harder to carry out, since they may affect a number of queries and updates scattered across application code. It is therefore important to carry out the database design phase with care, before building the rest of the database application.

Relational Database Design (RDD)

Relational databases differ from other databases in their approach to organizing data and performing transactions. In an RDD, the data are organized into tables and all types of data access are carried out via controlled transactions. Relational database design satisfies the ACID (atomicity, consistency, integrity and durability) properties required from a database design. Relational database design mandates the use of a database server in applications for dealing with data management problems.

The four stages of an RDD are as follows:

- **Relations and attributes:** The various tables and attributes related to each table are identified. The tables represent entities, and the attributes represent the properties of the respective entities.
- **Primary keys:** The attribute or set of attributes that help in uniquely identifying a record is identified and assigned as the primary key
- **Relationships:** The relationships between the various tables are established with the help of foreign keys. Foreign keys are attributes occurring in a table that are primary keys of another table. The types of relationships that can exist between the relations (tables) are:
  - One to one
  - One to many
  - Many to many

An entity-relationship diagram can be used to depict the entities, their attributes and the relationship between the entities in a diagrammatic way.

- **Normalization:** This is the process of optimizing the database structure. Normalization simplifies the database design to avoid redundancy and confusion. The different normal

forms are as follows:

- First normal form
- Second normal form
- Third normal form
- Boyce-Codd normal form
- Fifth normal form

By applying a set of rules, a table is normalized into the above normal forms in a linearly progressive fashion. The efficiency of the design gets better with each higher degree of normalization.

When we define an E-R diagram carefully, identifying all entities correctly, the relation schemas generated from the E-R diagram should not need much further normalization. However, there can be functional dependencies between attributes of an entity. Most examples of such dependencies arise out of poor E-R diagram design. Similarly, a relationship set involving more than two entity sets may result in a schema that may not be in a desirable normal form. Since most relationship sets are binary, such cases are relatively rare.

Functional dependencies can help us detect poor E-R design. If the generated relation schemas are not in desired normal form, the problem can be fixed in the E-R diagram. That is, normalization can be done formally as part of data modeling. Alternatively, normalization can be left to the designer's intuition during E-R modeling, and can be done formally on the relation schemas generated from the E-R model.

If a multivalued dependency holds and is not implied by the corresponding functional dependency, it usually arises from one of the following sources:

- A many-to-many relationship set.
- A multivalued attribute of an entity set.

For a many-to-many relationship set each related entity set has its own schema and there is an additional schema for the relationship set. For a multivalued attribute, a separate schema is created consisting of that attribute and the primary key of the entity set (as in the case of the phone number attribute of the entity set instructor). The universal-relation approach to relational database design starts with an assumption that there is one single relation schema containing all attributes of interest. This single schema defines how users and applications interact with the database.

5.4 Integrity Constraints

We have already discussed the structural part of the relational data model. A data model has two other parts: a manipulative part, defining the types of operation that are allowed on the data, and a set of integrity constraints, which ensure that the data is accurate.

Because every attribute has an associated domain, there are constraints (called ***domain constraints***) that form restrictions on the set of values allowed for the attributes of relations. In addition, there are two important ***integrity rules***, which are constraints or restrictions that apply to all instances of the database. The two principal rules for the relational model are known as ***entity integrity*** and ***referential integrity***. Other types of integrity constraint are ***multiplicity***, and ***general constraints***.

1. **Entity Integrity**

   The first integrity rule applies to the primary keys of base relations. In a base relation, no attribute of a primary key can be null. By definition, a primary key is a minimal identifier that is used to identify tuples uniquely. This means that no subset of the primary key is sufficient to provide unique identification of tuples. If we allow a null for any part of a primary key, we are implying that not all the attributes are needed to distinguish between tuples, which contradicts the definition of the primary key. By definition, this attribute must be a primary key, but it contains nulls. Because this relation is not a base relation, the model allows the primary key to be null.

2. **Referential Integrity**

   The second integrity rule applies to foreign keys. If a foreign key exists in a relation, either the foreign key value must match a candidate key value of some tuple in its home relation or the foreign key value must be wholly null.

3. **General Constraints**

   Additional rules specified by the users or database administrators of a database that define or constrain some aspect of the enterprise. It is also possible for users to specify additional constraints that the data must satisfy. Unfortunately, the level of support for general constraints varies from system to system.

5.5 Introduction to ORDBMS

We are already aware that the relational model has a strong theoretical foundation, based on first-order predicate logic. This theory supported the development of SQL, a declarative

language that has now become the standard language for defining and manipulating relational databases. Other strengths of the relational model are its simplicity, its suitability for Online Transaction Processing (OLTP), and its support for data independence. However, the relational data model, and relational DBMSs in particular, are not without their disadvantages. Here are some weaknesses of the RDBMSs:

- **Poor representation of "real-world" entities**

  The process of normalization generally leads to the creation of relations that do not correspond to entities in the "real-world." The fragmentation of a "real-world" entity into many relations, with a physical representation that reflects this structure, is inefficient leading to many joins during query processing.

- **Semantic overloading**

  The relational model has only one construct for representing data and relationships between data: the relation. For example, to represent a many-to-many (*:*) relationship between two entities A and B, we create three relations, one to represent each of the entities A and B and one to represent the relationship. There is no mechanism to distinguish between entities and relationships, or to distinguish between different kinds of relationship that exist between entities. For example, a 1:* relationship might be Has , Owns, Manages, and so on. If such distinctions could be made, then it might be possible to build the semantics into the operations. It is said that the relational model is *semantically overloaded*.

- **Poor support for integrity and general constraints**

  Integrity refers to the validity and consistency of stored data. Integrity is usually expressed in terms of constraints, which are consistency rules that the database is not permitted to violate. Unfortunately, many commercial systems do not fully support these constraints and it is necessary to build them into the applications. This, of course, is dangerous and can lead to duplication of effort and, worse still, inconsistencies. Furthermore, there is no support for general constraints in the relational model, which again means that they have to be built into the DBMS or the application.

- **Homogeneous data structure**

  The relational model assumes both horizontal and vertical homogeneity. Horizontal homogeneity means that each tuple of a relation must be composed of the same attributes. Vertical homogeneity means that the values in a particular column of a

relation must all come from the same domain. Additionally, the intersection of a row and column must be an atomic value. This fixed structure is too restrictive for many "real-world" objects that have a complex structure, and it leads to unnatural joins, which are inefficient as mentioned previously. In defense of the relational data model, it could equally be argued that its symmetric structure is one of the model's strengths.

- **Limited operations**

  The relational model has only a fixed set of operations, such as set and tuple-oriented operations, operations that are provided in the SQL specification. However, SQL does not allow new operations to be specified. Again, this is too restrictive to model the behavior of many real-world objects. For example, a GIS application typically uses points, lines, line groups, and polygons, and needs operations for distance, intersection, and containment.

- **Difficulty handling recursive queries**

  Atomicity of data means that repeating groups are not allowed in the relational model. As a result, it is extremely difficult to handle recursive queries, that is, queries about relationships that a relation has with itself (directly or indirectly).

  Relational DBMSs are currently the dominant database technology. Until recently, the choice of DBMS seemed to be between the relational DBMS and the object-oriented DBMS. However, many vendors of RDBMS products are conscious of the threat and promise of the OODBMS. They agree that traditional relational DBMSs are not suited to the advanced applications, and that added functionality is required. However, they reject the claim that extended RDBMSs will not provide sufficient functionality or will be too slow to cope adequately with the new complexity.

If we examine the advanced database applications that are emerging, we find they make extensive use of many object-oriented features such as a user-extensible type system, encapsulation, inheritance, polymorphism, dynamic binding of methods, complex objects including non-first normal form objects, and object identity. The most obvious way to remedy the shortcomings of the relational model is to extend the model with these types of features. This is the approach that has been taken by many extended relational DBMSs, although each has implemented different combinations of features. Thus, there is no single extended relational model; rather, there are a variety of these models, whose characteristics depend upon the way and the degree to which extensions were made. However, all the models do share the same

basic relational tables and query language, all incorporate some concept of "object," and some have the ability to store methods (or procedures or triggers) as well as data in the database.

Various terms have been used for systems that have extended the relational data model. The original term that was used to describe such systems was the ***Extended Relational DBMS (ERDBMS)*** and the term Universal Server or Universal DBMS (UDBMS) has also been used. However, in recent years the more descriptive term ***Object-Relational DBMS*** has been used to indicate that the system incorporates some notion of "object."

Three of the leading RDBMS vendors—Oracle, Microsoft, and IBM—have all extended their systems into ORDBMSs, although the functionality provided by each is slightly different. The concept of the ORDBMS, as a hybrid of the RDBMS and the OODBMS, is very appealing, preserving the wealth of knowledge and experience that has been acquired with the RDBMS—so much so that some analysts predict the ORDBMS will have a 50% larger share of the market than the RDBMS.

As might be expected, the standards activity in this area is based on extensions to the SQL standard. The national standards bodies have been working on object extensions to SQL since 1991. These extensions have become part of the SQL standard, with releases in 1999, referred to as SQL:1999, 2003, (SQL:2003), 2006 with extensions for XML (SQL:2006), 2008 (SQL:2008) and 2011 (SQL:2011). These releases of the SQL standard are an ongoing attempt to standardize extensions to the relational model and query language.

Advantages of ORDBMSs

- The main advantages of extending the relational data model come from ***reuse and sharing***. Reuse comes from the ability to extend the DBMS server to perform standard functionality centrally, rather than have it coded in each application. These advantages also give rise to increased productivity, both for the developer and for the end-user.
- Another obvious advantage is that the extended relational approach preserves the significant body of knowledge and experience that has gone into developing relational applications. This is a significant advantage, as many organizations would find it prohibitively expensive to change. If the new functionality is designed appropriately, this approach should allow organizations to take advantage of the new extensions in an evolutionary way without losing the benefits of current database features and functions. Thus, an ORDBMS could be introduced in an integrative fashion, as proof-of-concept projects. The SQL:2011 standard is designed to be compatible with the SQL2 standard,

and so any ORDBMS that complies with SQL:2011 should provide this capability.

Disadvantages of ORDBMSs

The ORDBMS approach has the obvious disadvantages of complexity and associated increased costs. In addition, there are the proponents of the relational approach who believe the essential simplicity and purity of the relational model are lost with these types of extension. There are also those who believe that the RDBMS is being extended for what will be a minority of applications that do not achieve optimal performance with current relational technology.

Object-oriented models and programs deeply combine relationships and encapsulated objects to more closely mirror the real world. This defines broader sets of relationships than those expressed in SQL, and involves functional programs interspersed in the object definitions. In fact, objects are fundamentally not extensions of data, but a completely different concept with far greater power to express real-world relationships and behaviors.

5.6 Summary

- Database design mainly involves the design of the database schema. The entity-relationship (E-R) data model is a widely used data model for database design. It provides a convenient graphical representation to view data, relationships, and constraints.
- In the formal relational model terminology, a row is called a *tuple*, a column header is called an *attribute*, and the table is called a *relation*. The data type describing the types of values that can appear in each column is called a *domain*.
- Relational databases differ from other databases in their approach to organizing data and performing transactions. In an RDD, the data are organized into tables and all types of data access are carried out via controlled transactions.
- The main advantages of extending the relational data model come from *reuse and sharing*.
- The ORDBMS approach has the obvious disadvantages of complexity and associated increased costs. In addition, there are the proponents of the relational approach who believe the essential simplicity and purity of the relational model are lost with these types of extension.

5.7 Key Terms

- **Primary keys:** The attribute or set of attributes that help in uniquely identifying a

record is identified and assigned as the primary key.

- **Relationships:** The relationships between the various tables are established with the help of foreign keys. Foreign keys are attributes occurring in a table that are primary keys of another table.
- **Domain Constraints:** The constraints that form restrictions on the set of values allowed for the attributes of relations.

5.8 Check Your Progress

Short- Answer Type

Q1) A row is called a _____ , a column header is called an _____.

Q2) The relation has a name that is distinct from all other relation names in the relational schema. (True/ False?)

Q3) _____ simplifies the database design to avoid redundancy and confusion.

Q4) The attribute or set of attributes that help in uniquely identifying a record is identified and assigned as the _____ key.

Q5) Domains may be distinct for each attribute, or two or more attributes may be defined on the same domain. (True/ False?)

Long- Answer Type

Q1) What are integrity constraints in Relational DBMS? Explain its types.

Q2) Explain the phases of designing in Relational database design.

Q3) Give the advantages and disadvantages of ORDBMS.

Q4) State the properties of a relation in RDBMS.

Q5) Discuss the weaknesses of RDBMS that lead to development of ORDBMS.

References

- *Database Systems: A Practical Approach to Design, Implementation, and Management*, Thomas Connolly & Carolyn Begg, Pearson, 6th Edition.
- *Database System Concepts*, Henry F Korth, Abraham Silberschatz, and S. Sudharshan, 6th Edition, McGraw Hill, 2011.

Unit 6: Products and Applications

6.0 Introduction

Having a standard for a particular type of database system is very important, because it provides support for portability of database applications. ***Portability*** is generally defined as the capability to execute a particular application program on different systems with minimal modifications to the program itself. In the object database field, portability would allow a program written to access one Object-Database Management System (ODBMS) package. This is important to database users because they are generally wary of investing in a new technology if the different vendors do not adhere to a standard.

A second potential advantage of having and adhering to standards is that it helps in achieving ***interoperability***, which generally refers to the ability of an application to access multiple distinct systems. In database terms, this means that the same application program may access some data stored under one ODBMS package, and other data stored under another package. A third advantage of standards is that it allows customers to compare commercial products more easily by determining which parts of the standard are supported by each product.

The lack of a standard for ODBMSs until recently may have caused some potential users to shy away from converting to this new technology. A consortium of ODBMS vendors, called ***ODMG (Object Data Management Group)***, proposed a standard that is known as the ODMG-93 or ODMG 1.0 standard. This was revised into ODMG 2.0 later on.

The standard is made up of several parts: the object model, the object definition language

(ODL), the object query language (OQL), and the bindings to object-oriented programming languages. Language bindings have been specified for three object-oriented programming languages—namely, C++, SMALLTALK, and JAVA.

We have already examined some of the basic concepts of Object- Oriented Database Management Systems (OODBMSs). Here, we will continue our study of these systems and examine the object model and specification languages proposed by the ***Object Data Management Group (ODMG)***. The ODMG object model is important because it specifies a standard model for the semantics of database objects and supports interoperability between compliant systems. It became the de facto standard for OODBMSs. To put the discussion of OODBMSs into a commercial context, we also examine the architecture and functionality of ObjectStore, a commercial OODBMS. This unit also describes the Object Definition Language (ODL), and Object Query Language (OQL). We will also discuss an overview of SQL3 and the Nested Relational systems.

6.1 Unit Objectives

After completing this unit, the reader will be able to:

- Illustrate the basic concept of Object Data Management Group (ODMG).
- Learn about the Object definition language (ODL) and Object Query Language (OQL).
- Gain knowledge about SQL3.
- Discuss the concept of Nested relational systems and Implementation issues.
- Differentiate between OODBMS & ORDBMS.

6.2 Overview of object model of ODMG

The ***ODMG object model*** is the data model upon which the object definition language (ODL) and object query language (OQL) are based. In fact, this object model provides the data types, type constructors, and other concepts that can be utilized in the ODL to specify object database schemas. Hence, it is meant to provide a standard data model for object-oriented databases, just as the SQL report describes a standard data model for relational databases. It also provides a standard terminology in a field where the same terms were sometimes used to describe different concepts.

Objects and ***literals*** are the basic building blocks of the object model. The main difference between the two is that an object has both an object identifier and a state (or current value),

whereas a literal has only a value but no object identifier. In either case, the value can have a complex structure. The object state can change over time by modifying the object value. A literal is basically a constant value, possibly having a complex structure, that does not change.

An object is described by four characteristics: ***(1) identifier, (2) name, (3) lifetime, and (4) structure.***

- The object ***identifier*** is a unique system-wide identifier (or Object_Id). Every object must have an object identifier.
- In addition to the Object_Id, some objects may optionally be given a unique ***name*** within a particular database. This name can be used to refer to the object in a program, and the system should be able to locate the object given that name. Obviously, not all individual objects will have unique names. Typically, a few objects, mainly those that hold collections of objects of a particular object type—such as extents—will have a name. These names are used as entry points to the database; that is, by locating these objects by their unique name, the user can then locate other objects that are referenced from these objects. Other important objects in the application may also have unique names. All such names within a particular database must be unique.
- The ***lifetime*** of an object specifies whether it is a persistent object (that is, a database object) or transient object (that is, an object in an executing program that disappears after the program terminates).
- Finally, the ***structure*** of an object specifies how the object is constructed by using the type constructors. The structure specifies whether an object is atomic or a collection object.

  In the object model, a literal is a value that does not have an object identifier. However, the value may have a simple or complex structure. There are three types of literals: ***(1) atomic, (2) collection, and (3) structured***.
- ***Atomic literals*** correspond to the values of basic data types and are predefined. The basic data types of the object model include long, short, and unsigned integer numbers (these are specified by the keywords Long, Short, Unsigned Long, Unsigned Short in ODL), regular and double precision floating point numbers (Float, Double), boolean values (Boolean), single characters (Char), character strings (String), and enumeration types (Enum), among others.
- ***Structured literals*** correspond roughly to values that are constructed using the tuple

constructor. They include Date, Interval, Time, and Timestamp as built-in structures, as well as any additional user-defined type structures as needed by each application. User-defined structures are created using the ***Struct*** keyword in ODL, as in the C and C++ programming languages.

- ***Collection literals*** specify a value that is a collection of objects or values but the collection itself does not have an Object_Id. The collections in the object model are Set<t>, Bag<t>, List<t>, and Array<t>, where t is the type of objects or values in the collection.
- The notation of ODMG uses three concepts: ***interface, literal, and class***. Following the ODMG terminology, we use the word behavior to refer to operations and state to refer to properties (attributes and relationships).
- An interface specifies only behavior of an object type and is typically ***non-instantiable*** (that is, no objects are created corresponding to an interface). Although an interface may have state properties (attributes and relationships) as part of its specifications, these cannot be inherited from the interface. Hence, an interface serves to define operations that can be inherited by other interfaces, as well as by classes that define the user-defined objects for a particular application.
- A class specifies both state (attributes) and behavior (operations) of an object type and is ***instantiable***. Hence, database and application objects are typically created based on the user-specified class declarations that form a database schema.
- Finally, a literal declaration specifies state but no behavior. Thus, a literal instance holds a simple or complex structured value but has neither an object identifier nor encapsulated operations.
- In general, operations are applied to objects using the ***dot notation***. For example, given an object O, to compare it with another object P, we write

$$O.same\_as\ (P)$$

The result returned by this operation is Boolean and would be true if the identity of P is the same as that of O, and false otherwise. Similarly, to create a copy P of object O, we write

$$P = O.copy\ ()$$

An alternative to the dot notation is the ***arrow notation***: $O \rightarrow same\_as\ (P)$ or O→copy ().

Inheritance in the Object Model of ODMG

In the ODMG object model, two types of inheritance relationships exist: behavior only

inheritance and state plus behavior inheritance. ***Behavior inheritance*** is also known as ISA or interface inheritance and is specified by the colon (:) notation. Hence, in the ODMG object model, behavior inheritance requires the supertype to be an interface, whereas the subtype could be either a class or another interface.

The other inheritance relationship, called ***EXTENDS inheritance***, is specified by the keyword extends. It is used to inherit both state and behavior strictly among classes, so both the supertype and the subtype must be classes. Multiple inheritance via extends is not permitted. However, multiple inheritance is allowed for behavior inheritance via the colon (:) notation. Hence, an interface may inherit behavior from several other interfaces. A class may also inherit behavior from several interfaces via colon (:) notation, in addition to inheriting behavior and state from at most one other class via extends.

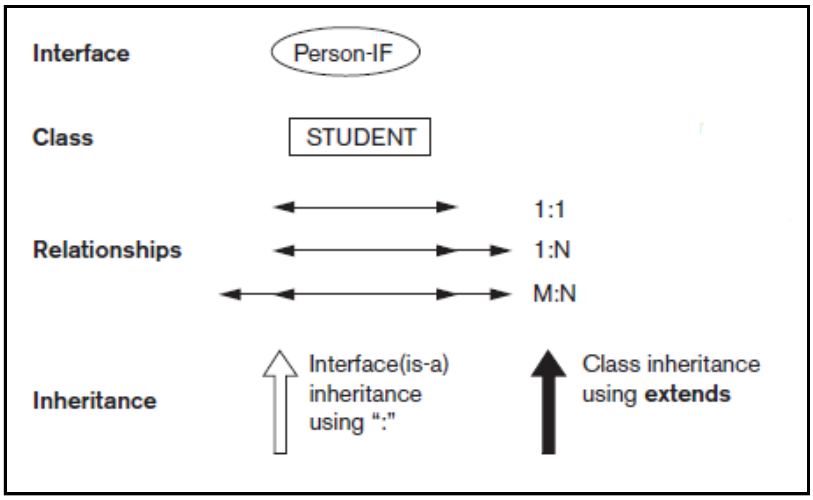
6.3 Object Definition & Query Language

After our overview of the ODMG 2.0 object model in the previous section, we now show how these concepts can be utilized to create an object database schema using the object definition language ODL. The ODL is designed to support the semantic constructs of the ODMG 2.0 object model and is independent of any particular programming language. Its main use is to create object specifications—that is, classes and interfaces. Hence, ODL is not a full programming language. A user can specify a database schema in ODL independently of any programming language, then use the specific language bindings to specify how ODL constructs can be mapped to constructs in specific programming languages, such as C++, SMALLTALK, and JAVA.

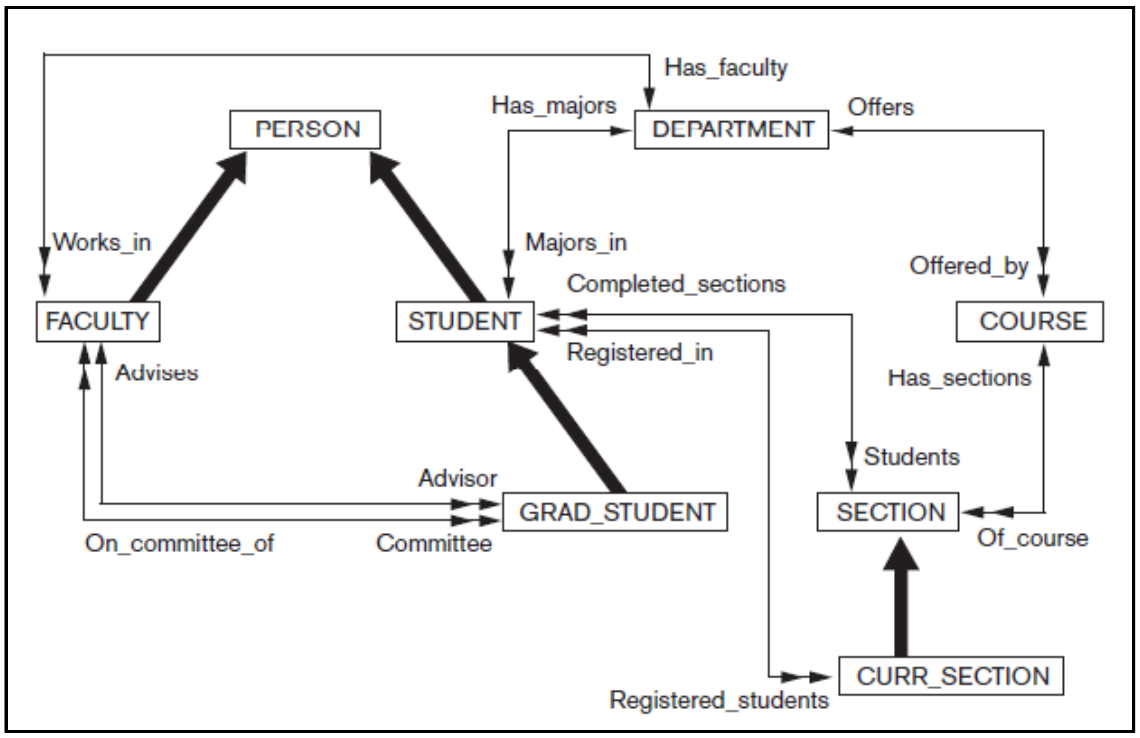
We will describe the concepts of ODL using an example. Figure 14.1 (b) shows a possible object schema for part of the UNIVERSITY database. The graphical notation for Figure 14.1(b) is shown in Figure 14.1 (a) and can be considered as a variation of EER diagrams with the added concept of interface inheritance but without several EER concepts, such as categories (union types) and attributes of relationships.

Program 14.1 shows one possible set of ODL class definitions for the UNIVERSITY database. In general, there may be several possible mappings from an object schema diagram (or EER schema diagram) into ODL classes. Program 14.1 shows the straightforward way of mapping part of the UNIVERSITY database. Entity types are mapped into ODL classes, and inheritance is done using ***extends***. However, there is no direct way to map categories (union types) or to do

multiple inheritance. In Program 14.1 the classes PERSON, FACULTY, STUDENT, and GRAD_STUDENT have the extents PERSONS, FACULTY, STUDENTS, and GRAD_STUDENTS, respectively. Both FACULTY and STUDENT extends PERSON and GRAD_STUDENT extends STUDENT. Hence, the collection of STUDENTS (and the collection of FACULTY) will be constrained to be a subset of the collection of PERSONs at any time. Similarly, the collection of GRAD_STUDENTs will be a subset of STUDENTs. At the same time, individual STUDENT and FACULTY objects will inherit the properties (attributes and relationships) and operations of PERSON, and individual GRAD_STUDENT objects will inherit those of STUDENT.



(a)



(b)

Figure 6.1 An example of a database schema. (a) Graphical notation for representing ODL schemas. (b) A graphical object database schema for part of the UNIVERSITY database

The classes DEPARTMENT, COURSE, SECTION, and CURR_SECTION in Program 6.1 are straightforward mappings of the corresponding entity types in Figure 6.1 (b).

Program 6.1: Possible ODL schema for the UNIVERSITY database

```
class PERSON
(       extent          PERSONS
key             Ssn )
{       attribute       struct Pname { string          Fname,
 string  Mname,
 string  Lname } Name;
attribute       string                          Ssn;
attribute       date                            Birth_date;
attribute       enum Gender{M, F}               Sex;
```

```
attribute     struct Address { short          No,
  string      Street,
  short       Apt_no,
  string      City,
  string      State,
  short       Zip }     Address;
short               Age(); };
class FACULTY extends PERSON
(       extent      FACULTY )
{       attribute   string      Rank;
attribute     float         Salary;
attribute     string        Office;
attribute     string        Phone;
relationship  DEPARTMENT Works_in inverse DEPARTMENT::Has faculty;
relationship  set<GRAD_STUDENT> Advises inverse GRAD_STUDENT::Advisor;
relationship  set<GRAD_STUDENT>
On_committee_ofinverseGRAD_STUDENT::Committee;
void          give_raise(in float raise);
void          promote(in string new rank); };
class GRADE
(       extent GRADES )
{
attribute     enum GradeValues{A,B,C,D,F,l, P} Grade;
relationship  SECTION Section inverse SECTION::Students;
relationship  STUDENT Student inverse STUDENT::Completed_sections; };
class STUDENT extends PERSON
(       extent      STUDENTS )
{       attribute   string      Class;
attribute     Department  Minors_in;
relationship  Department  Majors_in inverse DEPARTMENT::Has_majors;
relationship  set<GRADE> Completed_sections inverse GRADE::Student;
relationship  set<CURR_SECTION>
```

```
Registered_in INVERSE CURR_SECTION::Registered_students;
void        change_major(in string dname) raises(dname_not_valid);
float       gpa();
void register(in short secno) raises(section_not_valid);
void assign_grade(in short secno; IN GradeValue grade)
raises(section_not_valid,grade_not_valid); };
class DEGREE
{   attribute   string  College;
attribute   string  Degree;
attribute   string  Year; };
class GRAD_STUDENT extends STUDENT
(   extent      GRAD_STUDENTS )
{   attribute   set<Degree> Degrees;
relationship    Faculty advisor inverse FACULTY::Advises;
relationship    set<FACULTY> Committee inverse FACULTY::On_committee_of;
void assign_advisor(in string Lname; in string Fname) raises(facuIty_not_valid);
void assign_committee_member(in string Lname; in string Fname) raises(facuIty_not_valid); };
class DEPARTMENT
(   extent      DEPARTMENTS
key         Dname )
{   attribute   string  Dname;
attribute   string  Dphone;
attribute   string  Doffice;
attribute   string  College;
attribute   FACULTY Chair;
relationship set<FACULTY> Has_faculty inverse FACULTY::Works_in;
relationship set<STUDENT> Has_majors inverse STUDENT::Majors_in;
relationship set<COURSE> Offers inverse COURSE::Offered_by; };
class COURSE
(   extent      COURSES
key         Cno )
{   attribute   string Cname;
```

```
attribute       string Cno;
attribute       string Description;
relationship    set<SECTION> Has_sections inverse SECTION::Of_course;
relationship    <DEPARTMENT> Offered_by inverse DEPARTMENT::Offers; };
class SECTION
(       extent          SECTIONS )
{       attribute       short   Sec_no;
attribute       string  Year;
attribute       enum    Quarter{Fall, Winter, Spring, Summer}
Qtr;
relationship    set<Grade> Students inverse Grade::Section;
relationship    COURSE Of_course inverse COURSE::Has_sections; };
class CURR_SECTION extends SECTION
(       extent          CURRENT_SECTIONS )
{       relationship    set<STUDENT> Registered_students
inverse STUDENT::Registered_in
void            register_student(in string Ssn)
raises(student_not_valid, section_full); };
```

Thus, Multiple inheritance of interfaces by a class is allowed, as is multiple inheritance of interfaces by another interface. However, with ***extends*** (class) inheritance, multiple inheritance is not permitted. Hence, a class can inherit from at most one class (in addition to inheriting from zero or more interfaces).

The Object Query Language (OQL)

The object query language OQL is the query language proposed for the ODMG object model. It is designed to work closely with the programming languages for which an ODMG binding is defined, such as C++, Smalltalk, and Java. Hence, an OQL query embedded into one of these programming languages can return objects that match the type system of that language. Additionally, the implementations of class operations in an ODMG schema can have their code written in these programming languages. The OQL syntax for queries is similar to the syntax of the relational standard query language SQL, with additional features for ODMG concepts, such as object identity, complex objects, operations, inheritance, polymorphism, and

relationships.

The basic OQL syntax is a *select … from … where …* structure, as it is for SQL. For example, the query to retrieve the names of all departments in the college of 'Engineering' can be written as follows:

```
Q0:   select  D.Dname
from  D in DEPARTMENTS
where  D.College = 'Engineering';
```

In general, an entry point to the database is needed for each query, which can be any named persistent object. For many queries, the entry point is the name of the extent of a class. Recall that the extent name is considered to be the name of a persistent object whose type is a collection (in most cases, a set) of objects from the class. Looking at the extent names in Program 14.1, the named object DEPARTMENTS is of type set<DEPARTMENT>; PERSONS is of type set<PERSON>; FACULTY is of type set<FACULTY>; and so on.

The use of an extent name—DEPARTMENTS in Q0—as an entry point refers to a persistent collection of objects. Whenever a collection is referenced in an OQL query, we should define an *iterator variable*—D in Q0—that ranges over each object in the collection. In many cases, as in Q0, the query will select certain objects from the collection, based on the conditions specified in the where clause. In Q0, only persistent objects D in the collection of DEPARTMENTS that satisfy the condition D.College = 'Engineering' are selected for the query result. For each selected object D, the value of D.Dname is retrieved in the query result. Hence, the *type of the result* for Q0 is bag<string> because the type of each Dname value is string (even though the actual result is a set because Dname is a key attribute). In general, the result of a query would be of type bag for select … from … and of type set for select distinct … from … , as in SQL (adding the keyword distinct eliminates duplicates).

Using the example in Q0, there are three syntactic options for specifying iterator variables:

```
D in DEPARTMENTS
DEPARTMENTS D
DEPARTMENTS AS D
```

The named objects used as database entry points for OQL queries are not limited to the names of extents. Any named persistent object, whether it refers to an atomic (single) object or to a collection object, can be used as a database entry point.

In general, the result of a query can be of any type that can be expressed in the ODMG object

model. A query does not have to follow the select … from … where … structure; in the simplest case, any persistent name on its own is a query, whose result is a reference to that persistent object. For example, the query

Q1: DEPARTMENTS;

returns a reference to the collection of all persistent DEPARTMENT objects, whose type is set<DEPARTMENT>. Similarly, suppose we had given a persistent name CS_DEPARTMENT to a single DEPARTMENT object (the Computer Science department); then, the query

Q1A: CS_DEPARTMENT;

returns a reference to that individual object of type DEPARTMENT. Once an entry point is specified, the concept of a path expression can be used to specify a path to related attributes and objects. A ***path expression*** typically starts at a persistent object name, or at the iterator variable that ranges over individual objects in a collection. This name will be followed by zero or more relationship names or attribute names connected using the dot notation. For example, referring to the UNIVERSITY database in Program 14.1, the following are examples of path expressions, which are also valid queries in OQL:

Q2: CS_DEPARTMENT.Chair;

Q2A: CS_DEPARTMENT.Chair.Rank;

Q2B: CS_DEPARTMENT.Has_faculty;

The first expression Q2 returns an object of type FACULTY, because that is the type of the attribute Chair of the DEPARTMENT class. This will be a reference to the FACULTY object that is related to the DEPARTMENT object whose persistent name is CS_DEPARTMENT via the attribute Chair; that is, a reference to the FACULTY object who is chairperson of the Computer Science department. The second expression Q2A is similar, except that it returns the Rank of this FACULTY object (the Computer Science chair) rather than the object reference; hence, the type returned by Q2A is string, which is the data type for the Rank attribute of the FACULTY class.

Path expressions Q2 and Q2A return single values, because the attributes Chair (of DEPARTMENT) and Rank (of FACULTY) are both single-valued and they are applied to a single object. The third expression, Q2B, is different; it returns an object of type set<FACULTY> even when applied to a single object, because that is the type of the relationship Has_faculty of the DEPARTMENT class. The collection returned will include a

set of references to all FACULTY objects that are related to the DEPARTMENT object whose persistent name is CS_DEPARTMENT via the relationship Has_faculty; that is, a set of references to all FACULTY objects who are working in the Computer Science department.

Now, to return the ranks of Computer Science faculty, we cannot write

Q3′: CS_DEPARTMENT.Has_faculty.Rank;

because it is not clear whether the object returned would be of type set<string> or bag<string> (the latter being more likely, since multiple faculty may share the same rank). Because of this type of ambiguity problem, OQL does not allow expressions such as Q3′. Rather, one must use an iterator variable over any collections, as in Q3A or Q3B below:

```
Q3A:  select    F.Rank
from    F in CS_DEPARTMENT.Has_faculty;
Q3B:  select    distinct F.Rank
from    F in CS_DEPARTMENT.Has_faculty;
```

Here, Q3A returns bag<string> (duplicate rank values appear in the result), whereas Q3B returns set<string> (duplicates are eliminated via the distinct keyword). Both Q3A and Q3B illustrate how an iterator variable can be defined in the from clause to range over a restricted collection specified in the query. The variable F in Q3A and Q3B ranges over the elements of the collection CS_DEPARTMENT.Has_faculty, which is of type set<FACULTY>, and includes only those faculty who are members of the Computer Science department.

6.4 An overview of SQL3

SQL (Structured Query Language) was first specified in the 1970s and underwent enhancements in 1989 and 1992. The language is continuing its evolution toward a new standard called ***SQL3***, which adds object-oriented and other features. SQL can be extended to deal simultaneously with tables from the relational model and classes and objects from the object model. The SQL3 standard includes the following parts:

- SQL/Framework, SQL/Foundation, SQL/Bindings, SQL/Object.
- New parts addressing temporal, transaction aspects of SQL.
- SQL/CLI (Call Level Interface).
- SQL/PSM (Persistent Stored Modules).

SQL/Foundation deals with new data types, new predicates, relational operations, cursors, rules and triggers, user-defined types, transaction capabilities, and stored routines. SQL/CLI

(Call Level Interface) provides rules that allow execution of application code without providing source code and avoids the need for preprocessing. It provides a new type of language binding and is analogous to dynamic SQL in SQL-92. Based on Microsoft ODBC (Open Database Connectivity) and SQL Access Group's standard, it contains about 50 routines for tasks such as connection to the SQL server, allocating and deallocating resources, obtaining diagnostic and implementation information, and controlling termination of transactions. SQL/PSM (Persistent Stored Modules) specifies facilities for partitioning an application between a client and a server. The goal is to enhance performance by minimizing network traffic. SQL/Bindings includes Embedded SQL and Direct Invocation as in SQL-92. Embedded SQL has been enhanced to include additional exception declarations. SQL/Temporal deals with historical data, time series data, and other temporal extensions, and it is being proposed by the TSQL2 committee. SQL/Transaction specification formalizes the XA interface for use by SQL implementers.

New types of operations have been added to SQL3. These include:

- SIMILAR- It allows the use of regular expressions to match character strings.
- UNKNOWN- Boolean values have been extended with this operation when a comparison yields neither true nor false because some values may be null.
- Linear Recursion- A major new operation is linear recursion for specifying recursive queries. To illustrate this, suppose we have a table called PART_TABLE(Part1, Part2), which contains a tuple <p1, p2> whenever part p1 contains part p2 as a component. A query to produce the bill of materials for some part p1 (that is, all component parts needed to produce p1) is written as a recursive query as follows:

WITH RECURSIVE

BILL_MATERIAL (Part1, Part2) AS

(SELECT Part1, Part2

FROM PART_TABLE

WHERE Part1 = 'p1'

UNION ALL

SELECT PART_TABLE(Part1), PART_TABLE(Part2)

FROM BILL_MATERIAL, PART_TABLE

WHERE PART_TABLE.Part1 = BILL_MATERIAL(Part2))

*SELECT * FROM BILL_MATERIAL*

ORDER BY Part1, Part2;

The final result is contained in BILL_MATERIAL(Part1, Part2). The UNION ALL operation is evaluated by taking a union of all tuples generated by the inner block until no new tuples can be generated. Because SQL2 lacks recursion, it was left to the programmer to accomplish it by appropriate iteration.

For security in SQL3, the concept of ***role*** is introduced, which is similar to a "job description" and is subject to authorization of privileges. The actual persons (user accounts) that are assigned to a role may change, but the role authorization itself does not have to be changed. SQL3 also includes syntax for the specification and use of ***triggers*** as active rules. Triggering events include the INSERT, DELETE, and UPDATE operations on a table. The trigger can be specified to be considered BEFORE or AFTER the triggering event. This feature is present in both of the ORDBMS systems we discussed. The concept of ***trigger granularity*** is included in SQL3, which allows the specification of both row-level triggers (the trigger is considered for each affected row) or statement-level trigger (the trigger is considered only once for each triggering event). For distributed (client-server) databases, the concept of a ***client module*** is included in SQL3. A client module may contain externally invoked procedures, cursors, and temporary tables, which can be specified using SQL3 syntax.

SQL3 also is being extended with programming language facilities. Routines written in computationally complete SQL with full matching of data types and an integrated environment are referred to as ***SQL routines***. To make the language computationally complete, the following programming control structures are included in the SQL3 syntax: CALL/RETURN, BEGIN/END, FOR/END_FOR, IF/THEN/ELSE/END_IF, CASE/END_CASE, LOOP/END_LOOP, WHILE/END_WHILE, REPEAT/UNTIL/END_REPEAT, and LEAVE. Variables are declared using DECLARE, and assignments are specified using SET. ***External routines*** refer to programs written in a host language (ADA, C, COBOL, PASCAL, etc.), possibly containing embedded SQL and having possible type mismatches. The advantage of external routines is that there are existing libraries of such routines that are broadly used, which can cut down a lot of implementation effort for applications. On the other hand, SQL routines are more "pure," but they have not been in wide use. SQL routines can be used for server routines (schema-level routines or modules) or as client modules, and they may be procedures or functions that return values.

A number of built-in functions enhance the capability of SQL3. They are used to manage

handles, which in turn are classified into ***environment handles*** that refer to capabilities, connection handles that are connections to servers, and ***statement handles*** that manage SQL statements

The SQL/Object specification extends SQL-92 to include object-oriented capabilities. New data types include Boolean, character, and ***binary large objects (LOBs)***, and large object locators. SQL3 proposes LOB manipulation within the DBMS without having to use external files. Certain operators do not apply to LOB-valued attributes—for example, arithmetic comparisons, group by, and order by. On the other hand, retrieval of partial value, LIKE comparison, concatenation, substring, position, and length are operations that can be applied to LOBs.

Under SQL/Foundation and SQL/Object Specification, SQL allows user-defined data types, type constructors, collection types, user-defined functions and procedures, support for large objects, and triggers. Objects in SQL3 are of two types:

- Row or tuple types whose instances are tuples in tables.
- Abstract Data Types (shortened as ADT or value ADT), which are any general types used as components of tuples.

A ***row type*** may be defined using the syntax

CREATE ROW TYPE row_type_name (<component declarations>);

In SQL3 a construct similar to class definition is provided whereby the user can create a named user-defined type with its own behavioral specification and internal structure; it is known as an ***Abstract Data Type (ADT)***. The general form of an ADT specification is:

CREATE TYPE <type-name> (list of component attributes with individual types declaration of EQUAL and LESS THAN functions declaration of other functions (methods));

An ADT has a number of ***user-defined functions*** associated with it. The syntax is

FUNCTION <name> (<argument_list>) RETURNS <type>;

Two types of functions can be defined: internal SQL3 and external. Internal functions are written in the extended (computationally complete) version of SQL. External functions are written in a host language, with only their signature (interface) appearing in the ADT definition. The form of an external function definition is

DECLARE EXTERNAL <function_name> <signature>

LANGUAGE <language_name>;

Many ORBDMSs have taken the approach of defining a set of ADTs and associated functions

for specific application domains, and packaging them together. For example, the Data Blades in Informix Universal Server and the cartridges in Oracle can be considered as such packages or libraries of ADTs for specific application domains.

ADTs can be used as the types for attributes in SQL3 and the parameter types in a function or procedure, and as a source type in a distinct type. ***Type Equivalence*** is defined in SQL3 at two levels. Two types are ***name equivalent*** if and only if they have the same name. Two types are ***structurally equivalent*** if and only if they have the same number of components and the components are pairwise type equivalent. Under SQL-92, the definition of UNION-compatibility among two tables is based on the tables being structurally equivalent. Operations on columns, however, are based on name equivalence.

Attributes and functions in ADTs are divided into three categories:

- PUBLIC (visible at the ADT interface).
- PRIVATE (not visible at the ADT interface).
- PROTECTED (visible only to subtypes).

It is also possible to define virtual attributes as part of ADTs, which are computed and updated using functions. SQL3 has rules for dealing with inheritance (specified via the UNDER keyword), overloading, and resolution of functions. They can be summarized as follows:

Inheritance

- All attributes are inherited.
- The order of supertypes in the UNDER clause determines the inheritance hierarchy.
- An instance of a subtype can be used in every context in which a supertype instance is used.

Overloading

A subtype can redefine any function that is defined in its supertype, with the restriction that the signature be the same.

Resolution of Functions

- When a function is called, the best match is selected based on the types of all arguments.
- For dynamic linking, the runtime types of parameters are considered.
- SQL3 supports constructors for collection types, which can be used for creating nested structures for complex objects. List, set, and multiset are supported as built-in type constructors. Arguments for these type constructors can be any other type, including row types, ADTs, and other collection types. Instances of these types can be treated as

tables for query purposes. Collections can be unnested by correlating derived tables in SQL3.

Another facility in SQL3 is the ***supertable/subtable facility***, which is not equivalent to super and subtypes and no substitutability is assumed. However, a subtable inherits every column from its supertable; every row of a subtable corresponds to one and only one row in the supertable; every row in the supertable corresponds to at most one row in a subtable. INSERT, DELETE, and UPDATE operations are appropriately propagated.

6.5 Nested relations and collections

The nested relational data model follows the concept of ***non-normal form relations.*** No commercial DBMS has chosen to implement this concept in its original form. The nested relational model removes the restriction of the first normal form from the basic relational model, and thus is also known as the ***Non-1NF or Non-First Normal Form (NFNF) or NF²*** relational model. In the basic relational model—also called the ***flat relational model***—attributes are required to be single-valued and to have atomic domains. The nested relational model allows composite and multivalued attributes, thus leading to complex tuples with a hierarchical structure. This is useful for representing objects that are naturally hierarchically structured. For example, to define the DEPT schema as a nested structure, we can write the following:

DEPT = (DNO, DNAME, MANAGER, EMPLOYEES, PROJECTS, LOCATIONS)

EMPLOYEES = (ENAME, DEPENDENTS)

PROJECTS = (PNAME, PLOC)

LOCATIONS = (DLOC)

DEPENDENTS = (DNAME, AGE)

First, all attributes of the DEPT relation are defined. Next, any nested attributes of DEPT—namely, EMPLOYEES, PROJECTS, and LOCATIONS—are themselves defined. Next, any second-level nested attributes, such as DEPENDENTS of EMPLOYEES, are defined, and so on. All attribute names must be distinct in the nested relation definition. Notice that a nested attribute is typically a ***multivalued composite attribute,*** thus leading to a "nested relation" within each tuple.

When a nested relational database schema is defined, it consists of a number of external relation schemas; these define the top level of the individual nested relations. In addition, nested attributes are called ***internal relation schemas***, since they define relational structures

that are nested inside another relation. In our example, DEPT is the only external relation. All the others—EMPLOYEES, PROJECTS, LOCATIONS, and DEPENDENTS—are internal relations. Finally, ***simple attributes*** appear at the leaf level and are not nested. We can represent each relation schema by means of a tree structure, where the root is an external relation schema, the leaves are simple attributes, and the internal nodes are internal relation schemas. Notice the similarity between this representation and a hierarchical schema.

It is important to be aware that the three first-level nested relations in DEPT represent independent information. Hence, EMPLOYEES represents the employees working for the department, PROJECTS represents the projects controlled by the department, and LOCATIONS represents the various department locations. The relationship between EMPLOYEES and PROJECTS is not represented in the schema; this is an M:N relationship, which is difficult to represent in a hierarchical structure.

Extensions to the relational algebra and to the relational calculus, as well as to SQL, have been proposed for nested relations. Here, we illustrate two operations, NEST and UNNEST, that can be used to augment standard relational algebra operations for converting between nested and flat relations. Consider the flat EMP_PROJ relation and suppose that we project it over the attributes SSN, PNUMBER, HOURS, ENAME as follows:

$EMP\_PROJ\_FLAT \leftarrow \pi_{SSN, ENAME, PNUMBER, HOURS}(EMP\_PROJ)$

To create a nested version of this relation, where one tuple exists for each employee and the (PNUMBER, HOURS) are nested, we use the NEST operation as follows:

$EMP\_PROJ\_NESTED \leftarrow NEST_{PROJS = (PNUMBER, HOURS)}(EMP\_PROJ\_FLAT)$

The effect of this operation is to create an internal nested relation PROJS = (PNUMBER, HOURS) within the external relation EMP_PROJ_NESTED. Hence, NEST groups together the tuples with the same value for the attributes that are not specified in the NEST operation; these are the SSN and ENAME attributes in our example. For each such group, which represents one employee in our example, a single nested tuple is created with an internal nested relation PROJS = (PNUMBER, HOURS). Notice the similarity between nesting and grouping for aggregate functions. In the former, each group of tuples becomes a single nested tuple; in the latter, each group becomes a single summary tuple after an aggregate function is applied to the group.

The UNNEST operation is the inverse of NEST. We can reconvert EMP_PROJ_NESTED to EMP_PROJ_FLAT as follows:

EMP_PROJ_FLAT ã UNNEST$_{PROJS = (PNUMBER, HOURS)}$*(EMP_PROJ_NESTED)*

Here, the PROJS nested attribute is flattened into its components PNUMBER, HOURS.

6.6 Implementation Issues for Extended type

There are various implementation issues regarding the support of an extended type system with associated functions (operations).

- The ORDBMS must dynamically link a user-defined function in its address space only when it is required. As we saw in the case of the two ORDBMSs, numerous functions are required to operate on two-or three-dimensional spatial data, images, text, and so on. With a static linking of all function libraries, the DBMS address space may increase by an order of magnitude. Dynamic linking is available in the two ORDBMSs that we studied.
- Client-server issues deal with the placement and activation of functions. If the server needs to perform a function, it is best to do so in the DBMS address space rather than remotely, due to the large amount of overhead. If the function demands computation that is too intensive or if the server is attending to a very large number of clients, the server may ship the function to a separate client machine. For security reasons, it is better to run functions at the client using the user ID of the client. In the future functions are likely to be written in interpreted languages like JAVA.
- It should be possible to run queries inside functions. A function must operate the same way whether it is used from an application using the application program interface (API), or whether it is invoked by the DBMS as a part of executing SQL with the function embedded in an SQL statement. Systems should support a nesting of these "callbacks."
- Because of the variety in the data types in an ORDBMS and associated operators, efficient storage and access of the data is important. For spatial data or multidimensional data, new storage structures such as R-trees, quad trees, or Grid files may be used. The ORDBMS must allow new types to be defined with new access structures. Dealing with large text strings or binary files also opens up a number of storage and search options. It should be possible to explore such new options by defining new data types within the ORDBMS.

6.7 Comparing OODBMS & ORDBMS

We conclude our treatment of object-relational DBMSs and object-oriented DBMSs with a brief comparison of the two types of system. For the purposes of the comparison, we examine the systems from three perspectives: data modeling (Table 6.1), data access (Table 6.2), and data sharing (Table 6.3). We assume that future ORDBMSs will be compliant with the

SQL:2011 standard.

Table 6.1 Data modeling comparison of ORDBMS and OODBMS.

FEATURE	ORDBMS	OODBMS
Object identity (OID)	Supported through REF type	Supported
Encapsulation	Supported through UDTs	Supported but broken for queries
Inheritance	Supported (separate hierarchies for UDTs and tables)	Supported
Polymorphism	Supported (UDF invocation based on the generic function)	Supported as in an object-oriented programming model language
Complex objects	Supported through UDTs	Supported
Relationships	Strong support with user-defined referential integrity constraints	Supported (for example, using class libraries)

Table 6.2 Data access comparison of ORDBMS and OODBMS.

FEATURE	ORDBMS	OODBMS
Creating and accessing persistent data	Supported but not transparent	Supported but degree of transparency differs between products
Ad hoc query facility	Strong support	Supported through ODMG 3.0
Navigation	Supported by REF type	Strong support
Integrity constraints	Strong support	No support
Object server/page server	Object server	Either
Schema evolution	Limited support	Supported but degree of support differs between products

Table 6.3 Data sharing comparison of ORDBMS and OODBMS.

FEATURE	ORDBMS	OODBMS
ACID transactions	Strong support	Supported
Recovery	Strong support	Supported but degree of support differs between products
Advanced transaction models	No support	Supported but degree of support differs between products
Security, integrity, and views	Strong support	Limited support

6.8 Summary

- The *ODMG object model* is the data model upon which the object definition language (ODL) and object query language (OQL) are based.
- *Objects* and *literals* are the basic building blocks of the object model. The main difference between the two is that an object has both an object identifier and a state (or current value), whereas a literal has only a value but no object identifier.
- An object is described by four characteristics: *(1) identifier*, *(2) name*, *(3) lifetime*, *and (4) structure*. There are three types of literals: *(1) atomic*, *(2) collection*, *and (3) structured.*
- In the ODMG object model, two types of inheritance relationships exist: behavior only inheritance and state plus behavior inheritance.
- The ODL is designed to support the semantic constructs of the ODMG 2.0 object model and is independent of any particular programming language. Its main use is to create object specifications—that is, classes and interfaces.
- The nested relational model removes the restriction of the first normal form from the basic relational model, and thus is also known as the *Non-1NF or Non-First Normal Form (NFNF) or* NF^2 relational model.

6.9 Key Terms

- **Atomic literals:** They correspond to the values of basic data types and are predefined.
- **Structured literals:** They correspond roughly to values that are constructed using the

tuple constructor.

- **Collection literals:** They specify a value that is a collection of objects or values but the collection itself does not have an Object_Id.
- **SQL Routines:** Routines written in computationally complete SQL with full matching of data types and an integrated environment are referred to as SQL routines.

6.10 Check Your Progress

Short- Answer Type

Q1) Full form of ODMG-

a) Object Data Merging Goal b) Object Data Management Goal

c) Object Data Merging Guide d) Object Data Management Group

Q2) A major new operation in SQL3 is _____ for specifying recursive queries.

Q3) _____ literals correspond to the values of basic data types and are predefined.

Q4) The nested relational data model is also known as the Non-1NF data model. (True/ False?)

Q5) ORDBMS supports advanced transaction models for data sharing. (True/ False?)

Long- Answer Type

Q1) Explain what the following terms mean in object-oriented database terminology: method, signature, message, collection, extent.

Q2) What are the differences and similarities between objects and literals in the ODMG object model?

Q3) Give a brief overview of SQL3.

Q4) Write a short note on the nested relational data model.

Q5) Differentiate between OODBMS and ORDBMS.

References

- *Fundamentals of Database Systems*, R. Elmasri, S.B. Navathe, Fifth Edition, Pearson Education/Addison Wesley, 2007.

Database Systems: A Practical Approach to Design, Implementation, and Management, Thomas Connolly & Carolyn Begg, Pearson, 6th Edition.

Unit 7: Data Warehouse & Data Mining

7.0 Introduction

Data warehouse is a repository of an organization's electronically stored data [Wikipedia, 2008]. A data ware-house is the consistent store of data which is made available to end users, so that they can understand and use in a business context [Gatziu, and Vavouras, 1999]. It is not just an individual repository product, rather an overall strategy or process for building decision support systems and a knowledge-based architecture & environment that supports everyday tactical decision making as well as long-term business strategy.

Data Mining may be viewed as automated search procedures for discovering credible and actionable insights from large volumes of high dimensional data [Apte, 2003]. Data mining or knowledge discovery is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase revenue, cuts costs, or both [Palace, 1996]. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases [Palace, 1996]. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analysis offered by data mining move beyond the analysis of past events provided by retrospective tools typical of decision support systems [Thearling, 2009].

7.1 Unit Objective

This Unit shall inform the learners on Data Warehous, Data Warehousing, Data Mining.

7.2 Data Warehouse And Data Warehousing

Data warehouse is a repository of an organization's electronically stored data [Wikipedia, 2008]. A data ware-house is the consistent store of data which is made available to end users, so that they can understand and use in a business context [Gatziu, and Vavouras, 1999]. It is not just an individual repository product, rather an overall strategy or process for building decision support systems and a knowledge-based architecture & environment that supports everyday tactical decision making as well as long-term business strategy. Data warehouse is used in the businesses to convert data into business intelligence and making management decisions, based on the facts and not on intuition. The data warehouse environment positions a business to utilize an enterprise-wide data store to link information from diverse sources and make the information accessible for a variety of user purposes, most notably data warehouses are designed to facilitate reporting and strategic analysis. Business analysts must be able to use the warehouse for such strategic purposes as trend identification, forecasting, competitive analysis, and targeted market research. Data warehouse is one ofthe steps on the long road towards the ultimate goal of accomplishing the objectives of a concern.

Data warehousing is defined as a process of centralized data management and retrieval [Palace, 1996]. It is a process of organizing the storage of large, multivariate data sets in a way that facilitates the retrieval of information for analytic purposes. Data warehousing is a collection of decision support technologies, aimed at enabling the knowledge worker (executive, manager, and analyst etc.) to make better and faster decisions [Chaudhuri and Dayal, 1997] [Jarke and Yannis. 1997]. It is expected to present the right information at the right place and at the right time with the right cost in order to support the right decision [Jarke and Yannis, 1997]. Data warehousing is about molding data into information, and storing this information based on the subject rather than application. Centralization of data is needed to maximize user access and analysis [Palace, 1996]. As mentioned by W.H. Inmon, in one of his articles, “the data warehouse environment is the foundation of decision support systems (DSS) [Inmon, 1995]”. Data warehousing has become an important strategy to integrate heterogeneous information

sources, and to enable online analytic processing in organizations. Data warehousing technology comprises a set of new concepts and tools which support the knowledge worker with information material for decision making. Data warehousing makes it possible to:

- Extract archived operational data and overcome inconsistencies between different legacy data formats.
- Integrate data throughout an enterprise, regardless of location, format, or communication requirements.
- Incorporate additional or expert information, discover and represent the knowledge by mining the hidden facts from stored data.

7.2.1 Characteristics of a Data Warehouse

W.H. Inmon, father of data warehousing, defined data warehouse as "A data warehouse is a Subject Oriented, Integrated, Non-volatile, and Time-variant collection of data in support of management's decisions [Inmon, 1995]". Characteristics of data warehouse are given below.

Integrated: Though the data in the data warehouses is scattered around different tables, databases or even servers but the data is integrated consistently in the values of variables, naming conventions and physical data definitions [Inmon, 1995]. Data is integrated from various, heterogeneous operational systems (like database systems, flat files, etc.) and further external data sources (like demographic and statistical databases, WWW, etc.). Before the integration, structural and semantic differences have to be reconciled i.e. data have to be "homogenized" according to a uniform data model. Furthermore, data values from operational systems have to be cleaned in order to get correct data into the data warehouse [Gatziu and Vavouras, 1999].

Nonvolatile: Data is not updated or changed in any way once it is entered the data warehouse. Data is only loaded and accessed. Being the snapshot of operational data on a given specific time, the data in the data warehouses should not be changed or updated once it is loaded from operational system, as the snapshot shows operational data at some moment of time and one expects data warehouse to reflect accurate values of that time frame. There exist only two operations: the time based loading of data, and accessing the loaded data. Modifications to the warehouse data takes place only when modifications to the source data are propagated into the warehouse [Gatziu and Vavouras, 1999].

Subject-Oriented: In data warehousing the prime objective of storing data is to facilitate decision process of a company, and within any company data naturally concentrates around subject areas. Subject included the high level entities of enterprise, like customer, product etc. This leads to the gathering of information around these subjects rather than around the applications or processes. [Minnesota Historical Society, 2002] [Inmon, 1995]

Time-variant: The value of operational data changes on the basis of time. As data warehouse gives accurate picture of operational data at a specific time and the change in the data in warehouse is based on time based change in operational data, data in the data warehouse is called 'time-variant'. The data warehouse contains a place for storing data that are five to ten years old, or older called historical data. Maintaining historical data means that periodical snapshots ofthe corresponding operational data are propagated and stored in the warehouse without overriding previous warehouse states. For instance, this data is used for comparisons, trends, and forecasting but this data is not updated. However, the potential volume of historical data and the associated storage costs must always be considered in relation to their potential business benefits [Gatziu and Vavouras, 1999].

Multidimensional: To facilitate complex analysis and visualization, the data in a warehouse is typically modeled as multidimensional data base. For example, in a sales data warehouse, time ofsale, sales district, salesperson, and product might be some ofthe dimensions of interest [Chaudhuri and Dayal, 1997].

7.2.2 Benefits and Uses of Data Warehousing

Data warehousing represents an ideal vision of maintaining a central repository of all organizational data. Some of the benefits and uses of this central repository are given below.

Common Data Model: A data warehouse provides a common data model for all data of interest regardless of its source. This makes it easier to report and analyze information than it would be if multiple data models were used to retrieve information such as sales invoices, order receipts, general ledger charges, etc. [Wikipedia, 2008] [Jun, 1998] [Caideira, 2008].

Pre-processed Data: Prior to load the data into data warehouse, inconsistencies are identified

and resolved. This greatly simplifies reporting and analysis [Wikipedia, 2008] [Jun, 1998] [Caideira, 2008].

Customized Information: Information in data warehouse is under the control of data warehouse users so that, even if the source system data is purged over time, the information in the warehouse can be stored safely for extended periods of time [Wikipedia, 2008] [Jun, 1998] [Caideira, 2008]. The user can get the desired view ofdata stored in warehouse.

Operational Independency: Because they are separate from operational systems, data warehouses provide retrieval of data without slowing down operational systems [Wikipedia, http://en.wikipedia.Org/wiki/Data_warehouse#History] [Jun, 1998] [Caldeira, 2008].

Sustaining Traditional Systems (Operational System): Data warehouses can work in conjunction with and, hence, enhance the value of operational business applications, notably customer relationship management (CRM) systems [Wikipedia, 2008] [Jun, 1998] [Caldeira, 2008],

Sustaining Decision Support System (DSS): Data warehouses facilitate decision support system applications such as trend reports (e.g., the items with the most sales in a particular area within the last two years), exception reports, and reports that show actual performance versus goals [Wikipedia, http://en.wikipedia.org/wiki/Data_warehouse #History] [Jun, 1998] [Caldeira, 2008].

More Cost-effective Decision Making: A data warehouse allows reduction of staff and computer resources required to support queries and reports against operational and production databases. This typically offers significant savings. Having a data warehouse also eliminates the resource drain on production systems when executing long-running, complex queries and reports [Kimball, 2002-03].

Better Enterprise Intelligence: Increased quality and flexibility of enterprise analysis arises from the multi-tiered data structures of a data warehouse that support data, ranging from detailed transactional level to high-level summary information. Data accuracy and reliability resulted from ensuring that a data warehouse contains only "trusted" data [Kimball, 2002-03].

A business intelligent system i.e. the applicability ofthe data mining with the business decision-making process, allowed for optimizing future proceedings and modifying organizational, financial or technological aspects of company performance [Gill and Hooda, 2009d].

Enhanced Customer Service: An enterprise can maintain better customer relationships by correlating all customer data via single data warehouse architecture [Kimball, 2002]. Business Re-engineering: Allowing unlimited analysis of enterprise information often provides insights into enterprise processes that may yield breakthrough ideas for reengineering those processes. Defining the requirements for a data warehouse ultimately results in better enterprise goals and measures. Knowing important information to an enterprise will provide direction and priority for re-engineering efforts [Kimball, 2002]. Information System Re-engineering: A warehouse that is based upon enterprise-wide data requirements provides a cost-effective means of establishing both data standardization and operational system interoperability. Data warehouse development can be an effective step in re-engineering the enterprise's legacy systems (traditional systems) [Kimball, 2002].

Central storage: It is a central store that uses very simple data structures with very little assumptions about the relationships between data and against which certain queries are run for the solutions ofthe problems entered by different users.

Data Mart: Data warehouse contains a data mart that is a small warehouse which provides subsets of the main store, summarized information depending on the requirements of a specific group/department. Marts often use multidimensional databases that can speed up query processing as they can have data structures which reflect the most likely questions.

7.2.3 Disadvantages of Data Warehousing

Though data warehousing is ideal for storing and maintaining data in a central repository and provides many benefits, there are also disadvantages of using a data warehouse. Some ofthese are [Wikipedia, 2008] given below.

- Data warehouses are not the optimal environment for unstructured data.
- Because data must be extracted, transformed and loaded into the warehouse, there is an element oflatency in data warehouse data, so it is a time consuming process.
- Integration ofthe data from various sources is quite complex, time consuming and

require a lot of efforts.

- Schema design of data warehouse and data marts are complicated tasks and dynamic also. It is somewhat difficult to select the optimal number of dimensions in a data warehouse.
- Over their life, data warehouses can have high costs that mean the overall cost of maintaining data warehouse may be high if not controlled and managed properly.
- It is fairly hectic to select the best model for mining the data from the central repository, for analysis and retrival of information, reporting and * knowledge discovery.
- Data warehouses can get outdated relatively quickly, that means schema design, their data and physical media as well, required lots ofrenovations. There is a cost ofdelivering sub-optimal information to the organization.
- There is often a fine line between data warehouses and operational systems. So duplicate, expensive functionality may be developed in the data warehouse that, in retrospect, should have been developed in the operational systems and vice versa.
- Serving and satisfying the thousands of different user requirements over the time is unfeasible.

7.2.4 Applications of Data Warehouse

The fundamental reason for building a data warehouse is to improve the quality of information in the organization [Gatziu and Vavouras, 1999]. Some ofthe applications of data warehousing are [Wikipedia, 2008] as given below.

- Credit cards chum analysis
- Insurance fraud analysis
- Call record analysis
- Logistics management

Data warehousing technologies have been successfully deployed in many industries [Chaudhuri and Dayal, 1997] such as:

- Manufacturing (for order shipment and customer support)
- Retail (for user profiling and inventory management)
- Financial services (for claims analysis, risk analysis, credit card analysis, and fraud detection)
- Transportation (for fleet management)

- Telecommunications (for call analysis and fraud detection)
- Utilities (for power usage analysis)
- Healthcare (for outcomes analysis)

7.3 Data Mining

Data Mining may be viewed as automated search procedures for discovering credible and actionable insights from large volumes of high dimensional data [Apte, 2003]. Data mining or knowledge discovery is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase revenue, cuts costs, or both [Palace, 1996]. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases [Palace, 1996]. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analysis offered by data mining move beyond the analysis of past events provided by retrospective tools typical of decision support systems [Thearling, 2009]. Data mining tools can answer business questions that traditionally were too time-consuming to resolve. They scour databases for the hidden patterns, finding predictive information that experts can miss because it lies outside their expectations [Thearling, 2009]. Data mining techniques can be applied on a wide variety of data types including databases, text, spatial data, temporal data, images, and other complex data [Frawley, 1991].

Data mining is a technology to enable data exploration, data analysis as well as data visualization of large databases at a high level of abstraction, without a specific hypothesis in mind. [Williams, Hegland, Markus and Roberts, 1998] It can be defined as the non-trivial extraction of novel, implicit, and actionable knowledge from large datasets that includes [Williams, Hegland, Markus and Roberts, 1998]:

• Extremely large datasets.
• Discovery ofthe non-obvious i.e. hidden and unknown facts.
• Useful knowledge that can improve processes.
• Can not be done manually.

Data Mining employs techniques from statistics, pattern recognition, and machine learning, high performance computers, parallel algorithms, visualizations, database, etc. Many of these methods are also frequently used in vision, speech recognition, image processing, handwriting recognition, and natural language understanding. However, the issues of scalability and automated business intelligence solutions differentiate data

mining from the other applications of machine learning and statistical modeling [Apte, 2003] [Williams, Hegland, Markus and Roberts, 1998],

The current study focuses on the data mining and its most common type of mining techniques; all are described in following chapters in detail.

7.3.1 Web Mining

In a simplified way, web mining is a part ofthe broader area of data mining [Sirmakessis, 2003-04] or it is the data mining adapted to the particularities ofthe web [kaushik, 2007]. It is the use of data mining techniques to automatically discover and extract the information from the web documents and services [Zhang, and Xu, 2009]. So a natural combination of the two areas data mining and WWW, sometimes referred to as Web mining [Cooley, Mobasher, and Srivastava, 1997]. Web mining can involve all of the traditional data mining processes of classification, segmentation, clustering, association, prediction, modeling and examination of sequential pattemsfsometimes with the aid of software graphing applications) [Mena, 2007], [Heverlee, 2005]. While web mining is a part ofthe broader area of data mining, there are several important issues that are unique to the web paradigm and come into play ifsophisticated types of analyses are to be done on server side data collections, including the integration of various data sources such as server access logs, referrer logs, user registration or profile information. Apart from data mining, it deals with resolving difficulties in the identification of users due to missing unique key attributes in data collected and also the identification of user sessions or transactions from usage data, site topologies, and models of user behavior [Sirmakessis, 2003-04]. Unlike data mining, web mining is dependant on the use of software agent to trigger targeted offers as events take place in real time. These software agents (or intelligent software agents/intelligent agents) are defined as being a software program that can perform specific tasks for a user and possesses a degree of intelligence that permits it to perform parts of its tasks autonomously, also to interact with its environment in a useful manner [Abedin and Sohrabi, 2005] like customer retention, risk assessment, fraud detection and counterterrorism [Mena, 2007]. Moreover, the data sets in web mining are extremely large [Jhosi, from http://www.cs.umbc.edu/~ajoshi/web-mine] and it works by performing data analysis via networks, using software agents to mine, collaborate and discover conditions as well as features which can lead to increases in sales, cross-selling opportunities and the targeting ofspecific products or services [Mena, 2007]. It is also important to realize that web data for the most part is completely anonymous, usually incomplete and really unstructured

[kaushik, 2007]. Web mining can be categorized into three main parts: structure mining, content mining and usage mining known as knowledge discovery domains [Galeas, 2009] which is out ofthe scope ofthe current study and thus includes only most common types of data mining techniques for the research work, their analysis and development.

7.4 Historical Background

The concept of data warehousing and data mining dates back to the late 1980s, and has great strides during 1990s with the digital revolutions and dramatic advances in information technology. The data warehousing concept was intended to provide an architectural model for the flow of data from the operational systems to decision support environments that includes data mining too. The historical milieu of data warehouse, data warehousing and data mining is discussed below.

7.4.1 Data Warehouse and Data Warehousing

The concept of data warehousing dates back to the late 1980s: when IBM researchers Barry Devlin, and Paul Murphy, developed the "business data warehouse" (The Story, 2002-04-15) which was published as “Information Warehouse framework” as early in 1987. Other was Teradata Corporation, which originated the database machine that could handle a terabye of data. It was the first industry-hardened massively parallel computer. Teradata became one of the fastest growing companies at that time (Wan 2007). With integrated data warehouse, transactions can be transferred back to the operational systems every day, and this can allow data to be analyzed by companies and organizations. There are a number of devices that will be present in the typical data warehouse. Some ofthese devices are the source data layer, reporting layer, data warehouse layer, and transformation layer.

There are a number different data sources for data warehouses. Some popular forms of data sources are Teradata, Oracle database, or Microsoft SQL Server [Data Warehousing Introduction, 2000-09, from: http://www.exforsys.com /tutorials/ data-warehousing/ data warehousing-introduction, html]. In essence, concept of data warehousing was intended to provide an architectural model for the flow of data from operational systems to decision support environments and attempted to address the various problems associated with this flow, mainly the high costs associated with it. In the absence of a data warehousing architecture, an enormous amount of redundancy was required to support multiple decision support environments. They were developed to meet a growing demand for management information

and analysis that could not be met by operational systems. Operational systems were unable to meet this need for a range of reasons such as the processing load of reporting reduced the response time of the operational systems and development of reports in operational systems often required a writing specific computer program which was slow and expensive. As a result, separate computer databases began to be built that were specifically designed to support management information and analysis purposes. These data warehouses were able to bring the data from a range of different data sources, such as mainframe computers, minicomputers, as well as personal computers and office automation software such as spreadsheet, and integrate this information in a single place. This capability, coupled with user-friendly reporting tools and freedom from operational impacts, has led to a growth of this type of computer system [Data Warehousing: History of Data Warehousing, http://www.dedupe.com/ history.php]. Another important concept that is related to data warehouses is called data transformation. As the name suggests, data transformation is a process in which information transferred from a specific source/s, is cleaned and loaded into a repository [Data Warehousing Introduction, 2000-09].

In larger corporations, it was typical for multiple decision support environments to operate independently. Each environment serves different users but often required much of the same stored data. The process of gathering, cleaning and integrating data from various sources, usually from long-term existing operational systems (usually referred to as legacy systems), was typically in part replicated for each environment. Moreover, the operational systems were frequently re-examined as new decision support requirements emerged. Often new requirements necessitated gathering, cleaning and integrating new data from "data marts" that were tailored for ready access by users [The Story, 2002-04-15]. Data warehousing then became the key trend in corporate computing in the 1990s. Data warehousing is not really a technology trend per se. It was primarily driven by the business environment [Wan, 2007]. Since early 1990s, the data warehouse has become the foundation of advanced decision support applications [Shim, 2002]. Using sophisticated on-line analytical processing (OLAP) and data mining tools, some corporations are able to exploit insights gained from their data warehouse to significantly increase sales [Whiting, 1999], reduce costs [Watson & Haley, 1998] [Whiting, 1999], and offer new and better products or services [Watson & Haley, 1998). The payoff from a well-managed data warehouse can be huge. For instance, a study conducted by

IDC, a leading research firm, found the average return on investments in data warehousing projects to be about 400 percent [Desai, 1999]. By the late 1990s, most large corporations had either built or were planning to build a data warehouse [Joshi & Curtis, 1999]. However, the implementation of a data warehouse is both very expensive and highly risky. Due the technological advances (lower cost for more performance), and increased user's requirements (faster data load cycle times and more features), data warehouse has evolved; the various stages of its evolutions [Data Warehousing: History of Data Warehousing, http://www.dedupe.com/history.php] [Wikipedia, 2008] [Wikipedia, http://en.wikipedia.Org/wiki/Data_warehouse#History] are given below.

Offline Operational Databases: Data warehouses in this initial stage are developed by copying the database of an operational system to an off-line server where the processing load ofreporting does not impact on the operational system's performance. Offline Data Warehouse: Data warehouses at this stage of evolution are updated on a regular time cycle (usually daily, weekly or monthly) from the operational systems and the data is stored in an integrated reporting-oriented data structure.

Real Time Data Warehouse: Data warehouses at this stage are updated on a transaction or event basis, whenever an operational system performs a transaction (e.g. an order or a delivery or a booking etc.).

Integrated Data Warehouse: Data warehouses at this stage are used to generate activity or transactions; which are passed back into the operational systems for use in the daily activity ofthe organization.

Some of the key developments in early years of data warehousing [Wikipedia, http://en.wikipedia.0rg/wiki/Data_wareh0use#Hist0ry] are given below. 1960s: General Mills and Dartmouth College, in a joint research project, developed the terms dimensions andfacts.

1970s: AC Nielsen and IR1 provide dimensional data marts for retail sales. 1983: Teradata introduces a database management system specifically designed for decision support.

1988: Barry Devlin and Paul Murphy publish the article "An architecturefor a business and information systems in IBMSystems Journal" where they introduced the term "business data

warehouse".
1990: Red Brick Systems introduces Red Brick Warehouse, a database management system specifically for data warehousing.
1991: Prism Solutions introduces Prism Warehouse Manager, software for developing a data warehouse.
1991: Bill Inmon publishes the book "Building the Data Warehouse ". 1995: The Data Warehousing Institute, a for-profit organization that promotes data warehousing, is founded.
1996: Ralph Kimball publishes the book "The Data Warehouse Toolkit". 1997: Oracle 8, with support for star queries, is released.

7.4.2 Data Mining

Data mining emerged during the late 1980s, and has great treads during 1990s when the work of mathematicians, logicians, and computer scientists combined to create artificial intelligence (AI) and machine-learning [Buchanan, 2006]. In the 1960s, AI and statistics practitioners developed new algorithms such as regression analysis, maximum likelihood estimates, neural networks, bias reduction, and linear models of classification [Dunham, 2003]. The term 'data mining' was coined during this decade, but it was pejoratively used to describe the practice of wading through data and finding patterns that had no statistical significance [Fayyad, 1996].

Data mining, in several ways, is fundamentally the adaptation from machine learning techniques to business applications. Statistics are the foundation of most technologies on which data mining is built, e.g. regression analysis, standard distribution, standard deviation, standard variance, discriminant analysis, cluster analysis, and confidence intervals. All of these are used to study data and its relationships. Artificial intelligence, (Al), which is built upon heuristics as opposed to statistics, attempts to apply human thought-like processing to statistical problems. Certain Al concepts which were adopted by some high-end commercial products, such as query optimization modules for relational database management systems (RDBMS). Machine learning is the union of statistics and Al. It could be considered an evolution of Al, because it blends Al heuristics with advanced statistical analysis. Machine learning attempts to let computer programs learn about the data they study, such that programs make different decisions based on the qualities of the studied data, using statistics for

fundamental concepts, and adding more advanced Al heuristics and algorithms to achieve its goals [Unc.Edu, http://www.unc.edu/~xluan/258/datamining.html]. Many of these methods are also frequently used in vision, speech recognition, image processing, handwriting recognition, and natural language understanding. However, the issues of scalability and automated business intelligence solutions drive much of and differentiate data mining from the other applications of machine learning and statistical modeling [Apte, 2003], [Williams, Hegland, Roberts, 1998]. Though data mining is the evolution of a field with a long history, the term itself was only introduced relatively recently, in the 1990s. Its roots are traced back along three family lines:

• Classical Statistics

• Artificial Intelligence

• Machine Learning

The longest ofthese three lines is classical statistics. Without statistics, there would be no data mining, as statistics is the foundation of the many technologies on which data mining is built. Certainly, within the heart of today's data mining tools and techniques, classical statistical analysis plays a significant role. Because this approach requires vast computer processing power, it was not practical until the early 1980s, when computers began to offer useful power at reasonable prices. Al found a few applications at the very high end scientific/govemment markets, but the required supercomputers of the era priced AI out ofthe reach of virtually everyone else. The third family line of data mining is machine learning, which is more accurately described as the union of statistics and AI. Data mining is getting increasing acceptance in science and business areas which need to analyze large amounts of data to discover trends which could not otherwise be revealed [Data Warehousing: History of Data Warehousing, http:// www.dedupe.com/history.php] [Ayre, 2006]. Thus data mining techniques are the result of a long process of research and product development. Although data mining is a relatively new term, the technology is not. This evolution began when business data was first stored on computers, continued with improvements in data access, and more recently, generated technologies that allow users to navigate through their data in real time. Companies have used powerful computers to filter through volumes of supermarket scanner data and analyze market research reports for years. However, continuous innovations in computer processing power, disk storage, and statistical software are dramatically increasing the accuracy of analysis while driving down the cost. Data mining algorithms embody

techniques that have existed for at least 10 years, but have only recently been implemented as mature, reliable, understandable tools that consistently outperform older statistical methods [Thearling, 2009].

In the evolution from business data to business information each new stage has been built upon the previous one. For example, dynamic data access is critical for drill-through in data navigation applications, and the ability to store large databases is critical to data mining. From the user's point of view, the four steps listed in Table 1.1 [Thearling, 2009] were revolutionary because they allowed new business questions to be answered accurately and quickly. With data mining, a retailer could use point-of-sale records of customer purchases to send targeted promotions based on an individual's purchase history. By mining demographic data from comment or warranty cards, the retailer could develop products and promotions to appeal to specific customer segments.

Table 1.1 is given below.

Table 1.1: Steps in evolution ofdata mining.

Evolutionary Step	Business Question	Enabling Technologies	Product Providers	Characteristics
Data Collection (1960s)	"What was my total revenue in the last five years?"	CDC Retrospective, static data delivery		
Data Access (1980s) & Data Warehousing	"What were unit sales in New England last March? Drill down to Boston."	Structured Query Language (SQL), ODBC	computers, massive databases Oracle, Sybase, Informix, IBM, Microsoft	Retrospective, dynamic data delivery at record level
Decision Support (1990s)	"What's likely to happen to Boston unit sales next month? Why?"	On-line analytic processing (OLAP), multidimensional databases, data warehouses	Pilot, Comshare, Arbor, Cognos, Microstrategy	Retrospective, dynamic data delivery at multiple levels
Data Mining (Emerging Today)	"What were unit sales in New England last March?"	Relational databases (RDBMS),	Advanced algorithms, multiprocessor	Pilot, Lockheed, IBM, SGI, numerous startups (nascent industry)	Prospective, proactive information delivery

The core components of data mining technology have been under development for decades, in research areas such as statistics, artificial intelligence, and machine learning. Today, the maturity of these techniques, coupled with high-performance relational database engines and broad data integration efforts, make these technologies practical for current data warehouse environments.

7.5 Unit Summary

Data warehouse is a repository of an organization's electronically stored data [Wikipedia, 2008]. A data ware-house is the consistent store of data which is made available to end users, so that they can understand and use in a business context [Gatziu, and Vavouras, 1999]. It is not just an individual repository product, rather an overall strategy or process for building decision support systems and a knowledge-based architecture & environment that supports everyday tactical decision making as well as long-term business strategy.

Data Mining may be viewed as automated search procedures for discovering credible and actionable insights from large volumes of high dimensional data [Apte, 2003]. Data mining or knowledge discovery is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase revenue, cuts costs, or both [Palace, 1996]. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases [Palace, 1996]. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analysis offered by data mining move beyond the analysis of past events provided by retrospective tools typical of decision support systems [Thearling, 2009].

Unit 8: Improved Data Warehouse Architecture For Designing Of Business Intelligence System

8.0 Introduction

In the open market and sea of latest technologies especially, information technology, every enterprise is passing through a tough competition on all fronts. Success of an enterprise does not alone depend on efforts put in by the enterprise but also its ability to take advantage of all available information. This challenge becomes more complicated with the constantly increasing volume and complexity of information. It is further aggravated by increased knowledge-centric enterprises. There is a great need to access variety of information in order to be successful. Data warehouse is one step forward on the long road to achieve the targeted goal in accomplishing the objectives of an organization. Data warehousing is a collection of decision making techniques aimed at enabling the knowledge workers to make better, faster, and quality decisions based on facts, not on intuition. Data warehousing represents an ideal vision of maintaining a central repository of all organizational data. It is a process of organizing and storage of large multivariate data sets in a way that facilitates the retrieval of useful information for analytic purposes for intelligent decision making. Data warehousing has become an important strategy to integrate heterogeneous information sources in organizations, and to enable online analytic processing as well as knowledge discovery. Data mining as a technique of data warehousing is popular in businesses community as a management tool to

enhance the knowledge for decision-making and forecasting for surprising rewards. Thus, the current chapter devoted to demonstrate the functioning of data warehouse and its renovated architecture for designing a business intelligence system.

8.1 Unit Objective

This unit shall introduce the learners about Improved Data Warehouse Architecture For Designing Of Business Intelligence System.

8.2 Activities Involved In Data Warehousing

After analyzing various research papers, it is discovered that data warehousing requires both business and technical expertise and involves the following activities:

• Accurately identifying the business information available in the warehouse

• Identifying and prioritizing subject areas to be included in the data warehouse and managing the scope of each subject area which will be implemented into the warehouse on an iterative basis

• Developing a scaleable architecture to serve as the warehouse's technical & application foundation, identifying and selecting the hardware/software /middleware components to implement the same.

• Extracting, cleaning, aggregating, transforming and validating the data to ensure accuracy and consistency

• Defining the correct level ofsummarization to support business decision making • Establishing a refresh program that is consistent with business needs, timing and cycles

• Providing user-friendly, powerful tools at the desktop to access the data in the warehouse

• Educating the business community about the realm of possibilities that are available to them through data warehousing and establishing a data warehouse help desk and training for users to effectively utilize the desktop tools

• Establishing processes for maintaining, enhancing, and ensuring the ongoing success and applicability ofthe warehouse

8.3 Rework On The Architecture Of Data Warehouse

Architecture, in the context of an organization's data warehousing efforts, is a conceptualization of building the data warehouse. The architecture states the way; the data is stored in a data warehouse from various sources, the manner in which different views are created & stored, and the methods by which data is controlled. Data warehousing may require

integration of data from heterogeneous and distributed sources, which may involve dealing with multiple formats, multiple database systems, distributed databases, cleaning the data, and creating unified logical view of the underlying non homogeneous data with respect to organizational criteria defined by OLAP applications [Bouzeghoub, Fabret and Franfoise, 1999] [Apte, 2003]. Different architectures have been evaluated in this context, but none could be said complete and final in itself. Designing and rolling out a data warehouse is a complex process, consisting of the following activities [Chaudhuri and Dayal, 1997]:

• Define the architecture, do capacity planning, and select the storage servers, database and OLAP servers, and tools

• Select the data sources i.e. internal as well as external sources as required • Sustaining the traditional systems like, operational systems or flat files • Integrate the servers, storage, and client tools (Front End Tools)

• Design the warehouse schema, data marts schema, and views

• Define the physical warehouse organization, data placement, partitioning, and access methods. Connect the sources using gateways, ODBC drivers, or other wrappers

• Design and implement scripts for data extraction, cleaning, transformation, load, and refresh

• Populate the repository with the schema and view definitions, scripts, and other metadata

• Design and implement end-user applications. Roll out (maintain and reengineering) the warehouse and applications

Based on these activities a revised architecture of data warehousing is presented which demonstrates an absolute picture ofthe functioning of data warehousing. Data warehouse architecture consists ofthe interconnected layers discussed below.

8.3.1 Data Acquisition Layer (First Layer)

The first layer includes all the internal as well as external sources from where data is to be collected. Internal sources include operational systems (data bases) and file systems (also known as flat files). External Sources are the outside sources (which are gathered by others for their purposes) and may require lots of pre-processing. At this point data is extracted and integrated, cleaned and transformed, loaded and refreshed into data warehouse. An organization's Enterprise Resource Planning system falls into this layer. Data staging (pre-processing of data) takes source data as an input and output the preprocessed data. Data processed by this layer will be submitted as the input to the second layer.

8.3.2 Designing and Storing of Data (Second Layer)

The second layer includes the meta data repository, data marts and data warehouse as stated below.

Metadata Repository; This is usually more detailed than an operational system data directory. There are dictionaries for the entire warehouse and sometimes dictionaries for the data that can be accessed by a particular reporting and analysis tool.

Schema Design: Shows the model used for structuring the data in data warehouse as well as data marts, E-R model is the most popular and suitable model for the present vast and heterogeneous data.

Data Marts'. A data mart is a subset of an organizational data store, usually oriented to a specific purpose or major data subject that may be distributed to support business needs [Inmon, 1995], Data marts are analytical data stores designed to focus on specific business functions for a specific community within an organization. Data marts are often derived from subsets of data in a data warehouse, though in the bottom-up data warehouse design methodology, the data warehouse is created from the union of organizational data marts. Each data mart is a collection oftables organized according to the particular requirements of a user or group of users. Retrieving a collection of different kinds of data from a normalized warehouse can be complex and time-consuming. Hence there is a need to re-arrange the data so that it can be retrieved more easily. The notion of a mart suggests that it is organized for the ultimate consumers and does not have to follow any particular inherent rules or structures for its organization. Though these marts are organized initially, the requirements are almost certain to change once the user has seen the implications ofthe request.

8.3.3 Access and Delivery of Data (Third Layer)

The data is accessed for analysis, query/ reporting and data mining for the knowledge discovery. This data can be extracted from the data warehouse through the OLAP server and presented to user by using different front end tools. An OLAP (On-Line Analytical Processing) server enables a more sophisticated end-user business model to be applied when navigating the data warehouse. The multidimensional structures allow the user to

analyze the data as they want to view their business, summarizing the data by product line, region, and other key perspectives oftheir business. The main object in OLAP is the cube, which contains the current analytical data in interest ofthe end user. Codd exposed the basic conceptions of OLAP and defined 12 rules used for defining and evaluating OLAP software.

These are multidimensional conceptual view, transparency, accessibility, consistent-reporting performance, client-server architecture, generic dimensionality, dynamic sparse matrix handling, multi-user support, unrestricted cross-dimensional operations, intuitive data manipulation, flexible reporting and unlimited dimensions and

aggregation levels. To support the questions which the user asks, the cubes organize the data in dimensions and measures in multidimensional structure. Data mining analysts try to go beyond OLAP by providing abilities for discovering insights that are computer driven and not end-user driven. Data size is increasing at a far exceeding rate that end users can cope with. Providing solutions when end-users cannot reasonably supply all possible aggregates to pre-compute, or when it is not possible to express an insight as a pre-computed aggregate, is the goal of data mining analytics. Data mining may include any of its techniques for the extraction of the information or the knowledge from data warehouse. Report/Query: Query initiation, formulation, and results presentation are provided to the user. A reporting environment could be a set of preformatted reports. Besides OLAP, other tools may be SQL Server, ORACLE, or any other Business intelligence tools [Bouzeghoub, Fabret and Fran9oise, 1999] [Apte, 2003] [Chaudhuri and Dayal, 1997].

8.4 Predictive Data Mining

The vital goal of data mining is predictions for the unknown values and projecting the future by applying certain techniques of data mining, that's the reason it is named as predictive data mining which offers the knowledge about certain important facts to the decision makers that may be hidden otherwise [Gill and Hooda, 2009d]. Predictive data mining is the most common type of data mining that actually clarify the working of data mining and one that has the most direct business applications [StatSoft, 1984-2006]. The predictive data mining is usually applied to identify data mining projects with the goal to develop a model (or set ofmodels) that can be used to predict some unknown information which may be most decisive for successful existence of the business. For case in point, a credit card company may want to engage in predictive data mining, to developed a model or set of models (e.g., k-nearest neighbor) that can quickly identify transactions which have a high probability of being fraudulent [Gill and Hooda, 2009d].

Predictive mining offers knowledge discovery for certain applications like: ability to detect buying patterns, determine purchase relationships between products, analyzing customer

profiles, ad-revenue forecasting, chum management, credit risk analysis, cross marketing, customer retention, electronic commerce, exception reports, food-service menu analysis, government policy setting, hiring profiles, marketing, medical management, member enrollment, new product development, pharmaceutical research, process control, quality control, store management, student recruiting & retention, warranty analysis and many more. Through the effective employment of data mining technology, managers can quickly analyze operations in the areas akin to production (inventory and supply chain management), marketing, optimal pricing, distribution (wholesale, retail and e-commerce), finance, human resources and customer relationship management [Gill and Hooda, 2009d].

8.5 Uses Of Data Mining

Data mining is used to discover patterns and relationships in the data in order to help make better business decisions, serve customers and gain the competitive edge. The various utilizations of data mining are as follows:

Applicable to Various Organizations: Data mining can be used by any organization profitable, non-profitable, educational institution and even for diverse fields like: retail,

financial, communication, and marketing organizations etc. for strong consumer focus, their retention, employee satisfaction & development, and even for the modernization of certain organization for their long survival and success.

Automated Prediction of Trends and Behaviors: Data mining automates the process of finding predictive information in large databases. A typical example of a predictive problem is targeted marketing. For instance, data mining uses data on past promotional mailings to identify the targets most likely to maximize return on investment in future mailings. Other predictive problems include forecasting bankruptcy and other forms of

default, and identifying segments of a population likely to respond similarly to given events [Thearling, 2009].

Automated Discovery of Previously Unknown Patterns: Data mining tools sweep through databases and identify previously hidden patterns in one step. An example of pattern discovery is the analysis of retail sales data to identify seemingly unrelated products that are often purchased together. Other pattern discovery problems include detecting fraudulent credit card transactions and identifying anomalous data that could represent data entry keying errors [Thearling, 2009].

Analysis of Various Factors in the Market: It enables various 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. It assists businessman to determine the impact on sales, customer satisfaction, and corporate profits [Palace, 1996].

Marketing: uses of data mining in marketing can be divided into following parts: • Market segmentation: Identify the common characteristics of customers who buy the same products from your company [Gill and Hooda, 2009c].

• Target marketing: Identify the segment to target [Gill and Hooda, 2009c]. • Customer churn: Predict which customers are likely to leave the company and go to a competitor.

• Direct marketing: Identify which prospects should be included in a mailing list to obtain the highest response rate.

• Interactive marketing: Predicting the interest of each individual accessing a web site.

• Market basket analysis: Understand the products or services that are commonly purchased together; e.g., beer and diapers.

Extensibility: Data mining techniques can yield the benefits of automation on existing software and hardware platforms, can be implemented on new systems as existing platforms are upgraded and new products developed. When data mining tools are implemented on high performance parallel processing systems, they can analyze massive databases in minutes. Faster processing means that users can automatically experiment with more models to understand complex data. High speed makes it practical for users to analyze huge quantities of data. Larger databases, in turn, yield improved predictions [Thearling, 2009].

Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery. Data mining is ready for application in the business community because it is supported by three technologies that are now sufficiently mature [Thearling, 2009]:

• Massive data collection

• Powerful multiprocessor

• Data mining algorithms

Data mining also facilitates to "drill down" into summary information to view detailed transactional data [Palace, 1996].

8.6 Data Mining And Intelligence

Due to complexity of decision making [Shahzad, from http://www.oracular.

com/whitej>aper_pdfs/DataWarehousingwithOracle.pdfj in this data driven world, competitive forces and socio-economic reality of contemporary organizations has forced them to look for effective acquiring, processing and exploring the cosmic amounts of data that come from dispersed sources for discovering new facts and knowledge [Gill and Hooda, 2009d]. This requires the accumulation of planning and decision making process with a tool such as data mining that can serve in accomplishing their ravenous missions. This amalgamation of data mining and the business decision making is called business
intelligent system [Gill and Hooda, 2009d]. Then knowledge can be acquired from thriving implementation of this automated system (Business Intelligence System) which may include intelligent exploration, integration, aggregation and a multidimensional analysis of data [Forgy, 1965]. Such kind of intelligent systems are meant to provide adequate and reliable up-to-date information on different aspects of enterprise activities [Gero, and Reffat, 2003]. By getting the value-added information through the triumphant implementation of Business Intelligent System (applying data mining methods on the data stored in the data warehouse of the business or a company), decision makers can exploit this extracted information for better formulation and realization of business strategy in order to manage and optimize the obtainable resources so that maximum profits can be achieved with minimum cost, to acquire a competitive periphery around the world and realization of business objectives [Gill and Hooda, 2009d]. Business Intelligence System can be applied for developing different comparative reports, e.g. on historical results, profitability of particular offers, effectiveness of distribution channels along with carrying out simulations of development or forecasting future results. The systems also allows for optimizing future proceedings and modifying organizational, financial or technological aspects of company performance appropriately in term of lucrative implementation of the budgets of an enterprise and confiscate the deviations [Gill and Hooda, 2009d].

8.7 Data Mining Issues

Although data mining aids business man. individuals to analyze the data from different perspective and facilitate them for knowledge discovery and superior decision making
but there are also various social issues that may arise with the use ofthe data mining are stated below.

Privacy: It is the issue of individual privacy. Data mining makes it possible to analyze routine

business transactions and gives a significant amount of information about individuals buying habits and preferences [Palace, 1996]. Therefore, the private data may be revealed that may result in a high personal loss.

Data Integrity: Clearly, data analysis can only be as good as the data that is being analyzed. Integrating conflicting or redundant data from different sources is a key challenge for implementation. For example, a bank may maintain credit cards accounts on several different databases. The addresses (or even the names) of a single cardholder may be different in each. Software must translate data from one system to another and
select the address most recently entered [Palace, 1996].

Application Suitability: A hotly debated technical issue is whether it is better to set up a relational database structure or a multidimensional one. In a relational structure, data is stored in tables, permitting ad hoc queries. In a multidimensional structure, on the other hand, sets of cubes are arranged in arrays, with subsets created according to category. While multidimensional structure facilitates multidimensional data mining, the relational structure performs better in client/server environment. And, with the explosion of the Internet, the world is becoming one big client/server environment [Palace, 1996].

Cost: While system hardware costs have dropped dramatically within the past five years, data mining and data warehousing tend to be self-reinforcing. The more powerful the data mining queries, the greater the utility ofthe information being gleaned from the data, and the greater the pressure to increase the amount of data being collected and maintained, which increases the pressure for faster, more powerful data mining queries. This increases pressure for larger and faster systems which are more expensive [Palace, 1996].

8.8 Techniques Of Data Mining

Data mining techniques are not stand alone but are drawn from various fields like statistics, machine learning, visualization, pattern recognition etc.

Database Techniques:

- Database Design and Moceling (tables, procedures, functions, constraints)
- Database Interface to Data Min ng System
- Efficient Import and Export of Data
- Database Data Visualization
- Database Clustering for Access Efficiency
- Database Performance Tuning (memory usage, query encoding)

• Database Parallel Processing (multiple servers and CPUs) • Distributed Information Repositories (data warehouse)

Data: 3D cubes, distribution charts, curves, surfaces, link graphs, image frames and movies, parallel coordinates Results: Pie charts, scatter plots, box plots, association rules, parallel coordinates, dendograms, temporal evolution

■« 4i.- : ‘ ' .V

Coordinates,

Optimization Techniques:

• Highly nonlinear search space (global versus local maxima)

• Gradient based optimization

• Genetic algorithm based optimization

• Optimization with sampling

8.9 Summary

In this chapter, the key notion of data warehousing and the activities involved in building a data warehouse are portrayed. To comprehend the overall functioning of the data warehouse, a three layered architecture has been designed in which OLAP and data marts are the most significant. Data marts are the views or the subset of whole data kept in a data warehouse and OLAP is used to serve the end users/applications through diverse front end tools. This whole architecture is monitored and administered by the management persons. Data mining is one of the important techniques of data warehousing which has evolved from data warehousing and decision support systems due to advent in information technology and digital revolutions. This has been recalled as predictive data mining. Furthermore, data mining techniques are categorized in to classical techniques and next generation techniques based on their development period and methodologies used. In this chapter, the utility of data mining in mounting business person’s intelligence in better formulation of planning, decision making, reporting and realization of business objectives are elaborated. The predictive data mining provides intelligent exploration, integration, aggregation, multidimensional analysis of data and development of a business intelligence system. Also aids in optimizing future proceedings, modifying organizational, financial and technological aspects of a company or a business performance through up-to-date information, removing the deviations from the standards and maximizing the profits. In the following chapters, certain techniques of data mining are being discussed.

Unit 9: Introduction

Structure

9.0 Introduction

In the past 20 years, machine learning has become an important component of the growing field of data science. With the increase in the amount of available data, there are good reasons to believe that the analysis of data will become more common and become an essential part of technological progress. Being an application of Artificial Intelligence (AI), Machine learning aims at computers to learn like humans. With the help of machine learning, computers can learn automatically without human interference or assistance.

In simple terms, learning is defined as the process of converting experiences into expertise or knowledge. Humans can learn with their past experiences. Similarly, a machine can also learn with the help of past data and algorithms. A learning algorithm considers the training data (experiences) as input and delivers expertise as output. This expertise usually takes the form of another computer program performing some task.

Let's begin with an example. Assume that John likes to listen to music a lot. He makes his choice according to the intensity and tempo of the music. On analyzing the kind of songs he likes, it can be concluded that he prefers songs with both high intensity and tempo. Now, if John listens to a new song, then it can be predicted with the help of his previous song choice that he will like the song or not. This implies that there is a certain algorithm on which John's

song choice works. By following this algorithm anyone can predict whether John will like the new song or not. This example can simply explain the basis of machine learning. To learn how a machine can learn to behave and think like humans, we should first discuss an overview, definition, importance, and framework of machine learning.

9.1 Unit Objectives

After completing this unit, the reader will be able to:

- Gain knowledge about the basic overview of machine learning.
- Illustrate the basic components of the learning process.
- Describe the phases of machine learning lifecycle.
- Learn about various applications of machine learning.
- Discuss the importance and issues in machine learning.

9.2 Definition of Machine Learning

The term Machine Learning was coined by Arthur Samuel, a pioneer in the field of artificial intelligence, in 1959 at IBM. According to him, *"Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed."* [1]

Different authors have different perspectives on machine learning. However, there is no universal definition for machine learning. Some other definitions of machine learning are given below:

1. Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data or both.[2]The field of study known as machine learning is concerned with the question of how to
2. construct computer programs that automatically improve with experience.[3]
3. Machine learning is the hot new thing.

— John L. Hennessy, President of Stanford (2000–2016)

1*Arthur L Samuel. "Some studies in machine learning using the game of checkers". In: IBM Journal of research and development 3.3 (1959), pp. 210–229.*

2*Ethem Alpaydin, Introduction to Machine Learning, The MIT Press, Cambridge, Massachusetts, 2004.*

3*Mitchell T., Machine Learning, McGraw Hill, 1997.*

4. A breakthrough in machine learning would be worth ten Microsofts.

— Bill Gates, Microsoft Co-Founder

5. A computer program is said to learn from ***experience*** *(E)* with respect to some class of ***tasks*** *(T)* and ***performance measure*** *(P)*, if its performance at tasks T, as measured by P, improves with experience E.[3] Table 1.1 provides some situations as examples and depict the corresponding performance, task, and experience.

Table 9.1 Situational examples for tasks, performance, and experience

Situation	Details
Handwriting recognition learning problem	• *Task (T):* Recognising and classifying handwritten words within images • *Performance measure (P):* Percent of words correctly classified • *Training experience (E):* A dataset of handwritten words with given classifications
A robot driving learning problem	• *Task (T):* Driving on highways using vision sensors • *Performance measure (P):* Average distance traveled before an error • *Training experience (E):* A sequence of images and steering commands recorded while observing a human driver
A chess learning problem	• *Task (T):* Playing chess • *Performance measure (P):* Percent of games won against opponents • *Training experience (E):* Playing practice games against itself

A computer program that learns from experience is called a ***machine learning program*** or simply a learning program. Sometimes such a program is also referred to as a ***learner***. Figure 9.1 represents the difference between traditional programming and machine learning.

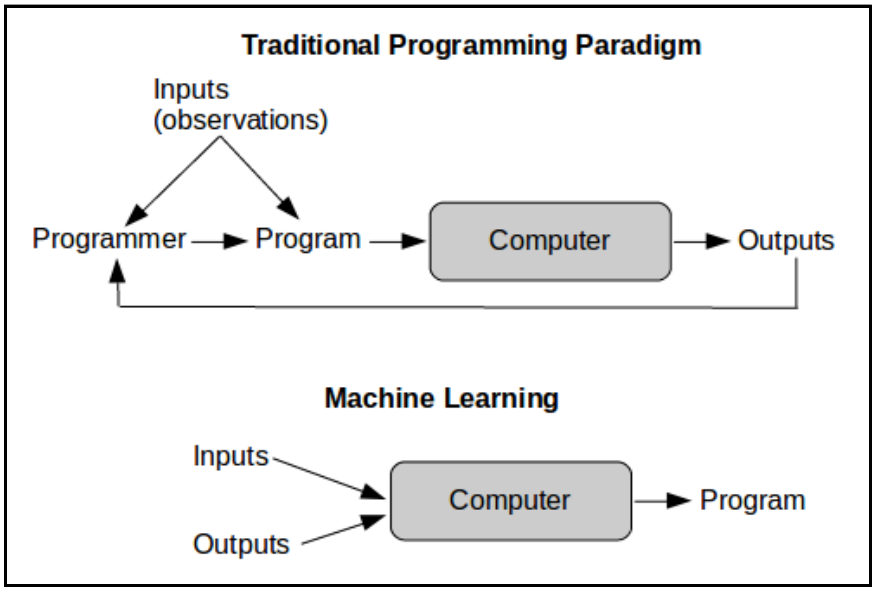


Figure 9.1 Machine learning vs Traditional programming

9.3 Basic Components of Learning Process

Either for humans or a machine, the learning process includes four components, data storage, abstraction, generalization, and evaluation. Figure 1.2 represents the basic components of the learning process.

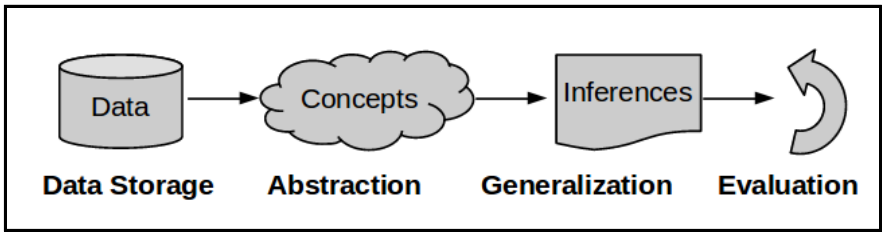


Figure 9.2 Components of Learning Process

1. *Data storage:*

   Storing and retrieving large amounts of data are important components of the learning process. Both humans and computers utilize data storage as a foundation for advanced reasoning. Humans store the data in the brain and retrieve it using electrochemical signals. On the other hand, computers use hard disk drives, flash memory, random access memory, and similar devices to store data and use cables and other technology to retrieve data.

2. *Abstraction:*

   Abstraction being the second component of the learning process, is the process of

extracting knowledge about already stored data. This involves creating general concepts about the data with the help of known models and new models. ***Training*** is the process of fitting a model to a dataset. Once the model is trained, the data is transformed into an abstract form that summarizes the original information.

3. *Generalization:*

   The third component generalization of the learning process describes the process of turning the knowledge about stored data into a form that can be utilized for future action. These actions are to be carried out on tasks that are similar, but not identical, to those that have been seen before. In generalization, the goal is to discover those properties of the data that will be most relevant to future tasks.

4. *Evaluation:*

   Evaluation, being the last component of the learning process, is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilized to effect improvements in the whole learning process.

9.4 Phases of Machine Learning

We are already aware of the definition of machine learning. In simple terms, machine learning provides an automatic learning platform for computer systems without being explicitly programmed. But inquisitively, it becomes interesting to know how a machine learning system works. This can be analyzed by defining the seven phases of machine learning. The cyclic process of machine learning is capable of finding an appropriate solution to problems or projects. Before proceeding with a machine learning process, It is important to understand the problem completely. This helps in analyzing the process better and provides good results.

The machine learning system is based on a *model* and this model is created by providing *training*. The seven phases of a machine learning system include:

- ➔ Data Collection
- ➔ Data Preparation
- ➔ Data Wrangling
- ➔ Data Analyzation
- ➔ Training the model
- ➔ Testing the model
- ➔ Deployment

Data Collection

Collecting or gathering the data is the initial phase of the system. Data collection aims at identifying and obtaining all data-related problems. The data sources for collecting the data include files, databases, the internet, or mobile devices. The efficiency of the data is determined by the quality and quantity of the data collected. More is the data, more accurate will be the prediction. The data obtained after collection is termed as a ***dataset***.

Data Preparation

After gathering the data, the next step is to prepare the data according to the requirement. The data is put into a suitable place and prepared to be used in machine learning training. Initially, all the data is put together and the order of the data is randomized. The further process is divided into:

- *Data Exploration:* In this step, the nature of the data is analyzed. This includes understanding the characteristics, format, and quality of the data. The data is analyzed using correlations, general trends, and outliers.
- *Data Pre-processing*: In this step, the data is pre-processed for its analysis.

Data Wrangling

Data wrangling is one of the most important phases of the machine learning lifecycle. It is the process of cleaning and converting raw data into a usable format. This again contributes to making the data suitable for analysis. Cleaning of data is needed for addressing the quality issues. The issues concerned with data collection are missing values, invalid data, duplicate data, or noise. It is necessary to detect and remove these issues, ultimately cleaning the data using various filtering techniques, to improve the quality of the outcome.

Data Analyzation

The data analysis phase aims at building a machine learning model to analyze the data using distinct analytical techniques and review the output. The steps involved in this phase are:

- Selection of analytical techniques
- Building models
- Review the outcome

Firstly, the type of issue is determined and then the suitable machine learning technique is selected. These techniques can be *classification, regression, cluster analysis, association,* etc. Secondly, the model is built using the prepared data. Lastly, the model is evaluated.

Training the Model

In this phase, the model is trained to improve its performance for a better outcome of the problem. Datasets are used for this purpose using various machine learning algorithms. It is essential to train the model so that it can understand various patterns, rules, and features.

Testing the Model

Once the machine learning model has been trained on a given dataset, then testing of the model is performed. In this phase, the accuracy of the model is checked by providing a test dataset to it. Testing the model determines the percentage accuracy of the model as per the requirement of the project or problem.

Deployment

This is the last phase of the machine learning lifecycle, where the model is deployed in the real-world environment. If the model produces accurate results as per the requirements with desired speed, then the model is deployed in the real system. Before deploying the project, it is also checked whether the performance of the project is improving or not with the available data.

9.5 Importance of Machine Learning

At an early age, only companies stocked their data. The data was stored and processed at the computer centers. With the advancement in computer technology and wireless communication, we all became producers of data. Every time visiting any webpage, buying a product online, posting on social media, even simply surfing the internet generates data. In this age of Big Data, each of us is not only a generator but also a consumer of data. We want to have products and services specialized for us. We want our needs to be understood and interests to be predicted.

The stored data is counted as experience and it helps in making further decisions. For example, in an online shopping platform, thousands of products are sold to hundreds of customers every day. The details of each transaction are stored, like customer Id, product details, date, amount,

offer applied, etc. This results in a lot of data every day. Storing this information helps in analyzing and predicting the choice of customers, to maximize sales and profit. The customers can also get a review of the particular products. However, this prediction for the customer choice is not evident but there exists a certain pattern in the data. Such patterns may help us understand the process or we can use those patterns to make predictions: Assuming that the future, at least the near future, will not be much different from the past when the sample data was collected, the future predictions can also be expected to be right.

Generally, computers use algorithms to solve any issue. An algorithm is defined as a set of instructions that should be carried out to transform the input to output. For example, a sorting algorithm puts a set of numbers as input and the output is their ordered list. However, for some tasks, there is no certain algorithm. For example, predicting customer behavior.

The constantly evolving field of machine learning has led to a rise in demand and importance of the field too. Scientists' preference for machine learning is based on the fact that high-value predictions offer better decisions and smart actions in real-time without any human interference. Machine learning helps in analyzing large amounts of data and also simplifies the task data. The automation process has acquired much prominence and recognition by researchers. Machine learning involves automatic sets of generic methods. These methods have replaced the traditional statistical methods of data extraction and interpretation.

Machine learning is a core part of artificial intelligence. It also helps in finding solutions to the problems in vision, speech recognition, and robotics. The past experiences or example data help in optimizing the computer programs. ***Models*** define some parameters and ***learning*** depicts the execution of computer programs to optimize these parameters of the models using past experiences or training data. These models can be either termed as ***predictive*** for making predictions in the future or ***descriptive*** for gaining knowledge from data, or both. In machine learning, the theory of statistics is utilized to build mathematical models.

9.6 Applications of Machine Learning

Applying machine learning to large databases is known as ***data mining***. In data mining, a large amount of data is processed to build a simple model of use value such as high prediction accuracy. Following is a list of some of the applications of machine learning.

- Machine learning is used in the retail business to study consumer behavior.
- Image recognition is one of the important applications of machine learning. It is used to

identify objects, persons, places, digital images, etc. Automatic friend tagging suggestion in Facebook is one of the popular examples of machine learning in image recognition.

- Machine learning is used in speech recognition. For example, Alexa, Google search by voice, etc.
- In the field of finance, banks analyze their past data to build models for use in credit applications, fraud detection, and stock marketing.
- In manufacturing industries, training models are used for optimization, control, and troubleshooting.
- Learning programs are even used for medical diagnosis.
- In telecommunications, call patterns are analyzed to optimize the network and increase the quality of service.
- In artificial intelligence, it is used to train a system to learn and adapt to changes so that the system designer does not need to predict and come up with solutions for all situations that may occur.
- In the field of science, large amounts of physical, astronomical, and biological data can only be analyzed quickly enough by computers. Due to the constant evolution of the World Wide Web, finding relevant information manually becomes an impossible task.
- Machine learning is applied in the design of computer-controlled vehicles to drive accurately when driving on various types of roads. For example, Google Maps.
- Machine learning methods have been used to develop programs that play games with the users.
- Machine learning is widely used by various e-commerce and entertainment companies such as Amazon Prime, Netflix, etc., for product recommendations to the user.
- Machine learning is applied for filtering spam emails and malware.

9.6.1 Examples of Machine Learning Applications

Let's discuss some of the examples for machine learning applications to have a broader idea about the basics of this field.

Learning Associations

Consider an example of a sports shop. The distributor is interested in knowing if there is any

certain pattern in the purchase of the products. Like, if a customer buys a bat and ball together, then he is likely to also buy batting gloves. After analyzing this customer behavior, it can be interpreted that there must be an association between the set of products {bat, ball} and the set {batting gloves}.

Association rule in machine learning is a method for analyzing the relations between the variables in large databases, using the concept of mutual interest.

The association rule for the above example is represented as:

$$\{bat, ball\} \Rightarrow \{batting\ gloves\}$$

The measure of how likely a customer, who has bought a bat and a ball, to buy batting gloves also is given by the conditional probability

$$P(\{bat, ball\} \mid \{batting\ gloves\})$$

Let this conditional probability is 0.8, then the rule may be stated more precisely as follows:

"80% of customers who buy a bat and a ball also buy batting gloves."

Consider an association rule of the form

$$X \Rightarrow Y,$$

that is, if people buy X then they are also likely to buy Y.

Suppose there is a customer who buys X and does not buy Y . Then that customer is a potential Y customer. Once we find such customers, we can target them for cross-selling. A knowledge of such rules can be used for promotional pricing or product placements.

In finding an association rule $X \Rightarrow Y$, we are interested in learning a conditional probability of the form $P(Y \mid X)$ where Y is the product the customer may buy and X is the product or the set of products the customer has already purchased.

If we may want to make a distinction among customers, we may estimate $P(Y \mid X, D)$ where D is a set of customer attributes, like gender, age, marital status, and so on, assuming that we have access to this information.

Classification

Classification, in machine learning, is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.

Considering an example, there is a training set of data as shown in table 1.2. There are two

attributes "Score1" and "Score2". The class label is called "Result". The class label has two possible values: "Pass" and "Fail". The data can be divided into two categories or classes: The set of data for which the class label is "Pass" and the set of data for which the class label is "Fail".

Let us assume that we have no knowledge about the data other than what is given in the table. Now, the problem can be posed as follows: If we have some new data, say "Score1 = 33" and "Score2 = 28", what value should be assigned to "Result" corresponding to the new data.

Table 9.2 Example data for a classification problem

Score1	22	29	10	31	32
Score2	43	47	49	54	40
Result	Pass	Pass	Fail	Pass	Pass

To answer this question, using the given data alone we need to find the rule or the formula, or the method that has been used in assigning the values to the class label "Result". The problem of finding this rule or formula or the method is the *classification problem*. In general, even the general form of the rule or function, or method will not be known. So several different rules, etc. may have to be tested to obtain the correct rule or function, or method.

Another real-life example of the classification problem can be of a credit card company. The company receives hundreds of thousands of applications for new cards. The applications contain information regarding several attributes like annual salary, age, etc. The problem is to devise a rule to classify the applicants to those who are credit-worthy, who are not credit-worthy, or to those who require further analysis.

Some general features of the classification problems are:

- A classification problem requires that examples be classified into one of two or more classes.
- A classification can have real-valued or discrete input variables.
- A ***discriminant*** of a classification problem is a rule or a function that is used to assign labels to new observations.
- A problem with two classes is often called a ***two-class*** or ***binary*** classification problem.
- A problem with more than two classes is often called a ***multi-class*** classification

problem.

- A problem where an example is assigned multiple classes is called a ***multi-label*** classification problem.

There are several machine learning algorithms for classification. The following are some of the well-known algorithms.

- Logistic regression
- Naive Bayes algorithm
- k-NN algorithm
- Decision tree algorithm
- Support vector machine algorithm
- Random forest algorithm

Regression

In machine learning, a ***regression problem*** is a problem of predicting the value of a numeric variable based on the observed values of the variable. The value of the output variable may be a number, such as an integer or a floating-point value. These are often quantities, such as amounts and sizes. The input variables may be discrete or real-valued.

Let x denote the set of input variables and y the output variable. In machine learning, the general approach to regression is to assume a model, that is, some mathematical relation between x and y, involving some parameters say, θ, in the following form:

$$y = f(x, \theta)$$

The function $f(x, \theta)$ is called the ***regression function***. The machine learning algorithm optimizes the parameters in the set θ such that the approximation error is minimized; that is, the estimates of the values of the dependent variable, y are as close as possible to the correct values given in the training set.

For example, consider Table 1.3 showing the prices of cars. If the input variables are "Age", "Distance" and "Weight" and the output variable is "Price", the model maybe

$$y = f(x, \theta)$$

$$\text{Price} = a_0 + a_1 \times (\text{Age}) + a_2 \times (\text{Distance}) + a_3 \times (\text{Weight})$$

where x = (Age, Distance, Weight) denotes the set of input variables and $\theta = (a_0, a_1, a_2, a_3)$

denotes the set of parameters of the model.

Table 9.3 Example data for a regression problem: Prices of used cars

Price (US$)	Age (Years)	Distance (Kms.)	Weight (Pounds)
14950	25	38500	1170
14500	30	41000	1165
13750	27	41711	1165
12950	23	71138	1245

There are various types of regression techniques available to make predictions. These techniques mostly differ in three aspects, namely, the number and type of independent variables; the type of dependent variables; and the shape of the regression line. Some of these are listed below.

- *Simple linear regression:* There is only one continuous independent variable x and the assumed relation between the independent variable and the dependent variable y is

$$y = a + bx.$$

- *Multivariate linear regression:* There are more than one independent variable, say $x_1, \ldots, x_n$, and the assumed relation between the independent variables and the dependent variable is

$$y = a_0 + a_1 x_1 + \cdots + a_n x_n.$$

- *Polynomial regression:* There is only one continuous independent variable x and the assumed model is

$$y = a_0 + a_1 x + \cdots + a_n x^n.$$

- *Logistic regression:* The dependent variable is binary, that is, a variable that takes only the values 0 and 1. The assumed model involves certain probability distributions.

9.7 Issues in Machine Learning

There are various common practical issues in machine learning. Some of them are listed below:

1. *Lack of Quality Data:* The absence of good data is one of the main issues in Machine Learning. Data quality is fundamental for the algorithms to work as proposed.

Incomplete data, unclean data, and noisy data are the typical rivals of an ideal machine learning. Different reasons for low data quality are-

- Data can be noisy which will result in inaccurate predictions. This often leads to less accuracy in classification and low-quality results. It is noted as one of the most common errors faced in terms of data.
- Incorrect or incomplete information can also lead to faulty programming through Machine Learning. Having less information will lead the program to analyze based on the minimal data present. Hence, decreasing the accuracy of the results.
- For better future actions, the generalization of input and output of past data is crucial. But a common issue that occurs is, the output data can become difficult to generalize.

2. *Understanding which processes need Automation:* It is difficult to choose facts from fiction with respect to machine learning. Before deciding on the AI platform to work upon, we need to thoroughly evaluate the types of problems to be solved. The processes that are carried out manually every day with no variable output, are the easiest processes to automate. The complicated processes need to be inspected before automation. Machine learning can help in the automation of some processes but not for all the processes.
3. *Inadequate Infrastructure:* Machine learning requires large amounts of data stirring abilities. The frameworks based on inheritance are not able to deal with the responsibility. One should check if the infrastructure can deal with the issues in machine learning. If it can't, then one should look for upgrading it completely with good hardware and flexible storage.
4. *Implementation:* Organizations generally have analytics engines before upgrading to machine learning. Integrating newer Machine Learning methodologies into existing methodologies is a complicated task. Maintaining proper interpretation and documentation goes a long way to easing implementation. Partnering with an implementation partner can make the implementation of services like anomaly detection, predictive analysis, and ensemble modeling much easier. Some issues in machine learning regarding implementation are-

- Slow deployment – Although the models of Machine Learning are time efficient but their creating process is quite the opposite. As it is still a young innovation the implementation time is slow.

- Data Security – Saving confidential data on ML servers is a risk as the model will not be able to differentiate between sensitive and insensitive data.
- Lack of data is another key issue faced during the implementation of the model. Without adequate data, it is not possible to give out valuable intel.

5. *Lack of Skilled Resources:* Machine learning and Deep analytics in their current forms are still new technologies. Thus, there is a shortage of skilled employees available to manage and develop analytical content for Machine Learning. Data scientists often need a combination of domain experience as well as in-depth knowledge of science, technology, and mathematics.

9.8 Summary

- Being an application of Artificial Intelligence (AI), Machine learning aims at computers to learn like humans. With the help of machine learning, computers can learn automatically without human interference or assistance.
- Either for humans or a machine, the learning process includes four components, data storage, abstraction, generalization, and evaluation.
- The machine learning system is based on a *model* and this model is created by providing *training*. The seven phases of a machine learning system include Data Collection, Data Preparation, Data Wrangling, Data Analyzation, Training the model, Testing the model, and Deployment.
- Machine learning has a vast range of applications in the real world. It can be used in speech recognition, e-commerce and entertainment companies, filtering spam emails and malware, in the field of finance, banks, etc.
- Association rule in machine learning is a method for analyzing the relations between the variables in large databases, using the concept of mutual interest.
- Classification, in machine learning, is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.
- In machine learning, a regression problem is a problem of predicting the value of a numeric variable based on the observed values of the variable.

9.9 Key Terms

- ***Machine learning Program:*** A computer program that learns from experience is called

a machine learning program or simply a learning program.

- ***Training:*** It is the process of fitting a model to a dataset.
- ***Dataset:*** The data obtained after collection is termed as a dataset.
- ***Data Mining:*** Applying machine learning to large databases is known as data mining.
- ***Discriminant:*** A discriminant of a classification problem is a rule or a function that is used to assign labels to new observations.

9.10 Check Your Progress

Short-Answer Type

Q1) A problem with more than two classes is often called a _____ classification problem.

Q2) Machine learning is applied for filtering spam emails and malware. (True/ False?)

Q3) Logistic regression algorithm is used for _____ problem in machine learning.

Q4) Applying machine learning to large databases is known as data mining. (True/ False?)

Long-Answer Type

Q1) What is Machine Learning? What is the importance of machine learning?

Q2) List at least five issues in machine learning.

Q3) Describe the phases of the machine learning lifecycle.

Q4) What are the basic components of the learning process?

Q5) List the applications of machine learning in detail.

References:

Machine Learning: a Probabilistic Perspective, Kevin Patrick Murphy, MIT Press, March 2014.

Introduction to Machine Learning, Ethem Alpaydin, The MIT Press, Cambridge, Massachusetts, 2004.

Unit 10 – Types of Machine Learning

10.0 Introduction

By now, we are aware of the basics, phases, and major issues in machine learning. Before gaining knowledge in depth, it is important to learn about the types of machine learning. We will discuss the categories of machine learning in brief, the details will be explained in the upcoming units.

Machine learning is mainly divided into two categories. The first type of machine learning, ***supervised*** learning (or ***predictive***) approach aims at mapping from inputs x to outputs y, given a labeled set of input-output pairs,

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N.$$

Here, $\mathcal{D}$ is known as the ***training set***, and N is the number of training examples. Each training input x_i is a D-dimensional vector of numbers, representing, say, the height and weight of a person. These are called features, attributes, or covariates. In general, however, x_i could be a complex structured object, such as an image, a sentence, an email message, a time series, a molecular shape, a graph, etc.

The second main type of machine learning is the ***unsupervised*** learning (or ***descriptive***) approach. Here we are only given inputs,

$$\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$$

The goal is to find "interesting patterns" in the data. This is sometimes called ***knowledge discovery.*** This is a much less well-defined problem, since we are not told what kinds of

patterns to look for, and there is no obvious error metric to use (unlike supervised learning, where we can compare our prediction of y for a given x to the observed value).

Apart from these two main categories, there is a third type of machine learning, known as ***reinforcement*** learning, which is less commonly used. This is useful for learning how to act or behave when given occasional reward or punishment signals.

10.1 Unit Objectives

After completing this unit, the reader will be able to:

- Gain knowledge about types of machine learning.
- Illustrate supervised, unsupervised, reinforcement, and semi-supervised learning.
- Discuss the relation of machine learning to other fields.

10.2 Supervised Learning

Supervised learning is the task of learning a function that maps an input to an output based on example input-output pairs. In supervised learning, each example in the training set is a pair consisting of an input object (typically a vector) and an output value. Figure 2.1 represents the basic workflow in supervised learning.

A supervised learning algorithm analyzes the training data and produces a function, which can be used for mapping new examples. In the optimal case, the function will correctly determine the class labels for unseen instances. Both ***classification*** and ***regression*** problems are supervised learning problems. A wide range of supervised learning algorithms is available, each with its strengths and weaknesses. There is no single learning algorithm that works best on all supervised learning problems.

A "supervised learning" is so-called because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers (that is, the correct outputs), the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

In simple terms, supervised learning is the type of machine learning that focuses on learning a classification or regression model, that is, learning from ***labeled training data*** (i.e., inputs that also contain the desired outputs or targets; basically, "examples" of what we want to predict). A training set of examples with the correct responses (targets) is provided and, based on this

training set, the algorithm generalizes to respond correctly to all possible inputs. This is also called learning from exemplars.

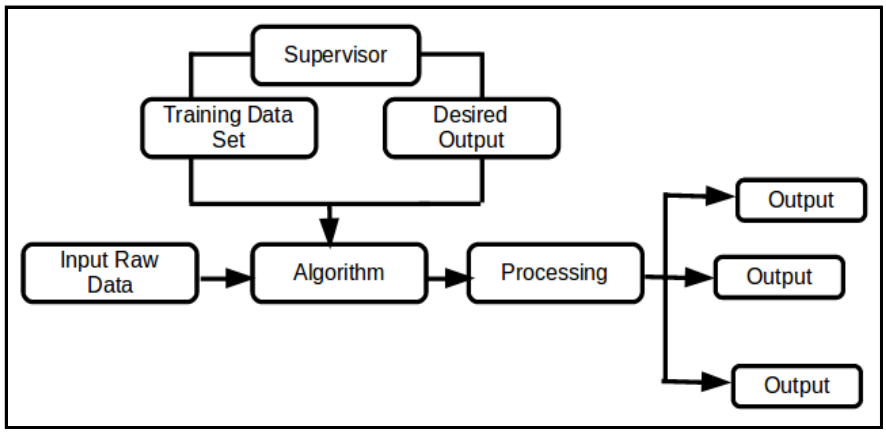


Figure 10.1 Supervised Learning

Classification problems categorize all the variables that form the output. For mapping from inputs x to outputs y, where $y \in \{1, \ldots, C\}$, with C being the number of classes.

- If C = 2, this is called ***binary classification*** (in which case we often assume $y \in \{0, 1\}$)
- If C > 2, this is called ***multiclass classification***.
- If the class labels are not mutually exclusive (e.g., somebody may be classified as tall and strong), it is called ***multi-label classification***.

Classification is probably the most widely used form of machine learning and has been used to solve many interesting and often difficult real-world problems. Let's discuss some of the examples here.

E-mail spam filtering and Document classification

The goal of *document classification* is to classify the given document, say, a web page or e-mail message, into one of the C classes. Consider e-mail spam filtering to be a special case of document classification, where the classes are spam (y = 1) or (y = 0). According to most of the classifiers, the input x has a fixed size. A transformation is defined as $x_{ij} = 1$ if word j occurs in document i. On applying this transformation to every document in our dataset, a binary document vs word co-occurrence matrix is obtained. For example, it should be noted that most spam messages have a high probability of containing words that are unwanted (like "cheap", "bye", etc.).

Face Detection and Recognition

Finding objects within an image is known as *object detection* or *object localization*. Face detection is a special case of object detection. One approach to solve this problem is to divide the image into small overlapping patches at different locations, scales, and orientations. Each patch is classified based on whether it contains a face-like texture or not. The system then returns the locations where the probability of face is adequately high. This is known as a ***sliding window detector***. Such face-detection systems are used by modern digital cameras.

Face detection can further help in the face recognition of any person. The identity of any person can be estimated using face detection. In the case of face recognition, the number of class labels C might be very large. [4]

Regression is similar to classification except the response variable in regression is continuous. Regression aims at predicting the value of a numeric variable based on the observed values of the variable. Some examples of real-world regression problems are predicting tomorrow's stock market price given current market conditions and other possible side information; Predicting the temperature at any location inside a building using weather data, time, door sensors, etc.

10.3 Unsupervised Learning

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data ***without labeled responses***. In unsupervised learning algorithms, a classification or categorization is not included in the observations. There are no output values and so there is no estimation of functions. Since the examples given to the learner are unlabeled, the accuracy of the structure that is output by the algorithm cannot be evaluated. The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data.

Unsupervised learning does not provide correct responses, but instead the algorithm tries to identify similarities between the inputs so that inputs that have something in common are categorised together. The statistical approach to unsupervised learning is known as ***density estimation.*** Figure 10.2 represents the basic workflow of unsupervised learning.

There are two major differences between supervised and unsupervised learning.

1. Supervised learning is conditional density estimation, whereas unsupervised learning is

4 *Szeliski, R. (2010), Computer Vision: Algorithms and Applications, Springer.*

unconditional density estimation. For supervised learning, the models are build of the form, $p(x_i|\ \theta)$, while for unsupervised learning, the models are build of the form, $p(y_i|\ x_i,\theta)$.

2. x_i is a vector of features, so we need to create multivariate probability models. By contrast, in supervised learning, y_i is usually just a single variable that we are trying to predict. This means that for most supervised learning problems, we can use univariate probability models (with input-dependent parameters), which significantly simplifies the problem.

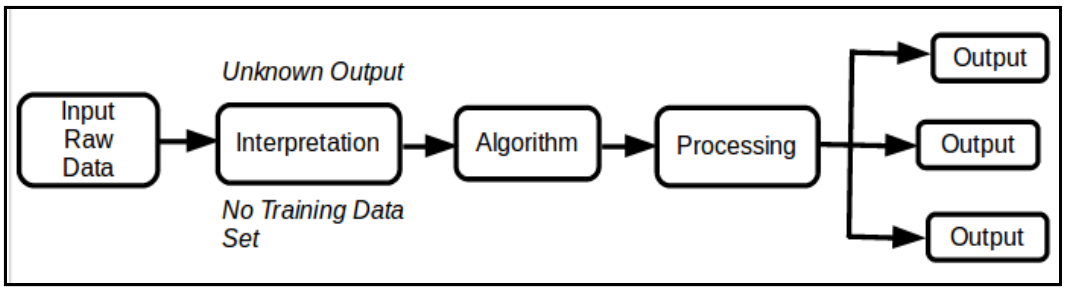


Figure 10.2 Unsupervised Learning

Unsupervised learning is more applicable than supervised learning, since it does not require a human expert (supervisor) to manually label the data. Labeled data is not only expensive, it also contains little information. This information is certainly not enough to reliably predict the parameters of complex models.

Depending on the problem, the unsupervised learning model can organize the data in the following ways:

- ***Clustering:*** The training data that is similar to each other are selected and grouped together.
- ***Anomaly Detection:*** Unsupervised learning can be used to flag outliers in a particular dataset. For example, banks can detect the fraud transactions on the basis of unusual purchasing behavior of a customer.
- ***Association:*** Certain features of a data sample correlate with other features. By analyzing some key attributes of a data, the other attributes can be predicted using unsupervised learning.
- ***Autoencoders:*** Autoencoders take input data, compress it into a code, then try to

recreate the input data from that summarized code.

10.4 Reinforcement Learning

Reinforcement learning is the third type of machine learning, which is useful for learning how to act or behave when given occasional reward or punishment signals. It is somewhat less commonly used. Reinforcement learning is the problem of getting an agent to act in the world so as to maximize its rewards. A learner (the program) is not told what actions to take as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situations and, through that, all subsequent rewards. Figure 10.3 represents the basic workflow of reinforcement learning.

Reinforcement is the process of learning from rewards while performing a series of actions. In reinforcement learning, we do not tell the learner or agent, for example, a (ro)bot, which action to take but merely assign a reward to each action and/or the overall outcome. Instead of having a "correct/false" label for each step, the learner must discover or learn a behavior that maximizes the reward for a series of actions. In that sense, it is not a supervised setting and somewhat related to unsupervised learning; however, reinforcement learning really is its own category of machine learning. Reinforcement learning will not be covered further in this class.

For example, consider teaching a dog a new trick: we cannot tell it what to do, but we can reward/punish it if it does the right/ wrong thing. It has to find out what it did that made it get the reward/ punishment. We can use a similar method to train computers to do many tasks, such as playing chess, scheduling jobs, and controlling robot limbs.

Reinforcement learning is different from supervised learning. Supervised learning is learning from examples provided by a knowledgeable expert.

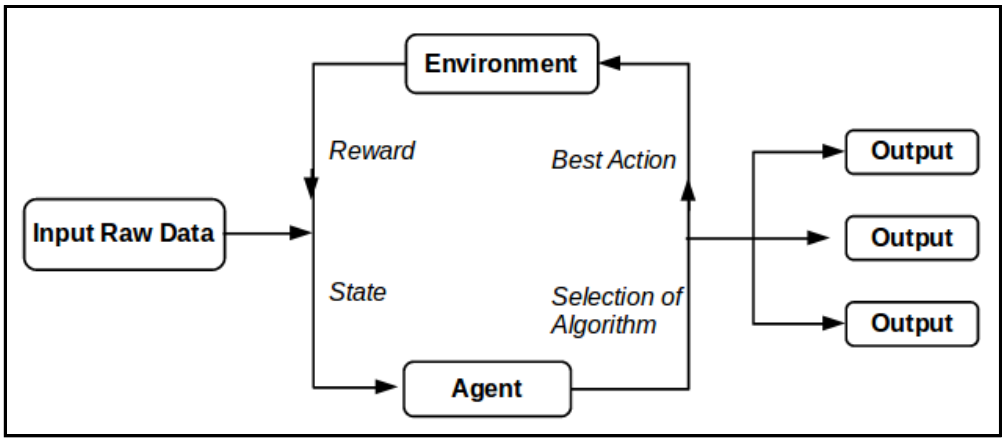


Figure 10.3 Reinforcement Learning

Typical applications of reinforcement learning involve playing games (chess, Go, Atari video games) and some form of robots, e.g., drones, warehouse robots, and more recently self- driving cars.

10.5 Semi-supervised learning

Simply, semi-supervised learning can be defined as a mix of supervised and unsupervised learning. We already know that in supervised learning tasks, some training examples contain outputs, but some do not. Then, a labeled training subset is used to label the unlabeled portion of the training set, which we then also utilize for model training.

Semi-supervised learning contains the training dataset with both labeled and unlabeled data. This method is particularly useful when extracting relevant features from the data is difficult, and labeling examples is a time-intensive task for experts.

Common situations for this kind of learning are medical images like CT scans or MRIs. A trained radiologist can go through and label a small subset of scans for tumors or diseases. It would be too time-intensive and costly to manually label all the scans — but the deep learning network can still benefit from the small proportion of labeled data and improve its accuracy compared to a fully unsupervised model.

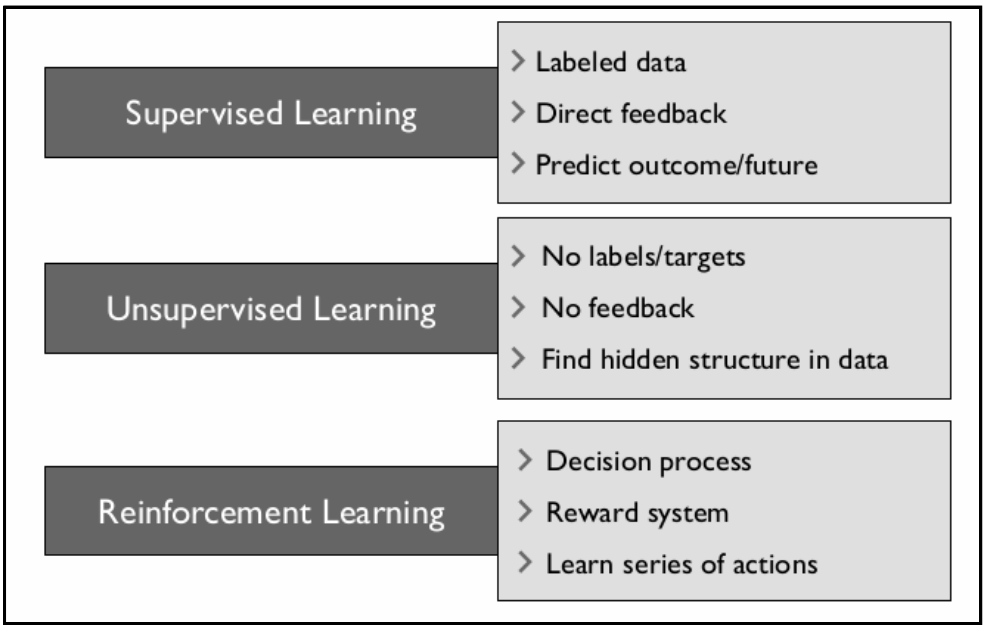


Figure10.4 Overview of the categories of Machine Learning

(**Source-** Raschka and Mirjalili: Python Machine Learning, 2nd Ed.)

10.6 Relation to other fields

Machine learning being an interdisciplinary field, shares common threads with the field of information theory, mathematical fields of statistics, game theory, and optimization. Naturally, machine learning is a subfield of computer science and a branch of AI (Artificial Intelligence). However, it should be noted that, in contrast with traditional AI, machine learning is not trying to build automated imitation of intelligent behavior, but rather to use the strengths and special abilities of computers to complement human intelligence, often performing tasks that fall way beyond human capabilities. For example, the ability to scan and process huge databases allows machine learning programs to detect patterns that are outside the scope of human perception.

The component of experience, or training, in machine learning often refers to data that is randomly generated. The task of the learner is to process such randomly generated examples toward drawing conclusions that hold for the environment from which these examples are picked.

Machine learning and Data Mining

Data mining focuses on the discovery of patterns in datasets or "gaining knowledge and insights" from data – often, this involves a heavy focus on computational techniques, working with databases, etc. Nowadays, the term data mining is more or less synonymous to "data science". We can then think of machine learning algorithms as tools within a data mining project. Data mining is not "just" but also emphasizes data processing, visualization, and tasks that are traditionally not categorized as "machine learning" (for example, association rule mining).

Machine learning, AI, and deep learning

Artificial intelligence (AI) is a subfield of computer science focussing on solving tasks that humans are good at (for example, natural language processing, image recognition, etc.). There are two subtypes of AI: *Artificial general intelligence (AGI)* and *narrow AI*. AGI refers to an intelligence that equals humans in several tasks, i.e., multi-purpose AI. In contrast, narrow AI is more narrowly focused on solving a particular task that humans are traditionally good at

(e.g., playing a game, or driving a car, etc.). The field of machine learning has emerged as a subfield of AI. It was concerned with the development of algorithms so that computers can automatically learn (predictive) models from data.

Consider an example in which we want to develop a program that can recognize handwritten digits from images. One approach would be to look at all of these images and come up with a set of (nested) if-this-then-that rules to determine which digit is displayed in a particular image. Another approach would be to use a machine learning algorithm, which can fit a predictive model based on a thousands of labeled image samples that we may have collected in a database.

Deep learning is a subfield of machine learning, referring to a particular subset of models that are particularly good at certain tasks such as image recognition and natural language processing. Deep learning imitates the way humans gain certain types of knowledge. It is an important element of data science, which includes statistics and predictive modeling. At its simplest, deep learning can be thought of as a way to automate predictive analytics. While traditional machine learning algorithms are linear, deep learning algorithms are stacked in a hierarchy of increasing complexity and abstraction.

To conclude, deep learning and machine learning definitely helps to develop AI, however, AI doesn't necessarily have to be developed using machine learning. Figure 2.5 depicts the relationship between machine learning, deep learning, and artificial intelligence.

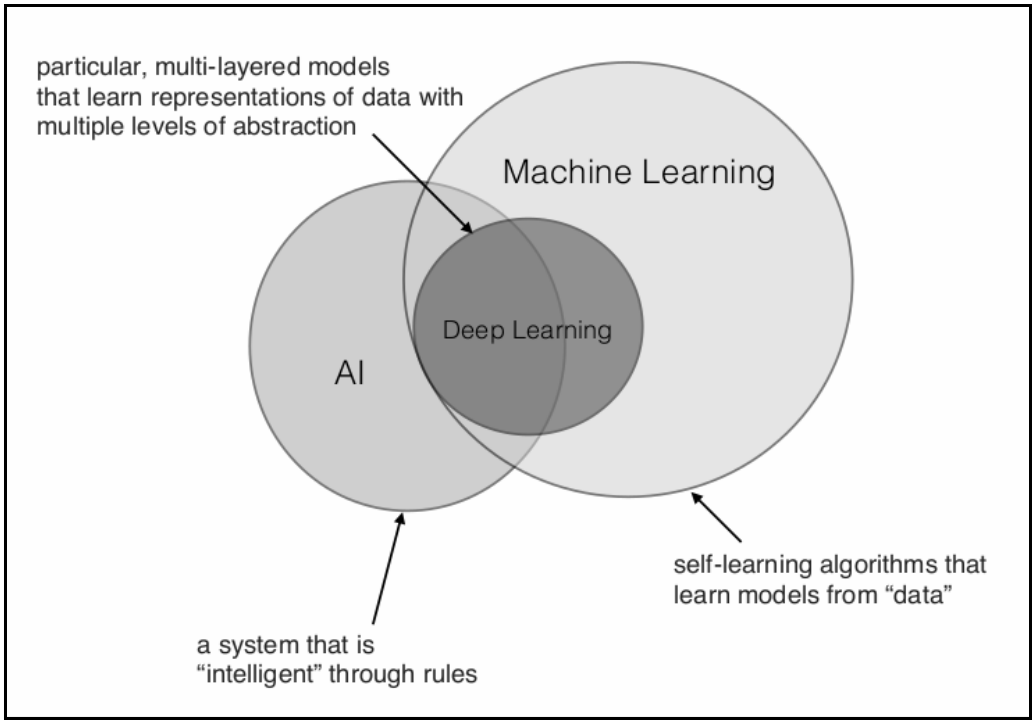


Figure 10.5 Relationship between machine learning, deep learning, and artificial intelligence

Machine learning and Statistics

Machine learning serves a close relationship with statistics. Indeed there is a lot in common between the two disciplines, in terms of both the goals and techniques used. There are, however, a few significant differences of emphasis; if a doctor comes up with the hypothesis that there is a correlation between smoking and heart disease, it is the statistician's role to view samples of patients and check the validity of that hypothesis (this is the common statistical task of hypothesis testing).

In contrast, machine learning aims to use the data gathered from samples of patients to come up with a description of the causes of heart disease. The hope is that automated techniques may be able to figure out meaningful patterns (or hypotheses) that may have been missed by the human observer. In contrast with traditional statistics, in machine learning in general, algorithmic considerations play a major role. Machine learning is about the execution of learning by computers; hence algorithmic issues are pivotal. We develop algorithms to perform the learning tasks and are concerned with their computational efficiency. Another difference is that while statistics is often interested in asymptotic behavior (like the convergence of sample-

based statistical estimates as the sample sizes grow to infinity), the theory of machine learning focuses on finite sample bounds. Namely, given the size of available samples, machine learning theory aims to figure out the degree of accuracy that a learner can expect on the basis of such samples.

10.7 Summary

- Machine learning is mainly divided into two categories: supervised and unsupervised learning.
- Supervised learning is the task of learning a function that maps an input to an output based on example input-output pairs.
- Supervised learning focuses on learning a classification or regression model, that is, learning from ***labeled training data.***
- Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data ***without labeled responses***.
- Unsupervised learning is more applicable than supervised learning, since it does not require a human expert (supervisor) to manually label the data.
- Reinforcement learning is the third type of machine learning, which is useful for learning how to act or behave when given occasional reward or punishment signals. It is somewhat less commonly used.
- Semi-supervised learning contains the training dataset with both labeled and unlabeled data. This method is particularly useful when extracting relevant features from the data is difficult, and labeling examples is a time-intensive task for experts.
- Machine learning being an interdisciplinary field, shares common threads with the field of artificial intelligence, statistics, deep learning, data mining, etc.

10.8 Key Terms

- ***Training example:*** A row in the table representing the dataset. Synonymous to an observation, training record, training instance, training sample (in some contexts, sample refers to a collection of training examples).
- ***Sliding Window:*** Sliding windows play an integral role in object classification, as they allow us to localize exactly "where" in an image an object resides.
- ***Deep learning:*** It is a subfield of machine learning, referring to a particular subset of

models that are particularly good at certain tasks.

- ***Clustering:*** The training data that is similar to each other are selected and grouped together.
- ***Autoencoders:*** Autoencoders take input data, compress it into a code, then try to recreate the input data from that summarized code.

10.9 Check Your Progress

Short-Answer Type

Q1) Supervised learning is based on _____ data.

Q2) Reinforcement learning is the most widely used category of machine learning. (True/ False?)

Q3) Full form of AGI-

a) Artificial general interpretation
b) Artificial general intelligence
c) Artificial gain interpretation
d) Artificial gain intelligence

Q4) The statistical approach to _____ learning is known as density estimation.

Long-Answer Type

Q1) Explain supervised learning with an example.

Q2) Distinguish between classification and regression.

Q3) What are the differences between supervised and unsupervised learning?

Q4) Write a short note on the relation between machine learning and artificial intelligence.

Q5) List the two major categories of machine learning. Give examples.

References:

Machine Learning: a Probabilistic Perspective, Kevin Patrick Murphy, MIT Press, March 2014.

Introduction to Machine Learning, Ethem Alpaydin, The MIT Press, Cambridge, Massachusetts, 2004.