Unit 1. Introduction

1.1. History, database system and its applications

1.2. Characteristics of DBMS

1.3. Application architecture (one tier, two tier, n-tier)

1.4. Data abstraction and Independence

1.5. Schemas and Instances

1.6. Database Manager and users

Datum:

- **Definition:** A datum is a single piece of raw data. It is an individual element of information that by itself may not convey much meaning.
- **Example:**
  - "42" is a datum representing a number.
  - "September 14, 2024" is a datum representing a specific date.

Data:

- **Definition:** Data is a collection of multiple pieces of data (or datums) that are often unprocessed. Data can be quantitative or qualitative and consists of various units of measurement or observation.
- **Example:**
  - A list of numbers like [42, 67, 89] represents data.
  - A database table with multiple rows of entries such as names, ages, and addresses.

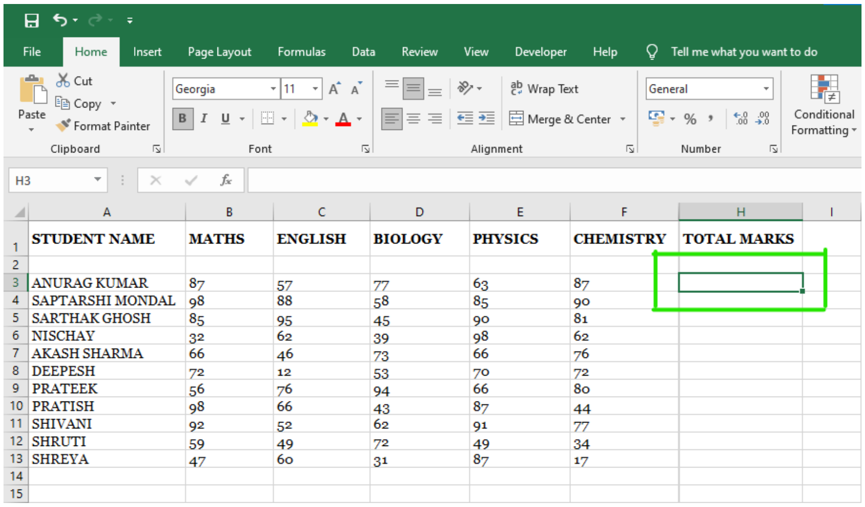
Information:

- **Definition:** Information is what you get when data is processed, organized, or interpreted to provide meaning and context. It helps in decision-making and understanding patterns or insights.
- Converting data into information involves organizing and analyzing it to make it meaningful.
- **Example:**
  - A sales report that shows trends and insights over a period, derived from raw sales data.
  - A summary of customer reviews that highlights common themes and sentiment.

History, Database System, and Its Applications

History:

- **Early Beginnings:** Databases have their roots in early data management systems like file systems and flat files. In the 1960s, the concept of databases started to evolve with hierarchical and network databases.



H3

	A	B	C	D	E	F	H	I
1	STUDENT NAME	MATHS	ENGLISH	BIOLOGY	PHYSICS	CHEMISTRY	TOTAL MARKS	
2								
3	ANURAG KUMAR	87	57	77	63	87		
4	SAPTARSHI MONDAL	98	88	58	85	90		
5	SARTHAK GHOSH	85	95	45	90	81		
6	NISCHAY	32	62	39	98	62		
7	AKASH SHARMA	66	46	73	66	76		
8	DEEPESH	72	12	53	70	72		
9	PRATEEK	56	76	94	66	80		
10	PRATISH	98	66	43	87	44		
11	SHIVANI	92	52	62	91	77		
12	SHRUTI	59	49	72	49	34		
13	SHREYA	47	60	31	87	17		
14								
15								

- **Relational Model:** Introduced by E.F. Codd in 1970, the relational model used tables (relations) to represent data and relationships. This became the foundation for modern relational database management systems (RDBMS).

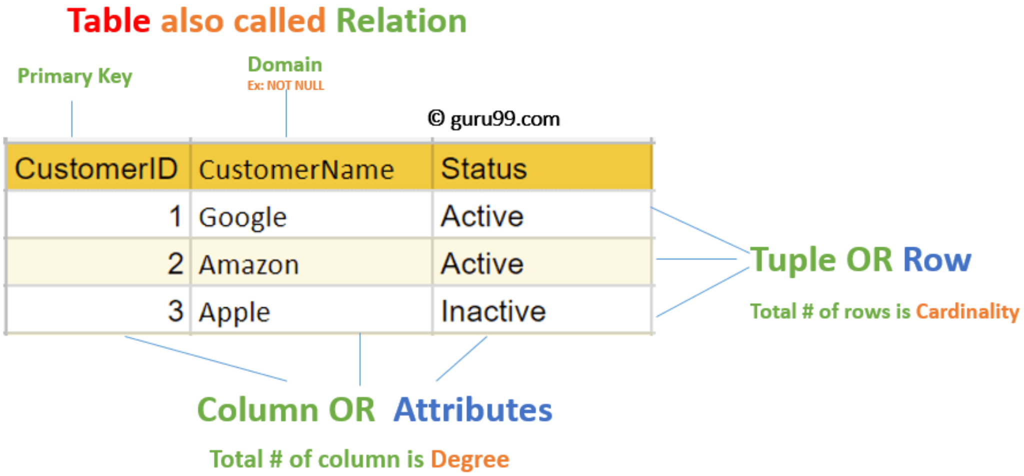
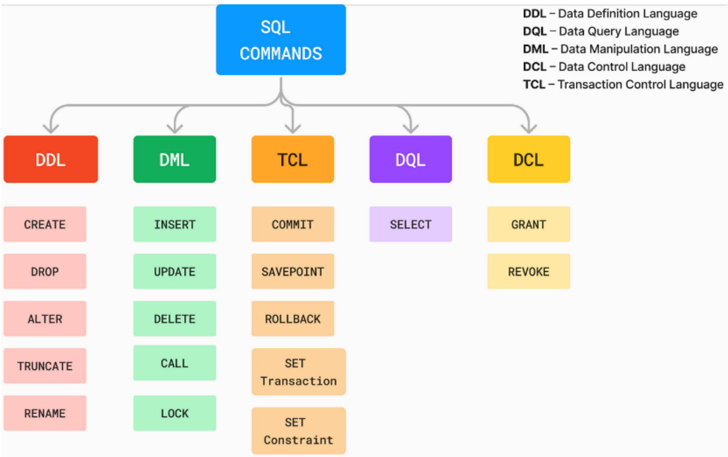
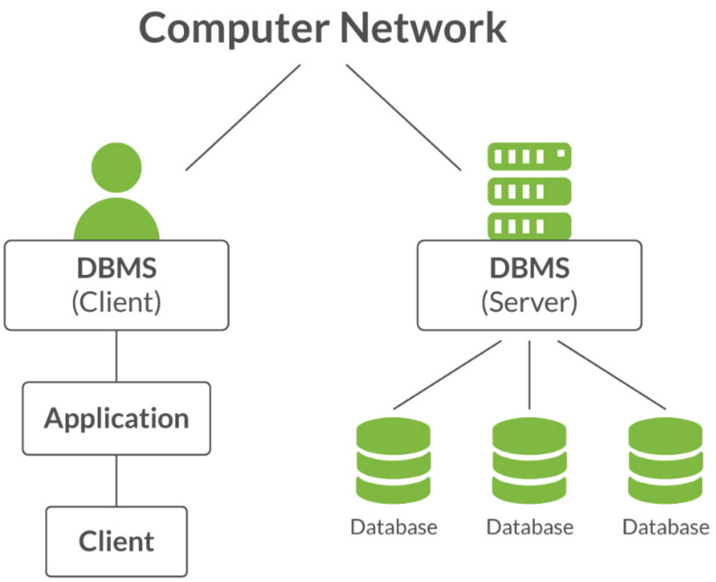


Table also called Relation
Primary Key
Domain
Ex: NOT NULL
© guru99.com
Tuple OR Row
Total # of rows is Cardinality
Column OR Attributes
Total # of column is Degree

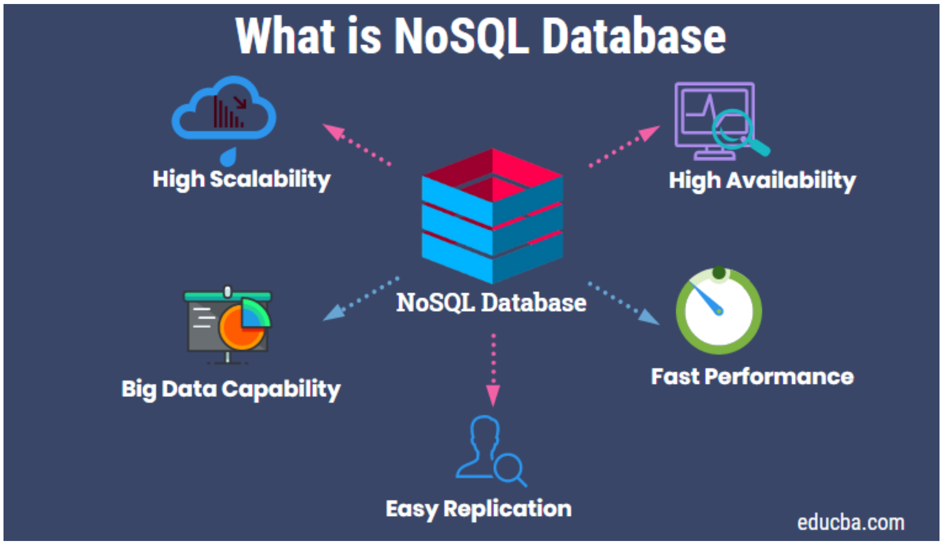
CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

- **SQL:** Structured Query Language (SQL) became the standard for querying and managing relational databases. It was standardized by ANSI in 1986.



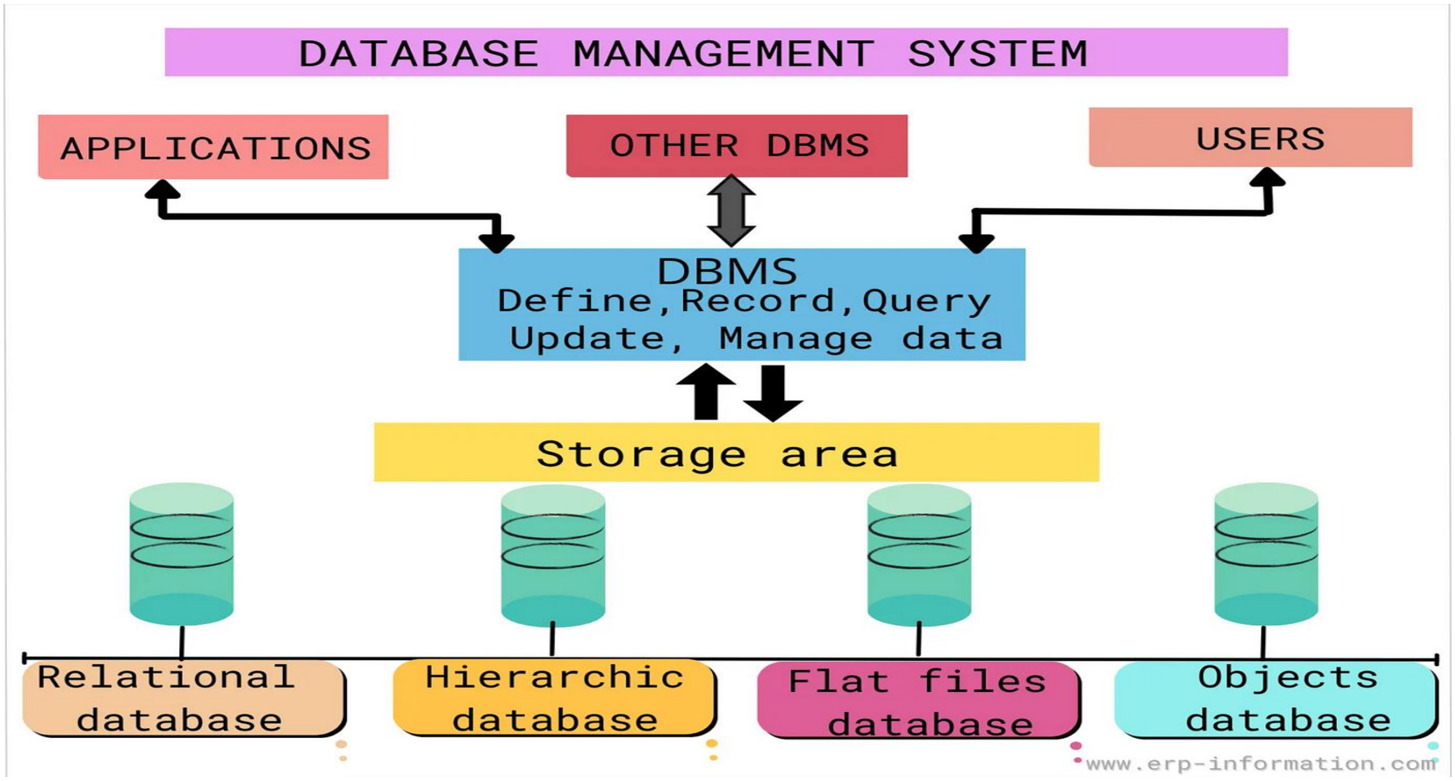


- **NoSQL and Beyond:** In the 2000s, NoSQL databases emerged to handle unstructured data and high-velocity workloads. These include key-value stores, document stores, column-family stores, and graph databases.



Database System:

- **Definition:** A database system is a software application that manages and facilitates access to databases. It provides tools for creating, querying, updating, and managing data.
- **Components:** Typically includes a database engine (for handling data storage and retrieval), a database schema (defining the structure), and a database management system (DBMS) which provides the user interface and interaction tools.



Applications:

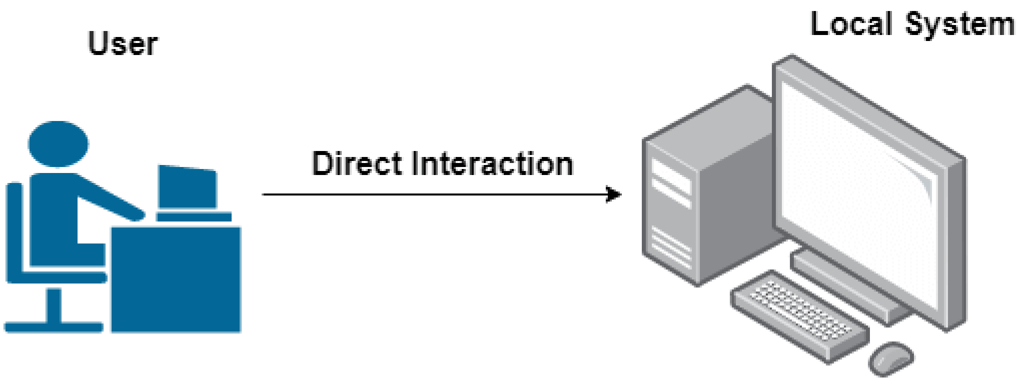
- **Transactional Systems:** Used in banking, retail, and e-commerce for handling transactions.
- **Analytical Systems:** Used in business intelligence and data warehousing for analysis and reporting.
- **Web and Mobile Applications:** Handle user data, preferences, and interactions.
- **Enterprise Applications:** Manage complex business processes and data across various departments.

Characteristics of DBMS

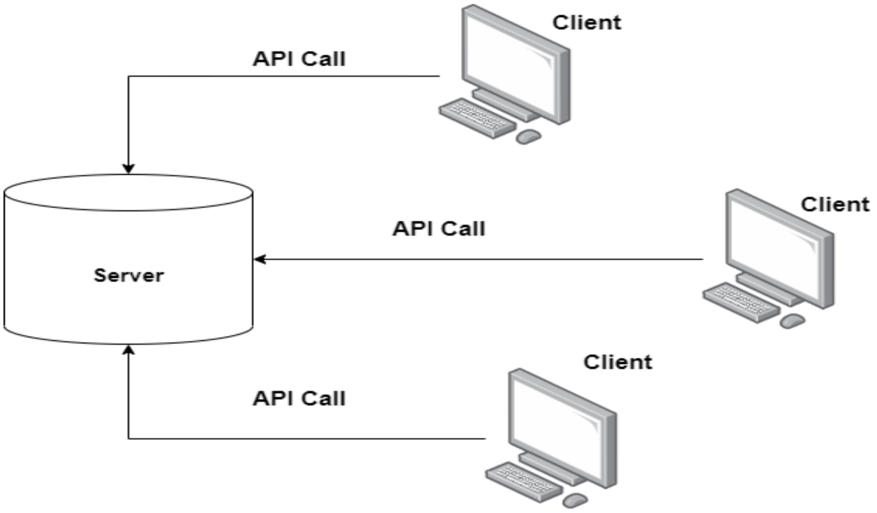
- **Data Abstraction:** Provides a simplified view of the data, hiding the complexities of the underlying data storage.
- **Data Integrity:** Ensures accuracy and consistency of data through constraints, rules, and validation.
- **Data Security:** Controls access to data through user authentication and authorization.
- **Concurrency Control:** Manages simultaneous data access by multiple users to prevent conflicts and ensure consistency.
- **Recovery:** Provides mechanisms to recover data in case of failures or crashes.
- **Data Independence:** Allows changes to the data structure without affecting the application programs.

Application Architecture (One-Tier, Two-Tier, N-Tier)

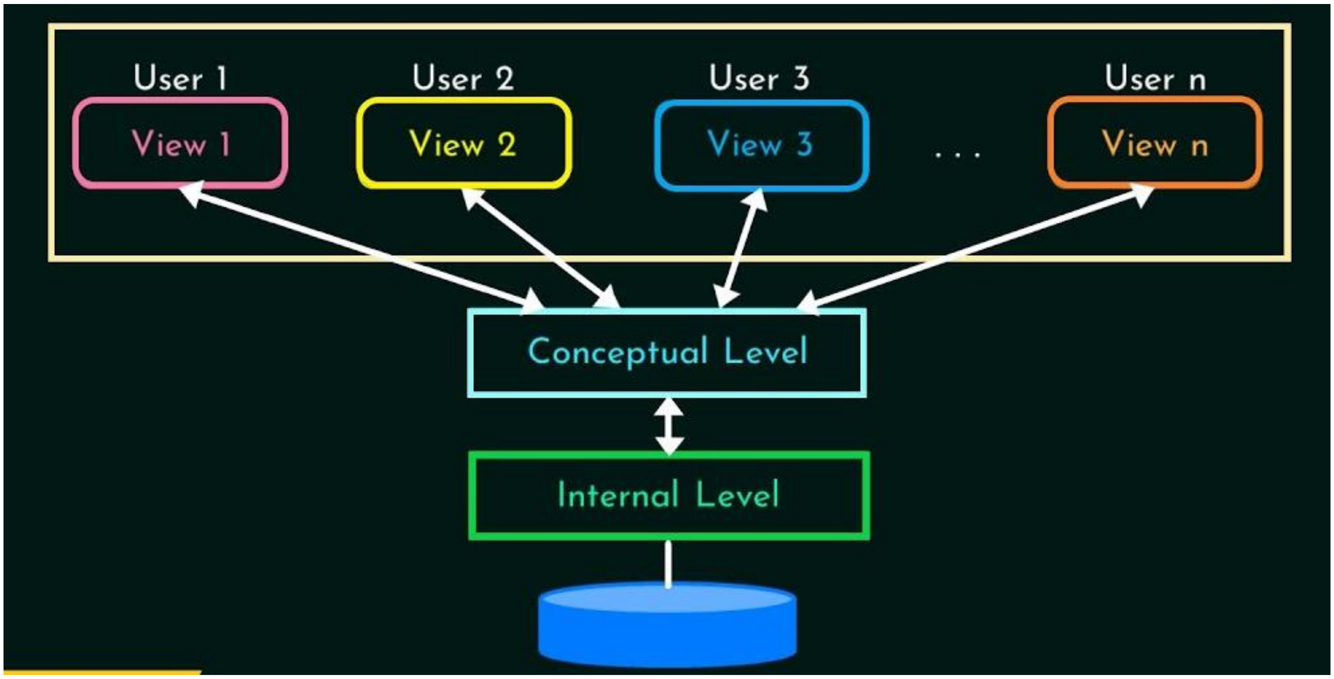
1. **One-Tier Architecture:**
    a. **Description:** The database and application logic are on the same machine or layer.
    b. **Example:** A desktop application that directly accesses a local database.



2. **Two-Tier Architecture:**
    a. **Description:** The application is split into two layers: the client and the server. The client interacts with the database server directly.
    b. **Example:** A client-server application where the client handles the user interface and business logic, while the server manages the database.



3. **N-Tier Architecture:**
    a. **Description:** The application is divided into multiple layers, typically including presentation, application logic, and data management. This design enhances scalability and maintainability.
    b. **Example:** A web application with a presentation layer (web server), business logic layer (application server), and data layer (database server).

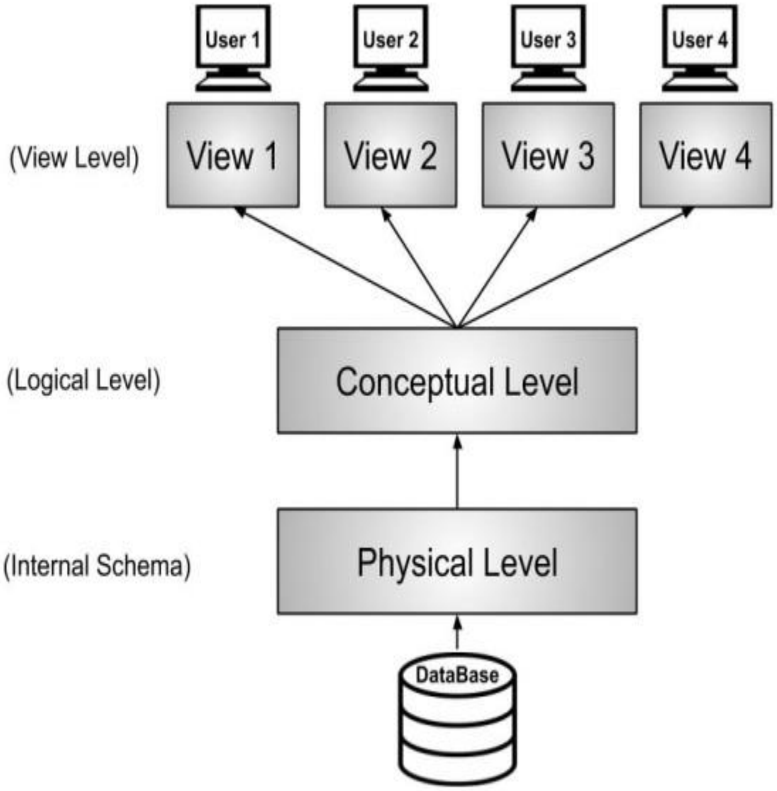


Data Abstraction and Independence

Data abstraction: Data abstraction refers to the process of hiding the complexity of data from the users and showing only the essential information. In database systems, abstraction is necessary to simplify interactions with the database and to ensure that users and applications do not need to understand how data is structured or stored. **Data abstraction is divided into three levels:**

- Physical or Internal Level
- Logical or Conceptual Level
- View or External Level

1. **Physical Level (Internal Level):**
    a. This is the lowest level of data abstraction and deals with how data is physically stored in the database.
    b. It involves details of how the data is organized on disk, including file structures, indexing, and storage paths.

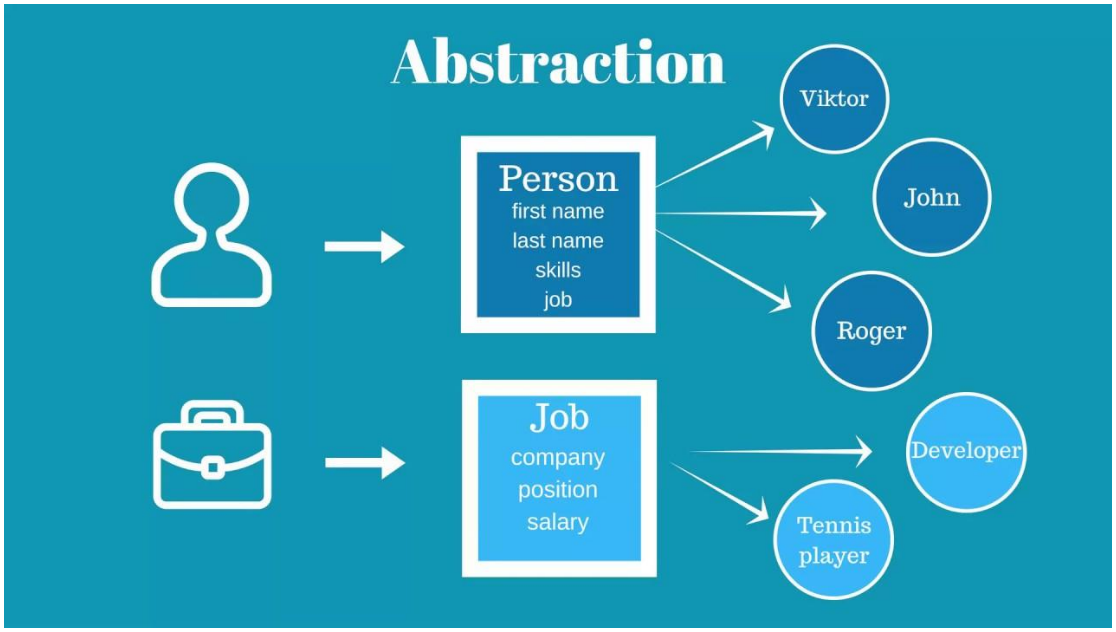


Levels of Data Abstraction

c. Example: A record may be stored as a binary string in a particular file system on a hard drive, but this detail is hidden from users.

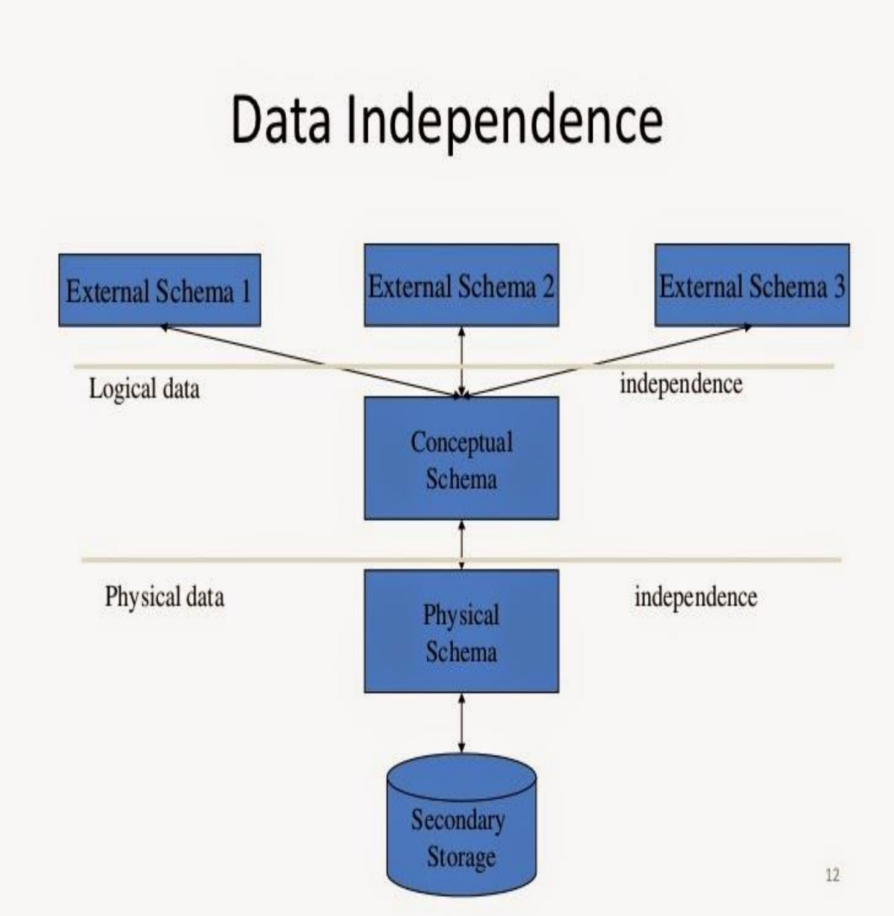
2. **Logical Level (Conceptual Level):**
   a. This level describes what data is stored in the database and what relationships exist among those data items. It focuses on the structure of the database without concerning how it is physically implemented.
   b. The database schema is defined at this level. Users can understand the types of data stored and how they relate, such as entities, attributes, and relationships.
   c. Example: A user sees an entity such as "Employee" with attributes like "Name," "ID," and "Salary," without knowing how these attributes are stored.

3. **View Level (External Level):**
   a. The highest level of abstraction that deals with how individual users interact with the database. It defines different views for different users, depending on their needs.
   b. This level provides different views or interfaces of the same data, ensuring that users can only access the relevant parts of the database.
   c. Example: An HR manager may have a view of employee salary details, while a project manager only sees the project assignments of employees.



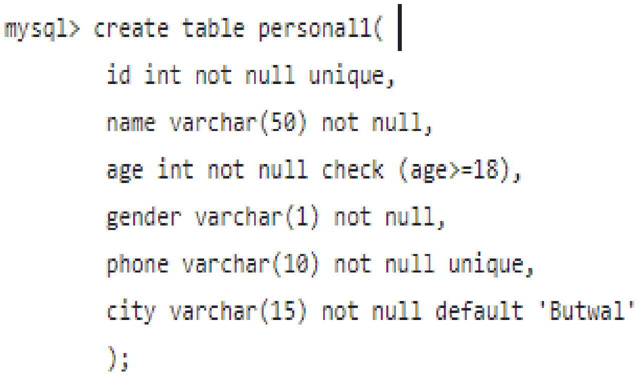
Data Independence: Data independence refers to the ability to change the schema at one level of a database system without having to change the schema at the next higher level. It ensures that changes in the database do not affect the overall system's functionality. ***Data independence is categorized into two types:***

1. *Logical Data Independence:*
2. *Physical Data Independence*



- **Logical Data Independence:**
  - Refers to the ability to change the conceptual (logical) schema without affecting the external schema or application programs.
  - Changes like adding a new field or modifying relationships between entities should not require altering the views or applications.
  - Example: Adding a new "PhoneNumber" attribute to an "Employee" entity should not affect the views or programs that interact with other attributes like "Name" or "Salary."

- **Physical Data Independence:**
  - Refers to the ability to change the physical storage schema without affecting the conceptual schema or application logic.
  - It means that changes like reorganizing data storage, adding indexes, or changing file structures should not impact the logical structure of the database.
  - Example: Changing the storage format of an "Employee" table from one file system to another should not affect how users interact with the "Employee" entity.



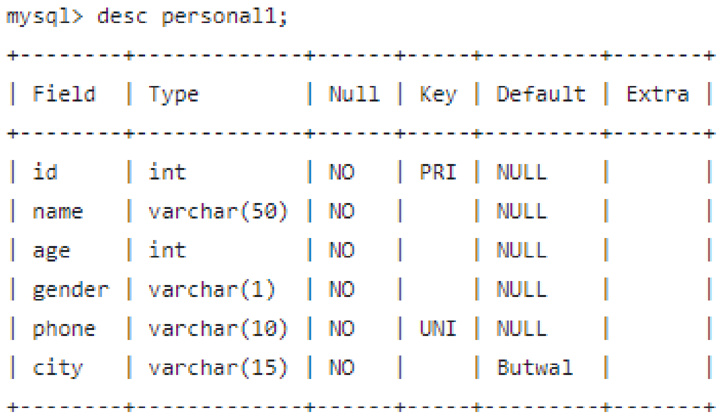
```
mysql> create table personal1(
        id int not null unique,
        name varchar(50) not null,
        age int not null check (age>=18),
        gender varchar(1) not null,
        phone varchar(10) not null unique,
        city varchar(15) not null default 'Butwal'
        );
```

Schemas and Instances

Schemas:

- **Definition:** The structure of the database, including tables, columns, relationships, and constraints. It defines the organization of data.
  - **Types:**
    - **Conceptual Schema:** Represents the entire database structure as a whole.

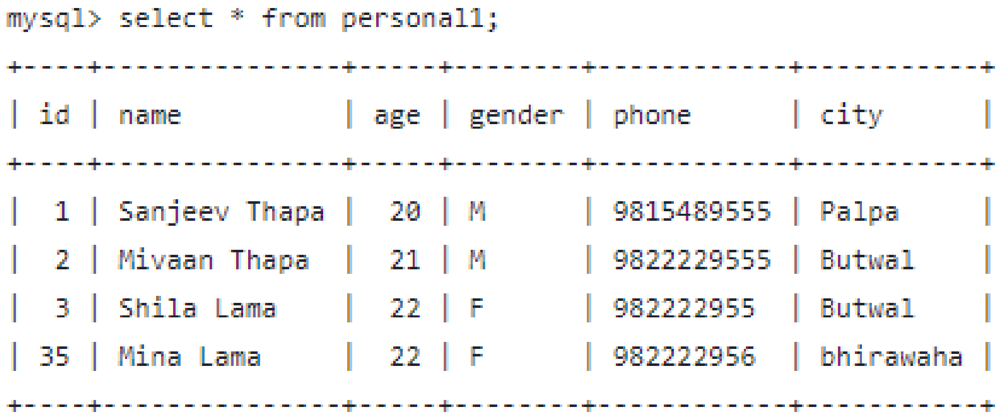
- **Logical Schema:** Defines the logical structure of the data (e.g., tables, keys).
- **Physical Schema:** Describes how data is stored on hardware (e.g., file structures).



```
mysql> desc personal1;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| id     | int         | NO   | PRI | NULL    |       |
| name   | varchar(50) | NO   |     | NULL    |       |
| age    | int         | NO   |     | NULL    |       |
| gender | varchar(1)  | NO   |     | NULL    |       |
| phone  | varchar(10) | NO   | UNI | NULL    |       |
| city   | varchar(15) | NO   |     | Butwal  |       |
+--------+-------------+------+-----+---------+-------+
```

Instances:

- **Definition:** The actual data stored in the database at a particular point in time. It represents the current state of the database.
- **Key Features of an Instance:**
  - **Dynamic State:**
    - Unlike the schema, which remains static, an instance changes as data is added, deleted, or updated in the database. It reflects the current content of the database at any given time.
  - **Snapshots**:
    - Each time data is modified, the database instance changes, creating a new snapshot of the data at that moment.



```
mysql> select * from personal1;
+----+---------------+-----+--------+------------+-----------+
| id | name          | age | gender | phone      | city      |
+----+---------------+-----+--------+------------+-----------+
|  1 | Sanjeev Thapa |  20 | M      | 9815489555 | Palpa     |
|  2 | Mivaan Thapa  |  21 | M      | 9822229555 | Butwal    |
|  3 | Shila Lama    |  22 | F      | 982222955  | Butwal    |
| 35 | Mina Lama     |  22 | F      | 982222956  | bhirawaha |
+----+---------------+-----+--------+------------+-----------+
```

Database Manager and Users

Database Manager:

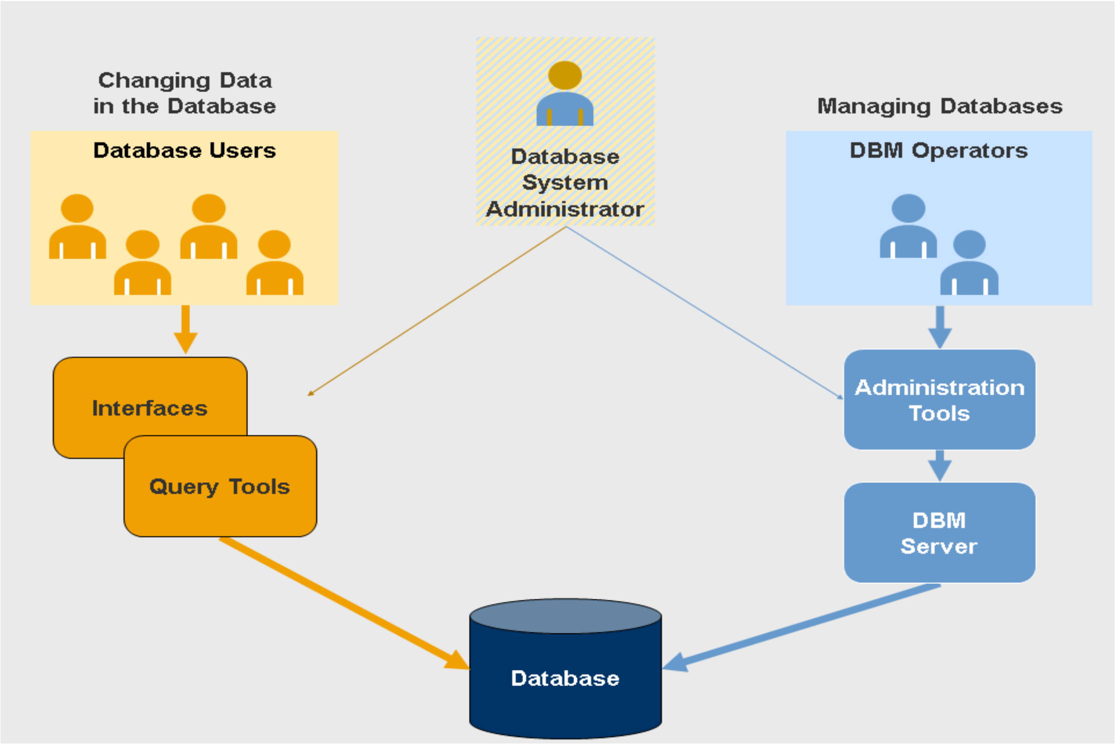
- **Role:** A component of the DBMS responsible for handling database operations such as data storage, retrieval, and updates. It includes the database engine, which executes SQL queries and maintains data integrity.

- **Responsibilities:**

  ✓ **Query Processing:** Interprets and executes database queries.

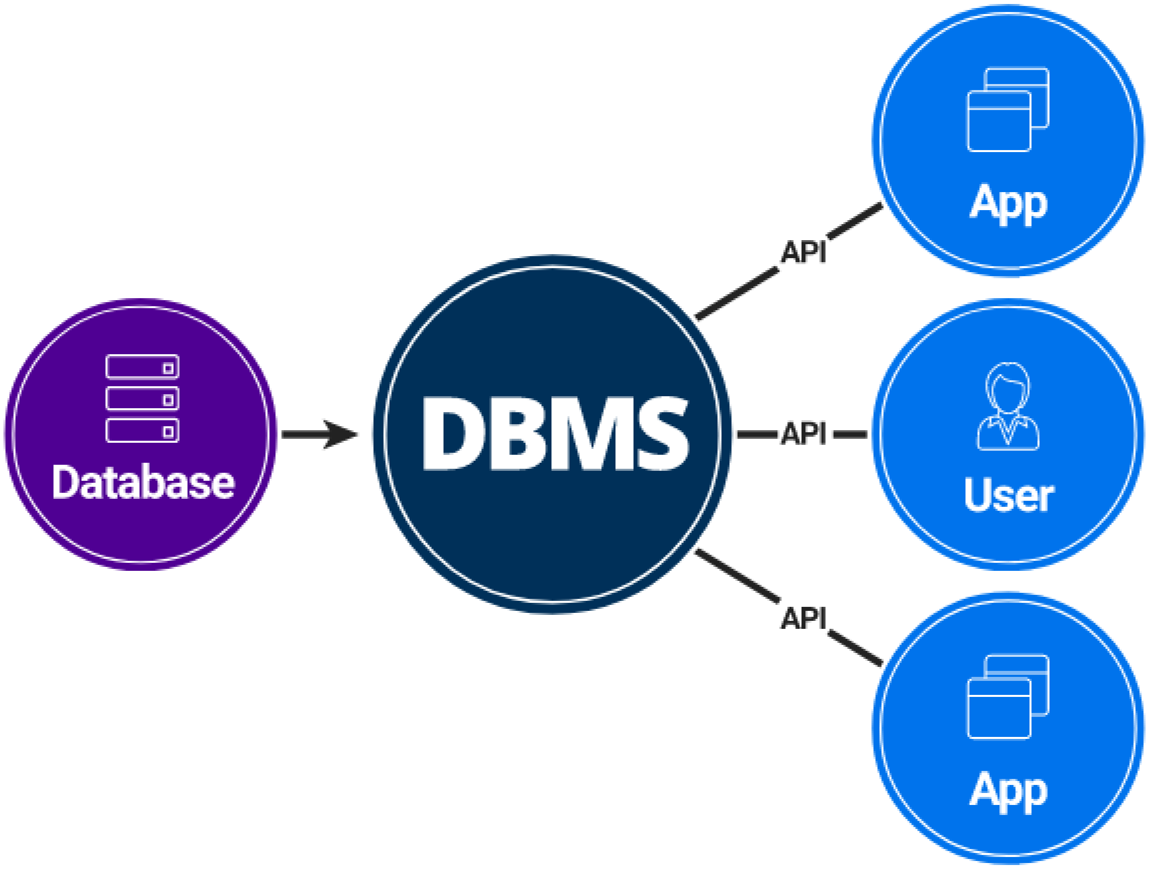
  ✓ **Transaction Management:** Ensures ACID properties (Atomicity, Consistency, Isolation, Durability) for transactions.

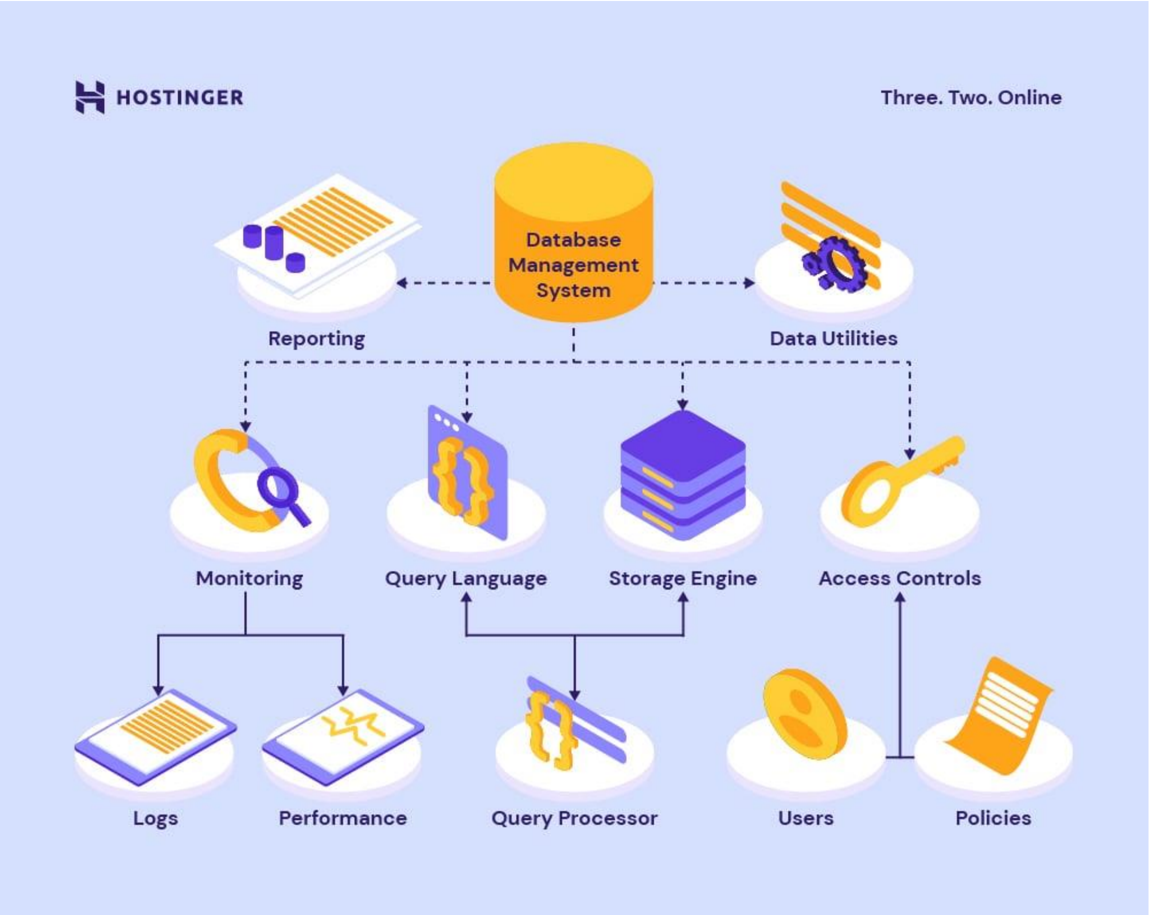
  ✓ **Backup and Recovery:** Manages data backups and restores in case of failure.



Users:

- **Database Administrators (DBAs):** Manage and maintain the database system, handle security, backups, and performance tuning.
- **Application Programmers:** Develop and maintain applications that interact with the database.
- **End Users:** Use applications to perform tasks such as data entry, retrieval, and reporting.
- **Data Analysts:** Analyze and interpret data, often using queries and reporting tools.


Database
DBMS
API
App
API
User
API
App


HOSTINGER
Three. Two. Online
Database Management System
Reporting
Data Utilities
Monitoring
Query Language
Storage Engine
Access Controls
Logs
Performance
Query Processor
Users
Policies

Q/A

Fill in the Blanks (20 Questions)

1. The first generation of database management systems was known as _________.
2. A DBMS ensures that data is stored securely and can be accessed using a _________.
3. One of the main characteristics of DBMS is _________, which eliminates redundancy.
4. In a two-tier architecture, the database resides on the _________ tier.
5. _________ refers to the process of hiding the complexity of data from users and only showing essential information.
6. A _________ schema describes the logical structure of a database.
7. The current content of a database at any given moment is called a _________.
8. In an n-tier architecture, the presentation layer is typically handled by the _________.
9. The DBMS allows multiple users to access data simultaneously, a feature known as _________.
10. _________ is the property that allows changes in physical storage without affecting the logical schema.
11. The part of the DBMS that manages storage on disk is called the _________.
12. Data at the view level is abstracted for users based on their _________.
13. A _________ is a collection of related data stored electronically in a DBMS.
14. _________ is the highest level of data abstraction.
15. The process of defining relationships between tables in a database is known as _________.
16. _________ refers to the ability to change the schema at one level without altering the schema at the next higher level.
17. The _________ architecture separates the presentation, logic, and data layers.
18. A database _________ refers to a user or a role that interacts with the DBMS.
19. In a one-tier architecture, the application and database reside on the same _________.

20.A DBMS provides controlled access to the database through ________.

Multiple Choice Questions (MCQ) (20 Questions)

1. **Which of the following is NOT a characteristic of a DBMS?**
   - a) Data Redundancy Control
   - b) Secure Data Access
   - c) Inconsistent Data Storage
   - d) Data Integrity
2. **In which architecture does the database and application reside on the same machine?**
   - a) One-Tier
   - b) Two-Tier
   - c) Three-Tier
   - d) N-Tier
3. **Data abstraction involves hiding the ________ details from the user.**
   - a) Logical
   - b) Conceptual
   - c) Physical
   - d) Internal
4. **The current state of the data in a database is referred to as a(n) ________.**
   - a) Schema
   - b) Instance
   - c) View
   - d) Key
5. **Which of the following architectures separates the presentation layer, logic layer, and data layer?**
   - a) One-Tier Architecture
   - b) Two-Tier Architecture
   - c) N-Tier Architecture

- d) Centralized Architecture

6. **Which is a primary function of a Database Manager?**
   - a) Manage user interaction with the data
   - b) Store data permanently
   - c) Ensure logical schema stability
   - d) Maintain physical file structures
7. **Which level of data abstraction describes what data is stored and the relationships among them?**
   - a) Physical Level
   - b) Logical Level
   - c) View Level
   - d) Internal Level
8. **A schema that defines the entire structure of the database is called a ________.**
   - a) Physical Schema
   - b) Logical Schema
   - c) External Schema
   - d) Subschema
9. **Which of the following users interact with the DBMS without writing complex queries?**
   - a) End Users
   - b) Database Administrators
   - c) Application Developers
   - d) Data Scientists
10. **What type of independence allows changes in the physical structure of data without affecting the logical schema?**
    - a) Logical Independence
    - b) Physical Independence
    - c) Data Abstraction
    - d) Conceptual Independence
11. **The database instance can be best described as:**

- a) The blueprint of the database
- b) The current data in the database
- c) A snapshot of past transactions
- d) A view of the database

12. Which of the following best describes a Database Management System (DBMS)?

- a) A system that stores data manually
- b) A system that allows data to be created, managed, and retrieved electronically
- c) A system that duplicates data for security
- d) A system that only allows single-user access

13. In a two-tier architecture, the user interface layer communicates directly with:

- a) Data Layer
- b) Application Layer
- c) Business Logic Layer
- d) External Layer

14. Schemas are classified into how many levels in a DBMS?

- a) 1
- b) 2
- c) 3
- d) 4

15. Which of the following is NOT a function of the Database Manager?

- a) Backup and Recovery
- b) Transaction Management
- c) Data Encryption
- d) Data Visualization

16. Which data abstraction level defines different views of the same database?

- a) Physical
- b) Conceptual
- c) View

- d) Logical

17. **What is the main role of an external schema?**
    - a) Define how data is stored
    - b) Define different views for various users
    - c) Define database constraints
    - d) Handle backup and recovery
18. **Which architecture type provides better scalability in a large enterprise environment?**
    - a) One-Tier
    - b) Two-Tier
    - c) N-Tier
    - d) Centralized
19. **Who is responsible for defining database schemas in an organization?**
    - a) Database Manager
    - b) Database Designer
    - c) End Users
    - d) Application Developer
20. **The DBMS characteristic that ensures that multiple transactions occur simultaneously without conflict is:**
    - a) Atomicity
    - b) Isolation
    - c) Durability
    - d) Consistency

Short Questions (20 Questions)

1. Define a database and explain its primary purpose.
2. What are the key characteristics of a DBMS?
3. Differentiate between one-tier and two-tier application architectures.
4. Explain data abstraction in the context of a DBMS.

5. What is the significance of database schemas?
6. What is the difference between a schema and an instance?
7. What role does the database manager play in DBMS?
8. Describe physical data independence.
9. What is an external schema in a database?
10. How does a DBMS handle multiple users accessing the same data?
11. What is data redundancy, and how does DBMS address it?
12. Explain logical data independence.
13. What is a view schema?
14. Differentiate between the physical level and logical level of data abstraction.
15. Define and explain database instances with an example.
16. What is the role of a database user?
17. How does a DBMS ensure data consistency?
18. What are the components of a two-tier architecture?
19. What is an n-tier architecture, and where is it used?
20. How does database security fit into DBMS management?

Comprehensive Questions (20 Questions)

1. Explain the evolution of database systems and discuss how modern DBMS has transformed data management in businesses.
2. Discuss the characteristics of DBMS in detail and explain how these features provide an advantage over traditional file-based systems.
3. Compare and contrast one-tier, two-tier, and n-tier architectures, highlighting the advantages and disadvantages of each.
4. How does data abstraction in DBMS enhance flexibility and efficiency in database management? Provide examples.
5. Discuss the different types of schemas in DBMS. How do they interact with each other?

6. Explain with examples the concept of physical and logical data independence. Why are they important for database design and maintenance?

7. Define the roles and responsibilities of a Database Manager in ensuring the smooth operation of a database system.

8. Discuss how DBMS supports multiple users and ensures data consistency and isolation during concurrent transactions.

9. How do database schemas contribute to the design and architecture of a database system? Illustrate with an example.

10. Describe the challenges of managing data in multi-tier architectures and how they are addressed in DBMS.

11. Analyze the importance of database backup and recovery techniques in preventing data loss and maintaining data integrity.

12. Discuss how logical independence helps in maintaining a database system's adaptability to changes without affecting applications.

13. Explain the concept of a database instance and its relationship with database schemas.

14. What role do views (external schemas) play in providing data security and customized user experiences in a DBMS?

15. Analyze the pros and cons of n-tier architectures in the context of scalability, security, and performance.

16. Discuss how database managers ensure security and data privacy in an organization with multiple database users.

17. Explain the significance of database independence (physical and logical) in making future system changes.

18. How does DBMS manage concurrency, and why is it critical in a multi-user environment?

19. Describe the process of database recovery from catastrophic failure and the role of the database manager in this process.

20. Evaluate the role of schemas in ensuring efficient database design and performance in an enterprise environment.