"Jai Shree Ram" ...

06/June (5:44)

DBMS – Data Base Management System.

↳ A DBMS is a collection of interrelated data and a set of Programs to access those data

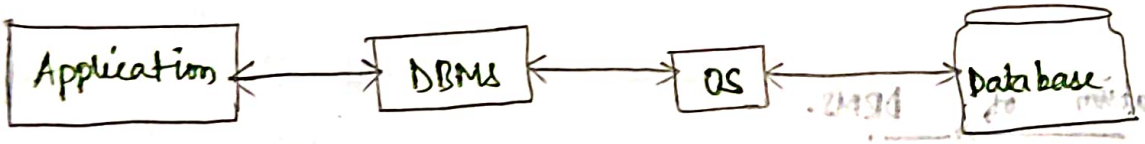
↳ DBMS is used to organize the data in the form of a table, schema, view, report, etc.

↳ Primary goal of DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.

↳ DBMS is a collection of related information stored so that it is available to many users for diff. purpose.

↳ DBMS is a collection of Program that enables users to create and maintain the database

Fig



↳ DBMS can also be define as an interface bw the application program and OS to Access and Manipulate that database.

↳ DBMS is a software which is used to manage the database.

For eg: MySQL, Oracle, etc are a very popular commercial database which is used in different application.

When large amount of data is present, then DBMS comes into picture.

Characteristics of DBMS.

↳ It uses a digital ref... to store and manage ...

↳ It can provide a ... the process that ...

↳ DBMS contains autom... procedures.

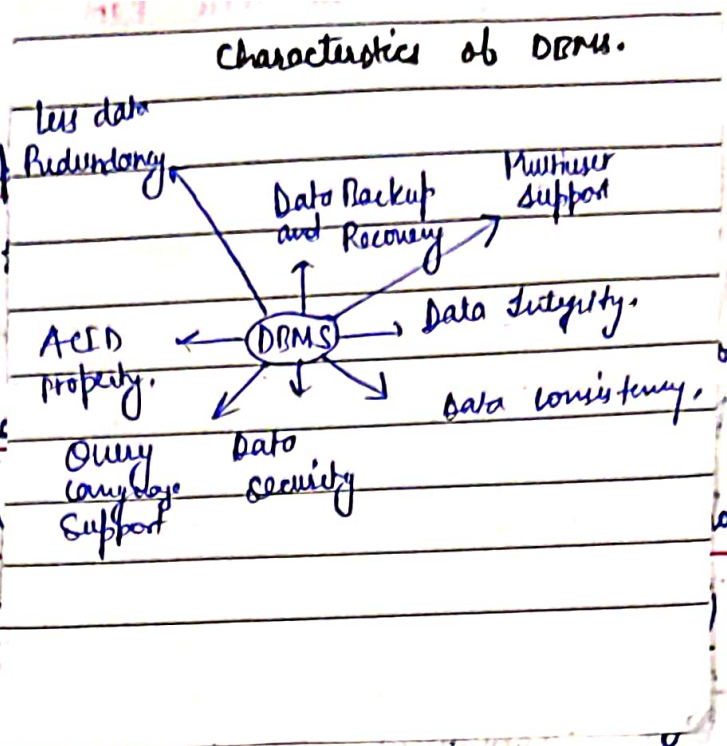
↳ It contains ACID (Atom...) properties which maint... state in case of failure.

↳ It can reduce the complex relationship b/w data.

↳ It is used to support manipulation and processing of data.

↳ It Provides security of data.

↳ It can view the DB from diff. viewpoint according to the requirement of the user.



* Application of DBMS.

(i) Banking :- For maintaining customer information, accounts, loans and banking transaction.

(ii) Universities :- For maintaining student records, course registration and grades.

(iii) Railway Reservation :- For checking the availability of Reservation in diff. trains, tickets, etc.

(iv) Airlines :- For reservation and schedule information.

(v) Telecommunication – Call made, monthly bills, etc.

(vi) Finance – Holiday, Sales

(vii) Sales. → Purchase information.

Advantages of DBMS :-

(i) control DB redundancy :- (Repeatation) 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 user of an organization can share data among multiple users.

(iii) Easily Maintainence :- It can be easily maintainable due to centralized nature of database system.

(iv) Reduce time :- It reduces development time and maintance need.

(v) Backup :- It provide backup and recovery subsystem which make automatic backup of data from hardware and software failure and retore the data is Required.

(vi) Multiple user Interface :- It provides diff types of user Interface like GUI, Application program Interface (API).

* Disadvantages of DBMS :-

(i) Cost of Hardware and software :- It Requires a high speed of data processor and large memory size to run DBMS

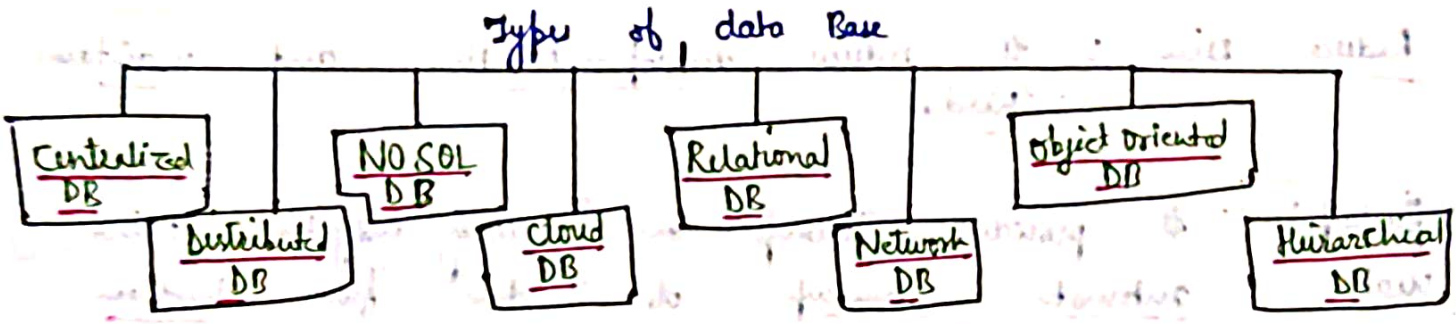
(ii) Size :- It occupies a large space of disks and large memory to run them efficiently.

(iii) Complexity :- DB system creats additional complexity and Requirements.

Higher impact of failure :- Failure is highly impacted the db because in most of the organisation all the data stored in a single DB and if the DB is damaged due to electric failure or DB corruption then the data may be lost Forever.

* Types of database :-

↳ There are various type of DB used for storing diff varieties of data.



(1) Centralized Data Base

↳ It is a type of DB that stores data of a centralized database system. (CDBS)

↳ It comforts the users to access the stored data from different location through several application.

↳ This App. contains the authentication process to let users access data security. (security access karega data ko).

↳ An example of a centralized database can be central library that carries a central database of each library in a college / university

↳ All data is stored in one central place.

(ii) Distributed database :- ↳ data stored in Multiple Physical location.

↳ In distributed database data is distributed among diff. DB system of an organisation.

↳ These db system are connected via communication links such links help the end users to access the data easily.

↳ It is divided into two sub parts

distributed database

(I) Homogenous database
(All nodes use same DBMS and are aware of each other.)

(II) Heterogenous database.
(Different nodes may use diff. DBMS and may not be aware of Each other.)

(iii) Relational database

↳ It stores data in the form of rows (Tuple) and columns (attributes) and together forms a table (Relation)

↳ A RDB use SQL for storing, manipulating as well as maintaing the data.

↳ Each table in the database carries a key that make the data unique from others

↳ Eg of RDB are MySQL, Microsoft SQL server, Oracle, etc.

(IV) No SQL database ↳ It doesn't use a fixed schema. (fixed structure).

↳ No SQL / 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 store data not only in tabular form but in several diff. ways.

It is also divided into four sub-parts

(i) Key value storage → Redis, Dynamo DB

(ii) Document oriented database → Mango DB, Couch DB

(iii) Graph database → HBase, Apache Cassandra

(iv) Wide-column store → Neo4j, ArangoDB

(v) Cloud database

↳ It is a type of database where data is stored in a virtual environment and executes over the cloud computing platform.

↳ It provides users with various cloud computing services (Daas, Paas, Iaas, etc) for accessing the database.

↳ Some example of cloud dB such as Amazon web services (AWS), microsoft Azure, Google cloud SQL, etc.

(vi) Object-oriented database:

↳ This type of dB that uses the object base data model approach for storing data in the database system.

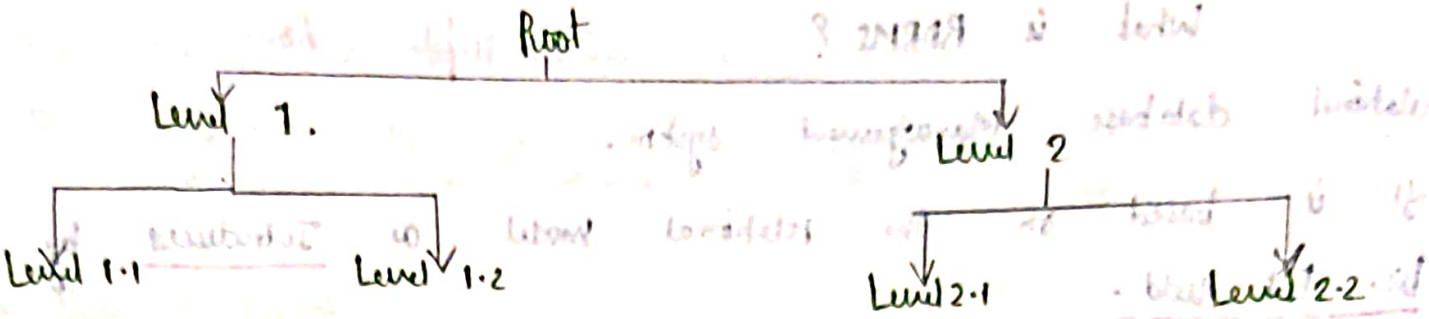
↳ The data is represented and stored as objects which are similar to the objects used in the object oriented Programming language. (OOPS)

↳ Example :- db4o (database for objects), object DB, versant object database

(vii) Hierarchical databases

↳ It is a type of database that stores data in the form of parent-children relationship node.

↳ It organizes data in 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.

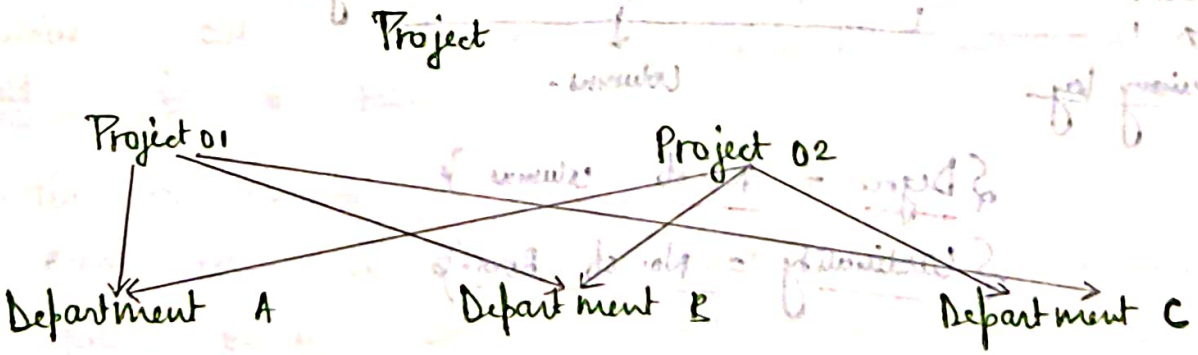
↳ Examples :- IBM Information Management System (IMS), XML DB, Windows Registry

(viii) Network database

↳ It is the database that typically follows the network data Model.

↳ In this representation of data is in the form of node connected via links blw them.

↳ It allows each record to have multiple children and parent node to form a generalised graph structure.



What is RDBMS ?

tuple - Row

↳ Relational database Management system.

↳ It is based on the relational model as Introduced by Dr. EF codd.

↳ It stores data in the form of Related tables.

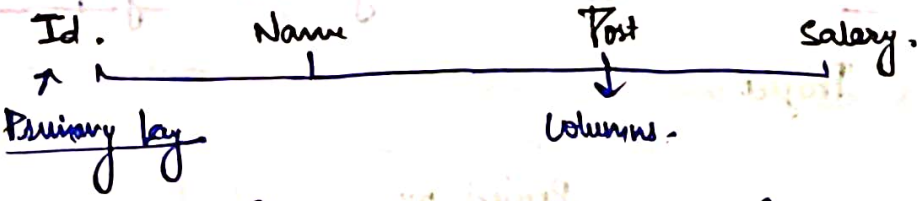
↳ An important Feature of RDBMS is that a single database can be spread across several tables.

↳ All modern DBMS like SQL, MS SQL Server, IBM DB2, ORACLE My-SQL and Microsoft Access are based on RDBMS.

↳ A Relational database is the most commonly used DB. It contains several tables and each table has its primary key.

↳ Due to a collection of an organised set of tables data can be easily accessed in RDBMS.

↳ Everything in a Relational DB is stored in the form of Relations (Tables)



{ Degree = No. of columns }

{ Cardinality = No. of Rows. }

* Properties of Relational tables

↳ Value are atomic (individual value).

↳ Column value are of same kind

↳ Each Row is unique

↳ Each column has a unique name.

↳ The sequence of rows & columns is insignificant.

↳ Row or Record
  ↳ A row of a table is also called a record or tuple.
↳ Row contains the specific information of each entry in the table.
↳ It is the horizontal entity in the table.

↳ Properties of a Row
  ↳ No two Rows are identical to each other in all their entries.
  ↳ All tuples/Rows of the relation have the same format and same no. of entries.
  ↳ The order of the Rows is irrelevant. They are identified by their content, not by their Position.

↳ Columns / Attributes / fields
  ↳ A column is a vertical entity in the table which contains all information associated with a specific field in a table.

* Properties of columns
  ↳ Every column of a relation must have a name.
  ↳ Null values are permitted for the columns
  ↳ Default values can be specified for an columns Automatically inserted if no other values is specified for an column.
  ↳ Column that uniquely identify each Row of a relation are the Primary Key.

Data item / cell

↳ The smallest unit of data in the table is the individual data item.

↳ It is stored at the intersection of row and column.

Degree

↳ The total no. of columns that comprise a relation is known as the degree of the table.

Cardinality

↳ The total no. of rows at any one time in a relation is known as the table's cardinality.

↳ The relation whose cardinality is zero is called an empty table.

Domain

↳ The domain refers to the possible values each column can contain.

↳ It can be specified using standard datatypes such as Integer, floating number, etc.

* Dr. E.F Codd's Rules for RDBMS

↳ Dr. E.F Codd is an IBM researcher who first developed the Relational data Model in 1970.

↳ In 1985 Dr. Codd published a list of 12 rules that define an ideal RDB and has provided a guideline for the design of all Relational dB.

Rule 1 The Information Rule

↳ This Rule simply requires that all data should be presented in table form. This is the basis of Relational Model.

Rule 2 Guaranteed Access Rule

↳ every single data element (value) is guaranteed to be accessible logically with a combination of table-name, Primary-key (row value) and Attribute - name (column - name)

Rule 3 Systematic treatment of null values

↳ The null values in a dB must be given a systematic and uniform treatment. This is very important Rule because a null can be interpreted as one of the following -
data is missing, data is not known or data is not Applicable.

Rule 4 Active online Catalog

↳ The structure description of the entire dB must be stored in an online Catalog, known as

- data dictionary, which can be accessed by authorized users
  - Users can use the same Query Language to access the catalog which they use to access the DB itself.

Rule 5 Comprehensive data sub language Rule.

- A DB can only be accessed using a language having linear syntax that support data difination, data manipulation and transaction management operations.
- All commercial RDB use form of SQL as their supported language.

Rule 6 View updating Rule

- Data can be presented in different logical combination called views.
- Each view should support the same full Range of data manipulation that has direct Access to a table available.

Rule 7 High Level Insert, update and Delete.

- A DB must support high level insertion, updation and deletion. This must not be limited to a single row, that is it must also support union, intersection and minus operation to yield sets of data records

Rule 8 Physical data Independence

- The data stored in DB must be independent of the application that Access the databases.

↳ Any change in the Physical structure of a dB must not have any impact on "How the data is being Accessed by external application".

Rule 9 Logical data Independence.

↳ The logical data in a dB must be independent of its user's view (App.). Any change in logical data must not affect the App. use it.

Rule 10 Integrity Independence.

↳ The dB language (like SQL) should support constraints on user input that maintain dB integrity.

↳ No component of a Primary key can have a null value

↳ If a foreign key is defined in one table, any value in it must exist as a Primary key in another table.

Rule 11 Distribution Independence

↳ The end user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only. This Rule has been regarded as the foundation of distributed DBS.

Rule 12 Non-subversion Rule

↳ There should be no way to modify the DB structure other than through the multiple Row DB Language (SQL) most DB today support Administrative tools that allows some direct manipulation of data structures.

* DBMS

↳ Relationship blw two tables or files are maintained Programatically.

↳ DBMS App. store data as files

↳ Normalization is not present in DBMS

↳ It does not support client / server Architecture.

↳ It does not apply any security with regards to data manipulation

↳ It uses file system to store data, so there will be no relation blw the tables.

RDBMS.

↳ Relationship blw two tables and files can be specified at the time of table creation.

↳ RDBMS App stores data in tabular form.

↳ Normalization is present

↳ Most of it support client / server Architecture.

↳ RDBMS defines the integrity constraints for the purpose of ACID.

↳ data values are stored in the form of tables so a relationship blw these data values will be stored in the form of a table as well

- It does not support distributed database.
- It is meant to be for small organisation and deal with small data. it support single user.
- It may satisfy less than (7) to (8) Rules of Dr. E.F Codd.
- Ex:- File system, XML, etc.

- It support distributed database.
- It is designed to handle large amount of data, it supports multiple users.
- It satisfies all (12) rules of Dr. E.F codd.
- Ex:- MySQL, Oracle, SQL etc, Postgrace.SQL.

DBMS

- It is a collection of data. In DBMS user is not Required to write the Procedures.
- Due to centralized approach, data sharing is easy.
- It gives an abstract view of data that hide the details.
- Data Inconsistency does not exist.

File Processing System.

- It is a collection of data. In user has to write the Procedures for managing the DB.
- Data is distributed in many files and it may be of different format, so it's not easy to share data.
- It provides the details of the data Representation and storage of data. (external view and internal view) of data
- Data Inconsistency exists.

Replication

- Data Redundancy Problem is not found.
- Data Redundancy Problem is found.
- database structure is complex to design.
- The file system has a simple structure.
- Accessing DB is easier
- Accessing DB is difficult.
- Data independence exists and it can be of two types
  (i) logical data Independence
  (ii) Physical data Independence.

  Agar logical level main kuch update karoge to uska effect physical level main nahi hoga and vice versa.
- In File system approach there doesn't exists data Independence.
- Integrity constraints are easy to Apply.
- Integrity constraints are difficult to implement in File system.
- In the DB approach 3 types of data Models exists
  - Hierarchical data Model
  - Network data Model
  - Relational data Model
- In the file system approach there is no concept of data Model exists.

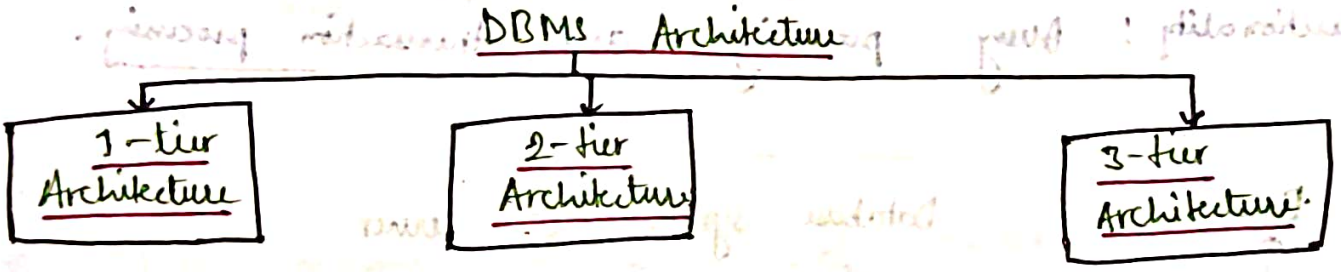
Application Architecture of DBMS

↳ The DBMS design depends upon its architecture

↳ The basic client / server Architecture is used to deal with a large no. of PCs, web servers, database servers and other components that are connected with networks.

↳ DBMS architecture depends upon how users are connected to the db to get their request done.

Types of DBMS.



Database architecture can be seen as a single tier or multitier But logically, database Architecture is of two types like 2-tier Architecture and 3-tier Architecture.

1-tier Architecture

↳ In this Architecture, the db is directly available to the user. It means the user can directly sit on the DBMS and use it.

↳ Any changes done here will directly be done on the db itself. It doesn't provide a handy tool for end users.

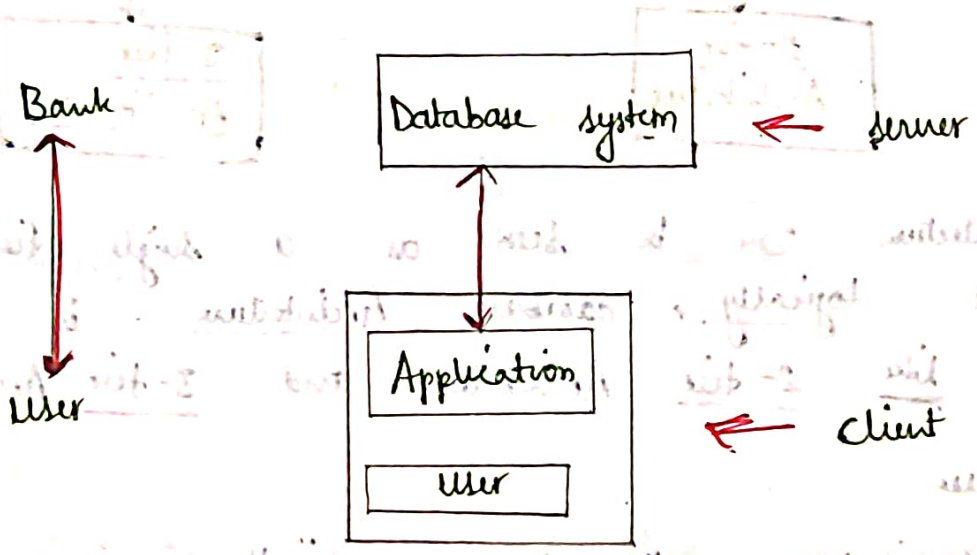
↳ The 1-tier Architecture is used for development of the local App., where programmers can directly communicate with the db for the quick response.

2-tier Architecture

↳ The 2-tier Architecture is same as basic client/server. In the two-tier architecture, app. on the client end can directly communicate with the db at the server side. For this interaction, API's like ODBC, JDBC are used.

↳ The user Interface and application program are run on the client-side.

↳ The server side is responsible to provide the functionality: Query processing and transaction processing.



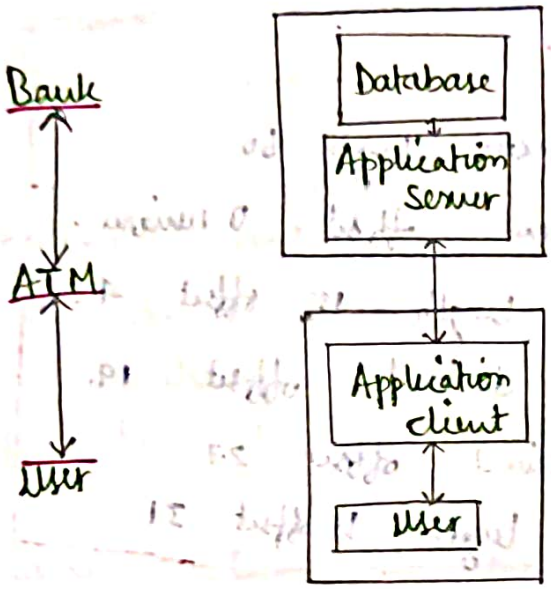
3-tier Architecture

↳ The 3-tier architecture contains another layer b/w the client and server.

↳ In this Architecture, client can't directly communicate with the server.

↳ The app. on the client-end interacts with an application server with further communicates with db system.

↳ The 3-tier architecture is used in case of large web application.

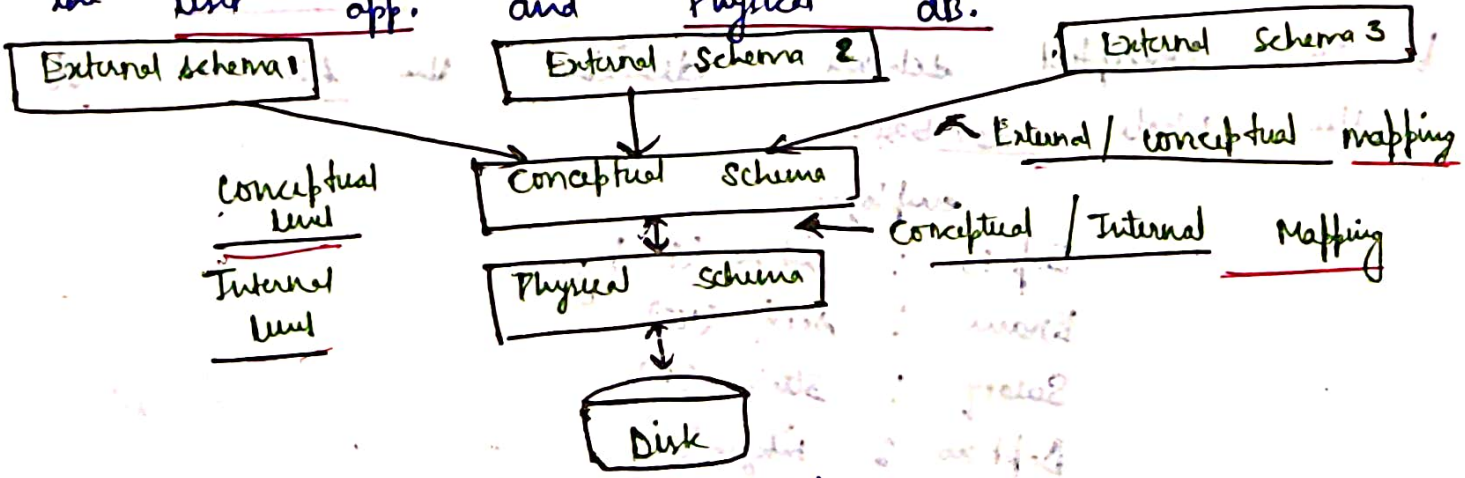


* Three schema Architecture of DBMS

↳ The overall design of the database is called the db schema.

↳ The tree schema Architecture is also called ANSI/SPARC (American National Standards Institute) (Standards Planning and Requirements Committee) architecture or three level Architecture.

↳ The tree schema Architecture is also used to separate the user app. and Physical db.



Three schema Architecture.

1 Internal Level / Internal View

↳ The internal level has an state Internal schema which describes the Physical storage of the DB.

Internal view

Stored employee record length 60
Emp no. : 4 decimal offset 0 unique.
Ename : String length 15 offset 4.
Salary : 8, 2 decimal offset 19.
Dept no : 4 Decimal offset 27
Post : string length 15 offset 31

↳ The internal schema is also known as Physical schema.

↳ It use Physical data model. It is used to define that "how the data will be stored in a block."

↳ The Physical level is used to describe complex low-level data structure in detail.

2 conceptual level / logical level.

↳ The conceptual schema describes the design of a DB at the conceptual level.

↳ Conceptual level is also known as logical level.

↳ The conceptual schema describes the structure of the whole database.

employee

Emp no. : integer (4) key.
Ename : string (15)
Salary : string (8)
Dept no : integer (4)
Post : string (15)