

# ASSINGNMENT-2

## DATABASE MANAGEMENT SYSTEM

<b>Assignment Date</b>	<b>10 OCTOBER 2022</b>
<b>Student Name</b>	<b>ROSARIO FUEDAL.E</b>
<b>Student Roll Number</b>	<b>810419205035</b>
<b>Maximum Marks</b>	<b>2 MARKS</b>

# INTRODUCTION OF DBMS (DATABASE MANAGEMENT SYSTEM)

- A database is a collection of inter-related data which helps in the efficient retrieval, insertion, and deletion of data from the database and organizes the data in the form of tables, views, schemas, reports, etc. For Example, a university database organizes the data about students, faculty, admin staff, etc. which helps in the efficient retrieval, insertion, and deletion of data from it. **DDL** is the short name for Data Definition Language, which deals with database schemas and descriptions, of how the data should reside in the database.

# PURPOSE OF DBMS

## **1. Data redundancy and inconsistency**

- Same information may be duplicated in several places.
- All copies may not be updated properly.

## **2. Difficulty in new program to carry out each new task**

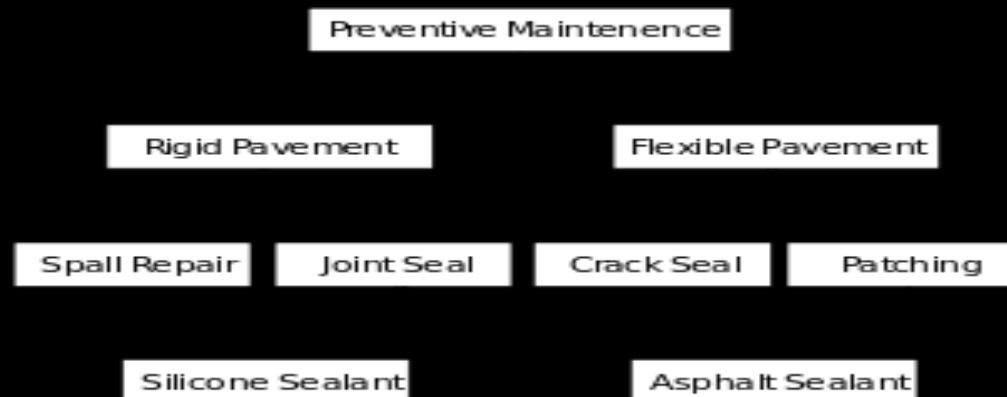
## **3. Data isolation**

- Data in different formats.
- Difficult to write new application programs.
- files and formats

# NETWORK MODEL


- The popularity of the network data model coincided with the popularity of the hierarchical data model. Some data were more naturally modeled with more than one parent per child.
- So, the network model permitted the modeling of many-to-many relationships in data. In 1971, the Conference on Data Systems Languages (CODASYL) formally defined the network model.


## Network Model




# BRIEF HISTORY

- Early 1960s: first general purpose database by Charles Bachman from GE. Used the network data model.
- Late 1960s: IBM developed Information Management System (IMS). Used the hierarchical data model. Led to SABRE, the airline reservation system developed by AA and IBM. Still in use today.

- 
- **Data Security:** Only authorized users are allowed to access the data in DBMS. Also, data can be encrypted by DBMS which makes it secure.
  - **Concurrent access to data:** Data can be accessed concurrently by different users at same time in DBMS.
  - **Backup and Recovery mechanism:** DBMS backup and recovery mechanism helps to avoid data loss and data inconsistency in case of catastrophic failures.

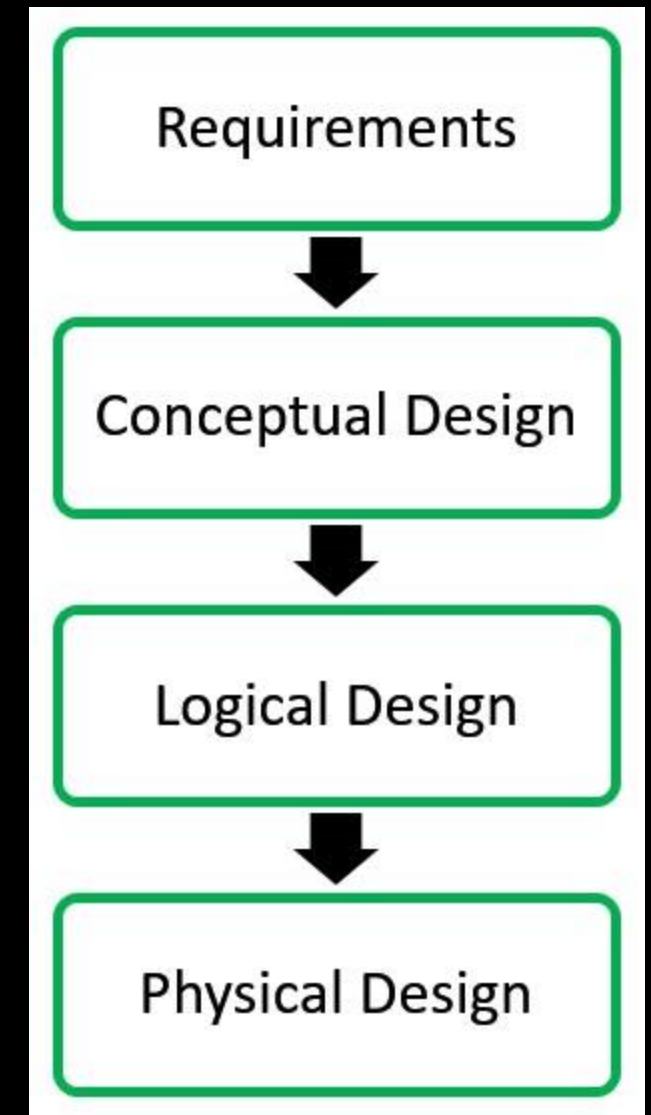
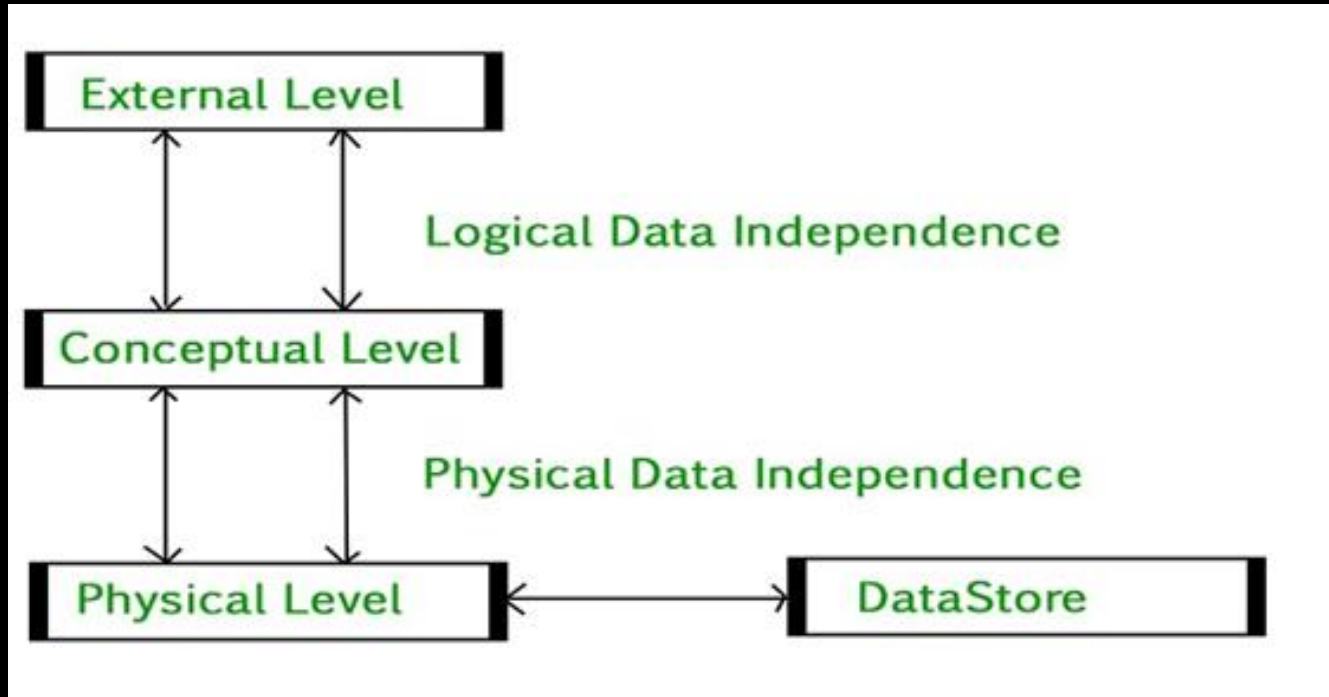
- 
- 1970: Edgar Code of IBM developed the relational data model. Led to several DBMS based on relational model, as well as important theoretical results. Code wins Turing award.
  - 1980s: relational model dominant. SQL standard.
  - Late 1980s, 1990s: DBMS vendors extend systems, allowing more complex data types (images, text).

- 
- . **CREATE:** to create a database and its objects like (table, index, views, store procedure, function, and triggers)
  - . **ALTER:** alters the structure of the existing database
  - . **DROP:** delete objects from the database
  - . **TRUNCATE:** remove all records from a table, including all spaces allocated for the records are removed
  - . **COMMENT:** add comments to the data dictionary
  - . **RENAME:** rename an object



# INTRODUCTION OF 3-TIER ARCHITECTURE IN DBMS

- **Physical Level:** At the physical level, the information about the location of database objects in the data store is kept. Various users of DBMS are unaware of the locations of these objects.
- **Conceptual Level:** At conceptual level, data is represented in the form of various database tables. For Example, STUDENT database may contain STUDENT and COURSE tables which will be visible to users but users are unaware of their storage. Also referred as logical schema.
- **External Level:** An external level specifies a view of the data in terms of conceptual level tables. Each external level view is used to cater to the needs of a particular category of users.



# DBMS ARCHITECTURE 1-LEVEL, 2-LEVEL, 3-LEVEL

- A Database store a lot of critical information to access data quickly and securely. Hence it is important to select a correct Architecture for efficient data management.
- Types of DBMS Architecture:
  - 1- Tier Architecture
  - 2- Tier Architecture
  - 3- Tier Architecture

# ONE- TIER ARCHITECTURE

- In One-Tier Architecture the database is directly available to the user, the user can directly sit on the DBMS and use it i.e.; the client, server, and the Database are all present on the same machine. For Example- To learn SQL we set up an SQL server and the database on the local system. This enables us to directly interact with the relational database and execute operations. The industry won't use this architecture they logically go for 2-Tier and 3-Tier Architecture.

# TWO-TIER ARCHITECTURE

- The two-tier architecture is similar to a basic **client-server** model. The application at the client end directly communicates with the database at the server side. APIs like ODBC and JDBC are used for this interaction. The server side is responsible for providing query processing and transaction management functionalities. On the client side, the user interfaces and application programs are run. The application on the client side establishes a connection with the server side in order to communicate with the DBMS.  
An advantage of this type is that maintenance and understanding are easier, and compatible with existing systems. However, this model gives poor performance when there are a large number of users.

# THREE TIER ARCHITECTURE

- In this type, there is another layer between the client and the server. The client does not directly communicate with the server. Instead, it interacts with an application server which further communicates with the database system and then the query processing and transaction management takes place. This intermediate layer acts as a medium for the exchange of partially processed data between server and client. This type of architecture is used in the case of large web applications.

# ADVANTAGES

- **Enhanced scalability** due to distributed deployment of application servers. Now, individual connections need not be made between client and server.
- **Data Integrity** is maintained. Since there is a middle layer between the client and the server, data corruption can be avoided/removed.
- **Security** is improved. This type of model prevents direct interaction of the client with the server thereby reducing access to unauthorized data.

# DISADVANTAGES

- Increased complexity of implementation and communication. It becomes difficult for this sort of interaction to take place due to the presence of middle layers.