RJS First Grade College

Koramangala, Bengaluru – 560034

Department of Computer Science and Applications

Subject: Data Base Management and Systems

Chapter-1 Introduction to DBMS

1. Define Data and Data base?

Data is a representation of Facts, Figures, statistics having no particular meaning. It can be in the form of numbers, characters, symbols or pictures. Database is made up of two components mainly, data and a meaningful method for accessing and manipulating data. Without these two, a database is just a random set of data. A more precise example of a database can be a dictionary, which stores a large equantity of data as Key-Value pairs.

2. Define DBMS?

Database Management System (**DBMS**) is a software for storing and retrieving users' data while considering appropriate security measures. ... The **DBMS** accepts the request for data from an application and instructs the operating system to provide the specific data

3. Explain any two functions of DBMS?

DBMS performs several important functions that guarantee the integrity and consistency of the data in the database. The most important functions of Database Management System are

Data Dictionary Management,

Data Storage Management,

Data Transformation and Presentation,

Security Management,

Multi user Access Control,

4. State two advantages of DBMS?

The DBMS serves as the intermediary between the user and the database. The main advantages of DBMS is Improved data sharing.
An advantage of the database management approach is, the DBMS helps to create an environment in which end users have better access to more and better-managed data.

5. **Improved data security**

The more users access the data, the greater the risks of data security breaches. Corporations invest considerable amounts of time, effort, and money to ensure that corporate data are used properly

6. Define Data Base Administration?

A **database administrator** (**DBA**) is a specialized computer systems administrator who maintains a successful database environment by directing or performing all related activities to keep the data secure. The top responsibility of a **DBA** professional is to maintain data integrity.

7. Describe different data base users?

Different DBA users are as follows

Application user. The application user is someone who accesses an existing application program to perform daily tasks.

Sophisticated user. Sophisticated users are those who have their own way of accessing the database. ...

Application Programmers. ...

Database Administrators (DBA)….

Long Answers

1. Different types of people behind DBMS?

People involved behind DBMS is divided into Actors on the Scene and workers behind the scene.

Actors on the scene

They are the people who actually controls the content. They are classified as

DBA

A **database administrator** (**DBA**) is a specialized computer systems administrator who maintains a successful database environment by directing or performing all related activities to keep the data secure. The top responsibility of a **DBA** professional is to maintain data integrity.

Database Designers and

They are responsible for identifying the data to be stored in the DB for choosing appropriate structures to represent and store the data.It is his responsibility to commu icate with all DB users to understand their requirements and to meet those requirements

End Users

Application **user**. The application **user** is someone who accesses an existing application program to perform daily tasks.

Sophisticated **user**. Sophisticated **users** are those who have their own way of accessing the **database**. ...

Application Programmers. ...

Database Administrators (DBA)….

System Analyst and Application programmers.

They identify the requirements of end users and deelop specifications for transactions. Application programmers implement these specifications as they test , debug , document and maintain these transactions.

workers behind the scene.

Those are not interested in DBMS,they include DBMS system designers and implementers , Tool Developers, Operators and maintenance personnel.

DBMS system designers and implementers ,

They and implement DBMS modules and interfaces as a software package.

Tool Developers

They are the persons who design and implement tools the software packages that facilitate DB design and use that help to improve performance. These packages include DB design, performance monitoring graphical interfaces , prototyping , simulation and test data generation.

Operators and maintenance personnel

They are the system administration personnel who are responsible for the actual running and maintenance of the hardware and S/W.

2. Explain the advantages and disadvantages of DBMS?

Advantages	Disadvantages
Reduces data redundancy	The cost of H/W and S/W is very expensive
Restricting unauthorized access	Conversion of data requires more storage capacity and upgradation involves expensive process.
Providing storage for efficient query processing	Training of the staff to use DBMS is mandatory and it requires lots of investment.
Providing backup and recovery	Appointing Technical staff is needed for the organization. So the organization has to pay more amount of salary .
Providing multiple user interfaces	Database damage

3. Explain the characteristics of database approach?

There are number of characteristics

The main characteristics of the database approach (feature of database approach) versus the file-processing approach are the following:

1. **Self-describing nature of a database system**
2. **Insulation between programs and data, and data abstraction**
3. **Support of multiple views of the data**
4. **Sharing of data and multiuser transaction processing**

- Self-Describing Nature of a Database System

A fundamental characteristic of database approach is that the database system contains not only the database itself but also a complete definition or description of the database structure and constraints. This definition is stored in the DBMS catalog, which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data. The information stored in the catalog is called meta-data, and it describes the structure of the primary database.

- Insulation between Programs and Data, and Data Abstraction

In traditional file processing, the structure of data files is embedded in the application programs, so any changes to the structure of a file may require changing all programs that access that file. By contrast, DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property program-data independence.

- Support of Multiple Views of the Data

A database typically has many users, each of whom may require a different perspective or view of the database. A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored. Some users may not need to be aware of whether the data they refer to is stored or derived. A multiuser DBMS whose users have a variety of distinct applications must provide facilities for defining multiple views. For example, one

user of the database may be interested only in accessing and printing the transcript of each student; the view for this user is shown in Figure 1.5(a). A second user, who is interested only in checking that students have taken all the prerequisites of each course

- Sharing of Data and Multiuser Transaction Processing

A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database. The DBMS must include concurrency control software to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct.

4. Explain the Roles and Responsibilities of DBA

A Database Administrator is a person or a group of person who are responsible for managing all the activities related to database system

DBA is responsible for installing the database software. He configure the software of database and then upgrades it if needed.

- Deciding the hardware device

Depending upon the cost, performance and efficiency of the hardware, it is DBA who have the duty of deciding which hardware devise will suit the company requirement.

- Managing Data Integrity

Data integrity should be managed accurately because it protects the data from unauthorized use. DBA manages relationship between the data to maintain data consistency.

- Decides Data Recovery and Back up method

If any company is having a big database, then it is likely to happen that database may fail at any instance. It is require that a DBA takes backup of entire database in regular time span.

- Tuning Database Performance

Database performance plays an important role for any business. If user is not able to fetch data speedily then it may loss company business. So by tuning an modifying sql commands a DBA can improves the performance of database.

- Capacity Issues

All the databases have their limits of storing data in it and the physical memory also has some limitations. DBA has to decide the limit and capacity of database and all the issues related to it.

- Database design

The logical design of the database is designed by the DBA. Also a DBA is responsible for physical design, external model design, and integrity control.

- Database accessibility

DBA writes subschema to decide the accessibility of database. He decides the users of the database and also which data is to be used by which user.

- Decides validation checks on data

DBA has to decide which data should be used and what kind of data is accurate for the company. So he always puts validation checks on data to make it more accurate and consistence.

- Monitoring performance

If database is working properly then it doesn't mean that there is no task for the DBA. Yes f course, he has to monitor the performance of the database. A DBA monitors the CPU and memory usage.

- Decides content of the database

A database system has many kind of content information in it. DBA decides fields, types of fields, and range of values of the content in the database system. One can say that DBA decides the structure of database files.

5.Explain Database Applications?

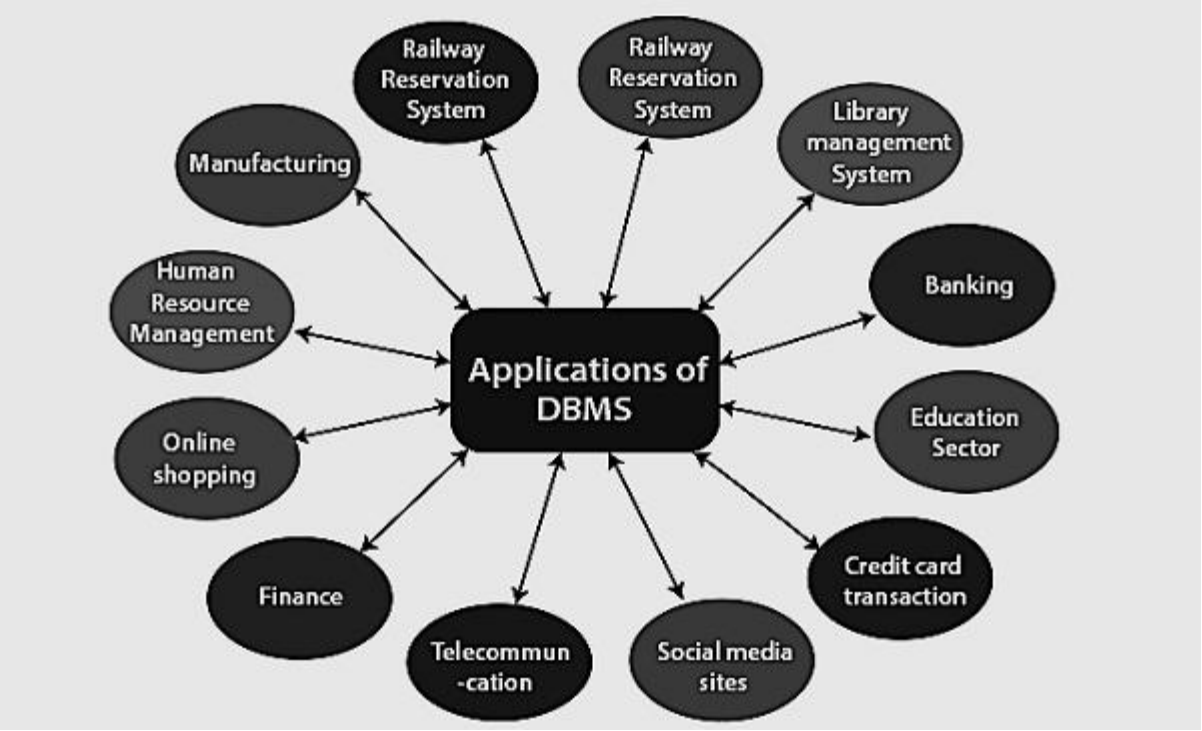
The Database Management System (DBMS) is defined as a software system that allows the user to define, create and maintain the database and provide control access to the data.

It is a collection of programs used for managing data and simultaneously it supports different types of users to create, manage, retrieve, update and store information.

Applications of DBMS

In so many fields, we will use a database management system.

- **Railway Reservation System** − The railway reservation system database plays a very important role by keeping record of ticket booking, train's departure time and arrival status and also gives information regarding train late to people through the database.
- **Library Management System** − Now-a-days it's become easy in the Library to track each book and maintain it because of the database. This happens because there are thousands of books in the library. It is very difficult to keep a record of all books in a copy or register. Now DBMS used to maintain all the information related to book issue dates, name of the book, author and availability of the book.
- **Banking** − Banking is one of the main applications of databases. We all know there will be a thousand transactions through banks daily and we are doing this without going to the bank. This is all possible just because of DBMS that manages all the bank transactions.
- **Online Shopping** − Now-a-days we all do Online shopping without wasting the time by going shopping with the help of DBMS. The products are added and sold only with the help of DBMS like Purchase information, invoice bills and payment.
- **Human Resource Management** − The management keeps records of each employee's salary, tax and work through DBMS.
- **Manufacturing** − Manufacturing companies make products and sell them on a daily basis. To keep records of all those details DBMS is used.
- **Airline Reservation system** − Just like the railway reservation system, airlines also need DBMS to keep records of flights arrival, departure and delay status.



6.When DBMS should not be used?

In spite of the advantages of using a DBMS, there are a few situations in which a DBMS may involve unnecessary overhead costs that would not be incurred in traditional file processing. The overhead costs of using a DBMS are due to the following:

High initial investment in hardware, software, and training

The generality that a DBMS provides for defining and processing data

Overhead for providing security, concurrency control, recovery, and integrity functions.
real-time requirements for some application programs that may not be met because of DBMS overhead Embedded systems with limited storage capacity, where a general-purpose DBMS would not fit

No multiple-user access to data

Chapter-2 Database concepts and its architecture

1. What is Datamodel and Relational datamodel?

Data model is a set of data structures and conceptual tools used to describe the structure of a database.

Relational datamodel

Its defined as a DB that allows you to group its group its dataitems into one / more independent tables that can be related to one another by using common fields related to each table.

2. What is Schema?

The database schema of a database is its structure described in a formal language supported by the database management system (DBMS). The term "schema" refers to the organization of data as a blueprint of how the database is constructed (divided into database tables in the case of relational databases).

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It’s the database designers who design the schema to help programmers understand the database and make it useful.

3. What is mapping?

The process of transforming requests and results between different levels is called mappings. There are two types of mapping in DBMS- External Mapping and Internal l Mapping

External Mapping:

It provides the correspondence among the records and relationships of the external and conceptual view.

Internal l Mapping:

It enables DBMS to find the actual records in physical storage that constitute a logical record in conceptual mapping

4. Explain DDL?

Structured Query Language(SQL)

SQL uses certain commands like Create, Drop, Insert etc. to carry out the required tasks.

These SQL commands are mainly categorized into four categories as:

DDL – Data Definition Language

DQl – Data Query Language

DML – Data Manipulation Language

DCL – Data Control Language

DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

5. Explain DML

DML(Data Manipulation Language) : The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

Examples of DML

INSERT – is used to insert data into a table.

UPDATE – is used to update existing data within a table.

DELETE – is used to delete records from a database table.

6. Explain Hierarchical and Network model?

A hierarchical database model is a data model in which the data are organized into a tree-like structure. The data are stored as records which are connected to one another through links. A record is a collection of fields, with each field containing only one value.

A network model is a database model that is designed as a flexible approach to representing objects and their relationships. A unique feature of the network model is its schema, which is viewed as a graph where relationship types are arcs and object types are nodes.

7. Explain Logical data dependence and physical data dependence?

In DBMS there are two types of data independence

Physical data independence

Logical data independence.

It means we change the physical storage/level without affecting the conceptual or external view of the data. The new changes are absorbed by mapping techniques.

Logical data independence is the ability to modify the logical schema without causing application program to be rewritten.

Long Answers

1. Explain different types of data models

The different types of data models in DBMS that are used are as given below:

1. Flat Data Model.
2. Entity-Relationship Model.
3. Relation Model.
4. Record base Model.
5. Network Model.
6. Hierarchical Model.
7. Object-oriented Data Model.
8. Object Relation Data Model.

More items...

Flat Data Model:

Flat data model is the first introduced traditional data model where data is kept in the same plane. This is a very old model which is not much scientific.

Entity Relationship Data Model:

The Entity relationship data model structure based on the impression of the real world entities and the existing relationship between them. An entity contains a real-world property called attribute. Attributes are defined by a set of values known as domains. For example, in an office the employee is an entity, the office is the database, employee ID, name are the attributes. The logical association between the different entities are known as the relationship among them.

Relational Data Model:

The data model allows the data to be stored in tables called relation. The relations are normalized and the normalized relation values are known as atomic values. Each of the rows in a relation is called tuples which contains the unique value. The attributes are the values in each of the columns which are of the same domain.

Network Data Model:

In the network data model, all the entities are organized in graphical representations. There may be several parts in the graph in which the entities can be accessed.

Hierarchical Data Model:

The hierarchical model is based on the parent-child hierarchical relationship. In this model, there is one parent entity with several children entity. At the top, there should be only one entity which is called root. For example, an organization is the parent entity called root and it has several children entities like clerk, officer and many more.

Object-oriented Data Model:

An object-oriented data model is one of the most developed data models which contains video, graphical files, and audio. This consists of the data piece and the methods in the form of database management system instructions.

Record base Data Model:

The record-based data model is used to determine the overall design of the database. This data model contains different kinds of record types. Each of the record types has a fixed length and a fixed number of fields.

Object-relational Data Model:

The object-relational data model is a powerful data model but for the design of the object-relational data, the model is very complex. This model gives efficient results and widespread with huge application thus some part of the complexity problem can be ignored because of this. It also offers features like working with other data models. Using the object-relational data model we can work with the relational model also.

2. Explain DBMS Architecture.

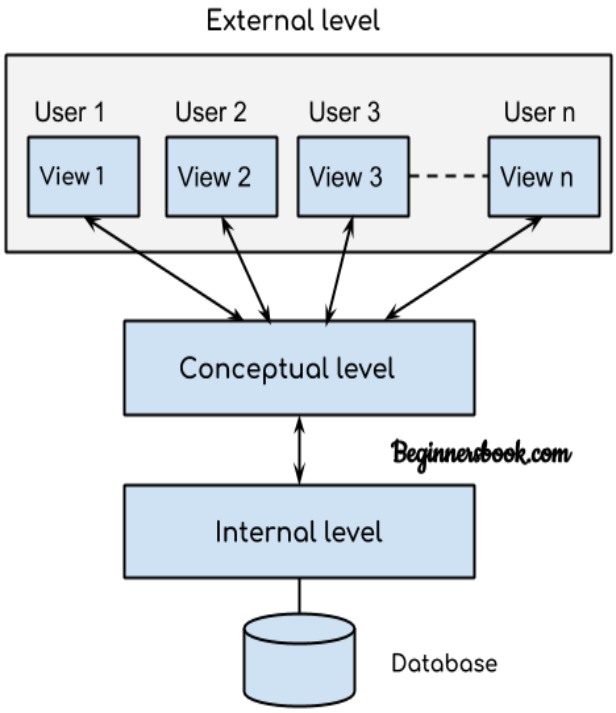
There are three levels of DBMS architecture
External level, conceptual level and Internal level

External Level

This is the highest level in the three level architecture and closest to the user. It is also known as the view level. The external level only shows the relevant database content to the users in the form of views and hides the rest of the data.

The user doesn't need to know the database schema details such as data structure, table definition etc. user is only concerned about data which is what returned back to the view level after it has been fetched from database (present at the internal level).

External level is the "top level" of the Three Level DBMS Architecture.



2. Conceptual level

It is also called logical level. The whole design of the database such as relationship among data, schema of data etc. are described in this level.

Database constraints and security are also implemented in this level of architecture. This level is maintained by DBA (database administrator).

3. Internal level

This level is also known as physical level. This level describes how the data is actually stored in the storage devices. This level is also responsible for allocating space to the data. This is the lowest level of the architecture.

3. Explain client server Architecture

The client-server architecture is also termed as a network-computing structure because every request and their associated services are distributed over a network.

In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested, process it and deliver the data packets requested back to the client.

the clients and servers are two different computers in different parts of the world that are connected through the Internet.

The architecture of DBMS depends on the computer system on which it runs. For example, in a client-server DBMS architecture, the database systems at server machine can run several requests made by client machine. We will understand this communication with the help of diagrams.

There are three types of DBMS architecture:

1. Single tier architecture

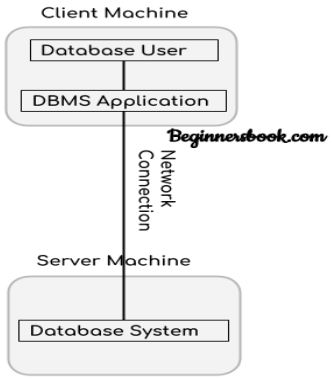
In this type of architecture, the database is readily available on the client machine, any request made by client doesn't require a network connection to perform the action on the database.

For example, lets say you want to fetch the records of employee from the database and the database is available on your computer system, so the request to fetch employee details will be done by your computer and the records will be fetched from the database by your computer as well. This type of system is generally referred as local database system.

2. Two tier architecture

In two-tier architecture, the Database system is present at the server machine and the DBMS application is present at the client machine, these two machines are connected with each other through a reliable network as shown in the above diagram.

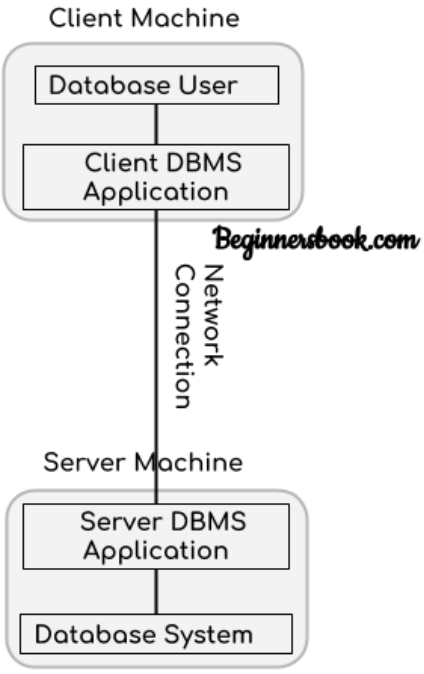
Whenever client machine makes a request to access the database present at server using a query language like sql, the server perform the request on the database and returns the result back to the client. The application connection interface such as JDBC, ODBC are used for the interaction between server and client.



Two-Tier architecture

3. Three tier architecture

In three-tier architecture, another layer is present between the client machine and server machine. In this architecture, the client application doesn't communicate directly with the database systems present at the server machine, rather the client application communicates with server application and the server application internally communicates with the database system present at the server.



Three-Tier architecture

4. Write a short note on database languages?

A data sublanguage mainly has two parts:

Data Definition Language (DDL) and Data Manipulation Language (DML).

The Data Definition Language is used for specifying the database schema, and the Data Manipulation Language is used for both reading and updating the database. These languages are called data sub-languages as they do not include constructs for all computational requirements.

Data Definition Language (DDL) statements are used to classify the database structure or schema. It is a type of language that allows the DBA or user to depict and name those entities, attributes, and relationships that are required for the application along with any associated integrity and security constraints. Here are the lists of tasks that come under DDL:

CREATE - used to create objects in the database

ALTER - used to alters the structure of the database

DROP - used to delete objects from the database

TRUNCATE - used to remove all records from a table, including all spaces allocated for the records are removed

COMMENT - used to add comments to the data dictionary

RENAME - used to rename an object

Data Manipulation Language

A language that offers a set of operations to support the fundamental data manipulation operations on the data held in the database. Data Manipulation Language (DML) statements are used to manage data within schema objects. Here are the lists of tasks that come under DML:

SELECT - It retrieves data from a database

INSERT - It inserts data into a table

UPDATE - It updates existing data within a table

DELETE - It deletes all records from a table, the space for the records remain

MERGE - UPSERT operation (insert or update)

CALL - It calls a PL/SQL or Java subprogram

EXPLAIN PLAN - It explains access path to data

LOCK TABLE - It controls concurrency

Data Control Language

There are another two forms of database sub-languages. The Data Control Language (DCL) is used to control privilege in Database. To perform any operation in the database, such as for creating tables, sequences or views we need privileges. Privileges are of two types,

System - creating a session, table, etc. are all types of system privilege.

Object - any command or query to work on tables comes under object privilege. DCL is used to define two commands. These are:

Grant - It gives user access privileges to a database.

Revoke - It takes back permissions from the user.

Transaction Control Language (TCL)

Transaction Control statements are used to run the changes made by DML statements. It allows statements to be grouped into logical transactions.

COMMIT - It saves the work done

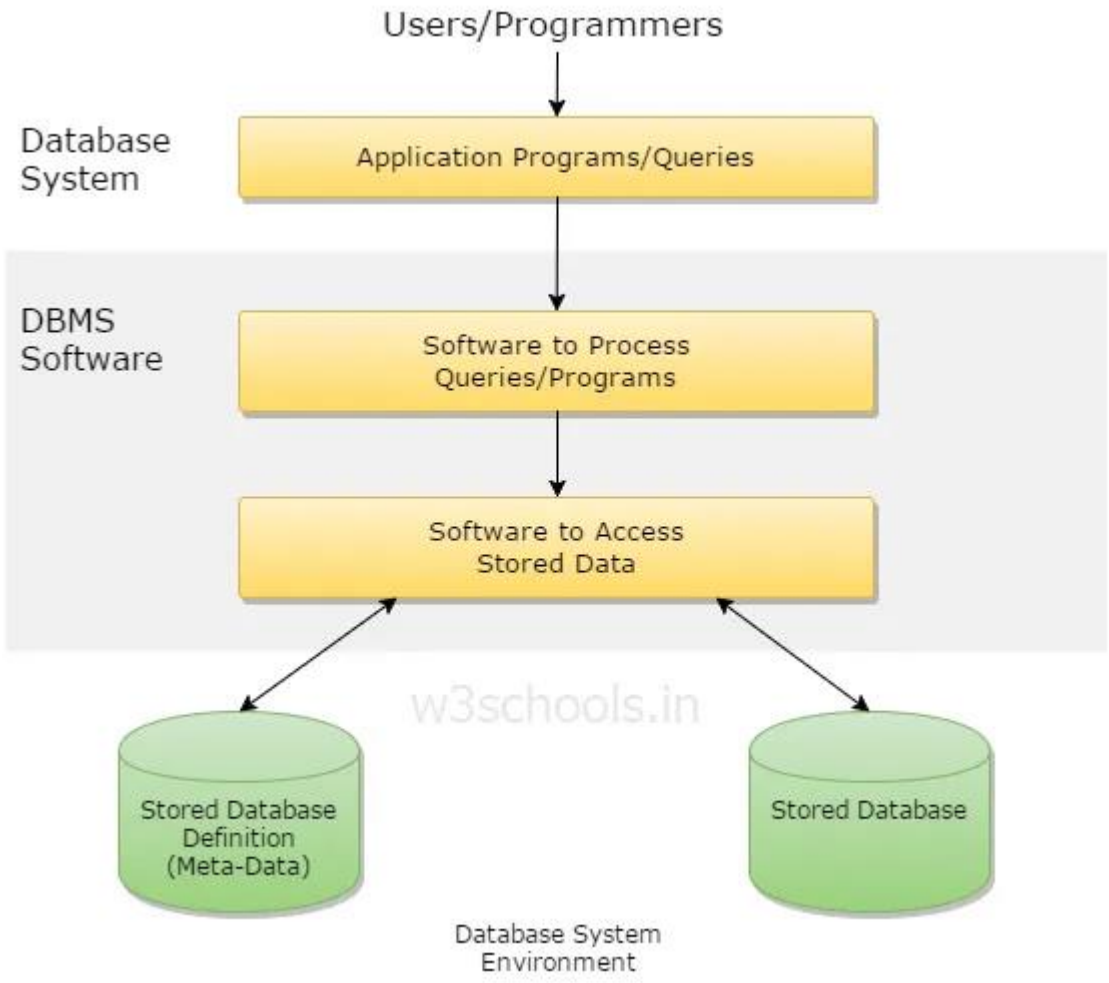
SAVEPOINT - It identifies a point in a transaction to which you can later roll back

ROLLBACK - It restores the database to original since the last COMMIT

SET TRANSACTION - It changes the transaction options like isolation level and what rollback segment to use.

5.Explain Data Base System Environment?

One of the primary aims of a database is to supply users with an abstract view of data, hiding a certain element of how data is stored and manipulated.



A database environment is a collective system of components that comprise and regulates the group of data, management, and use of data, which consist of software, hardware, people, techniques of handling database, and the data also.

Here, the hardware in a database environment means the computers and computer peripherals that are being used to manage a database.

The software means the whole thing right from the operating system (OS) to the application programs that include database management software like M.S. Access or SQL Server.

Again the people in a database environment include those people who administrate and use the system.

The techniques are the rules, concepts, and instructions given to both the people and the software along with the data with the group of facts and information positioned within the database environment.

5. What are the classifications of DBMS?

   Database management systems can be classified based on several criteria, such as the data model, user numbers and database distribution.

   ## Classification Based on User Numbers

   A DBMS can be classification based on the number of users it supports. It can be a *single-user database system*, which supports one user at a time, or a *multiuser database system*, which supports multiple users concurrently.

   ## Classification Based on Database Distribution

   There are four main distribution systems for database systems and these, in turn, can be used to classify the DBMS.

   centralized database system: the DBMS and database are stored at a single site that is used by several other systems too

   Distributed database system: the actual database and the DBMS software are distributed from various sites that are connected by a computer network

   Heterogeneous distributed database system: different sites might use different DBMS software, but there is additional common software to support data exchange between these sites

   Homogeneous distributed database systems: use the same DBMS software at multiple sites

Multiuser database system: a database management system which supports multiple users concurrently

Object-oriented data model: a database management system in which information is represented in the form of objects as used in object-oriented programming

Single-user database system: a database management system which supports one user at a time

Traditional models: data models that preceded the relational model

Chapter-3 Data Modeling Using Entity-Relational Model

1. **What is an entity?**
   An entity set is a collection of similar types of entities. An entity set may contain entities with attribute sharing similar values. For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties. Entity sets need not be disjoint.

2. **Define relationship. Give an example**
   Relationships are associations between entities. Typically, a relationship is indicated by a verb connecting two or more entities.
   Example: Employees are assigned to projects.

3. **Name the different types of relationship types?**
   There are three types of cardinalities for Binary Relationships:
   1. One-to-One
   2. One-to-many
   3. Many-to-Many
   4. Many-to-One

4. **What is weak entity and strong entity?**
   Weak Entity
   An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.
   Strong Entity
   Entity types that have primary keys are called as strong entity.

5. **What is an attribute? Mention its types**

   Entities are represented by means of their properties, called attributes. All attributes have values. For example, a student entity may have name, class, and age as attributes.

   There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

   Types of Attributes

   Simple attribute.

   Composite attribute

   Derived attribute

   Single-value attribute Multi-value attribute

6. **What is an entity set?**

   An Entity Set is a collection of related entities all of the same type. An Entity Set is also known as the extension of an Entity type.

7. **Define generalization.**

   Generalization proceeds from the recognition that a number of entity sets share some common features are described by the same attributes and participate in the same relationships sets.

8. **What is ER-diagram?**

   The graphical representation of organizational system elements and the association among the elements is called ER Diagram.

9. **What is specialization?**

   The process of designating sub-groupings within an entity set is called specialization.

10. **What is degree of relationship?**

    The number of participating enity types is called degree of relationship.

11.Explain Degree of Relationship.

Relationship: The association among entities is called a relationship. For example, an employee **works_at** a department, a student **enrolls** in a course. Here, Works_at and Enrolls are called relationships.

Relationship Set

A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called **descriptive attributes**.

Degree of Relationship

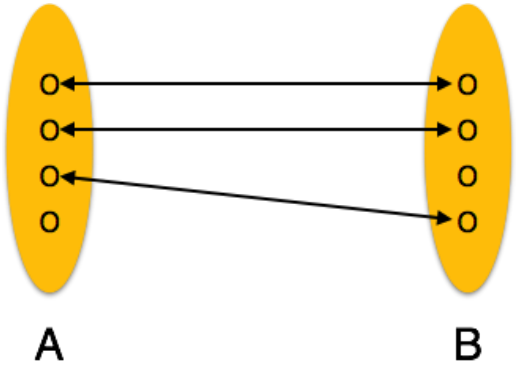
The number of participating entities in a relationship defines the degree of the relationship.

- Binary = degree 2
- Ternary = degree 3
- n-ary = degree

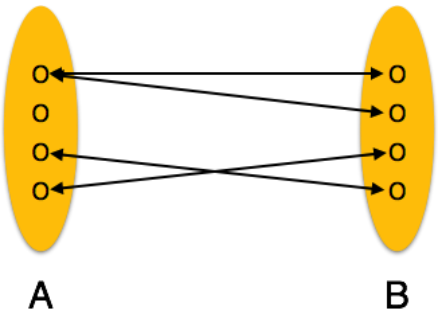
Mapping Cardinalities

Cardinality defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.

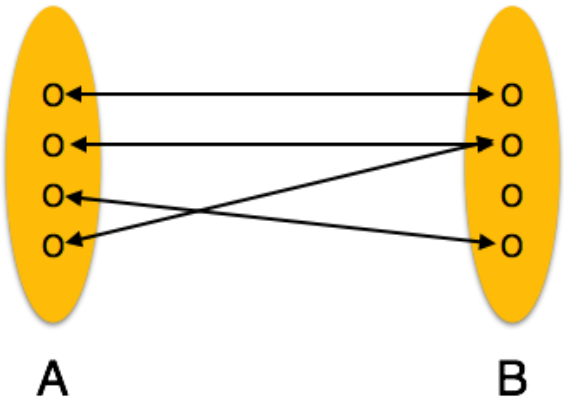
- **One-to-one** − One entity from entity set A can be associated with at most one entity of entity set B and vice versa.



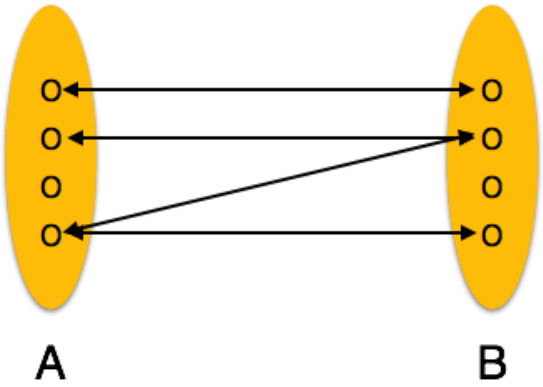
- **One-to-many** − One entity from entity set A can be associated with more than one entities of entity set B however an entity from entity set B, can be associated with at most one entity.



- **Many-to-one** − More than one entities from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A.



- **Many-to-many** − One entity from A can be associated with more than one entity from B and vice versa.



12. Explain different types of attributes with an example.

Types of Attributes

Simple attribute − Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.

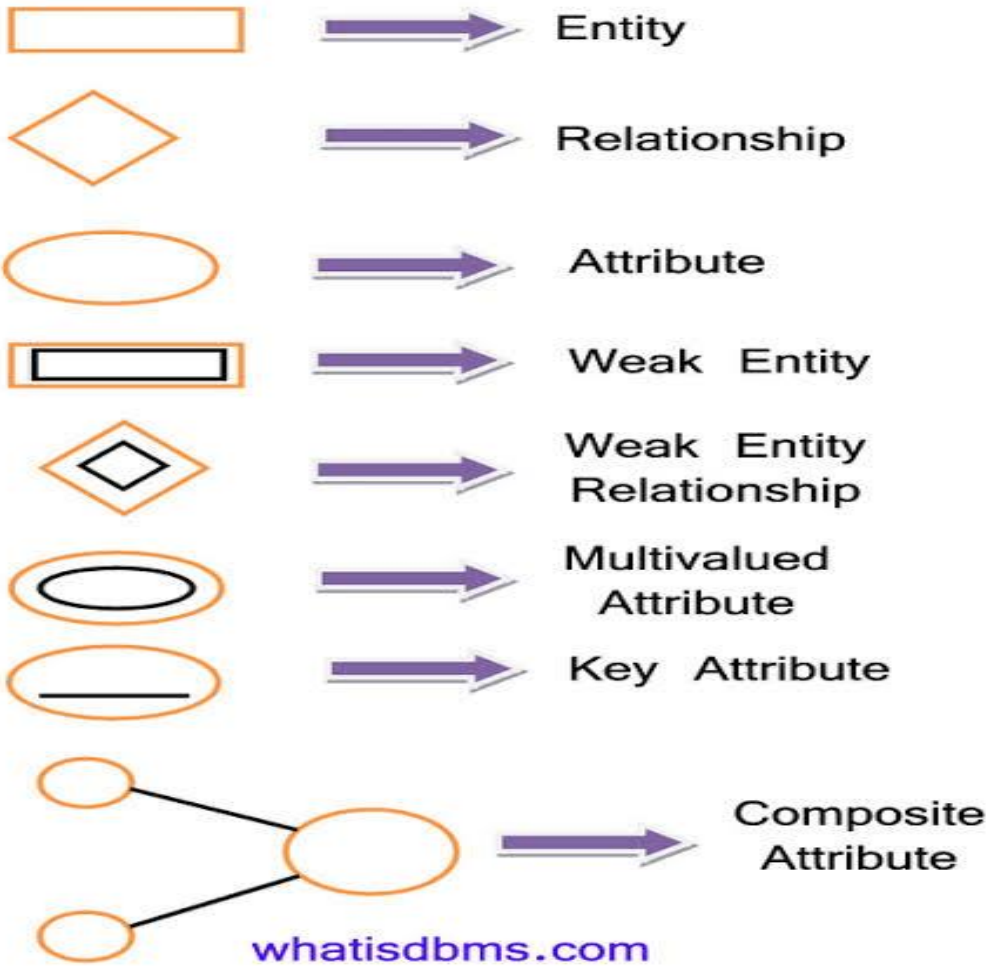
Composite attribute − Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first_name and last_name.

Derived attribute − Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For example, average_salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data_of_birth.

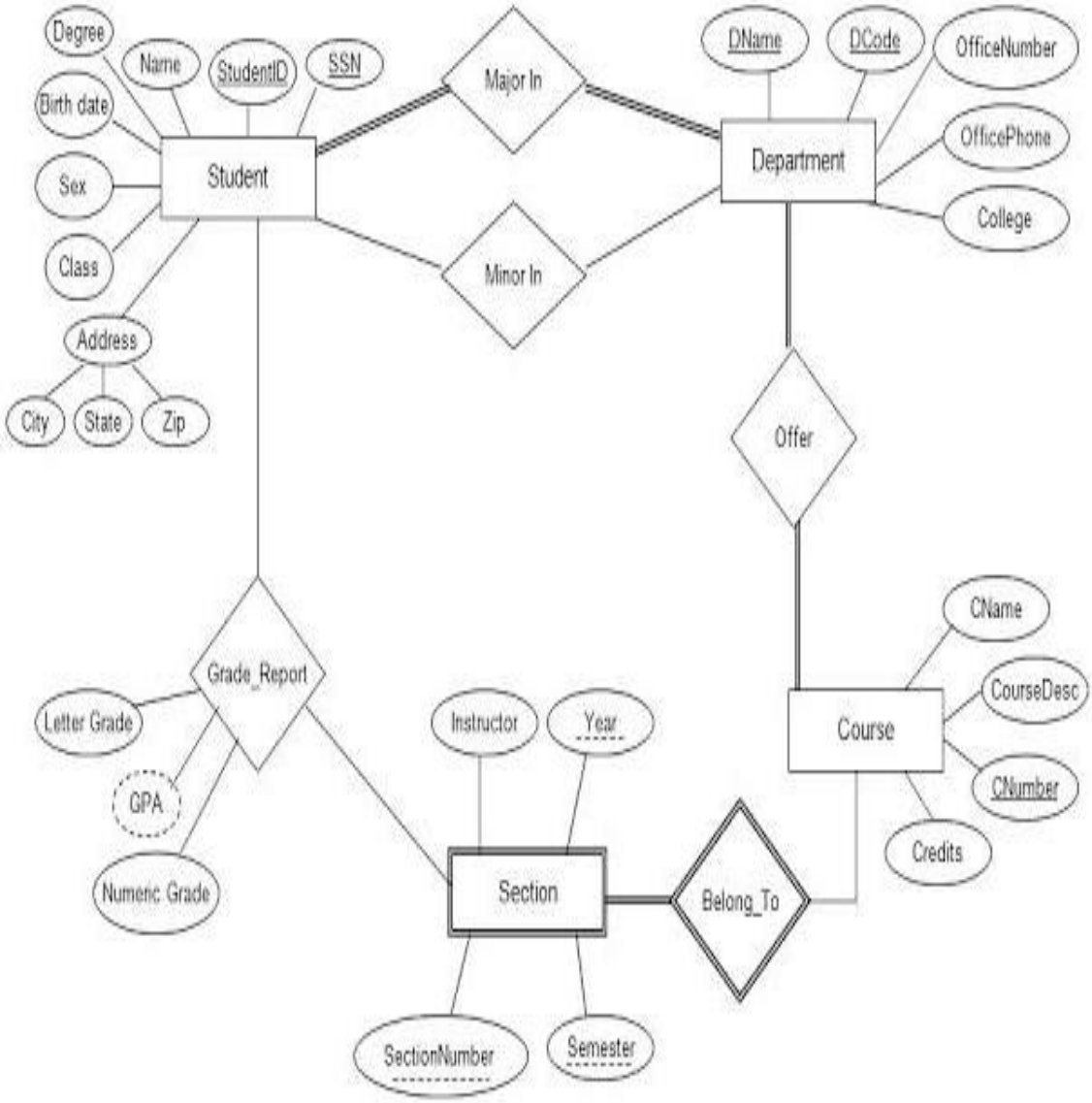
Single-value attribute − Single-value attributes contain single value. For example − Social_Security_Number.

Multi-value attribute − Multi-value attributes may contain more than one values. For example, a person can have more than one phone number, email_address, etc.

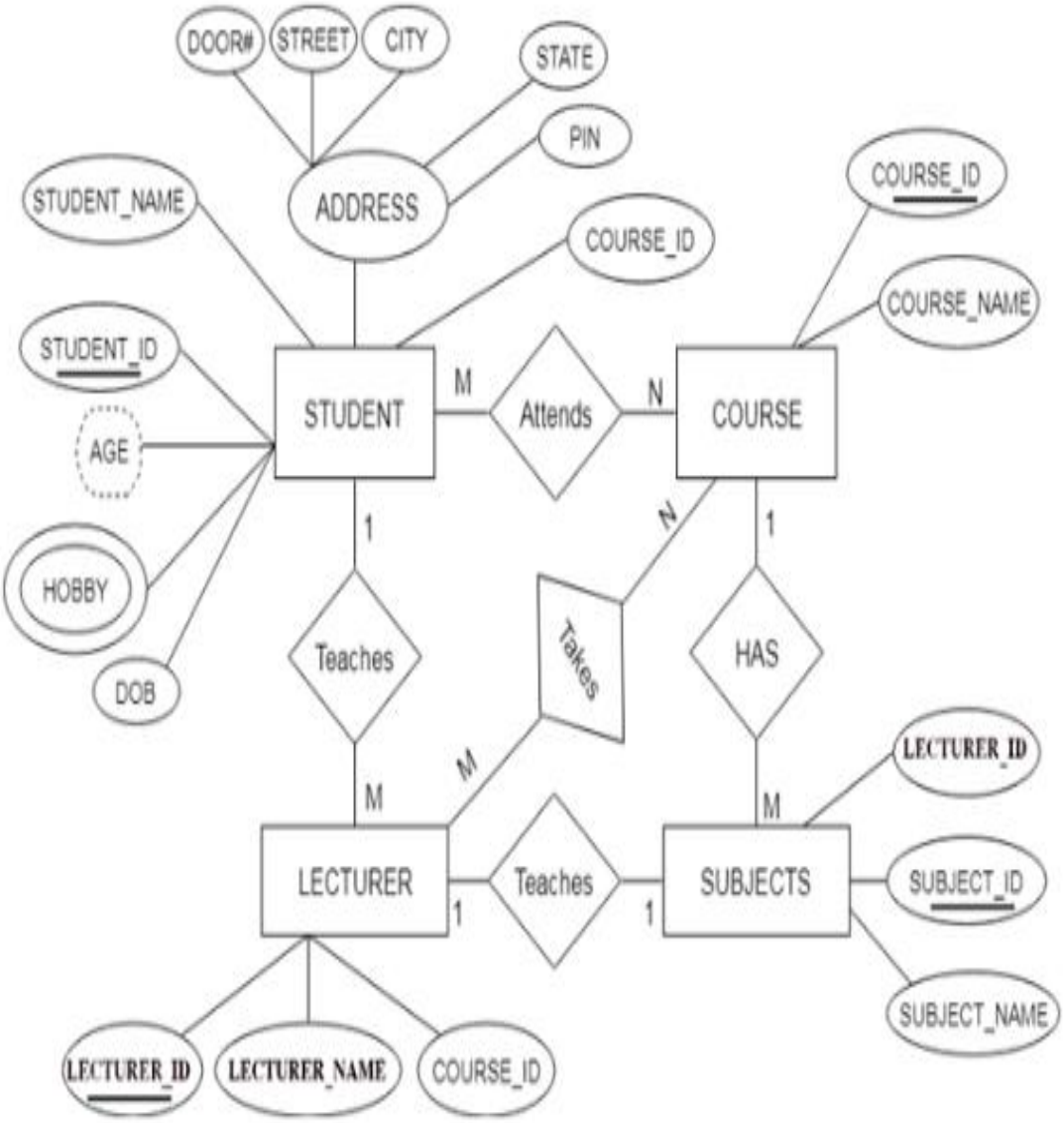
13. Explain the different notations used in ER-diagram.



14.Construct an ER-diagram for university database.



15.Construct an ER-diagram for student database.



Chapter-4 Record Storage and Primary file Organization

1. **What is record and record type?**
   Record is a collection of related data values or items. A collection of filed names and their corresponding data type.

2. **Mention the different record type.**
   Fixed length record type
   Variable length record type.

3. **Define Block?**
   The records of a file must be allocate to disk blocks because a block is the unit of data transfer between disk and memory.

4. **Define File header.**
   File header is a disk addresses of blocks and record format description like field length, order of fields, field type code, separator characters and record type code.

5. **Mention the three types of file organization**
   Heap File
   Sequential File
   Hash File

6. **What is Hashing?**
   Hashing is the process in which we place the each data item at the index the memory location for the purpose of case of usability.

7. **What are the types of Hashing?**
   Internal Hashing
   External Hashing

8. **Expand RAID**

RAID- stands for Redundant Arrays of Inexpensive Disks (or Redundant Arrays of independent disks.

9. **Explain various methods of allocating file blocks on disks.**

The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

The main idea behind these methods is to provide:

- Efficient disk space utilization.
- Fast access to the file blocks.

1. **Contiguous Allocation**

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: b, b+1, b+2,……b+n-1. This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file.

The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

Advantages:

- Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).
- This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.

Disadvantages:

- This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilization.
- Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

2. **Linked List Allocation**

In this scheme, each file is a linked list of disk blocks which need not be contiguous. The disk blocks can be scattered anywhere on the disk.

The directory entry contains a pointer to the starting and the ending file block. Each block contains a pointer to the next block occupied by the file.

Advantages:

- This is very flexible in terms of file size. File size can be increased easily since the system does not have to look for a contiguous chunk of memory.
- This method does not suffer from external fragmentation. This makes it relatively better in terms of memory utilization.

Disadvantages:

- Because the file blocks are distributed randomly on the disk, a large number of seeks are needed to access every block individually. This makes linked allocation slower.
- It does not support random or direct access. We can not directly access the blocks of a file. A block k of a file can be accessed by traversing k blocks sequentially (sequential access) from the starting block of the file via block pointers.
- Pointers required in the linked allocation incur some extra overhead.

3**. Indexed Allocation**

In this scheme, a special block known as the Index block contains the pointers to all the blocks occupied by a file. Each file has its own index block. The ith entry in the index block contains the disk address of the ith file block. The directory entry contains the address of the index block .

Advantages:

- This supports direct access to the blocks occupied by the file and therefore provides fast access to the file blocks.
- It overcomes the problem of external fragmentation.

Disadvantages:

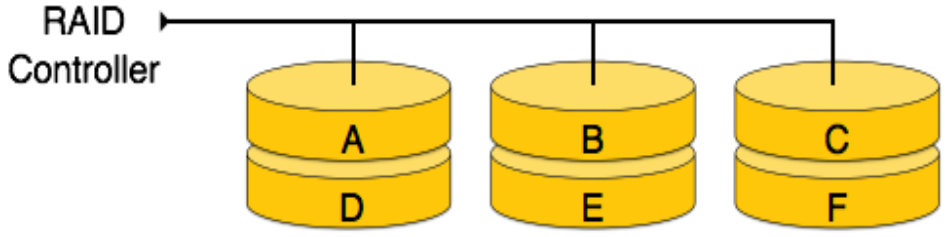
- The pointer overhead for indexed allocation is greater than linked allocation.
- For very small files, say files that expand only 2-3 blocks, the indexed allocation would keep one entire block (index block) for the pointers which is inefficient in terms of memory utilization. However, in linked allocation we lose the space of only 1 pointer per block.

10. Explain briefly RAID technology.

RAID or **R**edundant **A**rray of **I**ndependent **D**isks, is a technology to connect multiple secondary storage devices and use them as a single storage media. RAID consists of an array of disks in which multiple disks are connected together to achieve different goals. RAID levels define the use of disk arrays.

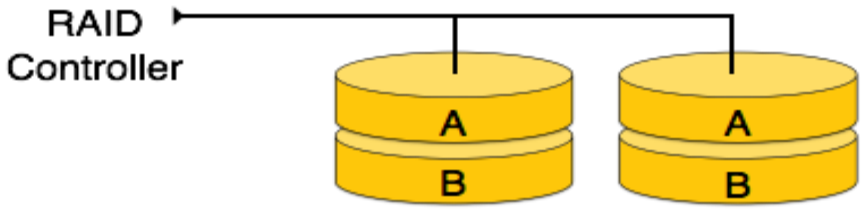
RAID 0

In this level, a striped array of disks is implemented. The data is broken down into blocks and the blocks are distributed among disks. Each disk receives a block of data to write/read in parallel. It enhances the speed and performance of the storage device. There is no parity and backup in Level 0.



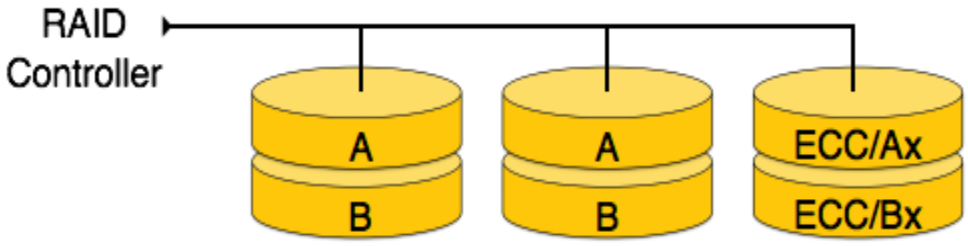
RAID 1

RAID 1 uses mirroring techniques. When data is sent to a RAID controller, it sends a copy of data to all the disks in the array. RAID level 1 is also called **mirroring** and provides 100% redundancy in case of a failure.



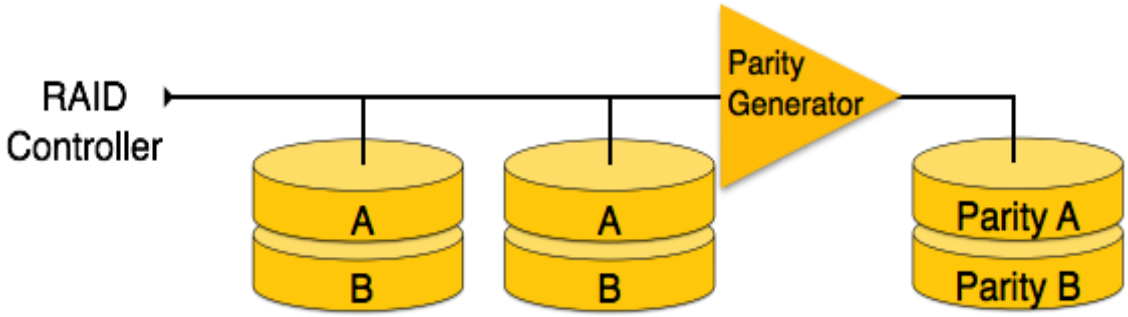
RAID 2

RAID 2 records Error Correction Code using Hamming distance for its data, striped on different disks. Like level 0, each data bit in a word is recorded on a separate disk and ECC codes of the data words are stored on a different set disks. Due to its complex structure and high cost, RAID 2 is not commercially available.



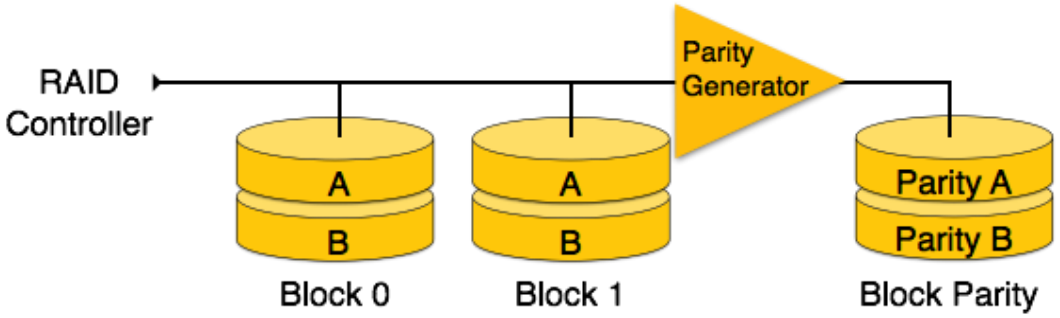
RAID 3

RAID 3 stripes the data onto multiple disks. The parity bit generated for data word is stored on a different disk. This technique makes it to overcome single disk failures.



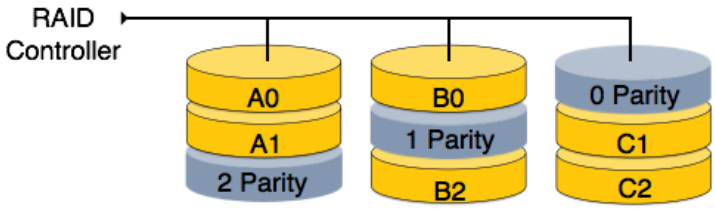
RAID 4

In this level, an entire block of data is written onto data disks and then the parity is generated and stored on a different disk. Note that level 3 uses byte-level striping, whereas level 4 uses block-level striping. Both level 3 and level 4 require at least three disks to implement RAID.



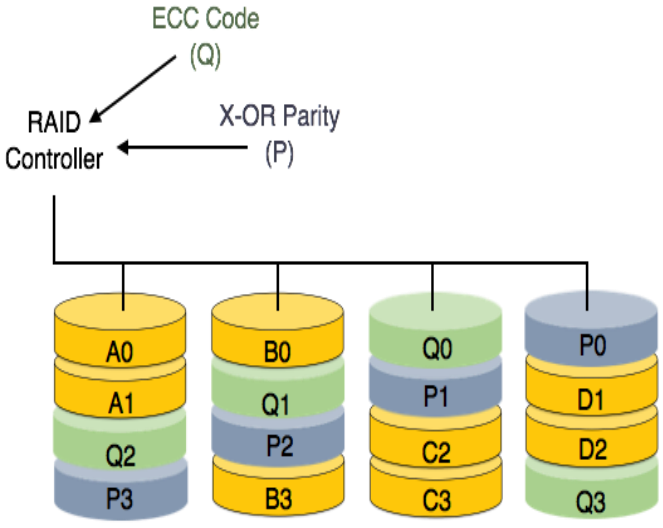
RAID 5

RAID 5 writes whole data blocks onto different disks, but the parity bits generated for data block stripe are distributed among all the data disks rather than storing them on a different dedicated disk.



RAID 6

RAID 6 is an extension of level 5. In this level, two independent parities are generated and stored in distributed fashion among multiple disks. Two parities provide additional fault tolerance. This level requires at least four disk drives to implement RAID.



11.Explain File Organization in DBMS?

What is File Organization?
File Organization refers to the logical relationships among various records that constitute the file, particularly with respect to the means of identification and access to any specific record. In simple terms, Storing the files in certain order is called file Organization. **File Structure** refers to the format of the label and data blocks and of any logical control record.

Types of File Organizations –
Some types of File Organizations are :

- Sequential File Organization
- Heap File Organization
- Hash File Organization
- B+ Tree File Organization
- Clustered File Organization

12.Explain Different types of file organization ?

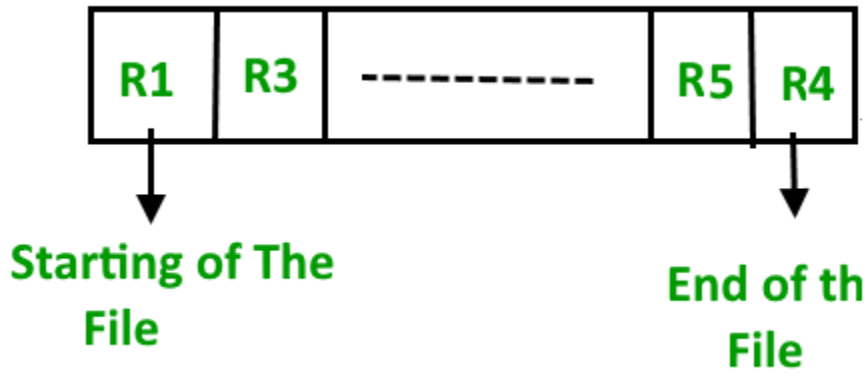
- Some types of File Organizations are :

- Sequential File Organization
- Heap File Organization
- Hash File Organization
- B+ Tree File Organization
- Clustered File Organization

Sequential File Organization –

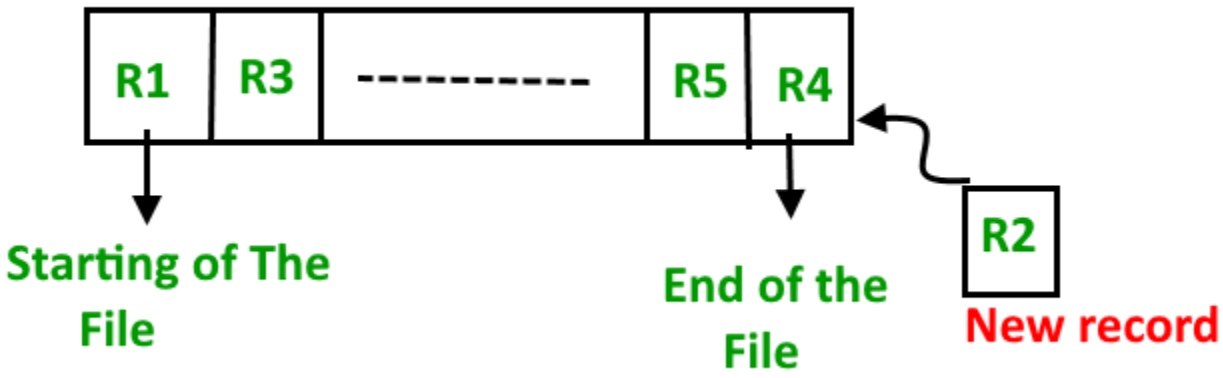
The easiest method for file Organization is Sequential method. In this method the file are stored one after another in a sequential manner. There are two ways to implement this method:

Pile File Method –we store the records in a sequence i.e one after other in the order in which they are inserted into the tables.

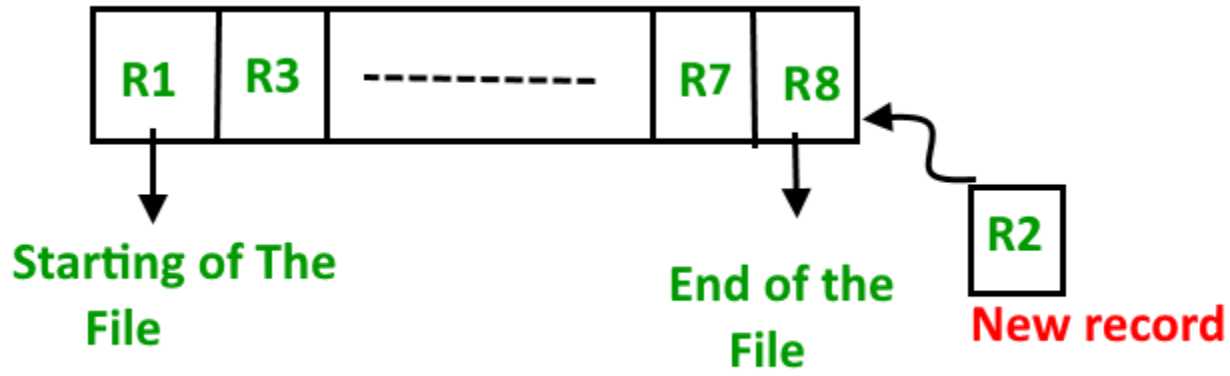


Insertion of new record –

Let the R1, R3 and so on upto R5 and R4 be four records in the sequence. Here, records are nothing but a row in any table. Suppose a new record R2 has to be inserted in the sequence, then it is simply placed at the end of the file.

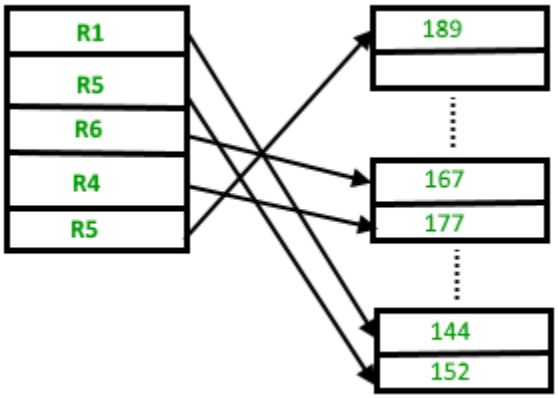


- **Sorted File Method** –In this method, As the name itself suggest whenever a new record has to be inserted, it is always inserted in a sorted (ascending or descending) manner. Sorting of records may be based on any primary key or any other key.



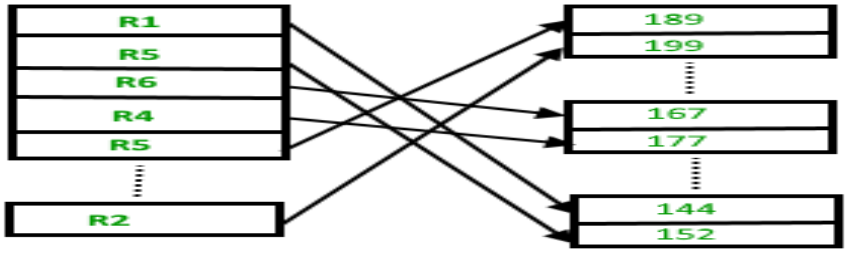
Heap File Organization –

Heap File Organization works with data blocks. In this method records are inserted at the end of the file, into the data blocks. No Sorting or Ordering is required in this method. If a data block is full, the new record is stored in some other block, Here the other data block need not be the very next data block, but it can be any block in the memory. It is the responsibility of DBMS to store and manage the new records.



Insertion of new record –
Suppose we have four records in the heap R1, R5, R6, R4 and R3 and suppose a new record R2 has to be inserted in the heap then, since the last data block i.e data block 3 is full it will be inserted in any of the data blocks selected by the DBMS, lets say data block 1.

If we want to search, delete or update data in heap file Organization the we will traverse the data from the beginning of the file till we get the requested record. Thus if the database is very huge, searching, deleting or updating the record will take a lot of time.



Hashing is an efficient technique to directly search the location of desired data on the disk without using index structure. Data is stored at the data blocks whose address is generated by using hash function. The memory location where these records are stored is called as data block or data bucket.

B+ Tree File Organization –

It uses a tree like structure to store records in File. It uses the concept of Key indexing where the primary key is used to sort the records. For each primary key, an index value is generated and mapped with the record. An index of a record is the address of record in the file.

All the information is stored in leaf node and the intermediate nodes acts as pointer to the leaf nodes. The information in leaf nodes always remain a sorted sequential linked list.

Cluster File Organization –

In cluster file organization, two or more related tables/records are stored within same file known as clusters.

These files will have two or more tables in the same data block and the key attributes which are used to map these table together are stored only once.

Chapter-5 Functional Dependencies and Normalization for Relational Databases

1. **What is key Constraint and Domain Constraint?**
   Keys are the entity set that is used to identify an entity within its entity set uniquely.
   An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.
   Domain constraints can be defined as the definition of a valid set of values for an attribute.
   The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

2. **Define Entity integrity.**
   The entity integrity constraint states that primary key value can't be null.
   This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
   A table can contain a null value other than the primary key field.

3. **Define Referential integrity.**
   A referential integrity constraint is specified between two tables.
   In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.

4. **What is primary key?**
   It is the first key which is used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys as we saw in PERSON table. The key which is most suitable from those lists become a primary key.

In the EMPLOYEE table, ID can be primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License_Number and Passport_Number as primary key since they are also unique.

For each entity, selection of the primary key is based on requirement and developers.

5. **What is candidate key?**

   A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.

   The remaining attributes except for primary key are considered as a candidate key. The candidate keys are as strong as the primary key.

6. **What is Foreign key?**

   Foreign keys are the column of the table which is used to point to the primary key of another table.

   In a company, every employee works in a specific department, and employee and department are two different entities. So we can't store the information of the department in the employee table. That's why we link these two tables through the primary key of one table.

   We add the primary key of the DEPARTMENT table, Department_Id as a new attribute in the EMPLOYEE table.

   Now in the EMPLOYEE table, Department_Id is the foreign key, and both the tables are related.

7. **What is Functional dependency?**

   The functional dependency is a relationship that exists between two attributes. It typically exists between the primary key and non-key attribute within a table.

   $X \rightarrow Y$

   The left side of FD is known as a determinant, the right side of the production is known as a dependent.

8. **What is Transitive dependency?**

   A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies. For e.g.

X -> Z is a transitive dependency if the following three functional dependencies hold true:
X->Y
Y does not ->X
Y->Z
Note: A transitive dependency can only occur in a relation of three of more attributes. This dependency helps us normalizing the database in 3NF (3rd Normal Form).

9. **Define Normalization. Explain First, Second and Third normal forms**
   Normalization is the process of organizing the data in the database.
   Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.
   Normalization divides the larger table into the smaller table and links them using relationship.
   The normal form is used to reduce redundancy from the database table.

First Normal Form (1NF)

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

Example: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385, 9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389, 8589830302	Punjab

The decomposition of the EMPLOYEE table into 1NF has been shown below:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
14	John	9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389	Punjab
12	Sam	8589830302	Punjab

Second Normal Form (2NF)

- In the 2NF, relational must be in 1NF.
- In the second normal form, all non-key attributes are fully functional dependent on the primary key

Example: Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.

TEACHER table

TEACHER_ID	SUBJECT	TEACHER_AGE
25	Chemistry	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

In the given table, non-prime attribute TEACHER_AGE is dependent on TEACHER_ID which is a proper subset of a candidate key. That's why it violates the rule for 2NF.

To convert the given table into 2NF, we decompose it into two tables:

TEACHER_DETAIL table:

TEACHER_ID	TEACHER_AGE
25	30
47	35
83	38

TEACHER_SUBJECT table:

TEACHER_ID	SUBJECT
25	Chemistry
25	Biology
47	English
83	Math
83	Computer

Third Normal Form (3NF)

- A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.
- 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.
- If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in third normal form if it holds atleast one of the following conditions for every non-trivial function dependency X → Y.

1. X is a super key.
2. Y is a prime attribute, i.e., each element of Y is part of some candidate key.

Example:

EMP_ID	EMP_NAME	EMP_ZIP	EMP_STATE	EMP_CITY
222	Harry	201010	UP	Noida
333	Stephan	02228	US	Boston
444	Lan	60007	US	Chicago
555	Katharine	06389	UK	Norwich
666	John	462007	MP	Bhopal

EMPLOYEE_DETAIL table:

Super key in the table above:

1. {EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP }....so on

Candidate key: {EMP_ID}

Non-prime attributes: In the given table, all attributes except EMP_ID are non-prime.

Here, EMP_STATE & EMP_CITY dependent on EMP_ZIP and EMP_ZIP dependent on EMP_ID. The non-prime attributes (EMP_STATE, EMP_CITY) transitively dependent on super key(EMP_ID). It violates the rule of third normal form.

That's why we need to move the EMP_CITY and EMP_STATE to the new

EMP_ID	EMP_NAME	EMP_ZIP
222	Harry	201010
333	Stephan	02228
444	Lan	60007
555	Katharine	06389
666	John	462007

<EMPLOYEE_ZIP> table, with EMP_ZIP as a Primary key.

EMPLOYEE table:

EMP_ZIP	EMP_STATE	EMP_CITY
201010	UP	Noida
02228	US	Boston
60007	US	Chicago
06389	UK	Norwich
462007	MP	Bhopal

EMPLOYEE_ZIP table:

Chapter-6 Relational Data Model and Relational Algebra

1) **What is relational model?**
The relational model represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. In the relational model, data are stored as tables.

2) **What is relational Algebra?**
It collects instances of relations as input and gives occurrences of relations as output. It uses various operation to perform this action. Relational algebra operations are performed recursively on a relation. The output of these operations is a new relation, which might be formed from one or more input relations.

3) **What is Key Constraint?**
A key constraint is a statement that a certain minimal subset of the fields of a relation is a unique identifier for a tuple.

4) **What is super key?**
It is a set of one or more attributes which put together enable us to identify uniquely an entity inthe entity set.

5) **What is candidate key?**
Candidate Key in DBMS. Definition of Candidate Key in DBMS: A super key with no redundant attribute is known as candidate key. Candidate keys are selected from the set of super keys, the only thing we take care while selecting candidate key is that the candidate key should not have any redundant attributes.

6) **Define Domain?**
A domain is a unique set of values permitted for an attribute in a table. For example, a domain of month-of-year can accept January, February….December as possible values, a domain of integers can accept whole numbers that are negative, positive and zero.

7) **Define domain constraint?**
Domain constraints are user defined data type and we can define them like this: Domain Constraint = data type + Constraints (NOT NULL / UNIQUE / PRIMARY KEY / FOREIGN KEY / CHECK / DEFAULT)

8) Define Primary Key?
A primary key is a minimal set of attributes (columns) in a table that uniquely identifies tuples (rows) in that table.

the concept of primary key. In the following table, there are three attributes: Stu_ID, Stu_Name & Stu_Age. Out of these three attributes, one attribute or a set of more than one attributes can be a primary key.

9) Define Foreign Key
Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

10) Define composite key

A key that has more than one attributes is known as composite key. It is also known as compound key.
Any key such as super key, primary key, candidate key etc. can be called composite key if it has more than one attributes.

Long Answers

11) Explain relational Model?
RDBMS stands for relational database management system. A relational model can be represented as a table of rows and columns. A relational database has following A table is a collection of data represented in rows and columns. Each table has a name in database. For example, the following table "STUDENT" stores the information of students in database.

Record or Tuple
Each row of a table is known as record. It is also known as tuple. For example, the following row is a record that we have taken from the above table.

Field or Column name or Attribute

The above table "STUDENT" has four fields (or attributes): Student_Id, Student_Name, Student_Addr & Student_Age.

Domain

A domain is a set of permitted values for an attribute in table. For example, a domain of month-of-year can accept January, February,…December as values, a domain of dates can accept all possible valid dates etc. We specify domain of attribute while creating a table.

An attribute cannot accept values that are outside of their domains. For example, In the above table "STUDENT", the Student_Id field has integer domain so that field cannot accept values that are not integers for example, Student_Id cannot has values like, "First", 10.11 etc.

Instance and Schema

I have already covered instance and schema in a separate guide, you can refer the guide here.

Keys

This is our next topic, I have covered the keys in detail in separate tutorials. You can refer the keys index here.

12) Explain Selection and projection with syntax and example?

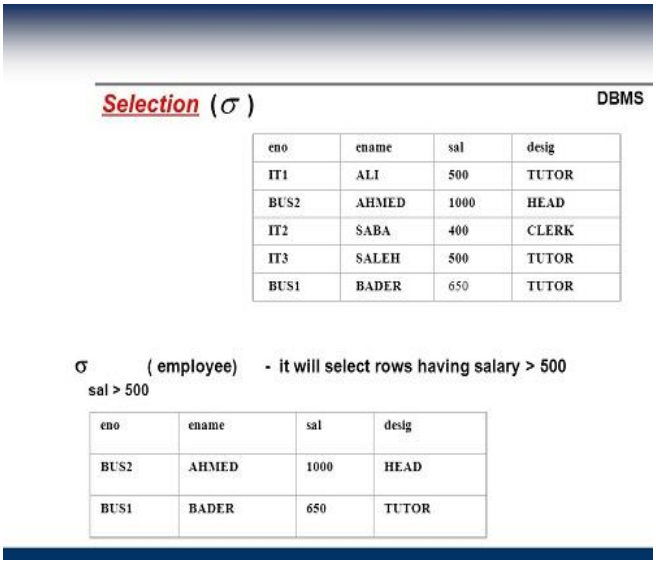
Select Operation : This operation is used to select rows from a table (relation) that specifies a given logic, which is called as a predicate. The predicate is a user defined condition to select rows of user's choice.

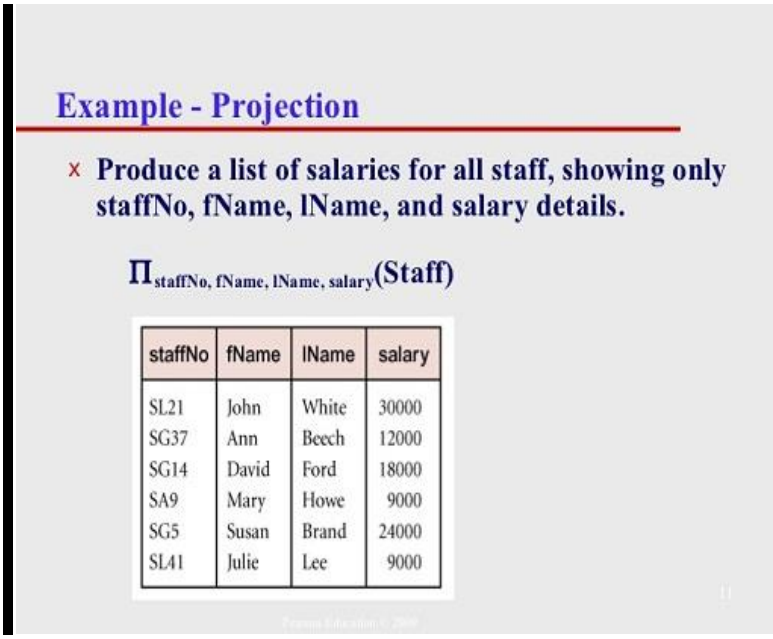
Project Operation : If the user is interested in selecting the values of a few attributes, rather than selection all attributes of the Table (Relation), then one should go for PROJECT Operation.

Example : Selection

Notation uses lower case sigma:

σ-condition(relation)





Both are some of the fundamental operations in relational algebra

Selection is for rows

Projection is for columns

Selection selects rows based on condition(s) specified
Projection selects all rows for the specified columns

13**) Explain set operations in example?**

SET Operations in SQL

SQL supports few Set operations which can be performed on the table data. These are used to get meaningful results from data stored in the table, under different special conditions.

In this tutorial, we will cover 4 different types of SET operations, along with example:

1. UNION
2. UNION ALL
3. INTERSECT
4. MINUS

UNION Operation

UNION is used to combine the results of two or more SELECT statements. However it will eliminate duplicate rows from its resultset. In case of union, number of columns and datatype must be same in both the tables, on which UNION operation is being applied.

UNION ALL

This operation is similar to Union. But it also shows the duplicate rows.

Union All query will be like,

```
SELECT * FROM First

UNION ALL

SELECT * FROM Second;
```

INTERSECT

Intersect operation is used to combine two SELECT statements, but it only retuns the records which are common from both SELECT statements. In case of Intersect the number of columns and datatype must be same.

Intersect query will be,

```
SELECT * FROM First

INTERSECT
```

SELECT * FROM Second;

MINUS

The Minus operation combines results of two SELECT statements and return only those in the final result, which belongs to the first set of the result.

Minus query will be,

SELECT * FROM First

MINUS

SELECT * FROM Second;

14) Explain unary relational operation?

Union Operator (∪)

Union operator is denoted by ∪ symbol and it is used to select all the rows (tuples) from two tables (relations).

Lets discuss union operator a bit more. Lets say we have two relations R1 and R2 both have same columns and we want to select all the tuples(rows) from these relations then we can apply the union operator on these relations.

Note: The rows (tuples) that are present in both the tables will only appear once in the union set. In short you can say that there are no duplicates present after the union operation.

Syntax of Union Operator (∪)

table_name1 ∪ table_name2

Union Operator (∪) Example

Table 1: COURSE

Course_Id Student_Name Student_Id
--------- ------------ ----------

C101 Aditya S901
C104 Aditya S901
C106 Steve S911
C109 Paul S921
C115 Lucy S931

Table 2: STUDENT

Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

Query:

∏ Student_Name (COURSE) ∪ ∏ Student_Name (STUDENT)

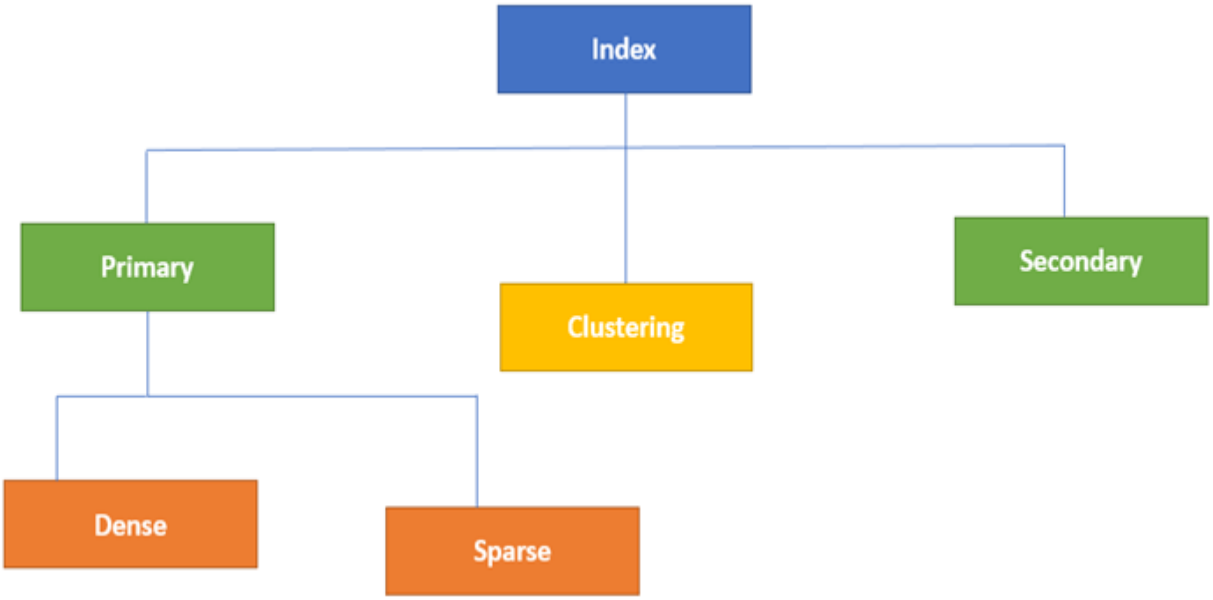
Output:

Student_Name
Aditya
Carl
Paul
Lucy
Rick
Steve

Note: As you can see there are no duplicate names present in the output even though we had few common names in both the tables, also in the COURSE table we had the duplicate name itself.

14.What are the types of Indexing in DBMS?

Indexing is a data structure technique which allows you to quickly retrieve records from a database file. An Index is a small table having only two columns. The first column comprises a copy of the primary or candidate key of a table. Its second column contains a set of pointers for holding the address of the disk block where that specific key value stored.



Two main types of indexing methods are:

- Primary Indexing
- Secondary Indexing

Primary Index in DBMS

Primary Index is an ordered file which is fixed length size with two fields. The first field is the same a primary key and second, filed is pointed to that specific data block. In the primary Index, there is always one to one relationship between the entries in the index table.

The primary Indexing in DBMS is also further divided into two types.

- Dense Index
- Sparse Index
- **Dense Index**

- In a dense index, a record is created for every search key valued in the database. This helps you to search faster but needs more space to store index records. In this Indexing, method records contain search key value and points to the real record on the disk.
- **Sparse Index**
- It is an index record that appears for only some of the values in the file. Sparse Index helps you to resolve the issues of dense Indexing in DBMS.

Secondary Index in DBMS

- The secondary Index in DBMS can be generated by a field which has a unique value for each record, and it should be a candidate key. It is also known as a non-clustering index.

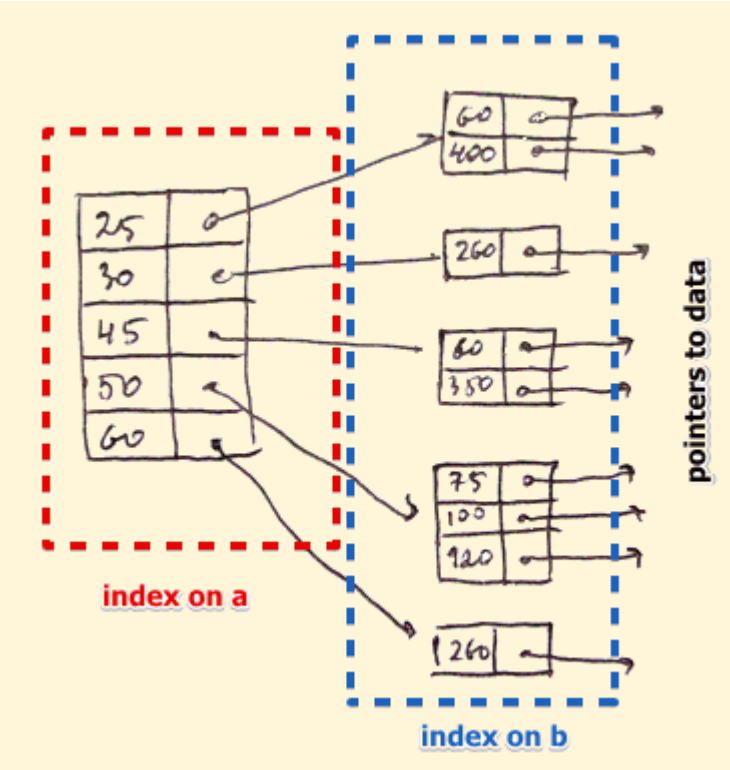
Clustering Index in DBMS

In a clustered index, records themselves are stored in the Index and not pointers. Sometimes the Index is created on non-primary key columns which might not be unique for each record. In such a situation, you can group two or more columns to get the unique values and create an index which is called clustered Index. This also helps you to identify the record faster.

15.Explain Indexes on multiple keys?

Multilevel Indexing in Database is created when a primary index does not fit in memory. In this type of indexing method, you can reduce the number of disk

acesses to short any record and kept on a disk as a sequential file and create a



sparse base on that file.

Chapter-7 Relational Database Language

1) Define RDBMS?

It is in which data is stored in the form of tables and the relationship among the data is also stored in the form of tables.

2) What is SQL?

SQL is a structured query language for updating , deleting and requesting information from db.

3) What are the advantages of SQL?

- Support SQL data manipulation.
- Provide facilities like conditional checking, branching and looping.

4) What are the different categories of SQL statements?

- DDL (Data Definition Language
- DML (Data manipulation Language)
- DCl (data Control language)
- TCL (Transaction Control Language)

5) What is the purpose of ALTER statement?

The **ALTER** TABLE **statement** is used to add, delete, or modify columns in an existing table. The **ALTER** TABLE **statement** is also used to add and drop various constraints on an existing table.

6) **What is default Constraint?**

The **DEFAULT constraint** inserts a **default** value into a column of a table when you insert a new row into the table without specifying the value for the column. Creating **SQL DEFAULT constraint**. There are two ways to create **DEFAULT constraints** for columns: Use CREATE TABLE **statement** if the table is new; Use ALTER TABLE **statement** for an existing table.

7) **What are the different types of Joins?**

- (INNER) JOIN: Returns records that have matching values in both tables
- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
- RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
- FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

8) **Write the syntax for drop column?**

Drop table <table name>.

9) **Write general The basic syntax of the ORDER BY clause?**

SELECT column-list
FROM table_name
[WHERE condition]
[ORDER BY column1, column2, .. columnN] [ASC | DESC];

10) Define View?

In SQL, a **view** is a virtual table based on the result-set of an SQL statement. A **view** contains rows and columns, just like a real table. The fields in a **view** are fields from one or more real tables in the database.

11) **Explain order by, group by and having clause with syntax and example?**

GROUP BY

The GROUP BY clause is a SQL command that is used to **group rows that have the same values**.

The GROUP BY clause is used in the SELECT statement .

Optionally it is used in conjunction with aggregate functions to produce summary reports from the database.

```
SELECT statements... GROUP BY column_name1[,column_name2,...] [HAVING condition];
```

"SELECT statements..." is the standard SQL SELECT command query.

"**GROUP BY** *column_name1*" is the clause that performs the grouping based on column_name1.

"[,column_name2,...]" is optional; represents other column names when the grouping is done on more than one column.

"[HAVING condition]" is optional; it is used to restrict the rows affected by the GROUP BY clause. It is similar to the WHERE

Grouping using multiple columns

```
SELECT `category_id`,`year_released` FROM `movies` ;
```

The GROUP BY Clause is used to group rows with same values .

The GROUP BY Clause is used together with the SQL SELECT statement.

The SELECT statement used in the GROUP BY clause can only be used contain column names, aggregate functions, constants and expressions.

The HAVING clause is used to restrict the results returned by the GROUP BY clause.

HAVING clause

The HAVING clause is an optional clause of the SELECT statement. It is used to filter groups of rows returned by the GROUP BY clause. This is why the HAVING clause is usually used with the GROUP BY clause.

The following illustrates the syntax of the Oracle HAVING clause:

SELECT column_list FROM T GROUP BY c1 HAVING group_condition;

In this statement, the HAVING clause appears immediately after the GROUP BY clause.

If you use the HAVING clause without the GROUP BY clause, the HAVING clause works like the WHERE clause.

HAVING clause filters groups of rows while the WHERE clause filters rows. This is a main difference between the HAVING and WHERE clauses.

ORDER BY

The SQL **ORDER BY** clause is used to sort the data in ascending or descending order, based on one or more columns. Some databases sort the query results in an ascending order by default.

The basic syntax of the ORDER BY clause is as follows −

SELECT column-list

```
FROM table_name
[WHERE condition]
[ORDER BY column1, column2, .. columnN] [ASC | DESC];
```

You can use more than one column in the ORDER BY clause. Make sure whatever column you are using to sort that column should be in the column-list.

```
SQL> SELECT * FROM CUSTOMERS
ORDER BY NAME, SALARY;
SELECT * FROM CUSTOMERS
ORDER BY NAME DESC;
```

12. Describe Indexes in SQL?

- Indexes are **special lookup tables** that the database search engine can use to speed up data retrieval. Simply put, an index is a pointer to data in a table. An index in a database is very similar to an index in the back of a book.
- For example, if you want to reference all pages in a book that discusses a certain topic, you first refer to the index, which lists all the topics alphabetically and are then referred to one or more specific page numbers.
- An index helps to speed up **SELECT** queries and **WHERE** clauses, but it slows down data input, with the **UPDATE** and the **INSERT** statements. Indexes can be created or dropped with no effect on the data.
- Creating an index involves the **CREATE INDEX** statement, which allows you to name the index, to specify the table and which column or columns to index, and to indicate whether the index is in an ascending or descending order.

- Indexes can also be unique, like the **UNIQUE** constraint, in that the index prevents duplicate entries in the column or combination of columns on which there is an index.

The CREATE INDEX Command

- The basic syntax of a **CREATE INDEX** is as follows.
- CREATE INDEX index_name ON table_name;

Single-Column Indexes

- A single-column index is created based on only one table column. The basic syntax is as follows.
- CREATE INDEX index_name
- ON table_name (column_name);

Unique Indexes

- Unique indexes are used not only for performance, but also for data integrity. A unique index does not allow any duplicate values to be inserted into the table. The basic syntax is as follows.
- CREATE UNIQUE INDEX index_name
- on table_name (column_name);

Composite Indexes

- A composite index is an index on two or more columns of a table. Its basic syntax is as follows.
- CREATE INDEX index_name

- on table_name (column1, column2);
- Whether to create a single-column index or a composite index, take into consideration the column(s) that you may use very frequently in a query's WHERE clause as filter conditions.
- Should there be only one column used, a single-column index should be the choice. Should there be two or more columns that are frequently used in the WHERE clause as filters, the composite index would be the best choice.

Implicit Indexes

- Implicit indexes are indexes that are automatically created by the database server when an object is created. Indexes are automatically created for primary key constraints and unique constraints.

The DROP INDEX Command

- An index can be dropped using SQL **DROP** command. Care should be taken when dropping an index because the performance may either slow down or improve.
- The basic syntax is as follows −
- DROP INDEX index_name;
- You can check the INDEX Constraint chapter to see some actual examples on Indexes.

13. Write short note on outer joins?

There are three kinds of OUTER JOIN: left outer join, right outer join and full outer join. Let's examine each kind of join in more detail.

SQL OUTER JOIN – left outer join

SQL left outer join is also known as SQL left join. Suppose, we want to join two tables: A and B. SQL left outer join returns all rows in the left table (A) and all the matching rows found in the right table (B). It means the result of the SQL left join always contains the rows in the left table.

The following illustrate SQL left outer syntax of joining 2 tables: table_A and table_B:

SELECT column1, column2... FROM table_A LEFT JOIN table_B ON join_condition WHERE row_condition

SQL OUTER JOIN – left outer join example

The following query selects all customers and their orders:

SELECT c.customerid, c.companyName, ordered FROM customers c LEFT JOIN orders o ON o.customerid = c.customerid ORDER BY orderid

All rows in the *c*ustomers table are listed. In case, there is no matching row in the orders table found for the row in the customers table, the orderid column in the orders table is populated with NULL values.
We can use Venn diagram to visualize how SQL LEFT OUTER JOIN works.

SQL OUTER JOIN – right outer join

SQL right outer join returns all rows in the right table and all the matching rows found in the left table. The syntax of the SQL right outer join is as follows:

SELECT column1, column2... FROM table_A RIGHT JOIN table_B ON join_condition WHERE row_condition

SQL right outer join is also known as SQL right join.

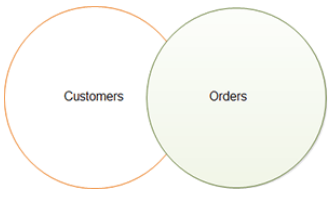
SQL OUTER JOIN – right outer join example

The following example demonstrates the SQL right outer join:

SELECT c.customerid, c.companyName, ordered FROM customers c RIGHT JOIN orders o ON o.customerid = c.customeridORDER BY orderid

The query returns all rows in the *orders* table and all matching rows found in the *customers* table.

The following Venn diagram illustrates how the SQL right outer join works:



SQL OUTER JOIN – full outer join

The syntax of the SQL full outer join is as follows:

SELECT column1, column2... FROM table_A FULL OUTER JOIN table_B ON join_condition WHERE row_condition

SQL full outer join returns:

- all rows in the left table table_A.
- all rows in the right table table_B.
- and all matching rows in both tables.

Some database management systems do not support SQL full outer join syntax e.g., MySQL. Because SQL full outer join returns a result set that is a combined result of both SQL left join and SQL right join. Therefore you can easily emulate the SQL full outer join using SQL left join and SQL right join with UNION operator as follows:

SELECT column1, column2... FROM table_A LEFT JOIN table_B ON join_condition UNION SELECT column1, column2... FROM table_A RIGHT JOIN table_B ON join_condition

Chapter-8 PL/SQL

1) Define PL/SQL?

PL/SQL is an extension of **Structured Query Language** (SQL) that is used in Oracle. Unlike SQL, PL/SQL allows the programmer to write code in a procedural format. Full form of PL/SQL is **"Procedural Language extensions to SQL"**. It combines the data manipulation power of SQL with the processing power of procedural language

2) Mention the two groups of PL/SQL?

- Named Block
- Anonymous block.

3) Define Procedure?

It is a **subprogram** is a program unit/module that performs a particular task. These subprograms are combined to form larger programs. This is basically called the 'Modular design'. A subprogram can be invoked by another subprogram or program which is called the **calling program**.

4) Define Exception handling in PL/SQL?

Exceptions in PL/SQL. An exception is an error condition during a program execution. PL/SQL supports programmers to catch such conditions using **EXCEPTION** block in the program and an appropriate action is taken against the error condition. There are two types of exceptions −

- System-defined exceptions
- User-defined exceptions

5) Define Database Triggers?

Triggers in PL/SQL. Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events −

A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)

A **database definition (DDL)** statement (CREATE, ALTER, or DROP).

A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

6) Define Cursor?

cursors in PL/SQL. Oracle creates a memory area, known as the context area, for processing an SQL statement, which contains all the information needed for processing the statement; for example, the number of rows processed, etc.

A **cursor** is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the **active set**.

You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors −

- Implicit cursors
- Explicit cursors

7) **Define Packages?**

Packages are schema objects that groups logically related PL/SQL types, variables, and subprograms.

A package will have two mandatory parts −

- Package specification
- Package body or definition

8) Explain Implicit cursor and explicit cursor?

cursors in PL/SQL. Oracle creates a memory area, known as the context area, for processing an SQL statement, which contains all the information needed for processing the statement; for example, the number of rows processed, etc.

A **cursor** is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the **active set**.

You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors −

- Implicit cursors
- Explicit cursors

Implicit Cursors

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the information in it.

Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted. For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

Explicit Cursors

Explicit cursors are programmer-defined cursors for gaining more control over the **context area**. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row.

The syntax for creating an explicit cursor is −

CURSOR cursor_name IS select_statement;

Working with an explicit cursor includes the following steps −

- Declaring the cursor for initializing the memory
- Opening the cursor for allocating the memory
- Fetching the cursor for retrieving the data
- Closing the cursor to release the allocated memory

Declaring the Cursor

Declaring the cursor defines the cursor with a name and the associated SELECT statement. For example −

```
CURSOR c_customers IS
 SELECT id, name, address FROM customers;
```

Opening the Cursor

Opening the cursor allocates the memory for the cursor and makes it ready for fetching the rows returned by the SQL statement into it. For example, we will open the above defined cursor as follows −

```
OPEN c_customers;
```

Fetching the Cursor

Fetching the cursor involves accessing one row at a time. For example, we will fetch rows from the above-opened cursor as follows −

```
FETCH c_customers INTO c_id, c_name, c_addr;
```

Closing the Cursor

Closing the cursor means releasing the allocated memory. For example, we will close the above-opened cursor as follows −

```
CLOSE c_customers;
```

9) Explain Triggers?

Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events −

- A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)
- A **database definition (DDL)** statement (CREATE, ALTER, or DROP).

- A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

Benefits of Triggers

Triggers can be written for the following purposes −

- Generating some derived column values automatically
- Enforcing referential integrity
- Event logging and storing information on table access
- Auditing
- Synchronous replication of tables
- Imposing security authorizations
- Preventing invalid transactions

Creating Triggers

The syntax for creating a trigger is –

```
CREATE [OR REPLACE ] TRIGGER trigger_name  {BEFORE | AFTER
|INSTEAD OF }  {INSERT [OR] | UPDATE [OR] | DELETE}  [OF col_name]
ON table_name  [REFERENCING OLD AS o NEW AS n]  [FOR EACH ROW]
WHEN (condition)
 DECLARE
   Declaration-statements
BEGIN
   Executable-statements
EXCEPTION
   Exception-handling-statements
END;
```

Where,

- CREATE [OR REPLACE] TRIGGER trigger_name − Creates or replaces an existing trigger with the *trigger_name*.
- {BEFORE | AFTER | INSTEAD OF} − This specifies when the trigger will be executed. The INSTEAD OF clause is used for creating trigger on a view.
- {INSERT [OR] | UPDATE [OR] | DELETE} − This specifies the DML operation.
- [OF col_name] − This specifies the column name that will be updated.
- [ON table_name] − This specifies the name of the table associated with the trigger.
- [REFERENCING OLD AS o NEW AS n] − This allows you to refer new and old values for various DML statements, such as INSERT, UPDATE, and DELETE.
- [FOR EACH ROW] − This specifies a row-level trigger, i.e., the trigger will be executed for each row being affected. Otherwise the trigger will execute just once when the SQL statement is executed, which is called a table level trigger.
- WHEN (condition) − This provides a condition for rows for which the trigger would fire. This clause is valid only for row-level triggers.

10) Explain PL/SQL Packages?

Packages are schema objects that groups logically related PL/SQL types, variables, and subprograms.

A package will have two mandatory parts −

- Package specification
- Package body or definition

Package Specification

The specification is the interface to the package. It just **DECLARES** the types, variables, constants, exceptions, cursors, and subprograms that can be referenced

from outside the package. In other words, it contains all information about the content of the package, but excludes the code for the subprograms.

All objects placed in the specification are called **public** objects. Any subprogram not in the package specification but coded in the package body is called a **private** object.

The following code snippet shows a package specification having a single procedure. You can have many global variables defined and multiple procedures or functions inside a package.

```
CREATE PACKAGE cust_sal AS  PROCEDURE find_sal(c_id
customers.id%type);
END cust_sal;
/
```

When the above code is executed at the SQL prompt, it produces the following result −

Package created.

Package Body

The package body has the codes for various methods declared in the package specification and other private declarations, which are hidden from the code outside the package.

The **CREATE PACKAGE BODY** Statement is used for creating the package body. The following code snippet shows the package body declaration for the ***cust_sal*** package create

```
CREATE OR REPLACE PACKAGE BODY cust_sal AS     PROCEDURE
find_sal(c_id customers.id%TYPE) IS   c_sal customers.salary%TYPE;
  BEGIN
    SELECT salary INTO c_sal
    FROM customers
    WHERE id = c_id;
    dbms_output.put_line('Salary: '|| c_sal);
```

```
  END find_sal;
 END cust_sal;
 /
```

When the above code is executed at the SQL prompt, it produces the following result −

Package body created.

Using the Package Elements

The package elements (variables, procedures or functions) are accessed with the following syntax −

package_name.element_name;

Consider, we already have created the above package in our database schema, the following program uses the ***find_sal*** method of the ***cust_sal*** package −

```
DECLARE
  code customers.id%type := &cc_id;
BEGIN
  cust_sal.find_sal(code);
END;
/
```

When the above code is executed at the SQL prompt, it prompts to enter the customer ID

The Package Specification

```
CREATE OR REPLACE PACKAGE c_package AS
  -- Adds a customer
  PROCEDURE addCustomer(c_id   customers.id%type,
  c_name  customerS.No.ame%type,
  c_age  customers.age%type,
  c_addr customers.address%type,
  c_sal  customers.salary%type);

  -- Removes a customer
```

```
  PROCEDURE delCustomer(c_id  customers.id%TYPE);
  --Lists all customers
  PROCEDURE listCustomer;
END c_package;
/
```

When the above code is executed at the SQL prompt, it creates the above package and displays the following result −

Package created.

Creating the Package Body

```
CREATE OR REPLACE PACKAGE BODY c_package AS
  PROCEDURE addCustomer(c_id  customers.id%type,
    c_name customerS.No.ame%type,
    c_age  customers.age%type,
    c_addr  customers.address%type,
    c_sal   customers.salary%type)
  IS
  BEGIN
    INSERT INTO customers (id,name,age,address,salary)
      VALUES(c_id, c_name, c_age, c_addr, c_sal);
  END addCustomer;

  PROCEDURE delCustomer(c_id   customers.id%type) IS
  BEGIN
    DELETE FROM customers
    WHERE id = c_id;
  END delCustomer;

  PROCEDURE listCustomer IS
  CURSOR c_customers is
    SELECT  name FROM customers;
  TYPE c_list is TABLE OF customers.Name%type;
  name_list c_list := c_list();
```

```
  counter integer :=0;
  BEGIN
    FOR n IN c_customers LOOP
    counter := counter +1;
    name_list.extend;
    name_list(counter) := n.name;
    dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
    END LOOP;
  END listCustomer;

END c_package;
/
```

The above example makes use of the **nested table**. We will discuss the concept of nested table in the next chapter.

When the above code is executed at the SQL prompt, it produces the following result −

Package body created.

11) Describe PL/SQL features and its advantages?

Features of PL/SQL

- PL/SQL is tightly integrated with SQL.
- It offers extensive error checking.
- It offers numerous data types.
- It offers a variety of programming structures.
- It supports structured programming through functions and procedures.
- It supports object-oriented programming.
- It supports the development of web applications and server pages.

Advantages of PL/SQL

- SQL is the standard database language and PL/SQL is strongly integrated with SQL. PL/SQL supports both static and dynamic SQL. Static SQL

supports DML operations and transaction control from PL/SQL block. In Dynamic SQL, SQL allows embedding DDL statements in PL/SQL blocks.

- PL/SQL allows sending an entire block of statements to the database at one time. This reduces network traffic and provides high performance for the applications.
- PL/SQL gives high productivity to programmers as it can query, transform, and update data in a database.
- PL/SQL saves time on design and debugging by strong features, such as exception handling, encapsulation, data hiding, and object-oriented data types.
- Applications written in PL/SQL are fully portable.
- PL/SQL provides high security level.
- PL/SQL provides access to predefined SQL packages.
- PL/SQL provides support for Object-Oriented Programming.
- PL/SQL provides support for developing Web Applications and Server Pages.

Chapter-9 Transaction Processing Concepts & Concurrency Control Techniques

1. What are the db access operations of a transaction?

A transaction is the logic unit of execution in an information system. A transaction of operations that must be executed as a whole taking a consistent db state into another consistent db state.

2. What is system log?

A system log is a file containing events that are updated by the operating system components.It may contain information such as device drivers,events,operations or even device changes.

3. Define transaction processing system?

A Transaction Processing System (TPS) is a type of information system that collects, stores, modifies and retrieves the data transactions of an enterprise. e.g.; airline reservation systems, electronic transfer of funds, bank account processing systems.

4. What are the different operations of Transactions?

1) Begin _ Transaction
2) Read or Write
3) End_ Transaction
4) Commit Transaction
5) Rollback

5. Describe the properties of transaction?

A transaction is a very small unit of a program and it may contain several lowlevel tasks. A transaction in a database system must maintain **A**tomicity, **C**onsistency, **I**solation, and **D**urability − commonly known as ACID properties − in order to ensure accuracy, completeness, and data integrity.

6. What is concurrency?

Concurrency is the procedure in DBMS for managing simultaneous operations without conflicting with each another. Concurrent access is quite easy if all users are just reading data

7. What is concurrency control?

Concurrency Control deals with **interleaved execution** of more than one transaction.

8. Define Lock in DBMS?

A **lock** is a data variable which is associated with a data item. This **lock** signifies that operations that can be performed on the data item. **Locks** help synchronize access to the database items by concurrent transactions.

9. Define Time Stamp?

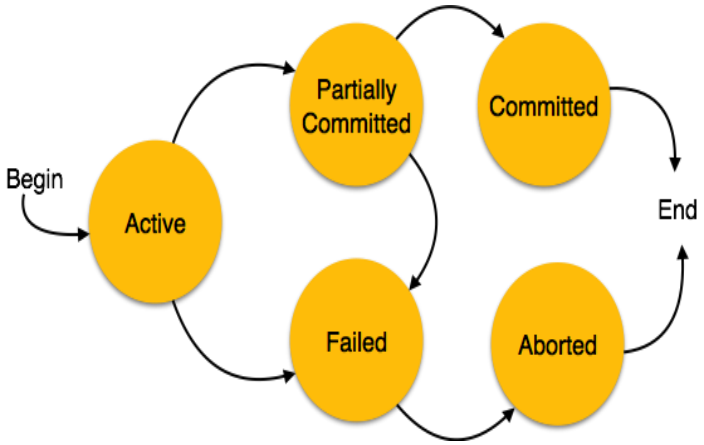
The Timestamp Ordering Protocol is used to order the transactions based on their Timestamps. The order of transaction is nothing but the ascending order of the transaction creation.

The priority of the older transaction is higher that's why it executes first. To determine the timestamp of the transaction, this protocol uses system time or logical counter.

10. Explain different states of transaction in DBMS?

States of Transactions

A transaction in a database can be in one of the following states −



- **Active** − In this state, the transaction is being executed. This is the initial state of every transaction.
- **Partially Committed** − When a transaction executes its final operation, it is said to be in a partially committed state.
- **Failed** − A transaction is said to be in a failed state if any of the checks made by the database recovery system fails. A failed transaction can no longer proceed further.
- **Aborted** − If any of the checks fails and the transaction has reached a failed state, then the recovery manager rolls back all its write operations on the database to bring the database back to its original state where it was prior to the execution of the transaction. Transactions in this state are called aborted. The database recovery module can select one of the two operations after a transaction aborts −

- Re-start the transaction
- Kill the transaction
- **Committed** − If a transaction executes all its operations successfully, it is said to be committed. All its effects are now permanently established on the database system.

11. Define Lock and various types of Locks?

- Here are two types of Locks available **Shared S(a)** and **Exclusive X(a)**. Implementing this lock system without any restrictions gives us the Simple Lock based protocol (or *Binary Locking*), but it has its own disadvantages, they **does not guarantee Serializability**. Schedules may follow the preceding rules but a non-serializable schedule may result.
- To guarantee serializablity, we must follow some additional protocol *concerning the positioning of locking and unlocking operations* in every transaction. This is where the concept of Two Phase Locking(2-PL) comes in the picture, 2-PL ensures serializablity.

Two Phase Locking –

A transaction is said to follow Two Phase Locking protocol if Locking and Unlocking can be done in two phases.

- **Growing Phase:** New locks on data items may be acquired but none can be released.
- **Shrinking Phase:** Existing locks may be released but no new locks can be acquired.

If lock conversion is allowed, then upgrading of lock(from S(a) to X(a)) is allowed in Growing Phase and downgrading of lock (from X(a) to S(a)) must be done in shrinking phase.

Transaction T_1:

- Growing Phase is from steps 1-3.
- Shrinking Phase is from steps 5-7.
- Lock Point at 3

Transaction T_2:

- Growing Phase is from steps 2-6.
- Shrinking Phase is from steps 8-9.
- Lock Point at 6

12. Explain Concurrency control techniques in indexes?

Concurrency Control Techniques

i) enforce isolation among transactions.

(ii) preserve database consistency through consistency preserving execution of transactions.

(iii) resolve read-write and write-read conflicts.

Various concurrency control techniques are:

1. Two-phase locking Protocol

2. Time stamp ordering Protocol

3. Multi version concurrency control

4. Validation concurrency control

1. Two-Phase Locking Protocol:

Locking is an operation which secures: permission to read, OR permission to write a data item. Two phase locking is a process used to gain ownership of shared resources without creating the possibility of deadlock.

The 3 activities taking place in the two phase update algorithm are:

(i). Lock Acquisition

(ii). Modification of Data

(iii). Release Lock

Two phase locking prevents deadlock from occurring in distributed systems by releasing all the resources it has acquired, if it is not possible to acquire all the resources required without waiting for another process to finish using a lock. This means that no process is ever in a state where it is holding some shared resources, and waiting for another process to release a shared resource which it requires. This means that deadlock cannot occur due to resource contention.

A transaction in the Two Phase Locking Protocol can assume one of the 2 phases:

i) Growing Phase:

In this phase a transaction can only acquire locks but cannot release any lock. The point when a transaction acquires all the locks it needs is called the Lock Point.

(ii) Shrinking Phase:

In this phase a transaction can only release locks but cannot acquire any.

2. TimeStampOrderingProtocol:

A timestamp is a tag that can be attached to any transaction or any data item, which denotes a specific time on which the transaction or the data item had been used in any way. A timestamp can be implemented in 2 ways. One is to directly assign the current value of the clock to the transaction or data item. The other is to attach the value of a logical counter that keeps increment as new timestamps are required.

The timestamp of a data item can be of 2 types:

(i)W-timestamp(X):

This means the latest time when the data item X has been written into.

(ii)R-timestamp(X):

This means the latest time when the data item X has been read from. These 2 timestamps are updated each time a successful read/write operation is performed on the data item X.

3. Multiversion Concurrency Control:

Multiversion schemes keep old versions of data item to increase concurrency.

Multiversion 2 phase locking:

Each successful write results in the creation of a new version of the data item written. Timestamps are used to label the versions. When a read(X) operation is issued, select an appropriate version of X based on the timestamp of the transaction.

4. ValidationConcurrencyControl:

The optimistic approach is based on the assumption that the majority of the database operations do not conflict. The optimistic approach requires neither locking nor time stamping techniques. Instead, a transaction is executed without restrictions until it is committed. Using an optimistic approach, each transaction moves through 2 or 3 phases, referred to as read, validation and write.

(i) During read phase, the transaction reads the database, executes the needed computations and makes the updates to a private copy of the the database values. All update operations of the transactions are recorded in a temporary update file, which is not accessed by the remaining transactions.

(ii) During the validation phase, the transaction is validated to ensure that the changes made will not affect the integrity and consistency of the database. If the validation test is positive, the transaction goes to a write phase. If the validation test is negative, he transaction is restarted and the changes are discarded.

(iii) During the write phase, the changes are permanently applied to the database.