ਜਗਤ ਗੁਰੂ ਨਾਨਕ ਦੇਵ
ਪੰਜਾਬ ਸਟੇਟ ਓਪਨ ਯੂਨੀਵਰਸਿਟੀ
ਪਟਿਆਲਾ

JAGAT GURU NANAK DEV
PUNJAB STATE OPEN UNIVERSITY, PATIALA

(Established by Act No. 19 of 2019 of the Legislature of State of Punjab)

The Motto of the University
(SEWA)

SKILL ENHANCEMENT **EMPLOYABILITY** **WISDOM**
ACCESSIBILITY



DIPLOMA IN SOFTWARE DEVELOPMENT AND PROGRAMMING
SEMESTER-II
Course: DBMS
Course Code: DBMS-2-01T

ADDRESS: C/28, THE LOWER MALL, PATIALA-147001
WEBSITE: www.psou.ac.in

SELF-INSTRUCTIONAL STUDY MATERIAL FOR JGND PSOU, ALL COPYRIGHTS WITH JGND PSOU, PATIALA

DBMS-2-01T: Data Base Management System (DBMS)

Total Marks: 100
External Marks: 70
Internal Marks: 30
Credits: 6
Pass Percentage: 40%

Course: Data Base Management System (DBMS)	
Course Code: DBMS-2-01T	
Course Outcomes (COs) After the completion of this course, the students will be able to:	
CO1	Understand the fundamental elements of database management system.
CO2	Understands the three level architecture of DBMS and mapping between these levels.
CO3	Familiar with the hierarchical model, network model, entity relationship model and relational model.
CO4	Acquire knowledge of normalization technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies.
CO5	Apply SQL to solve problems

Detailed Contents:

Module No.	**Module Name**	**Module Contents**
Module 1	**Introduction of DBMS**	Database Approach, Characteristics of a Database Approach, Database System Environment. Roles in Database Environment: Database Administrators, Database Designers, End Users, Application Developers. Database Management Systems: Definition, Characteristics, Advantages of Using DBMS Approach, Classification of DBMSs. Three Level Architecture of DBMS: Database Schema and Database Instance, Mapping Between Different Views, Data Independence–Physical and Logical Data Independence, Difference between logical data independence and physical data independence, Components of a DBMS, Data Dictionary, DBMS Languages.
Module II	**Data Models**	Classification of Data Model, Hierarchical Model, Network Model, Entity Relationship Model, Database Conceptual Modeling by E-R model: Concepts, Entities and Entity Sets, Attributes, Mapping Constraints, E-R Diagram, Weak Entity Sets, Strong Entity Sets, Comparison between Data Models. Relational Data Model: Concepts and

		Terminology. Constraints: Integrity Constraints, Entity and Referential Integrity constraints, Keys,
Module III	**Relational Algebra & Relational Calculus**	**Relational Algebra:** Basic Operators, Additional Operators. **Relational Calculus:** Tuple Relational Calculus and Domain Relational Calculus, Difference between relational algebra and relational calculus.
Module IV	**Normalization**	Functional Dependency, Full Functional Dependency, Partial Dependency, Transitive Dependency, Normal Forms– 1NF, 2NF, 3NF, BCNF, Multi-valued Dependency, Join Dependency and Higher Normal Forms-4NF, 5NF.
Module V	**Transaction Management & Concurrency Control**	Transaction Management and Concurrency Control: ACID Properties. Database Protection: Security Issues, Discretionary Access Control-Granting and Revoking Privileges. Database Concurrency: Problems of Concurrent Databases, Serializability and Recoverability, Concurrency Control Methods-Two Phase Locking, Time Stamping. Deadlock, Database security and integrity, Different Methods of Database Security, Database Recovery: Recovery Concepts, Recovery Techniques-Deferred Update, Immediate Update, Shadow Paging.
Module VI	**SQL**	Introduction to SQL*PLUS, Data types, Parts of SQL: Data Definition Language, Data Manipulation Language, Data Control Language, and Transaction Control Language. SQL Operators, SQL Functions, Joins, Roll up operation, Cube operation, Nested query, Subquery, View, Disadvantages of SQL.

Books

1. Elmasry Navathe, “Fundamentals of Database System”, Pearson Education.
2. James Groff, Paul Weinberg, Andy Oppel, “Oracle SQL Complete Reference”, Tata McGraw-Hill.
3. T.Connolly, C Begg,“Database Systems”, Pearson Education.
4. Jeffrey D. Ullman, “Principles of Database Systems”, Galgotia Publications.
5. Henry F. Korth, A. Silberschhatz, “Database Concepts”, Tata McGraw Hill.
6. C. J. Date, "An Introduction to Database Systems”, Pearson Education

COURSE: DBMS

UNIT 1: INTRODUCTION OF DATABASE MANAGEMENT SYSTEM

1. INTRODUCTION

1.1 DATABASE CONCEPTS

1.2 TRADITIONAL FILE MANAGEMENT SYSTEM

1.3 DATABASE

1.4 DATABASE MANAGEMENT SYSTEM (DBMS)

1.4.1 CHARACTERISTICS OF DATABASE MANAGEMENT SYSTEM

1.4.2 OPERATIONS/FUNCTIONS OF DATABASE MANAGEMENT SYSTEM

1.4.3 ADVANTAGES OF DATABASE MANAGEMENT SYSTEM

1.4.4 DISADVANTAGES OF DATABASE MANAGEMENT SYSTEM

1.5 COMPONENTS OF DATABASE SYSTEM

1.6 DBA (DATABASE ADMINISTRATOR)

1.7 COMPARISON OF FILE MANAGEMENT SYSTEM WITH DATABASE MANAGEMENT SYSTEM

1.8 CATEGORIES OF DBMS

1.8.1 CENTRALIZED DBMS

1.8.2 PARALLEL DBMS

1.8.3 DISTRIBUTED DBMS

1.8.4 CLIENT/SERVER DATABASE SYSTEM

1. INTRODUCTION

The exponential growth of information technology and its dependency in different sectors of society results in collection of hugs data. The large data collection has to be stored in such a way that is should be retrieved and processed as per the requirement of the user. Traditionally, data was manually maintained, stored in fields, updated and retried manually. The system was worked with very small amount of data which was isolate handled by single user. With the increase in size of data and access of multiple

users for single source of data, manually management of such data was nearly impossible in such a scenario, the concept of database management system was originated. The goal of the database management system is to store information in such a way so that it can be access with ease. The database management system is aimed to perform basic operation like: storing, retrieval, sorting, searching, and deletion of records in database. It plays a critical role in almost all area where computer systems are used for information processing like business, engineering, medical, defence, education, library etc. The database sheared among different users. Sometime, it is called mediator between user and data as it communicate between user and data. It responses to the user with results after processing query raised by the user. Before going into more technical detail of DBMS, let us ho through the basic concepts:

1.1 DATABASE CONCEPTS

The conceptual understating of database is required to go through two elementary database concepts: Data and Information

I. ***Data: It is defined as representation of facts, concepts, and instruction in a from which is suitable for communication, interpretation by human or computer.***

- Data can be recorded and have meaning.
- Examples of data are: height, weights, prices, costs, names of things, marks, image, and sound.
- In a formal manner, data is suitable for understanding and processing's.

Data can be represented with different character set and format which are stipulated in table given below:

Representation of Data		
Sr. No.	**Format**	**Character set**
1	Alphabet	(A-Z a-z)
2	Digits	(0-9)
3	Special characters	(+,-,*,/,@,#)

4	picture	Picture in jpeg, Gif, Tiff, etc format
5	Sound	Sound in mp3, mp4 format

Table 1.1 Representations of Data (Format Character Set)

In a any information processing system like database management system, data us considered as raw material or figure as it itself is not significant. It requires to be processed to come up with suitable fact and figure which is called as information.

II. Information: It is defined as processed from of data which has significance in decision making or performing some action. In another words, information is data that has been converted into some useful form.

The information is a result of processing of data according to specific requirement. In hugs data collection user is asked to make query to fetch required information. The following logical diagram demonstrated the same concept:



Fig. 1.1: Data Processing for Information

It is very important to throw light on the basic difference between these two elements: Data and information, as these terms are quite miss used among beginners. They sometime us information on the place of data but both information and data are different from each other. The tabular representations of fact as shown below very well explained the differences.

Sr. No	**Data**	**Information**
1	Data is raw facts and unorganized figure that need to be processed.	Information is processed from of data which may be further processed to gain knowledge.
2	Data is useless until it is organized and does not convey any message.	Information is useful and conveys meaningful message.

3	Data is used as input for any data processing application.	Information is the result of any data processing application.
4	Decision making is not recommended on data as data may or may not be meaningful .That is why: data not help in decision making.	Information is always meaningful and plays vital role in any decision making process.
5	Data is available in unorganised and un specific format.	Information is always required in organised and in specific format.
6	Data is a collection of atomic levels of pieces. It collectively represents different fact and figure.	Information is organized collection of data and is always represent about specific entity.
7	Data itself has no significance in business as data is not in the form that can be interrelated.	Information is interrelated to data and has strong significance in business.
8	Data representation order is not significant as it may be in any order. It has no effect of meaning	Information must be in specific order otherwise It may have different meaning.
9	The data cannot be interpreted as it is very difficult to understand. It may have different meaning for different person in different situation.	Information is concrete in nature and easy to Understand. It has same meaning for everyone in any situation.
10	Example: Data may have figure like 20, 30, 50, 70 that it is a raw figure which has no significance.	Example: Data is proceed and associate with some meaningful facts like 20 year old,30 kg weight,50 gm Gold, 70 km/hr, etc May be processed from of data. Now

		these figures have some meaning.

Table 1.2: Different between Data and Information

Importance of information in organization: the organisation has maintained data of activities conducted during the session which include sale purchase data, human resource data, store data, etc as shown below in logical diagram.

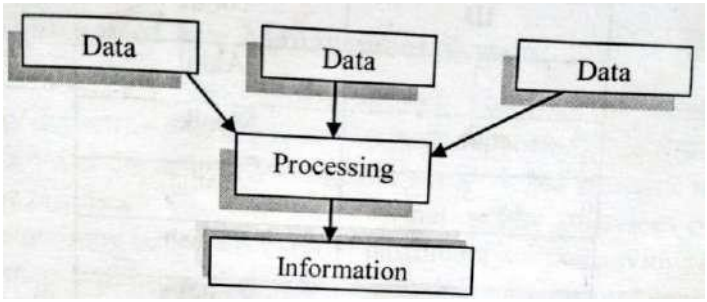


Fig. 1.2: Organisational Data Processing Conceptual Model

Every organisation has data processing systems that applied on different data to fetch information for smooth working of their organization. The information may be significant to the organisation for many purpose some of these are listed below which emphasis that information is very important for smooth working of any organisation.

Based on the above points, we conclude that the information help in planning, the action in the process of running and protecting the system.

- To gain information about the organisational sale and purchase.
- To access information of employee in the organisation.
- To know about the future predictions of the organisation.
- To know about the surroundings and whatever is happening in the society and universe.
- To keep the system up to data.

1.2 TRADITIONAL FILE MANAGEMENT SYSTEM

The file management system is a traditional approach to store and mange data in files. it is early day approach when records are stored in different files with different format. Each department in organisation have own file storage system where specific

application are designed to process these applications. The department have their own set rule to store and retrieve data from file system. The system to handle these file was called file management system. Such system are file department and incompatible to other file system. It means file system of one department may not work for file processing of another department. But such system is preferably good as compare to manual file management.

Key points:

- File processing system is a simple computer file system.
- It is a group of files storing data of an organization.
- File are in the form of text. Even records are also in text form.
- Each file is independent from one another.
- Each file is called a *falt file.*
- It uses hard disk or CD to store the data.
- File are designed by using programs written in programs written in programming languages such as C, C++.
- Searching is very difficult. Searching will start and continue till find the result.
- If data is very large then searching will take long time.

ID	Name
1	Akhil
2	Monika
3	Aastha
4	Ankush
5	Radhika

Formula for searching = n+1/2

- File are suitable when number of store items is small.
- It is not suitable when we have to perform data processing.
- As a system became more complex, file processing system presented many limitations and were difficult to maintain.

Example : To understand file management system in details we are taking practical example. Let us consider a business organisation where different department are organised and performed different tasks. Suppose department are Sale and Purchase Department, Inventory, Control Department, HR Department and Production unit. Every department have own system to store information in files. The following diagram shows how file management system works and manages data is different files.

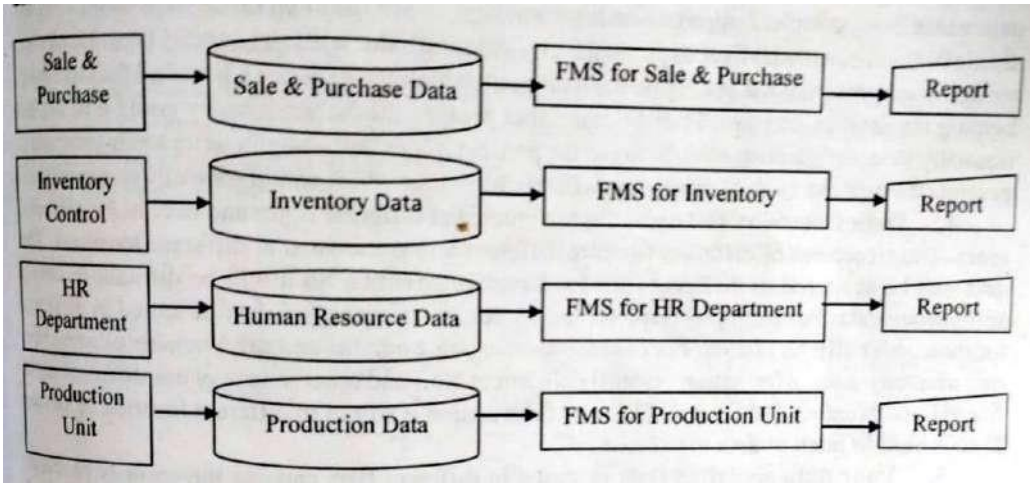


Fig. 1.3: File Management system in a business Organisation

It is very much clear form the above diagram that in file management system every department has separate storage of data and specific application for processing. In such as system inter-departmental access is not possible and there is duplication of organisational data. So the file management system has many limitations which are addressed in new concept, called, database management system.

Limitation/ Disadvantages of file Management System:

1. **Duplication of Data (Data Redundancy):** The file are created according to the application and every department in organistion have separate file system. So in that case the repetition of information about an entity cannot be avoided. For instance in Bank, the files are maintained about the customer. The personal information like addresses of customer holdings savings account and also the address od the customers will be present in file maintaining the current account. Even in case if same customers

have a saving account and current account his address will be present at two places. There is duplication of data as files are not shear able among different applications.

2. **Data Inconsistency and Inflexibility:** Data isolation limited the flexibility of file processing system in providing users with ad-hoc information requests. Data inconsistency means data about same entity stored in different files are not up-to-date and is not identical at same time. It is due to duplication of data which leads to greater problem than just wasting the storage. Same data which has been repeated at several places may not match after it has been updated at some places. For example: Suppose the customer requests to change the address for his account in the Bank and the Program is executed to update the saving bank account file only but his current bank account file is not updated. Now the addresses of the same customer have two addresses stored in two different locations that are called data inconsistency.

3. **Difficulty in Accessing Data:** In file management system, the program is designed for generating ad hoc reports. It means that program is for not general purpose and is data dependent. For example: Suppose administrator want to see list of all the customers holding the saving banks account who lives in particular city. Administrator will not have any program already written to generate that list but say he has a program which can generate a list of all the customers holding the savings account. Then he can either provide the information by going thru the list manually to select the customers living in the particular locality or he can write a new program to generate the new list. Both of these ways will take large time which would generally be impractical.

4. **Data Isolation:** The data files are created at different times and may be by different users. The structures of different files are different and are located at different locations. The data will be scattered in different files for a particular entity. So it will be difficult to obtain appropriate data. For example: Suppose the address of an employee may be stored in different location under different fields. For instance to store house number and street number of employee, one user may store information under (HNo, Street No.) and other one may use different name like (House Number and Street). This

way information is stored in different location of similar kind is hard to fetch as data is isolated.

5. **Poor data security:** Data is stored in different files causing the security problem. The data should be protected from unauthorized users. Every user should not be allowed to access every data.

6. **Difficult to Show Data According to User:** In file processing system, it was difficult to determine relationships between isolated data in order to meet user requirements.

7. **Concurrency Problems:** When more than one user are allowed to process the database. If in that environment two or more users try to update a shared data element at about the same time then it may result into inconsistent data. In case of file management system such concurrent access to data is hard to implement.

8. **Data in separated files:** Data is in more than one file and it is difficult to take data from more than one files.

9. **Data Dependence:** Data dependence means it is impossible to change storage structure without affecting the application program. If the format of a certain record was changes, the code in each file containing that format must be updated.

10. **Incompatible File Formats:** Each programmer stores the data in the file in the format as per the choice as there is no standard file format for storing the file. It becomes very difficult to handle the different files in different format.

The database management systems are designed to overcome above listed problems along with other advanced database concepts. The following section is designed to address issues like Database, DBMS, Difference between Database and DBMS, Characteristics of DBMS, Functions of DBMS, Advantages and Disadvantages of DBMS.

1.3 DATABASE

Database: It is a computer based record keeping system whose over all purpose is to record and maintains data. It is designed to hold bundle of organizational data. It holds the records, fields, and cells of data.

The database stores the known facts that can be recorded and that have implicit meaning. Data is represented in database in different levels of abstraction in its architecture.

Typically, there are three levels: External, Conceptual, and internal. The following diagram show how data is represented in different levels:

- **External Level:** It defines how user views data. Single user may have multiple views.
- **Conceptual Level:** It is a communication medium between external and internal level. Its representation is unique regardless of external level and internal level.
- **Internal Level:** It defines how data is physically stored.

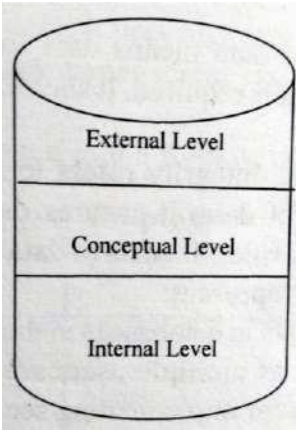


Fig. 1.4: Database Abstraction Levels

Key Points:

- The database is a shared collection of logically related data, designed to meet the information needs of an organization.
- It is a computer based record keeping system. Its overall purpose is to record and maintain the information.
- The database is a single large repository of data which can be used simultaneously by many departments and users.
- It holds not only the organisation's operational data but also a description of the data. It is also defined as a self-describing collection of integrated records.

- The description of the data is known as the Data Dictionary or Meta Data (the 'data about data").
- We can perform many operations on database such as:
    a) To add new operation.
    b) To modify/ edit the existing information.
    c) To remove/delete the unwanted information.
    d) To retrieve/ view the stored information.
    e) Arrange the information in a desired manner.
- Database is manages by an individual or group called Database Administrator (DBA), Who is responsible for designing, creating and maintaining the database to satisfy the needs of the users.
- All access to database is automated by special software called Database Management system (DBMS).
- The tern database is generally confused with DBMS. The database is a concept to representation data whereas DBMS an application program to provide access on database Both together represent Database System.

Characteristics of Database:

Database has some Characteristics in order to meet the standards which are as follows:

1. **Data sharing:** Database should ne capable to be shared among different users and applications.
2. **Persistence:** Persistence of data means data in a database exist permanently and available in time whenever it is required. It should live beyond the scope of the process that created it.
3. **Integrity/Correctness**: Data integrity refers to the property of data which enforce constrains to safe format of data. It ensures data should in a uniform format and implemented with integrity rules. It ensures data should be correct with respect to the real world entity that they represent.

4. **Security:** The security of data in database is in the top of priority. It should be protected from unauthorized access as multiple users are sharing database. Database should have their own mechanism for implementing security.
5. **Consistency:** The consistence of data is must whenever more than one data element in a database represents related real world values. The values should be consistent with respect to the relationship.
6. **Non-Redundancy**: The data in database should not be duplicated as no two data items in a database should represent the same real world entity. The non-redundancy helps to reduce size of the database and avoid inconsistency of data.
7. **Independence:** The database has three different levels (External, Conceptual, Internal) to represent data. These levels should be independent of each other so that the changes in one level should not affect the other levels.

1.1 DATABASE MANAGEMENT SYSTEM (DBMS)

DBMS: A database management system is a collection of interrelated data and a set of programs to access those data. The interrelated data is called database which is a shared collection of logically related data, designed to meet the information needs of an Organisation.

The primary goal of a DBMS is to provide methods to store and retrieve database information that is both convenient and efficient. Database sustems are designed to manage large bodies of information. In addition, the database system must ensure the safety of the information stored. Despite system crashes or attempts at unauthorized access. If data is to be shared among several users, the system must avoid possible anomalous results.

Key Points:

- DBMS is a software system that allows user to create , maintain and delete a database. It provides controlled access to the data.
- It centralzed the database.
- It is a computerized system which maintains the data.

- DBMS is an intermediate between programs and data. It is used to make information from data.
- DBMS is a collection of programs which are required to perform different task on database. It perform various operation on data like defining structure of data, accepting data, format data as per user requirement, hide data, allow concurrent access, backup and provide security to data.
- DBMS ensures the privacy of data. It prevent data from unauthorized users.
- Commercially available database management systems in the market are dbase, Foxpro, Oracle etc.
- In DBMS, data can be represented in the dorm of tables.

Employee

Ename	Empno	Job	Sal	Deptno
Nidhi	6258	Clerk	900	20
Aastha	6388	Manager	1500	30
Manmeet	6765	Clerk	1050	10
Navreet	6800	Analyst	1100	30

Entity : Employee

Attributes : Ename, Empno, Job, Sal, Deptno

Record : collection of related data i.e. Ename, Empno, Job, Sal, deptno

- In DBMS, each user can view data according to his/ her choice. Two users can use the same portion of data at the same time different forms.
- DBMS is used to create the reports and matahematical functions for the users.

1.1.1 Characteristics of Database Management system

A database management system is designed to define, manipulate, retrieve and manage data in a database. It generally manipulates the data itself, the data format, filed

names, record structure and file structure. It also defines rules to validate and manipulate this data. The modern DBMS has the following characteristics:

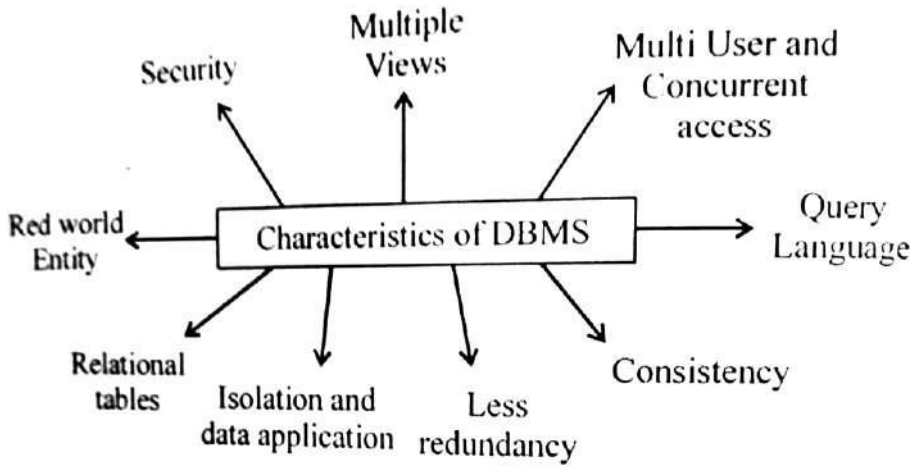
1. **Real world entity:** The DBMS is designed to represent real world entities consist of feature and behaviour of real world object. The DBMS have constructs that can easily define real world entity.

2. **Relational tables:** The database contains tables which are mapped with entities. These entity tables are related with other tables to define relational. This eases the concept of data saving. A user can understand the architecture of database just by looking at table names.

3. **Isolation and data application:** The DBMS is designed to isolate data form other complication working of the system as data in preserve into database. The application programs are written to access data

4. **Less redundancy :** DBMS Follows rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Following normalization, which itself is a mathematically rich and scientific process, make the entire database to contain as less redundancy as possible.

5. **Consistency:** DBMS always enjoy the state of consistency where the previous from of data storing application like file processing does not guarantee this. Consistency is a state where every relation database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state.



6. Query language: DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many any different filtering options, as her on she wants. Traditionally it was not possible where file-processing system was used.

7. Multiuser and concurrent access: DBMS support multi-user environment and allows them to access and manipulate data in parallel. Though there are restriction on transactions when they attempt to handle same data item, but users are always unaware of them.

8. Multiple views: DBMS offers multiples view for different users. A user who is in sales department will have a different view of database than a person working in production department. The enables user to have a concentrate view of database according to their requirements.

9. Security: Features like multiple views offers security at some extent where users are unable to access data of other user and department. DBMS offers methods to impose constraints while entering data into database and retrieving data at later stage. DBMS offers many different levels of security features, which enables multiple users to have different view with different features, for example, a user in sales department cannot see data of purchase department is one thing, additionally how much data of sales department he can see, can also be managed. Because DBMS offer many different features, for examples, a user in sales department cannot see data of purchase department

is one thing, additionally how much data of sales department he can see, can also be managed. Because DBMS is not saved on disk as traditional file system it is very hard for a theif to break the code.

1.1.2 Operations/Functions of Database Management System

A DBMS is an intermediate between user and database. The DBMS provides multiple useful interface to interact with database. There are several function that a DBMS performs to ensure data integrity and consistency of data in the database. The following are some important function of data management system.

1. **Data Dictionary Management :** Data Dictionary is where the DBMS stores definitions of the data elements and their relationships i.e. metadata. It is often hidden from the user and is used by Database Administrators and Programmers. It also shows which program use which piece of database and record.

The DBMS uses different function to look up the required data along with relationships into data Dictionary. Whenever a request is made for a particular data in database then DBMS programs access data dictionary. The function removes structural and data dependency and provides the user with data abstraction.

2. **Data Storage Management** : The data storage management is one of core function of DBMS which is used for the storage of data and any related data entry forms, report definitions, data validation rules, procedural code, screen definition and structures. The DBMS manage data in such a way that users do not need to know how data is stored or manipulated.

3. **Data Transformation and Presentation :** The data transformation and presentation is one of integrated function of DBMS as it is helpful to store data in simple format and display information in uniform format. The function exits to transform any data entered into required data structures. By using the data transformation and presentation function, the DBMS can determine the difference between logical and physical data formats.

4. Security Management: The security management is implemented in DBMS at different levels. Security management sets rules that determine specific user that are allowed to access the database. Users are given a username and password or sometimes through biometric authentication. DBMS must monitor user request. It can reject the request which break the security rules.

5. Data Definition: DBMS must be able to accept data definition commands. These commands are: create Alter and Drop.

6. Data Manipulation: DBMS must be able to handle the request from user to retrieve update and delete data. These commands are: Select, insert, Update and Delete.

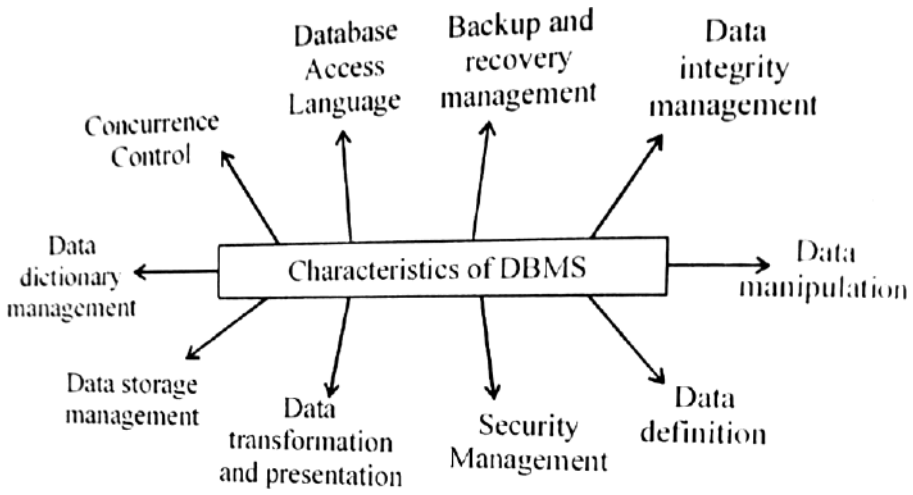


Fig. 1.6: Operations/Function of Database Management System

7. Data Integrity Management: Data integrity and data consistency are the core function of DBMS to provide security to data. The DBMS enforces these simultaneously without affecting the integrity of the database. The DBMS enforces these rules to reduce things such as data redundancy, which is when data is stored in more than one place unnecessarily, and maximizing data consistency, making sure database is returning correct/same answer each time for same question asked.

8. Backup and Recovery Management: Backup and recovery is done by DBMS to safeguard the old data so that unwanted damage to data can be recovered. Backup management refers to the data safety and integrity; for example backing up

document files. Similarly recovery of data can be implemented to go back to check the previous status of the data. DBMS software component ― transaction manager‖ is used to recover the data which was lost due to some misshapen.

9. Database Access Languages and User Interfaces : The DBMS provides multiple user interfaces to meet different requirements of the end user in different network environments DBMS may provides different terminals, web interfaces, etc. DBMS also provides user interface to interact with database. It is not feasible to provide everything in the form of drop down menu so DBMS supports a SQL (structured query language) language which is a non procedural language. The use of SQL language makes it easy for user to seek information according to the requirement. User can seek information by providing command to DBMS query processor which arranges data to the user.

10. Concurrency control : since DBMS support sharing of data among multiple users they must provide a mechanism for managing concurrent access to the database. DBMS ensure that the database kept in accurate state.

1.1.3 Advantages of Database Management System

1. Minimal Redundancy/Eliminate Duplication: In non-database system each application program has its own private files. In this case, the duplicated copies of the same data are created in many places. In DBMS, all data of an organization is integrated into a single database file. The data is recorded in only one place in the database and it is not duplicated. Centralized control of unnecessary duplication of data. It also reduce the total amount of data storage. It also eliminates the extra processing required to trace the results.

2. **Data Integrity:** In database management system, data in database is stored in tables. A single database contains multiple tables and relationships can be created between tables (or associated data entities). This makes easy to retrieve and update data. Data store in database is accurate and consistent. Integrity of data means that data in

database is always accurate, such that incorrect information cannot be stored in database. If the system "crashes", we can retrieve the data easily

3. **Improved Data Consistency:** By controlling the data redundancy, the data consistency is obtained. If a data item appears only once, any update to its value has to be performed only once and the updated value is immediately available to all users. If the DBMS has controlled redundancy, the database system enforces consistency.

4. **Data in Shared Form:** In DBMS, data can be shared by authorized users of the organization. The database administrator manages the data and gives rights to users to access the data. Many users can be authorized to access the same piece of information simultaneously. The remote users can also share same data. Similarly, the data of same database can be shared between different application programs. The DBMS allows the sharing of data under its control by any number of application, programs or users. There is no need to insert the data separately by each department or user. Data enter by one user can be share by all the users.

5. **Enforcement of Standards:** DBMS is enforced laws and policies in the form of standards which helps in maintaining database. Data is stored according to the standards and in uniform pattern The common standards can be implemented to all databases.

6. **Data Security:** Data must not be accessed by unauthorized persons with the help of DBMS we can ensure that proper access procedures can be implemented. Different level of security could be implemented for various types of data and operations.

7. **Solving Enterprise Requirement than Individual Requirement:** The DBMS is designed for general purpose and it is developed that person with different technical skills can use it as per the requirements. Since many types of users with varying level of technical knowledge use a database, a DBMS should provide a variety of user interface. The overall requirements of the enterprise are more important than the individual user requirements. So the DBA (Database Administrator) O can structure the database system to provide an overall service,

8. **Providing Backup and Recovery:** The DBMS provide the backup and recovery system so that data can be stored for future. It is also stored to manage the unwanted lose Similarly recovery policies are used which is automatically create the backup of data and restore data if required A DBMS must provide facilities for recovering from hardware or software failures.

9. **Cost of Developing and Maintaining System is Lower:** The cost of using DBMS is low as it requires basic resources which are generally available in the organisation. The cost involved in developing and maintaining the whole system is low whoever shitting manual data to electronic data may include labour cost which is addition to system cost.

10. **Concurrency Control:** The DBMS have control over the concurrent access to database It provides a common interface to perform access to database. The concurrent users may access data at same time being sharing by the DBMS. The DBMS designed polices such that one user access data other cannot perform updated task. At a time only one user is allowed to perform update operation other user need to wait for update the common data.

11. **Flexible System:** DBMS is a flexible system as it is designed for general purpose. The DBMS can handle small to large database and it can be redesigned to meet the requirements of the users. Moreover Database used by one system can be transformed into another system.

12. **Better Services to the User:** Because data are integrated into a single database, complex requests can be handled much more rapidly, then if the data were located in separate, non-integrated files. In many businesses, faster response means better customer service.

13. **Tools for Report Writing:** Most of the DBMS provide the report writer tools used to create reports. The users can create very easily and quickly. Once a report is created, it can be used may times and it can be modified very easily. The created reports are also saved along with database and behave like a software component.

14. **Controlled Concurrency:** In a computer file-based system, if two users are allowed to access data simultaneously, it is possible that they will interfere with each other. For example, if both users attempt to perform update operation on the same record, then one may overwrite the values recorded by the other. Most database management systems have sub-systems to control the concurrency so that transactions are always recorded with accuracy.

15. **Application Program/Data Independence:** The separation of data structure of database from the application program that uses the data is called data independence. In DBMS, we can easily change the structure of database without modifying the application program. Data is independent from one level to another level.

16. **Improved Decision Making Process:** Better-managed data and improved data access make it possible to generate better-quality information, on which better decisions are based. The quality of the information generated depends on the quality of the underlying data. Data quality is a comprehensive approach to promoting the accuracy, validity, and timeliness of the data. While the DBMS does not guarantee data quality, it provides a framework to facilitate data quality initiatives.

17. **Improvement in End-User Productivity:** The availability of data, combined with the tools that transform data into usable information, empowers end users to make quick, informed decisions that can make the difference between success and failure in the global economy.

18. **Application Development Ease:** The application programmer need not build the functions for handling issues like concurrent access, security, data integrity, etc. The programmer only needs to implement the application business rules. This brings in application development ease. Adding additional functional modules is also easier than in file-based systems.

19. **Data Atomicity:** A transaction in commercial databases is referred to as atomic unit of work. For example, when you purchase something from a point of sale (POS) terminal, a number of tasks are performed such as;

- Company stock is updated.

- Amount is added in company's account.
- Sales person's commission increases etc.

All these tasks collectively are called an atomic unit of work or transaction. These tasks must be completed in all, otherwise partially completed tasks are rolled back. Thus through DBMS, it is ensured that only consistent data exists within the database.

20. **No Data Isolation:** Data is stored in uniform format so there is no need to make different programs for each data.

21. **Advanced Capabilities:** DBMS also provides advance capabilities for online access and reporting of data through Internet. Today, most of the database systems are online. The database technology is used in conjunction with Internet technology to access data on the web servers

1.1.4 Disadvantages of Database Management System

In contrast to the lots of advantages, there are few disadvantages as well which are discussed below:

1. **Complexity:** The database designed is no of the major challenging task in DBMS as it is complex, difficult, and time-consuming. DBMS is an extremely complex piece of software database designer's developers DBA and end users must understand this functionality to take full advantage of it. Failure to understand the system can lead to bad design decisions.

2. **Large Size:** The complexity and breadth of functionality makes the DBMS an extremely large piece of software, occupying many megabytes of space and requiring huge amounts of memory to sun efficiently

3. **Performance:** File Based system is written for a specific application a result performance is generally very good. However, the DBMS is written to be more general to cater for many applications rather than just one. So performance is very poor.

4. **Higher Impact of Failure:** In most of the organizations, all data is integrated into a single database. If database is corrupted due to power failure or it is corrupted on the storage media, then our valuable data may be lost or whole system stops.

All users and applications rely on the availability of the DBMS the failure of any component can bring operations to a hall.

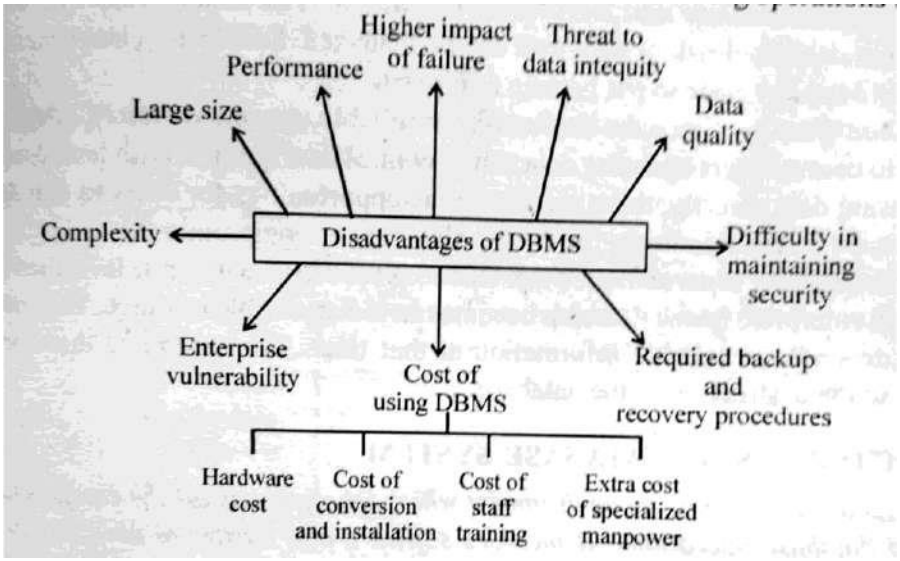


Fig. 1.7: Disadvantages of Database Management System

5. **Cost of using DBMS**

(a) **Hardware Cost:** If we want to implement DBMS then we need DBMS software which is very expensive. We need to upgrade the hardware the processing overheads to implement the security, integrity and sharing of data make the additional cost.

(b) **Cost of Conversion and Installation:** DBMS vendors frequently upgrade their products by adding new functionality. Such new features often come bundled in new upgrade versions of the software. Some of these versions require hardware upgrades. Not only do the upgrades themselves cost money, but it also costs money to train database users and administrators to properly use and manage the new features. Cost of DBMS and extra hardware may be insignificant compared with the cost of converting existing applications to run on the new DBMS and hardware. This cost includes cost of training staff to use these new systems and employment of specialist staff for help. That is way some organisations feel tied to their current systems and cannot switch to modern database technology.

(c) **Cost of Staff Training:** DBMS is a complex system which demands specialized users. The user need to get training which ultimately added to the total coast.

(d) **Extra Cost of Specialized Manpower:** The DBMS is managed by skilled people includes DBA, Programmer, and data entry staff.

6. **Threat to Data Integrity:** Database is shared among different users and concurrent access is permitted in DBMS. So there is always a threat to data integrity, especially when there is transition failure.

7. **Difficulty in Maintaining Security:** Data is reserved in common place and different users are accessing data with different security levels. The user management and security access is a challenging task which requires attention. Access policy design for secure access is difficult to maintain.

8. **Required Backup and Recovery Procedures:** The database need to be backed up in time so that unwanted risk of data lose can be managed. DBMS requires special extra hard disk space and special place to put backup of old data.

9. **Data Quality:** Since the database is accessible to users remotely, adequate controls are needed to control users updating data and to control data quality. With increased number of users accessing data directly, there are enormous opportunities for users to damage the data. Unless there are suitable controls, the data quality may be compromised.

10. **Enterprise Vulnerability:** When DBMS is used in an enterprise level then centralizing all data of an enterprise in one database becomes an indispensable resource. The survival of the enterprise depends on reliable information at that time. The enterprise therefore becomes vulnerable to the destruction of the database.

1.2 COMPONENTS OF DATABASE SYSTEM

The database system is an environment which incorporates all the components required to execute database operations. It includes software and hardware used in functioning of database.

The database system is designed to provide an environment that is both convenient and efficient to perform different task on data. There are five major components in the database system environment which are as follows:

1. ***Data (Data, Metadata)***
2. ***Software (DBMS and Application Programs)***
3. ***Hardware***
4. ***Users***
5. ***Procedure***

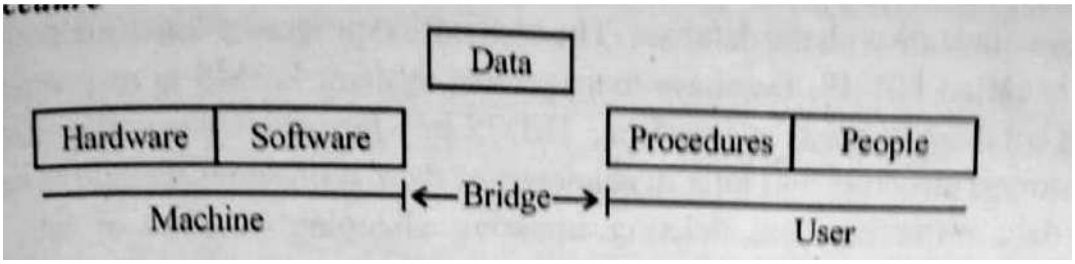


Fig. 1.8: Components of Database System

The following diagram shows these components and interaction with each other.

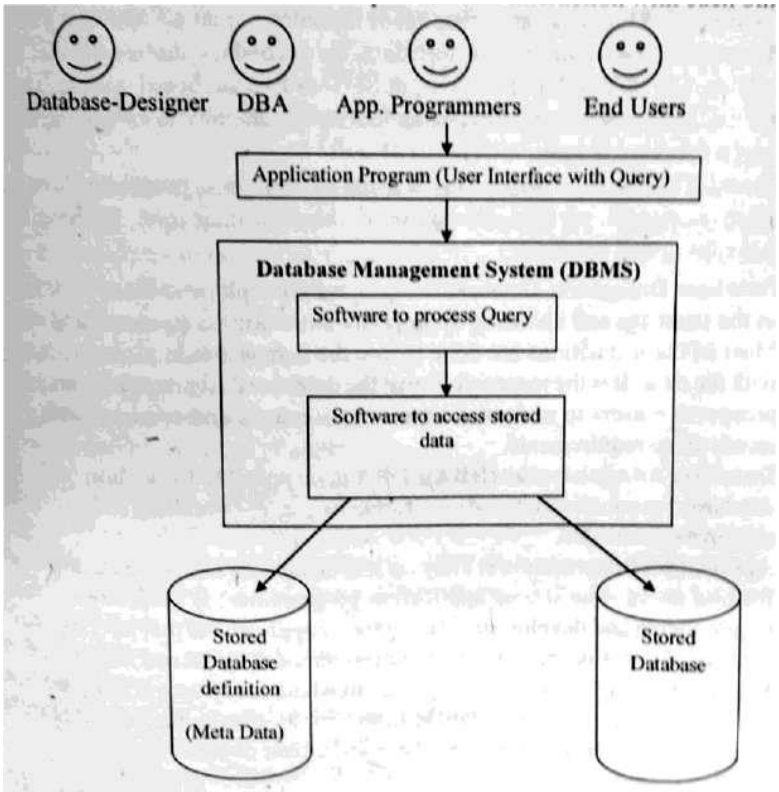


Fig. 1.9: Database System Environments

1. **Data (Data, Meta data):** The data is a fact and figure which is impotent to be stored in database. A database represents data and method used to preserve data in database. It designed to store data in such as way so that it can easily sharable and is being integrated. It also contains meta-data which is an information stored in catalo about the data. It is also called data about data or data dictionary which means it has information about data which stored in database. It defines the definition and representation information of data.

2. **Software (DBMS and Application Programs):** The software is actual DBMS. All requests from users for access to the database are handled by the DBMS. DBMS allows the users to communicate with the database. The application programs which are designed to handle database is called DBMS, Database management system. DBMS is responsible for smooth work with database and acts as interface. DBMS interface shields complex detail of data like physical storage structure and inter dependency of data. It provides method to handle data like insertion data, retrieving data, deleting, updating, changing structure of data, etc. It is also categorized into two components: Software to process Queries and Software to access stored data. The Software to process Queries deals with user interface and interaction to sort the query raised by the user whereas software to access data acts as an interface with physical database.

3. **Hardware:** The hardware component includes actual computer hardware used for keeping and accessing the database. To store data, the secondary storage devices are used such as Magnetic Disk, CD/DVD. Input and output devices like keyboard, mouse, scanner reader, monitors are used. Data processing hardware includes computer processor which plays significant role to support a database system.

4. **Users:** The users are person who is using the system as per their role or requirement. In typical database system, we categorized user on basis of their role. We have four types of users which are discussed below:

- **Database Designers:** Database designers are people who identify data to be stored in the database and choosing appropriate structures to represent and store the data Most of these functions are done before the database is implemented and populated with the data. It is the responsibility of the database designers to communicate with all prospective users to understand their requirements and come up with a design that meets these requirements.
- **Database Administrator (DBA):** DBA is responsible for authorizing access to the database, for coordinating and monitoring its use, and acquiring software and hardware resources as needed.
- **Application Programmers:** They are responsible for developing application program for end users. The job of application programmers is to determine the end use requirements and develop specifications for applications that meet the requirements. They implement the specifications as programs, then test and debug the programs.
- **End Users:** End users are those people to whom the system is designed. These use are actually accessing the database from their terminals. The end users are classified on the basis of knowledge of database and extent of use.

User	Role of User
Naive users	Naive users access the database through application programs that have been written by application programmers. These users don't have technical details of the database and its structures. The use system manual for accessing database. The example s of Naive users are: Operator in hotel for reservation, railway or airline clear, etc.
Casual User	Casual users are occasionally access database and use online query to fetch data from database. They have knowledge of query language and use it from their terminal to fetch data.
Sophisticated Users	These users interact with database without writing any program, use stand query to seek information in database. Examples of engineers, scientists, analysts who implement applications to meet their requirements.
Standard User	They interact with system with the help of menu driven interface. They don't have technical information of the database and use of query is minimum.
Specialized Users	Specialised user is system expert and hardcode professionals. They develop their own application program for system like expert system, knowledgebase system.

Table 1.3: Different End Users and their Rolls

The database system have different users as listed above, but role of DBA is significant in many aspects. We discuss the role of DBA in 1.6 in detail.

5. **Procedures:** Procedures refer to the instructions (rules) that govern the design and use of the database. The users of the system and the staff that manage the database require documented procedures to use or run the system. The followings are some instructions so that we can follow procedure systematically:

(i) Log on to the DBMS

(ii) Start and stop the DBMS

(iii) Make backup copies of the database

(iv) Handle hardware or software failures.

(v) Change the structure of a table.

1.3 DBA (DATABASE ADMINISTRATOR)

- The DBA is a person or group of persons who control the right to access to the data and over all maintain policies to ensure softy and smoothly working of the system.
- DBA controls the design and use of database. DBA is responsible for implementing the database system within an organization.
- DBA provides a necessary technical support for implementing policy for smooth working of the database.
- DBA is responsible for evaluation, selection and implementation of DBMS package.
- The DBA has to perform number of important task like authorizing access to database, coordinating and monitoring different types of users, handling software and hardware issues.
- Database Administrator's job requires a high degree of technical expertise.
- In practice, the DBA may consist of team of people rather than just one person.

Functions Responsibilities of Database Administrator (DBA)

The database administrator performs a critical role within an organization and has to perform different functions and responsibilities Depending on the organization and the department, the role DBA can either be highly specialized or incredibly diversified. The functions of DBA are as follows:

1. **Defining conceptual schema and database creation:** The DBA is responsible for designing conceptual schema of database. The DBA defined how data is to be represented in the database and how tables are related to each other.

2. **Storage structure and access method definition:** The DBA is responsible to define storage structure and provide access methods to database. The DBA defines access policies to an individual or a group. The DBA also decide how the data is to be represented in the database.

3. **Defining integrity constraints:** The DBA defines integrity rules to ensure the accuracy of the data. The integrity rules are defined according to the nature and requirement of the users. The DBA defines the checks and integrity policies so that users can access data with freedom as checks and integrity constrains do not allow illegal operations.

4. **Ensuring availability of data:** The DBA is ensured that whenever request is made for data, data should be available. The availability of data around the clock is possible with appropriate steps to take backup and switching load among different systems. The DBA defines policies such that data must be available in time.

5. **Deciding backup and recovery methods:** The backup and recovery methods are very crucial for the safety of the data as system accidental failure can happen any time. The DBA decides which data is to be backed up and when. DBA defines policy for backup so that data loss can be avoided.

6. **Granting authorisation to the users:** The DBA defines list of users with access level so that data can be access by the authorised users only. The authorisation of user is monitored and updated by DBA time to time as one user may changes his access level.

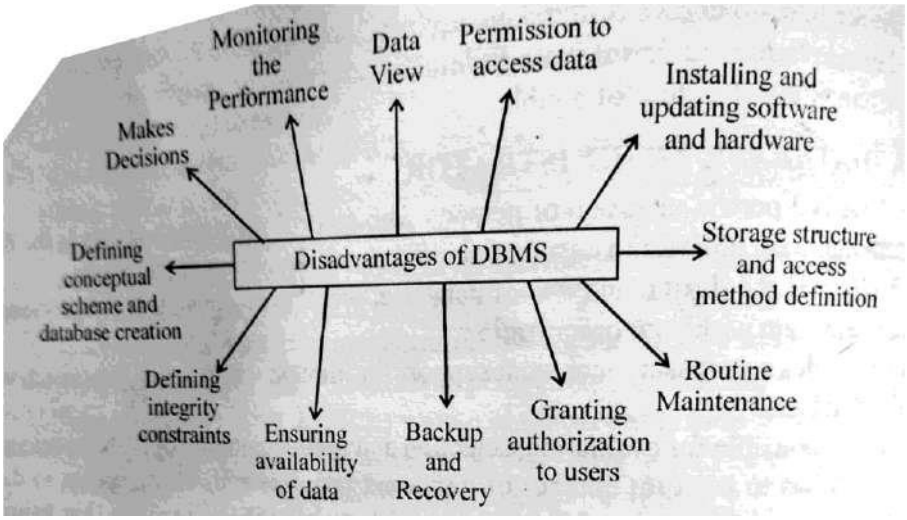


Fig. 1.10: Responsibilities of DBA

7. **Routine maintenance:** The routine maintenance includes up gradation of system, updating user profiles and other information regarding access policies. The Database Administrator understands the following routine maintenance activities:

- ***When transaction rollbacks occur*** DBA decides what to do when a transaction rollback. If such incidence occurs, then DBA checks the updating records and decide about re-do or undo of the transaction.
- ***When the database is out of system disk space:*** The database is stored into physical memory which in limited in size. When the data is about to reach maximum limit in disk space then DBA decide whether to erase some unwanted data or add new hard disk into system.
- ***When unique constraints have been violated:*** The access to data in database in done through some unique constrains whenever such rule is violated then DBA has to look into the matter The unique constraints policies are updated as their reports of violation such that violations can be avoided in future.
- When not to shut down the database while the application is running

8. **Installing and updating software and hardware:** The DBA is the person who is authorised to installed software into the database machine. The DBA decides the need of updating the hardware

9. **Permission to Access Data:** DBA gives permission to user to use database. Only authorised user can access data.

10. **Data view:** DBA can create different views of data that can be shown to different users.

11. **Monitoring the performance.** DBA is responsible for overall performance of the system. To improve the performance DBA regular monitor the system performance.

12. **Makes Decisions:** It is the DBA's job to decide exactly what information is to be held in the database.

1.4 COMPARISON OF FILE MANAGEMENT SYSTEM WITH DATABASE MANAGEMENT SYSTEM

Sr. No.	Concepts	File Management System	Database Management System
1.	Redundancy	Data redundancy (duplication) is possible. Data duplication is a commonly visible in file management system as multiple files are stored at different location.	Data redundancy (duplication) is not possible. In database management system, minimum redundancy occurred as duplication is avoidable.
2.	Consistency	Data is duplicated into number of files and consistency is a hard job in file management system. Consistency ensures that data at different location about single entity should be	Database management system avoids duplication as a result consistency can be avoided. Moreover data

		same with time.	
3.	Data Isolation	The data isolation is preserve in the file management system as data is stored in different file which are hardly associate with each other.	The data is stored into tables which are linked with each other so the isolation of data is not available.
4.	Standards enforcement	In an organization, every department has their own file system and format so uniform format and standards cannot be implemented. Moreover application programs are file dependent so new standard enforcement is hard to implement.	DBMS is enforced laws and policies in the form of standards which helps in maintaining database. Data is stored according to the standards and in uniform pattern so any programs are file dependent so new change in standard policies do not affect the working of the database.
5.	Data security	Data security is implemented at file level, whereas user level security itself is not feasible.	Data security is implemented at different levels and data security policies are upgradable to meet

			the requirements.
6.	Application dependency	The applications are developed according to the data file, it means we have to make certain changes whenever application applied on different format data file.	The application programs in DBMS are independent on database. Data independency is enforced in the system which helps the programmer to write general purpose programs
7.	Multiple Access	The multiple access to data file is not permitted in file management system.	In database management system, multiple access is permitted.
8.	Concurrency Problem	There is concurrency problem in file management system as multiple accesses is not desirable.	DBMS is designed to meet concurrent access which means more than one user can access same data without any problem.
9.	Technical platform	In file management system, every department developed their own applications to access data files. Generally applications are developed	DBMS provides integrated program kit which is developed using common language. Examples of DBMS are Oracle,

		in C, C++, COBOL etc.	Sequel & Foxpro etc.
10.	Real world modelling	Real world modelling is not possible as files management system stored data in files. The real world modelling requires object based data representation which is not possible in file based system.	Real world modelling is done through object representation in DBMS as data along with other attributes can be stored.
11.	No. of Files	There are less number of files as compared to DBMS	There are more number of files.
12.	Cost	It is cheaper as compared to DBMS.	It is costly
13.	Structure	It has simple structure	It is complex structure.
14.	Flexibility	It is less flexible as compared to DBMS.	It is more flexible.
15.	Efficiency	When the volume of data increases, its efficiency decreases	Volume of data not affect its working capability.

1.5 CATEGORIES OF DBMS

The classification of a database management system (DBMS) is greatly influenced by the underlying computing system on which it runs, in particular of computer architecture such as parallel, networked or distributed. However, the DBMS can be classified according to the number of users, the database site locations and the", expected

type and extent of use.

1.5.1 Centralized DBMS

1. In centralized .database systems, the database system, application programs, and user-interface all are executed on a single system and dummy terminals are connected to it.
2. It is physically confined to a single location.
3. The processing power of single system is utilized and dummy terminals are used only to display the information.
4. As the personal computers became faster, more powerful, and cheaper, the database system started to exploit the available processing power of the system at the user's side, which led to the development of client/server architecture,
5. In client/server architecture, the processing power of the computer system at the user's end is utilized by processing the user-interface on that system.
6. The centralised database system .consists of a single processor together with its associated data storage devices and other peripherals.
7. The system offers data processing capabilities to users who are located either at the same site, or, through remote terminals, at geographically dispersed sites.-
8. The management of the system and its data are controlled centrally form any one or central site.

The following diagram 2,6 show the centralized DBMS.

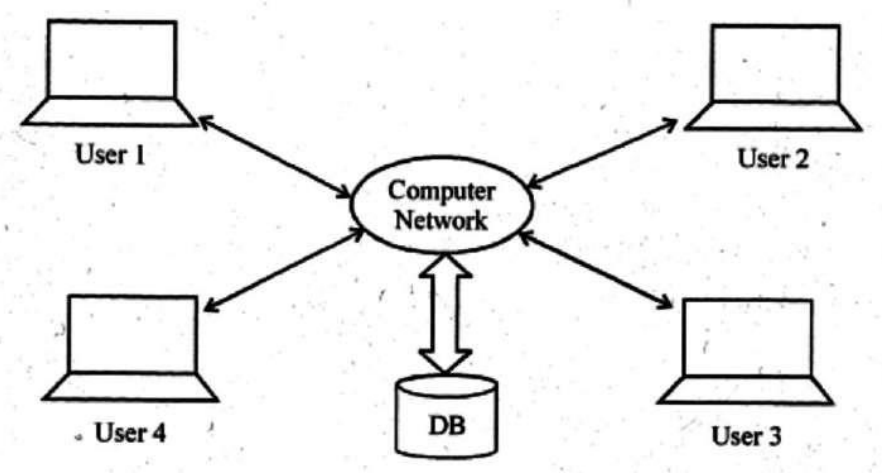


Fig. 2.6: Centralized DBMS

Advantage of a Centralized DBMS

1. **Centralized control:** The organization can exert centralized management and control over the data by Database Administrator (DBA). The database administrator is the focus of centralized control./
2. **Shared data:** A database allows the sharing of data under its control by any number of application programs or users.
3. **Reduction of redundancies: Centralized** control of data by DBA avoids unnecessary duplication of data and effectively reduces the total amount of data storage required. It also eliminates the extra processing necessary to trace the required data in a large mass of data.
4. **Integrity:** Centralized control can also ensure that adequate checks are incorporated in the DBMS to provide data integrity. Data integrity means that the data contained in the database is both accurate and consistent. Therefore data values being entered for storage could be checked to ensure that they fall within a specified range and in the correct format.
5. **Security:** Data is a vital importance to an organization and may be confidential. Such confidential data must not be accessed by unauthorized person.
6. **Data Independence:** Data independence allows dynamics changes and growth potential.
7. **Operations:** Most of the functions such as update, backup, query, control access and so on, are easier to accomplish in a centralised database system.
8. **Size of the Database:** The size of the database and the computer on which it resides need not have any bearing on whether the database is centrally located.

Disadvantage of a Centralized DBMS

1. **Problems associated with centralization:** Several problems are associated

with centralization like networking the excessive load on the system at the central site would likely causes all accesses to be delayed etc.

2. **Cost of software and migration:** The cost of purchasing or developing the software, the hardware has to be upgraded to allow for the extensive programs and the work spaces required for their execution and storage. The processing overhead is also added by implement security integrityof data causes a degradation of the response and through put times. It is also added the cost of migration from .a traditionally separate application environment to an integrated one.
3. **Complexity of backup and recovery:** The centralization reduces duplication, the lack of duplication required that the database be adequately backed up so that in the case of failure the data can be recovered. Backup and recovery operations are fairly complex in a DBMS environment.
4. **Server Down:** When the central site computer or database system goes down, then every user is blocked from using the system until the system comes back.
5. **Communication costs:** The communication costs from the terminals to the central site can be expensive.

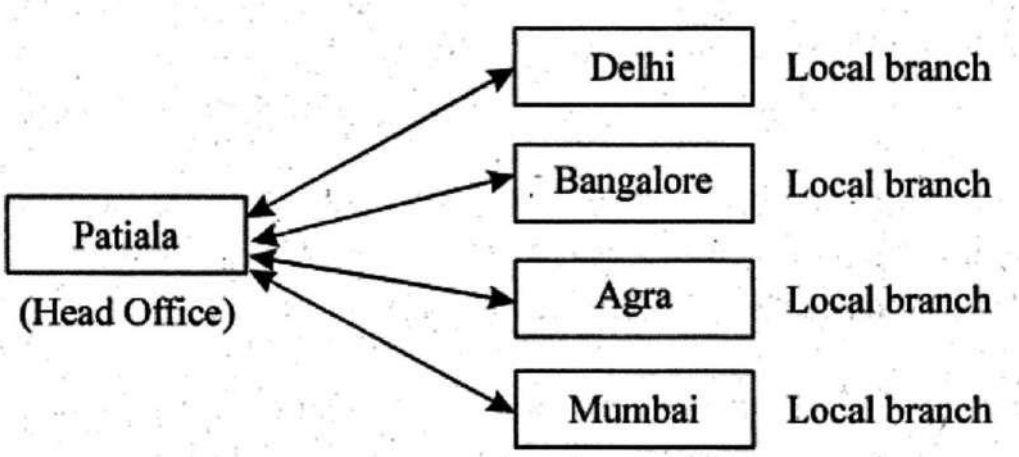
1.5.2 Parallel DBMS

1. Parallel database systems architecture consists of a multiple central processing units (CPUs) and data storage disks in parallel.
2. They improve processing and input/output (I/O) speeds.
3. Parallel DBMS are used in the applications that have to. query extremely large databases or that have to process an extremely large number of transactions per second.

The following diagram 2.7 shows the parallel DBMS.

- Shared data storage disk
- Shared memory
- Hierarchical

- Independent resources



Advantages of-a Parallel DBMS.

1. Parallel database systems are very useful for the applications that have to query extremely large databases.
2. In a parallel database system, the throughput and the response time are very high. Throughtput is number of tasks completed in -given time duration. Response time is amoung of time required by single task for completion.

Disadvantages of a Parallel DBMS

1. In a parallel DBMS, there is a .startup cost associated with initiating a single process and the startup-time may overshadow the processing time, affecting speedup adversely.
2. In parallel DBMS, the processes access the shared resources which slow down the result.

1.5.3 Distributed DBMS

1. Distributed DBMS consist soft of a single logical database that is spilt into number of fragments.
2. Distributed database systems are similar to client/server architecture in a number of ways.
3. Both typically involve the use of multiple computer systems and enable users to access data remote system.

4. Distribute database system broadens the extent to which data can be shared well beyond that which can be achieved with the client/server system.

Following diagram 2.8 shows the distributed DBMS architecture.

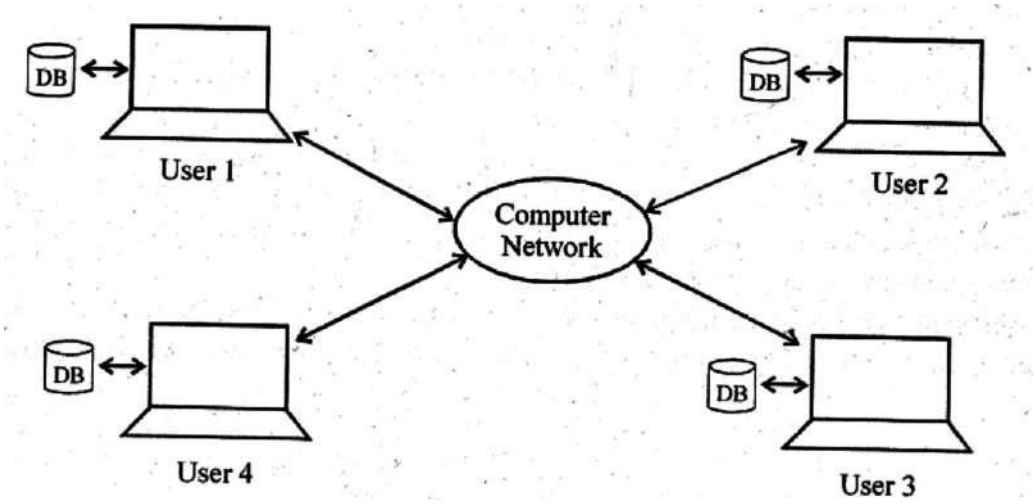


Fig. 2.8: Distributed DBMS

Advantages

1. **Efficiency and better Performance:** Distributed database architecture provides greater efficiency and better performance.
2. **Response time:** The response time and throughput is high as data is available at different places.
3. **Custom-built Machine:** The server database machine can be custom-built or tailored to the DBMS function and thus can provide better DBMS performance.
4. **Customized user interface:** The client application-database might be a personnel workstation tailored to the needs of the .end users and thus able to provide better interfaces, high availability, faster responses and overall improved ease of use to the user.
5. **Shearing of Database:** A single database on server can be shared across several distinct client application systems.
6. **Adding new location:** It causes less impact on ongoing operations when adding new locations. As data volumes and transaction rates increase, users

can grow the system incrementally.

7. Local autonomy: Distributed database system provides local autonomy.

Disadvantage of Distributed DBMS

The recovery from failure is more complex in distributed database systems than in centralized systems.

1.5.4 Client/Server Database System

1. Client/server architecture of database system has two logical components namely client, and server.
2. Clients are generally personal computers or workstations whereas server is large workstations, mini range computer system or a mainframe computers system.
3. The server computer is called backend and the client's computer is called front-end. These server and client computers are connected into a network.
4. The applications and tools of DBMS act as clients, making requests for its. services.
5. DBMS software resides on the server.
6. The DBMS, in turn, processes these requests and returns the results to the client(|).
7. The client/server architecture is a part of the open systems architecture in which all computing hardware, operating systems, network protocols and other software are interconnected as a network and work in concert to achieve user goals.
8. It is well suited for online transaction processing and decision support applications, which tend to generate a number of relatively short transactions and require a high degree of concurrency.

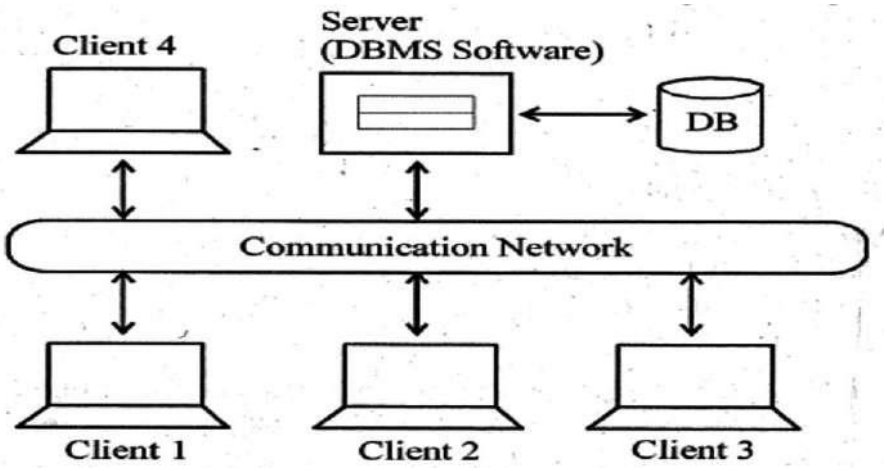


Fig. 2.9: Client Server DBMS

As shown in Fig. 2.9, the client/server database architecture consists of three components namely, client applications, a DBMS server and a communication network interface. The client applications may be tools, user-written applications or vendor-written applications. They issue SQL statements for data access. The DBMS server stores the related software; processes the SQL statements and returns results. The communication network interface enables client applications to connect to the server, send SQL statements and receive results or error messages or error return codes after the server has processed the SQL statements. In client/server database architecture, the majority of the. DBMS services are performed on the server.

Advantages of Client/Server DBMS

1. **Less expensive:** Client-server system has less expensive platforms to support^ applications that had previously, been running only on large and expensive mini or mainframe computers.

2. **Menu-drive interface:** Clients offer icon-based menu-driven interface, which is superior to the traditional command-line, dumb terminal interface typical of mini and mainframe computer systems.

3. **Flexible and productive environment:** Client-server database system is more flexible as compared to the centralised system. Client/server environment facilitates in more productive work by the users and making better use of existing data.

4. **Response time and throughput:** The client server model is based on

request and reply model when a machine make request to the server then server immediately reply so Response time and throughput is high.

5. **Custom-built Servers:** The database server machine can be custom-built or tailored to the DBMS function-and thus can provide a better DBMS performance.

6. **Custom build Client machine:** The client application database might be a personnel workstation, tailored to the needs of the end users and thus able to provide better interfaces, high availability, faster responses and overall improved ease of use to the user.

7. **Powerful Single Server:** A single database on server can be shared across several distinct client application systems.

Disadvantages of Client/Server DBMS

1. **High set cost:** The setup cost is high which include labour or programming, cost is high in client/server environments, particularly in initial phases.

2. **Lack of management tools:** There is a lack of management tools for diagnosis, performance monitoring and tuning and security control, for the DBMS, client and operating systems and networking environments.

Questions

1. What do you mean by data? How is it different from information, explain by example?
2. What is database system? What are four components of database system?
3. What are advantages of database system?
4. What is DBMS? What are the advantages and disadvantages offered by such system?
5. What are the main responsibilities of DBA? Explain.
6. What do you mean by file system? Explain it limitations.
7. Compare file management system with database management system.

COURSE: DBMS

UNIT 2: DBMS ARCHITECTURE

2. INTRODUCTION

2.1 THREE LEVEL ARCHITECTURE OF DBMS

2.1.1 OBJECTIVES OF ARCHITECTURE

2.1.2 EXTERNAL LEVEL/EXTERNAL VIEW

2.1.3 CONCEPTUAL LEVEL/COMMUNITY USER VIEW/LOGICAL LEVEL

2.1.4 INTERNAL LEVEL/STORAGE VIEW/PHYSICAL LEVEL

2.1.5 DATABASE SCHEMA AND DATABASE INSTANCE

2.1.6 MAPPING BETWEEN DIFFERENT VIEWS

2.2 EXAMPLE OF THREE LEVEL ARCHITECTURE

2.3 DATA INDEPENDENCE

2.4 DIFFERENCE BETWEEN LOGICAL DATA INDEPENDENCE AND PHYSICAL DATA INDEPENDENCE

2.5 COMPONENTS OF A DBMS

2.6 DATA DICTIONARY

2.7 DBMS LANGUAGES

2. INTRODUCTION

Architecture of Database Management System: *The architecture of DBMS is a framework for describing database concepts and specifies the structure of the database*

system. It describes Junctions of each component and describes how these components communicate with each other in a logical manner.

The database management system Is a sophisticated software application which is designed to provide interface to the user so that user can perform different operations on database with ease. The design of a database management system highly depends on its architecture. It can be centralized or decentralized or hierarchical depending upon the type of applications. Its architecture can be single tier or multi-tier. The multi-tier architecture divides the database management system into related but independent different modules which can be independently modified, altered, changed or replaced. In case of multi-tier architecture, the best suitable architecture is 3-tier architecture.

2.1 THREE LEVEL ARCHITECTURE OF DBMS

Database management system is described in three different levels which have separate functioning and working. These three different levels are named as **external level, conceptual level** and **internal level.** These levels are shown in a serial view of the architecture:

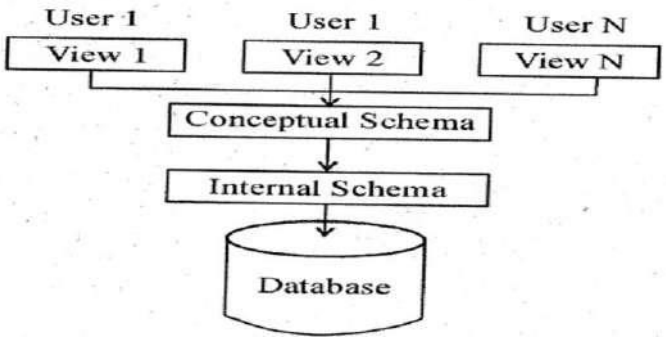


Fig. 2.1: Three Level Architecture (an Aerial view)

2.1.1 Objectives of Architecture

1. It should provide an interface to make changes into the structure of database without changing the application program at external schema.
2. Each user should be able to change the way he view the data and his change should not affect other users.
3. User should not directly deal with the physical database storage.

4. Users are independent of the storage complexities like indexing constraints etc. of the database
6. The conceptual structure of the database has no effect due to the change of the physical storage devices.
7. DBA should be able to change the storage structure and conceptual structure without affecting user's and his view level.

The core objective to design three levels is to provide data independence and physical independence. It is required to provide an easy to use interface to the end users.

2.1.2 External Level/External View

The external level is more concerned with the way in which the data is viewed by individual users. It is closer to the user and provides an interface to interact with the database. Each user has different requirement of the data so DBMS presents each user with a shared or single view or schema of the data. In external level, the different views may have different representations of the same data. For example, one user may view date in the form as (day-month-, year) while another may view as (year-month-day). Similar one view of data may show detail of employee with fields (Name, DOB, Address) and other view may show employee detail with salary (Name, DOB, Basic Pay, HRA, DA). The external view is user specific and provides an abstraction of data which helps to user to view important data and hide additional information.

Characteristics/Functions/Key Points of External Level

1. The external level is at the highest level of database abstraction where only those data is visible to the user which is concerned to the user at that time.
2. External view is user's view of database. It may provide limited and complete access to the database.
3. External schema consists of definition of logical records and their relationships in the external view.

4. External level is also known as view level and closest to the end users. It acts-as an interface to access data. User need not to know the details of data structure and physical storage.
5. External level provides the way in which individual users can view data according to his/her requirements i.e. one user may view data in the form (day, month, year) while another user may view data as (year, month, day).
6. Same database can have different views for different users.
7. Its core purpose is to provide user friendly interface to the end user.

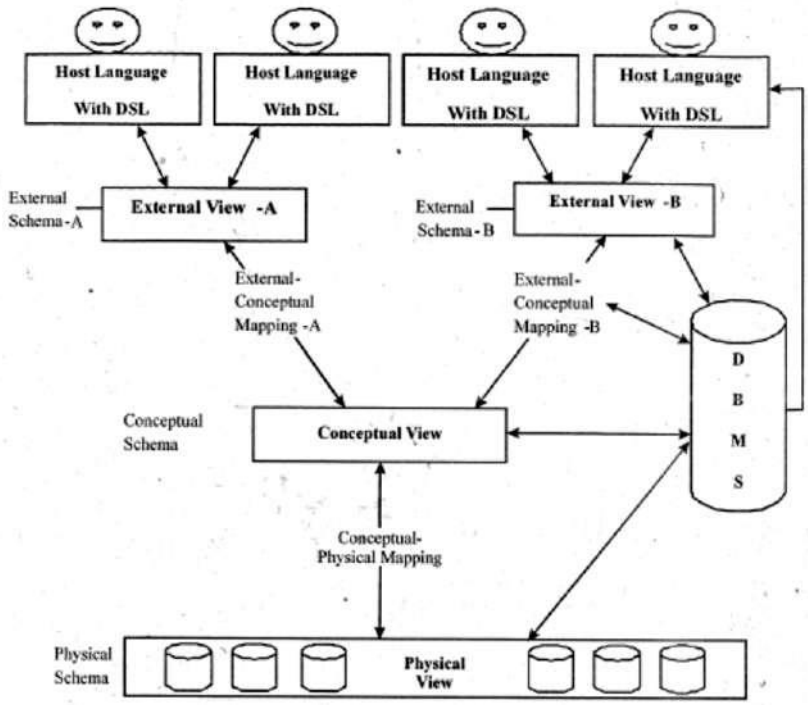


Fig. 2.2: Three Level Architecture with Different Schemas

2.1.3 Conceptual Level/Community User View/Logical Level

Conceptual Level represents the entire database. Conceptual schema describes the records and relationship included in the Conceptual view. The external level is concerned with individual user view whereas the conceptual level represents community user view. The conceptual schema hides the details of physical structure and concentrates on describing entities, data type, relationships, user operations and constraints. The view is

normally more stable than the other two views. The ultimate objective of the conceptual schema is to describe the complete enterprise-not just its data but also how that data is used, how it flows from point to point within the enterprise.

Characteristic/Functions/Key Points of Conceptual Level

1. Conceptual level is also known as middle level. It is created and maintained by DBA.
2. The conceptual schema hides the details of physical structure and concentrates on describing data type entities, their attributes and relationships, user operations.
3. It implements constraint on fie data.
4. At this level, different security and integrity rules can be imposed on data.
5. The semantic information about the data can be represented in conceptual view.
6. Different types of validation checks to retain data consistency and integrity are enforced at conceptual level.
7. It describes what data is stored in database and relationship among database.

2.1.4 Internal Level/Storage View/Physical Level

The internal level is closest to the physical storage which is concerned with the way in which the data is actually stored. The internal view is described by means of the internal schema, which not only defines the various stored record types but also specifies the indexes are in and so on.

Characteristic/Functions/Key Points of Internal Level

1. It is the physical representation of data.
2. It describes how the data is stored in database. It manages storage space allocation for data.
3. It concern with the physical implementation of the database to achieve optimal runtime performance and space utilization.
4. Record description for storage with stored sizes for data items.

5. Access path e.g. specification of primary and secondary keys, index and pointers.
6. Data compression and encryption techniques.
7. Optimization of the internal structures.
8. It builds the indexer, retrieve the data and so on.

2.1.5 Database Schema and Database Instance

While working with any data model, it is necessary to distinguish between the overall design or description of the database (database schema) and the database itself. The database schema is also known as intension of the database, and is specified while designing the database.

1. Schema

A schema is plan of the database that gives the names of the entities and attributes and the relationship among them. A schema includes the definition of the database name, the record type and the components that make up the records. Alternatively, it is defined as a framework into which the values of the data items are fitted. The values fitted into the framework changes regularly but the format of schema remains the same.

Key Points of Schema

- The plan or scheme of the database is known as Schema.
- It gives the names of the entities, attributes and relationship among them.
- It is the framework into which the values of data items are fitted.
- Overall description of database is known as database schema.

Types of Schema

Generally, a schema can be partitioned into/three categories which are as follows:

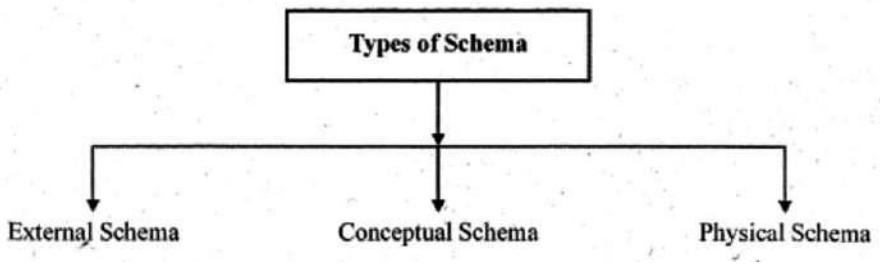


Fig. 2.3: Types of Schema

(a) ***External Schema:*** The external schema is concerned with the description of external view, correspond to different view according to the requirements of the users.

(b) ***Conceptual Schema:*** The conceptual schema is concerned with the description of all the entities, attributes and relationships along with the constraints. The logical (conceptual) schema is concerned with exploiting the data structures offered by the DBMS so that the schema becomes understandable to the computer.

(c) ***Physical Schema:*** The physical schema is concerned with the manner in which the conceptual database gets represented in the computer as a stored database. It is hidden behind the conceptual schema and can usually be modified without affecting the application programs.

2. Subschema

- A subschema is a subset of the schema having the same properties that a schema has.
- It identifies a subset of areas, sets, records, and data names defined in the database schema available to user sessions.
- It allows the user to view only that part of the database that is of interest to him.
- It defines the portion of the database as seen by the application programs and the application programs can have different view of data stored in the database.
- The different application programs can change their respective subschema without affecting other's subschema or view.

3. Instances

- The data in the database at a particular moment of time is called an instance or a database state.

- In a given instance, each schema construct has its own current set of instances. Many instances or database states can be constructed to correspond to a particular database schema.
- Every time we update (i.e., insert, delete or modify) the value of a data item in a record, one state of the database changes into another state.

The following figure shows an instance of the ITEM relation in a database schema.

ITEM		
ITEM-ID	**ITEM_DESC**	**ITEM_COST**
1111A	Disc	30
1112A	Mother Board	500
1113A	CD	100
1144B	Processor	5000

2.1.6 Mapping Between Different Views

Mapping: In three schema architecture, each user group refers only to its own external view. Whenever a user specifies a request to generate a new external view, the DBMS must transform the request specified at external level into a request at conceptual level, and then into a request at physical level. If the user requests for data retrieval, the data extracted from the database must be presented according to the need of the user. *This process of transforming the requests and results between various levels of DBMS architecture is known as mapping.*

The DBMS is responsible for mapping between the three types of schema. Two mapping are required in database systems which are as follows:

(a) External/Conceptual Mapping: Each external scheme is related to the conceptual schema by the external/conceptual mapping. The external/ conceptual mapping gives the correspondence among the records and the relationships of the external and conceptual views. A given external record could be derived from a number of conceptual records.

(b) Conceptual/Internal Mapping: Conceptual schema is related to the internal schema by the conceptual/internal mapping. The conceptual/ internal mapping specifies the method of deriving the conceptual record from the physical database.

Advantages of View Mapping

1. Each user is able to access the same data but have a different customized view of the data as per their own needs.
2. A user can change his/her view and this change does not affect other user views.
3. There user's interaction with the database is independent of physical data storage organization.
4. The database administrator is able to change the database storage structure without affecting the user's view.
5. The database administrator is able to change the conceptual structure of the database without affecting all users.
6. The database administrator can change existing storage devices with the new storage devices without affecting others user's.

2.2 EXAMPLE OF THREE LEVEL ARCHITECTURE

To understand the three-schema architecture, consider the three levels of the BOOK file in Online Book database as shown in Figure this figure, two views (view 1 and view 2) of the BOOK file have been defined at the external level. Different database users can see these views. The details of the data types are hidden from the users. At the conceptual level, the BOOK records are described by a type definition. The application programmers and the DBA generally work at this level of abstraction. At the internal level, the BOOK records are described as a block of consecutive storage locations such as words or bytes. The database users and the application programmers are not aware of these details; however, the DBA may be aware of certain details of the physical organization of the data.

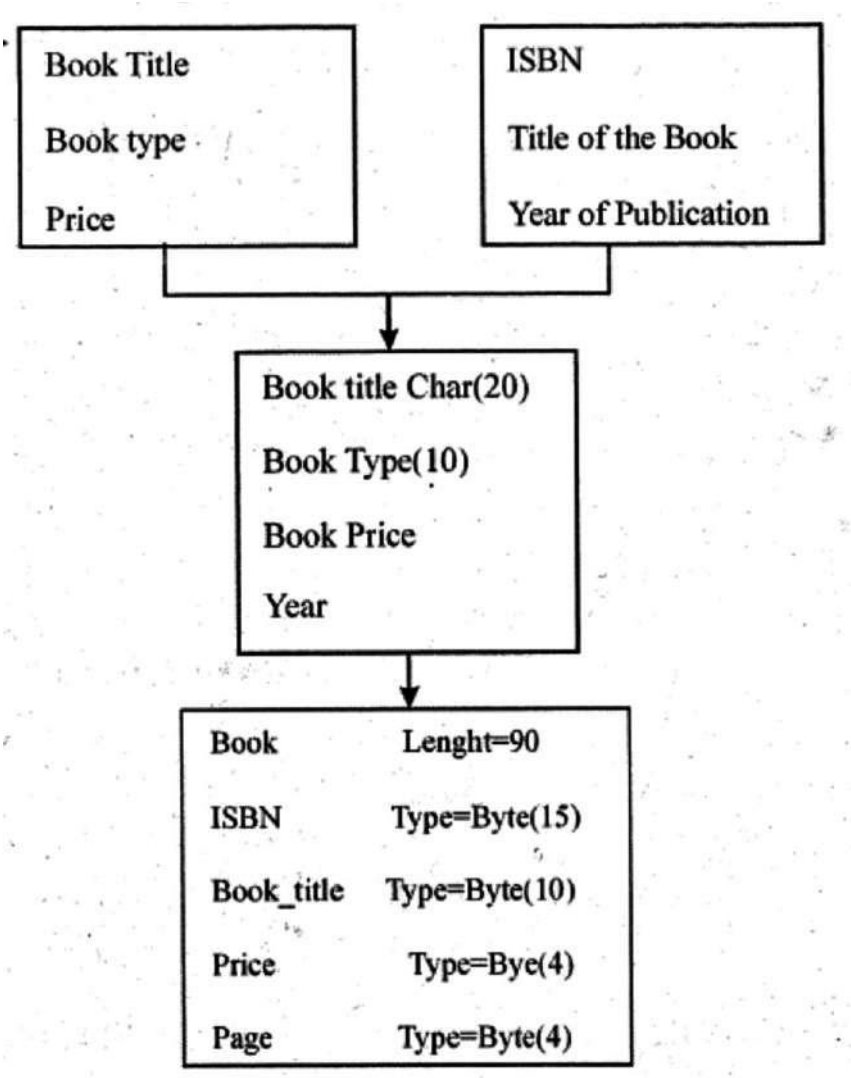


Fig. 2.4: Three Level Schema Architecture

2.3 DATA INDEPENDENCE

Data Independence: *The ability of a database management system to modify its Schema definition at one level without affecting a Schema definition at the next level is called data Independence. It provides flexibility to make changes at one Schema level without affecting the next level Schema.*

Key Points of Data Independence

- The main advantage of three-schema architecture is that it provides data independence.

- Data independence Is the ability to change the, schema at one level of the database system without having to change the schema at the other levels.
- The data independence deals with independence between the way the data is presented, structured and stored. It provides independence to make changes in one level without affecting other levels.
- Data independence means upper levels are unaffected by the changes in lower level.
- For example, DBMS may change the structure of the data without having to change application program. It is possible due to mapping between three levels which enable the user to make changes at one level without affecting the next level of architecture.
- Data Independence implies that the application programs should not need to know any of the following:
  - Ordering of data fields in a record
  - The size of the record
  - The size of the field
  - The format and type of each data item
  - The type of data structured used to store the data.
- The three level DBMS architecture provides two type of data independence. The first is called *logical data independence* and second is called physical data independence. The logical independence is enabled the user to change the conceptual view without affecting the external view. Whereas, Physical data independence is the idea to make changes into internal view without affecting the conceptual or external views. These two types of data independence are discussed in detail below:

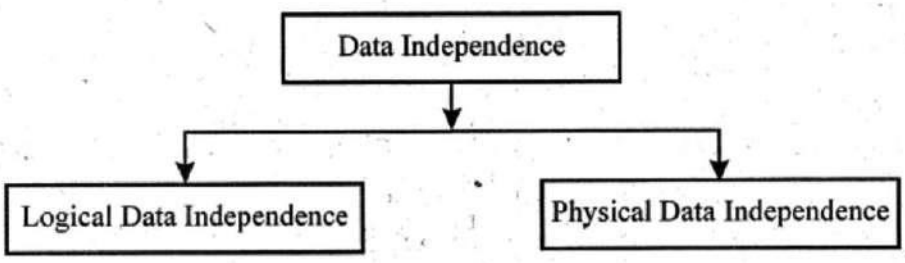


Fig. 2.5: Types of Data Independence Logical Data Independence*

- It is the ability to change the conceptual schema without affecting the external schemas or application programs.
- The conceptual schema may be changed due to change in constraints or addition of new data item or removal of existing data item, etc., from the database.
- The separation of the external level from the conceptual level enables the users to make changes at the conceptual level without affecting the external level or the application programs.
- For Example: The name field in conceptual view is stored as first hame, middle name and last name whereas in external view, it remains to be as a single name field.
- It indicates that the conceptual schema can be changes without affecting the existing external schema.
- It requires the flexibility-in the design of database.
- The programmer is required to make modification in the design as per the requirements.

(b) Physical data independence:

- It is the ability to change the internal schema without affecting the conceptual or external schema.
- An internal schema may be changed due to several reasons such as for creating additional access structure, changing the storage structure, etc.
- The separation of internal schema from the conceptual schema facilitates physical data independence.

- For Example: The location of the database, if changed from C drive to D drive will not affect the conceptual view or external view as the commands are independent of the location of the database.
- It indicates that the physical storage structure used for thedata could be change without affecting the conceptual schema.
- The storage structure and access methods used to retrieve of the data from physical storage medium are not concerned with conceptual schema.

Logical data independence is more difficult to achieve than the physical data independence because the application programs are always dependent on the logical structure of the database. Therefore, the change in the logical structure of the database may require change in the application programs.

2.4 DIFFERENCE BETWEEN LOGICAL DATA INDEPENDENCE AND PHYSICAL DATA INDEPENDENCE

Sr. No.	Logical Data Independence	Physical Data Independence
1.	Whenever, there is a change or modification at the conceptual level without t affecting the user level or external level, it is "known as logical data independence.	Whenever, the changes are made at the internal level without affecting the above layers, it is known as physical data independence.
2.	It is concerned with the structure of the data or changing the data definition	It is concerned with the storage of the data.
3.	It is concerned with the conceptual schema	It is concerned with the internal schema.
4.	Application program need not be change if new fields are added	Physical database is concerned with the change of the storage device
5.	It is very difficult to retrieve the data because data re heavily dependent on the	It is easy to retrieve the data.

	logical structure of data	

2.5 COMPONENTS OF A DBMS

The DBMS accepts the SQL commands generated from a variety of user interfaces, produces query evaluation these plans against the database, and returns the answers.

1. **Query processor:** The query processor transforms users queries into a series of low-level instructions directed to the run time database manager. It is used to interpret the online user's query and convert it into an efficient series of operations in a form capable of being sent to the run time data manager for execution. The query processor uses the data dictionary to find the structure of the relevant portion of the database and uses this information in modifying the query and preparing an optimal plan to access the database.

2. **Run time database manager:** Run time database manager is the central software component of the DBMS, which interfaces with user-submitted application programs and queries. It handles database access at run time. It converts operations in user's queries coming directly via the query processor or indirectly via an application program from the user's logical view to a physical file system. It-accepts queries and examines the external and conceptual schemas to determine what conceptual records are required to satisfy the users request. The run time data manager then places a call to the physical database to perform the request. It enforces constraints to maintain-the consistency and integrity of thewell as its security. It also performs backing and recovery operations. Run time database manager is sometimes referred to as the *database control system* and has the following components:

(i) **Authorization control:** The authorization control module checks that the user has necessary authorization to carry out the required operation.

(ii) **Command processor:** The command processor processes the queries passed by authorization control module.

(iii) **Integrity checker:** The integrity checker checks for necessary

integrity constraints for all the requested operations that changes the database.

(iv) **Query optimizer:** The query optimizer determines an optimal strategy for the query execution. It uses information on how the data is stored to produce an efficient execution plan for evaluating query.

(v) **Transaction manager:** The transaction manager performs the required processing of operations it receives from transactions. It ensures that (a) transactions request and release locks according to a suitable locking protocol and (b) schedules the execution of transactions.

(vi) **Scheduler:** The scheduler is responsible for ensuring that concurrent operations on the database proceed without conflicting with one another. It controls the relative order in which transaction operations are executed.

(vii) **Data manager:** The data manager is responsible for the actual handling of data in the database. This module has the following components:

(a) ***Recovery manager:*** The recovery manager ensures that the database remains in a consistent state in the presence of failures. It is responsible for (a) transaction commit and abort operations, (b) maintaining a log, and (c) restoring the system to a consistent state after a crash.

(b) ***Buffer manager:*** The buffer manager is responsible for the transfer of data between the main memory and secondary storage (such as disk or tape). It brings in pages from the disk to the main memory as needed in response to read user requests. Buffer manager is sometimes referred as the *cache manager*.

3. **DML processor:** Using a DML compiler, the DML processor converts the DML statements embedded in an application program into standard function calls in the host language. The DML compiler converts the DML statements written in a host programming language into object code for database access. The DML processor must interact with the query processor to generate the appropriate code.

4. **DDL processor:** Using a DDL compiler, the DDL processor converts the -

DDL statements into a set of tables containing metadata. These tables contain the metadata concerning the database and are in a form that can be used by other components of the DBMS. These tables are then stored in the system catalog while control information is stored in data file headers. The DDL compiler processes schema definitions, specified in the DDL and stores description of the schema (metadata) in the DBMS system catalog. The system catalog includes information such as the names of data files, data items, storage details of each data file, mapping information amongst schemas, and constraints.

2.6 DATA DICTIONARY

1. Data dictionary is also known as *Meta data*. A metadata is the data about the data. It is the self-describing nature of the database that provides program-data independence. It is also called as the *System Catalog.*

2. ***Data Dictionary is a repository-of information about a database that documents data elements of a database. It stores information about the database, attribute names and definitions for each table in the database***

3. It holds the following information about each data element in the databases, it normally includes:

- Name
- Type
- Range of values
- Source
- Access authorization

4. Data dictionary is the integral part of the DBMS. Maintaining the data dictionary is the responsibility of DBA (Database Administrator).

5. The most general structure of data dictionary is shown in figure....

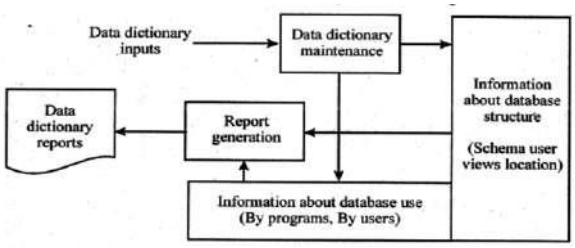


Fig. 2.10: Data Dictionary

6. Data dictionary is usually a part of the system catalog that is generated for each database. *A useful data dictionary system usually stores and manages the following types of information:*

- Descriptions of the schema of the database.
- Detailed information on physical database design, such as storage structures, access paths and file and record sizes.
- Description of the database users, their responsibilities and their access rights.
- High-level descriptions of the database transactions and applications and of the relationships of users to transactions.
- The relationship between database transactions and the data items referenced by them. This is useful in determining which transactions are affected when certain data definitions are changed.
- Usage statistics such as frequencies of queries and transactions and access counts to different portions of the database.
- Data dictionary provides the name of a data element, its description and data structure in which it may be found.
- Data dictionary provides great assistance in producing a report of where a data element is used in all programs that mention it.
- It is also possible to search for a data name, given keywords that describe the name. For example, one might want to determine the name of a variable that stands for net pay. Entering keywords would produce a list of possible identifiers and their definitions. Using keywords one can search the dictionary to locate the proper identifier to use in a program.

7. Data dictionary is used by developers to develop the programs, queries, controls-and procedures to manage and manipulate the data. It is available to database administrators (DBAs), designers and authorized user as on-line system documentation. This improves the control of database administrators (DBAs) over the information system

and the user's understanding and use of the system.

2.7 DBMS LANGUAGES

1. The main objective of a database management system is to allow its users to perform a number of operations on the database such as insert, delete, and retrieve data in abstract of data.
2. To provide the various facilities to different types of users, a DBMS normally provides one or more specialized programming languages called **Database** (or **DBMS**) **Languages.**
3. The DBMS mainly provides two database languages, namely, data definition language and data manipulation language to implement the databases.
4. Data definition language (DDL) is used for defining the database schema. The DBMS comprises DDL compiler that identifies and stores the schema description in the DBMS catalog.
5. Data manipulation language (DML) is used to manipulate the database.

The following are the DBMS languages:

1. **Data Definition Language:** DDL is used to specify the structure of table.

Sr.	Need And Usage	The SQL DDL Statement
1	Create schema objects	CREATE
2	Alter schema objects	ALTER
3	Delete schema objects	DROP
4	Rename schema objects	RENAME

We will discuss these statement in the chapter 10.

2. **Data Manipulation Language:** The DBMS provides data manipulation language (DML) that enables users to retrieve and manipulate the data. The statement which is used to retrieve the information is called a query. The part of the DML used to retrieve the information is called a query language.

S No.	**Need And Usage**	**The SQL DDL Statement**

1	Remove rows from tables or views	DELETE
2	Add new rows of data into table or view	INSERT
3	Retrieve data from one or more tables	SELECT
4	Change columns values in existing rows of a table or view	UPDATE

We will discuss these statement in the chapter 10.

3. **Data Control Language-(DCL):** DCL statements control access to data and the database using statements such as GRANT and REVOKE. A privilege can either be granted to a user with the help of GRANT statement. We can also revoke these statements by using REVOKES command.

S. NO.	**Need and Usage**	**The SQL DDL Statement**
1	Grant and take away privileges and roles	GRANT and REVOKE
2	Add a comment to the data dictionary	COMMENT

We will discuss these statement in the chapter 10.

Questions

1. Discuss the concept of data independence and explain its importance in a database environment.
2. What is logical data independence and why is it important?
3. What is the difference between physical data independence and logical data independence?
4. Explain the difference between external, conceptual and internal schemas. How are these different schema layers related to the concepts of physical and logical data independence?
5. Describe the structure of a DBMS.
6. Describe the main components of a DBMS with a neat sketch, explain the structure of DBMS.
7. What do you mean by a data model? Describe the different types of data models

used.

8. Explain the following with their advantages and disadvantages:
   (a) Hierarchical database model
   (b) Network database model E-R data models
   (c) Relational database model
   (d) E-R data models
   (e) Object-oriented data model.
9. Define the following terms:
   (a) Data independence
   (b) Query processor
   (c) DDL processor
   (d) DML processor.
   (e) Run time database manager.
10. What is meant by the term client/server architecture and what are the advantages and disadvantages of this approach?
11. Compare and contrast the features of hierarchical, network and relational data models. What business needs led to the development of each of them?
12. Differentiate between schema, subschema and instances.
13. Explain the advantages and disadvantages of a centralised DBMS.
14. Explain the advantages and disadvantages of a parallel DBMS.
15. Explain the advantages and disadvantages of a distributed DBMS.
16. Explain data dictionary in detail.

COURSE: DBMS

UNIT 3 : DATA MODELS

3. INTRODUCTION

3.0 Evolution of Major Data Models

3.1 RDBMS

3.2 E.F. CODD'S RULES

3.3 COMPARISON BETWEEN DBMS AND RDBMS

3.4 DATA MODEL

3.5 CLASSIFICATION OF DATA MODEL

3.6 RECORD BASED MODELS

3.6.1 Hierarchical Model

3.6.2 Network Model

3.6.3 Relational Model

3.7 PHYSICAL MODEL

3.8 OBJECT BASED MODELS

3.8.1 E-R Model (Entity Relationship Model)

3.8.2 Object Oriented Model

3.8.3 Semantic Model

3.8.4 Functional Model

3.9 COMPARISON OF DATA MODELS

3.10 OTHER TERMS USED IN E-R MODEL

INTRODUCTION

A Data Model defines the logical design of the data. It describes the relationships between different parts of the data. Data model tells how the logical structure of a

database is modeled. Data Models are fundamental entities to introduce abstraction in DBMS. Data models define how data is connected to each other and how it will be processed and stored inside the system.

Evolution of Major Data Models

The historical literature reported drastic changes in the year 1970-1994. The Edgar F. Codd, in year 1970 disclose new concept of data representation. Mr. Codd suggested that all data in a database could be represented as a tabular structure (tables with columns and rows, which he called relations) and that these relations could be accessed using a high-level nonprocedural language. This research was result of several Relational DBMS like Oracle, Informix, Ingres and DB2. The following was high lights related to evolution database:

- **1980s:** The several vendors had developed OODBMSs like Object Design, Versant, O2 and Objectivity. The OODBMSs were no threat in the late 1980s to the now big commercial vendors developing and selling hierarchical, network or relational databases.
- **1990s:** In 1990s. The Object Database Management Group was founded, mainly &thanks to Rick Cattell of JavaSoft. The Green Team started the development of a new programming language which was loosely based on C++.

  The language was named Oak after the trees outside the office window of the language designer - James Gosling.
- **1993s to till date:** In 1993 several vendors of OODBMSs agreed upon an OODBMS standard called ODMG-93. The relational databases already had its standard-SQL-92, defined by its ANSI committee and ISO. The concept of internet, xml and other database management system was evolved in the time span. The detailed summery is represented in table below:

Generation	Time	Model	Examples	Comments
First	1960s-1970s	File . system	VMS/VSAM	Used mainly on IBM mainframe system Managed records, not relationships

Second	1970s	Hierarchical and network IDS --II	IMS ADABAS	Early database systems. Navigational access
Third	Mid-1970s to present	Relational	DB2 Oracle MS SQL-Server	Conceptual simplicity Entity relationship (ER) modeling support for relational data modeling
Fourth	Mid-1980s to present	Object oriented Extended Relational	Versant VFS/Fast Objects Objectivity/DB	Support complex data Extended relational products support objects and data warehousing Web databases become common
Next Generation	Present to future	XML	dbXML Tamino DB2 UDB Oracle 10gMS SQL Server	Organization and management of unstructured data Relational and object modelsaddsupport for XML documents

Table 3.1: A brief summary of how the major data models where developed

3.1 RDBMS

1. RDBMS stands for "Relational Database Management System[9]',
2. ***'"RDBMS is a DBMS in which data is stored in the formthe form of tables and the relationship among the data is also stored in the form of tables."***
3. RDBMS also provide relational operators to manipulate the data stored into the database tables.
4. It is based on the relational model and was introduced by E.F. Codd.
5. E.F. Codd, the famous mathematician has introduced 12 rules (known as Codd's rules) toassist a database product to qualify a RDBMS. - .
6. RDBMSproduct has to satisfy at least 6 of the 12 rules, of Codd to be accepted as a full-fledged RDBMS.
7. Examples of RDBMS are: Oracle, Sybase, SQL- Server.
8. In short, all the information in RDBMS should be presented in tabular form and it follows Codd's rules.

3.2 E.F. CODD'S RULES

1. E.R Codd the famous mathematician has introduced 12 rules for the relational model for databases commonly known as Codd'srules.
2. These rules define what is required for a DBMS to be considered RDBMS.

3. The Codd's rules are as follows:

(a) **InformationRule:**Every information in RDBMS is represented in the form of tables.

(b) **Guaranteed Access Rule:**Every information in RDBMS is accessed by usingcombination of table name and primary key. A primary key helps to identify a rowname and column name.

(c) **Systematic Treatment of Null Values:** RDBMS supports null valuesfor representingmissingorInapplicable information.

(d) **The Description Rule:**The database description is represented at the logicallevel in the same way as ordinary data. The authorized users can apply the samerelational language for its manipulation as they apply to the regular data.

(e) **The Comprehensive Data Sublanguage Rule:** RDBMS supports many languages which allow users to define tables, query and update the data and set integrity constraints.

(f) **The View Updating Rule:**All the viewsthat are theoretically updatable must be updatable by the system.

(g) **High Level Insert, Update and Delete:** The system must support insert, update and delete operations.

(h) **Physical Data Independence:**It is the ability to change the internal schema (PhysicalLevel) without affecting the conceptual/logical or external schema.

(i) **Logical Data Independence:**Logical data independence is more difficult to achievethan the physical data independence. It is the ability to change the conceptual/logical schema without affecting the external schema.

(j) **Integrity Independence:**Integrity constraints should be specified separately fromapplication programs and stored in the catalog. Integrity constraints can be changedwithout affecting the application programs.

(k) **Distribution Independence:**User should not have to be aware of whether a databaseis distributedat different sites or not.

(l) **TheNon-SubversionRule:** If the RDBMS has a language that accesses the information of a record at a time, this language should not be used to bypass the integrityconstraints. .

3.3 COMPARISON BETWEEN DBMS AND RDBMS

DBMS	RDBMS
1. It stands for "Database Management System."	It stands for "Relational Database Management System."
2. It can store data in any format (graph, table, tree etc.)	It can store data only in tabular form.
3. It does not support client/server architecture.	It supports client/server architecture.
4. It does not satisfy Codd's rules.	It satisfyCodd's rules.
5. It requires low software and hardware requirements.	It requires high software and hardware requirements.
6. It can maintain only single user at a time. It supports single user.	It can maintain many users at a time. It supports multi-user.
7. It is designed for small organizations with small amount of data, where security of data is not a major issue.	It is designed for large organization with large amount of data where security of data is a major issue.
8. It does not support referential constraints.	It supports referential integrity constraints.
9. Examples of DBMS: Dbase, Foxpro	Examples of RDBMS: Oracle, Sybase, SQL-Server.

3.4 DATA MODEL

1. A model is a representation of reality, 'real world' objects and events,and their association.
2. A data model represents the organization itself.
3. Data model can be defined as an integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization.
4. The purpose of a data model is to represent data and to make the data understandable.

Objectives of Data Model

- The main objective of database system is to highlight only the essential features and to hide the storage and data organization details from the user.
- A database model provides the necessary means to achieve data abstraction.
- A data, model is an abstract model that describes how the data is represented and used.
- A data model consists of a set of data structures and conceptual tools that is used todescribe the structure (data types, relationships, and constraints) of a database.
- A data model not only describes the structure of the data, it also defines a set of operations that can be performed on the data.
- A data model generally consists of data model theory, which is a formal description of howdata may be structured and used.
- The process of applying a data model theory to create a data model instance is known asdata modeling.

3.5 CLASSIFICATION OF DATA MODEL

Data Model is a collection of concepts to provide abstraction into DBMS so that superfluous details can be hide while highlighting important detail of data entities. It defines the logical design of data and establishes relationship between them. Data representation provides mechanisms to structure data for entities being modeled and

allow a set of operations to be performed on them. A number of Models has been developed which are further categorized as below

1. Object based Logical Model
2. Record based Logical Model
3. Physical Data Model

Depending on the concept they use to model the structure of the database, the data models are categorized. The following logical tree display detailed classification:

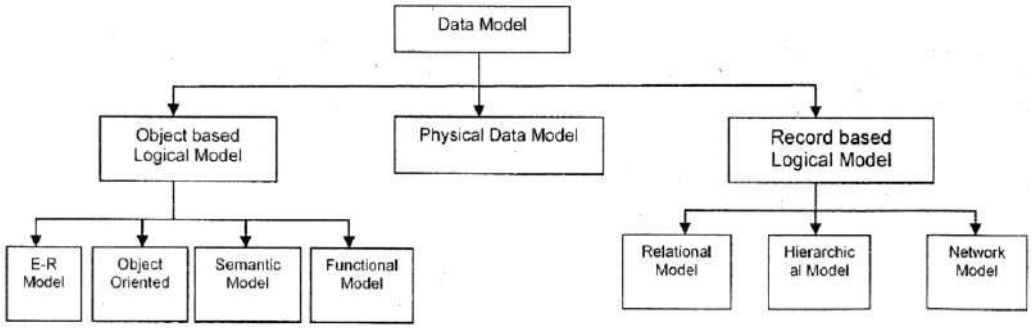


Fig. 3.1: Classification of Data Model

3.6 RECORD BASED MODELS

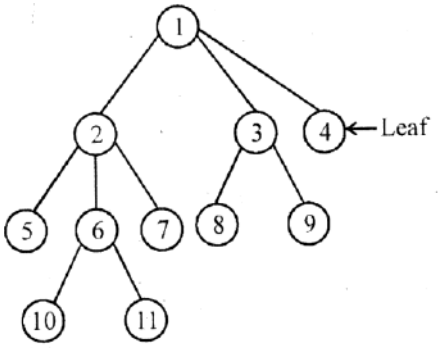
A record-based data models are used to specify the overall logical structures of the database. This model is used describing data at logical and view level. In the record based models, the database consists of a number of fixed-format records possibly of different types. Each record type defines a fixed number of fields, each typically of a fixed length. Data integrity constraints cannot be explicitly specified using record-based data models. There are three principle types of record-based data models:

3.6.1 Hierarchical Model

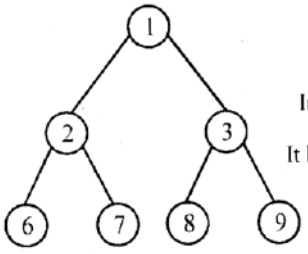
1. It was developed jointly by North American Rokwell Company and IBM.
2. It is the oldest model.
3. It follows tree as its basic structure.
4. Node at highest level is called Root.
5. A node may have any no. of children but each child node has only one parent.
6. Children of same parents are called siblings.

7. Anode that has no child is leaf node.

8. Representation of hierarchical model with suitable diagram

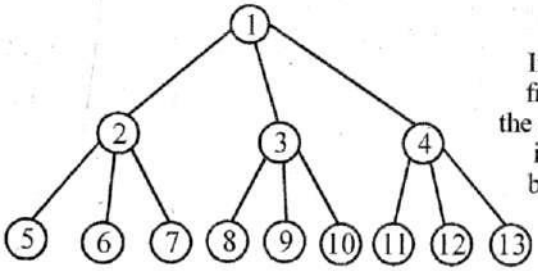


Leaf node is that node which has further no child. Here 5,10,11,7,8,9 & 4 are leaf nodes.



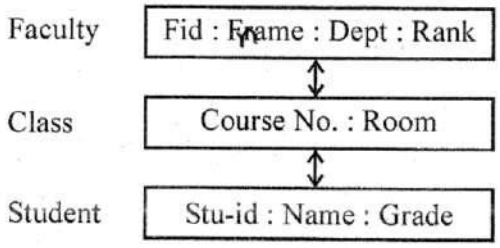
In binary tree, each node has not more than two child. It has exactly 2 child or 0 child.

Binary Tree

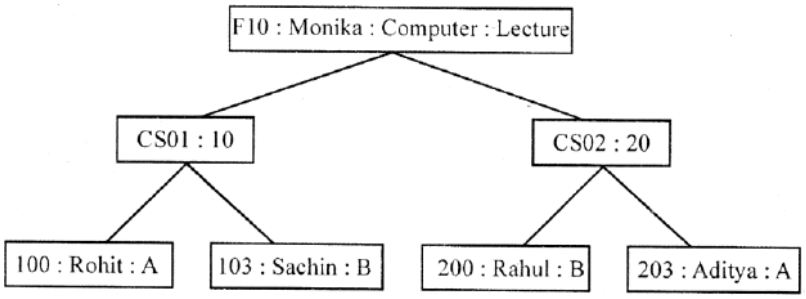


In balanced tree, every part from root node to a leaf has the same length. A balanced tree is always a binary tree but binary tree may or may not be a balanced tree.

Balanced Tree



- Faculty is on root node containing four attributes (Fid, Fname, Dept, Rank)
- Class is child of faculty node contains (Course no., room) attributes.
- There is one to many relationship between each faculty record and its class record.
- There is one to many relationship between each class record and its student record.



Operations on Hierarchical Model

(a) Insertion: A new class says CSO3 cannot be inserted unless some faculty is available at root level because without parent we can't insert any child node. This operation is used to insert a new record into the database. There are two possibilities:

(i) If the inserted, record is a root record then it creates new tree with the new record asthe root

(ii) If the inserted record is a child record, then we need to determine its parent first because no child record can exist without a parent record. So, insertion problem exists for the children who have no parents.

(b) Deletion: If we want to remove the class 100 then student Rohit will also have to be removed. This operation is used to delete a record from the database. To delete a record, we must first make it the current record of the database and then delete it. Here also, there are two possibilities:

(i) If the deleted record has no child node. It can be deleted easily.

(ii) If the deleted record has one or more child nodes, then the deletion process will deleteall the child nodes also. This may lead to loss of Information also.

(c) **Update:** If we want to updated the room 10, then we have to find all the recordsrelated to room 10 and have to modify. This operation is used to update a record. There are twopossibilities:

(i) If the record to be updated is a parent record, then updating it requires only one updation operation to be performed because there is only one occurrence of a parent record.

(ii) If the record to be updated is a child record, multiple updations may be required. If itnot happens, this may lead to inconsistency in the database.

(d) Record Retrieval: Recordretrieval methods for hierarchical model are complex and asymmetric. Retrieval means first searching the required record and then fetching it. Retrievalinvolves pointers from the parent node to the-child node in the tree and hence is complex andtime consuming.

Advantages of Hierarchical Model

1. **Simplicity:**The relationship between the various layers is logically simple. Thus the designof a hierarchical database is simple.
2. **Data Security:** Hierarchical model was the first database model that offered the data security that is provided and enforced by the DBMS.
3. **Data Integrity:** Hierarchical model is based on the parent/child there is always a link between the parent segment and the child segments under it. Thechild segments arealways automatically referred by its parents, so this model promotes data integrity.
4. **Efficiency:**The hierarchical database model is a very efficient one when the users require large number of transactions, using data whose relationships are fixed.

Disadvantages of Hierarchical Model

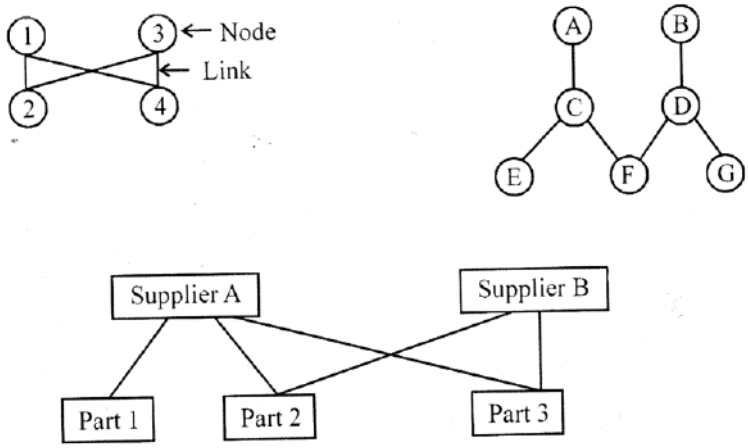
1. **Implementation Complexity:** Although the hierarchical databasemodel is conceptuallysimple and easy to design, it is quite complex to implement, the database designers shouldhave very good knowledge of the physical data storage characteristics.
2. **Lack ofStructural Independence:**Structural independence exists when the changes to the database structure does not affect the DBMS's ability to access data. Thus in a hierarchical database the benefits of data independence is limited by structural dependence.
3. **Programs Complexity:** Due to the structural dependence and the navigational structural, the application programs and the end users must know precisely how the data is distributed physically in the database in order to access data. This

requires knowledge of complex painter systems, which is often beyond the grasp of ordinary users.

4. **Operational Limitations:** Hierarchical model suffers from the Insert anomalies, anomalies and deletion anomalies, also the retrieval operation is complex and asymmetric, thus hierarchical model is not suitable for all the cases.
5. **Implementation Limitations:** Many of the common relationships do not confirm to the 1'N format required by the hierarchical model. The many-to-many (N;N) relationships, which are more common in real life are very difficult to implement in a hierarchical model.

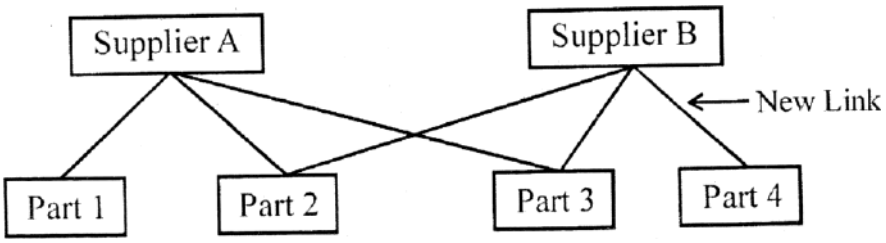
3.6.2 Network Model

1. Data in this model is represented by Links.
2. It looks like a tree structure containing nodes.
3. Every node may have one or more than one parent node.
4. Dependent node is called child node.

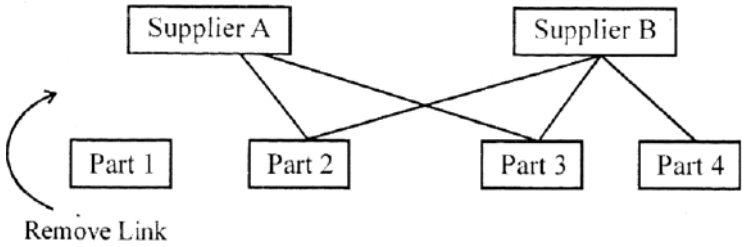


Operations on Network Model

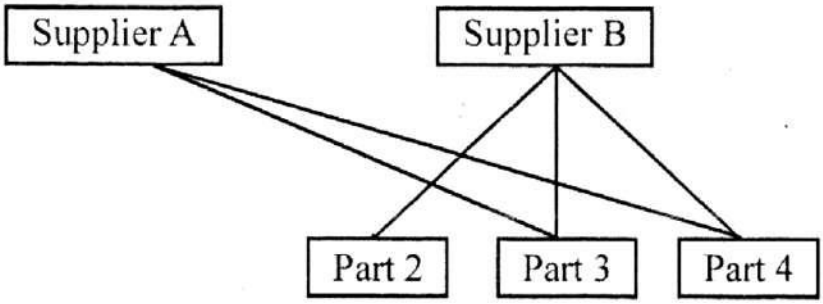
(a) **Insert Operation:** Insertion is easy i.e. supplier B supplies new part then we have to create a new link only. No other updation is required.



(b) **Delete Operation**: If we want to delete the information of any part, say supplier A doesn't want to supply part 1 now, so we have to remove only the link.



(c) **Updation Operation:** Updation is also easy. Suppose supplier A doesn't supply part 2 it supplies part 4 now.



(d) **Retrieval Operation:** Record retrieval method for network model aresymmetricbut complex.

Advantages of Network Model

1. **Conceptual Simplicity:** Like hierarchical model; the network model is also conceptually simple and easy to design.
2. **Capability to handle mass relationship types:** The network model can handle the one-to-many (1;N) and many to many (N:N) relationships, which is a real help in modeling the real life situations.

3. **Ease of data access:** The data access is easier than and flexible than the hierarchical model.
4. **Data Integrity:** The network model does not allow a member to exist without an owner. Thus a user must first define the owner record and then the member record. This ensures the data integrity.
5. **Data Independence:** The network model is better than the hierarchical model in isolating the programs from the complex physical storage details.
6. **Database Standard:** One of the major drawbacks of the hierarchical model was the nonavailability of universal standards for database design and modeling.

All the network database management systems conformed to these standards. These standards included a Data Definition Language [DDL] and the Data Manipulation Language [DML]$_9$ thus quality enhancing database administration and portability.

Disadvantages of Network Model

The network database model was significantly better than the hierarchical database model, it also had many drawbacks. These are

1. **System Complexity:** All the records are maintained using pointers and hence the whole database structure becomes very complex.
2. **Operational Anomalies:** Network model's insertion, deletion and updating operations of any record require large number of pointer adjustments, which makes its implementation very complex and complicated.
3. **Absence of Structural Independence:** If changes are mode to the database structure then all the application programs need to be modified before they can access data. Thus, every though the network database model succeeds in achieving data independence, it still fails to achieve structural independence.

Note: *We can conclude that network model does not suffers from the Insert anomalies, Update anomalies and Deletion anomalies. The retrieval operation is symmetric, as compared to hierarchical model, but the main disadvantage is complexity of the model.*

Hierarchical	Network

Each child node have only one parent.	Each child node may have more than one parent.
Hierarchical model records are organized as collection of trees.	Network model they are represented as arbitrary graphs.

3.6.3 Relational Model

1. It is primary data model for commercial data processing.The relational model was proposed by E.F.Codd of the IBM in 1972.
2. Relational model is a collection of tables. Tables are also known as relations.Therefore it is known as relational model.
3. Relational model represents the database as a collection of relations. Each relation (table)is a collection of row and columns.
4. Each table has a unique, name in database.
5. Columns are called attributes and rows are called tuples.
6. For each attribute there is a set of permitted values called domain.
7. Attribute name will be unique in a table.
8. Domain value can be NULL which shows that the value is unknown or does not exist.
9. The order of attribute has no significance. We can arrange attributes in any order.
10. We can insert record in any order.
11. Representation of data in Relational Model: A relational database consists of any number of relations. We can represent relation schemes by giving the name of the relation, followed by the attribute names in parenthesis.

Note: We will study Relational Model in detail in Unit-4

3.7 PHYSICAL MODEL

Physical model describes the in terms of a collection of files, indices, and other storage structures such as record formats, record ordering, and access paths. This model specifies how the database will be executed in a particular DBMS software such as Oracle, Sybase, etc., by taking into account the facilities and constraints of a given database management system. It also describes how the is stored on disk. Physical models are used for a higher-level description of storage structure and access mechanism.

They describe how data is stored in thecomputer, representinginformation such as record structures, record orderings and access paths. It is possible to implement the database at system level using physical data models. There are not as many physical data models so far. The most common physical data models are as follows:

- Unifying model
- Frame memory model.

3.8 OBJECT BASED MODELS

The object based models use the concepts of entities or objects and relationships among them. An entity is a distinct object (a person, place, concept, and 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 recordand a relationship is an association between 'entitles. It providesflexible structuring capabilities and allows data constraints to be specified explicitly. The object based logical model are classified as follows:

1. E-RModel
2. Object Oriented Model
3. Semantic Model
4. Functional Model

3.8.1 E-R Model (Entity Relationship Model)

E-R model is an effective and standard method of communication amongst different designers, programmers and end-users who tend to view data and its use in different ways. It is a non- technical method, which is free from ambiguities and provides a standard and a logical way of visualizing the data. It gives precise understanding of the nature of the data and how it is used by the enterprise. It provides useful concepts that allow the database designers to move from an informal description of what users want from their database, to a more detailed and precise description that can be implemented in a database management system. Thus, E-R modeling is an important technique for any database designer to master. It has found wide acceptance in database design. A basic

concept of E-R model has been introduced and few examples of E-R diagram of an enterprise database have been illustrated. The ER model is based on a concept of a real world entities and relationships among these entities. It can be used developed database design by allowing specification schema, which represents the overall logical structure of a database. It is very useful in mapping the meanings and interactions of real-world entities onto a conceptual schema.

1. E-R model is based on real world. It is a collection of basic objects, called entities and of relationships among these objects (Entity).
2. E-R model employs three basic features:
    - Entity
    - Attributes
    - Relationship
3. The overall logical structure of a database can be expressed graphically by an E-R diagram, which is built up by the following components:
    - Rectangle, which represent entity sets.
    - Ellipses, which represent attributes.
    - Diamonds, which represent relationships among entity sets.
    - Lines, which link attributes to entity sets and entity sets to relationships.

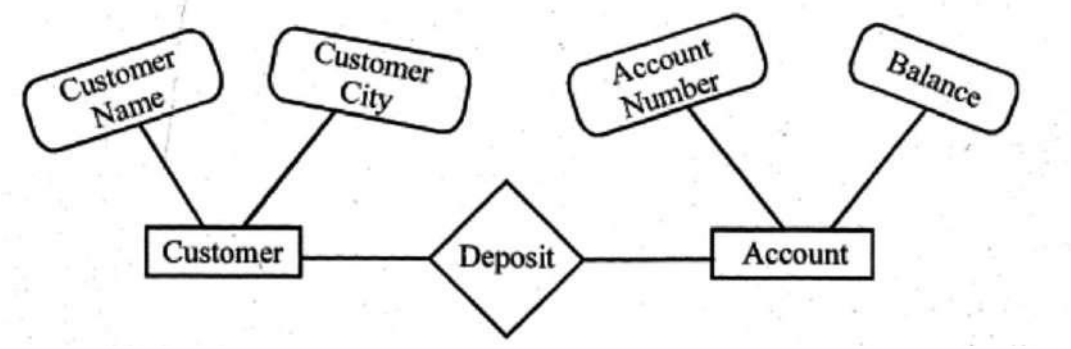


Fig. 3.2: E-R Model

Entity: Customer and Account are entity.

Attribute: Customer name, customer city are attributes of customer entity. Account number, balance are attributes of Account Entity.

Relation: Deposit is relationship among customer and Account.

I. ENTITY

- Entity is a thing which can be identified.
- Entity is a person, place thing event or concept which can be identified. We can say about which we want to store information i.e. employee, student, customer.
- Rectangle sign is used to represent the entity in E-R diagram.
- Entity can be of two types which are as follows:

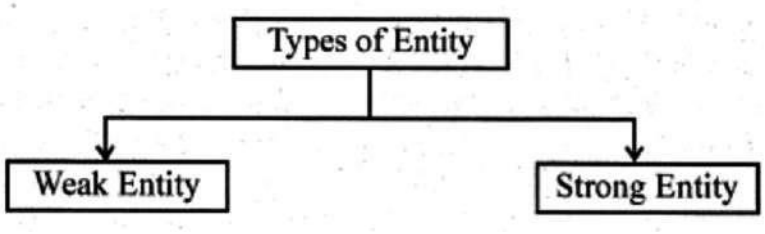
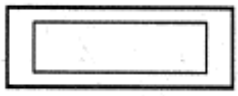
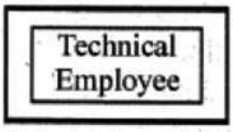


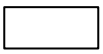
Fig. 3.3: Types of Entity

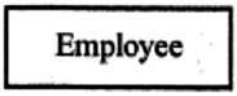
(a) Weak Entity:

- Weak entity depends on some other entity.
- It can't exist if other entity on which is depends does not exist.
- It is represented by 
- For example:



(b) Regular Entity/Strong Entity:

- Regular entity does not dependent on other entity.
- Its existence doesn't depend upon any other entity.
- It is represented by 
- For example:



(c) Entity Set: It is a set of entities of the same type that share the same properties (attributes) i.e. the set of all persons who are customer at same bank.

(d) Entity Subtype and Super type: Entity can be sub or super i.e. we have an Entity employee and employee can be programmer or operation.

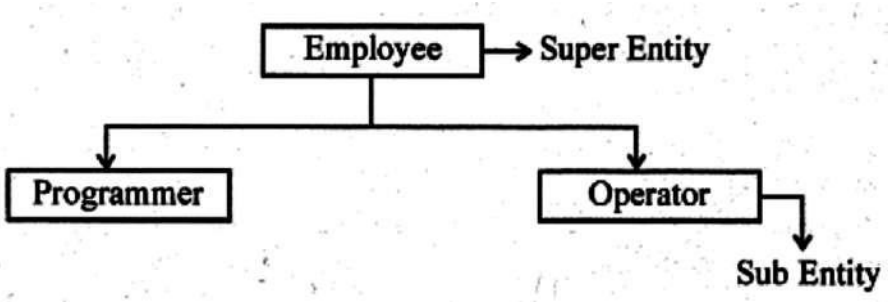
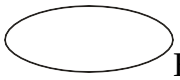


Fig. 3.4. Super Entity and Sub Entity

II. ATTRIBUTE

- Attributes are properties of entity. They are also called columns.
- Entity is about which we want to store information and attribute is what information we want to store.
- For example: If we want to store Name, City, and Salary information about employee. Then employee is our entity and name, city, salary are attributes.
-  Ellipse sign is used to represent attributes.
- Attribute can be of four types which are as follows:

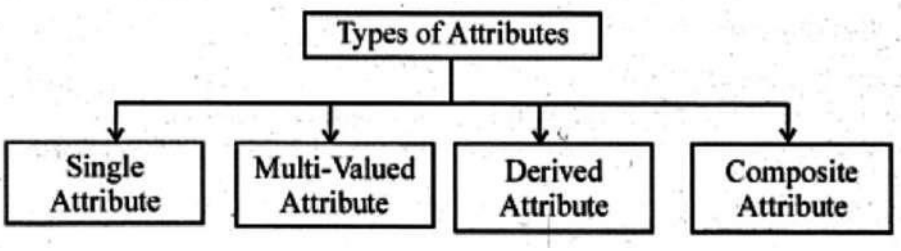
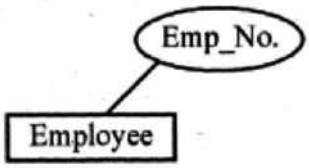
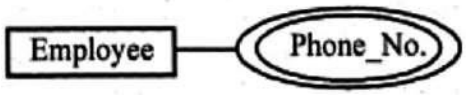


Fig. 3.5 :Types of Attribues

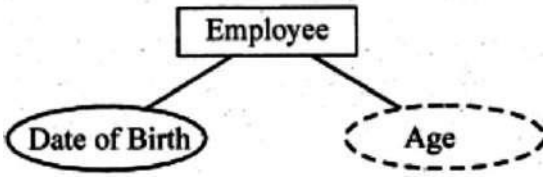
(a) **Single Attribute:** Single attributes are those attributes which can't be divided into sub parts i.e. Employee number is a simple attribute.



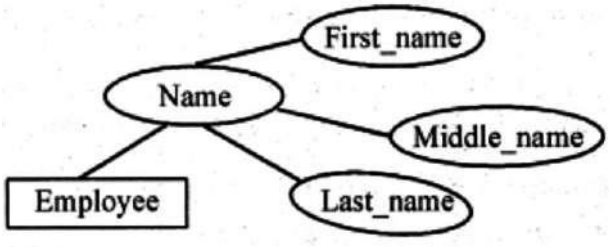
(b) **Multi Valued Attribute:** Multi valued attributes are those attribute which have more than one value i.e. Phone number is a multi-valued attribute. One employee may have more than one phone number.



(c) **Derived Attribute:** Derived attributes are those attributes whose value is derived from another attribute. For example: value of age attribute can be drive (calculate) from date of birth attribute and current date.



(d) **Composite Attribute:** Composite attributes are those attributes which can be divided into parts. For example: Name attribute can be divided into First Name, Middle Name and Last Name. Address attribute can be divided into street, city, state, zip-code.



III. RELATIONSHIP

- It is used to connect the entities.

- The entities involved in given relationship are called participants.
- The no. of participants in a given relationship is called degree of *relationship*.
-  sign is used to represent relationship among entities.
- Deposit is a relationship among entity customer and entity account.
- Relationship can be of four types which are as follows:

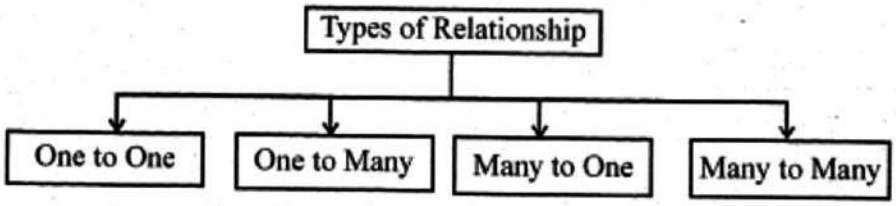
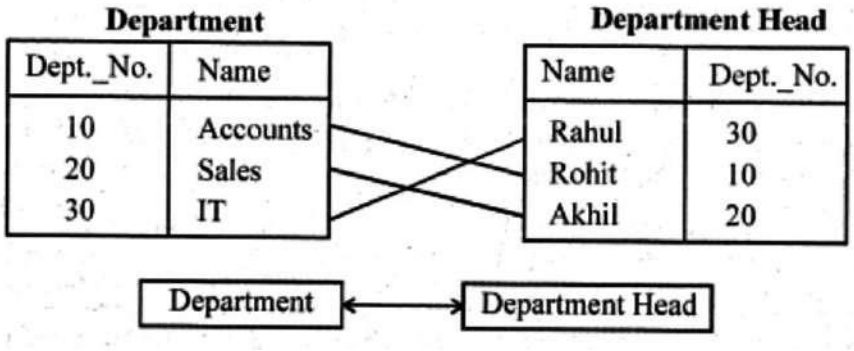
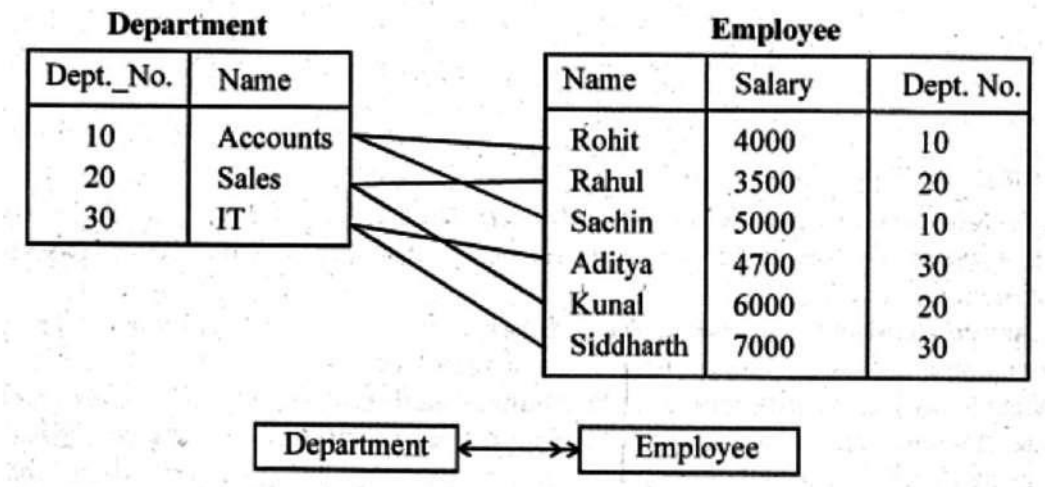


Fig. 4.5: Types of Relationship

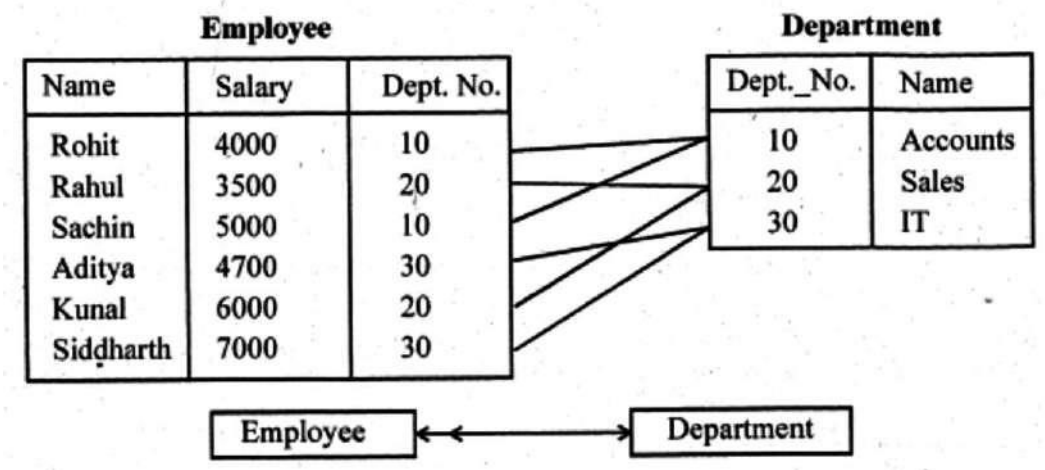
(a) One to One Relationship: In one to one relationship for one record in entity A, there is exactly one record in entity B. For example: we have two entities department and department head. There is one to one relationship because one department will be under one head and one head will be appointed for one department.



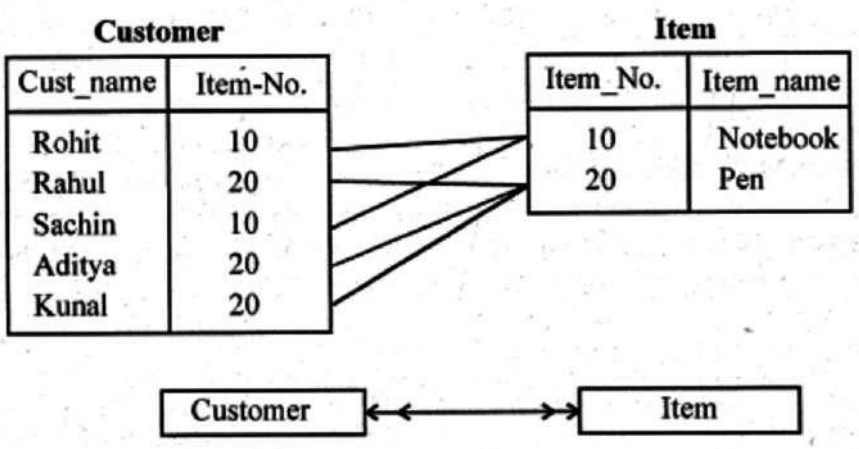
(b) One to Many Relationships: In one to many relationships for one record in entity A, there is more than one record in entity B. For example: We have two entities department and employee. There is one to many relationships because there will be one department in a company and more than one employee will work in that particular department.



(c) Many to One Relationship: In many to one relationship, for many records in entity A, there is only one record in entity B. For example: We have two entities employee and department. There is many to one relationship because there will be many employees in a single



(d) Many to Many Relationships: In many to many relationships, for many record is an entity A, there will be many record in entity B. There is many to many relationship because there will be many customers for many items.



Advantages of E-R Model

1. **Straight forward relation representation:** Having designed an E-R diagram for a database application, the relational representation of database model becomes relatively straight forward.
2. **Easy conversion for E-R over Data Model:** Conversion from E-R diagram to network or hierarchical data model can easily be accomplished.
3. **Graphical Representation for better understanding:** An E-R model gives graphical and diagrammatical representation of various entities, its attributes and relationship between entities. This helps in understanding the data structure in easy way, minimize the redundancy and other problem.

Disadvantages of E-R Model

1. Popular for high-level design: It is especially popular for high level design.
2. No Industry standard of Notation.

Difference between Strong Entity and Weak Entity

Sr. No.	Strong Entity	Weak Entity
1	It has an attribute having the capability that can act as primary key.	It does not have attribute which may act as a primary key.
2		

3.8.2 Object Oriented Model

The object oriented data model is an. adaptation of the object oriented programming language paradigm to database systems. The model is based on the concept of encapsulating data and code that operates on that data in an object. On the other hand, the object-relational data model is an., extension of relational data model. It combines the features of both the relational data model and object-oriented data model.

Object oriented data models for databases "extend the above mentioned data modeling features of the object oriented paradigm The extensions include data integrity constraints, persistence of data which allows transient data to be distinguished from persistent data and support for collections.

Advantages of Object-Oriented Data Model

1. **Capable of handling a large variety of data types:** hierarchical, network or relational), the object-oriented database are capable of storingdifferent types of data, for example, pictures, voices, video, including text, numbers and soon.
2. **Combining object-oriented programming with database technology:**Object-orienteddata model is capable of combining object-oriented programming with database technologyand thus, providing an Integrated application development system.
3. **Improved productivity:** Object-oriented data models provide powerful features such asinheritance, polymorphism and dynamic binding that allow the users to compose objects and provide solutions without writing object-specific code. These features increase-theproductivity of the database application developers significantly.
4. **Improved data access:** Object-oriented data model represents relationships explicitly,supporting both navigationaland associative access to information. It further improves the data access performance over relational-value-based relationships.

Disadvantages of Object-Oriented Data Model

1. **No precise definition:** It is difficult to provide a precise definition of what constitutes an object-oriented DBMS because the name has been applied to a variety of products and prototypes, some of which differ considerably from one another.
2. **Difficult to maintain:** The definition of objects is required to be changed periodically and migration of existing databases to confirm to the new object definition with change in organisational information needs. It possess real challenge when changing object definitions and migrating databases.
3. **Not suited for all applications:** Object-oriented data models are used where there is a need to manage complex relationships among data objects. They are especially suited for specific applications such as engineering, e-commerce, medicines and so on, and not for all applications. Its performance degrades and requires high processing requirements when used for ordinary applications.

3.8.3 Semantic Model

This model is used to express greater interdependencies among entities of interest. These independencies enable the models to represent the semantics of the data in the database. The Semantic Data Model (SDM), like other data models, is a way of structuring data to represent it in a logical way. SDM differs from other data models in that it focuses on providing more meaning of the data itself, rather than only on the relationships and attributes of the data.

SDM provides a high-level understanding of the data by abstracting it further away from the physical aspects of data storage.

In SDM, an entity represents some aspect or item in the real world, such as a student An entity is similar to a record in a relational system or an object in an object-oriented system. These entities in SDM focus on types, which are more general, instead of sets of data. In SDM, an entity is a very basic notion of a real-world or conceptual object that is defined by a single attribute.

For instance, an SDM entity type might be *person* which would be a list of names of people that are to be represented by the data. The objects in this domain would then point to specific instances of a person that are represented by each person entity.

3.8.4 Functional Model

The functional data model describes those aspects of a system concerned with transformations of values-functions, mappings, constraints and functional dependencies. The functional data model describesthe computations within a system.

- It shows how output value is derived from input values without regard for the orderwhich the values are computed. It also includes constraints among values.
- It consists of multiple data flow diagrams.
- Data flow diagrams show the dependencies between values and computation of output values from input values and functions, without regard for when the functions are executed.
- Traditional computation concepts such as expression trees are examples of functional models.

3.9 COMPARISON OF DATA MODELS

Sr. No.	Hierarchical Model	Network Data Model	Relational Data Model
1.	Hierarchical data model represents data in a tree format where Parent and Child relationship is represented to show association.	Network model represents data in graphs where data is a record which is linked by pointers.	Relational data model logically represents data in Tabular form where data is placed in row and column.
2.	Many to many relationship cannot be expressed in hierarchical model	Many to many relationship can be expressed in hierarchical model.	Many to many relationship can be expressed in hierarchical model.
3.	It is good for expressing data in < parent child relationship	It is good for modelling of many to many relationship.	It is good for modelling real world entities.

4	Relationship are represented by pointer and relationship among records are physical in nature	Network model also represents relationship through pointers and nature of the relationship is physical.	Relational model is stored data in form of rows and column. There is no physical connection is established between different tables whereas connection is logical in nature and established through keys.
5	Searching of a particular record is a time consuming task as to reach a particular child we have to process through its parent record.	Searching of a particular record is easy since there are multiple access path available to reach a node in graph.	In case of relations tables we use concept of keys to identify the records and search a key through indexing is quite simple task
6.	Insertion is done in the form of parent node and child node relationship. We cannot insert child node in tree without parent node.	Network model insertion can be performed by inserting new node in the graph with ease and has no insertion anomaly	in Relation model , new record can be added any time and has no insertion anomaly
7.	Updation operation may results in inconsistency as there are multiple child records in a tree	Updation operation is free from any anomaly as there is only single occurrence of each record in a graph which may be connected with multiple records.	Updation operation is safe in a relational model as duplication of record is avoidable by applying normalisation and Primary keys relationships
8.	Hierarchical model is based on parent child relationship and deleting of child is easy as compare to parent, if we delete parent then child node will automatically deleted from the tree.	There is no deletion anomaly as deleting of one node does not affect other nodes due to many to many relationships.	The deleting of record from a relation is again a simple process and there is no anomaly related to deleting of records. Deletion of reference records is not allowed as it may linked to other records

3.10. OTHER TERMS USED IN E-R MODEL

CONSTRAINTS

Relationship types usually have certain constraints that limit the possible combinations of entities that may participate in the corresponding relationship set. The constraints should reflect the restrictions on the relationships as perceived in the 'real

world'. For example, there could be a requirement that each department in the entity DEPT must have a person and each person in the PERSON entity must have a skill. The main types of constraints on relationships are multiplicity, cardinality, participation and so on.

1. **Multiplicity Constraints:** Multiplicity is the number (or range) of possible occurrences of an entity type that may relate to a single occurrence of an associated entity type through a particular relationship. It constrains the way that entities are related. It is a representation of the policies and business rules established by the enterprise or the user. It is important that all appropriate enterprise constraints are identified and represented while modeling an enterprise.

2. **Cardinality Constraints:** A cardinality constraint specifies the number of instances of one entity that can (or must) be associated with each instance of entity. There are two types of cardinality constraints namely minimum and maximum cardinality constraints. The minimum cardinality constraint of a relationship is the minimum number of instances of an entity that may be associated with each instance of another entity. The maximum cardinality constraint of a relationship is the maximum number of instances of one entity that may be associated with a single occurrence of another entity.

3. **Participation Constraints:** The participation constraint specifies whether the existence of an entity depends on its being related to another entity via the relationship type. There are two types of participation constraints namely total and partial participation constraints. Total participation constraints means that every entity in 'the total set' of an entity must be related to another entity via a relationship. Total participation is also called existence dependency. A partial participation constraint means that some or the 'part of the set of an entity are related to another entity via a relationship, but not necessarily all. The cardinality ratio and participation constraints are together known as the structural constraints of a relationship type.

4. **Exclusion and Uniqueness Constraints:** E-R modeling has also constraints such exclusion constraint and uniqueness constraint that results into poor semantic base and tries to make entity-attribute decisions early in the conceptual

modeling process. In exclusion constraint the normal or default treatment of multiple relationships is inclusive OR, which allows any or all of the entities to participate. In some situations, however, multiple relationships may be affected by the exclusive (disjoint or exclusive OR) constraint, which allows at most one entity instance among several entity types to participate in the relationship with a single root entity.

GENERALIZATION

1. A generalization hierarchy is a form of abstraction that specify the two or more entities that share the common attributes can be generalized into a higher-level entity type called a super type or generic entity.
2. The lower level of entities becomes the subtype. Subtypes are dependent entities.

SPECIALIZATION

1. Specialization is a process of taking subsets of higher level entity set to form lower level entity sets.
2. It is a process of defining a set of subclasses of an entity type which is called as super class of the specialization.

AGGREGATION

1. One limitation of the E-R model is that it cannot express relationship among relationships.
2. Aggregation is the process of compiling information on an object, thereby abstracting a higher level object.
3. Aggregation allows us to indicate that a relationship set participate in another relationship set.

Questions

1. What do you mean[1] by data models? Explain the answer.
2. How can we classify data models?
3. What do mean by relationships in a data model
4. What is an attribute in data modeling?

5. Explain the Relational Model? Write advantages and disadvantages.
6. Explain the Hierarchical Model?Write advantages and disadvantages.
7. Explain different operations that can be performed on Hierarchical
8. Compare different data models.
9 Define the following terms:
   (a) Entity Set
   (b) Attribute
   (c) Relationship Set
   (d) Simple attributes
   (e) Composite attributes
   (f) Multivalve attributes.
10 What are the different types of attributes? Explain using examples.
11 What are mapping constraints? What are its types?
12 What are weak entity sets? Why are they used?
13 What is generalization? Explain with a suitable example.
14 What is aggregation? Explain using a suitable example.
15 The E-R Diagram for an Employee Payroll System.
16 Discuss the advantages and disadvantages of ER model.
17 The E-R Diagram for Book Purchasing System..
18 Explain with diagrammatical illustrations about the different types of relationships

COURSE: DBMS

UNIT 4 : RELATIONAL DATA MODEL

4.1 RELATIONAL MODEL

4.2 COMPARISON OF DATA MODELS

4.3 RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

4.4 RELATIONAL ALGEBRA

4.5 RELATIONAL CALCULUS

4.6 DIFFERENCE BETWEEN RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

4.1 RELATIONAL MODEL

1. It is primary data model for commercial data processing.The relational model was proposed by E.F.Codd of the IBM in 1972.
2. Relational model is a collection of tables. Tables are also known as relations.Therefore it is known as relational model.
3. Relational model represents the database as a collection of relations. Each relation (table)is a collection of row and columns.
4. Each table has a unique, name in database.
5. Columns are called attributes and rows are called tuples.
6. For each attribute there is a set of permitted values called domain.
7. Attribute name will be unique in a table.
8. Domain value can be NULL which shows that the value is unknown or does not exist.
9. The order of attribute has no significance. We can arrange attributes in any order.
10. We can insert record in any order.
11. Representation of data in Relational Model: A relational database consists of any number of relations. We can represent relation schemes by giving the name of the relation, followed by the attribute names in parenthesis.

12. ***Components of Relational Model***

(1) Data Structure

(2) Data Integrity

(3) Integrity Constraints

1. **Data Structure**

(a) **Relation**

- All data is represented in table.
- Table contains rows and columns. Table isalso known as Relation.
- Columns are called attributed and rows are called tuples. Each cell of relation contains only single value.
- Relation contains information on one subject only.

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Pune	1225
7.	Kunal	8^{th}	Hyderabad	806

Student is the title of the relation. There are 5 column (S.No, Name, Class, City, Roll Number) and 7 rows.

(b) Attribute

- Columns-are called attributes.
- Attributes appear vertically in a relation.
- Attributes can appear In any order and provide specific information.
- Attributes.in a relation "Student" are S.No., Name, Class, City, Roll Number.

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Pune	1225
7.	Kunal	8^{th}	Hyderabad	806

(c) Tuple

- Rows are called tuples.
- Tuples appear horizontally in a relation.
- Tuples can appear in any order and provide complete information (full record).
- There are 7 tuples in a relation "Student".

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Pune	1225
7.	Kunal	8^{th}	Hyderabad	806

(d) Domain

- Domain is a set of all allowed or possible values for an attribute in a relation.

- Domain specifies the type of data used in an attribute.
- For example, in a relation student, students are from different cities. In this case, city is an attribute of relation student and Mumbai, Patiala, Gurgaon etc. are domain values of city attribute.

(e) Degree

- Degree is the number of attributes in a relation.
- If a relation have only one attribute, then its degree is one and known as called unary relation
- If a relation have two attributes, then its degree is two and known as called binary relation
- If a relation have three attributes, then its degree is three and known as called ternary and so on.
- Degree of relation "Student" is 5.

(f) Cardinality

- Cardinality is the number of tuples in a relation.
- Cardinality changes on the basis of insertions or deletions of records in a relation.
- Cardinality of relation "Student" is 7

2. Data Integrity: Data integrity ensures the accuracy of data. For this purpose, we should know about the keys.

"A key is a single attribute or a combination of two or more attributes of a relation. It is used to identify one or more instance of the set."

Types of Keys: There are six types of keys which are as follows:

(a) Candidate key:

- Candidate keys are those attributes which have unique values. But Null value is not allowed in a candidate key.
- If there is no attribute in a relation containing unique value then combination of two attributes of that relation can make candidate key.

- There can be any number of candidate keys in a relation (table).

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Mumbai	1225
7.	Kunal	8^{th}	Hyderabad	806
8.	Rohit	10^{th}	Chennai	1011

In the above relation "Student", S. No. and Roll Number both attributes have unique value, therefore they are candidate keys.

(b) Primary Key

- The attribute which have unique value is known as primary key. But Null value is notallowed in a primary key.
- Primary key is used for query purposes.
- There will be only one primary keys in a relation (table).

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Pune	1225
7.	Kunal	8^{th}	Hyderabad	806
8.	Rohit	10^{th}	Chennai	1011

In the above relation "Student", only one attribute Roll Number have unique value, therefore attribute Roll Number is a primary key.

(c) Alternate Key

- Alternate key also contains unique value.
- After identifying candidate keys, one key is known as primary key and another key (which is not selected as primary key) is known as alternate key.

Student

S. NO.	Name	Class	City	Roll Number
1.	Rohit	10^{th}	Chennai	1010
2.	Akhil	12^{th}	Mumbai	1229
3.	Aditya	8^{th}	Bangalore	801
4.	Sachin	9^{th}	Gurgaon	906
5.	Rahul	11^{th}	Patiala	1112
6.	Sid	12^{th}	Pune	1225
7.	Kunal	8^{th}	Hyderabad	806
8.	Rohit	10^{th}	Chennai	1011

In the above relation "Student", S. No. and Roll number both have unique values and called as candidate keys. S.No. is called alternate key and Roll number is known as primary key.

If Primary key: Roll Number

Then Alternate key: S. No.

(d) Composite Key

- Sometimes in a relation, there is no primary key. In that situation, more than one attributes are used to identify a unique entity.

- The combination of those attributes are known as composite key.

Name	Class	City	Age
Rohit	10^{th}	Chennai	16
Akhil	12^{th}	Mumbai	18
Aditya	8^{th}	Bangalore	14
Sachin	9^{th}	Gurgaon	15
Rahul	11^{th}	Patiala	17
Sid	12^{th}	Pune	18
Kunal	8^{th}	Hyderabad	14
Rohit	10^{th}	Chennai	17

In the above relation "Student‖, Name and Age are used to identify an entity, therefore both are called composite key.

(e) Artificial Key: Sometimes in a relations, there is no primary key and there is no possibility to make primary key.In that situation, we can insert a key in a relation which has no meaning is known as an artificial key.

Student

Name	Class	City	Age
Rohit	10^{th}	Chennai	16
Akhil	12^{th}	Mumbai	18
Aditya	8^{th}	Bangalore	14
Sachin	9^{th}	Gurgaon	15
Rahul	11^{th}	Patiala	17
Sid	12^{th}	Pune	18
Kunal	8^{th}	Hyderabad	14
Rohit	10^{th}	Chennai	17

In the above relation "Student", there is no unique key. We can insert a new attribute S.No. into relation as a artificial key. This attribute S. No. has no meaning.

Student

S. No.	Name	Class	City	Age
1	Rohit	10^{th}	Chennai	16
2	Akhil	12^{th}	Mumbai	18
3	Aditya	8^{th}	Bangalore	14
4	Sachin	9^{th}	Gurgaon	15
5	Rahul	11^{th}	Patiala	17
6	Sid	12^{th}	Pune	18
7	Kunal	8^{th}	Hyderabad	14
8	Rohit	10^{th}	Chennai	17

(f) Foreign Key

- Foreign key is the attribute of a relation which acts as a primary key of another table.
- Foreign key .allows only those values which appears in primary key or may be null.
- Foreign key is used to make a relationship between two tables and to maintain thereferential integrity.

Class

Class	Class Incharge
8^{th}	Mrs. Nidhi
9^{th}	Ms. Aastha
10^{th}	Mrs. Prathiba
11^{th}	Mrs. Manmeet
12^{th}	Mrs. Navreet

Student

S. No.	Name	Class	City	Roll Number
1	Rohit	10^{th}	Chennai	1010
2	Akhil	12^{th}	Mumbai	1229
3	Aditya	8^{th}	Bangalore	801
4	Sachin	9^{th}	Gurgaon	906
5	Rahul	11^{th}	Patiala	1112
6	Sid	12^{th}	Pune	1225
7	Kunal	8^{th}	Hyderabad	806
				1011

In the above relation, class acts as an foreign key.

3. Relational Model Constraints/Integrity Constraints

- Integrity constraints ensure that changes made to the database by authorized users and any change do not lose the data.
- Integrity constraints also ensure the restrictions on the data and provide the security against the accidental damage to the database.

Type of Constraints

(a) Domain Constraint:

- It ensures that each attribute have a correct value.
- The data type associated with domains includes integer, character, string, data, and time.
- For example: A is not allowed in the attribute Roll Number because Roll Number is an integer attribute.

S. No.	Name	Class	City	Roll Number
1	Rohit	10^{th}	Chennai	1010
2	Akhil	12^{th}	Mumbai	1229
3	Aditya	8^{th}	Bangalore	801
4	Sachin	9^{th}	Gurgaon	906
5	Rahul	11^{th}	Patiala	1112

6	Sid	12^{th}	Pune	1225
7	Kunal	8^{th}	Hyderabad	806
8	Rohit	10^{th}	Chennai	1011

(b) Tuple Uniqueness Constraint

- Relation is a set of tuples (rows).
- All tuples in a relation must be different from each other. It means there must be unique value or attribute by which we can identify a tuple.

(c) Key Constraint

- Primary key must have unique value in the relation (table).
- If S.No. is considered as a primary key then there must be unique value in this attribute. We cannot insert duplicate value in the primary key.

S. No.	Name	Class	City	Roll Number
1	Rohit	10^{th}	Chennai	1010
2	Akhil	12^{th}	Mumbai	1229
3	Aditya	8^{th}	Bangalore	801
4	Sachin	9^{th}	Gurgaon	906
5	Rahul	11^{th}	Patiala	1112
5/3	Sid	12^{th}	Pune	1225
7	Kunal	8^{th}	Hyderabad	806
7/4	Rohit	10^{th}	Chennai	1011

(d) Entity Integrity: Entity integrity ensures that primary key cannot have NULL value.

S. No.	Name	Class	City	Roll Number
1	Rohit	10^{th}	Chennai	1010
2	Akhil	12^{th}	Mumbai	1229
3	Aditya	8^{th}	Bangalore	801
4	Sachin	9^{th}	Gurgaon	906
5	Rahul	11^{th}	Patiala	1112

	Sid	12th	Pune	1225
7	Kunal	8th	Hyderabad	806
	Rohit	10th	Chennai	1011

(e) Referential Integrity: Referential integrity ensures that if a foreign key of a table I refers to the primary key of table II, then every value of the foreign key in table I must be null or be available in table II.

I	
Class	**Class Incharge**
8th	Mrs. Nidhi
9th	Ms. Aastha
10th	Mrs. Prathiba
11th	Mrs. Manmeet
12th	Mrs. Navreet

II				
S. No.	**Name**	**Class**	**City**	**Roll Number**
1	Rohit	10th	Chennai	1010
2	Akhil	12th	Mumbai	1229
3	Aditya	8th	Bangalore	801
4	Sachin	9th	Gurgaon	906
5	Rahul	11th	Patiala	1112
6	Sid	12th	Pune	1225
7	Kunal	8th	Hyderabad	806
8	Rohit	10th	Chennai	1011

Operations of Relational Model

1. **Insert Operation:** Relational model does not suffer from any insert anomaly.
2. **Update Operation:** Relational model does not suffer from any update anomaly.

3. **Delete Operation:**Relational model does not suffer from any delete anomaly.
4. **Retrieve Operation:** Retrieve operation for relational data model is simple and symmetric.

Advantages of Relational Data Model

1. **Simplicity:** A relational data model is even simpler than hierarchical and network models. It frees the designers from the actual physical data storage details, thereby allowing them to concentrate on the logical view of the database.
2. **Structural Independence:** Unlike hierarchical and network models, the relational data model does not depend on the navigational data access system. Changes in the database structure do not affect the data access. Ease of design, implementation, maintenance and uses: The relational model provides bothstructural independence and data independence. Therefore, itmakes the database design, implementation, maintenance and usage much easier.
3. **Flexible and Powerful Query Capability:** Its structured query capability makesad hoc queries a reality.The relational database model provides very powerful, flexible, and easy-to-use query facilities. Information in a table can be easily modified.
4. **Easy to Use:** To collect the information in table consisting columns and rows is very easy.
5. **Security:** In relational model, security control and authorization can be implemented.

Disadvantages of Relational Data Model

1. **Hardware overheads:** The relational data models need more powerful computing hardware and data storage devices to perform complex tasks. Consequently, they tend to be slower than the other database systems. However, with rapid advancement in computing technology and development of much more efficient operating systems, the disadvantage of being slow is getting faded.
2. **Easy-to-design capability leading to bad design:** Easy-to-use feature of relational database results into untrained people generating queries and reports

without much understanding and giving much thought to the need of proper database design. With the growth of database, the poor design results into slower system, degraded performance and data corruption.

4.2 COMPARISON OF DATA MODELS

Sr. No.	Hierarchical Model	Network Data Model	Relational Data Model
1.	Hierarchical data model represents data in a tree format where Parent and Child relationship is represented to show association.	Network model represents data in graphs where data is a record which is linked by pointers.	Relational data model logically represents data in Tabular form where data is placed in row and column.
2.	Many to many relationship cannot be expressed in hierarchical model	Many to many relationship can be expressed in hierarchical model.	Many to many relationship can be expressed in hierarchical model.
3.	It is good for expressing data in < parent child relationship	It is good for modeling of many to many relationships.	It is good for modeling real world entities.
4	Relationship are represented by pointer and relationship among records are physical in nature	Network model also represents relationship through pointers and nature of the relationship is physical.	Relational model is stored data in form of rows and column. There is no physical connection is established between different tables whereas connection is logical in nature and established through keys.
5	Searching of a particular record is a time consuming task as to reach a particular child we have to process through its parent record.	Searching of a particular record is easy since there are multiple access path available to reach a node in graph.	In case of relations tables we use concept of keys to identify the records and search a key through indexing is quite simple task
6.	Insertion is done in the form of parent node and child node relationship. We cannot insert child node in tree without parent node.	Network model insertion can be performed by inserting new node in the graph with ease and has no insertion anomaly	in Relation model , new record can be added any time and has no insertion anomaly

7.	Updation operation may results in inconsistency as there are multiple child records in a tree	Updation operation is free from any anomaly as there is only single occurrence of each record in a graph which may be connected with multiple records.	Updation operation is safe in a relational model as duplication of record is avoidable by applying normalisation and Primary keys relationships
8.	Hierarchical model is based on parent child relationship and deleting of child is easy as compare to parent, if we delete parent then child node will automatically deleted from the tree.	There is no deletion anomaly as deleting of one node does not affect other nodes due to many to many relationships.	The deleting of record from a relation is again a simple process and there is no anomaly related to deleting of records. Deletion of reference records is not allowed as it may linked to other records

4.3 RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

The relational model uses the concept of a mathematical relation in the form of table of values which acts as building block. The table is a logical representation of data in the form of rows and columns. The relational algebra is a formal query language applied on relational model. It is a procedural language which specifies the operations to be performed on relations. The operations are performed in form of sequence of algebra operations which results in a new relation/table. The relational algebra operations can be classified into two types.

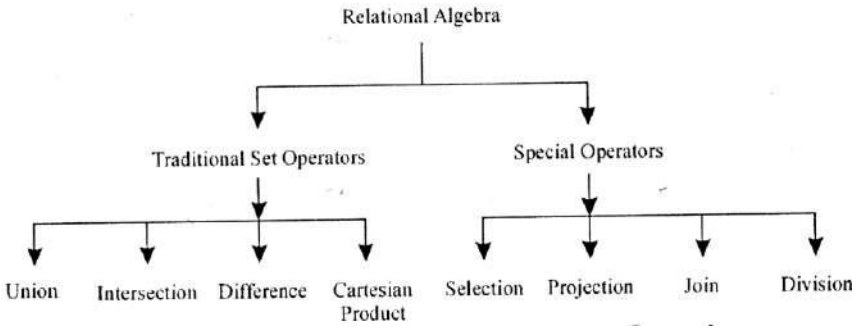


Figure 3.1: Classification of Relational Algebra Operations

Relational calculus is a non-procedural query language. Here, no procedures are provided to generate result based on 'query'. In relational calculus, query is expressed as

variables and formulas on these variables. There are two types of relational calculus: tuple Relational Calculus and Domain Relational Calculus:

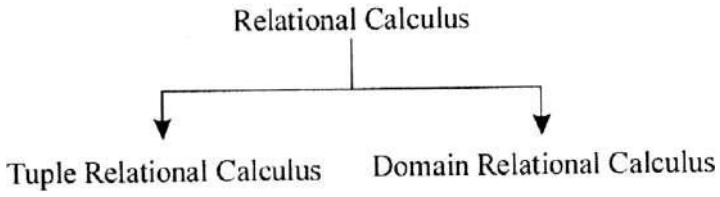


Figure 3.2: Classification of Relational Calculus

4.4 RELATIONAL ALGEBRA

1. Relational algebra is a procedural query language.
2. It consists of set of operators that take one or two relations as input and produce a new relation as output.
3. It uses relational operators.
4. It is of mainly two types which are as follows:

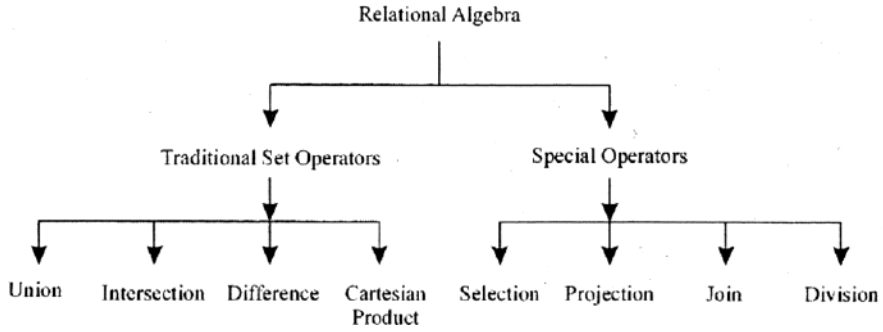


Figure 3.3: Classification of Relational Algebra

I. Traditional Set Operators

(a)Union Operator

(b)Intersection Operator

(c)Difference Operator

(d)Cartesian Product Operator

(a) Union Operator:

- Union of two relations is the set of all elements belonging to both relations.
- Result must not contain duplicate elements.
- It is denoted by U.

- For example: We want to list all the names and roll numbers which are present in both tables:'A' and 'B'.

AB

Name	Roll Number
Akhil	211
Monika	129

Name	Roll Number
Aastha	112
Akhil	211

Formula:πName, Roll Number (A) U πName, Roll Number (B).

AUB

Name	**Roll Number**
Akhil	211
Monika	129
Aastha	112

(b) Intersection Operator:

- Intersection of two relations produces a relation which contains all elements that are common to both relations.
- It is denoted by ∩.
- For example: We want to list only those names and roll numbers which are common inboth tables 'A' and 'B'.

A

Name	Roll Number
Akhil	211
Monika	129

B

Name	Roll Number
Aastha	112
Akhil	211

Formula:πName, Roll Number (A) ∩πName, Roll Number (B)

A∩B

Name	**Roll Number**
Akhil	211

(c) Difference Operator

- Difference operator is used to find those tuples which are present in one relation but not in another relation.
- It is denoted by (-) sign.
- For example: We want to list those names and roll numbers which are present in table 'A^9 only, not in table'B'.

AB

Name	Roll Number
Akhil	211
Monika	129

Name	Roll Number
Aastha	112
Akhil	211

Formula:πName, Roll Number (A) –πName, Roll Number (B)

A-BB-A

Name	Roll Number
Monika	129

Name	Roll Number
Aastha	112

(d) Cartesian Product

- Cartesian product operator is used to combine information from any two relations.
- It is denoted by (X) symbol.
- For example: We want to list the names of employees with all departments of tables 'A'and'B'.

AB

Name	**Emp_No**	**Dept_Id**
Akhil	101	11
Monika	102	12
Aastha	101	11

Dept_Name	**Dept_Id**
Production	11
Accounts	12

Formula: πName (A) X π Dept_Name (B)

AXB

Name	Dept_Name
Akhil	Production
Akhil	Accounts
Monika	Accounts
Monika	Production
Aastha	Production
Aastha	Accounts

II. Special Operators

(a) Selection Operator

(b) Projection Operator

(c) Join Operator

(d) Division Operator

(a) Selection Operator

- Selection operator selects tuples (rows) that satisfy a given condition.
- It is denoted by lower Greek letter sigma (a).
- We can also use folio wing symbols: = >,<>>=,<= #
- For example: We want to list the tuples (employees) who live in city 'chd'.

Formula: σcity = "chd" (employee)

(b) Projection Operator

- Projection operator returns a new relation as output with certain attributes.
- It is denoted by Greek letter pie (π).
- For example: We want to list all the emp_no and name of employee.

Formula: πemp_no, name (employee)

(c) Join Operator

- Join operator is also known as natural join operator.
- It is denoted by the symbol ([><]).

- Cartesian product operator is used to combine two tables, but the output of Cartesianproduct is not correct
- Join operator is used to combine the two tables instead of Cartesian product operator.
- For example: We want to combine the two tables 'A'and 'B'.

A

Name	Emp_No	Dept_Id
Akhil	101	11
Monika	102	12
Aastha	101	11

B

Dept_Name	Dept_Id
Production	11
Accounts	12

Formula : π Name, Dept_Id (A) ($\triangleright\!\!\triangleleft$) π Dept_Name, Dept_Id (B)

A $\triangleright\!\!\triangleleft$ B

Name	Dept_Name
Akhil	Production
Monika	Accounts
Aastha	Production

(d) Division Operator

- Division operator will work on two relations (tables).
- It make another relation consisting of values of an attribute of one relation that match all the values in the another relation.
- It is denoted by the (÷) symbol.

A

Branch_Name	Branch_Id
Chd	11
Delhi	12
Mumbai	13

B

Branch_Name	Branch_Id
Akhil	Delhi
Monika	Chd
Aastha	Mumbai
Ankush	Delhi
Radhika	Chd

Formula: πName (A ÷B)

(A ÷B)

Name
Akhil

4.5 RELATIONAL CALCULUS

1. It was first proposed by E.F.Codd.
2. It is a formal language used to symbolize logical arguments in mathematics.

3. In relational calculus, query is expressed as formula containing number of variables and expression.
4. User will only tell the requirement without knowing the methods of retrieval.
5. User is not concerned with the procedure to obtain the results.
6. It is the responsibility of DBMS to transform these queries and give the result to the user.
7. Relational calculus is of mainly two types which are as follows:

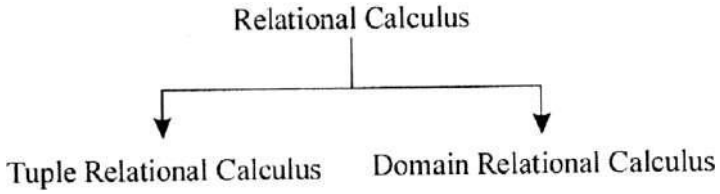


Figure 3.4: Classification of Relation Calculus

I. Tuple Oriented Relational Calculus

- It is based on specifying a number of tuples variables.
- The query of tuple relational calculus is

  {t/COND(t)}

  t-> is tuple variable

  COND (t)->is conditional expression.
- The result of such query is a relation that contains all the types (rows) that satisfy condition COND (t).

  Query of relational calculus is:

  {t. title, t. author/Book(t) and t. PRICE > 100}

  It will give us title, author of all the books whose price is greater than 100.

  Expression of tuple relational calculus is:

  {t1. A1, t2.A2, t3.t3,…tn.An/COND (t1, t2, t3, …tn)}

  t1, t2 …. are tuple variables.

  A1, A2 … are the attributes of relations.

  COND is condition.

II. Domain oriented relational calculus

- Domain calculus is different from tuple calculus in the type of variables used in formula.
- In domain oriented relational calculus, variable range will be single value rather than multiple values.
- Expression of domain oriented relational calculus is:

  {X1, X2, …Xn | COND (X1, X2, … Xn)}

  X1, X2, …Xn are domain variables.

  COND is condition or formula of domain relation calculus.

  i.e. Get employee no. of for job clerk

  EX where EMP (emp no: EX, job = 'clerk')

  Get employee name that belongs to dept no. 10 and having salary > 2000.

  Ex where EMP (ename: EX, deptno = 10, sal> 2000)

4.6 DIFFERENCE BETWEEN RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

Sr. No.	Relational Algebra	Relational Calculus
1	It is a procedural method of solving the queries.	It is a non-procedural method of solving the queries.
2	It is used as a vehicle for implementation of relational calculus.	The queries of relational calculus are transformed into equivalent relational algebra format and then implemented with the help of relational algebra operators.
3	The solution to the database access problem using a relational algebra is obtained by stating what is required? And what are the steps to obtain that / information?	The solution to the database access problem using a relational calculus is obtained by stating what is required? And system will find the answer?

Questions

1. What do you mean[1] by data models? Explain the answer.
2. How can we classify data models?
3. What do mean by relationships in a data model
4. What is an attribute in data modelling?
5. Explain the Relational Model? Write advantages and disadvantages.
6. Explain the Hierarchical Model?Write advantages and disadvantages.
7. Explain different operations that can be performed on Hierarchical
8. Compare different data models.
9. List the various relational operators available in a relational model.
10. What is the difference between select and project operators?
11. Explain the various set operators available in relational algebra.
12. What is the Cartesian product operation? Why is it rarely used without a select operation?
13. What is the significance of the join operator? Explain the different types of join.
14. Explain relational calculus in detail.

COURSE: DBMS

UNIT 5: NORMALIZATION

5.1 NORMALIZATION

5.2 FUNCTIONAL DEPENDENCY

5.3 FULLY FUNCTIONAL DEPENDENCY

5.4 PARTIAL FUNCTIONAL DEPENDENCY

5.5 TRANSITIVE FUNCTIONAL DEPENDENCY

5.6 MULTI VALUED DEPENDENCY

5.7 FIRST NORMAL FORM (1NF)

5.8 SECOND NORMAL FORM (2NF)

5.9 THIRD NORMAL FORM (3NF)

5.10 BOYCE CODD NORMAL FORM (BCNF)

5.11 FOURTH NORMAL FORM (4NF)

5.12 FIFTH NORNAL FORM (5NF)

5.1 NORMALIZATION

"Normalization is the process of efficiently organizing data to minimize redundancy in a database and makes database more flexible."

1. E.F. Codd introduced the concept of normalization.
2. Normalization technique is used in designing relational model.
3. It improves database design and removes anomalies for database activities.
4. Its objective is to reduce the redundancy (duplicity) and eliminates the insertion, updation, and deletion anomalies from the database.
5. To achieve its objective, it breaks the database into smaller tables and establishes the relationships between those tables.
6. It makes data consistent throughout the database.
7. Normalization follows some rules. Each rule is known as normal form.
8. E.F. Codd introduced the first normal form (INF) in 1970.
9. He introduced the second normal form (2NF), third normal form (3NF) in 1971 and boyce codd normal form (BCNF) in 1974.
10. For many applications, third normal form (3NF) is necessary.
11. Fourth normal form (4NF) was introduced by Ronald Fagin in 1977.
12. Normal forms are numbered from lowest (INF) to highest (5NF).
13. ***The following are the disadvantages of normalizations:***

- It is a difficult and time consuming process.
- Sometimes, the performance of database degrades from lowest (INF) to highest (5NF).

14. Un-normalized Form (UNF) is one in which a table contains non atomic values at each row. Non atomic values need further decomposition for simplification. For the simplification, un-normalized form goes into first normal form.
15. ***The levels/steps of normalization are as follows:***

Un-normalized Form (UNF)

↓

First Normal Form (INF)

↓

Second Normal Form (2NF)

↓

Third Normal Form (3NF)

↓

Boyce Cod'd Normal Form (BCNF)

↓

Fourth Normal Form (4NF)

↓

Fifth Normal Form (5NF)

Fig. 5.1: Steps of Normalization

5.2 FUNCTIONAL DEPENDENCY

1. Functional dependency is an association between two attributes (columns) of the same relation (table).
2. It is basically a constraint between two sets attributes from the same relation in a database.
3. One attribute is called determinant and other is called determined.
4. For each value of determinant, there is only one value of determined.
5. For example: A→B
   - "B is functionally dependent on A" because for each value of attribute 'A', there is exactly one value of attribute 'B'.
   - If A is determinant and B is determined then we can say that "A functionally determines B" OR "B is functionally dependent on A".

Supplier

Sr. No,	Name	Status	City
S1	Akhil	10	Delhi
S2	Monika	2.0	Patiala
S3	Aastha	30	Delhi

In above table "Supplier", attribute 'Name' is functionally dependent (FD) on attribute 'Sr. No.' because 'Name' has only one value for given 'Sr. No.'.

We can say Sr. No. $\rightarrow$ Name

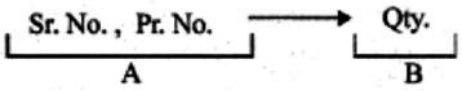
'Sr. No' is determinant and 'Name' is determined.

But attribute 'City' is not functionally dependent (FD) on attribute ' Sr. No.' because 'City' has more than one value for given 'Sr. No.'.

5.3 FULLY FUNCTIONAL DEPENDENCY

1. Fully functional dependency is a functional dependency in which all the non-key attributes are dependent on the key attributes.
2. For example: A $\rightarrow$ B
   - "B is fully functionally dependent on A" because 'B' is functionally dependent on 'A' but not on any proper subset of 'A'.
   - "B is fully functionally dependent on A" means we cannot identify the value of 'B' only from 'A', we can identify the value of 'B' from 'A' and another attribute from the same relation. Another attribute will help the 'A' to find the value of 'B'.
   - If we delete any attribute from the relation, then it will violate the concept of functional dependency.
3. In the below table, Qty. is F.F.D. on 'Sr. No.' and 'Pr. No.' because we can get the value of Qty. only by the combination of both 'Sf. No.' and 'Pr. No.'.

Sr. No.	Pr. No.	Qty.
S1	P1	270
S1	P2	300
S1	P3	700
S2	P1	270
S2	P2	700
S3	P2	300



5.4 PARTIAL FUNCTIONAL DEPENDENCY

1. Partial functional dependency occurs, when some non-key attribute depends on primary key attribute.
2. For example: A $\rightarrow$ B

The attribute 'B' is partial functional dependent on attribute 'A', if there is some attribute that can be removed from 'A' and yet the dependency holds.

5.5 TRANSITIVE FUNCTIONAL DEPENDENCY

1. Transitive functional dependency occurs, when some non-key attribute depends upon other non-key attributes.
2. For example: There are three attributes 'AVB' and 'C'.
   - A$\rightarrow$B
   - B$\rightarrow$C
   - $\Rightarrow$A$\rightarrow$C

   It means 'C' is transitively dependent on 'A'.

5.6 MULTI VALUED DEPENDENCY

1. Multivalued dependency is a full constraint between two sets of attributes in a relation.
2. It plays a role in fourth normal form (4NF) of normalization.
3. For example: If there are three attributes 'A', 'B' and 'C' in a relation.
   - 'B' and 'C' are independent from each other.
   - 'B' and 'C' are multi valued fact about A.

     Then A $\rightarrow\rightarrow$B

     A$\rightarrow\rightarrow$ C
   - Then we can say that "A multi determines B" OR "B is multi dependent on A".

Course_Student_Book

Course	Student	Book
Chemistry	Akhil	B1
Chemistry	Akhil	C1
Physics	Moaika	A1
Physics	Monika	D1
Chemistry	Aastha	Bl
Chemistry	Aastha	Cl
English	Rohit	A1
English	Rohit	D1

Course $\rightarrow\rightarrow$ Student

Course $\rightarrow\rightarrow$ Book

5.7 FIRST NORMAL FORM (1NF)

1. E.F. Cold introduced the first normal form (1NF) in 1970.
2. First normal form (1NF) eliminates the repeating columns from an un-normalized table.
3. In 1NF, there is no repeating column (group).
4. We convert un-normalized table into normalized for.
5. Primary key is required in each table to identify a record.
6. The purpose of primary key is to uniquely identify a record.
7. First normal from depends on the functional dependency.
8. Formula : f(x)=y

   For every value of x, there is only one value for y.
9. For example: The following table "Student" having columns (Name, Course, Roll Number) is an un-normalized table. We have to convert this un-normalized table into normalized table.

Student

Name	Course	Roll Number
Akhil	Science	211, 128
Monika	Computer	129
Aastha	Management	112

The above table "Student" is un-normalized because it contains more than one value for the column 'Roll Number'. 'Akhil' has two values (211, 128) for the column 'roll number' which is not possible. For normalization, there should be only one value in one column.

The following are two methods to convert un-normalized table into normalized table:

- **Method 1:** To convert the un-normalized table "Student" into normalized form, we decompose (divide) this un-normalized table into two tables.

Student 1

Name	Course
Akhil	Science
Monika	Computer
A'astha	Management

Student 2

Name	Roll Number
Akhil	211
Akhil	128
Monika	129
Aastha	112

- **Method 2:** To convert the un-normalized table "Student" into normalized form, we convert this this un-normalized table into flat table.

Student

Name	Course	Roll Number
Akhil	Science	211
Akhil	Science	128
Monika	Computer	129
Aastha	Management	112

Anomalies in First Normal Form (INF)

1. **Insert Anomaly: We** cannot insert any information of new student in table "Student" until he join any course. Similarly, we cannot insert any information about the course until there is any student. This phenomenon is known as insert anomaly.

Student

Name	Roll Number	Course
Akhil	211	Science
Monika	129	Computer
Aastha	112	Management

Rohit	**111**

The details of new student 'Rohit' cannot insert into the table "Student" until he join any course. It is called insert anomaly.

2. **Update Anomaly :In** the update anomaly, if we want to change (update) the course of any student, then we have to change (update) the multiple records. If we change the course of the student but forget to change the details of that student from all the locations where it occures, then data become inconsistent/This phenomenon is known as update anomaly.
3. **Delete Anomaly**: If we delete any course from table "Student", then all the related information to that course automatically deletes.

For Example: if we delete the course 'management' from the table "Student", then it automatically ceases the name 'Aastha' and roll number '112'.

Student

Name	Roll Number	Course
Akhil	211	Science
Monika	129	Computer
Aastha	112	Management

After deletion, table "Student will be look like:

Student

Name	Roll Number	Course
Akhil	211	Science
Monika	129	Computer

5.8 SECOND NORMAL FORM (2NF)

1. E.F. Codd introduced the second normal form (2NF) in 1971.
2. A relation is in 2NF if it fulfills the following conditions
   - Relation should be in INF and
   - Every non-key attribute (non-prime attribute) is fully functionally dependent on Primary key.
3. For example-.The following table "Products" having columns (Item, Price, Quantity, Order Number, and Order Date) is in INF.

Products

Item	Price	Quantity	Order Number	Order Date
Mobile	2000	20	11	1-7-2015
Sunglasses	1000	15	12	2-7-2015

Watch	800	18	13	3-7-2015
Wallet	600	12	14	4-7-2015

- The table "Products" has two primary key columns (Item and Order Number).
- Price (non-primary key column) is fully functionally dependent on Item (prime key column).
- Order Date (non-primary key column) is fully functionally dependent on Order Number (prime key column).
- The table "Products" can be converted into second normal form (2NF) by decomposing it into sub tables such as:

Item	Price
Mobile	2000
Sunglasses	1000
Watch	800
Wallet	600

Order Number	Order Date
11	1-7-2015
12	2-7-2015
13	3-7-2015
14	4-7-2015

Item	Quantity	Number
Mobile	20	11
Sunglasses	15	12
Watch	18	13
Wallet	12	14

Anomalies in Second Normal Form (2NF):

1. **Insert Anomaly:** Second form (2NF) also Suffers from the inset anomaly same like the first normal form (1NF). We cannot insert any information of 'Price' in table "Products" until is associates with any 'item'. Similarly, we cannot insert any information about the 'item' in the table "Products" until its price is fixed. This phenomenon is known as insert anomaly.

Products

Item	Price	Quantity	Order Number	Order Date
Mobile	2000	20	11	1-7-2015
Sunglasses	1000	15	12	2-7-2015
Watch	800	18	13	3-7-2015
Wallet	600	12	14	4-7-2015
	1200	16	15	5-5-2015

The details of new price '1200' cannot insert into the table "Products" until it associates with any 'item'. We cannot left blank the value of any column. It is called insert anomaly.

2. **Update Anomaly:** In the update anomaly, if we want to change (update) the 'price' of any 'item', them we has to change (update) the multiple records. It we change the 'price' of any 'item' but forget to change the details of that 'item' from all the locations where is occurs, then data become inconsistent. This phenomenon is known as update anomaly.

3. **Delete Anomaly:** Like 1NF, 2NF also suffers with delete anomaly. If we delete any 'item' from table "Products", then all the related information to that 'item' automatically deletes. For example : if we delete the item 'watch' from the table "products", then it automatically deletes it's all related information (price, quantity, order number, order date).

Products

Item	Price	Quantity	Order Number	Order Date
Mobile	2000	20	11	1-7-2015
Sunglasses	1000	15	12	2-7-2015

Watch	800	18	13	3-7-2015
Wallet	600	12	14	4-7-2015

After deletion, table "products" will be look:

Product

Item	Price	Quantity	Order Number	Order Date
Mobile	2000	20	11	1-7-2015
Sunglasses	1000	15	12	2-7-2015
Wallet	600	12	14	4-7-2015

5.9 THIRD NORMAL FORM (3NF)

1. E.F. Codd introduced the third normal form (3NF) in 1971.
2. It means a relation (table) is in 3NF if it is in 2NF and there is no transitive dependency.
3. The objective to 3NF is to remove all transitive dependencies.
4. A relation is in 3NF if it fulfills the following conditions:
   - Relation should be in 2NF and
   - Every non-key attribute (non-prime attribute) is transitively dependent on Primary key only.
5. It removes the anomalies of 2NF.
6. For many applications, third-normal form (3NF) is necessary.
7. For example: The following table "Record" having columns (Name, Roll Number, System, Number, Hours_Rate) is in 2NF.

Record

Name	Roll Number	System Number	Hours Rate
Aastha	112	S1	20

Akhil	211	S2	18
Monika	129	S3	17
Rohit	219	S2	15
Aditya	285	S3	16
Kunal	712	S4	12
Sachin	125	S1	23
Rahul	231	S4	25
Siddharth	123	S5	13

- 'Name' is a primary key and the entire non-key attributes (Roll Number, System Number, Homrs_Rate) are dependent on it.
- To convert the table "Record" into 3NF, we decompose it into two tables (Student 'Record, Charge Record).

Student Record

Name	Roll Number	System Number
Aastha	112	SI
Akhil	211	S2
Monika	129	S3
Rohit	219	S2
Aditya	285	S3
Kunal	712	S4
Sachin	125	S1
Rahul	231	S4
Siddharth	123	S5

Charge Record

System Number	Hours Rate

SI	43
S2	33
S3	35
S4	37
S5	13

- Table "Student Record" provides the detail of student like Name, Roll Number and System Number used by him/her.
- Table "Charge Record" provides the details of system like System Number, Charges for using System.

Anomalies in Third Normal Form (3NF)

1. **Insert Anomaly:** Third normal form (3NF) is also suffers from insert anomaly but upto some extent. It is possible to insert in advance, the rate to be charged from student for a system.
2. **Update Anomaly:** If Hours_Rate for a system in table "System Record" changed (updated), then we need only to change a single record in table "Charge Record".
3. **Delete Anomaly:** It we delete the record of a student who is only student working on a particular system, then we will not lose the information of the system and hours_rate of that system.

5.10 BOYCE CODD NORMAL FORM (BCNF)

1. E.F. Codd introduced the Boyce Codd Normal Form (BCNF) in 1 974.
2. A relation is in BCNF, if it is in 3NF and every determinant (attribute) is a candidate key.
3. It means BCNF have multiple candidate keys (more than one primary key).

5.11 FOURTH NORMAL FORM (4NF)

1. Fourth normal form (4NF) was introduced by Ronald Fagin in 1977.

2. 2NF, 3NF and BCNF are concerned with functional dependencies whereas 4NF concerned with multivalued dependencies.
3. A relation is in 4NF if it is in 3NF or BCNF and contains no multi valued dependencies.
4. For example: The following table "Course_Student_Book" is in 3NF.

Course Student Book

Course	Student	Book
Chemistry	Akhil	Organic Chemistry
Chemistry	Akhil	Physical Chemistry
Physics	Monika	Optics
Physics	Monika	Mechanics
Chemistry	Aastha	Organic Chemistry
Chemistry	Aastha	Physical Chemistry
English	Rohit	English Literature
English	Rohit	English Grammar

- Attributes 'Student' and 'Book' are multivalued facts about the attribute 'Course'. There are many students for one course and many books for one course.
- The condition of 4NF is that there should be no multi valued attribute in a table.
- To convert the table "Course_Student_Book" into 4NF, we decompose it into two tables (Course Student, Course Book).
- Table "Course-Student" tells us which student is studying which course.
- Table "Course_Book" tells us which book is available for which course.

Course Student

Course	Student
Chemistry	Akhil
Physics	Monika
Chemistry	Aastha

Course Book

Course	Student
Chemistry	Organic Chemistry
Chemistry	Physical Chemistry
Physics	Optics

English	Rohit

Physics	Mechanics
English	English Literature
English	English Grammar

Note: If anew student 'Rahul' wants to join a course 'English' and use books of 'English' and 'Chemistry', then we have to insert new information of student 'Rahul'. We will insert the name 'Rahul' twice. First entry for 'English' and second entry for 'Chemistry'.

5.12 FIFTH NORNAL FORM (5NF)

1. A relation is in 5NF if it is in 4NF and based on join dependency.
2. Join dependency means when a table is decompose/divide into three or more tables, and then the resulting tables (divided tables) can be rejoined to form the original table.
3. The following are three sub tables (Course_Student, Course_Book and Student_Book) of original table "Course_Student_Book". *The table "Course_Student_Book" is used in the fourth normal form (4NF).*

Course_Student

Course	Student
Chemistiy	Akliil
Physics	Monika
Chemistry	Aastha
English	Rohit

Course	Book
Chemistry	Organic Chemistry
Chemistry	Physical Chemistry
Physics	Optics
Physics	Mechanics
English	English Literature
English	English Grammar

Student_Book

Student	Book
Akhil	Organic Chemistry
Akhil	Physical Chemistry
Monika	Optics
Monika	Mechanics
Aastha	Organic Chemistry
Aastha	Physical Chemistry '
Rohit	English Literature
Rohit	English Grammar

4. When we will join these three tables (Course_Student, Course_Book and Student_Book), then we will get the original table "Course_Student_Book".

Questions

1. What is Normalization? State and explain its types.
2. What is the need of Normalization of data? What are the various techniques for normalization in relational database model?
3. What is Functional dependency? Explain in detail Give an example also.
4. What do you mean by redundancy? Explain the ways to remove it from the database?
5. What do you mean by Normal forms? Explain the various types of it along with the suitable example.
6. What is the difference between First and second Normal Forms?
7. What is INF? Give example to demonstrate how INF improves a table.
8. Discuss 2NF. Discuss the problems that can be encountered in a table, which is in INF, How 2NF solve them?
9. Define 3NF? Discuss its need.
10. Explain Boyce Codd Normal Form.
11. Explain multivalued dependency. Give an example.

12. Explain Join dependency. Give an example.
13. Explain 4NF along with example.
14. What do you mean by FDs? Explain the Closure of a Set of FDs.
15. Explain 5NF along with example.
16. What is fully functional dependency? Give an example.

COURSE: DBMS

UNIT 6: TRANSACTION MANAGEMENT AND CONCURRENCY CONTROL

6.1 Transaction

6.1.1 States Of Transaction

6.1.2 Acid Properties of Transaction

6.1.3 Scheduling of Transaction

6.1.3.1 Types of Schedules

6.1.4 Serializability

6.2 Concurrency Control

6.2.1 Need of Concurrency Control

6.3.1 Locks

6.3.1.1 Types of Locks

6.3.1.2 Compatibility of Locks

6.3.2 Concurrency Control Algorithms

6.3.2.1 Pessimistic Approach

6.3.2.2 Optimistic Approach

6.4 Deadlock

6.4.1 Reasons for the Occurence of Deadlock

6.4.2 Deadlock Prevention

6.4.3 Deadlock Detection

6.5 Database Security And Integrity

6.6 Database Security.

6.6.1 Issues in Security

6.6.2 Need/Requirement of Security

6.6.3 Levels of Security

6.6.4 Different Methods of Database Security

6.6.4.1 View

6.6.4.2 Privilege

6.6.4.3 Roles

6.6.4.4 Encryption

6.6.4.5 Role of DBA *in* Security

6.7 Basic Concepts Of Security

6.8 Database Integrity

6.9 Recovery

6.9.1 Cause/Reason of Failure

6.9.2 Terms Used in Recovery Process

6.10 Atomicity Of Transaction

6.10.1 Log Based Recovery

6.10.1.1 Deffered Database Modification

6.10.1.2 Immediate Database Modification

6.10.2 Shadow Paging

6.11 Disaster Management

6.1 TRANSACTION

1. One or more operations collectivelya single unit of database work is known as database transaction.
2. A transaction is a group of actions such as select, insert, update and delete performed on the database to change the state of a database.
3. A transaction is an action orof actions carried out by the user or the application.

4. It is an atomic operation by the use in reality and goes through number of states during its-lifetime.
5. Once a transaction starts, it ends with success or failure.
6. A successful transaction commits andreaches a new consistent state.
7. A failed transaction and 'rolled back' or 'undone'. In the failed transaction; database restores the previous consistent state.
8. A database transaction must be ACID ([Atomicity-Consistency, Isolation, and Durability]. We will discuss ACID of transaction in *section 7.1.2.*
9. Transactions are supported by SQL [Structured Query Language]. We will discuss SQL in *chapter 10.*
10. A transaction is required to manage of oversees the sequence of events (transactions).

6.1.1 States of Transaction

The database transaction can be in one of the following four states:

1. Active State

- A transaction is in active state while its statements start to be executed.
- Once astate, it starts executing its statements and ends with the commit state.
- Sometimes, with partially committed state. At this phase, the database has its but it is still, possible for the transaction to be aborted because the output is residing temporarily in main memory due to hardware failure.

2. Failed State

- After the active state, sometimes transaction enters in the failed state because its execution can no longer proceed (due to program error or hardware error).

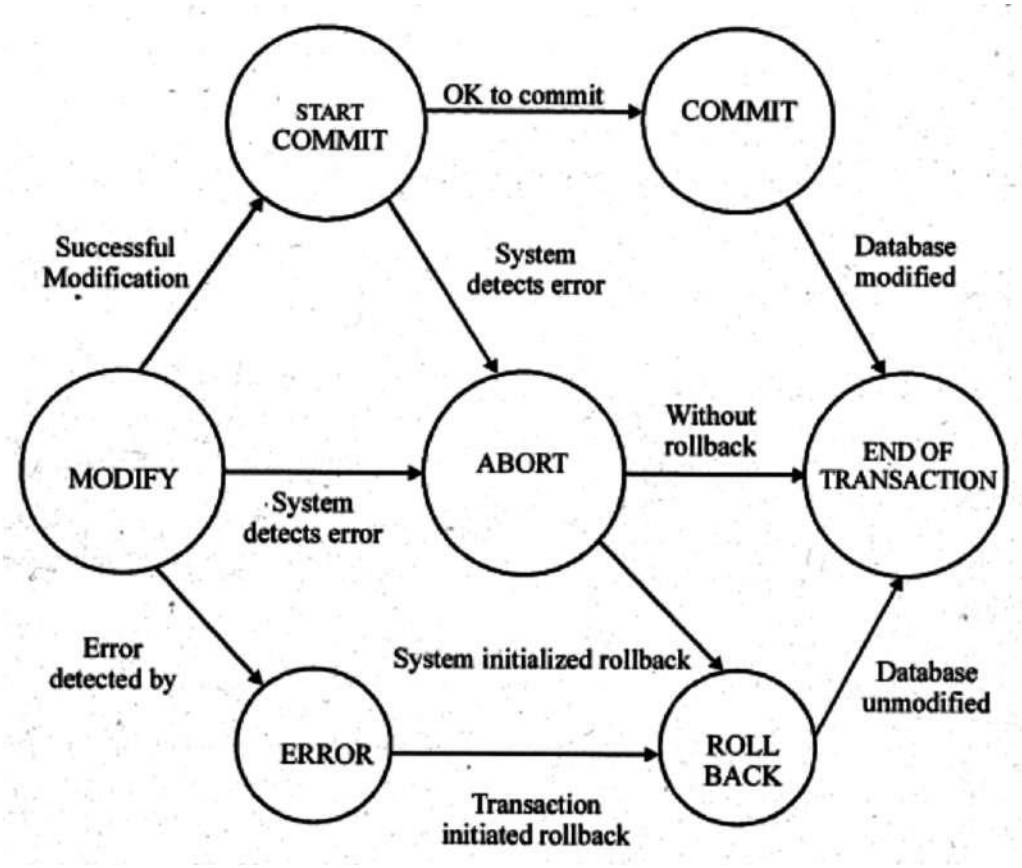


Fig. 6.1: States of Transaction

3. Aborted State

- Aborted state comes, when transaction end with failure.
- A failed transaction enters in the aborted state and 'rolled back' or 'undone'.
- An aborted transaction has no effect on database and can retain its consistent state.
- In the failed transaction, database restores the previous consistent state.

4. Committed State

- Committed state comes, when transaction end with success.
- A successful transaction commits and database reaches in a new consistent state.
- A successful transaction cannot 'rolled back' or 'undone'

6.1.2 Acid Properties of Transaction

To ensure the integrity of data we require that the database system maintains the properties of transactions abbreviated as ACID. The ACID properties of transaction are as follows:

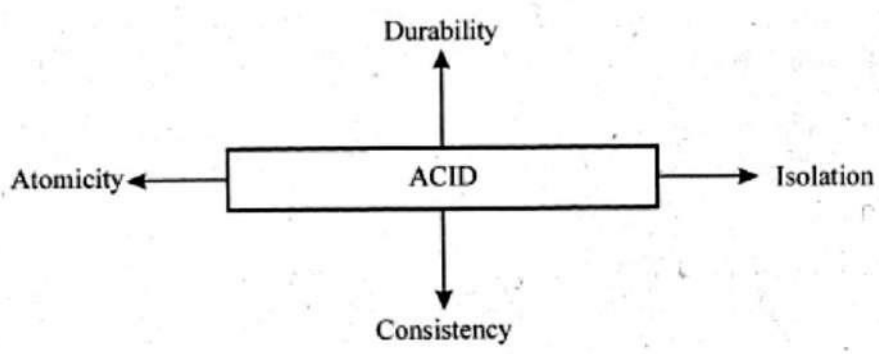


Fig. 6.2: Acid Properties Atomicity

1. Atomicity

- Atomicity means either 100% modification on 0% modification.
- According to atomicity, we cannot perform half operations of any transaction.
- It means we should perform all the operations of the transaction or we should not perform any operation.
- It ensures that either a transaction ends with committed state or rolled back state.
- Committed state comes, when transaction end with success and database reaches in a new consistent state.
- Aborted state comes, when transaction end with failure and database restores the previous consistent state. A failed transaction enters in the aborted state and 'rolled back' or 'undone'.
- Ensuring atomicity is the responsibility of the database system itself. It is handled by a component called the *transaction management component.*

2. Consistency

- According to the -consistency property of the transaction, database remains in a valid state (consistent state) before and after the transaction is committed.
- It means transaction cannot violate the rules (integrity constraints) of the database.
- Ensuring consistency for an individual transaction is the responsibility of the *application manager* who codes the transaction.

3. Isolation

- It means that the execution of one transaction in not affected the other concurrent transactions.
- According to isolation property, data used during the execution of one transaction is not used by another transaction until the execution is not completed.
- It means that the actions performed by a transaction will be isolated or hidden from outside the transaction until the transaction terminates.
- Ensuring the isolation property is the responsibility of a component of a database system called the concurrency control component.

4. Durability

- According to the durability, once a transaction completes successfully, all the updates that it carried out on the database persists even if there is a system failure after the transaction completes execution.
- After the transaction has been successfully completed, all the modifications of a transaction will permanent (cannot 'roll back' or 'undone') even if the system failure occurs.

6.1.3 Scheduling of Transaction

- A schedule is a list of actions (Reading, Writing, Aborting or Committing) from a set of transactions.

- A schedule is a sequence of the operations by a set of concurrent transactions that preserves the order of the operations in each of the individual transactions.
- For example: [4]M' is a schedule. Schedule 'M' is a set of three transactions Tl, T2 and T3. Transaction Tl reads and writes to object A.
  Transaction T2 reads and writes to object B.
  Transaction T3 reads and writes to object C.

Schedule 'M'

Tl	T2	T3
Read (A)		
Write (A)		
Commit		
	Read (B)	
	Write (B)	
	Commit	
		Read (C)
		Write (C)
		Commit

Example of Scheduling of Three Transactions

6.1.3.1 Types of Schedules

There are two types of schedules:

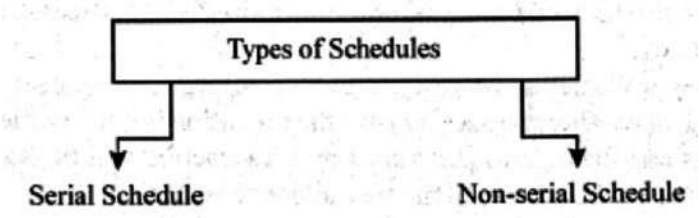


Fig. 6.3: Types of Schedules

1. Serial Schedule

- In a serial schedule, the transactions are performed in serial order. T there is no interference between transactions and only one transaction is executing at a given time.
- In a serial schedule, all the steps (operations) of each transaction executed consecutively without overlapping.
- For example: Schedule 'M' is a set of three transactions Tl, T2 and T3. Schedule 'M' is a serial schedule because the actions of the 3 transactions (Tl, T2 and T3) are not interleaved and executed consecutively.

  Transaction Tl reads and writes to object A.

  Transaction T2 reads and writes to object B.

  Transaction T3 reads and writes to object C.

Schedule 'M'

Tl	T2	T3
Read (A)		
Write (A)		
Commit		
	Read (B)	
	Write (B)	
	Commit	
		Read(C)
		Write (C)
		Commit

Example of Serial Schedule

2. Non-Serial Schedule

- A schedule where the operations from a set of concurrent transactions are interleaved.

- In a non-serial schedule, if the operations of the transactions are not properly interleaved, then they result in problems such as lost update, dirty read and inconsistency analysis.
- For example: Schedule 'N' is a set of three transactions T1, T2 and T3. Schedule 'N' is a non-serial schedule because the actions of the 3 transactions (T 1, T2 and T3) are interleaved.

  Transaction Tl reads and writes to object A.

  Transaction T2 reads and writes to object B.

  Transaction T3 reads and writes to object C.

Schedule 'N'

Tl	T2	T3
Read (A)		
	Read (B)	
		Read (C)
Write (A)		
	Write (B)	
		Write (C)
Commit	Commit	Commit

Example of Non-Serial Schedule

6.1.4 Serializability

- The objective of serializability is to find out the non-serial schedules and allow the all transactions to execute concurrently without creating any problem.
- In a non-serial schedule, if the operations of the transactions are not properly interleaved, then they result in problems such as lost update, dirty read and inconsistency analysis.

Schedule 'N'

Tl	T2	T3
Read (A)		
	Read (B)	
		Read (C)
Write (A)		
	Write (B)	
		Write (C)
Commit	Commit	Commit

Example of Non-Serial Schedule

- Serializability is used to find out or prevent the inconsistency (problems) occurs during the non-serial schedule.

6.2 CONCURRENCY CONTROL

1. ***"Concurrency control is the activity of coordinating concurrent accesses to a database in a multi-user database management system.lt is used to coordinate simultaneous transactions while preserving data integrity".***
2. When many transactions execute concurrently, then the concurrency control scheme is used to make isolation.
3. The concurrency control is required when there are multiple accesses to same data by multiple users.
4. Concurrency control in DBMS ensures that transactions are performed concurrently without the concurrency violating the data integrity of a databse.
5. Executed transaction should follow the ACID rules [Atomicity, Consistency, Isolation, and Durability]. We discussed ACID rules in the section 7.1.2.
6. The DBMS must guarantee that only realizable.
7. It also guarantees that no effect of committed transactions is lost, and no effect of aborted (rolled back) transactions remains in the related database.

8. For example: Two travelers who go to electronic ticket booking center at the same time to purchase a train ticket to the same destination on the same train. There's only one seat left in the coach, but without concurrency control, it's possible that both travelers will end up purchasing a ticket for that one seat, However, with concurrency control, the database wouldn't allow this to happen. Both travelers would still be able to access the train seating database, but concurrency control would preserve data accuracy and allow only one traveler to purchase the seat.
9. Similarly, the concurrency control protocol is used to schedule transactions in such a way as to avoid any interference between them. It allows only one transaction to execute at a time; one transaction is committed before the next transaction is allowed to begin.

6.2.1 Need of Concurrency Control

Several problems can occur when concurrent transactions execute in an uncontrolled manner. The some of the problems are as follows:

1. **The Lost Update Problem:** This problem occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect. Successfully completed update is overridden by another user.

2. **The Temporary Update Problem:** This problem occurs when one transaction updates a database item and then the transaction fails for some reason. The updated item is accessed by another transaction before it is changed back to its original value. Occurs when one transaction can see intermediate results of another transaction before it has committed.

3. **The Incorrect Summary Problem:** If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.

6.3 CONCURRENCY CONTROL METHODS/SCHEMES

There are many concurrency control methods/schemes to prevent the conflicts between the transactions. The following are some concurrency control methods:

6.3.1 Locks

1. A lock is a variable associated with the data item to describe its status.
2. Locks are used in concurrent transactions to ensure serializability.
3. It prevents undesired or inconsistent operations on shared resources by other current transactions.
4. They are used to make the isolation property of transaction in the concurrent environment.
5. They describe the status of the data item whether it has been modified or not.
6. A lock on any database object needs to be acquired by the transaction before accessing it.
7. If transaction 'A' acquires a lock on a database object and another transaction 'B' needs to access that database object, then the existing type of lock is checked.
8. According to the locking scheme, if the existing type of lock (transaction 'A') is matched with another transaction's lock (transaction ^{C}B'), then transaction ^{4}B' can use that object.
9. But, if the existing type of lock (transaction 'A') is not matched with another transaction's lock (transaction 'B'), then transaction attempting access is aborted or blocked.
10. There are many types of locks but only one lock is used for each item in database.

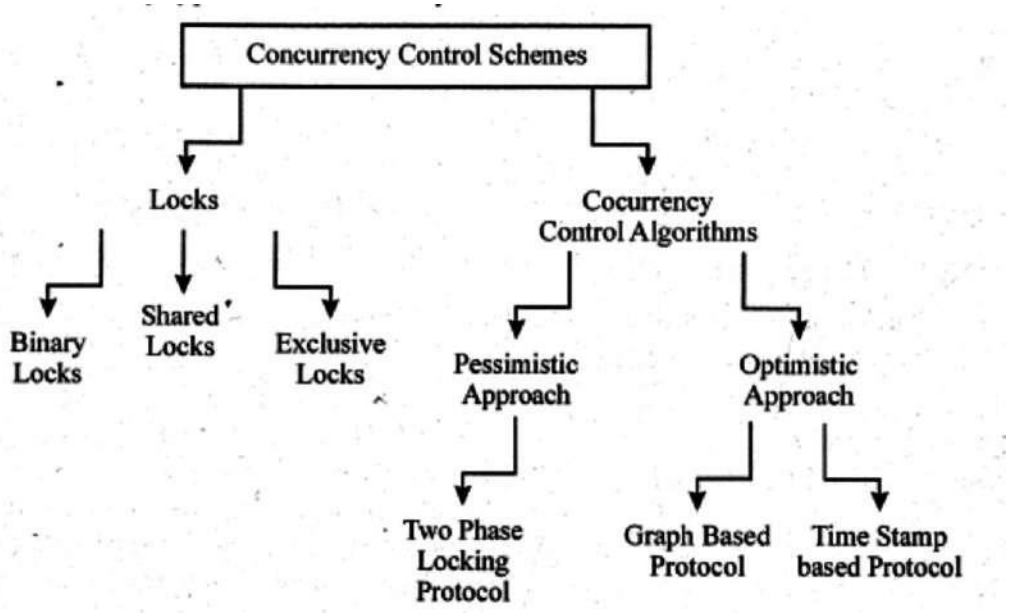


Fig. 6.4: Concurrency Control Methods/Schemes

6.3.1.1 Types of Locks

The following are the types of locks:

I. Binary Locks

1. Binary lock has two states i.e. locked or unlocked.
2. When we use binary lock, it may assign 0 or 1 to the data items.
3. Locked state is represented by 1. It means item cannot be accessed.
4. Unlocked state is represented by 0. It means item can be accessed.
5. If a transaction 'A' wants to access a data item, then it must request for lock.
6. If the data item is already used by another transaction ^{4}B, then transaction 'B' got lock [1] on that date item. It means transaction 'A' cannot access that data item and get zero [0] state.
7. It means transaction 'A' has to wait to access that data item until transaction 'B' finished.
8. Binary lock follow some rules which are as follows:
   - A transaction must get a lock on data item on which it wants to perform read or write operation.

- After the read or write operation, transaction must unlock the data item.
- If any data item holds a lock, then no other transaction can make a lock on that particular data item.
- No two transactions can get the lock on the same data item. It means only one transaction can get the lock on a particular data item.

9. They are not used in practice because only one transaction can hold a lock on a given data item at a particular time which is very impractical.

II. Shared Locks

1. In a binary lock, only one transaction can get the lock on a particular data item. But in shared lock, more than one transaction can use shared fock at a particular time.
2. It is denoted by 'S'.
3. Shared lock is used only for reading purpose. It means, if a transaction want to read data then it will use shared lock on it.
4. Read lock is a shared lock. It means multiple transactions can have read lock on the same item in order to read it.
5. If a transaction 'A' has a shared lock on data item 'M', then other transaction 'B' can only read that data item 'M' not write.
6. For example:

   Lock_S (M): $\rightarrow$ It is used to request a shared lock on data item 'M'.

   Unlock (M): $\rightarrow$ It is used to unlock data item 'M'.

III. Exclusive Locks

1. In a binary lock, only one transaction can get the lock on a particular data item. But in exclusive look, more than one transaction can use exclusive lock at a particular time.
2. It is denoted by 'X'
3. Exclusive lock is used only for writing purpose. It means, if a transaction want to write data then it will use exclusive lock on it.

4. Write tock is an exclusive lock. It means multiple transactions can have write lock on the same item in order to write it.
5. If a transaction 'T1' has obtains an exclusive lock on a data item then another transaction 'T2' cannot perform read but performs write operation.
6. If a transaction 'A' has a exclusive lock on data item 'M', then other transaction 'B' can only write that data item 'M' not read.
7. For example:

   Lock_X (M): → It is used to request an exclusive lock on data item 'M'.

   Unlock (M): →It is used to unlock data item 'M'.

6.3.1.2 Compatibility of Locks

Compatibility of Locks	Shared	Exclusive
Shared	True	False
Exclusive	False	False

1. **Shared lock is compatible with shared lock:** According to this, more than one transaction can read a data item. It means multiple transactions can have read lock on the same item in order to read it.
2. **Shared lock is not compatible with exclusive lock:** According to this, if a data item has exclusive lock, then no other transaction can make shared lock on that particular data item.
3. **Exclusive lock is not Compatible with exclusive** lock: According to this, if a data item has exclusive lock, then no other transaction can make exclusive lock on that particular data item. No two transactions can make exclusive lock simultaneously.

6.3.2 Concurrency Control Algorithms

To control the concurrency problems, there are two algorithms which as follows:

6.3.2.1 Pessimistic Approach

1. In this approach, if the transactions conflict with each other, then there should be some delay in the transactions.

2. Two phase locking protocol based on pessimistic approach as follows:

- It is a common locking protocol which guarantees the serializability.
- It does not ensure the freedom from deadlock.
- It has two phases i.e. growing phase and shrinking phase.

(a) ***Growing Phase:*** In the growing phase, number of locks increases. In this phase, all locks are requested and no one is released. When a transaction begins, it is in a growing phase and required lock is provided.

(b) ***Shrinking Phase:*** In the shrinking phase, number of locks decreases. In this phase, all locks are released and no one is requested. When a transaction ends, it is in shrinking phase. It releases the lock and cannot get any more lock.

6.3.2.2 Optimistic Approach

1. Optimistic approach is also known as validation or certification method.
2. In this approach, there is an assumption that conflicts in database operations are very rare.
3. There is no checking process during the execution of a transaction. Transaction runs unsynchronized and conflicts are checked only at the end.
4. According to this approach, first let the transaction run to the completion, then check the conflicts before the transaction commits.

5. Advantages of Optimistic Method

- This technique is very efficient when conflicts are rare.
- The rollback involves only the local copy of data.

6. Disadvantages of Optimistic Method

- Conflicts are expensive to deal.
- Longer transactions are more likely to have conflicts and may be repeatedly rolled back.

7. Non-Two phase locking protocol based on optimistic approach as follows:

(a) Graph Based Protocol

1. It is also known as tree protocol.
2. In this protocol, data items are arrange in a tree.
3. We must have prior knowledge about the order in which the database items will be accessed.
4. (X, Y) are shows that X is parent of Y.
5. If there is directed path from X to Y then X is called ancestor of Y.
6. It ensures serializability.

7. The following are some rules of graph based protocol:

- No data can be accessed unless transaction locks it.
- Transaction can unlock data any time.
- Transaction after unlocking data cannot relock it again.

8. Advantages of Graph Based Protocol

- Unlocking may occur earlier which may lead to shorter waiting time.
- It is deadlock free, No rollback is required.

9. Disadvantages of Graph Based Protocol

- In most of the cases, it is not known prior what data will need to be lock.

(b) Time Stamp Based Protocol

1. Time stamp based protocol is used in relational databases to safely handle transactions
2. In this protocol, we must have the prior knowledge of order of transactions and data items.
3. The time stamp is assigned by the database system before the transaction starts.
4. A unique fixed time stamp [TS(Ti)] is associated with each transaction.
5. It follows the serializability order.
6. **There are two methods for assigning the time stamp to each transaction as follows:**

- Use a separate counter to assign the time stamp.
- Use a system clock to assign the time stamp.

7. We can implement time stamp on data item 'M' as follows:
   - W-timestamp (M) :-> It is a time stamp for writing on data item 'M'.
   - R-timestamp (M):-> It is a time stamp for reading on data item 'M'.

6.4 DEADLOCK

1. Deadlock is basically a mutual blocking between transactions.
2. A system is in deadlock state if there are set of transaction and every transaction is waiting for other transaction to release lock.
3. Example of deadlock:
   - There are two transactions CTP and 'T2'.
   - Transaction 'T1' has exclusive lock and transaction 'T2' has shared lock
   - Transactions 'Tl' and 'T2' execute concurrently.
   - Transaction 'Tl' make exclusive lock (X) on data item 'A'.
   - Transaction 'T2' make shared lock (S) on data item 'B'.

Tl	T2
Lock-X(A)	
Read (A, a)	
a = a-50	
Write (A, a)	
	Lock-S(B)
	Read(B, b)
	Unlock (B)
	Lock-S(A)
	Wait.....
Lock -X(B)	
Wait……	

- Transaction 'T2' make shared lock (S) on data item 'B' and want to make shared lock on data item 'A'. But transaction 'T2' has to wait till the transaction 'Tl' release an exclusive lock (X) on data item 'A'.
- On the other hand, transaction 'T' make exclusive lock (X) on data item 'A' and want to make exclusive lock on data item 'B'. But the transaction 'TP' has to wait till the transaction 'T2' release the shared lock (S) on data item 'B'.
- Both transactions 'Tl' and 'T2' cannot release the lock because shared and exclusive locks are incompatible with each other.
- This is the situation where neither of the transactions can proceed. Both transactions wait for each other to release the lock but both are incompatible and cannot release the lock. This situation is called deadlock.

4. A deadlock can be resolved by aborting a transactions and breaking the cycle.
5. When deadlock occur, system must rollback one of transaction.
6. One transaction rollback the data that were locked by that transaction are unlocked and is available to other transaction.

6.4.1 Reasons for the Occurence of Deadlock

A set of processes is in a deadlock state if every process in the set is waiting for an event to release that can only be caused by some other process in the same set. The following are some reasons for the occurrence of deadlock:

1. Mutual Exclusive: When a single process is used by two or more processes, means a single resource if used for performing the two or more activities as a shared based. But this is will also create a problem because when a second user request for the system resource which is being used by the user.

2 Hold and Wait: A single process may need two or more system resources. And suppose if a process have a single resource, and is waiting the second resource. Then process can't leave the first resource and waiting for the second resource. So that there will also be the condition of Deadlock.

3. No Preemption: If there is no rule to use the system resources. It means if all the system resources are not allocated in the manner of scheduling. Then this will also create a problem for a Deadlock because there is no surety that a process will release the system resources after the completion.

4. Circular Wait: When two or more requests are waiting for a long period of time and no one can access the resource from the system resources, then this is called as Circular Wait. For example if two or more users request for a Printer, at a same tifrie, they request to print a page. Then they will be on the Circular Wait means System will display a busy sign.

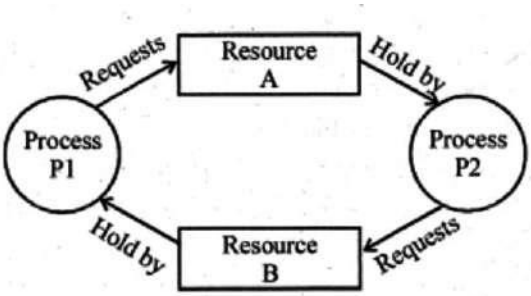


Fig. 6.5: Circular Wait

Example of occurrence of deadlock: There exists a set of waiting transactions {Processl,.,.,.,., Process 4} such that Process1 is waiting for data item that is held by Process2, Process! is waiting for a data item that is held by Process3, so on. None of the transactions can make progress in such a situation.

For avoiding a Dead Lock first of all we have to detect Dead-Lock means firstly we have to detect why and how a Deadlock has occurred and then avoid or solve the problems those are occurred due to occurrence of Deadlock.

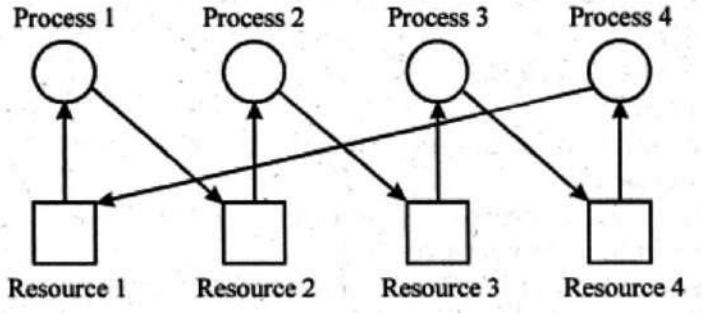


Fig. 6.6: Deadlock condition

6.4.2 Deadlock Prevention

To prevent the system from the deadlock state. There are two methods as follows:

1. According to first method: Transaction manager should not allow a transaction which goes in a waiting state for a data item. The following are some rules/protocols which help to prevent deadlock state:

- Transaction manager should avoid the waiting cycles. He should use deadlock prevention protocol so that system never enters in a deadlock state.
- One protocol to ensure that hold and wait condition never occurs. Each process must request and get all of its resources before it begins execution.
- Each process can request resources only when it does not occupy any resources. If a process holding some resources, requests another resource (new resource) which is not allocated to it, then a process must release all the allocated resources so that it can access the new resource.
- If there are multiple requests for a particular resource, then each process can access that resource in increasing order of priority.

2. According to the second method

- If the system enters in a deadlock state, then we should try to recover a transaction from deadlock state by using deadlock detection and deadlock recovery techniques.

6.4.3 Deadlock Detection

1. If a system has no deadlock prevention and no deadlock avoidance scheme, then it needs a deadlock detection scheme with recovery from deadlock capability.
2. In the deadlock detection scheme, an algorithm is used to determine whether the system entered in a deadlock state or not
3. The deadlock detection algorithm should be invoked periodically.
4. The deadlock detection algorithm is as follows:

   Data Structure is as: Available [m]

Allocation [n, m] as in Banker's Algorithm.

Request [n, m] indicates the current requests of each process.

Let work and finish be vectors of length m and n, as in the safety algorithm.

The algorithm is as follows:

1. Initialize Work = Available

   For i= 1 ton do

   If Allocation (i) = 0 then Finish[i] = true else Finish[i] = false
2. Search an i such that

   Finish[i] = false and Request (i)≤ Work

   If no such i can be found, go to step 4.
3. For that i found in step 2 do: Work = Work + Alloeation(i)

   Finish[i] = true

   Go to step 2.
4. If Finish[i] ?true for some i then the system is in deadlock state else the system is safe.

6.5 DATABASE SECURITY AND INTEGRITY

The information stored in database is very valuable for an organization and it must be protected from unauthorized access and unwanted damage. Database security is a method to protect database from unwanted damages due to various reasons like unwanted access, physical damage, technical or mechanical damage, accidental loss, corruption of data etc. The database security allows or disallows users from performing actions on the objects contained within organizational database.

The database security is concerned with various policies which are framed by DBA to protect data. The DBA is responsible for the overall security of the database system. The DBA design overall policies, procedures and appropriate controls to protect and safe the data in database.

6.6 DATABASE SECURITY

1. Database security is the protection of database from internal and external threats.

2. Security is important because database is very valuable. All decisions of organization depend on data.
3. Data should be protected from unauthorized access.
4. Any corruption of data would affect the day-to-day operation.

6.6.1 Issues in Security

Sr. No.	Issue	Description
1	**Data Tampering**	Data cannot be modified; or viewed during transit. In case of distributed database where data moves between different sites, data can be modified during transit. So data security is important.
2	**Data Theft**	Data can be stolen from within the organization or from Internet so data security is important.
3	**Unauthorized Access**	Data should be protected from unauthorized users.
4	**Password Related Threats**	Usually users use their name, date of birth as password which can easily be traced there for security is important to protect data in the case of password threats.

6.6.2 Need/Requirement of Security

Sr. No.	Issue	Description
1	**Confidentiality**	Data should be confidential and user should be able to see the data he is supposed to see
2	**Authentication**	This is a process of verify the user's identity on database. Authenticity can be check after asking user name and password.
3	**Secure Storage**	After confidential data has been entry it should be store or protected in secure database

4	**Privacy of Communications**	DBMS should be able to secure the private data of use like health, employment and credit card number etc.
5	**Availability**	It is the duty of DBMS that data should be available to the user when it is required. It is possible only if it is secure and in authorized hands.
6	**Authorization**	After authenticity authorization get information about the operation that user may perform and the database user may access that may be • Read Authorization • Insert Authorization • Update Authorization • Delete Authorization • Drop Authorization • Alteration Authorization
7	**Integrity**	Integrity ensures that data is protected from deletion and corruption while it is store in database and while it is being transmitted over network.

6.6.3 Levels of Security

If database security has to maintain then it should be maintain at all levels. The security levels are:

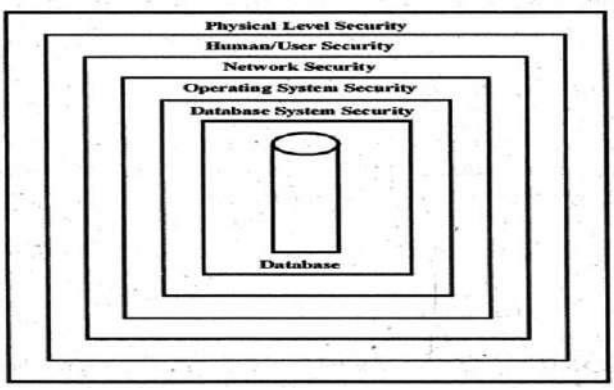


Fig. 6.7: Levels of Security

Sr. No.	Level	Description
1	**Physical Level**	Database must be secure from armed and weapons.
2	**Human**	Database should be secure from unauthorized users.
3	**Network**	Network security is important if database allow to access data , remotely. Network security features must be strong.
4	**Operating System**	If operating system is weak then no strong security features of a database can protect it from unauthorized access.
5	**Database**	The database should be secured from outside world (unauthorized people). Various database security methods are discussed in *section 9.1.4.*

6.6.4 Different Methods of Database Security

There are different methods for protecting data. These methods are as follows:

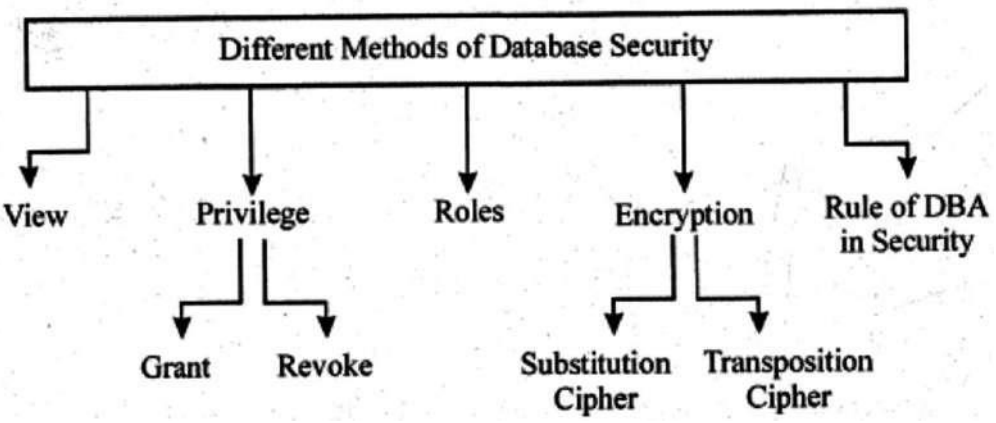


Fig. 6.8: Different Methods of Database Security

6.6.4.1 View

1. View is logical table based on one or more tables.
2. Table on which view is made is called base table.
3. It is just like a table but does not store the data.
4. Using views, we can restrict the set of rows and columns of table.
5. User can only see the provided row and columns.

6. All operations performed on view effect the base table.

7. Syntax

Create view VI as

Select Emp_No, Name, Dept from EMP

Where City ='Chc';

VI is a view and all DML commands can be used on view VI.

6.6.4.2 Privilege

1. Privilege is a permission given to the user to access the database objects.
2. After getting the privilege, user can use any SQL command.
3. With the help of privileges, we can perform the following tasks:
   - Create a table
   - Select rows from table
   - Insert new record in table
   - Update data of table
   - Delete data from table.
4. DBA give privileges to the users.
5. There are two commands used to give and withdrawal the privileges which are as follows:

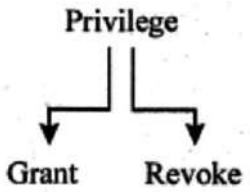


Fig. 6.9: Two Commands in Privilege

(a) Grant

- Granting a privilege to user means giving permission to user for some specific task.
- Granting of privileges to users is done by Grand Command.
- On clause is used to specify the object name.
- To clause is used to specify user name.

- ***Syntax:***

  Grant <privilege> on <object name>to <user name>;

- ***Examples of Grant Command :***

  (a) Grant to user 1 for select the records on employee table, then the query will be:

  Grant Select on Emp to Userl;

  (b) Grant to all users for select the records on employee table, then the query will be:

  Grant Select on Emp to Public;

  (c) Grant all privilege to all the users on employee table, then the query will be:

  Grant all on Emp to Public;

(b) Revoke

- Revoke withdraws granted privileges.
- Revoke takes back all the privileges given to the users.
- ***Syntax:***

  Revoke <privilege> on <table name> from <user name>;

- ***Example of Revoke Command:***

  Withdraw the select grant permission on employee table from user 1, then the query will be:

  Revoke Select on Emp from Userl;

6.6.4.3 Roles

1. Role is a mechanism that Is used to provide authorization.
2. It is a group of privileges.
3. A single person or group of persons can be granted a role.
4. Using roles, DBA can manage access privilege more easily.
5. Suppose there are two roles in college database.,

   (a) Role 1: Which have created, after, drop, insert, select, update, and delete privileges?

   (b) Role 2: Which have select privilege only?

6. Role 1 will be provided to all the users related to staff. Role 2 will be provided to all the users related to students.
7. When a new staff user will joint it will be provided Role 1 and get all the privileges related to Role1. When a new student user will joint it will be provided Role 2 and get all the privileges related to Role 2.

6.6.4.4 Encryption

1. Encryption is a technique by which we can convert the data in coding form.
2. This process of converting is called Encryption.
3. Encrypted key Is needed to convert the plain text to cipher text
4. Plaintext: The message or data which Is to be converted.
5. Cipher text: The converted data.
6. Cipher text is then transmitted to the network.
7. Decryption key is needed to decode the cipher text back to plain text.
8. The process of converting cipher text (coded data) into plain text (original data) is called decryption.

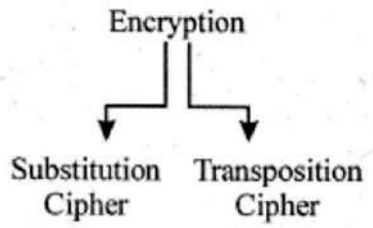


Fig. 6.10: Techniques Used in Encryption

Techniques Used in Encryption

I. Substitution Ciphers

- In this technique, each letter Is replaced by another letter.
- For example: A Is replaced with D, B is replaced with E, C with F and so on such as Attack becomes OWWDFN.
- This technique is not secure because It can be guess easily.

II. Transposition'Ciphers

- In this technique, letters are re-ordered not replaced.

- We arrange the letters In different order but not replace these letters with new letters.
- For example: theft can be .coded as eftth.

6.6.4.5 Role of DBA *in* Security

- Database Administrator plays an important role to provide security.
- DBA is responsible to provide the overall security to the database.
- DBA has a special account which is called system account.
- DBA create new user and provide them user_id and password.
- He creates roles and assigns privileges to the role.
- DBA can assign or change the role of a user.
- DBA create views and assign to the user.
- DBA can also check the log file which describes the detail of a user.
- DBA can detect the violation area.

6.7 BASIC CONCEPTS OF SECURITY

1. Security. Policy

The purpose of a security policy is to elaborate the three general security objectives of secrecy, integrity and availability, in the context of a particular system.

In general, security policy is largely determined within an organization rather than imposed by mandate from outside. This is particularly so in the integrity and availability areas. There are three main objectives to consider while designing a secure database application:

Secrecy: It is concerned with improper disclosure of information. Information should not be disclosed to unauthorized users. For example, a student should not be allowed to examine other students' grades.

Integrity: It is concerned with improper modification of information or processes. Only authorized users should be, allowed to modify data. For example, students may be allowed to see their grades, yet not allowed to modify them.

Availability: It is concerned with improper denial of access to information. The term denial of service is also used as a synonym for availability. Authorized users should not be denied access. For example, an instructor who wishes to change a grade should be allowed to do so.

2. Prevention.

Prevention ensures that security breaches cannot occur. The basic technique is that the system examines every action and checks its conformance with the security policy before allowing it to occur. This technique is called access control.

3. Detection

Detection ensures that sufficient history of the activity in the system is recorded in an audit trail, so that a security breach can be detected after the fact. This technique is called auditing.

4. Assurance

Security mechanisms, whether preventive or detective in nature, can be implemented with various degrees of assurance. Assurance is directly related to the effort required to threaten the mechanism. Low assurance mechanisms are easy to implement but also relatively easy to disrupt. Subtle bugs in system/application software have led to numerous security breaches. On the other hand, high assurance mechanisms are notoriously difficult to implement. They also tend to suffer from degraded performance.

Note: *Prevention is the more fundamental technique. An effective detection mechanism requires a mechanism to prevent improper modification of the audit trail. Moreover, detection is ultimately useful only to the extent that it prevents improper activity by threatening punitive action.*

6.8 DATABASE INTEGRITY

1. It concerned with correctness and consistency of data.
2. This is a main task in multi-user database.
3. Integrity violation may arise from many different sources like:
4. Typing error by data entry clerks.j

5. Logical errors in application.
6. Error in system software.
7. Result of all these violations in data corruption.
8. Database integrity is responsible for monitoring and detecting integrity violations.
9. When integrity violation occur, system then take following actions:
    - Rejection the operation
    - Reporting violation
    - Returning the database to consistent state.
10. ***The following are the database integrity rules:***

(a) **Domain Integrity Rules:** It is used to maintain the correct value of attributes, i.e. for age attribute integrity rules should be in integer and should be positive and it should be possible to specify upper and lower bounds for values of age.

(b) **Entity Integrity Rules**

- It is used to preserve the key uniqueness. (Primary key)
- It specifies that all entries are unique.
- There is no NULL entry in primary key.

(c) **Referential Integrity Rules**

- These rules are concerned with maintaining the correctness and consistency of relationship among relations.
- It specifies that foreign key must have either a NULL value or match with primary key value.
- It ensures not to enter invalid value.
- Referential integrity rule make possible not to delete a row in one table whose matching foreign key value is existing.

6.9 RECOVERY

1. Recovery is the process of restoring the database after the failure.
2. Failure may be the result of system crash due to hardware or software.
3. Recovery is the responsibility of DBA (Database Administrator).

4. Various procedures and strategies (backup and recovery)are used in recovery to protect the database.
5. Whenever a transaction is submitted to a DBMS for execution, the system is responsible for making sure that either
    - All the operations in the transaction are completed successfully and their effect is recorded permanently in the database or.
    - The transaction has no effect on the database or on any other transactions.
6. Before understand the concept of recovery, it is important to understand the cause of failures.

6.9.1 Cause/Reason of Failure

Failures are generally classified as transaction, system, and media failures. There are several possible reasons transaction to fail in the middle of execution:

I. System Crash/Computer Failure

1. In system crash, system hangs up and needed to reboot.
2. A hardware, software, or network error occurs in the computer system during transaction execution.
3. In this case, the data which is in main memory is lost and transaction rolls back.
4. The permanent data which is in permanent storage devices are not affected by system crash.
5. The reasons of this failure are hardware, database software and operating system.

II. User Error

1. User drops a full table.
2. User deletes a record.
3. User purchases the hardware of poor quality.

III. Statement Failure

1. A transaction which has multiple statements and one statement might fall.
2. Result will be error message by database software or operating system.

3. The recovery in this case will be automatic because the transaction will rollback and user can re-execute the statement again,
4. Reasons may be
    - Selecting rows from- a table which doesn't exists.
    - Inserting records by there is not enough space.

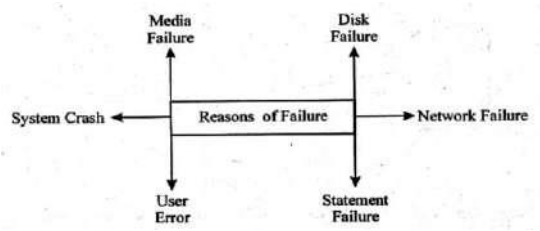


Fig. 6.11:-Reasons of Failure

IV. Network Failure

1. Network failure effect distributed database where data is .coming from different sites.
2. Reason may be
    - Client server configuration
    - Communication software failure

V. Disk Failure

1. This may happen during a read or a write operation of the transaction.
2. Some disk blocks may lose their data because *of* a read or write mal function or because of a disk read/write head crash.

VI. Media Failure (Disasters)

1. This is most dangerous failure.
2. It is very difficult to recover the data effect from these failures.
3. It has only one solution is to take regular backup, i ;,
4. It is because of fires, floods and earthquakes. We will discuss disaster and its management techniques in *section 9.3.*

6.9.2 Terms Used in Recovery Process

1. To understand the concept of recovery, we must understand the concept of main memory (RAM) and secondary memory (Hard disk).
2. Each disk is partitioned into blocks.
3. Block in main memory is called buffer block or disk buffer.
4. A block in disk is called physical block.
5. When a transaction starts, data is transfer form physical block to buffer block.
6. When a transaction ends, data is transfer back from buffer block to physical block.
7. ***There will be two main operations which are as follows:***

   (a) ***Input Operation:*** Transfer data from physical block to buffer block.

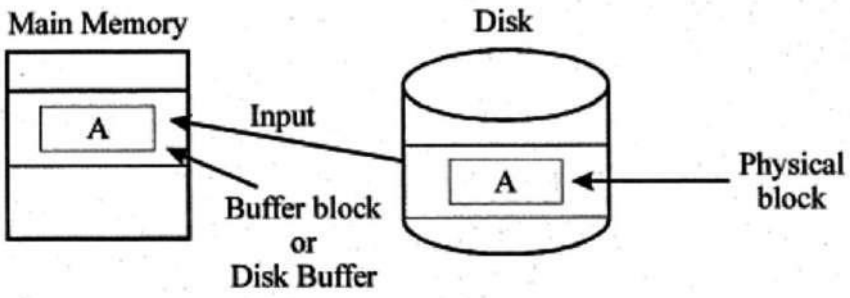


Fig. 6.12: Input Operation

(b) ***Output Operation:*** Transfer data from buffer block to physical block. .Main MemoryDisk

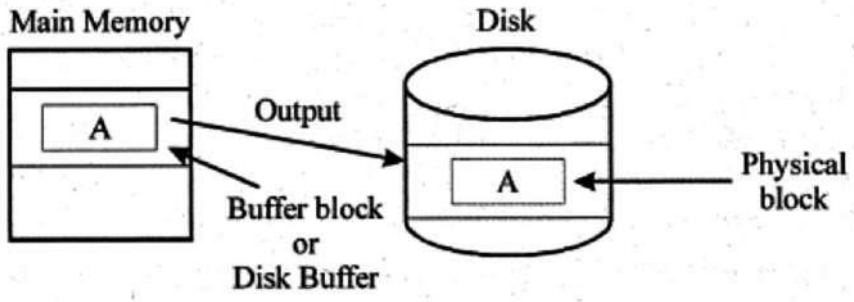


Fig. 6.13: Output Operation

8. Each transaction has a private working area where transaction executes. This area is created and removed according to the transaction.

   For example: we want to perform the following transaction.

Read (A, .a)

a = a-100

Write (A, a)

Three operations will be performed to complete the above transaction as follows:

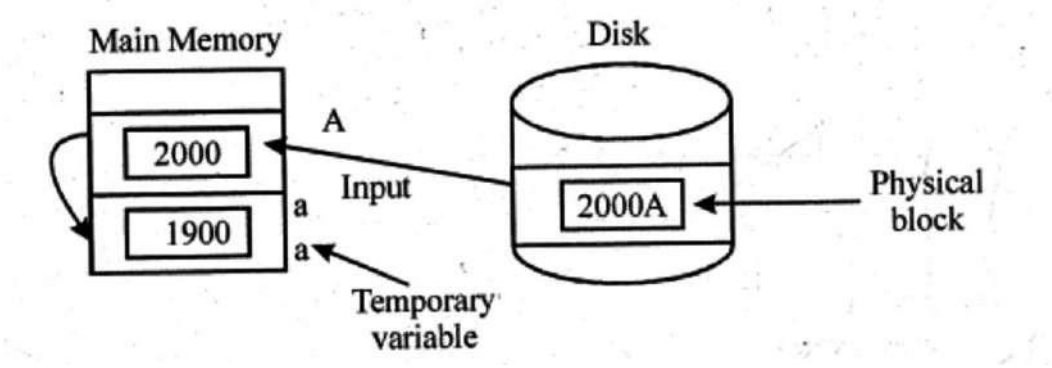
(a) Read (A, a)

(b) Write (A, a)

(c) Output (X)

(a) Read (A, a)

- Find the database item 'A' in block X of disk.
- Then, transfer database item 'A^5 in block X from disk to main memory.
- In main memory, database item 'A' will be copy in temporary variable "a^5.
- The arithmetic operation will be performed as



Fig, 6.14: Read Operation

(b) Write (A, **a)**

- The data item 'A' belongs to ^{C}X' block.
- If 'X^5 is in main memory, then data of temporary variable ^{c}a' is copy to data item 'A'.
- If 'X' is not in main memory then input operation performed again.

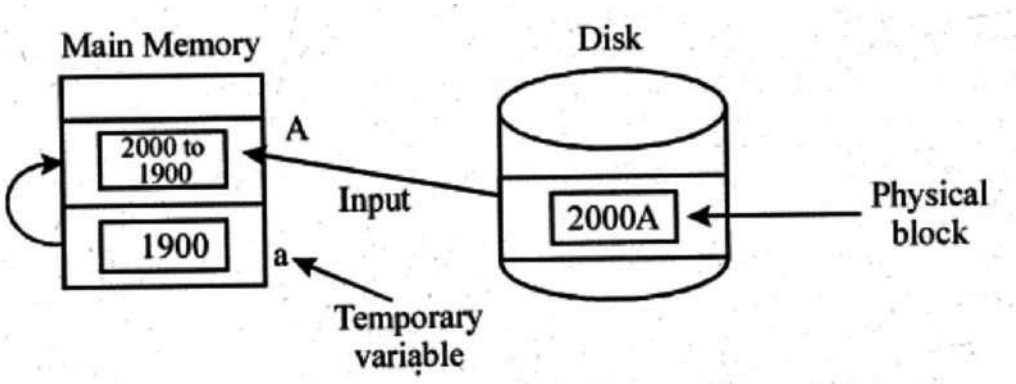


Fig. 6.15: Write Operation

(c) Output X

- After performing the write operation, block ^{C}X' is now written in disk.
- Now, output operation is performed.
- Database item 'A^{9} is copy to disk permanently.

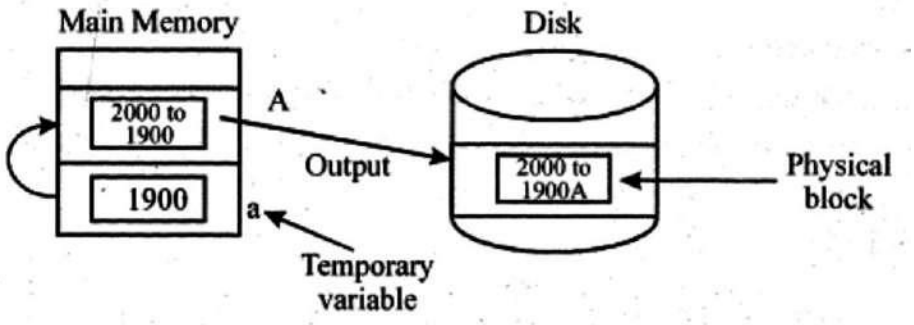


Fig. 6.16: Output Operation

6.10 ATOMICITY OF TRANSACTION

1. Atomicity means either all operations of the transaction are reflected properly in database or none.
2. It ensures that either a transaction ends with committed state or rolled back state.
3. Committed state comes, when transaction end with success and database reaches in a new consistent state.
4. Aborted state comes, when transaction end with failure and database restores the previous consistent state. A failed transaction enters in the aborted state and 'rolled back' or 'undone'.

5. According to atomicity, we cannot perform half operations of any transaction.
6. It means we should perform all the operations of the transaction or we should not perform any operation.
7. In short, atomicity means either 100% modification on 0% modification.
8. Ensuring atomicity is the responsibility of the database system itself. It is handled by a component called the *transaction management component,*
9. *For example:*

   There are two accounts 'A' and 'B'. Account 'A' contains Rs.2000. Account 'B' contains Rs.1000.

   The transaction Tl transfers Rs.100 from account "A" to account 'B'.

Transaction Tl

A is account	Read (A, a)	A = 2000, a = 2000
a is temp variable	a=a-100	a -1900
B is account	write (A, a)	A =1900
b is temp variable	output (A, X)	
AX, BX buffer block -	Read (B,b)	B = 1000, b= 1000
	b = b + 100	b =1.100.
	Write (B,'b)	B = 1100
	Output (BX)	

Example 6.1: Example of Atomicity

In the transaction Tl, if a system crash occurs after the output (AX) but before output (BX), then the output will be:

A is account	Read (A, a)	A = 2000, a = 2000
a is temp variable	a = a- 100	a= 1900
B is account	write (A, a)	A =1900
	output (A, X)	

After the reduction of Rs.100 from account 'A', system crashes. Now, system is in inconsistent state and there is a loss of Rs.100.

Account A has Rs. 1900

Accounts has Rs. 1000

To recover this loss, if we execute transaction Tl again, then database will again in inconsistent state because account 'A^9 has Rs.1900 and account 'B^9 has Rs.1000. We cannot recover this Rs.100 again if we re-execute the transaction Tl after the modification. This example gives the concept of atomicity which means either 100% modification on 0% modification.

10. Atomicity can be achieved by transferring the output in storage without modifying database. There are two ways to achieve atomicity as follows:

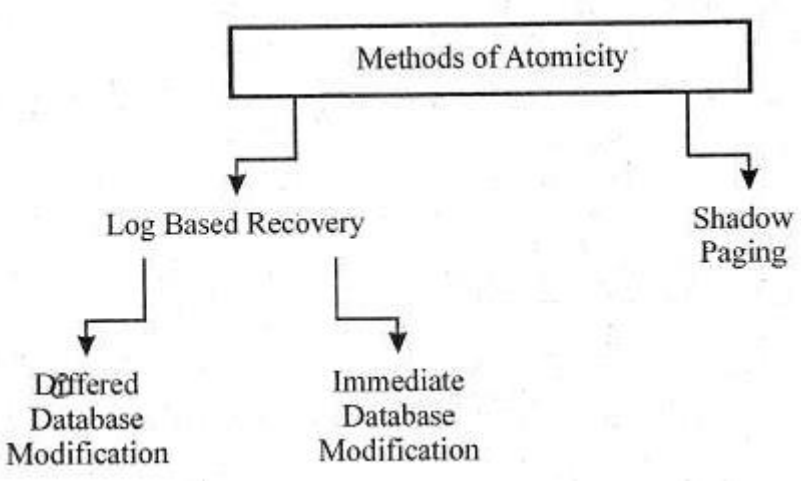


Fig. 6.17: Methods of Atomicity

6.10.1 Log Based Recovery

1. A log file is maintained in Log based recovery.
2. A log file is used to maintain the record of all the operations of database.
3. Basically, it is a sequence of log record.
4. The following are the different types of log records:

1	**<Start> Log Record**	It contains the information about the start of each transaction and used to differentiate the state of

		different transactions. <Ti start>
2	**<Update> Log Record**	It updates the data item under the transaction. <Ti, Xj, VI, V2> Ti is transaction Xj is data item VI is old value V2 is new value
3	**<Commit > Log Record**	It tells that the transaction is completed. When transaction Ti successfully completed, then log record is store in the log file.
4	**<Abort> Log Record**	It tells that the transaction is not completed When transaction Ti is not successful completed then it will be abort and log record is stored in log file.

5. After the log record, database can be modified.
6. We can recover the data from the log record. In the previous example discussed in atomicity (example 9.1), when a system crashes after the output (AX) but before output (BX), there is a loss of Rs.100 after the execution of transaction Ti.
7. We can recover that loss by copying the old value of ' A' Rs.2000 from log record to database. It is called undo operation.
8. The following are the two techniques for log based recovery:
   - Differed Database Modification
   - Immediate Database Modification

6.10.1.1 Deffered Database Modification

1. During write operation, the modified value of local variable is stored in log record not in database.
2. After the successful execution of transaction, the modified value is copied in the database.

3. If the transaction fails to complete, then the modified value and log record in ignored. In this case, the value of data item maintains its old value.

4. For example:

There are two accounts 'A' and 'B'. Account 'A' contains Rs.2000. Account 'B' contains Rs. 1000. Account 'C' contains Rs.3000.

The transaction TI transfers Rs.100 from account 'A' to account 'B'. The transaction T2 withdraws Rs.100 from account 'C'.

Transaction Tl

Read (A, a)
= a-100
write (A, a)
Read (B, b)
= b+100
Write (B, b)

Transaction T2

Read (C, c)
= C-100
Write (C,c)

The transactions Tl and T2 will execute in sequence.

In the deferred database modification, data will be modified in the database if log contain both <Ti, start> and <Ti, commit> otherwise transaction will rollback and updation will be cancelled.

During Transaction Tl		
Tl	**Log**	**Database**
Read (A, a)	<T1, start>	(Buffer)
a = a- 100	^TUA, 1900>	
Write (A, a)	<T1,B, 1100>	
Read (B, b)	<T1, commit>	A = 1900
B = b+100		B = 1100
Write (B,b)		
During Transaction T2		

T2	Log	Database
Read (C, c)	<T2, Start>	
c = c-200	<T2, C, 3800>	C = 28'00
Write (C, c)	<T2, commit>	

Example 6.2: Example of Deferred Database Modification

6.10.1.2 Immediate Database Modification

1. This technique allows database modification while transaction is still in active state.
2. If the system crashes, then the transaction abort and log record is used to restore value.
3. The log record can restore the value with the help of undo operation. <Ti,Xj,Vold,Vnew>

   Ti = Transaction id

   Xj = Data item

   Void = Old value

   Vnew = New value
4. For example:

   There are two accounts 'A' and 'B'. Account 'A' contains Rs.2000. Account 'B' contains Rs.1000.Account 'C' contains Rs.3000.

   The transaction Tl transfers Rs. 100 from account 'A' to account [C]B'. The transaction T2 withdraws Rs. 100 from account 'C'.

During Tl		
Tl,	**Log**	**Database**
Read (A, a)		
a = a- 100	<Tl,Start>	
Write (A, a)	<T1,A,2000? 1900>	.A = 1900
Read (B, b)		

B = b + 100	<T1,B, 1000, 1100>	B = llOO
Write (B, b)		
	<T1, Commit>	
During Transaction T2		
T2		
Read(C, e)	<T2,Start>	
c = c - 200	<T2, C? 4000, 3800>	C = 3800
* Write (C, c)		
	<T2?Commit>	

Example 6.3: Example of Immediate Database Modification

6.10.2 Shadow Paging

1. Database is divided into blocks.
2. Blocks are of fixed lengths.
3. We use a page table. The page table has entries for each database table.
4. Shadow paging technique use two page tables during transaction execution.
   - Current page table
   - Shadow page table
5. At beginning of transaction both pages are identical.
6. Each page table entry contains a pointer to a page on disk.
7. Current page table may change during write operation.
8. Shadow page table never change during transaction.
9. There are two cases of recovery in the shadow paging technique which are as follows:
   - ***Case 1:*** If system crashes before the successful completion of transaction, then current page table will removed. This is just like undo operation.lt means if system fail before commit transaction then it will get the previous state of data from shadow table.

- ***Case* 2.** If system crashes after the successful completion of transaction, then current page table will become the shadow page table. It means if system fails after commit transaction then it will recover the data from shadow table,

6.11 DISASTER MANAGEMENT

1. *"A disaster can be defined as an occurrence either nature or manmade that causes human suffering and creates human needs that victims cannot alleviate without assistance".*

2. ***There are four main types of disaster as follows:***

   - **Natural disasters:** Natural disasters include floods, hurricanes, earthquakes and volcano eruptions that can have immediate impacts on human health.
   - **Environmental emergencies:** Environmental emergencies include technological or industrial accidents, usually involving hazardous material, and occur where these materials are produced, used or transported.
   - **Complex emergencies:** Complex emergencies include a break-down of authority, looting and attacks on strategic installations. Complex emergencies include conflict situations and war.
   - **Pandemic emergencies:** Pandemic emergencies include a sudden onset of a contagious disease that affects health but also disrupts services and businesses, bringing economic and social costs.

Disaster Management

1. Disaster management is also known as emergency management.
2. It avoids both natural and man-made disasters.
3. *"Disaster management can be defined as the organization and management of resources and responsibilities for dealing with all humanitarian aspects of emergencies, in particular preparedness, response and recovery in order to lessen the impact of disasters."*

4. It involves preparedness before disaster, rebuilding and supporting society after natural disasters such as, earthquakes, drought, tsunami etc.
5. The following are the techniques to manage disaster:
    - **Disaster Management Teams:** Worldwide, governments, business and non-business organization are setting up disaster or crisis management teams in order to manage the disaster. The disaster management teams are broadly divided into three parts namely: the policy team, the management team and the liaison team.
    - **Systematic Planning:** Disaster management involves systematic planning to avoid a disaster. If disaster occurs, then systematic planning is required to overcome the crisis arising out of disaster. It indicates, what to do, when to do, how to do and who is to do certain activities to manage and overcome the problems of disaster.
    - **Training to Manpower:** There is a need to provide proper training to the disaster management personnel to manage a disaster effectively. The training will help to develop and improve the disaster management skills in the people.
    - **Suitability:** Disaster management is required before and after a disaster. It is suitable before a disaster in order to avoid a disaster, or to caution the people. It is also very much required after a disaster takes place, in order to undertake rescue, relief and rehabilitation measures at the time of floods, earthquakes.
    - **Stability:** Normally, disaster management teams lack stability. They are formed just prior to a disaster in order to avert it, whenever possible. But there should be some permanent disaster management teams.

Questions

1. What is Concurrency Control? Why is it required?
2. Explain the Concurrency Control schema based on timestamp protocol.
3. What are AICD properties of transaction? Explain the uses of each.
4. What are the different locking techniques for Concurrency Control?
5. What is a lock? Differentiate between exclusive and shared lock. Give suitable examples also.
6. What is time stamping?
7. Explain Concurrency Control without locking.
8. What is deadlock? Give an appropriate example.
9. Explain the following:
   (a) Deadlock prevention
   (b) Deadlock detection
10. What are the necessary conditions for deadlock?
11. Define Database Security. What are the various issues addressed by it?
12. What are the different security mechanisms?
13 What are the different access rights that may be given to the users of a database?
14 What do you mean by data encryption? How is it achieved?
15 Explain the following with suitable example:
   (a) Entity Integrity Constraint.
   (b) Domain integrity constraint
16 What is the role of DBA in security?
17. What is the concept of recovery in database? What are the techniques used for it.
18 What is deferred update and immediate update?
19 What is shadow paging?
20 Explain transaction rollback.
21 Explain the recovery and atomicity of transaction in detail with suitable examples.
22 What is log based recovery?
23 Discuss the disaster management in detail.

COURSE: DBMS

UNIT 7: SQL

7.1 INTRODUCTION TO SQL

7.2 INTRODUCTION TO SQL*PLUS

7.3 DIFFERENCE BETWEEN SQL AND SQL*PLUS

7.4 STARTING SQL*PLUS

7.5 DATATYPES

7.6 PARTS OF SQL

7.6.1 DDL (Data Definition Language)

7.6.2 DML (Data Manipulation Language)

7.6.3 DCL (Data Control Language)

7.6.4 TCL (Transaction Control Language)

7.6.5 Embedded SQL

7.6.6 Integrity

7.7 SQL OPERATORS

7.7.1 Arithmetic Operator

7.7.2 Comparison Operator

7.7.3 Logical Operator

7.7.4 Set Operator

7.7.5 Concatenation Operator

7.8 SQL FUNCTIONS

7.8.1 Single Row Functions

7.8.2 Group/Aggregate Functions

7.9 JOINS

7.9.1 Equi Join

7.9.2 Cross Join

7.9.3 Outer Join

7.9.4 Self Join

7.10 ROLL UP OPERATION

7.11 CUBE OPERATION

7.12 NESTED QUERY

7.13 SUBQUERY

7.14 VIEW

7.14.1 Creating a View

7.14.2 Modifying a View

7.14.3 Inline View

7.14.4 Materialized View

7.14.5 Dropping a View

7.14.6 Advantages of View

7.15 DISADVANTAGES OF SQL

7.16 KEY POINTS

7.16.1 SQLAlias

7.16.2 Null Values in SQL

7.16.3 Difference between Delete and Truncate Command

7.2 INTRODUCTION TO SQL

- SQL stands for "Structured Query Language".
- It is a widely used database language.
- It has been adopted as the standard relational database language.
- It can be pronounced as "SQL" or "SEQUEL".

- It was first introduced as a commercial database system in 1979 by Oracle Cooperation.
- SQL is different from other programming languages like C, C++, Java, Visual Basic etc.
- Unlike other languages, there is no need to specify the sequence of steps to perform any particular task, SQL statements directly provides the desired result.
- It is a non-procedural, pure English language rather than coding language but it has fixed syntax (structure).
- It processes set of records rather than one record at a time.
- It is made up of various commands and used to define, access and manipulate data in RDBMS.
- It is just not for query the database but it can do much more.
- All the programs written in SQL are portable. They can be moved from one database to another with little modification.\
- All the major relational database management system support SQL. SQL has proved to be very effective for heavy databases.

Functions/Tasks of SQL

- SQL can define the structure of database.
- It can execute the queries against a database.
- It can create new databases.
- It can create new tables in a database.
- It can insert records in a database.
- It can update records in a database.
- It can delete records in a database.
- It can retrieve data from a database.
- It can create views and stored procedures in a database.
- It can set permissions for users so that they can use the database.
- It can specify the security constraints in the database.
- It can create, replace and alter the objects.
- It guarantees the database consistency and language.

7.2 INTRODUCTION TO SQL*PLUS

- SQL*PLUS is command line tool.
- It allows user to type SQL statements to be executed directly against an Oracle database.

- With the help of SQL*PLUS, a user can perform the following tasks:
- User cap access Oracle databases with command procedures.
- User can interactively use the SQL commands (Enter, Edit, Store and Retrieve).
- User can produce reports.
- Access and copy data between SQL databases.
- Send messages and accept responses from an end user.
- List columns of any table.

7.3 DIFFERENCE BETWEEN SQL AND SQL*PLUS

Sr. No.	SQL	SQL*PLUS
1	It is a standard language to access RDBMS (Relational database management system)	It is a tool to implement SQL. It has a command line interface.
2	SQL commands are terminated by semicolon (;).	SQL*PLUS commands do not need semicolon (;) for termination. It only needs continuation character (-).
3	Data manipulation is possible in SQL; Data manipulation language/ command (DML) is used to modify the data.	Data manipulation is not possible in SQL*PLUS.
4	Commands are stored in buffer.	Commands are not stored in the buffer.
5	Commands are entered in one line or more than one line.	Commands are entered in one line at a time.
6	For formatting purpose, SQL use various functions.	For formatting purpose, SQL*PLUS use various formatting commands.
7	SQL commands are to be executed again and again.	SQL*PLUS commands remain in effect until any new command overwrite it.

7.4 STARTING SQL*PLUS

We can start the SQL*PLUS with either of the two methods. The following steps should be considered to start the SQL*PLUS:

Method 1

Step 1: Locate the **SQL PLUS** (Oracle Shortcut) on the desktop.

Step 2: Press **<Enter>** or **'double clock'** on the 'Oracle Shortcut'.

Method 2

If the SQL PLUS (Oracle Shortcut) is not available on the desktop, then follow the following steps:

Step 1: Click on **'Start'** □ **'All Programs'** □ **'Oracle-OraDb10g_home3'**□ **'Application Development'**□ **'SQL PLUS'.**

Step 2: Enter the **'User Name'** in the dialog box. The **'User Name'** should be 'Scott'.

Step 3: Enter the **'Password'** in the dialog box. The **'Password'** should be **'Tiger'**.

Step 4: Enter the **'Host String'** in the dialog box.

Step 5: Press the **"OK"** button to complete the Oracle log in process, then the SQL*PLUS prompt (SQL>) will appear.

7.5 DATATYPES

- Oracle uses the tables for storing and maintaining the information. Table consists of rows and columns.
- Each column contains only one type of data which we must define.
- A data type is an attribute that specifies the type of data.
- In short, a data type is a classification of a particular type of data (information).
- When we create a table, we must specify a data type for each of its columns.
- The following table show a list of different data types commonly used in Oracle:

Sr. No.	Data Type	Description
1	Char (n)	A fixed length character string with user specified length. The characters can be used in it. n = number of characters
2	Varchar (n)	A variable length character string with user specified length. n = number of characters
3	Int	A numeric value can be specified. Character cannot insert.

4	Numeric (P,D)	A floating number can be inserted. P indicate total digits. Whereas D indicate number of digits after decimal. For example: (3,1) allows 44.5 to be inserted but not 444.5 or 0.32.
5	Float (n)	A floating number with decimal value up to n number.
6	Date	It can contain date with year, month and day of month.
7	Time	It can contain time in hours, minutes and seconds.

7.6 PARTS OF SQL

- SQL consists of various commands. SQL commands are instructions used to perform the various operations like create, delete and manipulate the data in database.
- SQL commands are helpful in searching the data, drop the table, add data to table, set permissions for users etc.
- SQL is an easy, understandable and unified language.
- SQL languages/commands/statements are categorized into six major parts based on their functionality which are as follows:

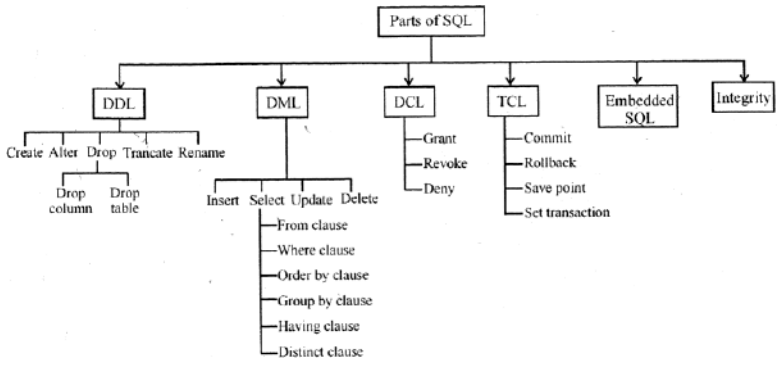


Fig. 10.1: Parts of SQL

7.6.1 DDL (Data Definition Language)

It is used for defining data structures. These SQL commands are used for creating, modifying and dropping the structure of database objects (relations).

These commands basically create, modify and drop the relations (tables) used in the database.

The following are the various DDL commands:

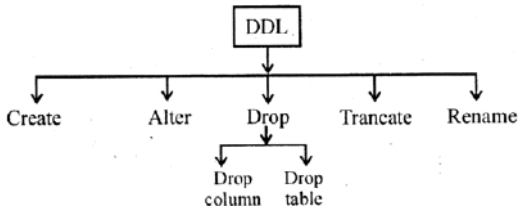


Fig. 10.2: Parts of DDL

1. Create: The create table command is used to create a new table. It creates the relation (table) in a database. It includes its name, names and attributes of its columns. One can create any number of columns with this command. If we want to add or remove the columns after creating the table then we use alter table.

Syntax of Create New Table:->

SQL>CRE^VTE TABLE **table_name**

(

column_name1 data type,

column_name2 data type,

…….

column_nameN datatype

);

Note: We can also create a table from existing table by copying the existing table's column.

Syntax of Create Table from Existing Table:->

SQL> CREATE TABLE new_table

As (SELECT * from old_table);

Examples of Create Command:-

1. We want to create a table 'STUD' in SQL.

   Then the query will be:

   SQL> CREATE TABLE STUD

   (

   NAME char (40),

   CLASS char (5),

   ROLL NUMBER (8)

   );

   Table created

2. We want to create a table 'BMP' in SQL. ***(Mostly queries of this book are based on this table 'EMP')***

SQL> CREATE TABLE EMP

(

ENAME char (15),

DEPTNO int,

JOB char (10),

EMPNO int,

SAL int,

HIREDATE int,

MGR int,

CITY char (10),

COMM int

);

Table created

2. Alter: It alters the structure of table from database. It alters the table along with the columns. One can add one more than one column in a particular table with alter command. With this command, filed type can be changed or a new field can be added. It is used to enable or disable the integrity constraint. It is used to modify the column values and constraints.

Syntax of Alter Command:

SQL> ALTER TABLE table_name

ADD/MODIFY/DROP column_name datatype;

Examples of Alter **Command:**

1. To add a column (DOB) in an existing table 'BMP'. Then the query will be:

   SQL> ALTER TABLE EMP

   ADD DOB date;

   Table altered

2. To add multiples columns (DOB and MOBNO) to an existing table 'EMP'. Then the query will be:

   SQL>ALTER TABLE EMP

   ADD (DOB date, MOBNO (11));

Table altered

3. Drop: With the drop command, we can drop the columns from table or we can remove the table. It drops the column or constraints from the table. It deletes the string of a table. It cannot be recovered. It use with caution. Drop operation is used with the alter table command. It removes single column or multiple columns.

(a) Dropping Column: If we want to remove column, then we use drop operation with alter table command.

Syntax of Dropping the Column:

SQL>ALTER TABLE table_name

DROP COLUMN column_name;

Examples of Dropping the Column:

1. To drop a column 'City' in an existing table 'EMP'. Then the query will be:

   SQL>ALTER TABLE EMP

   DROP COLUMN CITY;

   Table altered.

2. To drop multiple columns (Hiredate and City) in an existing table 'EMP'. Then the query will be:

   SQL>ALTER TABLE EMP

   DROP COLUMN (HIREDATE, CITY);

   Table altered.

(b) Dropping Table: If we want to remove the table, then there is no need to use it with alter table command. We can directly remove one or more columns with drop table command.

- This command removes one or more table definitions and all data, indexes, triggers, constraints and permission specifications.
- If we drop a table with drop table command, it deletes all rows from that particular table. The table structure is also removed from the database and it cannot get back.

Syntax of Dropping the Table

SQL> DROP TABLE table_name;

4. Truncate: It removes all the records from a table and memory. It releases the memory occupied by the records of the table. Data cannot be recovered after using the truncate command.

Truncate command removes all the rows from a table.

Syntax of Truncate Command:

SQL> TRUNCATE TABLE table_name;

Example of Truncate Command:

We want to delete all rows from the table 'EMP'. Then the query will be:

SQL> TRUNCATE TABLE EMP;

5. Rename: It is used to rename the old table with a new name. The data will remain same, only name of table will be change with 'Rename Command'.

Syntax of Rename Command:

SQL> RENAME <Old Table_Name>to<New Table_Name>;

Example of Rename Command:

If we want to change the name of table 'EMP' to new name 'EMPLOYEE'. Then the query will be:

SQL>RENAME EMP TO EMPLOYEE;

Note: We use drop command for tables and delete command for records.

7.6.2 DML (Data Manipulation Language)

- These commands are used for inserting, retrieving, deleting and modifying the data in a relation or a table.
- It includes the query language based on both relational algebra and tuple relation.
- These commands do not implicitly commit the current transaction.
- The folio wing are the various DML commands:

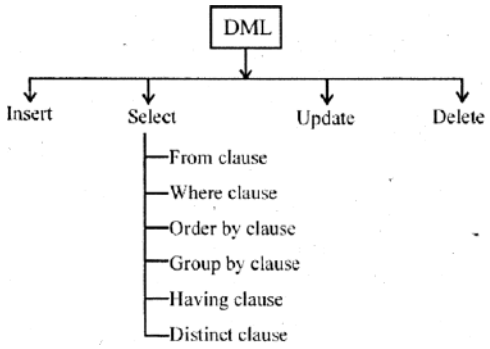


Fig. 10.3: Parts of DML

1. Insert

- When a new table is created, there is no data in the table.
- Insert command is used to insert the records in the new table.

- Insert command is used to add records to an existing table.
- 'Values clause' is used with inset command. This command will insert value in all the columns of a table in sequence.

Syntax of Insert Command:

SQL> INSERT INTO table_name

VALUES (value1, valueZ, valueS,. ...);

OR

SQL> INSERT INTO table_name (column1, column2, column3,.....)

VALUES (value1, value2, value3,..);

Examples of Insert Command:

1. Insert record in different order. Then the query will be:

   SQL> INSERT INTO EMP (name, city, salary, emp_no)

   VALUES ('Mona','Nba', 4500, 4);

2. Insert the Null value in record. Then the query will be:

   SQL> INSERT INTO EMP

   VALUES (3,'Mona', Null, 4000);

3. Insert the records in selected columns. Then the query will be:

   SQL> INSERT INTO EMP (name, city)

   VALUES ('Mona', 5000);

4. Insert the values in the table 'EMP'. Then the query will be:

   SQL> INSERT INTO EMP VALUES ('Nidhi',20,'Clerk',6258,900,9-5-83, 6801,'Chd');

   SQL> INSERT INTO EMP VALUES ('Aastha',30,'SaIesman',6388,1500,1-12- 89, 6587, 'Delhi', 300);

   SQL> INSERT INTO EMP VALUES **('Sachin',30,'Salesman',6410,1350,25-1-92,6587,'Pta',500);**

   SQL> INSERT INTO EMP VALUES ('Rohit',20,'Manager',6455,2875,27-12-91,6728,'Nba');

   SQL> INSERT INTO EMP VALUES **('Rahul',30,'Salesman',6543,1350,28-5-87,6587,'Nba',1400);**

   SQL> INSERT INTO EMP VALUES ('Aditya',30,'Manager',6587,2750,17-8-

86,6728,'Pta');

SQL> INSERT INTO EMP VALUES ('Siddharth',10,'Manager',6671, 2550,29-9-80,6728,'Chd',Null);

SQL> INSERT INTO EMP VALUES **('Kunar,20,'Analyst',6677,3000,8-12-82, 6455,'Delhi',Null);**

SQL> INSERT INTO EMP VALUES ('AkhiP,10,'President',6728,5000,2-11-85,Null,'DeIhi',NulI);

SQL> INSERT INTO EMP VALUES ('Prathiba',30,'Salesman',6733,1600,4-6-85,6587,'Pta',0);

SQL> INSERT INTO EMP VALUES ('Manmeet',20,'Clerk',6765,1050,11-1-84,6677;'Ldh',Null);

SQL> INSERT INTO EMP VALUES ('Navreet',30,'Clerk',6800,950,25-3-84,6587,'Pta',Null);

SQL> INSERT INTO EMP VALUES ('Saira',20,'Analyst',6801,3000,15-4-80,6455,'Chd',Null);

SQL> INSERT INTO EMP VALUES ('Amit',10,'Clerk',6823,1400,25-8-85,6671,'Ldh',Null);

After inserting, values, the table 'EMP' will look like:

EMP

ENAME	DEPTNO	JOB	EMPNO	SAL	HIREDATE	MGR	CITY	COMM
Nidhi	20	Clerk	6258	900	9-5-83	6801	Chd	
Aastha	30	Salesman	6388	1500	1-12-89	6587	Delhi	300
Sachin	30	Salesman	6410	1350	25-1-92	6587	Pta	500
Rohit	20	Manager	6455	2875	27-12-91	6728	Nba	
Rahul	30	Salesman .	6543	1350	28-5-87	6587	Nba	1400

Aditya	30	Manager	6587	2750	17-8-86	6728	Pta	
Siddharth	10	Manager	6671	2550	29-9-80	6728	Chd	
Kunal	20	Analyst	6677	3000	8-12-82	6455	Delhi	
Akhil	10	President	6728	5000	2-11-85		Delhi,	
Prathiba	30 :	Salesman	6733	1600	4-6-85	6587	Pta	0
Manmeet	20	Clerk	6765	1050	11-1-84	6677	Ldh	
Navrget	30	Clerk	6800	950	25-3-84	6587	Pta	
Saira	20	Analyst	6801	3000	15-4-80	6455	Chd	
Amit	10	Clerk	6823	1400	25-8-85	6671	Ldh	

NOTE: ***(Mostly queries of this book are based on this table*** *'EMP')*

2. Select: Once data in inserted into a table, the next step is to view the data contained in the table.

- In order to view the data contained in the table, the select statement is used.
- Select statement is a powerful tool and a most commonly used command.
- It is used to retrieve the data from a table in a database.
- We can also use arithmetic operators in select statement (see example 4, 5 and 6 of select statement).
- With the help of select command, one can retrieve information from one column or more than one column.
- The basic select statement has 6 clauses which are as follows:

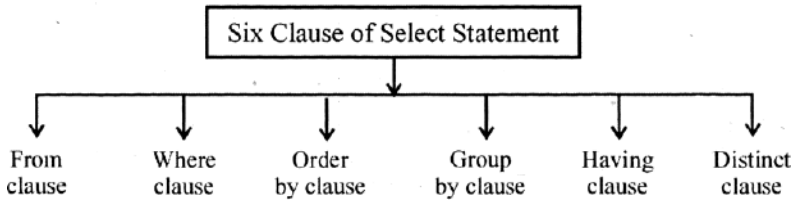


Fig. 10.4: Six Clauses of Select Statement

(a) Select: The select clause specifies the table columns that are retrieved. It always use with 'From Clause'.

Syntax of Select Command:

SQL> SELECT * FROM table_name;

OR

SQL> SELECT column_list FROM table_name

[WHERE Clause]

[GROUP BY Clause]

[HAVING Clause]

[ORDER BY Clause];

(b) From: From clause specifies the table accessed. It is mandatory. It always use with 'Select Command'.

Syntax of From Clause:

SQL> SELECT.* FROM table_name;

OR

SQL> SELECT column_list FROM table_name

[Where Clause]

[Group By Clause]

[Having Clause]

[Order By Clause];

(c) Where: Where clause is used when we want to retrieve the specific information from a relation excluding other irrelevant data.

Syntax of Where Clause:

SQL> SELECT column_list FROM table_name

[WHERE Clause];

Examples of'-Select Command', 'From Clause' and 'Where Clause':

1. Display all the information of all the employees from relation 'EMP'. Then the query will be:

   SQL> SELECT * FROM EMP;

Result:

BMP

ENAME	DEPTNO	JOB	EMPNO	SAL	HIREDATE	MGR	CITY	COMM
Nidhi	20	Clerk	6258	100	9-5-83	6801	Chd	
Aastha	30	Salesman	6388	1500	1-12-89	6587	Delhi	300
Sachin	30	Salesman	6410	1350	25-1-92	6587	Pta	500
Rohit	20	Manager	6455	2875	27-12-91	6728	Nba	
Rahul	30	Salesman	6543	1350	28-5-87	6587	Nba	1400
Aditya	30	Manager	6587	2750	17-8-86	6728	Pta	
Siddharth	10	Manager	6671	2550	29-9-80	6728	Chd	
Kunal	20	Analyst	6677	3000	8-12-82	6455	Delhi	
Akhil	10	President	6728	5000	2-11-85		Delhi	
Prathiba	30	Salesman	6733	1600	4-6-85	6587	Pta	0
Manmeet	20	Clerk	6765	1050	11-1-84	6677	Ldh	
Navreet	30	Clerk	6800	950	25-3-84	6587	Pta	
Saira	20	Analyst	6801	3000	15-4-80	6455	Chd	
Amit	10	Clerk	6823	1400	25-8-85	6671	Ldh	

2. Display only the name, job and salary of all the employees from table "EMP[55]. Then the query will be;

SQL> Select ENAME, JOB, SAL

From EMP;

Result:

ENAME	JOB	SAL
Nidhi	Clerk	900
Aastha	Salesman	1500
Sachin	Salesman	1350
Rohit	Manager	2875
Rahul	Salesman	1350
Aditya	Manager	2750
Siddharth	Manager	2550
Kunal	Analyst	3000
Akhil	President	5000
Prathiba	Salesman	1600
Manmeet	Clerk	1050

ENAME	JOB	SAL
Navreet	Clerk	950
Saira	Analyst	3000
Amit	Clerk	1400

14 rows selected.

3. Display name, city and salary of employees from relation 'EMP' where salary of each employee is increased by 1000. Then the query will be:

SQL> SELECT ENAME, CITY, SAL + 1000

FROM EMP;

Result:

ENAME	CITY	SAL+1000
Nidhi	Chd	1900
Aastha	Delhi	2500
Sachin	Pta	2350
Rohit	Nbh	3875
Rahul	Nbh	2350
Aditya	Pta	3750
Siddharth	Chd	3550
Kunal	Delhi	4000
Akhil	Delhi	6000
Prathiba	Pta	2600
Manmeet	Ldh	2050

ENAME	CITY	SAL+1000
Navreet	Pta	1950
Saira	Chd	4000
Amit	Ldh	2400

14 rows selected.

4. Display the name and salary of employees whose salary is less than 5000. Then the query will be:

SQL> SELECT ENAME, SAL from EMP

WHERE SAL <5000;

Result:

ENAME	SAL
Nidhi	900
Aastha	1500
Sachin	1350
Rohit	2875
Rahul	1350
Aditya	2750
Siddharth	2550
Kunal	3000
Prathiba	1600
Manmeet	1050
Navreet	950

ENAME	SAL
Saira	3000
Amit	1400

13 rows selected.

5. Display the names of all the employees belonging to the department number 10 from the relation 'BMP'. Then the query will be:

SQL>SELECT ENAME FROM EMP

WHERE DEPTNO = 10;

Result:

ENAME
Siddharth
Akhil
Amit

(d) Order By: The 'Order By Clause' is used with 'Select Statement' to sort the results either in ascending or descending order. By default, it provides results in ascending order. We use column values to sort the table. We can use more than one column to sort the results.

Syntax of Order By Clause:

SQL> SELECT column_list FROM table_name

[ORDER BY Clause];

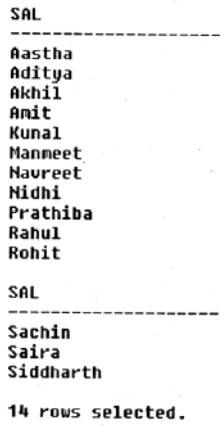
Examples of Order By Clause:

1. Sort the table 'EMP' by the salary of employees. Then the query will be:

SQL>SELECT ENAME SAL FROM EMP

ORDER BY SAL;

Result:

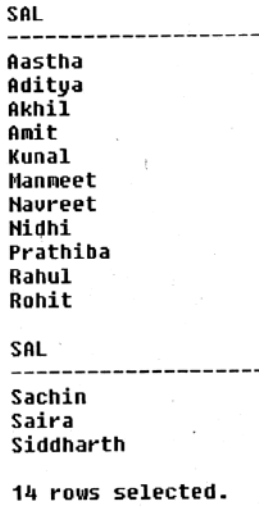


2. Sort the table 'BMP', by the name and salary of employees. Then the query will be:

SQL>SELECT ENAME SAL FROM EMP

ORDER BY ENAME, SAL;

Result:



(e) Group By: It is used to divide the rows into smaller groups. The 'Group By Clause' is used with 'Select Statement' to combine a group of rows based on the values of a particular column or expression. It groups the result after it retrieves the rows from a table. 'Group functions' can be used with 'Having Clause' and cannot be used with 'Where Clause'.

Syntax of Group By Clause:

SQL> SELECT column_list FROM table_name

[GROUP BY Clause];

Example of Group By Clause:

To find the total amount of salary spent on each department from the table 'EMP'. Then

the query will be:

SQL>SEI.ECT DEPTNO, SUM (SAL) AS TOTAL SALARY FROM EMP

GROUP BY DEPTNO;

Group within Group: 'Group By Clause' can be used to provide results for 'Groups Within Groups'. Suppose we want to know the average amount of salary spent on job type 'Clerk' from department number '20'. We calculate the total amount of salary spent on each department. This is one group. Then we calculate the average amount of salary spent on each type of job from that particular department. This is group within group.

Example of Group within Group Clause:

To find the average monthly salary for each job type within department Then the query will be:

SQL>SELECT DEPTNO, JOB, AVG (SAL) AS AVERAGE SALARY FROM EMP

GROUP BY DEPTNO, JOB;

(f) Having: It is similar to 'Where Clause', but it is used with group functions. It is used to filter the data. 'Having Clause' can be used with 'Group function' and cannot be used with 'Where Clause'. It restricts the groups that we return on the basis of group functions. It is used to specify which groups are to be displayed.

Syntax of Having Clause:

SQL> SELECT column_list FROM table_name

[HAVING Clause];

Example of 'Having Clause:

To find the department who has paid the total salary more than 8.00.6 to its employees. Then the query will be:

SQL>SELECT DEPTNO, SUM (SAL) AS TOTAL SALARY FROM EMP GROUP

BY DEPTNO

HAVING SUM (SAL)>8000;

(g) Distinct Clause: The 'Distinct Clause' is used with 'Select Statement' to suppress the duplicate values if any in a column.

Example of 'Distinct Clause':

Display all the different jobs available in the table 'EMP'. Then the query will be:

SQL>SELECT DISTINCT JOB FROM EMP;

Result:

```
JOB
---------------------------
President
Clerk
Analyst
Salesman
Manager
```

3. Update

- Update command is used when there is a need to modify the data in a table.
- It is used to update existing records in a table.
- It updates single record or multiple records in a table.

Syntax of Update Command:

SQL> UPDATE table_name

SET column1 = value, column2 = value2, ...

WHERE some_column = some_value;

Examples of Update Command:

1. To give everybody a commission of Rs. 100 in the table 'EMP'. Then the query will be:

   SQL>UPDATE EMP

   SET COMM = 100;

2. Update the Manager's salary to 8000 of department number 10 in the table 'EMP'. Then the query will be:

   SQL>UPDATE EMP

   SET SAL = 8000

   WHERE JOB = 'Manager' AND DEPTNO = 10;

4. Delete

- It deletes one or more records from a table and sends it to recycle.
- It doesn't release the memory occupied by the records of the table. Data can be recovered.
- If any subset is defined with condition, then specific records or rows, are deleted, otherwise all records are deleted.
- Executing a delete command may cause triggers to rum which may cause deletion in other tables.
- Example: Sometimes two tables are linked by the foreign key. If we delete rows in one table, then we have to delete those rows from the second table to maintain the referential

integrity.

Syntax of Delete Command:

SQL> DELETE FROM table_name [where condition];

OR

SQL> DELETE from table_name;

Examples of Delete Command:

1. Delete all the records of 'Manager' from the table 'EMP'. Then the query will be:

   SQL>DELETE FROM EMP

   WHERE JOB = 'Manager';

2. Delete all the records from the table 'EMP'. Then the query will be:

   SQL>DELETE FROM EMP;

7.6.3 DCL (Data Control Language)

- It is used to control access to data in a database. It also controls the security of the database.
- To control data in a database, privileges are given to user to access the data without any problem and with proper security.
- It basically provides security to database. Without privileges, no one can access the database.
- A user can access the database according to the privileges given to him.

The following are the various DCL commands:

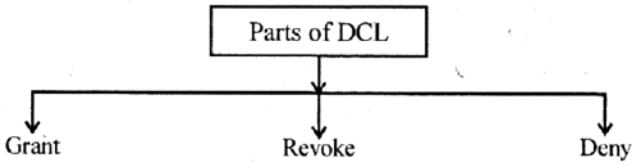


Fig. 10.5: Parts of DCL

(a) Grant: It is used to give the permission to the user for restricted access to the database. It allows specified users to perform specified tasks.

(b) Revoke: It is used to cancel the previously granted or denied permissions to the users.

(c) Deny: It disallows the specified users from performing specified tasks.

7.6.4 TCL (Transaction Control Language)

- TCL is used to manage the changes made by DML (data manipulation language) statements.

- These commands are used for revoking the transactions and to make the data commit to the database.
- Basically, it is used to manage the different transactions occurring within a database.
- Each transaction is completely isolated from other active transactions.
- User can make changes in the particular transaction in database with the transaction control language.
- At the end of the transaction, the database can make all the changes permanent in the database or undoes them all.
- If any problem fails in the middle of a transaction, then the database rolls back the transaction and restore the database into its former state.
- The following are the various TCL commands:

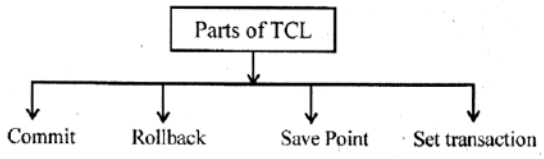


Fig. 10.6: Parts of TCL

(a) Commit

- Commit command is used to save work done. The changes made in the database by the user are not visible to other users until they become permanent in the database.
- Commit command is used to permanent any changes made to the database during the current transaction by the user.
- Commit command is used to save all the changes made to the database since the last commit or rollback command.

Syntax of Commit Command:

SQL> COMMIT;

Example of Commit Command:

To delete the records of the employees permanently, belonging to the city 'Chd'.

SQL>DELETE FROM EMP

WHERE CITY = 'Chd';

SQL>COMMIT;

(b) Rollback

- It is used to restore the database to its original state since the last 'commit'.

- It is the inverse of the commit statement.
- It is used to undo the transactions that have not already been saved to the database.
- Oracle provides a facility to-roll back to the last committed state.

Example: We are performing the operations on the database and some problem occurs into the computer system. Yet we have not performed the commit statement, and then rollback command helps to come back to the last committed state.

Syntax of Rollback Command:

SQL> ROLLBACK;

(c) Savepoint

- Savepoint command is used to identify a point in a transaction from which we can later rollback.
- The Savepoint statement defines a Savepoint within a transaction.
- It is a special mark inside a transaction that allows all commands that are executed after it was established to be rolled back, restoring the transaction state to what it was at the time of Savepoint.
- Changes made after a Savepoint can be undone at any time prior to the end of the transaction.
- A transaction can have multiple savepoints.

Syntax of Savepoint Command:

SQL> SAVEPOINT<savepoint name>;

(d) Set Transaction

- Set transaction command has no effect on any subsequent transactions.
- It is used to set the characteristics of the current transaction.
- This command is helpful to determine whether the transaction is read/write or read only.
- If a transaction is read only, then the insert, update, delete and copy commands are disallowed.

7.6.5 Embedded SQL

- Embedded SQL define how SQL statements can be embedded within general purpose programming language like C, C++, Java.
- All the SQL statements DDL, DML and DCL can be grouped into one body. Embedded SQL refers to the use of standard SQL commands embedded within a procedural

programming language.

- When the embedded statements of SQL are execute then all the statements in the body will be executed automatically.

Some of the embedded SQL statements are:

(a) ***Define:*** Define cursor

(b) ***Open:*** Open cursor

(c) ***Execute:*** To execute the command or SQL prompt.

7.6.6 Integrity

- SQL DDL includes commands for integrity constraints so that the data store in the database must satisfy the condition.

7.7 SQL OPERATORS

- SQL supports a wide variety of operators. These operators are extensively used in SQL statements used by the user for the purpose of issuing a query to the database.
- The operators are mainly used in the Where clause, Having clause to filter the data to be selected.
- An operator is a symbol which is used to manipulate the data items (operands).

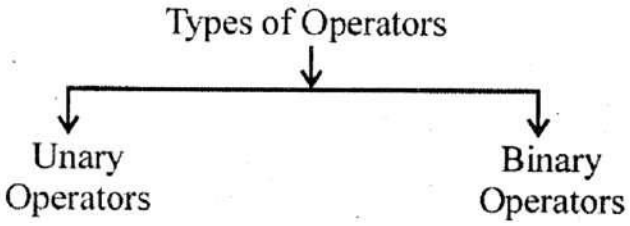


Fig. 10.7: Types of Operators

- Operators are represented by keywords or by special characters.

On the basis-of operands, there are two types-of operators:

Unary Operator: *An* unary, operator operates on only one operand.

Format □ *operator* operand.

Binary Operator: *A* binary operator operates on two operands.

Format □ operand1 *operator* operand 2

The following are the various SQL operators:

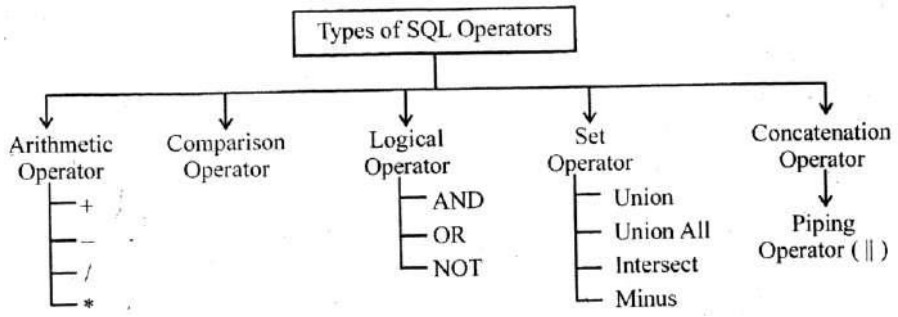


Fig. 10.8: Types of SQL Operators

7.7.1 **Arithmetic Operator**

- An arithmetic operator is used to add, subtract, multiply and divide the numeric values in an expression.
- It is used to perform the mathematical operations on one or more data items or operands of numeric data type.
- It also provides results in numeric values.

Sr. No.	Arithmetic Operator	Description
1	+	Used for addition in SQL
2	-	Used for subtraction in SQL
3	/	Used for division in SQL
4	*	Used for multiplication in SQL

Examples of Arithmetic Operator:

1. Add

Add Rs.500 in the employee's salary whose EMPNO is 6258 from the relation 'EMP'. Then the query will be:

```
SQL> SELECT SAL, SAL+500 FROM EMP
        WHERE EMPNO = 6258;
```

Result:

```
SAL        SAL + 500
----------------------------
900        1400
```

2. Subtract

Subtract the employee's commission from his salary whose EMPNO is 6388. Then the query will be:

```
SQL> SELECT SAL, SAL-COMM FROM EMP
```

WHERE EMPNO = 6388;

Result:

```
SAL        SAL-COMM
-----------------------------
1500       1200
```

3. Multiply

Multiply the salary of employee by 100 whose EMPNO is 6258 from the relation 'EMP'. Then the query will be:

SQL> SELECT SAL, SAL* 100 FROM EMP

WHERE EMPNO = 6258;

Result:

```
SAL        SAL * 100
-----------------------------
900        90000
```

7.7.2 **Comparison Operator**

- A comparison operator is used to compare the column data with specific values with the other column data values.
- It is also used along with the Select Statement to filter data based on specific conditions.

Sr. No.	Comparison Operator	Description
1	=	Equal to
2	!= OR o	Not equal to
3	<	Less than
4	>	Greater than
5	<=	Less than or equal to
6	>=	Greater than or equal to
7	LIKE	Performs pattern matching from columns. The LIKE operator is- used only with Char and match a pattern. % represents sequence of zero or more character.
8	IN	To check a value within a set. It is used to compare a column with more than one value.

9	BETWEEN	To check value within a range. It is used to compare data for a range of value.
10	ANY	To check whether one or more rows in the result set of a sub query meet the specified, condition
11	ALL	To check whether all rows in the result set of a sub query meet the specified condition.
12	EXISTS	To check whether a sub query returns any result.

Example of Equal to (=) Operator:

Display the records of the employees, who live in city 'Chd', from the relation 'EMP'. Then the query will be:

SQL> SELECT * FROM EMP

WHERE CITY = 'Chd';

Result:

ENAME	DEPTNO	JOB	EMPNO	SAL	HIREDATE	MGR	CITY
Nidhi	20	Clerk	6258	900	9-5-83	6801	Chd
Siddharth	10	Manager	6671	2550	29-9-80	6728	Chd
Saira	20	Analyst	6801	3000	15-4-80	6455	Chd

Example of Not Equal to (!= OR <>) Operator:

Display the records of the employees, whose city is not equal to 'Chd', from the relation 'EMP'. Then the query will be:

SQL> SELECT * FROM EMP

WHERE CITY! = 'Chd';

Result:

ENAME	DEPTNO	JOB	EMPNO	SAL	HIREDATE	MGR	CITY	COMM
Aastha	30	Salesman	6388	1500	1-12-89	6587	Delhi	300
Sacliin	30	Salesman	6410	1350	25-1-92	6587	Pta	500

Rohit	20	Manager	l3455	2875	27-12-91	6728	Nba	
Rahul	50	Salesman	6543	1350	28-5-87	6587	Nba_	1400
Aditya	30	Manager	6587	2750	17-8-86	6728	Pta	
Kunal	20	Analyst	6677	3000	842-82	6455	Delhi	
Akhil	10	President	6728	5000	2-11-85		Delhi	
Prathiba	30	Salesman	6733	1600	4-6-85	6587	Pta	0
Manmeet	20	Clerk	6765	1050	114-84	6677	Ldh	
Navreet	30	Clerk	6800	950	25-3-84	6587	Pta	
Amit	10	Clerk	6823 .	1400	25-8-85	6671	Ldh	

11 rows selected

Example of Less than (<) Operator:

Display the name of the employees, whose salary is less than '1400', from the table 'EMP'. Then the query will be:

SQL> SELECT ENAME FROM EMP

WHERE SAL = 1400'

Result:

```
ENAME
--------------
Nidhi
Sachin
Rahul
Nanmeet
Naureet
```

Example of Greater than (>) Operator:

Display the name of the employees, whose salary is greater than '1400', from the table 'EMP'. Then the query will be:

SQL> SELECT ENAME FROM EMP

WHERE SAI>1400;

Result:

```
ENAME
-------------
Nidhi
Sachin
Rahul
Manmeet
Navreet
Amit
6 rows selected.
```

Example of Less than or equal to (<=) Operator:

Display the name of the employees, whose salary is less than or equal to '1400', from the table 'EMP'. Then the query will be:

```
SQL> SELECT ENAME FROM EMP
     WHERE SAL< =1400;
```

Result:

```
ENAME
---------------
Nidhi
Sachin
Rahul
Manmeet
Navreet
Amit
6 rows selected.
```

Example of Greater than (>=) Operator:

Display the name of the employees, whose salary is greater than or equal to '1400', the table 'EMP'. Then the query will be:

```
SQL> SELECT ENAME FROM EMP
     WHERE SAL< =1400;
```

Result:

```
ENAME
```

```
---------------
Aastha
Rohit
Aditya
Siddharth
Kunal
Akhil
Prathiba
Saira
Amit
9 rows selected.
```

Examples of LIKE Operator:

1. Display the employees whose name start with 'S' from the table 'EMP'. Then the query will be:

```
SQL> SELECT ENAME FROM EMP
     WHERE ENAME LIKE 'S%';
```

Result:

```
ENAME
---------------
Sachin
Siddharth
Saira
```

2. Display the employees, whose name ends with 'S', from the table 'EMP'. Then the query will be:

```
SQL> SELECT ENAME FROM EMP
     WHERE ENAME LIKE '%S';
```

Result:

NO ROW SELECTED.

- Display the employees, where 'S' is in the middle of the name, from the Table 'EMP'. Then the query will be:

```
SQL> SELECT ENAME FROM EMP
```

WHERE ENAME LIKE '%S%';

Result:

```
ENAME
--------
Aastha
```

Example of IN Operator:

Display the names of the employees, who are analyst and clerk, from the table 'EMP'. Then the query will be:

SQL>SELECT ENAME FROM EMP

WHERE JOB IN ('Analyst', 'Clerk');

Result:

```
ENAME
Nidhi
Kunal
Manmeet
Navreet
Saira
Amit
6 rows selected.
```

Example of BETWEEN Operator:

Display the name and salary of all employees, whose salary is between 2000 and 3000, from the table 'EMP'. Then the query will be:

SQL>SELECT ENAME, SAL FROM EMP

WHERE SAL BETWEEN 2000 AND 3000;

Result:

```
ENAME                SAL
-----------------------------
Rohit                2875
Aditya               2750
Siddharth            2550
Kunal                3000
```

```
Saira                    3000
```

7.7.3 Logical Operator

- Logical operators compare two or more than two conditions at a time to determine whether a row can be selected for the output.
- When retrieving data using a Select Statement, we use logical operators in the Where Clause which allows us to combine more than one condition.

Sr. No.	Logical Operator	Description
1	AND	For the row to be selected all the specified conditions must be true.
2	OR	For the row to be selected at least one of the specified conditions must be true.
3	NOT	For the row to be selected, the specified conditions must be false.

- NOT is totally opposite of AND and OR operator. When we want to find those rows that do not satisfy a condition, then we use the NOT operator.

1. Examples of AND Operator:

- To find the names of the clerks from the table "EMP" who are working in the department number 20, then the query will be:

```
SQL> SELECT ENAME FROM EMP
     WHERE NOB = 'CLERK' AND DEPTNO = 20;
```

Result:

```
ENAHE
---------------
Nidhi
Manmeet
```

- To find the Ename, Sal, Job from the table "EMP" where salary is greater than 1500 and deptno is 30, then the query will be:

```
SQL> SELECT ENAME, SAL, JOB FROM EMP
     WHERE SAL>1500 AND DEPTNO = 30;
```

Result:

```
ENAME                          SAL JOB
---------------------------------------
Rohit                          2175/Manager
Aditya                         2758 Manager
Prathiba                       1600 Salesman
```

- To find all the information of the employee's from the table "EMP" whose job is manager and deptno is 10, then the query will be:

SQL> SELECT * FROM EMP

WHERE JOB = 'Manager' AND DEPTNO = 10;

Result:

ENAME	DEPTON	JOB	EMPNO	SAL	HIREDATE	MGR	CITY
Siddharth	10	Manager	6671	2550	29-9-80	6728	Chd

2. Examples of OR Operator:

- To find the names of the employees from the table "EMP", who are analysts and clerk, then the query will be:

SQL> SELECT ENAME FROM EMP

WHERE JOB = -'Analyst' OR JOB = 'CIerk';

Result:

```
ENAME
--------------
Nidhi
Kunal
Navreet
Saira
Amit
6 rows selected.
```

- Display the Ename, Empno from the table "EMP", whose job is clerk or deptno is 10, then the query will be:

SQL> SELECT ENAME, EMPNO FROM EMP

WHERE JOB = 'Clerk' .OR DEPTNO = 10;

Result:

ENAME	EMPNO
Nidhi	6258
Siddharth	6671
Akhil	6728
Manmeet	6765
Navreet	6888
Amit	6823

6 rows selected.

3. NOT

- Display the names of the employees from the table "EMP", who are not clerks, then the query will be:

SQL> SELECT ENAME FROM EMP

WHERE JOB <> 'Clerk';

OR

SQL> SELECT ENAME FROM EMP

WHERE JOB! = 'Clerk';

Result:

ENAME
Aastha
Sachin
Rohit
Rahul
Aditya
Siddharth
Kunal

```
Akhil
Prathiba
Saira
10 rows selected.
```

- Display the name and deptno of employees from the table "EMP", who are not belonging to deptno 10 or 20, then the query will be:

SQL> SELECT ENAME, DEPTNO FROM EMP

WHERE NOT (DEPTNO = 10 OR DEPTNO = 20);

Result:

```
ENAME                          DEPTNO
------------------------------ ----------
Aastha                         30
Sachin                         30
Rohit                          30
Rahul                          30
Aditya                         30
Prathiba                       30
Navreet                        30
7 rows selected.
```

7.7.4 Set Operator

- Set operators are used to combine the results from two or more Select statements.
- The result of each Select Statement can be treated as a SET. Set operators are applied on these SETS to achieve the final result.
- Set operators follow some rules which are as follows:
- Number of columns should be in exact same order in all the queries.
- Number of columns should be same in all the queries.
- Data types of retrieved columns (selected statements) should be matched.

UNION ALL

SELECT Column List FROM Table2;

Example of Union All Operator:

Display all the jobs in department 10 and 20 from the table 'EMP'. Then the query will

be:

```
SQL> SELECT JOB FROM EMP
     WHERE DEPTNO = 10
     UNION ALL
     SELECT JOB FROM EMP
     WHERE DEPTNO = 20;
```

Result:

```
JOB
---------------
Manager
President
Clerk
Clerk
Analyst
Clerk
Analyst
7 rows selected.
```

NOTE: ***Union operator provides results with automatically removal of duplicate values whereas Union All operator provides results without removal of any duplicate value.***

3. Intersect

Intersect operator combine the two table expressions into one and return a result set which consists of rows that appear in the results of both table expressions. It also removes all the duplicate rows from the result set.

Syntax of Intersect Operator:-

```
SQL> SELECT Column List FROM Table 1
     INTERSECT
     SELECT Column List FROM Table2;
```

Example of Intersect Operator:

Display all the jobs common in department 10 and 20 from the table 'EMP'. Then the query will be:

```
SQL> SELECT JOB FROM EMP
```

WHERE DEPTNO = 10

INTERSECT

SELECT JOB FROM EMP

WHERE DEPTNO = 20;

Syntax: ->

SQL><SELECT STATEMENT><SET OPERATOR>< SELECT STATEMENT >

<ORDER BY Clause>;

Sr. No.	Set Operator	Description
1	Union	Returns all distinct rows selected by either query, excluding all duplicate rows.
2	Union All	Returns all rows selected by either query, including all duplicate rows.
3	Intersect	Returns all distinct rows selected by both queries.
4	Minus	Returns all distinct rows selected by the first query but not the second.

1. Union

It combines the results of two queries (same number of columns and compatible data types) into a single table of all matching rows. Union automatically removes all the duplicate values.

Syntax of Union Operator:

SQL> SELECT Column List FROM Table1

UNION

SELECT Column List FROM Table2;

Example of Union Operator:

- Display the different jobs in department 10 and 20 from the table 'EMP'. Then the query will be:

SQL> SELECT JOB FROM EMP

WHERE DEPTNO = 10

UNION

SELECT JOB FROM EMP

WHERE DEPTNO = 20;

Result:

JOB

```
------------
Analyst
Clerk
Manager
President
```

2. Union All

It combines the results of two queries (same number of columns and compatible data types) into a single table of all matching rows. It includes (shows) all the duplicate values.

Syntax of Union All Operator:

SQL> SELECT Column List FROM Table1

Result:

```
JOB
-----------
Clerk
```

4. Minus

It compares each record in statement1 with a record in statement2. It returns the results with the records in statement1 that are not in statement2.

Rows retrieved by the second query are subtracted from the rows retrieved by the first query. Only those records are considered as a result which are present only in statement1 and not in statement2.

Syntax of Minus Operator:-

SQL> SELECT Column List FROM Table1
MINUS
SELECT Column List FROM Table2;

Example of Minus Operator:

Display all the unique jobs in the department 10 from the table 'EMP'. Then the query will be:

SQL> SELECT JOB FROM EMP
WHERE DEPTNO = 10
MINUS
SELECT JOB FROM EMP

WHERE DEPTNO = 20

MINUS

SELECT JOB FROM EMP

WHERE DEPTNO = 30;

Result:

```
JOB
---------------
President
```

7.7.5 Concatenation Operator

- Concatenation operator is used to combine the two or more data strings.
- The operands of the concatenation must be compatible strings.
- Character string cannot be concatenated with a binary string.
- Concat and vertical bars (..) both represent the concatenation operator.

Concatenation Operator	Description
Piping Operator (...)	It is used to combine two or more strings

Examples of Concatenation Operator:

- List the employee salary whose empno is 6728. Then the query will he:

  SQL> Select 'My Salary is =' .....Sal as Salary

  From EMP Where Empno = 6728.

Result:**My Salary is 5000.**

- List the employee name whose empno is 6728. Then the query will be:

  SQL> Select 'My Name is =' ....Ename as Name

  From EMP Where Empno = 6728.

Result:**My Name is Akhil.**

7.8 SQL FUNCTIONS

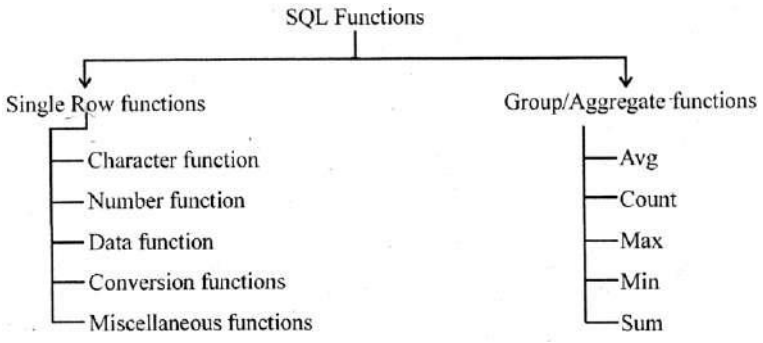


Fig. 10.9: SQL Functions

7.8.1 Single Row Functions

Single row functions operate on single rows only and returns one result per row. ***The types of single row functions are as follows:***

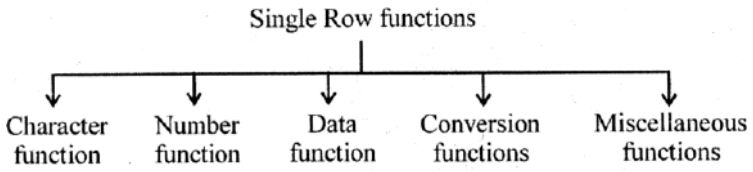


Fig. 10.10: Single Row Functions

1. Character Functions

- It is also known as text functions.
- It is used to manipulate text strings.
- It accepts character input only and returns either character or numeric values.

The following are the types of character functions:

(a) LOWER **(string):** It converts uppercase or mixed case character strings into lowercase character strings.

Example: SQL>SELECT LOWER (JOB) FROM EMP;

Result:

```
LOWER(JOB)
---------------
clerk
salesman
salesman
manager
salesman
manager
manager
analyst
president
salesman
clerk

LOWER(JOB)
---------------
clerk
analyst
clerk
14 rows selected.
```

(b) **UPPER (string):** It converts lowercase or mixed case character strings into uppercase character strings.

Example: SQL>SELECT UPPER (JOB) FROM EMP;

Result:

```
UPPER(JOB)
```

```
--------------
CLERK
SALESMAN
SALESMAN
MANAGER
SALESMAN
MANAGER
MANAGER
ANALVST
PRESIDENT
SALESMAN
CLERK

UPPER(JOB)
-------------
CLERK
ANALVST
CLERK
14 rows selected.
```

(c) CONCAT (string1, string2): It is equivalent to the concatenation operator. It returns string1 concatenated with string2. It joins (combines) two string values together.

Example: SQL>SELECT CONCAT ('MONIKA', 'TATHAK') FROM DUAL;

Result: MONIKA PATHAK

(d) LENGTH (string): It is used to get the length of a string as a numeric value. **Example:**

SQL>SELECT LENGTH (Akhil) FROM DUAL;

Result: 5

(e) ASCII (string): It is used to return the decimal representation of the first byte of string in the database character set.

Example: SQL> ASCII (Amit) FROM DUAL;

Result: 65

2. Number Functions

- It is used to perform operations on numbers.
- It accepts numeric input, only and returns numeric values.

The following are the types of numeric functions:

(a) **ABS (n):** It returns absolute value of numeric value.

Example: SQL>SELECT ABS (-29) FROM DUAL;

Result: 29

(b) **CEIL (n):** It returns the next smallest integer greater than or equal to parameter passed to n.

Example: SQL>SELECT CEIL (29.8) FROM DUAL;

Result: 30

(c) **FLOOR (n):** It returns the largest integer value less than or equal to parameter passed to n.

Example: SQL>SELECT FLOOR (29.8) FROM DUAL;

Result: 29

(d) **MOD (m,n):** It returns the remainder of m divided by n. It returns m if n is 0.

Example: SQL>SELECT MOD (16,3) FROM DUAL;

Result: 1

(e) **SQRT (n):** It returns the square root of n. The value of n cannot be negative.

Example: SQL>SELECT SQRT (25) ,FROM DUAL;

Result: 5

3. Date Functions

- Date functions operate on values of the Date datatype.
- It takes values of Date datatype as input and return values of Date datatype as output, except the Months_Between function, which returns a number.

The following are the types of date functions:

(a) **SYSDATE:** It returns the current system date and time on our local database.

Example: SQL>SELECT SYSDATE FROM DUAL;

Result:

```
SYSDATE
---------
18-JUN-15
```

(b) LAST_DAY: It returns the date of the last day of the month specified.

Example: SQL>SELECT SYSDATE LAST DAY (SYSDATE) FROM DUAL;

(c) CURRENT_DATE: It returns the current date in the Gregorian calendar for the session's time zone.

Example: SQL>SELECT SYSDATE CURRENT DAY (SYSDATE) FROM DUAL;

(d) NEXT_DAY: It returns the date of next specified day of the week after the 'date'.

Example: SQL>SELECT SYSDATE NEXT DAY (SYSDATE) FROM DUAL;

(e) ADD_MONTHS: It adds or subtracts the months to or from a date.

Example: SQL>SELECT SYSDATE, ADD_MONTHS (SYSDATE, 4) FROM DUAL;

Result:

```
SYSDATE         ADD_MONTH
----------------------
18-JUN-15       18-OCT-15
```

4. Conversion Functions

It converts the value from one form to another form.

The following are the types of conversion functions:

(a) Implicit Data Type Conversion: It occurs when the expression evaluator automatically converts the data from one data type to another.

(b) Explicit Data Type Conversion: It occurs when we explicitly converts the data from one data type to another.

5. Miscellaneous Functions

The following are the types of miscellaneous functions:

(a) GREATEST: It returns the greatest value in the list of expressions.

Example: SQL>SELECT GREATEST (2, 11, 25, 29) FROM DUAL;

Result: 29

(b) LEAST: It returns the smallest value in the list of expressions.

Example: SQL>SELECT LEAST (2, 11, 25, 29) FROM DUAL;

Result: 2

(c) USER: It returns the username of the current user logged on.

Example: SQL>SELECT USER FROM DUAL;

Result: SCOTT

7.8.2 Group/Aggregate Functions

- Aggregate functions are also known as Group functions or Summary functions.
- SQL supports the functions which can be used to select and compute the numeric, date columns and characters of the relation.
- These functions operate on multiple rows (group of rows) and return only one value for a group or table, therefore these functions are known as aggregate functions. By default, all rows are treated as one group in a table.

The types of aggregate functions are as follows:

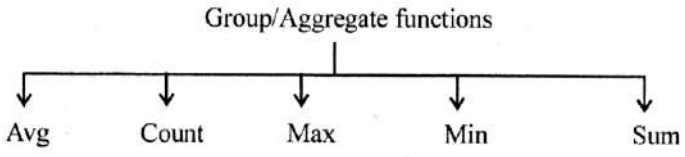


Fig. 10.11: Group/Aggregate Functions

STUD

Name	Class	Roll Number	Marks	Age
Akhil	C12	11	95	16
Monika	C12-	12	91	15
Aastha	M12	13	95	14
Rohit	E12	14	94	12
Rahul	E12	15	93	13
Ankush	C12	16	95	15
Radhika	M12	17	92	14

1. Avg: The Avg (average) function returns the arithmetic mean of the value of a column in a given relation. This function is applicable on numeric values.

Examples of Avg Function:->

- To find the average marks of the students from the table STUD, then the query will be:

  SQL> SELECT AVG (Marks) FROM STUD;

Result: 93.51

- To find the average salary of the employees from the table EMR Then the query will be:

  SQL> Select AVG (SAL) AS Average Salary FROM EMP;

Result: Average Salary

2091.07143

2. Count: The Count function returns the number of rows in a relation (table). This function is used for numeric, character values and date. The Count function returns value only if it satisfies the condition stated in the Where Clause.

Examples of Count Function:

- To find the number of students from the table 'STUD'. Then the query will be:

  SQL> Select COUNT (*) FROM STUD;

Result: 7

- To find the total number of employees from the table EMP, Then the query will be:

  SQL> SELECT %COUNT (*) AS TOTAL EMPLOYEE FROM EMP;

Result: TOTAL EMPLOYEE

14

3. Max: The Max function returns the maximum of the values of a column from the given relation.

Examples of Max Function:

- To find the maximum marks from the table 'STUD'. Then the query will be:

  SQL> MAX (Marks) FROM STUD;

Result: 95

- To find the maximum salary drawn by the employee from the table EMP. Then the query will be:

  SQL> MAX (SAL) AS Maximum Salary FROM EMP;

Result: **Maximum Salary**

5000

4. Min: The Min function returns the minimum of the values of a column from the given relation.

Examples of Min Function:

- To find the minimum marks from the table STUD. Then the query will be:

SQL> MIN (Marks) FROM STUD;

Result: 91

- To find the minimum salary drawn by the employee from the table EMP. Then the query will be:

  SQL> MIN (SAL) AS Minimum Salary FROM EMP;

Result: **Minimum Salary**

900

5. **Sum:** The Sum function returns the sum of values (numeric type) of a column.

Example of Sum Function:

- To find the sum of marks from the table STUD. Then the query will be:

  SQL> SELECT SUM (Marks) FROM STUD;

Result: **655**

- To find the total salary given to the employees from the table BMP. Then the query will be:

  SQL> SELECT SUM (SAL) AS Total Salary FROM EMP;

Result: Total Salary

29275

7.9 JOINS

- Mostly we retrieve data from one table at a time. But what will we do if we need to retrieve data from multiple tables.
- Oracle provides the facility to retrieve the data from multiple tables with the help of joins.
- Joins are used to combine columns from different tables.
- Joins allow us to retrieve the data from multiple users in a single query.
- Joins permits us to select data from more than one table in one SQL statement (query).
- A join is used to combine rows from multiple tables.
- Joins are used to relate information in different tables.
- The connection between tables is established through the Where Clause.
- Where Clause is known as join condition.
- The rows retrieved after joining the two tables based on a condition in which one table act as a primary key and other act as a foreign key. Columns in both tables should be matched.

Syntax of Join:

SQL> SELECT tablel.column, table2.column,.....tableN.column

FROM table1, table2,.....tableN.

WHERE tablel.column1 = table2. column2;

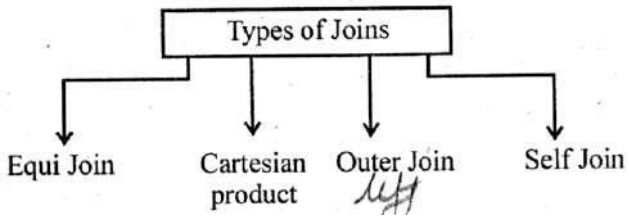


Fig. 10.12: Types of Join

7.9.2 Equi Join

- It is also known as Inner Join.
- When two tables are joined together using equality of values in one or more columns, they make an equi join.
- Equi join is used when we need to compare each record in two joined tables and comes with matching record.
- Table prefixes are utilized to prevent ambiguity.
- We use equi join (inner join) when we only want to return records where there is at least one row in both tables that match the join condition.
- Equi join uses the equal sign as the comparison operator.

Example of Equi Join:

First Table is BMP

Second Table is DEPT.

EMP

ENAME	DEPTNO	JOB	EMPNO	SAL	HIREDATE	MGR	CITY	COMM
Nidhi	20	Clerk	6258	900	9-5-83	6801	Chd	
Aastha	30	Salesman	6388	1500	1-12-89	6587	Delhi	300
Sachin	30	Salesman	6410	1350	25-1-92	6587	Pta	500
Rohit	20	Manager	6455	2875	27-12-91	6728	Nba	
Rahul	30	Salesman	6543	1350	28-5-87	6587	Nba	1400
Aditya	30	Manager	6587	2750	17-8-86	672S	Pta	

Siddhart h	10	Manager	6671	2550	29-9-80	6728	Chd	
Kunal	20	Analyst	6677	3000	8-12-82	6455	Delhi	
Akhil	10	President	6728	5000	2-11-85		Delhi	
Prathiba	30	Salesman	6733	1600	4-6-85	6587	Pta	0
Manmeet	20	Clerk	6765	1050	11-1-84	6677	Ldh	
Navreet	30	Clerk	6800	950	25-3-84	6587	Pta	
Saira	20	Analyst	6801	3000	15-4-80	6455	Chd	
Amit	10	Clerk	6823	1400	25-8-85	6671	Ldh	

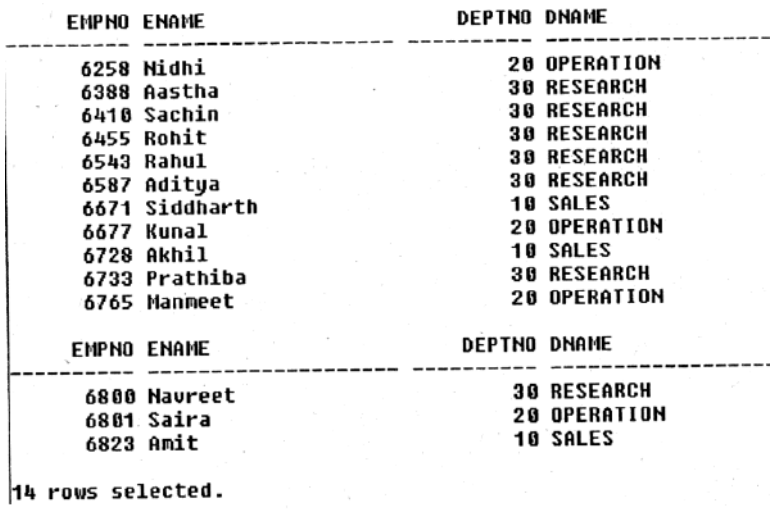
DEPT

DEPTNO	DNAME	LOG
10	Sales	London
20	Operation	Mumbai
30	Research	Paris
40	Accounting	New York

Then the query will be:

SQL> SELECT EMPNO, ENAME, EMP.DEPTNO, DNAME FROM EMP, DEPT WHERE EMP.DEPTNO = DEPT.DEPTNO;

Result:



```
     EMPNO ENAME                DEPTNO DNAME
---------- -------------------- ---------- --------------
      6258 Nidhi                        20 OPERATION
      6388 Aastha                       30 RESEARCH
      6410 Sachin                       30 RESEARCH
      6455 Rohit                        30 RESEARCH
      6543 Rahul                        30 RESEARCH
      6587 Aditya                       30 RESEARCH
      6671 Siddharth                    10 SALES
      6677 Kunal                        20 OPERATION
      6728 Akhil                        10 SALES
      6733 Prathiba                     30 RESEARCH
      6765 Manmeet                      20 OPERATION

     EMPNO ENAME                DEPTNO DNAME
---------- -------------------- ---------- --------------
      6800 Navreet                      30 RESEARCH
      6801 Saira                        20 OPERATION
      6823 Amit                         10 SALES

14 rows selected.
```

Explanation of Equi Join:

For Equi Join, both the table names should be mentioned.

Column name should be specified with the table name to avoid confusion.

Deptno of BMP table is joined with the deptno of DEPT table because Deptno exists in both the tables.

7.9.2 Cross Join

- It is also known as cartesian product or cartesian join.
- It returns the number of rows equal to the product of all rows in all rows in all the tables being joined.
- It provides results in mXn rows.
- It is used when we want to join every row of a table to every row of itself.

Example of Cross Join:

SQL>SELECT EMPNO, ENAME, DNAME, LOC FROM EMP, DEPT;

Result:

EMPNO	ENAME	DNAME	LOC
6258	Nidhi	SALES	LONDON
6388	Aastha	SALES	LONDON
6410	Sachin	SALES	LONDON
6455	Rohit	SALES	LONDON
6543	Rahul	SALES	LONDON
6587	Aditya	SALES	LONDON
6671	Siddharth	SALES	LONDON
6677	Kunal	SALES	LONDON
6728	Akhil	SALES	LONDON
6733	Prathiba	SALES	LONDON
6765	Manmeet	SALES	LONDON

EMPNO	ENAME	DNAME	LOC
6800	Navreet	SALES	LONDON
6801	Saira	SALES	LONDON
6823	Amit	SALES	LONDON
6258	Nidhi	OPERATION	MUMBAI
6388	Aastha	OPERATION	MUMBAI
6410	Sachin	OPERATION	MUMBAI
6455	Rohit	OPERATION	MUMBAI
6543	Rahul	OPERATION	MUMBAI
6587	Aditya	OPERATION	MUMBAI
6671	Siddharth	OPERATION	MUMBAI
6677	Kunal	OPERATION	MUMBAI

EMPNO	ENAME	DNAME	LOC
6728	Akhil	OPERATION	MUMBAI
6733	Prathiba	OPERATION	MUMBAI
6765	Manmeet	OPERATION	MUMBAI
6800	Navreet	OPERATION	MUMBAI
6801	Saira	OPERATION	MUMBAI
6823	Amit	OPERATION	MUMBAI
6258	Nidhi	RESEARCH	PARIS
6388	Aastha	RESEARCH	PARIS
6410	Sachin	RESEARCH	PARIS
6455	Rohit	RESEARCH	PARIS
6543	Rahul	RESEARCH	PARIS

EMPNO	ENAME	DNAME	LOC
6587	Aditya	RESEARCH	PARIS
6671	Siddharth	RESEARCH	PARIS
6677	Kunal	RESEARCH	PARIS
6728	Akhil	RESEARCH	PARIS
6733	Prathiba	RESEARCH	PARIS
6765	Manmeet	RESEARCH	PARIS
6800	Navreet	RESEARCH	PARIS
6801	Saira	RESEARCH	PARIS
6823	Amit	RESEARCH	PARIS
6258	Nidhi	ACCOUNTING	NEW YORK
6388	Aastha	ACCOUNTING	NEW YORK

EMPNO	ENAME	DNAME	LOC
6410	Sachin	ACCOUNTING	NEW YORK
6455	Rohit	ACCOUNTING	NEW YORK
6543	Rahul	ACCOUNTING	NEW YORK
6587	Aditya	ACCOUNTING	NEW YORK
6671	Siddharth	ACCOUNTING	NEW YORK
6677	Kunal	ACCOUNTING	NEW YORK
6728	Akhil	ACCOUNTING	NEW YORK
6733	Prathiba	ACCOUNTING	NEW YORK
6765	Manmeet	ACCOUNTING	NEW YORK
6800	Navreet	ACCOUNTING	NEW YORK
6801	Saira	ACCOUNTING	NEW YORK

EMPNO	ENAME	DNAME	LOC
6823	Amit	ACCOUNTING	NEW YORK

56 rows selected.

Explanation:

Table BMP has 14 rows.

Table DEPT has 4 rows.

Then, total number of rows = mXn

=14X4

=> Total number of rows =56 rows

7.9.3 Outer Join

- Outer join has symbol (+).
- It is used if there is any value in one table that do not have corresponding value in other table. Such rows are forcefully selected by it.
- It is used on one side of the join condition only and the corresponding columns for that row will have NULL value.

Example of Outer Join:

SQL>SELECT EMPNO, ENAME, E-MP.DEPTNO, DNAME, LOC FROM EMP,DEPT

WHERE EMP.DEPTNQ (+) = DEPT.DEPTNO;

Result:

EMPNO	ENAME	DEPTNO	DNAME	LOC
6258	Nidhi	20	Operation	Mumbai
6388	Aastha	30	Research	Paris
6410	Sachin	30	Research	Paris
6455	Rohit	30	Research	Paris
6543	Rahul	30	Research	Paris
6587	Aditya	30	Research	Paris
6671	Siddharth	10	Sale	Paris
6677	Kunal	20	Research	Paris
6728	Akhil	10	Sale	London
6733	Prathiba	30	Research	Mumbai
6765	Manmeet	20	Operation	London
6800	Navreet	30	Research	Paris
6801	Saira	20	Operation	Mumbai
6823	Amit	10	Sale	London

7 .9.4 Self Join

- Self join is used when a table is joined/compared to itself.
- A table is joined to itself means each row of the table is combined with itself and with every row of the table.
- If we want to use self join, then we need to open the two copies of same table by using table aliases
- Table name aliases are defined in the From Clause of the query.
- Table alias is used to avoid confusion among two same tables.

Example of Self Join:

SQL>SELECT WORKER.ENAME AS ENAME, MANAGER.ENAME AS MANAGER

FROM EMP WORKER, EMP MANGER

WHERE WORKER.MGR = MANAGER.EMPNO;

7.10 ROLL UP OPERATION

- It is just an extension of the Group of Clause.
- It appears only with Group by Clause.
- It is useful in generating reports (result set) that contain subtotals and totals.
- It is used by the report writers to extract statistics and summary information from result sets.
- It calculates a grand total. First, it calculates the standard aggregate values specified in the Group by Clause. Then, it creates progressively higher level subtotals, moving from right to left through the list of grouping columns. Finally, it creates a grand total.
- It is much like the Group by Clause except it gives subtotal and grand totals at the end.

Syntax of Roll up Operation:

SQL> SELECT.....

FROM.....

WHERE.....

GROUP BY ROLLUP [column1, column2,.....];

Example of Roll up Operation:

SQL>SELECT DEFTNO, JOB, COUNT (*), SUM (SAL) FROM EMP GROUP BY ROLLUP (DEPTNO/JOB);

Result:

DEPTNO	JOB	COUNT(*)	SUM(SAL)
10	Clerk	1	1400
10	Manager	1	2550
10	President	1	5000
10		3	8950
20	Analyst	2	6000
20	Clerk	2	1950
20		4	7950
30	Clerk	1	950
30	Manager	2	5625
30	Salesman	4	5800
30		7	12375
DEPTNO	JOB	COUNT(*)	SUM(SAL)
		14	29275

12 rows selected.

7.11 CUBE OPERATION

- It is a simple extension of Group by Clause with select statement.
- It takes aggregation one step further than Roll Up.
- The cube operation generate result set (subtotals) contains a cross tabulation of all the possible combinations of the grouping columns.
- Therefore, the result of a cube operation will contain all subtotals produced by an equivalent Roll Up operation and some additional subtotals.
- It applied to all aggregate functions. It produces subtotals and a grand total.

Syntax of Cube Operation:

SQL> SELECT.....

FROM.....

WHERE.....

GROUP BY CUBE [column1, column2, ...];

Example of Cube Operation:

SQL>SELECT DEPTNO, JOB, COUNT (*), SUM (SAL) FROM EMP GROUP BY CUBE (DEPTNO, JOB);

Result:

DEPTNO	JOB	COUNT(*)	SUM(SAL)
		14	29275
	Analyst	2	6000
	Clerk	4	4300
	Manager	3	8175
	President	1	5000
	Salesman	4	5800
10		3	8950
10	Clerk	1	1400
10	Manager	1	2550
10	President	1	5000
20		4	7950

DEPTNO	JOB	COUNT(*)	SUM(SAL)
20	Analyst	2	6000
20	Clerk	2	1950
30		7	12375
30	Clerk	1	950
30	Manager	2	5625
30	Salesman	4	5800

17 rows selected.

Difference between Roll Up Operation and Cube Operation

Sr. No.	Rollup Operation	Cube Operation

1	Roll up operation produces only a fraction of possible subtotal combinations.	Cube operation produces subtotals of possible combinations and a grand total.
2	Roll up operation provides us only the sets in the order listed which shows aggregates for a hierarchy of values in the selected columns	Cube operation provides us the subtotals of possible combinations of columns which show aggregates for all combinations of values in the selected columns.

7.12 NESTED QUERY

- Nested query means query within another query.
- First, we evaluate the inner query (sub query) within the Where Clause.
- Then, the result of inner query (sub query) is constituted in the condition of outer query.
- The result of inner query will pass to the outer query for the preparation of final result.
- In a nested query, there can be any number of sub queries.
- Sub query is used with Where or Having Clause. Sub query cannot be used with the Order by Clause.
- A query is called a sub query and complete select statement is called a nested query.

Syntax of Nested Query:

SQL> SELECT <column,.....>
FROM<table>
WHERE expression operator
(SELECT <column,. .>
FROM <Table>
WHERE <condition>);

Examples of Nested Query:

List the employee names belonging to the department of Akhil from table "EMP", then the query will be:

SQL> SELECT ENAME FROM EMP
WHERE DEPTNO = (SELECT DEPTNO FROM EMP :
WHERE ENAME = 'Akhil');

Result: **ENAME**

```
----------
Siddharth
```

```
Akhil
Amit
```

Explanation:

We divide this nested query into two parts because we do not know the department to which Akhil belongs.

In the first part: SELECT DEPTNO FROM EMP

WHERE ENAME ='Akhil';

In the inner query, we have to find out the department of Akhil.

In the second part: SELECT ENAME FROM EMP

WHERE DEPTNO = 10;

In the outer query, we use the department number (result of inner query) to find out the other employee names of that department.

7.13 SUBQUERY

Subquery is a inner query within another main query (Outer query).

Special operators in subquery: There are some special operators used in subqueries to perform specific tasks which are as follows:

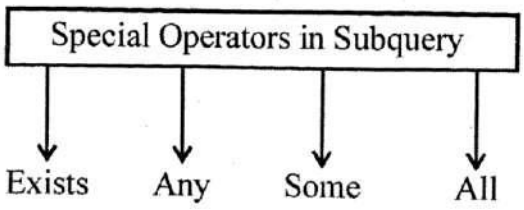


Fig. 10.13: Special Operators in Subquery

(a) Exists

It is used to check the existence of values.

It provides results in the form of true/false.

If the subquery provides any output, then the result will be true.

If the subquery does not provide any output, then the result will be false.

Example of Exists:

Display the employee details if and only if more than 8 employees working in the department number 20 from the table 'BMP'. Then the query will be:

SQL>SELECT * FROM EMP

WHERE DEPTNO = 20 AND EXISTS (SELECT COUNT (*) FROM EMP)

WHERE DEPTNO = 20

GROUP BY DEPTNO

HAVING COUNT (*) >20;

(b) Any

It is used to compares the lowest value from the set.

Any operator returns value TRUE if the comparison value matches any of the values in the list.

Examples of Any:

Display the employees whose salary is greater than any 'Manager' and he is not 'Manager'. Then the query will be:

SQL>SELECT ENAME, JOB, SAL FROM EMP

WHERE SAL>ANY (SELECT SAL FROM EMP WHERE JOB = 'Manager')

AND JOB<> 'Manager';

Result: NO ROW SELECTED

(c) Some

The Some operator and Any operator are equivalent.

Some operator works same like Any operator i.e. it compares the lowest value from the set.

Example of Some:

Display the employee names along with their salaries from the table 'EMP', whose salary is greater than the lowest salary of an employee belonging to department number '10'. Then the query will be:

SQL>SELECT ENAME, SAL FROM EMP

WHERE SAL>SOME (SELECT SAL FROM EMP WHERE DEPTNO = 10);

Result:

ENAME	SAL
Akhil	5000
Saira	3000
Kunal	3000
Rohit	2875
Aditya	2750
Siddharth	2550
Prathiba	1600
Aastha	1500

8 rows selected.

(d) All

It is used to compare the highest value from the set.

All operator returns value true If the comparison value matches with all the. values in the list.

Example:

Display the employees with salary less than those whose job is 'Manager'. Then the query will be:

SQL>SELECT ENAME, JOB, SAL FROM EMP

WHERE SAL < ALL (SELECT SAL FROM EMP WHERE JOB = 'Manager')

AND JOB<> 'Manager';

Result:

ENAME	JOB	SAL
Nidhi	Clerk	900
Aastha	Salesman	1500
Sachin	Salesman	1350
Rohit	Manager	2875
Rahul	Salesman	1350
Aditya	Manager	2750
Siddharth	Manager	2550
Kunal	Analyst	3000
Akhil	President	5000
Prathiba	Salesman	1600
Manmeet	Clerk	1050

ENAME	JOB	SAL
Navreet	Clerk	950
Saira	Analyst	3000
Amit	Clerk	1400

14 rows selected.

7.14 VIEW

- For security reasons, it is desired that a user sees only a part of a table or group of tables.
- On the other hand, a user may like to view some of the information from many tables.
- Hence, each user views the database from his own angle without seeing the entire database.
- With the help of view, we can see a selective portion of data from one or more tables.
- A view is a virtual table. Views do not contain data their own and always tasks the data from base table.
- A view is a specific representation of data from one or more tables.
- It is stored as a select statement in the database.
- The tables on which a view is based are known as base tables.
- We can query, insert into, delete from and update from views in almost the same way as tables.
- If we make changes in the tables, then changes automatically reflects in the views.

For example: We wish that a 'Manager' should have access only to employee's name, job and department in the table 'BMP'. It means 'Manager' should not have any access to the salaries or other details of the employees. It is possible only with the help of view.

Syntax of View:

```
SQL> CREATE VIEW view_name
As
SELECT column_list
FROM table_name [WHERE Condition];
```

7.14.1 Creating a View

- We must have .the create view privilege to create a view in our schema.
- We must have to create any view system privilege to create a view in another user's schema.
- The owner of the view must have been explicitly granted privileges to access all objects referenced in the view definition.
- View's functionality depends on the privileges of the view owner. It means if the owner of the view has only the select privilege, then he/she can only select row, cannot insert, update or delete rows.

Example of Creating View:

Create a view for the 'Akhil' from the table 'EMP'. He can access only employee's name, job and department. He cannot access the salary of all the employees.

SQL>CREATE VIEW Akhil

AS SELECT ENAME, JOB AND DEPTNO FROM EMP;

7.14.2 Modifying a View

- We use the 'OR REPLACE' option to modify the view.
- If view already exists, then it will replace with new definition or a new view will be created.
- The view will become invalid whenever the base table.is altered.

Example of Modifying a View:

SQL>CREATE OR REPLACE VIEW Akhil Garg

AS SELECT * FROM EMP;

View Created.

7.14.3 Inline View

- Inline view is a subquery with an alias (correlation name) that we can use like a view inside a SQL statement.
- It appears in the 'From Clause', of the select statement and enclosed in parenthesis.

Example of Inline View:

We want to-select first 3 employee hired by the company.

SQL>SELECT ENAME<HIREDATE FROM

(SELECT ENAME<HIREDATE FROM EMP ORDER BY HIREDATE)

WHERE ROWNUM<=3;

7.14.4 Materialized View

- Materialized view is a standard object which is used to summarize, replicate, precompute and distribute data.
- It is a database object that contains the results of a query.
- It provides indirect access to table data by storing the results of a query in a separate schema object.
- A materialized view can query tables, views and other materialized views.
- Materialized views are local copies, of data located remotely or are used to create summary tables based on aggregations of a table's data.

- Materialized views, which store data based on remote tables, are also known as snapshots.
- Materialized views can be used to replicate the data at distributed sites.
- Materialized views are suitable in various computing environments such as data warehousing, decision support, distributed computing or mobile, computing.
- It is used in data warehouses so as to increase the speed of queries on a large database.
- Materialized view improves query performance by precalculating expensive join and aggregation operations.
- It is used in mobile computing to download a subset of data from central servers to mobile clients.

7.14.5 Dropping a View

- We can drop any view contained in the schema.
- The 'Drop View' statement is used to drop a view from a database.
- The 'Drop Any View' privilege is used to drop a view in another's schema.

Syntax of Dropping View:

SQL>DROP VIEW view_name;

7.14.6 Advantages of View

- The database becomes secured because user is allowed a limited view of database.
- View is used to restrict access to the database or to hide data complexity.
- The user queries become simplified.
- The user gets consisted view of database.
- We can rename the table columns by giving the different names to columns while creating views.
- Views take very little space to store because database contains only definition of view not all the present data.
- It solves the redundancy (duplication) problem.
- As the data is taken from base table, accurate and up to date information is provided by the view.
- Different views can be created on the same base table for different users.

7.15 DISADVANTAGES OF SQL

- SQL statements are passed to Oracle engine/server one at a time which generates the

problem of network traffic. Every time when a SQL statement is executed, a call is made to Oracle engine/server.

- SQL cannot use the PL/SQL statements.
- SQL does not provide any procedural capabilities i.e. conditional checking, branching, looping etc.
- If any error occurs in the statement, SQL cannot fails to handle it, and then Oracle engine displays its own error message.
- PL/SQL is required to overcome the limitations of SQL. We will discuss PL/SQL in chapter 11.

7. 16 KEY POINTS

7.16.1 SQLAlias

- SQL Alias is used for columns and tables.
- SQL Alias is created to make the columns and tables more readable.
- It is used for columns when column names are big or not readable.
- It is used for tables when there are more than one tables involved in a query.

(a) SQL Alias for Columns: Display the names of all the employees from the relation 'EMP' through column alias. Then the query will be:

SQL> Select ENAME AS NAME From EMP;

In the above query, ENAME is given an alias 'NAME'. In the result, column name looks as 'NAME' instead of 'ENAME'.

Result:

```
SQL> SELECT ENAME AS NAME FROM EMP;
NAME
--------------
Nidhi
Aastha
Sachin
Rohit
Rahul
Aditya
Siddharth
```

```
Kunal
Akhil
Prathiba
Manmeet

NAME
-----------
Navreet
Saira
Amit
14 rows selected.
```

(b) SQL Alias for Tables: Display the names of all the employees from the relation 'EMP' through table alias. Then the query will be:

SQL> Select E.ENAME From EMP E;

In the above query, alias _E‘ is defined in the relation _EMP'.

Result:

```
ENAME
------------
Nidhi
Aastha
Sachin
Rohit
Rahul
Aditya
Siddharth
Kunal
Akhil
Prathiba
Manmeet

ENAME
---------------
```

```
Navreet
Saira
Amit
14 rows selected.
```

7.16.2 Null Values in SQL

- Null value represents an inapplicable or unknown value.
- It is not 0 or a blank space.
- For example: Display the names of the employees who are not eligible for the commission. Then the query will be:

  SQL>SELECT ENAME FROM EMP

  WHERE COMM IS NULL;

Result:



```
ENAME
----------
Nidhi
Aastha
Rohit
Aditya
Siddharth
Kunal
Akhil
Manmeet
Navreet
Saira
Amit

11 rows selected.
```

7.16.3 Difference between Delete and Truncate Command

	Delete	Truncate
1	Delete command delete the table and send it to recycle bin.	Truncate command removes data from memory.
2	Delete command is slow as compared to truncate.	Truncate command is faster as compared to delete command.

3	Data can be recovered.	Data can't be recovered.
4	DML	DDL
5	In delete command, we may or may not give condition like where.	In truncate command, we do not give where condition.
6	Doesn't release the memory occupied by records of table.	Releases the memory occupied by records of table.

Questions

1. What is SQL? Write its characteristics in detail.
2. Define DCL and explain different SQL commands come under DCL.
3. Define DML and explain different command with suitable example,
4. What is the use of DDL language? Explain with suitable example
5. Query to display the different designation in department no. 20 and 30 of from table _EMP‘.
6. Query to find the average of particular column.
7. Query to display the employees no. and name in department no. 10 and 3 0 from table 'EMP'
8. Query to select department that has total salary paid for its employees more than. 8000.

COURSE: DBMS

UNIT 8 : PL/SQL

8.1 INTRODUCTION

8.2 ADVANTAGES OF PL/QL

8.3 DIFFERENCE BETWEEN SQL AND PL/SQL

8.4 BLOCK STRUCTURE OF PL/SQL

8.5 ARCHITECTURE OF PL/SQL

8.6 ELEMENTS OF PL/SQL

8.7 DATA TYPES OF PL/SQL

8.8 PL/SQL VARIABLES

8.9 PL/SQL CONSTANTS

8.10 CONTROL STRUCTURES OF PL/SQL

8.10.1 Conditional Statements

8.10.2 Iterative Statements

8.10.3 Sequential Statements

8.11 CURSORS IN PL/SQL

8.11.1 TYPES OF CURSOR

8.12 EXCEPTION HANDLING IN PL/SQL

8.12.1 Guidelines to Avoid and Handle the Exceptions

8.12.2 Types of Exception

8.13 EXCEPTION PROPAGATION

8.14 SUBPROGRAMS

8.14.1 Advantages of Subprograms

8.14.2 Block Structure of PL/SQL Subprograms

8.14.3 Types of Subprograms

8.14.4 Procedures

8.14.5 Functions

8.15 STORED PACKAGES

8.16 TRIGGERS

8.16.1 Guidelines for Designing Triggers

8.16.2 States of Triggers

8.16.3 Parts of Trigger

8.16.4 Types of Trigger

8.16.5 Creating and Dropping A Trigger

8.16.6 Advantages of Triggers

8.1 INTRODUCTION

1. PL/SQL stands for procedural language/structured query language.
2. It is an extension of SQL.
3. It was developed by Oracle Corporation to enhance the capabilities of SQL.
4. It is a combination of SQL and procedural features of standard programming languages.
5. The aim of PL/SQL is to removethe restrictions of SQL language.
6. It is not a case sensitive language. It means, we can use both the lower case and uppercase letters.
7. It is a powerful transaction processing language which extends the capabilities of SQL by adding control statements, procedures and functions.
8. It supports all the data manipulation operations and all data types of SQL.
9. It is used to overcome the limitations of SQL. It. is used to write triggers and stored procedures.
10. In short, PL/SQL is a database oriented programming language which extends Oracle SQL with procedural capabilities.

8.2 ADVANTAGES OF PL/QL

1. **Support for SQL:** It allows us touseall the SQL commands, as wellas all SQL functions,operators and data types sothat we can manipulate Oracle's data flexibly and safely.

2. **Better Performance:** Without PL/SQL, Oracle server processes SQL statements one ata time. Every time a SQL statement is issued, it must be sent over the network whichcreates more traffic. But with PL/SQL, an entire block can be sent to Oracle server at atimereduces network traffic and improves the performance.
3. **Error Handling:** PL/SQL handles errors during the execution of its program. When, anerror (exception) is found, it takes some specifications depending upon the type of theerror.
4. **BlockStructure:**It is a block structures language. It consists of blocks of code. Each program written as a block. Blocks can be nested. Each block performs a specific task.Blocks can be reused.
5. **Portability:** Programs written in PL/SQL can run anywhere. These programs can be used in another new environment without any change. It means programs written in PL/SQL are portable to any platform (any operating system) on which Oracle runs. PL/SQL programs can be reused in different environments.
6. **Higher Productivity:** PL/SQL increases productivity by adding functionality to nonprocedural tools such as forms and reports. We can use an entire PL/SQL block in an Oracle from trigger without using multiple trigger steps(macros).

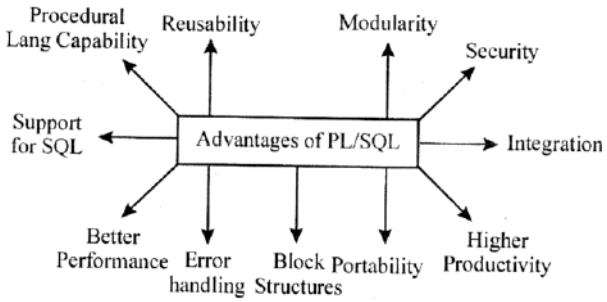


Fig. 8.1: Advantages of PL/SQL

7. **Integration:**It integrates well with SQL*PLUS and other application development products of Oracle Corporation.
8. **Security:** We can prevent client applications from modifying the sensitive data by using PL/SQL stored procedures. It enables the user to partition the application

logic from client to server and protect the data by giving proper access power to different users.

9. **Modularity:** PL/SQL divides an application/process into manageable and well defined modules such as procedures and functions. These modules are known as subprograms.
10. **Reusability:** PL/SQLprovides reusability because a single stored procedure can be used by different applications. If at any point of time procedure changes, then it will not affect the application program.
11. **Procedural Language Capability:** PL/SQLnot only supports SQL data manipulation commands but also allow control structures such as conditional statements (if else statements) and loops like (for loops). Control structures controls the procedural flow of the program and are very important PL/SQL extension to SQL.

8.3 DIFFERENCE BETWEEN SQL AND PL/SQL

Sr. No.	SQL	PL/SQL
1	SQL stands for Structured Query Language.	PL/SQL stands for Procedural Language/Structured Query Language.
2	SQL is used to manage database operations. It does not have any procedural capability.	PL/SQL is the procedural language extension to the non-procedural language SQL. It is a com6ination of SQL and procedural features of standard programming languages.
3	Only one statement is executed at a time.	The block of code is executed at a time.
4	SQL statements can be embedded within a PL/SQL program.	PL/SQL code cannot be embedded within a SQL statement.
5	It tell the database what to do and not how to do it.	It tells database how to do things.

6	There is no provision of error handling in SQL.	There is a provision of error handling in PL/SQL
7	It is used to code queries, data definition and manipulation statements.	It is used to code program blocks, functions,procedures and packages.

8.4 BLOCK STRUCTURE OF PL/SQL

1. Each program of PL/SQL consists of SQL and PL/SQL statements. It forms a PL/SQL block.
2. In PL/SQL, programs can be divided into logical blocks.
3. Comments can be used to document the code.
4. PL/SQL blocks can be nested within the other PL/SQL blocks.
5. ***PL/SQL block consists of three sections which are as follows:***
   - **(a) Declarative Section (Optional)**
   - **(b) Executable Section (Mandatory)**
   - **(c) Exception/Error Handling Section (Optional)**

6. ***The structure of PL/SQL block is as follows:***

DECLARE

<declarations>

BEGIN

<executafole statements>

EXCEPTION

<exception handlers>

END;

7. ***The description of these sections of PL/SQL block structure is as follows:***

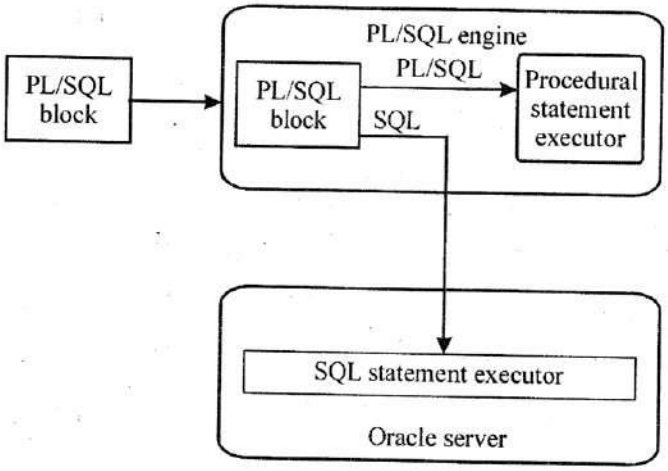
(a) Declarative Sections: This section is optional. This section starts with the reserved keyword 'DECLARE'. This section is used to declare any place holders like variables, constants, records and cursors, which are used to manipulate data in the executable form.

(b) **Executable Section:** This section is mandatory. This section starts with the reserved keyword 'BEGIN' and ends with 'END'. This is the section where the program logic is written toperform any task.

(c) **Exception Section:** This section is optional. This section starts with the reserved keyword 'EXCEPTION'.

8.5 ARCHITECTURE OF PL/SQL

1. PL/SQL is not an independent product.
2. It is a run time technology.
3. It is like an engine which installed in an Oracle Server or in application development tools such as Oracle Form Builder, Oracle Reports Builders etc.
4. PL/SQL engine executes PL/SQL blocks and subprograms.
5. PL/SQL resides in two environments:
   (a) Oracle Server
   (b) Oracle Tools



Fig, 8.2: Architecture of PL/SQL

6. These two environments (Oracle Server and Oracle Tools) are independent of each other.
7. ThePL/SQL engine executes-the procedural part of the statements and sends the SQL statements to the 'SQL Statement Executor' in the Oracle server.
8. PL/SQL code is stored in the;Oracle Server.

8.6 ELEMENTS OF PL/SQL

The elements of object are used to represent real world objects, and operations.

The PL/SQL has the following set of elements:

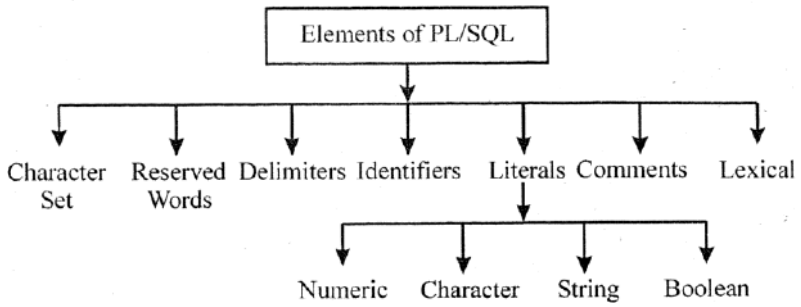


Fig. 8.3: Elements of PL/SQL

I. Character Set

1. PL/SQL programs are written as lines of text using a specific set of characters.
2. PL/SQL is not a case sensitive. In PL/SQL, programs can be written in both lower case and upper case.
3. The PL/SQL character set includes:
    - The upper case letters (A.....Z) and lower case letters (a...z).
    - Numerals (0...9)
    - Tabs, spaces and carriage returns
    - Symbols: (),+i-,*,/,<>,!,:,{}, Q,& etc.

II. Reserved Words

1. Reserved words have special, syntactic meaning to PL/SQL and cannot be redefined.
2. If we will try to redefine a reserved word, we will get a compilation error.
3. Reserved words can be written in both upper case and lower case. Generally, reservedwords are written in upper case to promote readability,
4. For example: The words 'BEGIN' and $^{C}END^{9}$ are reserved words, which bracket the executable part of a block or subprogram.

III. Delimiters

1. A *delimiter is* a simple or compound symbol that has a special meaning to PL/SQL.
2. Simple symbols consist of one character whereas compound symbols consist of two characters.
3. The list of simple symbols and compound symbols is as follows:

List of Simple Symbols

Simple Symbol	Meaning
+	Addition Operator
-	Subtraction/Negation Operator
/	Division Operator
*	Multiplication Operator
;	Statement Terminator
=	Relational Operator
<	Relational Operator
>	Relational Operator
%	Attribute Indicator
\	Character String Delimiter
(	Expression or List Delimiter
)	Expression or List Delimiter
.	Component Selector
:	Host Variable Indicator
'	Item Separator
"	Quoted Identifier Delimiter
@	Remote Access Indicator

List of Compound Symbols

Compound Symbol	Meaning
:=	Assignment Operator

=>	Association Operator
||	Concatenation Operator
**	Exponentiation Operator
<>	Relational Operator
!=	Relational Operator
~=	Relational Operator
^=	Relational Operator
..	Range Operator
<<	Label Delimiter (Begin)
>>	Label Delimiter (End)
/*	Multi-line comment delimiter (begin)
*/	Multi-line comment delimiter (end)

IV. Identifiers

1. Identifiers are used to name PL/SQL program items and units, which include constants, variables, exceptions, cursors, cursor variables, subprograms and packages.
2. Identifiers follow the following rules:
   - Identifiers must start will an alphabetic character.
   - An Identifier consists of a letter optionally followed by more letters, numerals, underscores dollar signs and number signs. Other characters such as hyphens, slashes and spaces are not allowed.
   - Identifiers can have maximum of 30 characters.
   - Identifier name and column name cannot be same in a table used in the block.
   - Identifiers are not case sensitive.

V. Literals

A literal is an explicit numeric, character, string or Boolean value which cannot be represented by an identifier.

(a) Numeric Literal: Numeric integers are represented either by simple value (129, 2.11) or by scientific integer (2E5 which means 2* 105). Numeric literals are of following two types:

- ***An Integer Literal:*** It is an optionally signed, whole number without a decimal point. For example: 129,211, +112, -29.
- ***A Real Literal:***Itis an optionally signed whole or fractional number with a decimal point. For example: 129,2.11,+1.12,-2.9.

(b) Character Literal: It is an individual character enclosed by single quotes (_...._). PL/SQL is case sensitive within the character literal. For example: It considers the literals _M‘ and _m‘ to be different.

(c) String Literal: A string literal is a sequence of zero or more characters enclosed by the single quotes. All the string literals except the null string have data types CHAR.

(d) Boolean Literal: Boolean literals are values not strings. Boolean literals are predefined values(True, False) and non-value (Null) which stands for a missing value (inapplicable or unknown value).

VI. Comments

1. We use comments to describe the purpose and use of each code segment.
2. It promotes the readability and aids understanding.
3. PL/SQL supports two comment styles:

(a) Single Line Comment: Single line comment begins with a double hyphen (—) and extends to the end of line.

(b) Multi Line Comment:Multi line comment begins with a slash-asterisk (/*) and ends with an asterisk-slash (*/) and can span multiple lines.

VII. Lexical Units

1. A line of PL/SQL text contains the group of characters known as lexical unit.
2. It can be classified as: Delimiters, Identifiers, Literals and Comments.

8.7 DATA TYPES OF PL/SQL

1. A data type specifies the storage format, constraints and valid range of values.

2. ***In PL/SQL, data types are divided into the following four categories:***

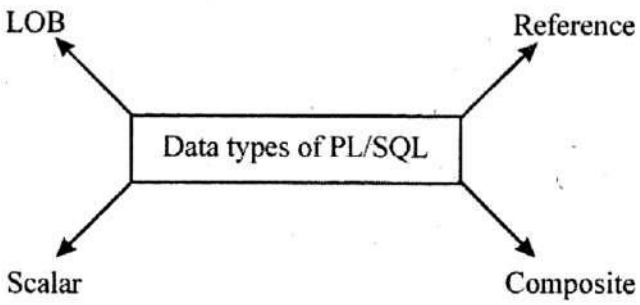


Fig. 8.4: Data Types of PL/SQL

(a) Scalar

- A scalar data type represents a single value.
- It has no internal components.
- For example: Number, Char, Boolean, Date.

(b) Composite

- *A* composite data type represents a collection of components.
- It is one that has components within it.
- It has internal components which can be manipulated individually.
- For example: Table, Record

(c) Reference

- A variable declared as reference variable can point to different storage locations over the life of a program.
- A reference data type represents a pointer that points to another item.
- For example: Ref Cursor, Ref object type.

(d) LOB

- LOB stands for large object.
- LOB data type includes BFILE, BLOB, CLOB and NCLOB.
- With LOB data type, we can store blocks of unstructured data up to 4 GB in size.
- Unstructured data means text, images, video clips and sounds.
- LOB data type allows efficient, random and piece-wise access to the data.

8.8 PL/SQL VARIABLES

1. Variables are placeholders that store the data values that can change during the execution of PL/SQL block.
2. We must first declare the variable and then use it
3. Variables can have SQL data type(such as Number, Date, Char, etc.) and PL/SQL datatype (Boolean, Binary etc.)
4. Forward references are not allowed in variables.
5. We can assign values to the variables directly from the database columns by using a 'Select Statement'.
6. **Syntax:**

   Variable _Nanie Data Type [NOT NULL: = Value];
7. ***Variables are of two types:***
   - **(a) Local Variable:** Local variables are declared in an inner block and cannot be referenced by outside the block.
   - **(b) Global Variable:** Global variables are declared in both an inner block and an outer block. It can also b6 referenced by itself.

8.9 PL/SQL CONSTANTS

1. In PL/SQL, the value of a constant remains unchanged throughout the program.
2. We can declare a constant and use it instead of actual value.
3. It is a user .defined literal value.
4. We must assign a value to a constant at the time we declare it
5. ***Syntax:***

   Constant_Name CONSTANT Data Type: = Value;

8.10 CONTROL STRUCTURES OF PL/SQL

1. Control structures are the most important PL/SQL extension to SQL.
2. It allows us to control, the behavior of the block.
3. PL/SQL supports programming language features such as conditional statements, iterative statements and sequential statements.
4. ***The various control structures are as follows:***
   - ***(i) Conditional Statement***

(ii) Iterative Statement

(iii) Sequential Statement

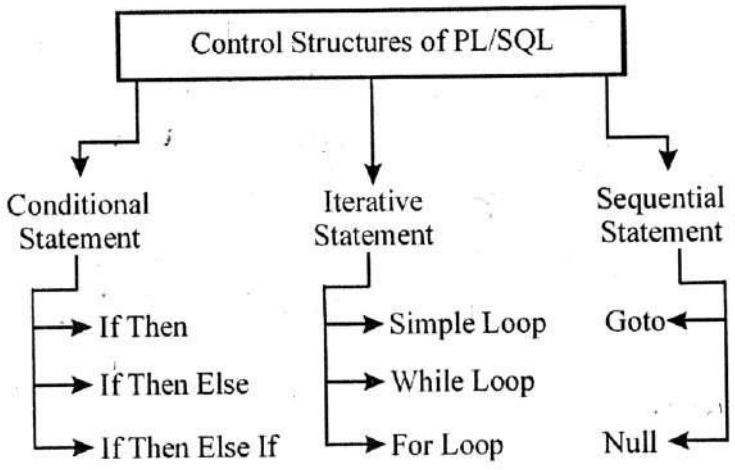


Fig.8.5: Control Structures of PL/SQL

8.10.1 Conditional Statements

It allows PL/SQL to perform the actions selectively based on conditions. The following are the forms of conditional statements:

(a) If Then Statement: It is the simplest form of If statement. It associates a condition with a sequence of statements enclosed by the keywords 'Then' and 'End If'.

Syntax:

IF condition THEN

Sequence of Statements;

END IF;

(b) If Then Else Statement: It adds the keyword 'Else' followed by the alternative sequence of statements.

Syntax:

IF condition THEN

Statement 1;

ELSE

Statement 2;

END IF;

(c) If Then Elself Statement: It adds the keyword 'Elself' to introduce the additional conditions.

Syntax:

IF condition1 THEN

Sequence of Statements1;

ELSEIF condition 2 THEN

Sequence of Statements2;

ELSE

Sequence of Statements3;

END IF;

8.10.2 Iterative Statements

These are also known as Loop Control Structures. These statements are used when we want to repeat the execution of one or more statements for specified number of times. The following are the forms of iterative control statements:

(a) Simple Loop Statement: It is also known as basic or infinite loop. It is used when a set of statements is to be executed at least once before the loop terminates. An 'Exit Condition' must be specified in the loop, which exits the process from the loop. If 'Exit Condition' is not specified in the loop, the loop will get into an infinite number of iterations (loops).

Syntax:

LOOP

Statements;

EXIT [WHEN CONDITION];

LOOP;

(b) While Loop Statement: It is used to repeat a sequence of statements until the controlling condition is 'True'. It evaluates the condition at the start of each iteration and terminates when the condition is'False'.

Syntax:

WHILE <condition>

LOOP Statements;

END LOOP;

(c) For Loop Statement: It is used to execute a set of statements for a predefined number of times. Between the given start and end integer values. Iteration occurs. The counter is incremented by 1 and loop exits when the counter reaches the value of the end integer.

Syntax:

FOR counter IN <Start Integer Value> ...<End Integer Value>

LOOP Statements;

END LOOP;

8.10.3 Sequential Statements

In PL/SQL all the blocks execute in top-down sequential process (begin statement to end statement). We use sequential statements to change the sequence of execution of statements. The following are the forms of sequential statements:

(a) GOTO Statement: It immediately transfers program control unconditionally to a named statement label. The statement label should be unique.

Syntax:

GOTO <<LabeI_name>>;

(b) NULL Statement: Itdoes nothing and passes control to the next statement.

Syntax:

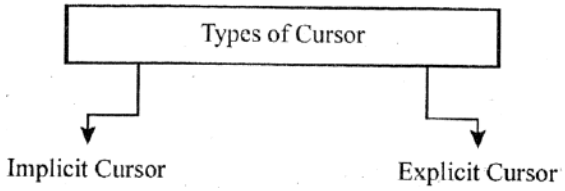
NULL;

8.11 CURSORS IN PL/SQL

1. Oracle uses work area to execute SQL statements and store processing information.
2. In PL/SQL, when we want to execute a SQL statement, Oracle server just opens the memory area (private SQL work area) to execute that command. That private SQL work area is known as cursor.
3. A cursor is a memory area (private SQL work area) that Oracle uses to execute SQL statements.

4. It is a temporary workarea created in a system memory when a SQL statement is executed.
5. This memory area (private SQL work area) is also used to store the data retrieved from the database.
6. The set of rows the cursor holds is known as Active Set.
7. A cursor can hold-more than one row, but can process only one row at a time.

8.11.1 TYPES OF CURSOR



Fig, 1L6: Types of Cursor

I. Implicit Cursor

1. Implicit cursors are declared by PL/SQL implicitly.
2. Implicit cursor is created to execute the DML (data manipulation language) statements such as Select, Insert, Update and Delete.
3. When DML operations are performed with implicit cursor, it automatically reserves somememory area for the execution of operations.
4. After the completion of DML operations, it releases the memory area.

5. Attributes of Implicit Cursor:

Oracle provides some attributes (%Found, %Not found, %'Isopen, %Rowcount) to check the status of DML operations. These attributes are known as implicit cursor attributes.

Sr. No.	Attribute	Retpra Value	Example
1	%FOUND	• The return value is 'TRUE', if the DML statements like Insert, Update and Delete affect at least one row and if Select.... Into statement return at least	SQL%FOUND

		one row. • The return value is 'FALSE', if the DML statements like Insert, Update and Delete do not affect any row and if Select.... Into statement does not return any row.	
2	%NOTFOUND	• The return value is 'TRUE', if the DML statements like Insert, Update and Delete do not affect any row and if Select.... Into statement does not return any row. • The return value is 'TALSE', if the DML statements like Insert, Update and Delete affect at least one row and if Select.... Into statement return at least one row.	SQL%NOTFOUND
3	%ISOPEN	The return value is always 'False' because Oracle automatically closes an implicit cursor after the successful execution of SQL operations.	SQL%ISOPEN
4	%ROWCOUNT	It returns (counts) the number of rows affected by the DML operations (Select, Insert, Update and Delete).	SQL%ROWCOUNT

6. Disadvantages of Implicit Cursor:

- Implicit cursors are less efficient and slightly slow as compared to explicit cursors.
- Implicit cursors provide less programming control. Unlike explicit cursors, implicit cursors cannot be opened and fetched automatically.

- Implicit cursors are more vulnerable to data errors because it has less programming control.

II. Explicit Cursor

1. Explicit cursors are declared explicitly by the user. Explicit cursors also are known as User Defined Cursors.
2. An explicit cursor is defined in the declaration sectionof the PL/SQL block.
3. Explicit cursor is declared in the declarative part of PL/SQL block to take control over query operations.
4. Explicit cursor can store multiple records, but can process one record (current row) at a time.
5. After processing the rows in the explicit cursor, we deallocate the memory occupied by the cursor using CLOSE statement.

6. Attribute of Implicit Cursor:

Orcle provides some attributes (%Found, %Not found, %Isopen, %Rowcount) to avoid errors which accessing cursors through OPEN, FETCH and CLOSE statements. These attributes are known as implicit cursor attributes.

Sr. No.	Attribute	Return Value	Example
1	%FQUND	• The return value is 'TRUE', iffetch statement returns at least one row. • The return value is 'FALSE', iffetch statement does not return any row.	Cursor_name%FOUND
2	%NOTFOUND	• The return value is 'TRUE', if not statement does not return any row. • The return value is 'FALSE', if fetch statement returns at	Cursor_name%NOTFOUND

		least one row.	
3	%ISOPEN	• TRUE,if the cursor is already open in the program. • FALSE, if the cursor is not opened in the program.	Cursor_name%ROWCOUNT
4	%ROWCOUNT	It returns (counts) the number of rowsfetchedby the fetchstatement. If no row is returned, the PL/SQL statement an error.	Cursor_name%ROWCOUNT

8.12 EXCEPTION HANDLING IN PL/SQL

1. Exception is basically an error.
2. Exceptions are identifiers in PL/SQL to terminate its main body of actions.
3. When an exception is raised, then Oracle searches for an appropriate 'exception handler'in the exception section.
4. Exception handler is used to handle the exceptions and to perform actions before the block terminates.
5. Exception handling part is used to specify the statements to be executed when an exception occurs.
6. When an error/exception arises during program execution, then the normal execution stops and the control transfers to the exception handling part of the PL/SQL block or subprogram.
7. Only one exception can be raised in the block. After the error is handled, the control does not return to the execution section.
8. An exception cannot be declared twice in the same block.

9. Exceptions declared in a block are considered as local to that block and global to its sub blocks.

10. An enclosing block cannot access exceptions declared in its sub block.

11. The PL/SQL exception message consists of three parts:

- ***Type of Exception***
- ***An Error Code***
- ***A Message***

12. Syntax:

DECLARE

Declaration Section

BEGIN

Execution Section

EXCEPTION

When Ex_Name1Then

Error Handling Statements

When Ex_Name2 Then

Error Handling Statements

When Others Then .

Error Handling Statements

END;

8.12.1 Guidelines to Avoid and Handle the Exceptions

The following are some guidelines to avoid and handle the exception:

- When there is any possibility of occurring an error, then add error checking code topredict an error.
- When there is any possibility of occurring an error, then use exception handler tohandle it.
- Test the code with different combinations of bad data to check its potential.

- Learn the names and causes of possible errors so that we can easily find and handle them.

8.12.2 Types of Exception

The exceptions are of following two types:

(a) Predefined Exceptions

- It is also known as Internal Exceptions.
- These are automatically raised by the Oracle team
- We can handle them directly within our program without declaring them.
- The following table defines some predefined exceptions:

Sr. No.	Exception Name	Oracle Error Number	Reason
1	CURSOR_ALREADY_OPEN	ORA-06511	When we open a cursor that is already open.
2	INVALID__CURSOR	ORA-01001	When we perform an invalid operation on a cursor that in not opened.
3	LOGIN_DENIED	ORA-01017	When we want to login Oracle with wrong username and password.
4	INVALID_NUMBER	ORA-01722	Conversion from character to number denied.
5	NO_DATA_FOUND	ORA-Q14Q3	When a Select...Into clause does not return any row from a table,
6	ZERO_DIVIDE	ORA-01476	When we attempt to divide a number by zero.

(b) User-defined Exceptions

- User defined exceptions are declared and defined by the user.
- These are explicitly declared in the declaration section.

- These are raised explicitly by raise statements,unlike predefined exceptions that are raised implicitly.

8.13 EXCEPTION PROPAGATION

When an exception is raised in the executable section of PL/SQL block, then it will handle in the following manner.

1. If an exception is handled in the current block, then the control passes to the enclosing block.
2. If an exception is not handled in the current block, then exception propagates.
    - Propagates means circulates. We just circulate an exception until it handles.
    - It means exception is sent to enclosing blocks from inside to outside until a handler is found or no more blocks to search,
    - If no handler is found for exception, then an exception is sent to the host environment.

8.14 SUBPROGRAMS

1. In PL/SQL, programs can be stored in the database as stored programs. Such stored programs are known as Subprograms.
2. Subprograms can be invoked whenever required.
3. We can declare and define a subprogram within either a PL/SQL block or anothersubprogram.
4. The main function of subprogram is to break the program into smaller and manageable parts because smaller programs are easier to maintain and debug as compared to large program.
5. These smaller and manageable parts are known as modules and this process is known as Modularization.
6. Subprograms provide easy maintenance because code is located at one place and we can easy modify it in this single location.

8.14.2 Advantages of Subprograms

- Modularity: Subprogram breaks the program into smaller and manageable parts which are easier to maintain and debug.
- Extensibility: It provides the facility to add functionality. It allows creatingnew program modules without affecting the old ones.
- Reusability: Any number of applications can use and execute the subprograms. A subprogram can be used by various number of applications is known as reusability.
- Better Performance: Subprogram can reduce the number of calls from application to Oracle, Reducing calls automatically increase the performance.
- Security: Subprogram helps to maintain the database security. It can restrict the users to perform specific tasks with security privileges.
- Abstraction: Subprograms aid abstraction as all the internal and compilation details are hidden from the user.
- Memory Allocation: Subprograms require less memory because it loads only a single copy of subprogram into memory for multiple users.

8.14.2 Block Structure of PL/SQL Subprograms

PL/SQL Subprograms consists of three sections which are as follows:

(a)Declarative Section

(b) Executable Section

(c)Exception/Error Handling Section

(i) Declarative Section: It contains the declarations of types, cursors, constants, variables,exceptions and nested subprograms.

(ii) Executable Section: It contains statements that assign values, control execution andmanipulate the data.

(iii) Exception/Error Handling Section: It contains exception handlers which deal with exceptions (errors) raised during execution.

8.14.3 Types of Subprograms

The subprograms are of following two types:

(a) Procedures

(b) Functions

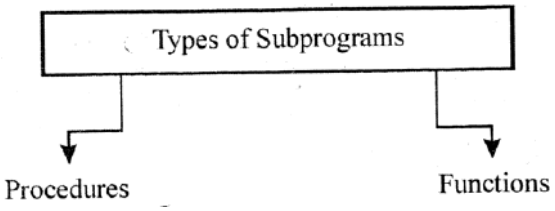


Fig. 11.7:Types of Subprograms

8.14.4 Procedures

1. We use procedure to perform an action.
2. It is a subprogram that performs specific task.
3. It may or may not return a value.
4. It has declaration section, an executable section and exception handling section.
5. It accepts parameters of type IN, OUT and IN OUT.
6. ***Syntax:***

CREATE [OR REPLACE] PROCEDURE Procedure_Name

[{Parameter 1 [Mode 1] Datatype1, Parameter 2 [Mode 2] Datatype2,…}]

IS I AS

[Local Declaration]

BEGIN

PL/SQLExecutable Statements

[EXCEPTION

Exception Handler]

END [PROCEDURE NAME];

7. ***The following are the two types of procedures:***

(a) Local Procedures

- Local procedure is defined in the declaration section of PL/SQL block.
- Its scope is limited to the parent block from where it belongs.
- It cannot be defined outside the block from where it created.
- It can be called anywhere in the module section area.

(b) Stored Procedures

- It performs one or more specific tasks.
- It displays an error message if any error occurs.

8.14.5 Functions

1. The difference between a procedure and a function is that a function must return a value but a procedure may or may not return a value.
2. We use function to compute a value.
3. Function has a return clause.
4. Like procedure, a function has declaration section, an executable section and exception handling section.
5. ***Syntax:***\

```
CREATE [OR REPLACE] FUNCTIONFunction_Name
        [{Parameter1 [Mode 1] Datatype1, Parameter 2 [Mode 2]Datatype1,. .......}]
        IS I AS
        [Local Declaration]
BEGIN
PL/SQL Executable Statements
[EXCEPTION
        Exception Handler]
END [FUNCTION NAME];
```

6. ***The following are the two types of functions:***
   - *(a)* *Local* ***Functions:*** It is defined in the declaration section of PL/SQL block. It cannotbe called by any block of PL/SQL outside the inner block.
   - ***(b)*** ***Stored Functions:*** It is also known as user defined function. It is set of PL/SQLstatements we can call by name.

8.15 STORED PACKAGES

1. A stored packageor package is the collection of PL/SQL elements.
2. It is a database object which groups logically,related PL/SQL objects such as cursors,exceptions, subprograms,procedures, functions, variables etc. into a single container.
3. It stores PL/SQL objects which perform similar tasks into a single container.

4. It is like a library which stored once in Oracle database and used by many applications.

5. The stored packages are of following two parts:

(a) Package Specification

(b) Package Body

(a) Package Specification

- In short form, package specification is known as 'Spep'.
- It holds public declarations, which are visible to the application.
- The scope of these declarations is local to the database schema andglobal to the package.
- It is required when we create a new package.
- We use the 'Create Package Statement' to create a new package or package specification.
- ***Syntax:***

  PACKAGE Package_Name

  IS

  [Declaration of Variables and Types]

  [Headers of Cursors]

  [Headers of Procedures and Functions]

  END [Package Name];

(b) Package Body

- It holds the implementation details and private declarations.
- It contains the code that implements the package specification
- It fully defines the cursors and subprograms declared in the package specification.
- We use the 'Create Package Body Statement' to create a package body.
- ***Syntax:***

  PACKAGE BODY Package_Name

  IS

[Declaration of Variables and Types]

[Header and Select Statement of Cursors]

[BEGIN

Executable Statements]

[EXCEPTION

Exception Handlers]s

END [Package Name];

6. 'Drop Package Command' is used to drop a package. It drops the specification and body of the package.

Syntax:

DROP PACKAGE <Package_Name>;

8.16 TRIGGERS

1. A trigger is a PL/SQL block structure which is fired when DML statements (Select, Insert,Update, Delete) execute on a database table.
2. It is stored in the database and executed automatically when specific event (user actions orsystem actions) occurs in the database.
3. It is a PL/SQL program unit which associates with specific database table.
4. Triggers are stored as text and compiled at execution time.
5. It does not include much code in them but it call out previously stored procedures or packages.
6. A database trigger includes SQL and PL/SQL statements to executes a unit and invokeother stored procedures.
7. It can be defined on tables and on views.
8. It is used to improve the performance of Oracle in order to provide a more convenientdatabase.
9. Trigger cannot perform commit or rollback operations.

8.16.1 Guidelines for Designing Triggers

1. Triggers can be used only when it is necessary. The excessive use of triggers can result in complex interdependencies which may be difficult to maintain in large applications.
2. Triggers guarantee that when a specific operation is performed, related actions are performed.
3. Limit the size of triggers.
4. If the logic for the trigger is very lengthy, create stored subprograms, put the code into stored subprograms and invoke them in the trigger body.
5. Do not create recursive triggers.
6. Do not define triggers that duplicate features already built into the Oracle database.
7. Use database triggers only for centralized, global operations that must fire for the triggering statements, regardless of which user or database application issues the statement.

8.16.2 States of Triggers

A trigger can be in either of two states:

I. Enabled State

1. An enabled trigger executes its trigger body if a triggering statement is entered and the trigger restriction (if any) evaluates to True.
2. By default, a trigger is created in enabled state.
3. To enable all triggers defined for a specific table, use the *'Alter Table Statement'* with *'Enable Clause'* and *'All Triggers Option'*.

   For example: To enable all the triggers defined for the table 'Student', the statement will be:

   SQL>ALTER TABLE STUDENT ENABLE ALL TRIGGERS;
4. To enable a disable trigger, use the *'Alter Table Statement'* with *'Enable Clause'*.

   For example: To enable the disabled trigger named 'Student', the statement will be:

   SQL>ALTER TRIGGER STUDENT ENABLE;

II. Disabled State

1. A disabled trigger does not execute its trigger body, even it a triggering statement is entered and the trigger restriction (if any) evaluates to True.
2. To create a trigger in disabled state, we use the disable clause in the create trigger statement.
3. To disable all triggers defined for a specific table, use the *'Alter Table Statement'* with *'Disable Clause'* and *'All Triggers Option'*.

   For example: To disable all the triggers defined for the table 'Student', the statement will be:

   SQL>ALTER TABLE STUDENT ENABLE ALL TRIGGERS;
4. To disable a trigger, use the *'Alter Table Statement'* with *'Disable Clause'*.

   For example: To disabled the trigger named 'Student', the statement will be:

   SQL>ALTER TRIGGER STUDENT DISABLE;

8.16.3 Parts of Trigger

A trigger has the following three parts:

I. Triggering Event or Statement

II. Trigger Restriction

III. Trigger Action

When any of the events occur, the trigger fires automatically and PL/SQL block performs the action. The trigger action is a procedure thatcontains the code to be executed when the trigger fires.

I. Triggering Event or Statement

1. It is the SQL statement that causes a trigger to be fired.
2. It can specify multiple SQL statement.
3. It can be an Insert, Update or Delete statement on a table.

II. Trigger Restriction

1. It must specify a Boolean expression that must be 'True' for the trigger to fire.
2. Its function is to control the execution of a trigger conditionally.

3. The trigger action is not run (executed) if the trigger restriction evaluates to 'False' or 'Unknown'.

III. Trigger Action

1. It is the procedure that contains the code to be executed when the trigger fires.
2. Trigger Action executes the PL/SQL block when the triggering statement is issued and trigger restriction evaluatesto'True'.

8.16.4 Types of Trigger

The different types of triggers are as follows:

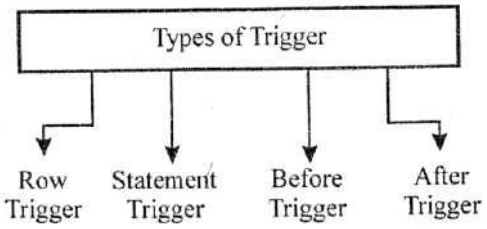


Fig. 11.8: Types of Triggers

I. Row Triggers

1. A row trigger is fired each time the table is affected by the triggering statement. A row trigger fires once onbehalf of the triggering statement.
2. If a triggering statement affects no rows; a row trigger is notrun.
3. Row triggers are useful if trigger action depends on number of rows affected.

II. Statement Triggers

A statement trigger is fired once on behalf of the triggering statement, regardless of the number of rows in the table that the triggering statement affects, even no rows are affected. It is default type of trigger.

III. Before Triggers

1. Before triggers run the trigger action before the triggering statement is run.
2. These triggers are used to check the validity of data before the action is performed.

IV. After Triggers

1. After triggers run the trigger action after the triggering statement is run.
2. These triggers are used when we want the triggering statement to complete before executing the trigger action.
3. Instead of Triggers

(i) Instead of trigger provide a transparent way of modifying views that cannot be modifieddirectly through DML statement.

(ii) With the help of this trigger, Oracle fires the trigger instead of executing the triggeringstatement.

8.16.5 Creating and Dropping A Trigger

Creating a Trigger

1. We use the 'Create Trigger Statement' to create a trigger.
2. By default, a trigger is created in enabled state.
3. If we want to create a trigger in disabled state, thenwe use the 'Disable Clause' of the 'Create Trigger Statement'.
4. **Syntax:**

```
CREATE [OR REPLACE] TRIGGER Trigger_Name
BEFORE I AFTER I INSTEAD OF
    INSERT [OR] I UPDATE [OR] I DELETE [OF Column_Name]
    ONTable_Name
[REFERENCING OLD AS Old, NEW AS New]]
[FOR EACH ROW [WHEN CONDITION]]
BEGIN
———SQL STATEMENTS
END;
```

Dropping a Trigger

1. We can delete the trigger with the help of 'Drop Command'.
2. If we want to delete the trigger'Monika$^{9}_{5}$ then the statement will be:

```
SQL>DROP TRIGGER MONIKA;
```

8.16.7 Advantages of Triggers

- It maintains the replicate tables.
- It prevents the invalid transactions.
- It implements complex security authorizations.
- It implements complex business rules which cannot be implemented by using integrity constraints.
- It automatically generates derived columns.
- It automatically performs an action when another concerned action takes place.

Questions

1. What is the difference between SQL and PL/SQL? Explain.
2. Discuss the block structure, of PL/SQL block.
3. Explain the elements of PL/SQL in detail.
4. Discuss the control structures of PL/SQL in detail.
5. Discuss the architecture of PL/SQL block.
6. Explain the following:
   (a) PL/SQL Variables.
   (b) PL/SQL Constants.
   (c) Data types of PL/SQL.
7. What is exception? Discuss the usages of pre-defined and user-defined exceptions in PL/SQL.
8. What is cursor? Discuss the role of implicit and explicit cursor.
9. How is exception handling performed in PL/SQL?
10. What is subprogram? Explain the types and block structure of PL/SQL subprogram in detail.
11. What are stored packages?
12. What is trigger? Explain the various types of triggers in Oracle.