# **Database Management System (DBMS) Concepts**

## **1. Introduction to DBMS**

A **Database Management System (DBMS)** is software that enables the creation, management, and manipulation of databases. It acts as an intermediary between users and the database, allowing users to perform operations like data storage, retrieval, and manipulation efficiently.

### **Key Functions of a DBMS:**

* Data Storage Management
* Data Retrieval and Query Processing
* Data Security and Integrity
* Data Backup and Recovery
* Multi-user Access Control

## **2. Database Models**

Several database models exist, each with its own structure and use cases. The most common are:

### **2.1 Relational Model**

* Data is organized in tables (relations).
* Each table consists of rows (records) and columns (attributes).
* Uses **SQL** (Structured Query Language) for database operations.

### **2.2 Hierarchical Model**

* Data is organized in a tree-like structure.
* Each record has a single parent and can have multiple children.

### **2.3 Network Model**

* Similar to the hierarchical model, but allows more complex relationships with multiple parent-child relationships.

### **2.4 Object-Oriented Model**

* Data is represented as objects, similar to object-oriented programming.

### **2.5 Document Store Model**

* Designed for semi-structured data, using documents as the primary unit of storage (e.g., JSON, XML).

## **3. Keys in DBMS**

**Keys** are crucial for uniquely identifying records in a database table. They help maintain the integrity and relationships of data.

### **3.1 Primary Key**

* A unique identifier for a record in a table.
* Cannot contain null values.
* Example: In a table of students, the StudentID can be a primary key.

### **3.2 Foreign Key**

* A field (or collection of fields) in one table that refers to the primary key in another table.
* It establishes a relationship between the two tables.
* Example: CourseID in a StudentCourses table referencing CourseID in a Courses table.

### **3.3 Candidate Key**

* A set of attributes that can uniquely identify a record.
* A table can have multiple candidate keys, but only one is chosen as the primary key.

### **3.4 Composite Key**

* A primary key composed of two or more attributes.
* Example: OrderID and ProductID can be combined to form a composite key in an OrderDetails table.

## **4. Normalization**

**Normalization** is the process of organizing data in a database to reduce redundancy and improve data integrity. It involves decomposing a database into smaller, related tables.

### **4.1 Normal Forms**

Normalization is typically achieved through several normal forms (NF):

* **First Normal Form (1NF)**:
  + Eliminate duplicate columns from the same table.
  + Create unique identifiers for each row.
* **Second Normal Form (2NF)**:
  + Meet all requirements of 1NF.
  + Remove partial dependencies (no non-key attribute should depend on a part of a composite key).
* **Third Normal Form (3NF)**:
  + Meet all requirements of 2NF.
  + Remove transitive dependencies (non-key attributes should not depend on other non-key attributes).

### **4.2 Example of Normalization**

Consider a Students table:

| **StudentID** | **StudentName** | **Course1** | **Course2** |
| --- | --- | --- | --- |
| 1 | Alice | Math | Science |
| 2 | Bob | Math | History |

After normalization into 1NF, we can separate the courses:

**Students Table**:

| **StudentID** | **StudentName** |
| --- | --- |
| 1 | Alice |
| 2 | Bob |

**Courses Table**:

| **StudentID** | **Course** |
| --- | --- |
| 1 | Math |
| 1 | Science |
| 2 | Math |
| 2 | History |

## **5. Transactions in DBMS**

A **transaction** is a sequence of operations performed as a single logical unit of work. A transaction must be atomic, consistent, isolated, and durable (ACID properties).

### **5.1 ACID Properties**

* **Atomicity**: Ensures that all operations within a transaction are completed successfully. If one operation fails, the entire transaction is aborted.
* **Consistency**: Ensures that a transaction takes the database from one valid state to another, maintaining database invariants.
* **Isolation**: Ensures that concurrent transactions do not interfere with each other. Each transaction must be isolated from others.
* **Durability**: Ensures that once a transaction is committed, it remains in the system even in the case of a failure.

### **5.2 Example of a Transaction**

Consider a bank transfer transaction:

1. Deduct money from Account A.
2. Add money to Account B.

If the deduction succeeds but the addition fails, the entire transaction should be rolled back to maintain consistency.

## **6. Data Manipulation Language (DML)**

**DML** is a subset of SQL used for managing data within database objects. Common DML commands include:

### **6.1 SELECT**

* Retrieves data from one or more tables.

Example:  
SELECT StudentName FROM Students WHERE StudentID = 1;

### **6.2 INSERT**

* Adds new records to a table.

Example:  
INSERT INTO Students (StudentID, StudentName) VALUES (3, 'Charlie');

### **6.3 UPDATE**

* Modifies existing records in a table.

Example:  
UPDATE Students SET StudentName = 'Alice Smith' WHERE StudentID = 1;

### **6.4 DELETE**

* Removes records from a table.

Example:  
DELETE FROM Students WHERE StudentID = 2;

## **7. Data Definition Language (DDL)**

**DDL** is a subset of SQL used for defining and managing database structures. Common DDL commands include:

### **7.1 CREATE**

* Creates new database objects like tables, indexes, or databases.

Example:  
CREATE TABLE Students (

StudentID INT PRIMARY KEY,

StudentName VARCHAR(100)

);

### **7.2 ALTER**

* Modifies the structure of an existing database object.

Example:  
ALTER TABLE Students ADD COLUMN Age INT;

### **7.3 DROP**

* Deletes an existing database object.

Example:  
DROP TABLE Students;

### **7.4 TRUNCATE**

* Removes all records from a table but keeps the table structure intact.

Example:  
TRUNCATE TABLE Students;

### **7. DCL Commands**

* **GRANT**: Gives specific privileges to users.

Example:  
GRANT SELECT, INSERT ON employees TO user1;

* **REVOKE**: Removes specific privileges from users.

Example:  
REVOKE INSERT ON employees FROM user1;

### **8. TCL Commands**

* **COMMIT**: Saves all changes made during the current transaction.
* **ROLLBACK**: Undoes all changes made during the current transaction.

### **9. Joins in SQL**

* **INNER JOIN**: Returns records with matching values in both tables.

Example:  
SELECT employees.name, departments.name

FROM employees

INNER JOIN departments ON employees.department\_id = departments.id;

* **LEFT JOIN**: Returns all records from the left table and matched records from the right table.
* **RIGHT JOIN**: Returns all records from the right table and matched records from the left table.
* **FULL OUTER JOIN**: Returns all records when there is a match in either left or right table records.

### **10. Indexes**

* **Definition**: An index is a database object that improves the speed of data retrieval operations on a database table at the cost of additional storage space.
* **Types of Indexes**:
  + **Unique Index**: Ensures that all values in a column are different.
  + **Composite Index**: An index on two or more columns.
* **Creating an Index**:

Example:  
CREATE INDEX idx\_employee\_name ON employees (name);

### **11. Views**

* **Definition**: A view is a virtual table based on the result of a SQL query. It does not store data physically but provides a way to represent data from one or more tables.
* **Creating a View**:

Example:  
CREATE VIEW employee\_salaries AS

SELECT name, salary FROM employees WHERE salary > 50000;

### **12. Transactions**

* **Definition**: A transaction is a sequence of operations performed as a single logical unit of work. It ensures data integrity and consistency.
* **ACID Properties**:
  + **Atomicity**: Ensures that all operations within a transaction are completed successfully; otherwise, the transaction is aborted.
  + **Consistency**: Ensures that a transaction brings the database from one valid state to another.
  + **Isolation**: Ensures that transactions do not interfere with each other.
  + **Durability**: Ensures that once a transaction is committed, it will remain so, even in the event of a system failure.

### **13. Data Integrity**

* **Types of Data Integrity**:
  + **Entity Integrity**: Ensures that each table has a primary key, and that the values in the primary key are unique and not null.
  + **Referential Integrity**: Ensures that foreign keys correctly point to primary keys in related tables.
  + **Domain Integrity**: Ensures that all entries in a column fall within a specified range of values.

### **14. Advanced SQL Concepts**

* **Subqueries**: A query within another query. It can be used in SELECT, INSERT, UPDATE, and DELETE statements.

Example:  
SELECT name FROM employees WHERE salary > (SELECT AVG(salary) FROM employees);

* **Stored Procedures**: A set of SQL statements that can be stored and executed in the database.
* **Triggers**: A set of instructions that are automatically executed in response to certain events on a table or view.

## **8. Indexing**

**Indexing** is a data structure technique used to quickly locate and access data in a database. Indexes can significantly improve query performance but may slow down data modification operations.

### **8.1 Types of Indexes**

* **Unique Index**: Ensures all values in a column are unique.
* **Composite Index**: An index on multiple columns.
* **Full-Text Index**: Used for full-text search capabilities.

### **8.2 Example of Creating an Index**

CREATE INDEX idx\_student\_name ON Students (StudentName);

## **9. Views**

A **view** is a virtual table based on the result of a SQL query. It can encapsulate complex queries and present data in a more understandable way.

### **9.1 Creating a View**

CREATE VIEW StudentCourses AS

SELECT Students.StudentName, Courses.Course

FROM Students

JOIN StudentCourses ON Students.StudentID = StudentCourses.StudentID;

### **9.2 Advantages of Views**

* Simplifies complex queries.
* Provides data abstraction and security.
* Can aggregate data for reporting.

## **10. Constraints**

**Constraints** are rules enforced on data columns to ensure data integrity.

### **10.1 Types of Constraints**

* **NOT NULL**: Ensures a column cannot have a NULL value.
* **UNIQUE**: Ensures all values in a column are different.
* **CHECK**: Ensures that values in a column satisfy a specific condition.
* **FOREIGN KEY**: Ensures referential integrity between tables.

### **10.2 Example of Adding Constraints**

ALTER TABLE Students

ADD CONSTRAINT chk\_age CHECK (Age >= 0);

### **DBMS Interview Questions**

1. **What is a DBMS?**
   * **Answer**: A Database Management System (DBMS) is software that allows users to create, manage, and manipulate databases. It provides an interface for users to interact with the data and ensures data integrity, security, and concurrent access.
2. **What are the types of DBMS?**
   * **Answer**: The main types of DBMS include:
     + **Hierarchical DBMS**: Data is organized in a tree-like structure.
     + **Network DBMS**: Data is organized in a graph structure, allowing multiple relationships.
     + **Relational DBMS (RDBMS)**: Data is organized in tables, using relationships between them.
     + **Object-oriented DBMS**: Data is stored in the form of objects, similar to object-oriented programming.
     + **NoSQL DBMS**: Designed for unstructured data and scalability, including document stores, key-value stores, column stores, and graph databases.
3. **What is a relational database?**
   * **Answer**: A relational database is a type of DBMS that organizes data into tables (relations) with rows and columns. It allows users to define relationships between tables using foreign keys and supports SQL for querying.
4. **What is normalization? Why is it important?**
   * **Answer**: Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. It involves dividing a database into smaller tables and defining relationships between them. It is important because it helps maintain data consistency and reduces the likelihood of anomalies.
5. **What are the different normal forms?**
   * **Answer**: The different normal forms include:
     + **First Normal Form (1NF)**: Ensures that each column contains atomic values and each entry is unique.
     + **Second Normal Form (2NF)**: Achieves 1NF and removes partial dependencies.
     + **Third Normal Form (3NF)**: Achieves 2NF and removes transitive dependencies.
6. **What is a primary key?**
   * **Answer**: A primary key is a unique identifier for each record in a table. It ensures that no two rows have the same value in the primary key column(s) and that the primary key cannot contain null values.
7. **What is a foreign key?**
   * **Answer**: A foreign key is a field (or a group of fields) in one table that uniquely identifies a row in another table. It establishes a relationship between the two tables, ensuring referential integrity.
8. **What are ACID properties?**
   * **Answer**: ACID properties ensure reliable processing of database transactions. They are:
     + **Atomicity**: Ensures that all operations in a transaction are completed; if not, the transaction is aborted.
     + **Consistency**: Ensures that a transaction brings the database from one valid state to another.
     + **Isolation**: Ensures that transactions do not interfere with each other.
     + **Durability**: Guarantees that once a transaction is committed, it remains so even in case of a failure.
9. **What is data redundancy?**
   * **Answer**: Data redundancy occurs when the same piece of data is stored in multiple places in a database. This can lead to data inconsistency and increased storage costs. Normalization aims to reduce data redundancy.
10. **What is a database schema?**
    * **Answer**: A database schema is the structure that defines how data is organized in a database. It includes the tables, fields, relationships, views, and other elements that determine how data is stored and accessed.

### **SQL Interview Questions**

1. **What is SQL?**
   * **Answer**: SQL (Structured Query Language) is a standard programming language used to manage and manipulate relational databases. It allows users to perform tasks such as querying data, inserting new records, updating existing records, and deleting records.
2. **What are the different types of SQL statements?**
   * **Answer**: SQL statements can be categorized into four main types:
     + **DDL (Data Definition Language)**: Commands for defining database structures (e.g., CREATE, ALTER, DROP).
     + **DML (Data Manipulation Language)**: Commands for manipulating data (e.g., SELECT, INSERT, UPDATE, DELETE).
     + **DCL (Data Control Language)**: Commands for controlling access to data (e.g., GRANT, REVOKE).
     + **TCL (Transaction Control Language)**: Commands for managing transactions (e.g., COMMIT, ROLLBACK).
3. **What is the difference between DELETE, TRUNCATE, and DROP?**
   * **Answer**:
     + **DELETE**: Removes specific rows from a table based on a condition. It can be rolled back.
     + **TRUNCATE**: Removes all rows from a table but retains the table structure. It cannot be rolled back and is faster than DELETE.
     + **DROP**: Deletes the entire table structure and its data. The table is permanently removed from the database.
4. **How do you retrieve unique values from a column in SQL?**
   * **Answer**: You can retrieve unique values using the DISTINCT keyword.

Example:  
sql  
Copy code  
SELECT DISTINCT column\_name FROM table\_name;

1. **What is a JOIN in SQL?**
   * **Answer**: A JOIN is used to combine rows from two or more tables based on a related column between them. Common types of JOINs include:
     + **INNER JOIN**: Returns only matching rows from both tables.
     + **LEFT JOIN**: Returns all rows from the left table and matching rows from the right table.
     + **RIGHT JOIN**: Returns all rows from the right table and matching rows from the left table.
     + **FULL OUTER JOIN**: Returns all rows when there is a match in either left or right table records.
2. **What is a primary key and how is it different from a unique key?**
   * **Answer**: A primary key uniquely identifies each record in a table and cannot contain null values. A unique key also enforces uniqueness but can contain null values. A table can have only one primary key but can have multiple unique keys.
3. **What is a subquery?**
   * **Answer**: A subquery is a query nested inside another query. It can be used in various SQL statements, such as SELECT, INSERT, UPDATE, and DELETE, to provide results that can be used in the main query.

Example:  
sql  
Copy code  
SELECT name FROM employees WHERE salary > (SELECT AVG(salary) FROM employees);

1. **What are aggregate functions in SQL?**
   * **Answer**: Aggregate functions perform calculations on multiple rows of a single column and return a single value. Common aggregate functions include:
     + COUNT(): Counts the number of rows.
     + SUM(): Calculates the total sum of a numeric column.
     + AVG(): Calculates the average value of a numeric column.
     + MAX(): Finds the maximum value.
     + MIN(): Finds the minimum value.
2. **How do you create a view in SQL?**
   * **Answer**: A view is created using the CREATE VIEW statement. A view is a virtual table based on the result of a SELECT query.

Example:  
sql  
Copy code  
CREATE VIEW employee\_salaries AS

SELECT name, salary FROM employees WHERE salary > 50000;

1. **What is the difference between UNION and UNION ALL?**
   * **Answer**:
     + **UNION**: Combines the results of two or more SELECT queries and removes duplicate rows.
     + **UNION ALL**: Combines the results of two or more SELECT queries but includes all rows, including duplicates.