UNIT-I

INTRODUCTION TO DBMS

DATA BASE

The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema views and reports etc.

For Ex- The college database organizes the data about the admin, staff, students and faculty etc.

Using the database, you can easily retrieve, insert and delete the information

DATA BASE MANAGEMENT SYSTEM

* DBMS is a software which is used to manage the database. For Ex: MySQL Oracle etc are a very popular commercial database which is used in different applications.

☆ DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.

☆ It provides protection and security to the database. In the case of multiple users.

☆ It also maintains data consistency

DBMS allows users the following tasks

DATA DEFINITION -

It is used for creation, modification, and removal of definition that defines the organization of data in the database

DATA UPDATION -

It is used for the insertion, modification and deletion of the actual data in the database

DATA RETRIEVAL -

It is used to retrieve the data from the database which can be used by applications for various purposes.

USER ADMINISTRATION -

It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by unexpected failure

CHARACTERISTICS OF DBMS

(A) It uses a digital repository established on a server to store and manage the information.

(B) It can provide a clear and logical view of the process that manipulates data.

(C) DBMS contains automatic backup and recovery procedures.

(D) It contains ACID (Accuracy Completeness Isolation Durability) properties which maintain data in a healthy state in case of failure

(E) It can reduce the complex relationship between data.

(F) It is used to support manipulation and processing of data

(G) It is used to provide security of data

(H) It can view the database from different viewpoints according to the requirements of the user

ADVANTAGES OF DBMS

(duplicate)

(I) CONTROLS DATABASE REDUNDANCY → It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database

(II) DATA SHARING - In DBMS, the authorized users of an organization and share the data among multiple users

(III) EASILY MAINTENANCE - It can be easily maintainable due to the centralized nature of the database system

(IV) REDUCE TIME - It reduces development time and maintenance need.

(V) BACKUP - It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restore the data if required.

(VI) MULTIPLE USER INTERFACE - It provides different types of user interface like graphical user interface, application programmable interfaces.

(VII) MAINTENANCE COST - DBMS can be costly but their maintenance price will be low.

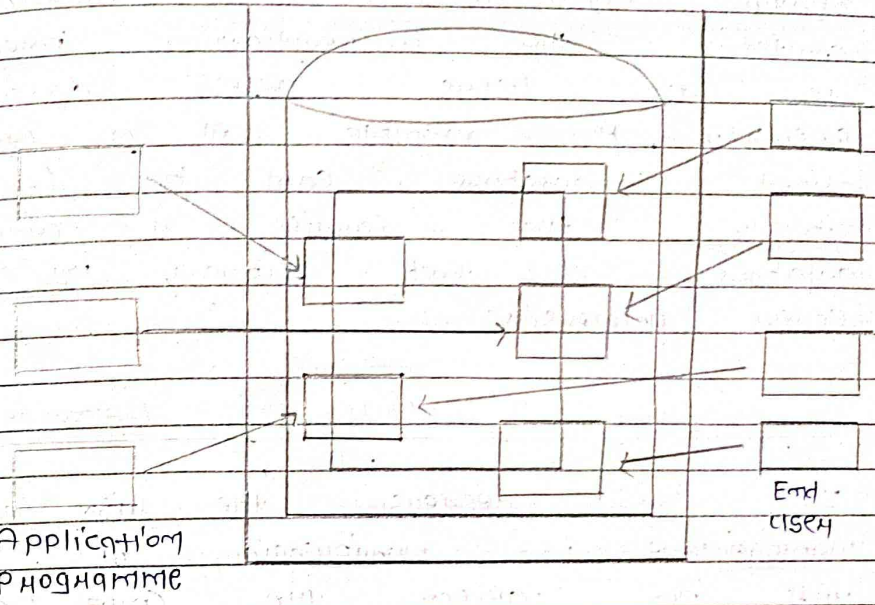
DISADVANTAGE OF DBMS

(I) **COST OF HARDWARE AND SOFTWARE** - It requires a high speed of data processor and large memory size to run DBMS software.

(II) **SIZE** - It occupies a large space of disks and large memory to run them efficiently.

(III) **COMPLEXITY** - DBMS syst creates additional complexity and requirements.

(IV) **HIGHER IMPACT OF FAILURE** - Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.



DBMS

TYPES OF DBMS

These are various types of database used for storing different varieties of data.

(I) **CENTRALIZED DATABASE** - It is the type of database that stores data at a centralized database system. It contents the users to access the stored data from different locations through

several applications. These applications contain the authentication process to let users access data securely. An example of a centralized database can be central library that carries a central database of each library in a college / university

ADVANTAGE OF CENTRALIZED DATABASE

(a) It has decreased the risk of data management i.e- manipulation of data will not affect the core data.

(b) Data consistency is maintained

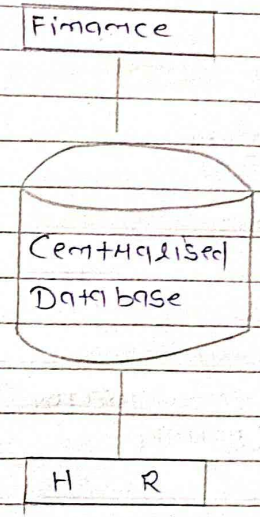
(c) It provides better data quality

(d) It is less costly

DISADVANTAGE OF CENTRALIZED DATABASE

(a) It is not easy to update such an extensive database system

(b) If any server failure occurs entire data will be lost which could be a huge loss

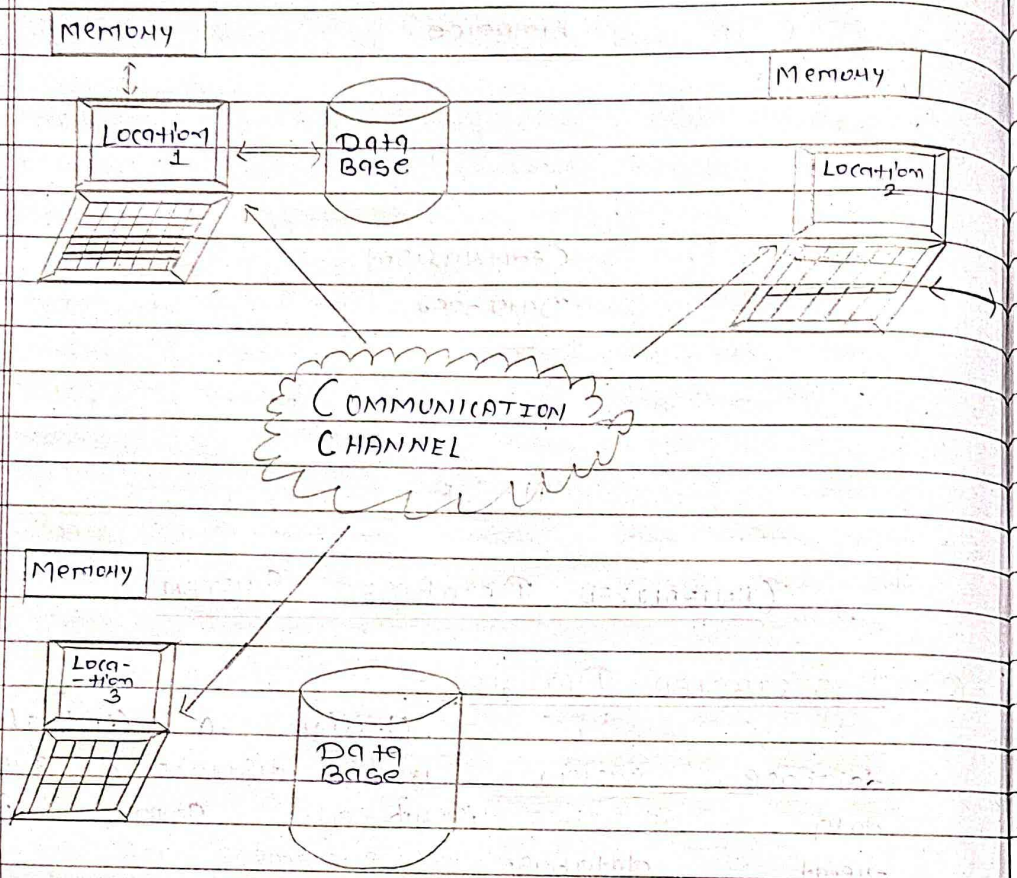


CENTRALIZED DATABASE SYSTEM

(2) DISTRIBUTED DATABASE -

Unlike a centralized database system, in distributed systems, data is distributed among different database systems of an organization. These database systems are connected via communication links Such links help the end-users to access the data easily.

Examples of the distributed database are Apache Cassandra, HBase, Ignite etc.



DISTRIBUTED DATABASE SYSTEM

(3) RELATIONAL DATABASE -

This database is based on the relational data model which stores data in the form of rows (tuple) and columns (attributes) and together forms a table (relation). A relational database uses SQL for storing, manipulating as well as maintaining the data E.F Codd Invented the database in 1970. Each table in the database carries a key that make the data unique from other.

Example of relational database are MY SQL, MICROSOFT SQL Server, Oracle etc.

Customer Name	Social Security	Customer City Stred	Customer City	A/C No.
RAM	192-83-7065	ALMA	INDORE	A-101
MOHAN	182-72-4095	NORTH	INDORE	A-102
MOHIT	172-82-4065	ALMA	INDORE	A-103

PROPERTIES OF RELATIONAL DATABASE

These are following four properties of Relational Model known as ACID Properties

(A) means Atomicity - This ensure the data operation will complete either with success or with failure

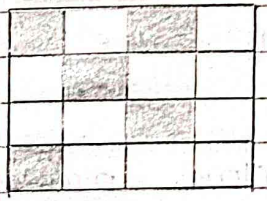
(C) means Consistency - If we perform any operation over the data it's value before and after the operation should be preserved.

(I) means Isolation - There can be concurrent users for accessing data at the same time from the database.

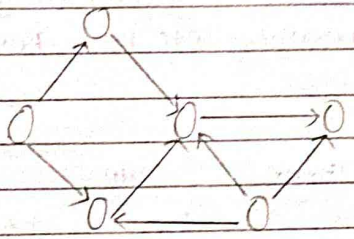
(D) means Durability - It ensure that once it completes the operation and commits the data, data changes should remain permanent.

(4) NO SQL DATABASE -

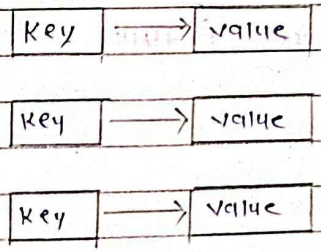
Not only SQL is a type of database that is used for storing a wide range of data sets. It is not a relational database as it stores data only in tabular form but in several diff. ways. It came into existence when the demand for building modern application increased. This, NO SQL presented a wide variety of data-base technologies in response to the demands.



COLUMN



GRAPH



Key - Value

NO SQL DBMS

ADVANTAGE OF NO SQL DATABASE

- It is a better option for managing and handling large data sets.
- It provides high scalability

(5) **CLOUD DATABASE** - A type of data-base where data is stored in a virtual environment and executes over the cloud computing platform. It provides users with various cloud computing services (SaaS, PaaS, IaaS, etc) for accessing the data-base.

There are numerous cloud platforms, but the best option are

(1) Amazon Web Services (AWS)
(2) Microsoft Azure
(3) Kamatera
(4) phoenix NAP
(5) Science Soft
(6) Google Cloud, SQL etc

(6) **OBJECT ORIENTED DATABASES** -

The type of database that uses the object based data model approach for storing data in the DBMS. The data is represented and stored as objects which are similar to the objects used in the object oriented programming lang-uage.

It is invented in 1980.

Obj-1:- Maintenance Report

Date	
Activity Code	
Route No.	
Daily Production	
Equipments Hours	
Labor Hours	

Obj-1 Instance

01-02-23
24
1-95
25
6.0
6.0

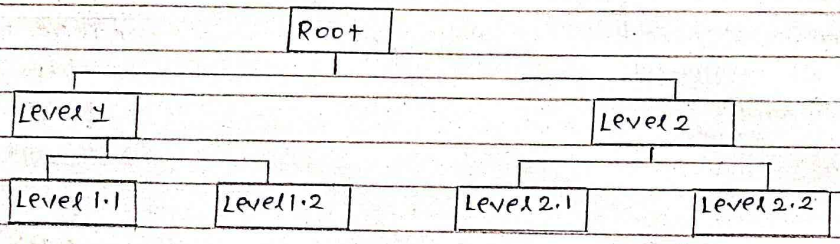
Obj-2: Maintenance activity

Activity Code	
Activity Name	
Production Unit	
Average Daily Production Rate	

(7) HIERARCHICAL DATABASES

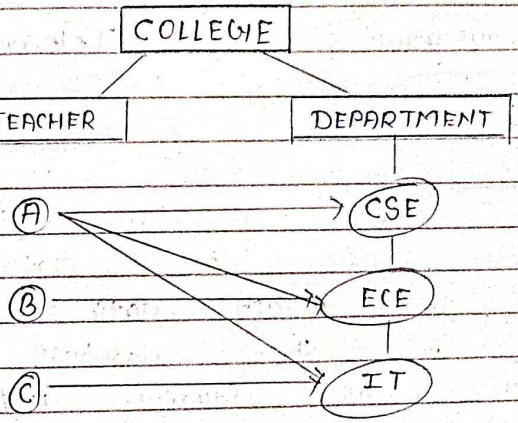
It is the type of database that stores data in the form of parent children relationship nodes. Here, it organizes data in a tree like structure.

Data get stored in the form of records that are connected via links. Each child record in the tree will contain only one parent. On the other hand, each parent record can have multiple child records.



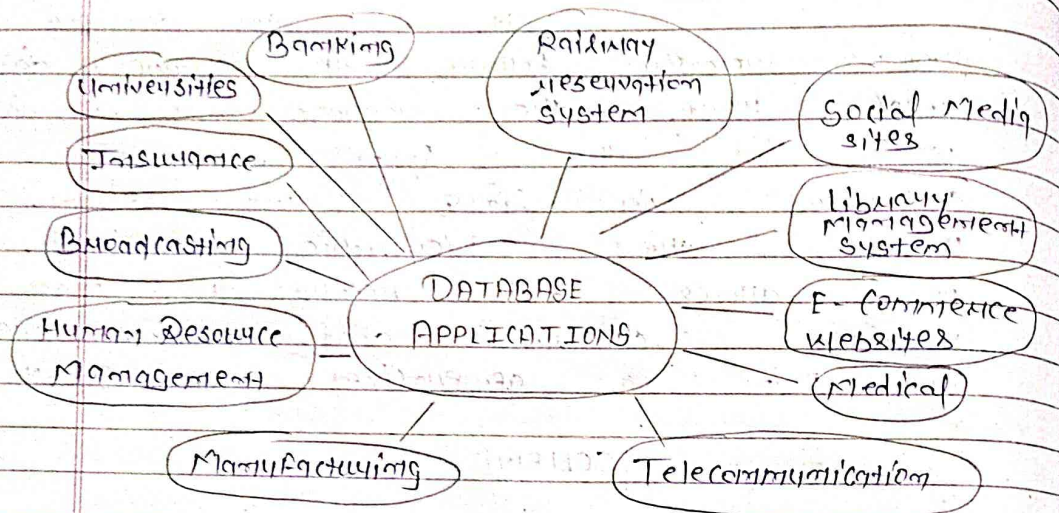
(8) NETWORK DATABASES -

It is the database that typically follows the network data model. Here, the representation of data is in the form of nodes connected via links between them. Unlike the hierarchical database, it allows each record to have multiple children and parent nodes to form a generalized graph structure.



(9) Personal Database
(10) Operational Database
(11) Enterprise Database

DATABASE APPLICATIONS



(I) Universities -

It is an undeniable application of the database. Universities have so much data which can be stored in the database

Such as Student information, Teachers Information, non-teaching Staff information, Course information, section information, Grade report information and many more. University information is kept safe and secure in the database

Anyone who needs information about the student, teachers or Course can easily

retrieve it from the database.

(II) BANKING -

It is one of the most application of database. Banks have a huge amount of data as millions of people have accounts that need to be maintained properly. The database keeps the record of each user in a systematic manner. Banking databases store a lot of information about account holders. It stores customer details balance sheets, credit and debit card details etc.

(III) LIBRARY MANAGEMENT SYSTEM -

There are hundreds and thousands of books in the library so it is not easy to maintain the records of the books in a register or diary so a balance management system is used which maintains the information of the library efficiently. The library database stores information like book name, author name book issued name etc.

(IV) INSURANCE -

An insurance company needs a database to store huge amount of data. Insurance database stores data such as policy details, user details, buyer details, payment details, Nominee details, address details etc.

(V) BROADCASTING -

It is distributing video and audio content to a dispersed audience by Television, Radio or other means. Broadcasting database stores data such as Subscribers information, event recordings, event schedules etc. So it becomes important to store broadcasting data in the database

(VI) MANUFACTURING -

In the manufacturing field a lot of data needs to be maintained regarding supply chain management, so the database maintains the data such as product details, customer information, order details purchase details, payment info., workers details, invoice etc.

Manufacturing companies produce and supply product every day

(VII) AIRLINE RESERVATION SYSTEM -

It is one of the applications of database management systems that contains data such as passengers name, passengers check-in, passenger departure, Flight schedule, number of flights, distance from source to destination, arrival-destination information, pilot details etc.

(VIII) INDUSTRIES -

The database management system is the main priority of industries because they need to store huge amounts of data. The industry database stores customer details, sales records, product lists, transactions etc. All the information is kept secure and maintained by the database.

EVOLUTION OF DB & DBMS

Data modelling and databases evolved together, and their history dates back to 1960.

The database evolution happened in the 5 "waves"

A. The first wave consisted of network, hierarchical, inverted list and object oriented DBMS. It took place from roughly 1960 - 1999

B. The relational wave introduced all of the SQL product (few non SQL) around 1990 and began to loss users around 2008.

C. The decision support wave introduced online analytical processing (OLAP) and specialized DBMS around 1990 and is still in full force today.

D. The graph wave began with the semantic wave stack from the world wide web Consortium in 1999. With property graphs appearing around 2008.

E. The NO-SQL wave includes big data and. It began in 2008

NO-SQL

GRAPH

MULTIDIMENSIONAL

THE RELATIONAL EMPIRE

PIONEERING DBMS TECHNOLOGIES

1960es | 1970es | 1980es | 1990es | 2000es | 2010es

NEED FOR DATA MANAGEMENT

It is the crucial first step to employing effective data analysis at skill, which leads to important insights that at value to your business and improve your bottom lines.

Some benefits of an effective data management solution.

(I) Visibility -

Data management can increase the visibility of your organization's data assets, making it easier for people to quickly and confidently find the white data for their analysis. Data visibility allows your company to do more organized and productive, allowing employees to find the data they need to better do their jobs.

(II) Reliability -

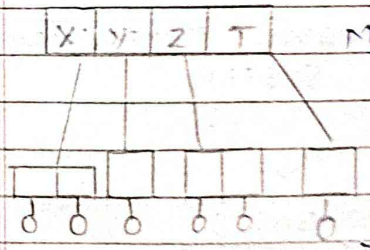
Data management helps minimize potential errors by establishing processes and policies for users and building trust in the data being used to make decisions across your organization with real, up to date data, companies can respond more efficiently to make changes and customer needs.

FILE SYSTEM VERSUS DBMS

(1) FILE SYSTEM -

The file system is basically a way of arranging the files in a storage medium like a hard disk. The file system organizes the files and helps in the retrieval of files when they are required. File systems consist of different files which are grouped into (folders) directories. The directories further contain other folders and files. The file system performs basic operations like management, file naming, giving access, rules etc.

Ex- NTFS (New Technology File System).

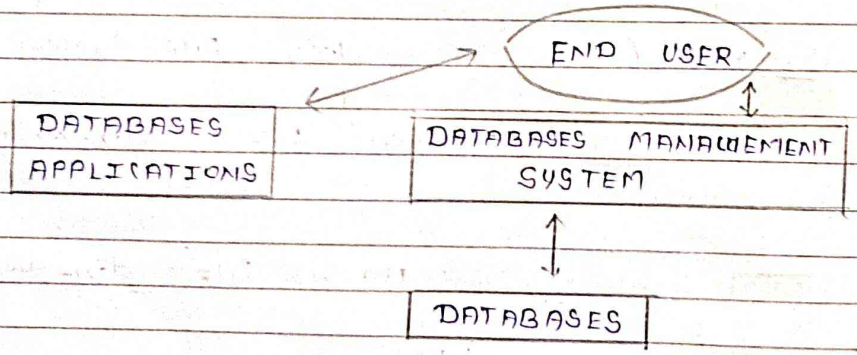


FILE SYSTEM

(2) Data Base Management System (DBMS)

It is basically software that manages the collection of related data. It is used for storing data and retrieving the data effectively when it is needed. It also provided proper security measures for protecting the data from unauthorized access. In Databases Management System the data can be fetched by SQL queries and relational algebra. It also provides ~~each~~ mechanisms for data recovery and data back-up

Ex- ORACLE, MY SQL, MS SQL Servers.



DBMS

Difference Bw File System & DBMS

BASIC	FILE SYSTEM	DBMS
STRUCTURE	It is a way of arranging the files in a storage H.D media within a computer	It is a Software for managing the databases
DATA REDUNDANCY	Redundant data can be present in a file System.	In DBMS there is no redundant data.
BACK UP AND RECOVERY	It doesn't provide inbuilt mechanism for backup and recovery of data if it is lost	It provides in house tools for backup and recovery of data even if it is lost
QUERY PROCESSING	There is no efficient Query processing in the file System	Efficient Query processing if there is DBMS

BASIC	FILE SYSTEM	DBMS
CONSISTENCY	There is less data Consistency in the file System	There is more data Consistency because of the process of normalization
COMPLEXITY	It is least Complex to as Compared to DBMS	It has more Complexity in Handling as Compared to the file system.
SECURITY CONTRAINTS	It provides least security in Comp-arison to DBMS	It has more security mechanisms as Compared to file system.
COST	It is less. Expensive then DBMS	It has a highey Cost Comparatively to a file System.
DATA INDEPENDENCY	There is no Data Independency	In it data independency exist mainly two types. (1) Logical data independency (2) Physical data

BASIC	FILE SYSTEM	DBMS
USER ACCESS	Only one user can access data at time	Multiple users can access data at a time.
MEANING	The users are not required to write Proc-edures	The users has to write procedures for managing databases.
SHARING	Data is distribute in many files So it is not easy to share data	Due to centralized archives data sharing is easy.
DATA ABSTRACTION	It gives details of storage and representation of data	It hides the internal details of databases
INTEGRITY CONSTRAINTS	They are diffi-cult to imple-ment.	They are easy to implement
ATTRIBUTES	To access data in a file users requires attributes such as File. Name, location	No such attribute are required. Mysql, SQL server

DATA MODELS

Data modelling is the modelling of the data description, data semantic and consistency constraints of the data. It provides the conceptual tools for describing the design of a database at each level of data abstraction. Therefore there are following four data model used for understanding the structure of the databases.

TYPES OF DATA MODELS

(1) Relational Data Model
(2) Semistructured Data Model
(3) Entity Relationship Data Model
(4) Object Based Data Model

(1) **Relational Data Model** - This type of model designs the data in the form of rows and columns within a table. Thus a relational data and in RM relationship table are also called relations. This model was originally described by Edgar Codd in 1969. The relational data model is the widely used model which is primarily used by commercial data processing applications.

(2) **Entity Relationship data Model** - An ER Model is the logical representation of data as object and relationship among them. Those object are known as Entities and relationship is an association among these entities. This model was designed by peter chan and published in 1976. It was widely used in database designing. A set of attributes describe the entities.

For E.g. → Student — name, Student — Id describes the 'Student' entity. A set of the same type of entities is known as an 'Entity set' and the set of the same type of relationship is known as 'relationship set'.

(3) **OBJECT - BASED DATA MODEL** - An extension of the ER model with notions of functions Encapsulation and object identity as well. This model support a rich type system that

includes structured and collection types. Thus in 1980 various database system following the object oriented approach developed here, the object are nothing but the data carrying its properties.

(4) SEMI - STRUCTURED DATA MODEL -

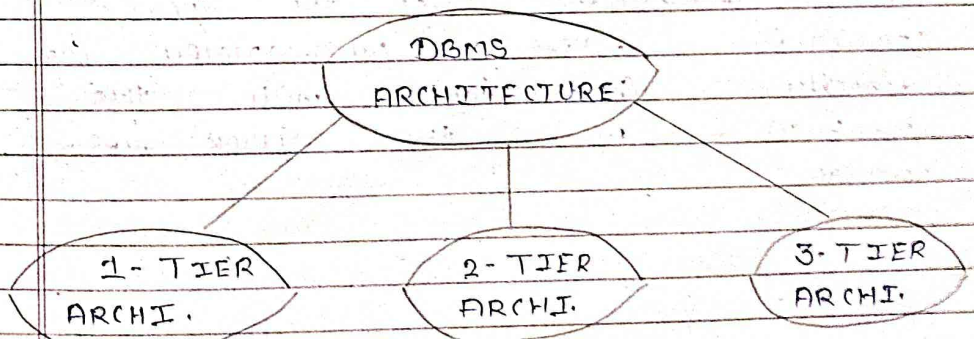
This type of data model is different from the other 3 data model (explain above). The semistructured data model allows the data specification at places where the individual data items of the same type may have different attributes sets the extensible mark up language also known as XML is widely used for representing the semi-structured data. Although XML was initially designed for including the markup information to the text doc. it gives gains importance because of its application in the exchange of data.

DBMS ARCHITECTURE -

The DBMS design depends upon it's basic client / server architecture. The client / server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.

- The client / server architecture consists of many PCs and a workstation which are connected via the network.
- DBMS architecture depends upon how users are connected to the database to get their request done.

TYPES OF DBMS ARCHITECTURE -



Database architecture can be seen as a single tier as multi-tier. But logically database architecture is of two types like: 2-Tier architecture and 3-Tier architecture

(I) 1-Tier Architecture

(A) In this's architecture, the database is directly available to the user It means the user can directly sit on the DBMS and uses it.

(B) Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.

(C) The 1-Tier architecture is used for development of the local application where programmer can directly communicate with the database for the quick response.

DATABASE | Server
NETWORK
USER | client

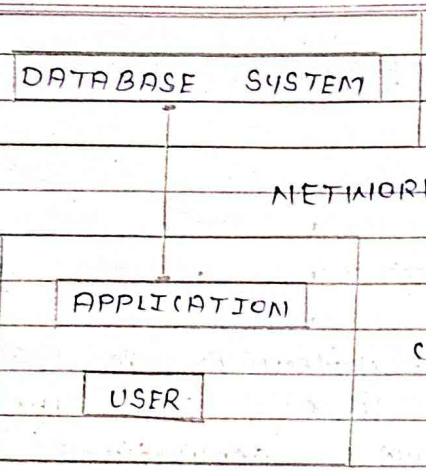
(II) 2-TIER ARCHITECTURE

(A) The 2-Tier Architecture is same as basic client-server. In the two tier architecture applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: ODBC, JDBC are used.

(B) The user interface and application programs are run on the client-side.

(C) The server side is responsible to provide the functionality like: Query Processing and transaction management.

(D) To communicate with the DBMS, client side application establishes a Connection with the server side.



(III)

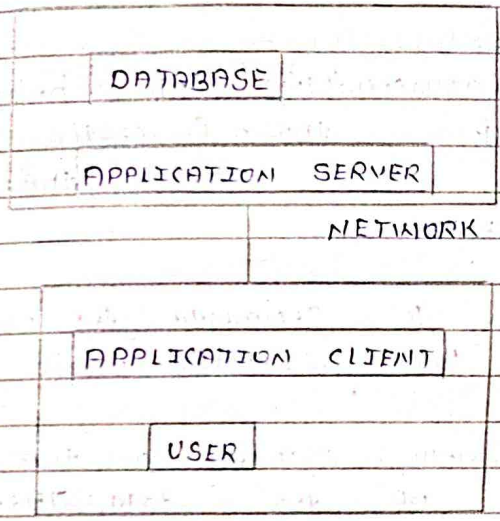
(A) 3-Tier Architecture

(A) The 3-Tier architecture contains another layer between the client and server. In this architecture client can't directly communicate with the server.

(B) The application on the client-end interacts with an application server which further communicates with the database system.

(C) End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application

(D) The 3-tier architecture is used in case of large web-development



DATA INDEPENDENCE

(A) Data Independence can be explained using the three-Schema architecture

(B) Data Independence refers characteristics of being able to modify the Schema at one level of the database System without altering the Schema at the next higher level

THERE ARE TWO TYPES OF DATA INDEPENDENCE -

(1) LOGICAL DATA INDEPENDENCE -

(I) IT REFERS CHARACTERISTICS OF being able to change the conceptual schema without having to change the external schemas

(II) IT IS USED to separate the external level FROM the conceptual view.

(III) IF we do any changes in the conceptual view of the data, THEN, THE user view of the data would not be affected.

(IV) IT occurs at the user interface level

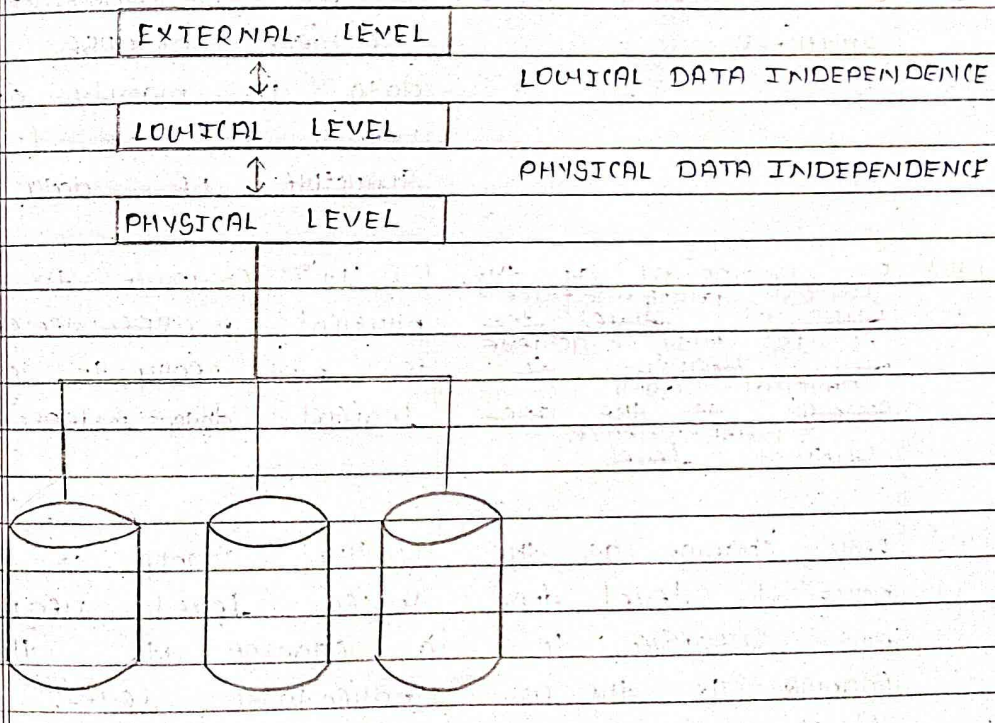
(2) PHYSICAL DATA INDEPENDENCE -

(I) IT can be defined as the capacity to change the internal schema without having to change the conceptual schema.

(II) IF we do any changes in the storage size of the database system server, THEN the conceptual structure of the database will not be affected

(III) IT is used to separate conceptual levels FROM the internal levels.

(IV) IT occurs at the logical interface level.



Difference Between Physical And Logical Data Independence

Physical Data Independence	Logical Data Independence
(I) It mainly concern about how the data is stored into the system.	(I) It mainly concerned about the structure on the changing data definition.
(II) It is easy to retrieve	(II) It is difficult to retrieve because the data is mainly depend-dent on the logical structure of data
(III) As compared at the ~~physical~~ logical independence ~~level does~~ it is easy to achieve ~~not require to~~ physical data independence, change at the application level	(III) It's compare to the physical independence it is not easy to achieve logical data independence
(IV) Any change at the physical level does not require to change at the application level.	(IV) The change in the logical level requires a change at the application level.

Physical Data Independence	Logical Data Independence
(V) The modifications made at the internal level may or may not be needed to improve the performance of the structure.	(V) The modifications made at the internal level is significant when even the logical structure of the data-base is to be changed
(VI) It is concerned with the internal schema.	(VI) It is concerned with the conceptual schema.
Eg → Change in Compression techniques, Hashing Algorithms and storage devices etc.	Eg → Add/Modify or Delete a new attri-bute.

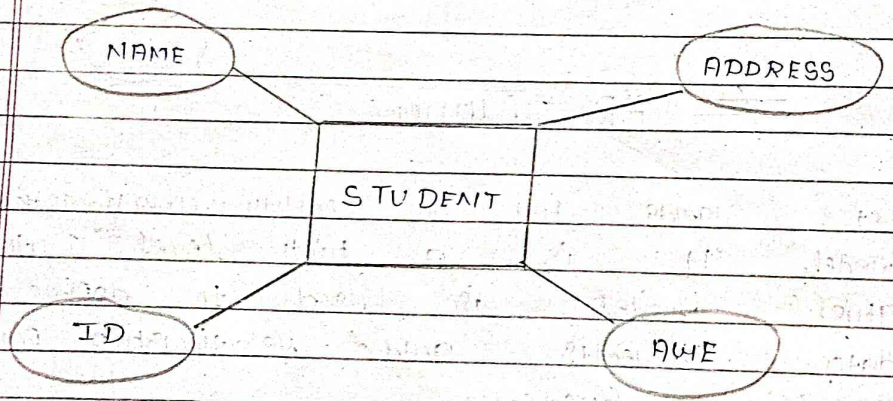
ER MODEL

(A) It stands for an entity - relationship model. It is a high level Data model. This model is used to define the data elements and relationship for a specified system.

(B) It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.

(C) In ER modeling, the database structure is portrayed as a diagram called an entity relationship diagram.

For E.g → Suppose we design a school database. In this database, the student will be an entity with the attributes like address, id, name, age etc. The address can be other entity with the attributes like city, street, name, pincode etc and there will be a relationship between them



STRUCTURE OF ER MODEL

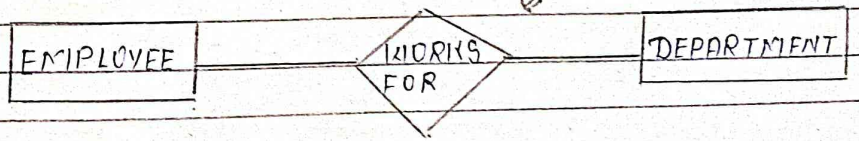
COMPONENTS OF ER - MODEL

ER - MODEL

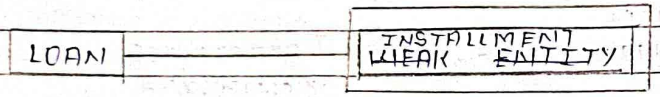
- ENTITY
  - WEAK ENTITY
- ATTRIBUTES
  - KEY ATTRIBUTES
  - COMPOSITE ATTRIBUTES
  - MULTIVALUED ATTRIBUTES
  - DERIVED ATTRIBUTES
- RELATIONSHIP
  - ONE TO ONE RELATIONSHIP
  - ONE TO MANY RELATIONSHIP
  - MANY TO ONE RELATIONSHIP
  - MANY TO MANY RELATIONSHIP

(1) ENTITY → An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.

Consider an organization as an example - Manager, product, employee, department etc can be taken as an entity. (Diamond is used for defining WORKS)

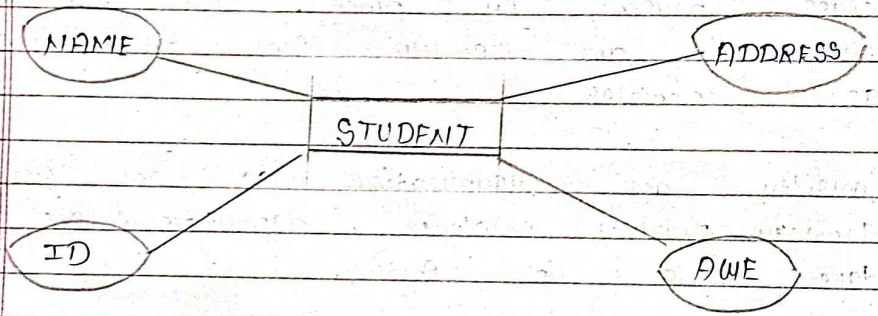


(I) WEAK ENTITY - An entity that depends on another entity called a weak entity. The weak entity does not contain any key attributes of it's own. The weak entity is represented by a double rectangle.

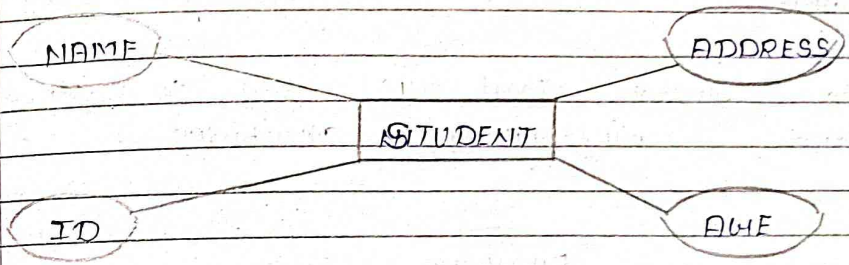


ENTITY WEAK ENTITY

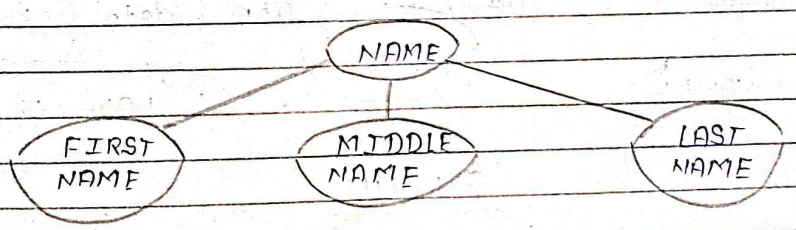
(2) ATTRIBUTES → The attributes is used to describe the property of an entity. Eclipse is used to represent an attributes.



(I) KEY ATTRIBUTES - It is used to represent the main characteristics of an entity. It represents a primary key. The key attributes is represent by an ecllipse with the text underlined.



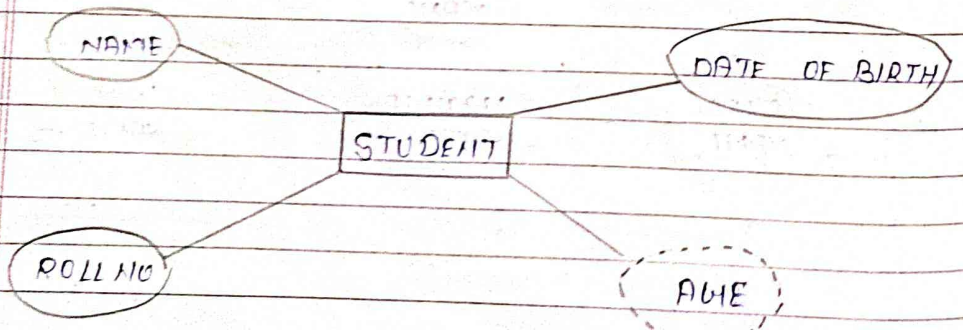
(II) COMPOSITE ATTRIBUTES - An attribute that composed of many other attributes is known as a composite attributes. The composite attributes is represented by an eclipse and those eclipse are connected with an eclipse



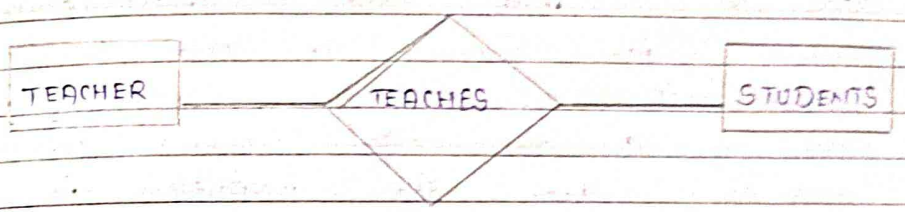
(III) MULTIVALUED ATTRIBUTES- An attribute can have more than one value. These attributes are known as multi-valued attributes. For Eg- A student can have more than one phone number.

The double oval is used to represent multivalued attributes

(IV) DERIVED ATTRIBUTES- An attribute that can be derived from another attribute is known as a derived attribute. It can be represented by a dashed oval as shown. For Eg- A person's age changes overtime and can be derived from another attribute like date of birth.



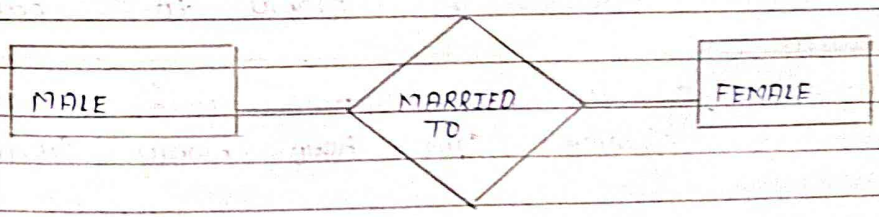
(3) RELATIONSHIP -> A relationship is used to describe the relation between entities. Diamond, Rhombus is used to represent the relationship.



TYPES OF RELATIONSHIP

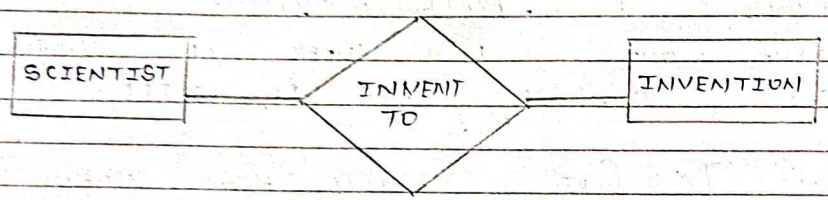
There are four types of relationship are available in ER MODEL.

(I) ONE TO ONE RELATIONSHIP- When only one instance of an entity is associated with the relationship, then it is known as one for one relationship. For Eg- A female can marry to one male and a male can marry to one female.



(II) ONE TO MANY RELATIONSHIP - When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as one to many relationship

For E.g. Scientist can invent many invention but the invention is done by the only specific Scientist

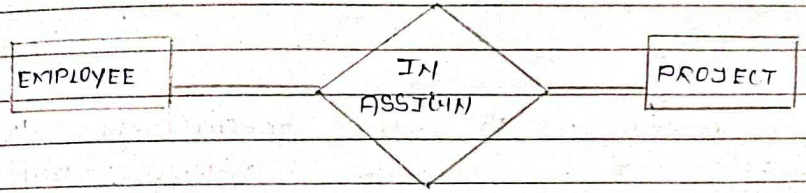


(III) MANY TO ONE RELATIONSHIP - When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as many to one relationship

For E.g- Student enrolls for only one course but a course can have many student

(IV) MANY TO MANY RELATIONSHIP - When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as many to many relationship.

For E.g- Employee can assign by many projects and project can have many employees.



ENHANCED ER MODELLING

Enhanced entity relationship diagrams are advanced database diagram very similar to ER diagram which represent the requirement and complexities of complex databases.

It is a diagrammatic technique for displaying the Sub class and Super class, Specialization and Generalization (Union) or Category. Aggregation etc.

GENERALIZATION AND SPECIALIZATION

These are very commom relationship found in real entities. However, this kind of relationship to the classical, ER Model. Specialized classes are often called Sub class while a generalized class is called Super class. It is probably inspired by object oriented programming.

An entity is a specialized type (class) of another entity. For E.g- A Technician is a special emply in a university system, faculty is a special class of emply, we call this phemomenom generalization / specialization. In this example here employee is a generalized entity class while the technician and faculty are specialized classes of employees.

Ex- The example instance of "sub class relationship". Here, we have four sets of employees : Secretary, technician and engineer. The employee is a super class of the rest three sets of individual sub class is a subset of employees set.

Employee (ENO , Name, Salary) Secretary Typing Speed

1001	EMP A	10500
1005	EMP B	24500
1007	EMP C	14500
1008	EMP D	10000
1009	EMP E	10200

1001	68

1009	Electrical Engineer

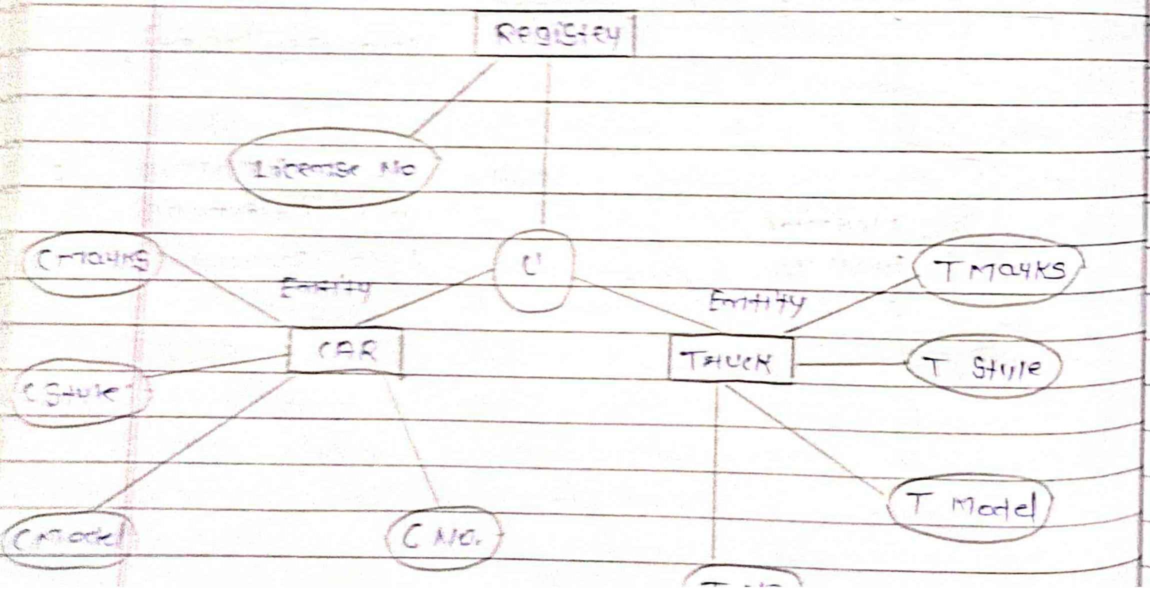
TECHNITCIAN (TRADE)

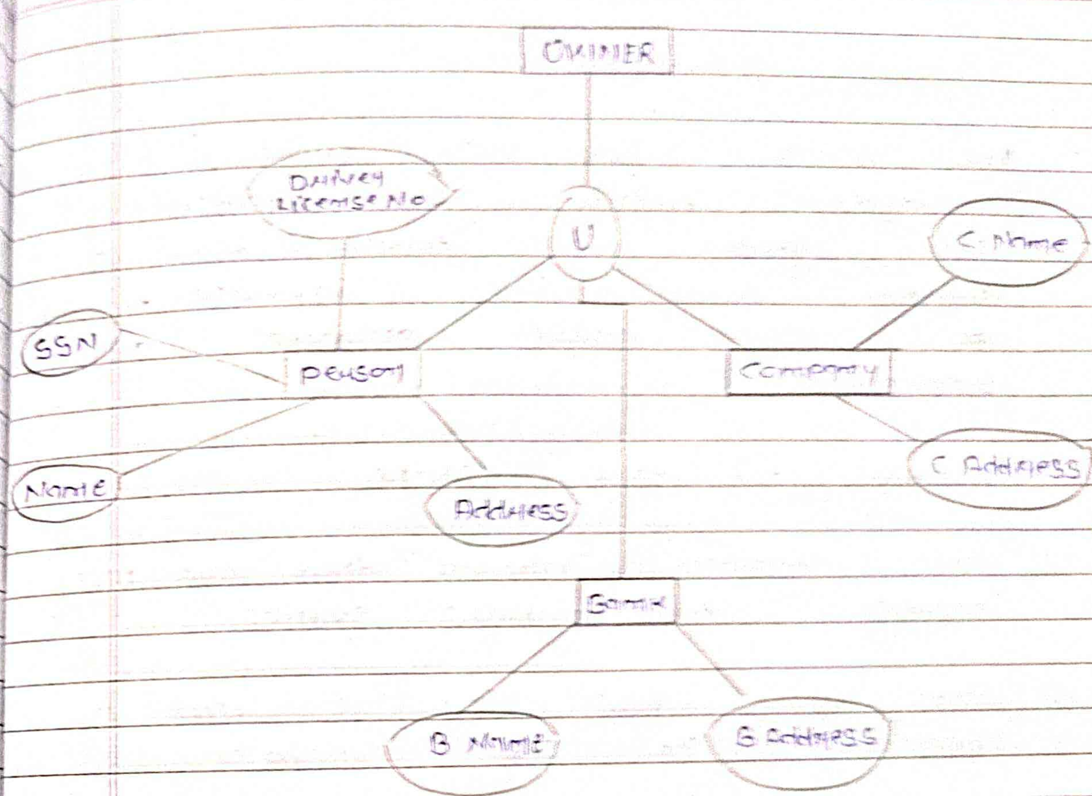
1005	Electrician
1007	Electronic
1008	Automobile

UNION

Set of library members is union of faculty, student and staff. A union relationship indicates either types. For Eg - A library members is either faculty, staff or student.

Below are two examples where shows how union can depicted. In ERD - vehicle owner is a union of person and company and RTO Registry vehicles consisting of car and truck.





AUGREWATION

EER allows for the creation of an aggregate entity that represent the group of entity as a single entity. The aggregate entity as its unique attributes and relationship

FEATURES OF EER MODEL

(1) It generates design that are more accurate to database schemas.

(2) The diagrammatical style is useful in displaying the EER schema.

(3) The modeling concepts that are included in the ER model are also included by the EER model

(4) The data properties and constants reflected by the EER model are very precise

(5) The incorporates the ideas of specialization and generalization.

(6) It is used to represent a collection of objects that are a union of distinct type of object.

UNIT-II

INTRODUCTION TO RELATIONAL DATABASE →

It represents how data is stored in relational database. A relational database is consist of a collection of tables, each of which is assigned a unique name.

Consider a relation STUDENT with attributes ROLL-NO, NAME, ADDRESS, PHONE and AGE shown in the table.

ROLL-NO	NAME	ADDRESS	PHONE	AGE
1	RAM	Delhi	91xxxxxxxx	18
2	Ramesh	Pune	91xxxxxxxx	18
3	Sujit	Delhi	91xxxxxxxx	20
4	Suresh	Mumbai	91xxxxxxxx	22

CHARACTERISTICS OF RELATIONA DATABASE

(A) Data is represented a rows and columns called relations.

(B) Data is stored in tables having relationships between them called the relational model

(C) It supports the operations like data definition, data manipulation and transaction management.

(D) Each column has a distinct name and they are atomic in nature.

(E) Each row represents a single entity.

ADVANTAGES OF RELATIONAL DATABASE

(1) Simple model - It is simple and easy to use in comparison to other languages.

(2) Flexible - It is more flexible than any other relational model present.

(3) Secure - It is more secure than other relational model.

(4) Data Accuracy - Data is more accurate in the relational data model

(5) Data Integrity - The integrity of data is maintain by the relational model.

(6) Easy Operations - It is better to perform operations in the relational model.

DISADVANTAGES OF RELATIONAL DATABASE

(A) It's model is not very good for large databases.

(B) Sometimes it becomes difficult to find the relation b/w tables.

(C) Because of the complex structure, the response time for queries is high.

RELATIONAL MODEL TERMINOLOGY

(1) RELATIONS - Most of us know what a table is from using Spreadsheet software. It is a grid made up of rows and columns. Cell in relational databases be often refer to these as "relations". In the relational model relation in a relational databases is the data being tracked at its core.

STUDENTS

RollNo	Name	Age	Address
1	HAUSH	18	Guna
2	Ram	18	Guna

(2) TUPLES - A row in a table is known as a record and a tuple. A tuple holds all the data on a single item. For Eg. If we have a table to store book details for a shop, then each tuple is an individual book, the shop sells and it will store all the data on that individual book such as the

'RAMAYAN', Book title and price

RAMAYAN	TULSIDAS	200 -/

(3) ATTRIBUTES- An individual piece of data in a record is known as field or attribute. Attributes are the properties that define an entity.

ROLL NO.	NAME	AGE	ADDRESS
1	HARSH	18	Lucknow
2	Ram	18	Lucknow
3	VEER	18	Lucknow

(4) RELATION INSTANCE- The set of the tuple of a relation at a particular instance of time is called relation instance. Table 1 shows the relation instance of the student at a particular time. It can change whenever there is an insertion, deletion or update in the database.

(5) Degree- The number of attributes in the relation is known as the degree of the relation. The student relation defined above has degree 4.

(6) CARDINALITY- The number of tuples in a relation is known as cardinality. The student relation defined above has Cardinality 3.

(7) COLUMN- The column represents the set of values for a particular attribute. The column NAME is extracted from the relation student.

NAME
VEER
Ram

(8) NULL VALUES- The value which is not known or unavailable is called a NULL value. It is represented by blank space

Roll No	Name	Age	Address
1	Veer		Guna
2	Radha	18	
3	Krishna	19	
4	Hansh		Guna

(9) RELATION KEY These are basically the keys that are used to identify the rows uniquely or also help identifying tables.

These are the following types

(1) Primary key
(2) Candidate key
(3) Super key
(4) Foreign key
(5) Alternate key
(6) Composite key

RELATIONAL ALGEBRA

Relational Algebra is a procedural query language. Relational Algebra mainly provides a theoretical foundation for relational databases and SQL. The main purpose of using Relational Algebra is to define operations that transform one or more input relations into an output relation.

Given that these operations accept relations as input and produce relations as output, they can be combined and used to express potentially many input relations (whose data are stored in the database) into a single output relation (the query results). As it is pure mathematics. there is no use of English keywords but Relational Algebra and operations are represented using Symbol

FUNDAMENTAL OPERATORS

These are the fundamental operations used in Relation Algebra.

(1) Selection (σ)

(2) Projection (π).

(1) SELECTION (σ) - It is used to select required tuples of the relations.

E.g →

A	B	C
1	2	4
2	2	3
3	2	3
4	3	4

For the above relation :- $\sigma(C>3)R$ will select the tuples which have C more than 3.

A	B	C
1	2	4
4	3	4

The selection operation only selects the required tuples but does not display them. For display the data projection operations is used.

Ex 2 →

STUDENT

Name	Age	Phone
LAKHAN	20	96 xx xx xx xx
Ravi	21	82 xx xx xx xx
Kiran	22	95 xx xx xx xx
Somu	23	86 xx xx xx xx
Somu	24	97 xx xx xx xx

→ "Name = "Somu" (Student)

Output -

Somu	23	86 xx xx xx xx
Somu	24	97 xx xx xx xx

→ "Name = "Somu" and Age "24" (Student)

Output -

Somu	24	97 xx xx xx xx

(2) PROJECTION OPERATION- It is used to project required column data from a relation and it is represented by pie symbol (π)

Ex 1 →

Consider a table 1, Suppose we want column B & C from the relation R

A	B	C
1	2	4
2	2	3
3	2	3
4	3	4

π (B, C) R will show following column

OUTPUT-

B	C
2	4
2	3
3	4

Ex 2 →

CUST - ID	CUST - NAME	CUST_STATUS
1	Rahul	Active
2	Ram	Active
3	Shyam	Inactive
4	Amit	Active

CUSTOMER TABLE

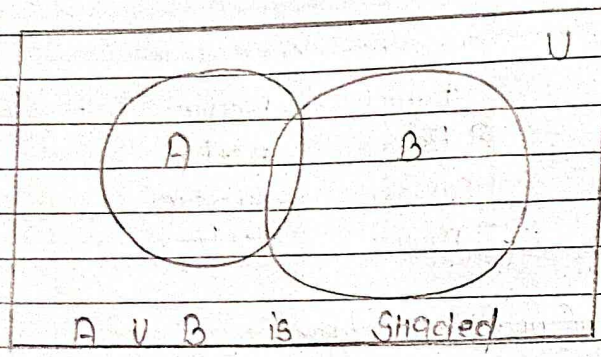
π (Customer Name, Customer Status) Customer Table

OUTPUT-

Customer Name	Customer Status
Rahul	Active
Ram	Active
Shyam	Inactive
Amit	Active

SET THEORETICAL OPERATION

(1) UNION OPERATION- Union operation in relational algebra is the same as union operation in set theory. Union of the set A & B is denoted by A ∪ B, is the set of distinct elements that's belongs to set A or set B or Both.



A ∪ B is Shaded

EXAMPLE-

Stud-Name	Roll-No.
Ram	01
Mohan	02
Vivek	13
Geeta	17

FRENCH

Stud-Name	Roll-No
Vivek	13
Geeta	17
Shyam	21
Rohan	25

GERMAN

Consider the following table of Student having different option subjects in their course

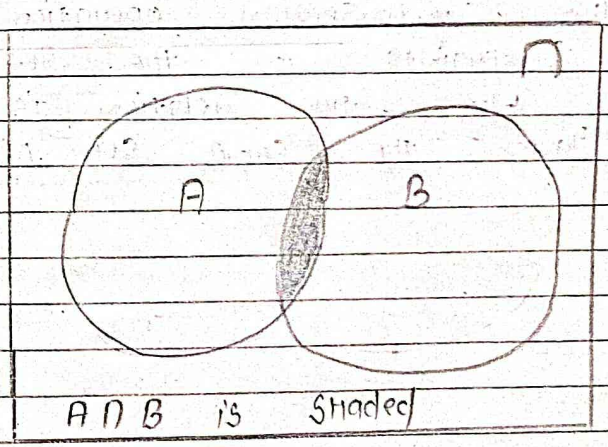
π (Stud-Name) French $\cup$ π (Stud-Name) Germany

OUTPUT-

Stud-Name
Ram
Mohan
Vivek
Geeta
Shyam
Rohan

In this Union operation, the only constraints in the Union of two relations must have the same set of attributes

(II) INTERSECTION OPERATION- Set intersection in relational algebra is the same set intersection operation in set theory. The intersection of the set A & B, denoted by $A \cap B$, is the set of elements that belongs to both A & B that is set of the common elements in A & B



A ∩ B is Shaded

EXAMPLE- We have two tables of French and Germany, the set intersection is used as follows

Stud - Name	Roll No.
Ram	01
Mohan	02
Vivek	03
Geeta	14

FRENCH

Stud_Name	Roll No.
Vivek	23
Geeta	14
Shyam	21
Roman	25

(GERMAN)

π (Stud_Name) French $\cap$ π (Stud - Name) (German)

OUTPUT-

Stud - Name
Vivek
Geeta

In this intersection operation, the only constraints in the set intersection b/w the relation is that both have the same set of attributes

JOIN OPERATION -

A join operation combines related tuples from diff. relations, if and only if a given join condition is satisfied. It is denoted by ⋈

Example -

EMPLOYEE

EMP_CODE	EMP_NAME
101	stephan
102	Jack
103	Harry

SALARY

EMP_CODE	EMP_SALARY
101	50,000
102	30,000
103	25,000

Operation: (Employee ⋈ Salary)

EMP_CODE	EMP_NAME	EMP_SALARY
101	stephan	50,000
102	Jack	30,000
103	Harry	25,000

Types OF JOIN OPERATION

- JOIN OPERATION
  - Natural Join
  - Outer Join
    - Left outer Join
    - Right outer Join
    - Full outer Join
  - Equal Join

(A) NATURAL JOIN

A natural join is a set of tuples of all combinations in 'R' 'S' that are equal on their common attribute names. and it is denoted by '⋈'

E.g→ Let's we have two tables that is "Emply" & "Salary",

EMPLOYEE

EMP_CODE	EMP_NAME
101	STEPHAN
102	JACK
103	HARRY

SALARY

EMP_CODE	EMP_SALARY
101	50,000
102	30,000
103	25,000

↗ EMP_NAME, EMP_SALARY (EMPLOYEE ⋈ SALARY)

EMP_NAME	EMP_SALARY
STEPHAN	50,000
JACK	30,000
HARRY	25,000

(B) OUTER JOIN

The outer join is an extension of a join operation i.e. it is used to deal with missing info.

E.g→

EMPLOYEE

EMP_NAME	STREET	CITY
RAM	Civil Line	Mumbai
SHYAM	Park Street	Kolkata
RAVI	M.G Street	Delhi
HARI	Nehru Nagar	Hydrabad

FACT_WORKERS

EMP_NAME	BRANCH	SALARY
RAM	INFOSYS	19,000
SHYAM	WIPRO	29,000
RAM	HCL	30,000
HARI	TCS	50,000

(EMPLOYEE ⋈ FACT_WORKERS)

EMP_NAME	STREET	CITY	BRANCH	SALARY
RAM	Civil Line	Mumbai	INFOSYS	19,000
SHYAM	Park Street	Kolkata	WIPRO	29,000
RAVI	M.G Street	Delhi	HCL	30,000
HARI	Nehru Nagar	Hydrabad	TCS	50,000

An outer Join is basically of 3 types.

(I) Left Outer Join
(II) Right Outer Join
(III) Full Outer Join

(I) LEFT OUTER JOIN

(1) It contains the set of tuples of all combinations in 'R' & 'S', that are equal on their common attributes names.

(2) In it, tuples in 'R' have no matching tuples in 'S'.

(3) It is denoted by ⟕

E.g.- We have 2 tables that are (EMPLOYEE & FACT - WORKERS)

EMPLOYEE

EMP_NAME	STREET	CITY
RAM	Civil Line	Mumbai
SHYAM	Park Street	Kolkata
RAVI	M.G. Street	Delhi
HARI	Nehru Nagar	Hyderabad

FACT — WORKER

EMP_NAME	BRANCH	SALARY
RAM	INFOSYS	19,000
SHYAM	WIPRO	29,000
KUBER	HCL	30,000
HARI	TCS	50,000

(EMPLOYEE ⟕ FACT — WORKER)

EMP_NAME	STREET	CITY	BRANCH	SALARY
RAM	Civil Line	Mumbai	Infosys	19,000
SHYAM	Park Street	Kolkata	WIPRO	29,000
HARI	Nehru Nagar	Hyderabad	TCS	50,000
RAVI	M.G. Street	Delhi	NULL	NULL

(II) RIGHT OUTER JOIN -

(1) It contains the set of tuples of all combinations in 'R' & 'S' that are equal on their common attributes names.

(2) In it, tuples in 'R' have no matching tuples 'S'.

(3) It is denoted by ⟖

E.g-

We have 2 tables that are (EMPLOYEE & FACT_WORKERS)

EMPLOYEE

EMP_NAME	STREET	CITY
RAM	Civil Line	Mumbai
SHYAM	Park Street	Kolkata
RAVI	M.G. Street	Delhi
HARI	Nehru Nagar	Hyderabad

FACT_WORKER

EMP_NAME	BRANCH	SALARY
RAM	INFOSYS	10,000
SHYAM	WIPRO	20,000
KUBER	HCL	30,000
HARI	TCS	50,000

1. EMPLOYEE ⋈ FACT_WORKERS

EMP_NAME	BRANCH	SALARY	STREET	CITY
RAM	Infosys	10,000	Civil Line	Mumbai
SHYAM	WIPRO	20,000	Park Street	Kolkata
HARI	TCS	50,000	Nehru Nagar	Hyderabad
RAVI	NULL	NULL	M.G. Street	Delhi

FULL OUTER JOIN-

(I) It is like a left or right join that it contains all rows from both tables

(II) In Full outer join, Tuples in 'R' that have no matching tuples in 'S' and tuples and S that have no matching tupple R in their common attribute name

(III) It is denoted by ⟗

E.g-> We have 2 table that are (EMPLOYEE & FACT_WORKERS)

EMPLOYEE

EMP_NAME	STREET	CITY
RAM	Civil Line	Mumbai
SHYAM	Park Street	Kolkata
RAVI	M.G. Street	Delhi
HARI	Nehru Nagar	Hyderabad

FACT WORKER

EMP_NAME	BRANCH	SALARY
RAM	INFOSYS	19,000
SHYAM	WIPRO	29,000
KUBER	HCL	30,000
HARI	TCS	50,000

EMPLOYEE TXT FACT_WORKERS

EMP_NAME	STREET	CITY	BRANCH	SALARY
RAM	Civil line	Mumbai	Infosys	19,000
SHYAM	Park Street	Kolkata	WIPRO	29,000
HARI	Nehru Nagar	Hyd-abad	TCS	50,000
RAVI	M.G Street	Delhi	NULL	NULL
KUBER	NULL	NULL	HCL	30,000

EQUI JOIN

It is also known as an inner join. It is most common join, It is built on matched data as per the equality condition the equi join uses the comparison operation (=)

CUSTOMER RELATION

CLASS_ID	NAME
1	John
2	Harry
3	Jackson

PRODUCT

PROD_ID	CITY
1	Delhi
2	Mumbai
3	Noida

CUSTOMER ⋈ PRODUCT

CLASS-ID	NAME	PROD-ID	CITY
1	John	1	Delhi
2	Harry	2	Mumbai
3	Jackson	3	Noida

RELATIONAL DATABASE DESIGN

RDB model information and data into a set of tables with rose and columns. Each row of a relation represents a record and each column rep an attributes of data. The SQL language is used to manipulate Relational Databases. The design of relational database is composed of 4 stages -

Where the data are modeled into a set of related tables.

The stages are:- [Relations/ attributes]

- Primary keys
- Relationships
- Normalization

Relational databases differ from other databases in their approach to organising data and performing transactions. In an RDD, the data are organised into Tables and all types of data access are carried out via controlled transactions.

Relational Databases Design satisfies 'ACID' (Automaticity (Atomicity Consistency Integrity Durability) properties requires from a (DD). RDD mandates the use of DBS server in applications for dealing with DM problems

The Four Stages of an RDD are as follows -

(I) Relations & Attributes -

The various tables and attributes related to each an table are identified. The tables represents entities and the attribute represent property of the respective entities

(II) Primary keys -

The attribute or set of attributes that uniquely identifying a record is identified and assigned as the primary key

(III) Relationships - It is like the various tables are established with the help of foreign key. They are attributes occurring in a table that are : Primary keys of another table. The types of relationships that can exist b/w the relations are :-

(1) One to one
(2) One to many
(3) many to many

An entity relationship diagram can be used to depict the entities. These attributes and the rela. b/w the entities in a diagrammatical way

(IV) Normalization - This is the process of optimizing the database structure. Normalization simplifies the database design to avoid redundancy and confusion. The diff. normal forms are as follows :-

(1) First Normal form
(2) Second Normal Form
(3) Third Normal form
(4) Boyce - Code Normal Form
(5) Fifth Normal form

FUNCTIONAL DEPENDENCY

A FD in DBMS is a fundamental concept that describes the relationship b/w attributes in a table. It shows how the values in one or more attributes determine the value in another. It describes how data in one column or set of columns can relate to data in another column. It helps to maintain the quality of data in DBMS

Functional Dependency is represented in the form of an equation. Here, you have a set of attributes (A, B, C, etc) and an arrow (→) denoting the dependency for ex- If we have a table of employee data with columns "EMP_ID", "First Name" "Last Name", we can express a FD like this EMP_ID → First Name, Last Name

EMP_ID → First Name, Last Name

THE FD is a relationship that exists b/w two attributes. It typically exists between the Primary key & Non key attributes with a table

$X \rightarrow Y$

The left side of Functional dependency is known as determinant. The right side of production is known as a dependent

TYPES OF FUNCTIONAL DEPENDENCY

Trivial FD Functional Dependency	Non-Trivial FD Functional Dependency

(I) Trivial Functional Dependency

1. A → B has a Trivial Functional dependency if B is a subset of A
2. The following dependencies are also Trivial like (A → A, B → B)

E.g →

Consider a table with 2 columns EMP-ID & EMP-NAME.

{EMP-ID, EMP-NAME} → EMP-ID is a Trivial Functional Dependency as EMP-ID is a subset of {EMP-ID, EMP-NAME}

Also,

EMP-ID → EMP-ID and EMP-NAME

ROLL NO.	AGE	NAME
42	24	HARSH
43	20	RAGHAV
44	26	RITIK

Here, {Roll No, Name} → Name is Trivial Functional Dependency, since the dependent Name is a subset of determinant set. Similarly Roll-No → Roll-No is a also an example of Trivial Functional Dependency

(II) NON-TRIVIAL FUNCTIONAL DEPENDENCY

In Non-Trivial Functional dependency the dependent is not strictly not a subset of the determinant that is if X → Y and

Y is not a subset of X, Then it is called Non-Trivial Functional Dependency

E.g→

Roll-No.	Age	Name
42	24	Harsh
43	20	Raghav
44	26	Rithik

Here, Roll No. Roll-No → Roll-No is Non-Trivial Functional dependency, Since the dependent name is not a subset of determinant Roll-No. Similarly {Roll-No, Name} → Age is also a Non-Trivial Functional dependency, Since A is not a subset of {Roll-No, Name}

NORMALIZATION

(I) It is the process organizing the data in the database

(II) It is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics insertion, updation → deletion Anomalies

(III) Normalization divides the big large table into smaller and links them using relationship.

(IV) The Normal foum is used to reduce the redundancy from the database table.

(V) A large database define as a single relation may result in data duplication. This repetition of data may result in

Making relations very large.

(VI) It is not easy to maintain and update data as it would involve searching many records in relation.

(VII) Wastage or poor utilisation of disk space and resources.

(VIII) The likelihood of errors and inconsistencies increase.

WHY DO WE NEED NORMALIZATION

The main reason for normalizing the relation is removing these Anomalies, Failure to remove Anomalies leads to data redundancy and can cause data integrity and other problems as the database grows.

Normalization consists of a series of guidelines that helps to guide you in creating a good database structure

DATA MODIFICATION ANOMALIES CAN BE CATEGORIZED INTO THREE TYPES

(I) INSERTION ANOMALY - It refers to when one cannot insert a new tupple into a relation due to lack of data

(II) DELETION ANOMALY - It refers to the situation where the deletion of data results in the unintended loss of some other important data

(III) UPDATION ANOMALY - The update Anomaly is when an update of a single data value requires multiples rows of data to be updated.

TYPES OF NORMAL FORMS

Normalization works through a series of stages called normal forms. The Normal form applied to individual relations. The relation is said to be in particular normal form if it satisfy constraints

Following are the various types of Normal forms :-

	1NF	2NF	3NF	4NF	5NF
Decomposition of Relation	R	R11	R21	R31	R41
		R12	R22	R32	R42
			R23	R33	R43
				R34	R44
					R45
	eliminate repeating groups	eliminate Partial Functional Dependency	eliminate Transitive dependency	eliminate Multi-value dependency	eliminate Join dependency

First Normal Form (1NF) -

A) A relation will be in 1NF if it contains an atomic value.

(B) It states that an attribute of a table cannot hold multiple values. It must hold only single valued attributes

(C) The 1st NF Disallows the multivalued attributes, composite attributes and their combination

E.g:- Relation EMPLOYEE is not in 1NF because of Multi-valued attribute EMP_PHONE

~~EMPLOYEE~~

EMP-ID	EMP-NAME	EMP PHONE	EMP STATE
14	John	91828 76425	UP
20	Harry	28666 53644	Bihar
12	Sam	62644 57099	Punjab

The Decomposition of the EMPLOYEE table into 1NF has been shown below-

EMPLOYEE

EMP-ID	EMP-NAME	EMP-PHONE	EMP-STATE
14	John	91 xx xx xx xx	UP
		91 xx xx xx xx	
20	Harry	91 xx xx xx xx	Bihar
		91 xx xx xx xx	Punjab
12	Sam	91 xx xx xx xx	

The Decomposition of the EMPLOYEE Table 1NF has been shown below-

EMP-ID	EMP-NAME	EMP-PHONE	EMP-STATE
14	John	91 xx xx xx xx	UP
14	John	91 xx xx xx xx	UP
20	Harry	91 xx xx xx xx	Bihar
12	Sam	91 xx xx xx xx	Punjab
12	Sam	91 xx xx xx xx	Punjab

SECOND NORMAL FORM (2NF) -

(A) In the 2NF, relational must be in 1NF

(B) In the second NF all known key attributes are fully functional dependent on the primary key

E.g:- 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

TEA-ID	TEA-SUB	TEA-AGE
25	CHEMISTRY	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

In the given table, Non prime attribute TEA-AGE dependent on TEA-ID which is a proper subset of A candidate key. That's why it violates the rule of 2NF

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

TEACHER_DETAILS

TEACHER_ID	TEACHER_AGE
25	30
47	35
83	38

TEACHER_SUB

TEACHER_ID	TEACHER_SUB
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 contains any transitive partial dependency.

(I) 3NF is used to achieve the data duplication.

(II) It is also used to achieve the data integrity.

(III) If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A Relation is 3NF if it holds atleast one of the following condition for every non-trivial functional dependency ($X \rightarrow Y$)

(1) X is a Super key

(2) Y is a prime attribute that is each element of Y is a part of some Candidate key

EMPLOYEE DETAILS TABLE

EMP_ID	EMP_NAME	EMP_ZIP	EMP_STATE	EMP_CITY
222	Harry	201010	UP	
333	Stephan	02228	US	
444	Lan	60007	US	
555	Katherine	00389	UK	
666	John	462007	MP	Bhopal

{EMP_ID}, {EMP_ID, EMP_NAME}, {EMP_ID, EMP_NAME, EMP_ZIP}..... So on.

(Super Key) Candidate Key : {EMP_ID}

NON - PRIME ATTRIBUTES

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

Here, EMP_STATE and 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.

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

Example-

EMPLOYEE Table:

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

EMPLOYEE_ZIP

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

(IV) ~~3NF~~ / BCNF (Boyce Code Normal Form):-

(A) BCNF is the advance version of 3NF. It is stricter than 3NF.

(B)

(C) A table is in BCNF if every functional dependency X->Y, X is the super key of the table.

(D) For BCNF the table should be in 3NF, and for every functional dependency left end side (LHS) is super key

E.g -> Let's assume there is a company, where employees work in more than one department.

EMPLOYEE TABLE

EMP_ID	EMP_COUNTRY	EMP_DEPT.	EMP_TYPE	EMP_DEPT NO.
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

In the above table functional dependencies are as follows:-

EMP_ID -> EMP_COUNTRY

EMP_DEPT -> {EMP_TYPE, EMP_DEPT_NO}

Candidate key = {EMP_ID, EMP_DEPT}

The table is not a BCNF because neither EMP_DEPT nor EMP_ID alone are keys

To convert the given table into BCNF we decompose it into 3 tables.

EMP_COUNTRY

EMP_ID	EMP_COUNTRY
264	India
264	India
364	UK
364	UK

EMP_TYPE

EMP_DEPT	EMP_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

EMP_ID	EMP_DEPT
264	Designing
264	Testing
364	Stores
364	Developing

MULTIVALUED DEPENDENCY

(A) It occurs when two attributes in a table are independent of each other but both depend on a third attribute

(B) A multivalued dependency consists of at least two attributes that are dependent on a third attribute that's why it always requires at least three attributes

Example:

Suppose there is a bike manufacturer company which produces two colours (white & black) of each model every year.

BIKE_MODEL	MANU_YEAR	COLOR
M2011	2008	white
M2001	2008	black
M3001	2013	white
M 3001	2013	black
M4006	2017	white
M4006	2017	black

Here columns COLOR and MANU-YEAR are dependent on BIKE-MODEL and independent of each other.

In this case, these two columns can be called as multivalued dependent on BIKE_MODEL. The representation of these dependencies show below.

BIKE_MODEL ->> MANUF-YEAR
BIKE_MODEL ->> COLOR

This can be read as "BIKE_MODEL multidetermined MANUF_YEAR" and BIKE_MODEL multidetermined COLOR"

FOURTH NORMAL FORM (4NF)

(A) A relation will be in 4NF if it is in Boyce Code Normal form and has no multivalued dependency.

(B) For a dependency A -> B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

Example-

STUDENT

STU-ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

The given STUDENT TABLE is in 3NF but the COURSE and HOBBY are two independent entity. Hence, there is no relationship bw COURSE & HOBBY of a student with STU_ID. In the STUDENT relation, Dancing and Singing 21 contains two courses, Computer and Math and two Hobbies Dancing & Singing. So there is a multivalued dependency on STU_ID which leads to unnecessary repetition of data.

So to make the above table into 4NF, we can decompose it into two tables

STUDENT — COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT — HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

FIFTH NORMAL FORM (5NF)

A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless

5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy

5NF is also known as Project join NF (Normal Form)

E.g-

SUB	LECT.	SEM.
Computer	Anshika	1st Sem
Computer	John	1st Sem
Math	John	1st Sem
Math	Akash	2nd Sem
Chemistry	Praveen	1st Sem

In above table John take both Computer and Math class for (Sem. 1) but he doesn't take math class for (Sem. 2) In this case combinations of all these fields

Samarth DATE PAGE

require to identify a valid data.

Suppose we add a new semester as Sem 3 but do not know about the Subj. and who will be taking that subj. So we leave lecturer and subject as NULL. But all 3 columns together acts as a primary key So we can't leave other Two Columns blanks

So to make above table into 5NF we can decompose it into 3 relations P^1, P^2, P^3

P^1

SEM	SUB.
Sem 1st	Computer
Sem 1st	Math
Sem 1st	Chemistry
Sem 2nd	Math

Samarth DATE PAGE

P^2

SUB	LECTURER
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

P^3

SEM	LECTURER
Sem 1	Anshika
Sem 1	John
Sem 1	John
Sem 2	Akash
Sem 1	Praveen

UNIT-3

DATABASE STORAGE AND QUERYING -

BASIC CONCEPTS OF INDEXING

Defin-

Indexing is used to optimize the performance of the database. By minimizing the number of disk accesses requisite when a query is processed

The Index is a type of data structure. It is used to locate and access the data in a data base table quickly.

INDEX STRUCTURE-

Index's can be created uses some DBS columns

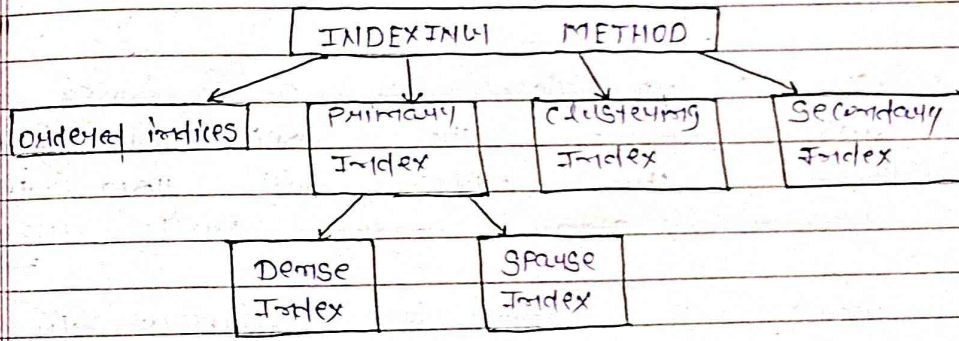
Search key	Data Reference

STRUCTURE OF Index

The 1st Column of DBS is search key that contains a copy of the primary key or Candidate key of the table The value of the primary key are stored in sorted Order. So that the corresponding data can be accessed easily

The 2nd Column of DBS is Data reference. It contains a set of pointers holding the address of the disk block where the value of the particular key can be found.

INDEXING METHOD -



(A) ORDERED INDICES -

The indices are usually sorted to make searching faster. The indices which are sorted known as Ordered Indices.

E.g- Suppose we have an employ table with thousands of records and each of which is located long

If these IDs start with 1, 2, 3, ... so on and we have to search student with ID - 543

- In the case of DBS with Non-Dense index, we have to search the disk block from starting till it reaches 543. The DBMS will read the record after reading.

  543 * 10 = 5430 bytes

- In the case of an index, we will search the file using index's and the DBMS will read the record after reading $5412 * 2 = 1084$ bytes

which are very less compared to the previous case

(B) PRIMARY INDEX -

- If the index is created on the basis of Primary key of the table, Then it is known as Primary Indexing. These Primary keys are unique to each record and contain 1:1 relation b/w the records.

As primary keys are stored in sorted order, The performance of the sorted operation is quite efficient

The primary index can be classified into two types.

1. Dense Index
2. Sparse Index

(1) Dense Index -

The contains an index record for every search key in the data file. It makes searching faster

In this, The No. of records in the index table is same as the no. of records in the main table

It needs more space to store index record itself. The index record's have search key and a pointer to the actual record on the disk

UP	UP	Agra	1,604,300
USA	USA	Chicago	0.789,378
NEPAL	NEPAL	Kathmandu	1,456,378
UK	UK	Cambridge	1,360,364

SPARSE INDEX:

- In Data File, Index record appears only for a few items. Each item points to a block
- In this, instead of pointing to each record in the main table, the index points to the record in the main table in a gap.

UP	UP	Agra	1,604,300
	USA	Chicago	0,789,378
NEPAL	NEPAL	Kathmandu	1,456,378
	UK	Cambridge	1,360,364
UK			

Ordered field

B1	2
	4
B2	6
	8
B3	10
	12
B4	14
	16
B5	18
	20
B6	22
	24

PTR. A

2
6
10
14
18
22

PTR A

2
18

BE: 2
DB File

Spare

Primary Indexing

Here, I level Space IBF for db file & IBF for Index file

CLUSTERING INDEX:

(A) It can be defined as an ordered data file. Sometimes the index is created on non-primary keys columns which may not be unique for each record.

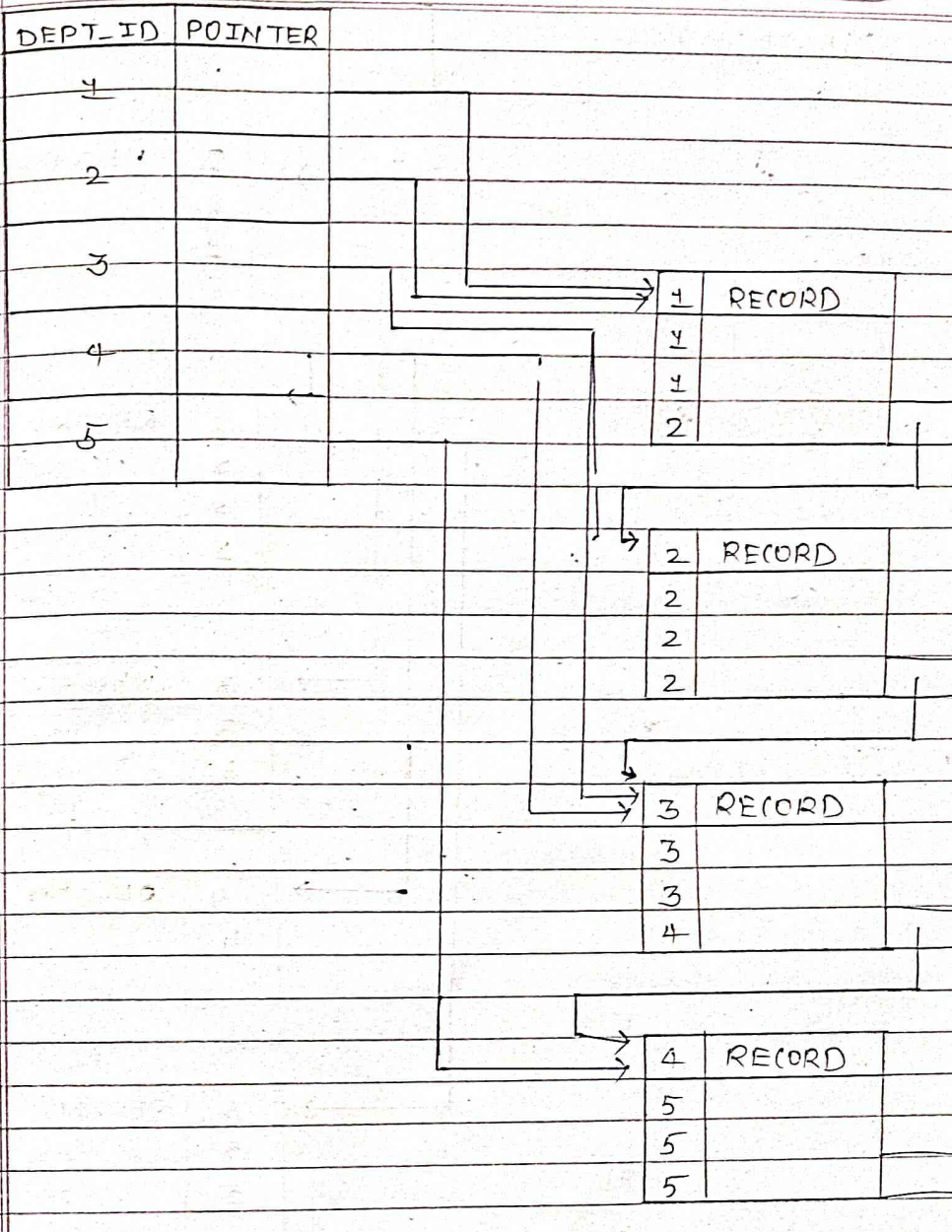
(B) In this case, to identify the record faster, we will group two or more columns to get the unique value and create index out of them. This method is called a Clustering Index.

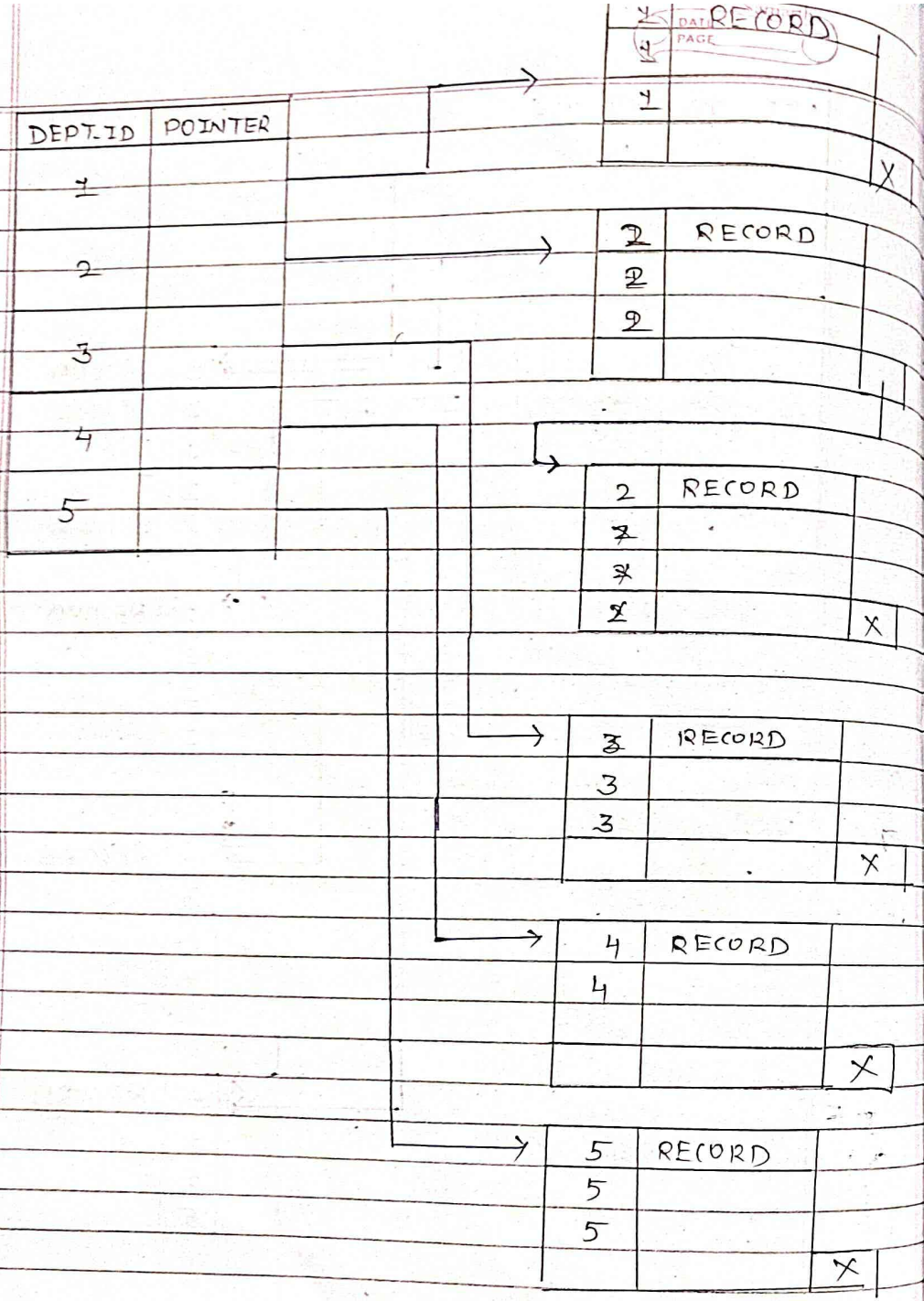
(c) The record which have similar characteristics are grouped, and indexes are created for these group

Example:

Suppose a company contains several employees in each department. Suppose we using clustering index, where all employees which belong to the same Dept_ID are considered within a single cluster and index pointers point to the cluster as a whole. Here Dept_ID is a non unique key

The previous sigma is little confusing because one disk block is shared by records which belong to the different cluster. If we use separate disk block for separate clusters then it is called better technique.



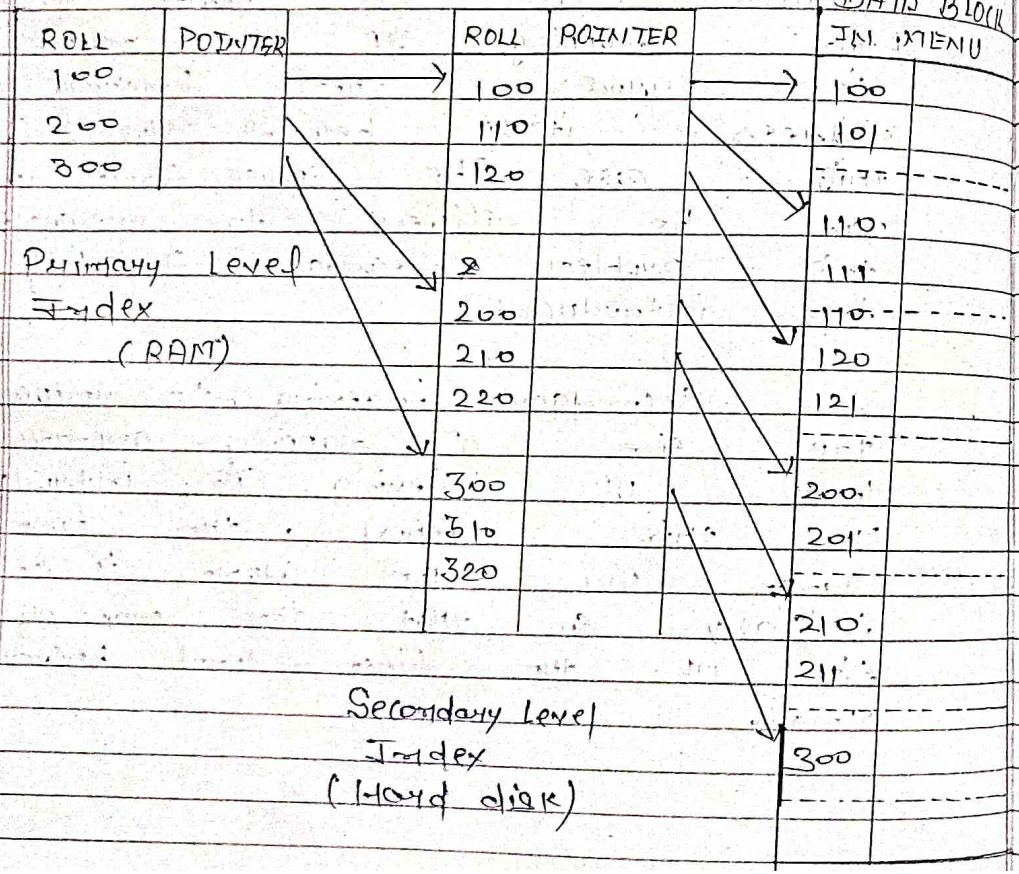


SECONDARY INDEX:

In the Sparse Indexing, as the size of the table grows, the size of mapping also grows, These mappings are usually kept in the primary memory so that address fetch should be faster. Then the secondary memory searches the actual database on the address got from mapping, If the mapping size grows then fetching the address itself becomes slower. In this case, The Sparse Index will not be efficient to overcome this problem, Secondary Indexing is introduced.

In Secondary indexing, to reduce the size of mapping, another level of indexing is introduced. In this method, the huge range for the column is selected initially so that the mapping size of the first level becomes small.

Then each range is further divided into smaller ranges. The mapping of the first level is stored in the primary memory, so that address fetch is faster. The mapping of the second level and actual data are stored in the secondary memory (Hard disk)



Now using the address it goes to the data block and start searching each record till it gets it.

This is how a search is formed in this method. Inserting, updating or deleting is also done in the same manner.

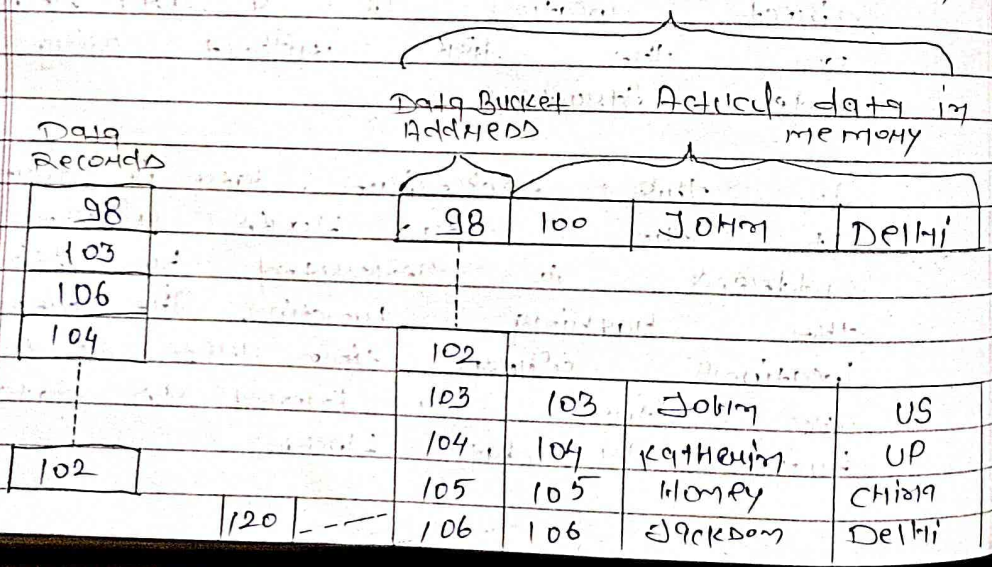
HASHING IN DBMS

In a huge database structure, it is very inefficient to search all the index values and reach the desired data. Hashing technique is used to calculate the direct location of a data record on the disk without using index structure.

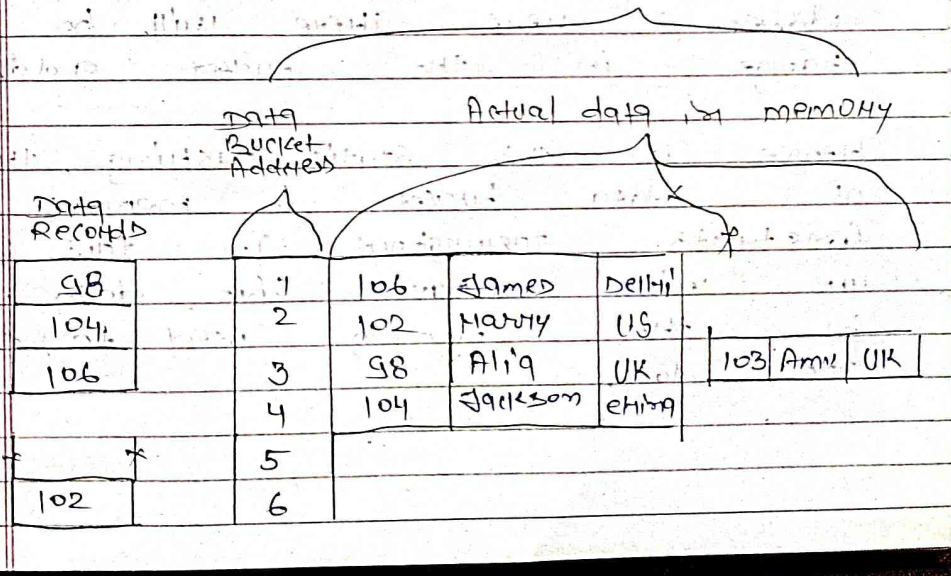
In this technique, data is stored at the data blocks whose address is generated by using the hashing function. The memory locations where these records are stored is known as data bucket or data blocks.

In this, a hash function can choose any of the column value to generate the address. Most of the time, the hash function uses the primary key to generate the address of the data block.

A hash function is a simple mathematical function to any complex mathematical function. We can even consider the primary key itself as the address of the data block. That means each row whose address will be the same as a primary key stored in the data block.



The above diagram shows data block address same as primary key value. This hash function may also be a simple mathematical function like exponential mod, cos, sin etc. Suppose we have mod (5) hash function to determine the address of the data block. In the case, it applies mod (5) hash function on the primary keys and generates 3, 3, 1, 4 & 2 respectively and records are stored in those data block addresses.



TYPES OF HASHING

Hashing
- Static Hashing
- Dynamic Hashing

Static Hashing
Dynamic Hashing

STATIC HASHING -

In it, the resultant data bucket address will always be the same. That means if we generate an address for EMP_ID = 103 using the hash function mod(5) then it will always result in same bucket address 3, Here, There will be no change in the bucket address.

Hence in this static hashing, the no. of data bucket in memory remains constants throughout. In this e.g → we will have five data buckets in the memory used to store the data.

DYNAMIC HASHING

The DH method is used to overcome the problems of static hashing like bucket overflow

In this method, data buckets grow or shrink as the records increases or decreases the method is also known as extendable hashing method

This method makes hashing dynamic, that is it allows insertion & deletion without resulting in poor performance

How To SEARCH A key

First, Calculate the hash add. of the key

Check how many bits are used in the directory, and these bits are called as i

Take the least significant i bits of the hash address

This gives an index of the directory

Now using the index go to the directory and find out the bucket addr. where the record might be

How To Insert A New Record

Firstly, you have to follow the same procedure for retrieval, ending up in some bucket

If there is still space in that bucket then place the record in it

If the bucket is full we will have to split the bucket and re-distribute the records

For Eg- Consider the following grouping of keys into buckets, depending on the prefix of their hash address

KEY	HASH - ADDRESS
1	11010
2	00000
3	11110
4	00000
5	01001
6	10101
7	10111

The last 2 bits of 2 & 4 are 00 so it will go into bucket B0, The last bits of 5 & 6 are 01, So it will go into bucket B1. The last two bits of 1 & 3 are 10. So it will go into bucket B2. The last 2 bit of 7 is 11. So it will go into B3.

	Data Records	Data Buckets
00		2 4
01		5 6
10		1 3
11		7

For access data of DBS

Measure Of Query Cost

QC is a cost in which the Enhancer considers to what amount of time your query will require. Then the analyzer attempts to pick the most ideal query plan by taking a glance adequate enquiry and insights of your information. Attempting a few execution design and choosing the most inexpensive of them.

The major of query cost in DBMS can be done by creating a framework that can make numerous design for an enquiry it tends to be finished by the means of contrasting every conceivable arrangements as far as there assessed cost.

For working out the net assessed cost of any f. arrangement, ought to be set in a deterministic and consolidated cost to get the net assessed cost of the query assessment plan.

E.g

we utilise the NO. of squares as exchanges that is basically the block from the disk and the quantity seeks to appraise the expense of a query assessment plan. Assuming that the disk subsystem takes a normal of tT seconds to move a square of information and has a normal block access time (disk lookup time in addition to rotational idleness) of tS seconds, then, at that point, an activity that moves b obstructs and performs S looks for would take b * tT + S * tS seconds. The upsides of tT and tS should be adjusted for the disk framework utilization, however, normal qualities for top-end disk today would be tS = 4 milliseconds and tT = 0.1 milliseconds, expecting a 4 kilobyte block size and an exchange pace of 40 megabytes each second.

tT - Time to transfer one block
tS - Time for one to seek
Cost for b block transfers plus S seeks

The expense assessment of a query assessment plan is determined by keeping in mind the different assets that follow as:

(1) The number of disk accesses.

(2) Time of execution taken by the CPU to execute a guy.

(3) The involved communication costs in either distributed or parallel database systems.

tT - time to transfer one block
tS - time for one to seek
Cost for b block transfer plus S seeks

b * tT + S * tS

The expense assessment of a query assessment plan is determined by keeping in mind the different assets that follow as:

1. The number of disk accesses.
2. Time of execution taken by the CPU to execute a query.
3. The involved communication costs in either distributed or parallel database systems.

Query processing for select

Query processing in DBMS

Query processing is the activity performed in extracting data from the database. In query processing, it takes various steps for fetching the data from the database. The steps involved are:

1. Parsing and translation
2. ~~Optim~~ Evaluation
3. Optimization

The query processing works in the following way:

1 Parsing and Translation

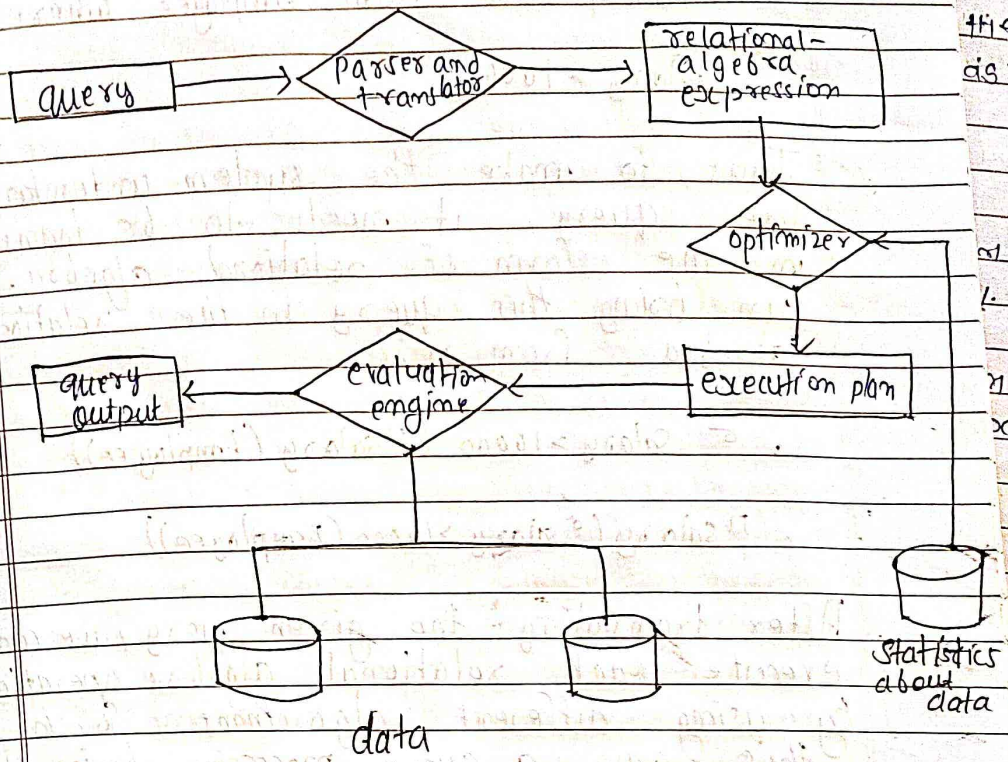
- As query processing includes certain activities for data retrieval. Initially, the given user queries get translated in high-level database languages such as SQL. It gets translated into expression that can be further used at the physical level of the file system.
- After this, the actual evaluation of the queries and a variety of query-optimizing transformations and takes place. Thus before processing a query, a computer system needs to translate the query into a human-readable and understandable language

Consequently, SQL or Structured Query language is the best suitable choice for humans. But, it is not perfectly suitable for the internal representation of the query to the system. Relational algebra is well suited for the internal representation of a query.

The translation process in query processing is similar to the parser of a query. When a user executes any query, for generating the internal form of the query, verifies the name of the relation in the database, the tuple, and finally the required attribute value.

The parser creates a tree of the query, known as 'parse-tree'. Further translate it into the form of relational algebra. With this, it evenly replaces all the use of the views when used in the query.

Thus, we can understand the working of a query processing in the below-described diagram:



Steps in query processing

Suppose a user executes a query. As we have learned that there are various methods of extracting the data from the database. In SQL, a user wants to fetch the records of the employees whose salary is greater than or equal to 10000. For doing this, the following query is undertaken:

Select emp-name from Employee where

salary > 10000;

Thus, to make the system understand the user query, it needs to be translated in the form of relational algebra. We can bring this query in the relational algebra form as:

$= \sigma_{Salary > 10000}(\Pi_{salary}(Employee))$

$= \Pi_{salary}(\sigma_{salary > 10000}(Employee))$

After translating the given query, we can execute each relational algebra operation by using different algorithms. So, in this way, a query processing begins its working.

2) Evaluation

For this, with addition to the relational algebra translation, it is required to annotate the translated relational algebra expression with the instructions used for specifying and evaluating each operation. Thus, after translating the user query, the system executes a query evaluation plan.

Query Evaluation Plan

- In order to fully evaluate a query, the system needs to construct a query evaluation plan.
- The annotations in the evaluation plan may refer to the algorithms to be used for the particular index or the specific operations.
- Such relational algebra with annotations (comment) is referred to as Evaluation Primitives. The evaluation primitives carry the instructions needed ~~primitives carry the~~ for the evaluation of the operation.
- Thus a query evaluation plan defines a sequence of primitive operations used for evaluating a query. The query evaluation plan is also referred to as the query execution plan.
- A query execution engine is responsible for generating the output of the given query. It takes the query execution plan, execute it, and finally make the output for the user query.

8] Optimization

- The cost of the query evaluation can vary for different types of queries. Although the system is responsible for constructing the evaluation plan, the user does need not to write their query efficiently.

- Usually, a database system generates an efficient query evaluation plan, which minimizes its cost. This type of task performed by the database system and is known as Query Optimization.

- For optimizing a query, the query optimizer should have an estimated cost analysis of each operation. It is because the overall operation cost depends on the memory allocations to several operations, execution costs, and so on.

Finally, after selecting an evaluation plan, the system evaluates the query and produces the output of the query.

Basics of Query optimization

Query optimization is the great important for the performance of a relational database, esp especially, for the execution of complex SQL statements. A Query optimizes decides the best methods for implementing for each query.

The query optimizer selects, for instance, whether or not to use indexes for a given query and which join methods to use when joining multiple tables. These decisions have a tremendous effect on SQL performance and query optimization is a key technology for every application, From operational system to data warehouse & analytical system to ~~contains~~ content management systems.

There is various principles of Query optimization are as follows:-

1) Understand how your database is executing your Query.

The first phase of query optimization is understanding what the database is performing different databases have different commands for this.

For ex: "Explain (SQL QUERY)" Keyword to see the Query plan. In oracle, one can used a "EXPLAIN PLAN FOR (SQL QUERY)" to see the Query plan.

2) Retrieve as little data as possible :-

The more information restored from the Query, the more resources the database is required to expand to process and solve this record. for ex:- if it can only required to fetch one column from a table, do not use 'Select *'.

3) Store intermediate result :-

Some times logic for a query can be quite complex. It is possible to produce the desired outcomes through the use of sub-queries, In line views, ~~date's~~ and UNION type statement. For those methods, the database but are directly used with in the Query this can lead to achievement issues; particulary when the transitional results have a huge number of rows.

There are various query optimization strategies are as follows :-

1) Use index ⇒ It can be using an index is the 1st strategy one should use to speed up a query.

2) Aggregate table ⇒ It can be used to prepopulating tables at higher labels. So less amount of information is required to be passed.

3) Vertical ~~partitioning~~ ⇒ partitioning ⇒ It can be used to partition the table by columns. This method reduces the amount of information a SQL query required to process.

4) Horizontal partitioning ⇒ It can be used to partition the table by data value, most often time. This method reduces the amount of information a SQL query required to process.

5) De-normalization ⇒ The process of de-normalization combines multiple table into a single table. This speeds up query implementation because fewer table joins are required.

6) Server Tuning ⇒ Each server has its parameters and provides tuning server parameters so that it can completely take benefit of the hardware

resources that can significantly speed up query implementation.

Transformation of Relational expression:-

The first step of the optimizer says to implement such expression that are logically equivalent to the given expression. For implementing such a step, we use the equivalence rule that describes the method to transform the generated expression into a logically equivalent expression.

Although there are different ways through which we can express a query, with different costs. But for expressing a query in an efficient manner, we will learn to create alternative as well as equivalent expression of the given expression, instead of working with the given expression.

Two relational algebra expressions are equivalent if both the expressions produce the same set of tuples on each legal database instance. A legal database instance refers to that database system which satisfies all the integrity constraints specified in the database schema.

However, the sequence of the generated tuples may vary in both expressions, but they are considered equivalent until they produce the same tuples set.

Equivalence Rules

The optimizer uses various equivalence rules on relational-algebra expressions for transforming the relational expressions. For describing each rule, we will use the following symbols:

$\theta, \theta_1, \theta_2 \ldots$: Used for denoting the predicates.

$L_1, L_2, L_3 \ldots$: Used for denoting the list of attributes.

$E, E_1, E_2 \ldots$: Represents the relational-algebra expressions.

Let's discuss a number of equivalence rules:

Rule 1: Cascade of σ

This rule states the deconstruction of the conjunctive selection operation into a sequence of individual selections. Such a transformation is known as a cascade of σ.

$$\sigma_{\theta_1 \wedge \theta_2}(E) = \sigma_{\theta_1}(\sigma_{\theta_2}(E))$$

Rule 2: Commutative Rule

a) This rule states that selections operations are commutative.

$$\sigma_{\theta_1}(\sigma_{\theta_2}(E)) = \sigma_{\theta_2}(\sigma_{\theta_1}(E))$$

b) Theta Join (θ) is commutative

$$E_1 \bowtie_\theta E_2 = E_2 \bowtie_\theta E_1$$ (θ is in subscript with the join symbol)

However, in the case of theta join, the equivalence rule does not work if the order of attributes is considered. Natural join is a special case of Theta join, and natural join is also commutative.

Rule 3: Cascade of π

This rule states that we only need the final operations in the sequence of the projection operations, and other operations are omitted. Such a transformation is referred to as a cascade of π.

$$\pi_{L1}(\pi_{L2}(\ldots(\pi_{Ln}(E))\ldots)) = \pi_{L1}(E)$$

Rule 4: We can combine the selections with Cartesian products as well as theta joins

1. $\sigma_\theta(E_1 \times E_2) = E_1 \bowtie_\theta E_2$
2. $\sigma_{\theta_1}(E_1 \bowtie_{\theta_2} E_2) = E_1 \bowtie_{\theta_1 \wedge \theta_2} E_2$

Rule 5: Associative Rule

a) This rule states that natural join operations are associative.

$$(E_1 \bowtie E_2) \bowtie E_3 = E_1 \bowtie (E_2 \bowtie E_3)$$

b) Theta joins are associative for the following expression:

$$(E_1 \bowtie_{\theta_1} E_2) \bowtie_{\theta_2 \wedge \theta_3} E_3 = E_1 \bowtie_{\theta_1 \wedge \theta_3} (E_2 \bowtie_{\theta_2} E_3)$$

In the theta associativity, $\theta 2$ involves the attributes from $E2$ and $E3$ only. There may be chances of empty conditions, and thereby it concludes that Cartesian Product is also associative.

[Note: The equivalence rules of associativity and commutativity of join operations are essential for join reordering in query optimization.]

Rule 6: Distribution of the Selection operation over the Theta join.

Under two following conditions, the selection operation gets distributed over the theta-join operation:

a) When all attributes in the selection condition θ_0 include only attributes of one of the expressions which are being joined.

$$\sigma_{\theta 0}(E_1 \bowtie_\theta E_2) = (\sigma_{\theta 0}(E_1)) \bowtie_\theta E_2$$

b) When the selection condition θ_1 involves the attributes of E_1 only, and θ_2 includes the attributes of E_2 only.

$$\sigma_{\theta 1 \wedge \theta 2}(E_1 \bowtie_\theta E_2) = (\sigma_{\theta 1}(E_1)) \bowtie_\theta ((\sigma_{\theta 2}(E_2))$$

Rule 7: Distribution of the projection operation over the theta join

Under two following conditions, the selection operation gets distributed over the theta-join operation:

a) Assume that the join condition θ includes only in $L_1 \cup L_2$ attributes of E_1 and E_2. Then, we get the following expression:

$$\Pi_{L1 \cup L2}(E_1 \bowtie_\theta E_2) = (\Pi_{L1}(E_1)) \bowtie_\theta (\Pi_{L2}(E_2))$$

b) Assume a join as $E_1 \bowtie E_2$. Both expressions E_1 and E_2 have sets of attributes as L_1 and L_2. Assume two attributes L_3 and L_4 where L_3 be attributes of the expression E_1, involved in the θ join condition but not in $L_1 \cup L_2$. Similarly, an L_4 be attributes of the expression E_2 involved only in the θ join condition and not in $L_1 \cup L_2$ attributes. Thus, we get the following expression

$$\Pi_{L1 \cup L2}(E_1 \bowtie_\theta E_2) = \Pi_{L1 \cup L2}((\Pi_{L1 \cup L3}(E_1)) \bowtie_\theta ((\Pi_{L2 \cup L4}(E_2))$$

Rule 8: The union and intersection set operations are commutative.

$E_1 \cup E_2 = E_2 \cup E_1$

$E_1 \cap E_2 = E_2 \cap E_1$

However, set difference operations are not commutative.

Rule 9: The union and intersection set operations are associative.

$(E_1 \cup E_2) \cup E_3 = E_1 \cup (E_2 \cup E_3)$

$(E_1 \cap E_2) \cap E_3 = E_1 \cap (E_2 \cap E_3)$

Rule 10: Distribution of selection operation on the intersection, union, and set difference operations.

The below expression shows the distribution performed over the set difference operation.

$\sigma_p(E_1 - E_2) = \sigma_p(E_1) - \sigma_p(E_2)$

We can similarly distribute the selection operation on $\cup$ and $\cap$ by replacing with -. Further, we get:

$\sigma_p(E_1 - E_2) = \sigma_p(E_1) - E_2$

Rule 11: Distribution of the projection operation over the union operation.

This rule states that we can distribute the projection operation on the union operation for the given expressions.

$\Pi_L(E_1 \cup E_2) = (\Pi_L(E_1)) \cup (\Pi_L(E_2))$

Apart from these discussed equivalence rules, there are various other equivalence rules also.

Estimating Statistics of Expression results in DBMS

In order to determine ideal plan for evaluating the query, it checks various details about the tables that are stored in the data dictionary. These informations about tables are collected when a table is created and when various DDL/DML operations are performed on it. The optimizer checks data dictionary for:

- Total number of records in a table, nr. This will help to determine which table needs to be accessed first. Usually smaller tables are executed first to reduce the size of the intermediary tables. Hence it is one of the important factors to be checked.

- Total number of records in each block, fr. This will be useful in determining blocking factor and is required to determine if the table fits in the memory or not.

- Total number of blocks assigned to a table, br. This is also an important factor to calculate number of records that can be assigned to each block. Suppose we have 100 records in a table and total number of blocks are 20, then fr can be calculated as nr/br = 100/20 = 5.

- Total length of the records in the table, lr. This is an important factor when the size of the records varies significantly between any two tables in the query. If the record length is fixed, there is no significant affect. But when a variable length records are involved in the query, average length or actual length needs to be used depending upon the type of operations.

- Number of unique values for a column, d Ar. This is useful when a query uses aggregation operation or projection. It will provide an estimate on distinct number of columns selected while projection. Number groups of records can be determined using this when aggregation operation is used in the query.
  E.g.; SUM, MAX, MIN, COUNT etc.

- Levels of index, X. This data provides the information like whether the single level of index like primary key index, secondary key indexes are used or multilevel indexes like B+ tree index, mergesort index etc are used. These index levels will provide details about number of block access required to retrieve the data.

- Selection cardinality of a column, S A. This is the number of records present with same column value as A. This is calculated as nr/d Ar. i.e.; total number of records with distinct value of A. For ex, suppose EMP table has 500 records and DEPT-ID has 5 distinct values. Then the selection cardinality of DEPT-ID in EMP table is 500/5 = 100. That means, on an average 100 employees are distributed among each department. This is helpful in determining average number of records that would satisfy selection criteria.

- There many other factors too like index type, data file type, sorting order, type of sorting etc.

Choice of Evaluation Plans

So far we saw how a query is parsed and traversed, how they are evaluated using different methods and what the different costs are when different methods are used. Now the important phase while evaluating a query is deciding which evaluation plan has to be selected so that it can be traversed efficiently. It collects all the statistics, costs, access / evaluation paths, relational trees etc. It then analyses them and chooses the best evaluation path.

Like we saw in the beginning of this article, same query is written in different forms of relational algebra. Corresponding trees for them too is drawn by DBMS. Statistics for them based on cost based evaluation and heuristic methods are collected. It checks the costs based on the different techniques that we have seen so far. It checks for the operator, joining type, indexes, number of records, selectivity of records, distinct values etc from the data dictionary. Once all these informations are collected, it picks the best evaluation plan.

Have look at below relational algebra and tree for EMP and DEPT.

Code

Π EMP-ID, DEPT-NAME (σ DEPT-ID = 10 AND EMP-LAST-NAME = 'Joseph' (EMP) ∞ DEPT)

or

Code

Π EMP-ID, DEPT-NAME (σ DEPT-ID = 10 AND EMP-LAST-NAME = 'Joseph' (EMP ∞ DEPT))

or

Code

σ DEPT-ID = 10 AND EMP-LAST-NAME = 'Joseph' (Π EMP-ID, DEPT-ID (EMP ∞ DEPT)), DEPT-NAME;

What can be observed here? First tree reduces the number of records for joining and seems to be efficient. But what happens if we have index on DEPT-ID? Then the join between EMP and EMP can also be more efficient.

But we see the filter condition on EMP table, we have DEPT-ID = 10, which is index column. Hence first applying selection condition and then join will reduce the number of records as well as make the join more efficient than without index.

Next are the projected columns - EMP-ID and DEPT-NAME. they are all distinct values. There cannot be duplicate values for them. But we are selecting those values for DEPT-ID = 10, hence DEPT-NAME has only one value. Hence their selectivity is same as number of employees working for DEPT-ID = 10.

But we are selecting only those employees whose last name is 'Joseph'. Hence the selectivity is min (distinct (employee (DEPT-ID)), distinct (employee (DEPT-ID, JOSEPH))). Obliviously distinct (employee (DEPT-ID, JOSEPH)) would have lesser value. The optimizer decides all these factors for above 3 trees, and then decides first tree would be more efficient. Hence it evaluates the query using first tree.

UNIT-4

CONCURRENCY CONTROL

DEFINATION OF CONCURRENCY:-

In DBMS when we excute all the transactions together, we use concurrency control technique. In which we ensure that data is updated correctly or not.

We get need of concurrency control, Because, whenever we update data in DBMS, sometimes it's not update correctly. It can become Incomplete or Inconsistent, For solving this problem we use concurrency control technique.

DBMS TRANJECTION :-

There is a group of Transaction operation. which treat them as an logical operation.

For E.g → When we withdraw money from Bank A/C, Or we deposit money in it.

UNIT-4

CONCURRENCY CONTROL

DEFINATION OF CONCURRENCY:-

In DBMS when we excute all the transactions together, we use concurrency control technique, In which we ensure that data is updated correctly or not.

We get need of concurrency control, Because, whenever we update data in DBMS, sometimes it's not update correctly, It can become Incomplete or Inconsistent, For solving this problem we use concurrency control technique.

DBMS TRANJECTION :-

There is a group of Transaction operation. which treat them as an logical operation.

For E.g -> when we withdraw money from Bank A/C, Or we deposit money in it.

ACID PROPERTIES :-

Transactions have four properties, we called them as ACID Properties.

(1) A - Atomicity
(2) C - Consistency
(3) I - Isolation
(4) D - Durability

(1)(A)tomicity - When a transaction is completed in one step, then it is atomic.

(2) (C)onsistency - When a transaction happened, the data is consistence b/w one state to another.

(3) (I)solations - When two or more than two transaction are execute at same time, then one transaction cannot effect another transaction.

(4) (D)urability - When a transaction is completed, then the changes happened, they will remain permanently.

TYPES OF TRANSACTION

They are of two types -
(1) Implicit Transaction
(2) Explicit Transaction

(1) IMPLICIT TRANSACTION -

In it, SQL database engine start a new transaction, when present transaction can be committed or roll back.

For on / off the transaction, we use this statement.

[SET IMPLICIT_TRANSACTIONS ON/OFF]

(2) EXPLICIT TRANSACTION - In it, Both starting & ending transaction are clearly defined.

For starting this transaction, we use Begin Transaction, and completing it we use Commit transaction or we use roll back transaction.

The main difference between Implicit and explicit are Implicit transaction can done automatically Explicit transaction can't be user defined.

LOST UPDATE TRANSACTION

This type of problem occur, when multiple transaction execute concurrently and update from one or more transactions get lost.

Example :-

Transaction T_1	Transaction T_2
R (x)	
W (x)	
	W (x)
	Commit
Commit	

Here,

(1) T_1 reads the value of x (= 10 say)
(2) T_2 updates the value to x (=15 say) in the buffer.
(3) T_2 does blind write x = 25 (write without read) in the buffer.
(4) T_2 Commits.
(5) When T_1 commits it writes x=25 in the database

IN THIS EXAMPLE :

(1) In T_1 data base x will getting value be written.
(2) In this way T_1 cannot be updated.

DIRTY READ PROBLEM :→

The : data which is written by uncommitted transaction and is read by. It is known as dirty Read.

* This read is known as dirty Read. Because -

(1) There is a possibility of comeback of Uncommitted transaction.
(2) In this way, Uncommited transaction makes another transaction a value which is not present
(3) This database can go in inconsistency way.

NOTE -

(1) Dirty Read always go in inconsistency
(2) It is become a problem when, Uncommitted transaction fails and it comebacks after some time, for some reasons

Transaction T_1	Transaction T_2
R (A)	R (A) // Dirty Read
W (A)	W (A)
Failure	Commit

Here,

(1) T_1 reads the value of A.
(2) T_2 updates the value of A in the buffer.
(3) T_2 reads the value of A from the buffer.
(4) T_2 writes the update value of A.
(5) T_2 commits.
(6) T_1 fails in later stages and Roll back.

IN THIS EXAMPLE-

(1) T_2 uncommited Transaction, It reads the A's dirty value which was provided by T_1.
(2) In after stages T_2 become unsuccessful, and it is setback.
(3) In this, The value read by T_2 is wrong.
(4) For this of database is in consistent.

UN-REPEATABLE PROBLEM-

This problem occurs when a transaction is provided for read unrepeated. For one variable there was different-different value for it in it, There was different-different read operation.

When the read update value in it.

T_1	T_2
R (x)	
	R (x)
W (x)	
	R (x) // Unrepeatable (read) Problem

Here,

(1) T_1 reads the value of x (=10 Say)
(2) T_2 reads the value of x (=10)
(3) T_1 reads the value of x (from 10 to 15 Say) in the buffer.
(4) T_2 again reads the value of x (but =15)

In this example

(1) In their second reading T2 gets another value to read.

(2) T2 gets amaze that how the value of x changes because, according to him, T1 is in isolation.

Phantom Read Problem :→

This problem is happened when a transaction read some variable from buffer, And after this it can read this variable. Then he came to know that, This variable is not present.

T1	T2
R(x)	
	R(x)
Delete(x)	
	R(x)

Here,

(1) T1 reads the value of x (=10 say)

(2) T2 reads the value of x (=10)

(3) T1 delete the value of x (=10)

(4) T2 tries reading of x but does not find.

Concurrency Control Techniques :→

Concurrency control provide by database-

(1) In Between transaction apply isolation.

(2) By maintaining the consistency of Transaction, by the help of serializability we have to maintain it's data base stability.

(3) Resolve read-write and write-read conflict.

Varies Concurrency control Techniques-

1. Two-Phase locking Protocols.
2. Time Stamp Ordering Protocol.
3. Multi version Concurrency Control.
4. Validation Concurrency Control.

1. Two-Phase Locking Protocols-

Locking is a operation. which confirms the permission of Read & write of Data Item.

Two-phase locking protocol does not create the possibility of dead lock.

Three Functions which are done in Two-Phase Locking

I. Lock Acquisition
II. Modification of data
III. Release lock

This locking Protocol allows all resources. and Stopping dead lock If any other process have to use lock without any wait They have possibility to get all resources.

It means that a process is not in any condition where they are not in a condition to get those resources And for using the those Resources and allowing them they are waiting. which they have need.

resources

It means, The dead lock in a condition where it is not become dead lock. transaction In Two-Phase locking Protocol they can use one phase from Two phase which can provided.

a. Growing-Phase-

this
In a phase only one transaction can get lock, But cannot allow the lock. The point when a transaction get all their required lock. Then it is known as lock Point.

b. Shrinking-Phase-

In a phase, a transaction can apply the lock, But cannot get them.

(2) TIME STAMP ORDERING PROTOCOL -

This protocol is a tag, which is attached with a transaction and data item.

It denotes a specific time which transactions and data items can be used anywhere.

Time stamp can be implimented by two ways.

First it is directly assigns with the current value of the clock to the transaction or data item

Second is attached with the value of the logical counter increment as a new time stamps are required.

Time stamp data items are of two types -

① W- Time Stamp (x)
② R- Time Stamp (x)

(A) W- Time Stamp -

It means with latest time, when data item is write in x

(B) R- Time Stamp (x) -

It means with latest time, when i.e. read data item x, These two stamps are updated always. when ce done a successfull Read write operation.

③ MULTIVERSION CONCURRENCY CONTROL -

Multiversion Schemes Concurrency for Smoothing them we can keep old version of data item.

Multiversion two phase locking prepares successful written data Time stamp uses for labelling the versions when read (x) operation applied.

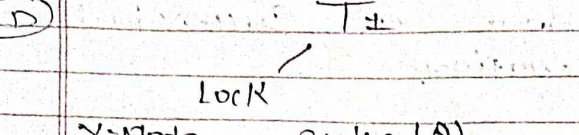
On the basis of Transaction Time Stamp, we choose required versions of x.

(A) Multiversion concurrency control provide us database access without locking data.

(B) It increases performance of database, Because, It is on Multiuser Environment.

(C) Database applications can't wait for long time. Because there could no load on read request.

(D)



(E) Multiversion Concurrency Control Techniques keeps data items of old data values. In System when a data item is updated.

(F) In it, many of values of data item values is maintained. For which we known it as Multiversion Concurrency Control Technique.

(4.) VALIDATION CONCURRENCY CONTROL -

Validation Phase is also known as Optimistic Concurrency Control Technique.

In this Validation Phase Protocol Transaction executes in three phase -

(I) Red Phase
(II) Validation Phase
(III) Write Phase

(I) RED PHASE -

In this phase Transaction read and executed. It is used to read and store by the values of Temporary variables. Without the actual data base variables. Done all kind of write of read in Temporary variables.

(II) VALIDATION PHASE -

In this phase, Temporary variable value is against the actual data validation/allowed by which we can see that, there is no kind of sequence breaking done.

(3) WRITE PHASE -

If Transaction registration is valid. Then Temporary result data base on write the System. Otherwise Transaction is asked go-back.

⇒ START (T_i) =

It contain time when T_i executes.

⇒ Validation (Ti) =

It contain time when read phase is ended. And starts the validation phase.

⇒ FINISH (Ti) =

It also Contain time when write phase of Ti is completed.

- The use of this protocol is validate phase by the use of time Stamp for transaction, Used for decide the Time Stamp. It is the actual phase who decides that a transaction can be commited or roll backed.
- Hence TS (T) = Validation (T)
- At the time of validation process Serializibility is decided. It cannot be predecided.
- When we excute the transaction, It clarifies the concurrency. And done Conflict in less amount.
- In this way this type of transaction is happened. In which the amount of roll back is less.

RECOVER TECHNIQUE OR RECOVERY CONCEPT :⇒

A data base of some Computer system can be crashed. Then restore the data in a same way is known as Recovery.

If data base does not came back in old condition. Then we can restore the data in which condition they are present.

Recovery is used to get the data-base back in the same old condition. The working of data base is done previous just like it is previously done.

For recovering data this is not important that we have to retrieve data in the present condition in DBA.

Because we have to save / update data every time in DBA (Data Base Administration)

which helps us to get back our data at the time when it was crashed.

Types Of Recovery -

(1) Log Base Recovery
(2) Shadow Aging Recovery

(1) Log Based Recovery -

This type of recovery is based on log, by records These records are called as log records, whenever a transaction is begin. Then beginning log is recorded in it there was start flage identifier.

Whenever we have to update our store data in data base. Then Log records the data in some following ways -

→ Amount of Transaction identifier
→ Those data on which work was done.
→ Old values of those data.
→ New values of those data.
→ Type of log record.

If data base of one log record have pointer of another log record. which points different - different log record, whenever it gives commit or roll-back. Them records store permanently In this log files present time backup is kept.
This was done by data base.

User doesn't have to no need to take backup of this.

For restore complete data base, DBA first restore all offline backup, Then it restore all online backup. By which log files comes in their previous state,
Data base Aggregate the log files and data in log files we keep transactions informations and

all the related data with that is kept in it.

In it, we adopt two main methods-

(A) Deffered Update

(B) Immediate Update

(2) SHADOW PAGING RECOVERY-

Database is in a defined size which was divided in blocks, which is known as pages.

Database is logically divided. There size is depended on data base. In a database there are many of pages. Their sequence is starting from 1. For finding them we use page table. With the process of one transaction, we can make two different types of pages.

(A) Current

(B) Shadow

When a transaction is begin. Then both pages of table are identicle By transaction there was no changes in Shadow page.

By a data store operation we can take changes in current page. For finding data page all transaction use current page table.

Shadow page store in a permanent medium. By permanent medium we mean to store memory. It's benefit is that, If a system is fail at the time of transaction. Then we can restore database easily.

Current page is stored in temporary memory. If System fail in between the process then the data is delete automatically, Then we have no need of Undo.

DATA BASE SECURITY.

(1) By database security. To secure the confidential information related to something. And stop the loss of the data.

(2) By the help of database security, technique. We can secure our data from attackers or Hackers.

(3) The work of Data Base Security is done by DBA.

(4) Data base security is a process by which important information of data base / data provide security. With its help user data is secured. By which a data is accessed by a authorized user.

(5) In simple words we can say that, data base security system technique which provides security to our data, By which we can save our data from Cyber-Attack.

(6) In Cyber Attack, a unauthorized person can try to steal to our data. By which attackers can user their data for wrong purpose.

(7) Data base security is used to manage data.

→ CONTROL METHODS OF DATABASE SECURITY

For securing our data base we use following methods -

(I) Authentication
(II) Access Control
(III) Inference Control
(IV) Flow Control
(V) Encryption.

(I) Authentication -

It means to authenticate user's id. It means that we have to make sure / Authenticate that the person who access the data is the same person who was is actual authorized person.

(II) Access (Control) -

Access Control confirms that which type of data can a user has access, means that a user has permission for it or not, They are of two types Physical and logical

(III) INFERENCE CONTROL -

It is a method of data base security, which is used to get the information from a user which was hidden from

them. Because of that it is also known as hiding Technique.

Inference data Base is very complex. which is a good things, Because The more data base is complex The more of our data is secured.

(4.) FLOW CONTROL-

It is a technique, who may take a look between the conversations of Sender & Receiver. By which we came to know that the flow of data is done correctly or not.

We have to take a supervision on flow of data. By which we can done changes in its speed. with it's receiver gets data speedly.

(5) ENCRYPTION -

It is used to give protection to digital data. For providing security to Encryption data we use password & key.

Encryption provides more security to data. It uses an algorithm, who converts our data into secret codes. By which our data is fully secured. And no one is able to read these codes.

AUTHORIZATION

It is a top level roles deciding technique, Here, we decide the roles like which type of data, by a user access or not.

Authorization have four main qualities-

OBJECTS -

In it objects are those who have need of security, we choose it in a first step of authorization. which we have to secure, Data base is a special data base objects, which we have to secure.

VIEW AS an OBJECT :> In database we can see data in diff - diff forms. If we have to more secure our data, Then we see it in the form

of Sub schemas. In it users can only see / access those data which he difine in users view

→ GRANULARITY -

It is used to increase the security of data. It presents in a file, Record Relation or in data item attributes, The more the object secured, The more security increased.

→ SUBJECT -

In it Subject is an active component, who works on object. Subject is a user who have rights to access some of the data. And some times Users / Application Programmer's groups is known as objects.

ACCESS TYPES :-

We can get access to manipulate users data. In it manipulations options are given.

(I) Read
(II) Insert
(III) Update Control Options.
(IV) Add
(V) Drop
(VI) Alter & Propogate Access Control.

(I) READ → We can only read objects

(II) INSERT → From it we can insert new data, But we cannot perform any kind of changes in previous data.

(III) UPDATE → In it, we can make changes in previously saved data. We but we can't delete it.

(IV) INDEX → With it's help we can make a new indices on previously made indices can be deleted with it's help.

ALTER

(VI) RESOURCES → We can add attributes in relation, and i can also remove it.

(VII) ALTER → RESOURCES → We can make new relation like Table in it.

(VIII) DROP → We can remove / Erase relation by it's medium

Authentication In DBMS →

In this process, we can search the identity of user, Before it can give access to system. If authentication process is successful then it given access to user. It is an good Technique for security, By which user's devices & confidential information is secured.

In simple words we can say that in Authentication, Ask a user "Who are you" in it's reply can a user name, password or finger Print By which it can be identified.

when we upload our credential in computer system. Then the computer System Identify it from there local operating System or authentication Server in their data-Base. If the provided credential is matched. Then, it gives access to user.

UNIT-V

Introduction To Current Trends

Centralized Server Architectures

These are those systems which run on a single computer system and do not interact with other computer systems. Thus, such database systems range from single user databases systems running on personal computers to high performance database systems running on High End Server Systems. In a modern General Purpose Computer System, It is made up of one to several CPUs and many device controllers and these devices are connected to a common bus that provides access to shared memory. To Speed up access to data, CPUs have memories that store local copies of sections of memory. Each device controllers is in charge of a specific type of device, for example a disk drive, audio and video device, or video display. CPUs and device controllers can execute

Simultaneously, competing for memory access. In this, competition for cache memory and memory access is reduced. Since it reduces the number and number of times the CPU needs to access shared memory. There are two ways of using a computer: Single user systems and multiuser systems.

On the other hand, an ideal multiuser system has more disks and more memory and can have more than one CPU and has multiuser operating systems. Database systems designed for single user use generally do not have many of the features provided by multiuser databases. In particular, they do not support concurrency control because it is not required. When only one user is generating updates and in such systems, provisions for crash recovery are either non-existent or very rudimentary.

E.g -

```
import sqlite3

# Connect to the database
conn = sqlite3.connect('ex.db')

# Create a cursor object
cursor = conn.cursor()

# Create a table
cursor.execute('''CREATE TABLE employees (id
INT PRIMARY KEY NOT FULL, name TEXT
NOT NULL, salary REAL);''')

# Commit the changes
conn.commit()

# Close the connection
conn.close()
```

CLIENT SERVER ARCHITECTURE

(A) It is a computing model in which a server hosts the services and provides these services to clients.

(B) In other words, "client Server architecture is a model of computer network in which the client requests for services hosted by the Server and the Server accepts this request and provides Service to the client."

(C) This architecture is also known as "Networked Computing Model" or "Client Server Network".

(D) In this architecture when the client sends a request for service to the Server through the internet, the Server accepts that request and sends the service to the client.

(E) In this architecture the client is often located in a workstation and personal computer while the Server is located to the network.

(F) In this, many clients can be simultaneously access data from a Server, and can as also perform other tasks.

(G) Where the no. of computers is large, client Server architecture is used so that the work can be handled easily.

(H) For example, If we connect the printer to the Server, we can print out files from any workstation.

(I) Examples of client Server architecture are:- Email and world wide web (www)

SERVER OF ARCHITEUTORE

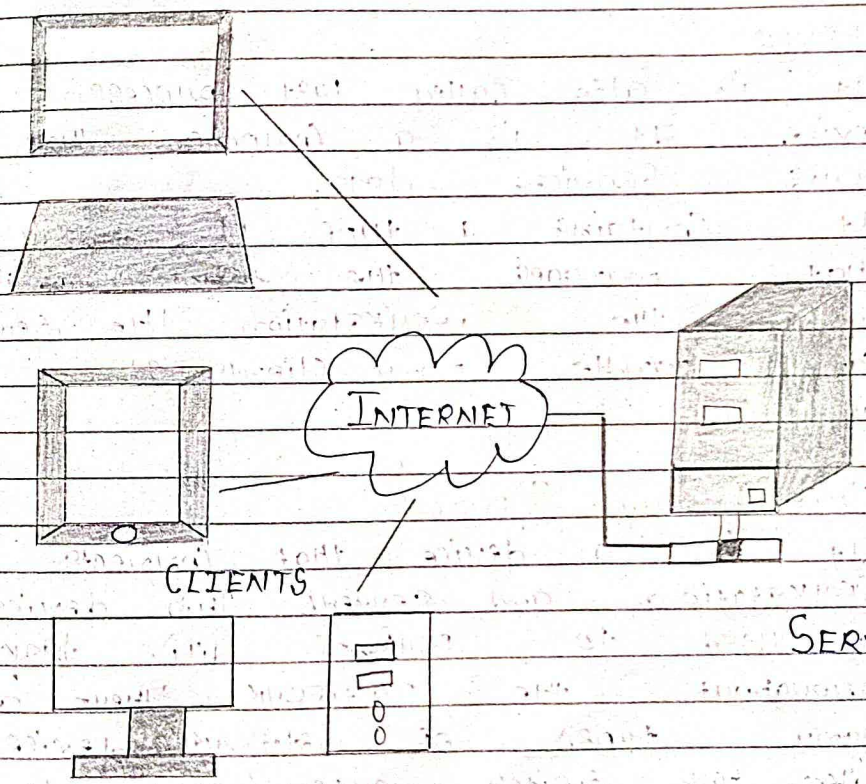


DIAGRAM OF CLIENTS

CLIENT SERVER ARCHITECTURE COMPONENTS

1. WORKSTATIONS
They are also called client computers. It manages files and databases. Diff. types of oper. System are used in workstations, out of which windows operating system is mostly used.
They are used by engineers, architect and graphic designers.

2. SERVER
It is also called fast processing device. It is a computer that stores services data.
It contains a type of memory that manages the requests coming from the workstation. The server can handle many clients at a time.

3. NETWORKING DEVICES
It is a device that connects workstations and servers. This device is used to perform diff. tasks throughout the network. There are many types of network devices like Hub, Switch, router and modem etc.

TYPES OF CLIENT SERVER ARCHITECTURE

There are four types of it which are given below:-

(1) One Tier Architecture
(2) Two Tier Architecture → Unit-I
(3) Three Tier Architecture
(4) N-Tier Architecture

(4) N-Tier Architecture
It is also called multi-tier architecture. This is a shorter form of all 3 architectures, which includes tasks like presentation, application, and management

ADVANTAGES OF CLIENT SERVER ARCHITECTURE

(1) In it, every client does not need to login the processor.

(2) In this the server has better control and resources.

(3) In this, processes and activities can be controlled with the help of a central network.

(4) In this architecture, the user can access the files present in the central storage at any time.

(5) It provides a better user interface for managing files.

(6) Sharing files is easy in this architecture.

(7) Security is very good in this.

DISADVANTAGES OF CLIENT SERVER ARCHITECTURE

(1) In this, if the primary server goes down, the entire architecture is affected.

(2) This architecture contains tools like hardware and software which are quite expensive. This architecture costs more to operate.

(3) It requires a special type of operating system for networking.

(4) Different types of technologies are required to maintain the network

(5) It is a difficult to handle traffic.

Diff. B/w Client Server Architecture & peer-to-peer network

CLIENT SERVER ARCHITECTURE	PEER-TO-PEER NETWORK
(I) It has centralized data management.	(I) It has its own data and application
(II) It's main function is to share info.	(II) It's main function is to maintain the connection.
(III) It is used for small and large networks.	(III) It is used for small networks.
(IV) It is more stable	(IV) These are less stable.
(I) It is expensive Compare to peer-to-peer network	(V) It is cheaper than client-server architecture.

DISTRIBUTED DATABASE :

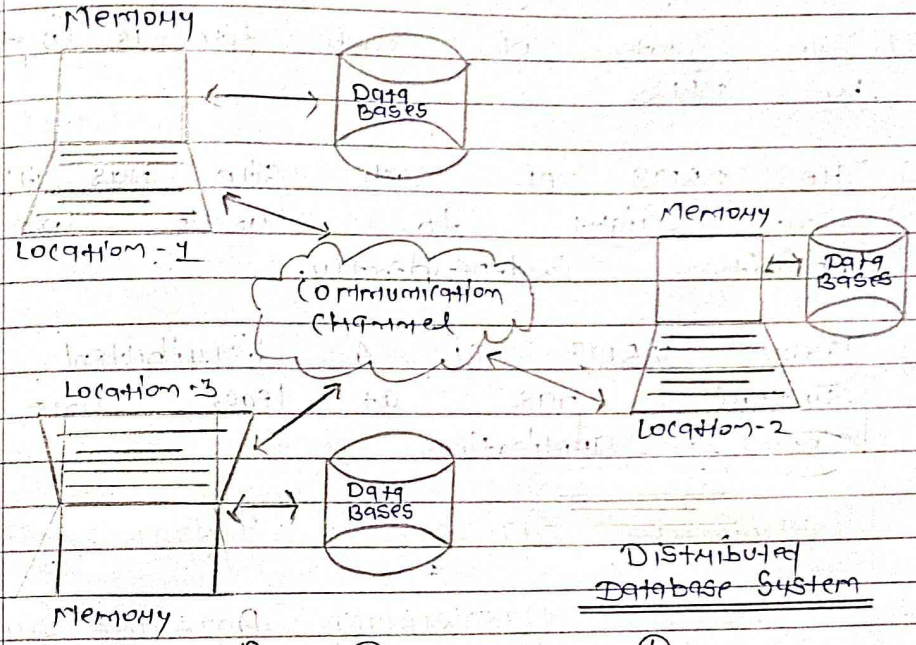
(1) It is a type of database which is not limited to a single system, It is spread across many sites or computers in the network.

(2) In other words, "distributed database is a collection of many interconnected databases which are spread over diff. locations and they communicate with each other through computer networks."

(3) In this, parts of the databases are stored in many physical locations and its processing is distributed among many database nodes.

(4) This database is controlled by distributed database Management System (DDBMS)

(5) The picture given below is a distributed DBMS in which communication channels have been used to communicate two diff. locations, And each system has its own database and memory



Distributed Database System

FEATURES OF DISTRIBUTED DATABASE

(1) The databases in the group are logically interrelated to each other.

(2) In this, data is physically stored in many computers or sites.

(3) It is not a loosely connected file system

(4) In this, all the sites remain connected through a communication network.

(5) The data of each site is controlled by DBMS.

(6) The DBMS of each site has its own right to handle local application independently.

(7) Every DBMS of a distributed system has at least one global application.

TYPES OF DISTRIBUTED DATABASE

(1.) HOMOGENEOUS DISTRIBUTED DATABASE SYSTEM

(1) In homogeneous, all sites use the same DBMS and operating system. In this the sites use similar type of software.

(2) In this, each site keeps information about all other sites and fulfils the users request by cooperating with each other.

(3) In this, data is accessed and modified simultaneously.

(4) It is of two types -
(1) Automomous
(2) Non-Automomous

2. HETEROGENEOUS DISTRIBUTED DATABASE SYSTEM

(1) In it, each site has a diff. DBMS data models and operating system.

(2) In this, each site has different schemas & softwares

(3) There are many types of DBMS in this system such as relational or network, hierarchical or object oriented

(4) Query Processing is very difficult in this because it has diff. schemas

(5) Transaction Processing is very difficult in this because it involves diff. Software.

(6) In this, one site does not have info. about other sites. Due to which they are able very little with each other.

(7) It is of two types - Federated and Un-federated

ADVANTAGE OF DISTRIBUTED DATABASE

(1) It increases reliability, availability and performance.

(2) In this, data can be accessed quickly.

(3) Data processing is faster in this.

(4) It reduces operating costs.

(5) It's interface is user-friendly.

(6) Even if one site is modified or updated, it has no effect on other sites.

(7) There is transparency in it.

(8) It can be updated easily.

DISADVANTAGE OF DISTRIBUTED DATABASE

(1) It's complexity is very high because extra work has to be done by DBA.

(2) It is very difficult to maintain integrity in this.

(3) Increasing complexity means that the cost will also increase.

(4) There is not much security in this.

(5) It requires additional software.

GOALS OF DISTRIBUTED DATABASE SYSTEM

1:- RELIABILITY →

In distributed DBMS, if every one system fails or stops working, the other system completes it's task.

2:- AVAILABILITY →

In this, if one server stops then the other server fulfills the request of the client.

(3) PERFORMANCE →

In this, databases are in different locations due to which data-base is available for each location which is easy to maintain and their performance improves.

OBJECT-ORIENTED DATABASE MODEL

Relational Database Technology has failed to handle the requirements of complex info. Systems. The major problem with Relational Database Systems is that they force application developers to model info. in tables where relationships b/w entities are defined by values.

Object classes used by programming languages are those classes used by object-oriented database management systems (OODBMS). Since models are consistent, therefore, there is no need to transform the object model of the program into a unique object for the database manager. A data model is a collection of mathematically well-defined concepts that help you understand and express the static and dynamic properties of a data intensive application.

A data model is made up of the following.

(1) Static Properties; Like- Objects, Attributes and Relationships

(2) Integrity rules applied to objects and operations.

(3) Dynamic Properties; Such as operations, by and rules that define the new database state based on applied state changes.

Object Oriented databases can model these 3 components in the database itself, providing a unifying paradigm that allows you to integrate Databases before.

Object-Oriented Databases directly support static properties and integrity Rules. But are relied on Applications to define the dynamic properties of the model.

Object Oriented Databases provide a Unifying Paradigm that allows you to integrate the three aspects of data modeling and apply them consistently to all users of the databases. Object Oriented model represents an entity as a class. A class represents both the attributes and behaviour of a particular entity. Object-oriented databases manage objects which are abstract data types. Object-Oriented Database Management System (OODBMS) is suitable for managing multimedia applications and data containing such complex relationships.

DATA MINING

It is also called data or Knowledge discovery. It is the process of searching for small data from a large set of data. Traditional Statistics artificial intelligence and computer graphics are used in this process.

In other words, "Data mining is a useful technique using which companies extract important information from large sets of data".

In it, data mining tools are used to analyze data. These tools are very powerful.

Data Mining has the following goals:-

1:- Explanatory - In this observed incident or situation is explained.

2:- Confirmatory - In this, hypotheses from possibilities are confirmed.

3:- Analyzatory - In this, new data is analyzed so that positive feedback can be given.

By using data mining, hidden patterns and useful data are discovered and them decision making is done on the basis of these patterns and data and by using the process of data mining, Organizations solve the problems arising in business.

Advantage Of Data Mining -

It's benefits are as follows-

(1) Through the technique of data mining, the company obtains information based on knowledge.

(2) Through this, organizations improve their production and operation.

(3) Data mining is cost effective compared to other statical data applications, that is, it saves cost.

(4) Through this, decisions can be taken easily.

(5) It is very easy to implement it in new systems.

(6) It's speed is very fast due to which big data can be analyzed in less time.

(7) It finds profitable customer easily, which makes it easier to sell the product and also improves the relationship with the customer.

Disadvantage Of Data Mining -

It's disadvantages are given below -

(1) It's major disadvantage is that there is no security and privacy of data in it. In this, all the data is collected such as social media messages, photos etc. This destroys people's privacy.

(2) The data collected through data mining is mostly incomplete.

(3) In this, irrelevant (useless) data is also collected.

Characteristics Of Data Mining -

It's features are as follows -

(1) It makes future predictions. This means that it predicts future events.

(2) It focuses on large datasets and databases.

(3) In this, the prediction of patterns is automatic and it is based on behavior analysis.

(4) It creates useful information.

TYPES OF DATA MINING -

There are two types of data mining analysis, which are as follows :-

(1) Predictive Data Mining Analysis.

(2) Descriptive Data Mining Analysis.

Predictive Data Mining Analysis - It predicts future events. It is of four events :-

A. Classification Analysis
B. Regression Analysis
C. Time Serious Analysis
D. Prediction Analysis

Descriptive Data Mining Analysis - It is used to convert data into useful information. These are also four type of this.

A. Clustering Analysis
B. Summarization Analysis
C. Association Rules Analysis
D. Sequence Discovery Analysis

APPLICATIONS OF DATA MINING -

It is used in many places. Its applications are as follows -

(1) IN THE FIELD OF HEALTHCARE - It is used to find out about the patient's disease. It gives info. about such hospitals where the patient can be treated at less money and in less time.

(2) IN THE FIELD OF MARKETING - Customer behaviour is detected through data mining. In this it is seen that if the customer has purchased something similar then what other item will be buy along with it.

(3) IN THE FIELD OF EDUCATION - Student's result is predicted using data mining. It also tells how to teach a student and what to teach.

(4) IN THE DETECTING FRAUD - A lot of frauds are happening nowadays. Due to which lakhs of people's money gets wasted. Data mining helps in avoiding this

DATA VISUALIZATION

You must be aware that when we see a picture etc we can understand it very fast and learn about it. And then we remember everything about the picture for a long time. THIS is the reason why when we see any graph, picture etc, It is very easy for us to understand it.

We will tell you that data visualization is done on a very large scale. As difficult and strange as the word data visualization may sound. It is used more in our daily lives.

It is used every time and everywhere such as, business main data visualization is used for reporting and marketing. Non profitable organizations use data visualization to tell stories and more. Reporters use visualization to show news and events to others. Data visualization makes any work more professional or easier to understand.

PRINCIPLES OF DATA VISUALIZATION -

The three main principles of data visualization are -

(1) VISUALIZATION - Displays data in visual form.

(2) INSIGHTS - Arranging data in the right way so that there is no wrong output of data for the user.

(3) SHARING - The data can be easily understood by the user and the user can easily share that data with any other person.

TYPES OF DATA VISUALIZATION

There are many diff. types of data visualization because people have come up with diff. ways of representing data as per their convenience. So that they can become much easier to understand, data visualization is divided into the following categories.

DATA VISUALIZATION

You must be aware that when you see a picture etc we can understand it very fast and learn about it. And then we remember everything about the picture for a long time. This is the reason why when we see any graph, picture etc, It is very easy for us to understand it.

We will tell you that data visualization is done on a very large scale. As difficult and strange as the word data visualization may sound, It is used more in our daily lives.

It is used every time and everywhere such as, business main data visualization is used for reporting and making profitable organizations use data visualization to tell stories and more. Reporters use visualization to share events to others. Data visualization makes any work more professional or easier to understand.

PRINCIPLES OF DATA VISUALIZATION:

The three main principles of data visualization are-

(1) VISUALIZATION - Displays data in visual form.

(2) INSIGHTS - Arranging data in the right way so that there is no wrong output of data for the user.

(3) SHARING - The data can be easily understood by the user and the user can easily share that data with any other person.

TYPES OF DATA VISUALIZATION

There are many diff. types of data visualization because people have come up with diff. ways of representing data as per their convenience. So that they can become much easier to understand, data visualization is divided into the following categories.

(A) Infographics
(B) Chart
(C) Diagram
(D) Map

CHART-

A data chart is a type of diagram or graph, that organizes and represents a set of numerical or qualitative data. Maps that are adorned with extra info. (map surround) for a specific purpose are often known as charts, such as a nautical chart or aeronautical chart, typically spread over several map sheets. Line bars, dots, slices and items are used into represes. data within a chart.

By looking all these charts, you must have understood one thing that any diff. data is shown by diff. charts. Some general types of chart are-