Module 5 – Introduction to DBMS

THEORY ASSIGNMENT

1. Introduction to SQL

1. What is SQL, and why is it essential in database management?

SQL (**Structured Query Language**) is the standard language used to manage and interact with relational databases.

Essential because it:

- Stores, retrieves, updates, and deletes data.
- Manages database structure.
- Ensures data security and consistency.
- Works across most database systems.

2. Explain the difference between DBMS and RDBMS.

DBMS (Database Management System):

- Manages data as files.
- No relationships between data.
- Example: Microsoft Access, XML DB.

RDBMS (Relational DBMS):

- Stores data in tables with rows & columns.
- Supports relationships between tables using keys.
- Example: MySQL, Oracle, PostgreSQL.

3. Describe the role of SQL in managing relational databases.

SQL is used to **store, retrieve, update, and manage data** in relational databases. It defines tables, builds relationships, enforces security, and ensures efficient data handling.

4. What are the key features of SQL?

Key Features of SQL (in short):

- Simple and easy to learn.
- Supports **CRUD operations** (Create, Read, Update, Delete).
- Can define and manage database structure.
- Ensures **data security** with permissions.
- Works with large data efficiently.
- Standardized across most databases.

2. SQL Syntax

1. What are the basic components of SQL syntax?

Basic components of SQL syntax (in short):

- **CLAUSES** → Keywords like SELECT, WHERE.
- **STATEMENTS** → Complete commands (e.g., SELECT * FROM Students;).
- **EXPRESSIONS** → Conditions/values (e.g., age > 18).
- **PREDICATES** → Used to filter data (WHERE, LIKE).
- **QUERIES** → Requests to get/manipulate data.

2. Write the general structure of an SQL SELECT statement.

SELECT column1, column2, ...

FROM table_name

WHERE condition

GROUP BY column

HAVING condition

ORDER BY column;

3. Explain the role of clauses in SQL statements.

Clauses are keywords in SQL statements that define what data to select, filter, group, or sort.

3. SQL Constraints

1. What are constraints in SQL? List and explain the different types of constraints.

Constraints in SQL:
Rules applied on table columns to maintain data accuracy and integrity.

Types of Constraints (in short):

- NOT NULL → Column cannot have NULL values.
- UNIQUE → Ensures all values in a column are unique.
- PRIMARY KEY → Uniquely identifies each row (NOT NULL + UNIQUE).
- FOREIGN KEY → Links two tables, maintains referential integrity.
- CHECK → Ensures values meet a condition (e.g., age > 18).
- DEFAULT → Assigns a default value if none is provided.

2. How do PRIMARY KEY and FOREIGN KEY constraints differ?

PRIMARY KEY:

- Uniquely identifies each row in a table.
- Cannot be NULL.

FOREIGN KEY:

- Links one table to another.
- Ensures referential integrity between tables.

3. What is the role of NOT NULL and UNIQUE constraints?

NOT NULL: Ensures a column **cannot have empty (NULL) values**.
UNIQUE: Ensures all values in a column are **distinct**.

4. Main SQL Commands and Sub-commands (DDL)

1. Define the SQL Data Definition Language (DDL).

DDL is used to define and manage database structures like tables, indexes, and schemas.
Examples: CREATE, ALTER, DROP.

2. Explain the CREATE command and its syntax.

```
CREATE TABLE table_name (
   column1 datatype constraints,
   column2 datatype constraints,
   ...
);
```

3. What is the purpose of specifying data types and constraints during table creation?

- **Data types:** Define the type of data a column can store (e.g., INT, VARCHAR).
- **Constraints:** Ensure data accuracy and integrity (e.g., NOT NULL, PRIMARY KEY).

5. ALTER Command

1. What is the use of the ALTER command in SQL?

ALTER is used to **modify the structure of an existing table** without deleting data.

2. How can you add, modify, and drop columns from a table using ALTER?

- **Add a column :** ALTER TABLE table_name ADD column_name datatype;
- **Modify a column :** ALTER TABLE table_name MODIFY column_name new_datatype;
- **Drop a column :** ALTER TABLE table_name DROP COLUMN column_name;

6. DROP Command

1. What is the function of the DROP command in SQL?

DROP is used to **delete a table, database, or other database objects permanently**.

2. What are the implications of dropping a table from a database?

Implications of dropping a table:

- All data in the table is **lost permanently**.
- Table structure, indexes, and constraints are **deleted**.
- Cannot be undone unless a backup exists.

7. Data Manipulation Language (DML)

1. Define the INSERT, UPDATE, and DELETE commands in SQL.

SQL Commands:

- **INSERT:** Adds new rows to a table.
- **UPDATE:** Modifies existing data in a table.
- **DELETE:** Removes rows from a table.

2. What is the importance of the WHERE clause in UPDATE and DELETE operations?

Importance of WHERE clause:

- Ensures only specific rows are updated or deleted.
- Without WHERE, all rows in the table will be affected.

8. Data Query Language (DQL)

1. What is the SELECT statement, and how is it used to query data?

SELECT column1, column2 FROM table_name;

2. Explain the use of the ORDER BY and WHERE clauses in SQL queries.

ORDER BY and WHERE Clauses:

- WHERE: Filters rows based on a condition.
- ORDER BY: Sorts the result in ascending (ASC) or descending (DESC) order.

9. Data Control Language (DCL)

1. What is the purpose of GRANT and REVOKE in SQL?

Purpose of GRANT and REVOKE:

- **GRANT:** Gives users permission to perform actions on database objects.
- **REVOKE:** Removes previously granted permissions.

2. How do you manage privileges using these commands?

Managing privileges:

- Use **GRANT** to allow specific operations (e.g., SELECT, INSERT).
- Use **REVOKE** to take back permissions when no longer needed.

10. Transaction Control Language (TCL)

1. What is the purpose of the COMMIT and ROLLBACK commands in SQL?

Purpose of COMMIT and ROLLBACK:

- COMMIT: Saves all changes made in the current transaction permanently.
- ROLLBACK: Undoes changes made in the current transaction.

2. Explain how transactions are managed in SQL databases.

Managing Transactions:

- A transaction is a sequence of SQL operations treated as a single unit.
- SQL ensures ACID properties: Atomicity, Consistency, Isolation, Durability.
- Use COMMIT to confirm changes, ROLLBACK to cancel if errors occur.

11. SQL Joins

1. Explain the concept of JOIN in SQL. What is the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN ?

Concept of JOIN and Types:

- JOIN: Combines rows from two or more tables based on a related column.
- 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 from both tables, with NULLs where there is no match.

2. How are joins used to combine data from multiple tables??

Using Joins:

- Joins merge data from multiple tables to get complete information in a single query.
- Example: Combining Students and Courses tables to show which student is in which course.

12. SQL Group By

1. What is the GROUP BY clause in SQL? How is it used with aggregate functions?

GROUP BY Clause:

- Purpose: Groups rows with the same values in specified columns.
- Use with Aggregate Functions: Allows calculations like SUM(), COUNT(), AVG() on each group.

2. Explain the difference between GROUP BY and ORDER BY.

Difference Between GROUP BY and ORDER BY:

- GROUP BY: Groups rows to perform aggregations.
- ORDER BY: Sorts rows without changing grouping.

13. SQL Stored Procedure

1. What is a stored procedure in SQL, and how does it differ from a standard SQL query?

Stored Procedure:

- A predefined set of SQL statements saved in the database.
- Differs from a standard query because it can take parameters, include logic, and be reused multiple times.

2. Explain the advantages of using stored procedures.

Advantages:

- Reusability: Write once, use many times.
- Improved performance: Precompiled and optimized by the database.
- Security: Can restrict direct access to tables.
- Maintainability: Easier to manage complex logic.

14. SQL View

1. What is a view in SQL, and how is it different from a table?

View in SQL:

- A virtual table based on a SQL query.
- Differs from a table because it does not store data physically, it shows data from underlying tables.

2. Explain the advantages of using views in SQL databases.

Advantages of Views:

- Simplifies complex queries.
- Enhances security by restricting column access.
- Provides consistent data representation.
- Can be used like a regular table in queries.

15. SQL Triggers

1. What is a trigger in SQL? Describe its types and when they are used.

Trigger in SQL:

- A **special procedure** that automatically executes in response to certain events on a table.
- **Types & Use:**
  - **BEFORE Trigger:** Executes before an action (e.g., validate data before insert).

  - **AFTER Trigger:** Executes after an action (e.g., update audit log after update).

2. Explain the difference between INSERT, UPDATE, and DELETE triggers

Difference Between Triggers:

- INSERT Trigger: Fires when a new row is added.
- UPDATE Trigger: Fires when existing data is modified.
- DELETE Trigger: Fires when a row is deleted.

16. Introduction to PL/SQL

1. What is PL/SQL, and how does it extend SQL's capabilities?

PL/SQL:

- Stands for Procedural Language/SQL.
- Extends SQL by adding procedural features like loops, conditions, and variables.

2. List and explain the benefits of using PL/SQL.

Benefits of PL/SQL:

- Block Structure: Organizes code into manageable blocks.
- Error Handling: Supports exceptions for robust programs.
- Reusability: Procedures, functions, and packages can be reused.
- Performance: Reduces network traffic by executing multiple SQL statements in a single block.

17. PL/SQL Control Structures

1. What are control structures in PL/SQL? Explain the IF-THEN and LOOP control structures.

Allow conditional execution and repetition in PL/SQL programs.

2. How do control structures in PL/SQL help in writing complex queries?

Role in Complex Queries:

- Enables conditional logic and iterative processing.
- Makes it possible to automate calculations, validations, and repetitive tasks in the database.

18. SQL Cursors

1. What is a cursor in PL/SQL? Explain the difference between implicit and explicit cursors.

Cursor in PL/SQL:

- A pointer that lets you retrieve and process rows from a query one at a time.

Types:

- Implicit Cursor: Automatically created by PL/SQL for single-row queries.
- Explicit Cursor: Defined by the programmer to handle multi-row queries.

2. When would you use an explicit cursor over an implicit one?

When to use Explicit Cursor:

- When you need to process multiple rows individually.
- When you want more control over fetching, looping, or closing the cursor.

19. Rollback and Commit Savepoint

1. Explain the concept of SAVEPOINT in transaction management. How do ROLLBACK and COMMIT interact with savepoints ?

SAVEPOINT in Transactions:

- Marks a **point within a transaction** to which you can later roll back.
- **ROLLBACK TO savepoint:** Undoes changes up to that point.
- **COMMIT:** Saves all changes, including those before the savepoint.

2. When is it useful to use savepoints in a database transaction?

Usefulness of Savepoints:

- Allows partial undo of complex transactions without affecting the entire transaction.
- Helps in error recovery and managing large operations safely.