Software engineering Module 4 – Introduction to DBMS

1. Introduction to SQL

Theory Questions:

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

Ans: SQL stands for 'Structured Query Language' it is a language which helps us to create, modify, maintain and filter data from databases and that is why it is important in database management.

2) Explain the difference between DBMS and RDBMS.

Ans:

DBMS	RDBMS
Here data is stored in file format	Here data is stored in table format
Here data can be store in small quantity	Here data can be stored in large quantity
It only supports single user	It supports multiply users
Ex- MS access, XML	Ex- Oracle, SQL server

3) Describe the role of SQL in managing relational database.

Ans: SQL helps to create relations between tables by primary keys, foreign keys and joins.

4) What are the key features of SQL?

Ans: Key features of SQL are: -

- It is fast and efficient
- It can store large quantity of data

- It can create relations between tables

Lab Exercises:

1) Create a new database named school_db and a table called students with the following columns: student_id, student_name, age, class, and address.

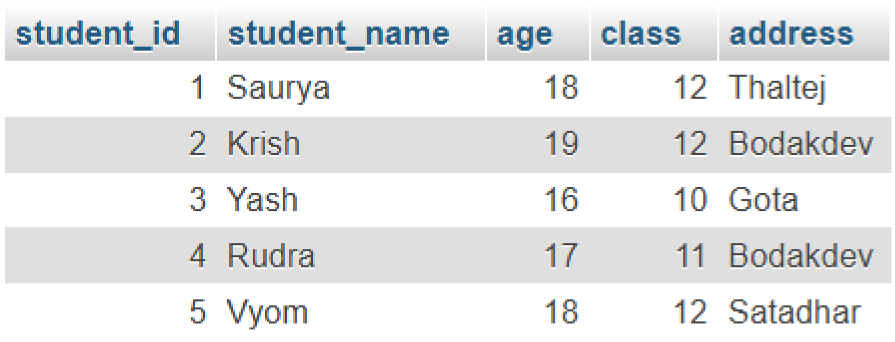
Ans:

```
CREATE DATABASE school_db;

CREATE TABLE students

(

   student_id INT,

   student_name VARCHAR(25),

   age INT,

   class INT,

   address TEXT

);
```

2) Insert five records into the students table and retrieve all records using the SELECT statement.

Ans:

```
INSERT INTO students VALUES(1, 'Saurya', 18, 12, 'Thaltej'), (2, 'Krish', 19, 12, 'Bodakdev'), (3, 'Yash', 16, 10, 'Gota'), (4 ,'Rudra', 17, 11, 'Bodakdev'), (5, 'Vyom', 18, 12, 'Satadhar');


SELECT * FROM students;
```



student_id	student_name	age	class	address
1	Saurya	18	12	Thaltej
2	Krish	19	12	Bodakdev
3	Yash	16	10	Gota
4	Rudra	17	11	Bodakdev
5	Vyom	18	12	Satadhar

2. SQL Syntax

Theory Questions:

1) What are the basic components of SQL syntax?

Ans: Basic components of SQL syntax are: -

- Command languages – Like DDL, DML, DQL, DCL and TCL
- Clauses – like WHERE, HAVING, GROUP BY, etc

2) Write general structure of an SQL SELECT STATEMENT.

Ans: SELECT <column_names> FROM <table_name>

3) Explain the role of clauses in SQL statements.

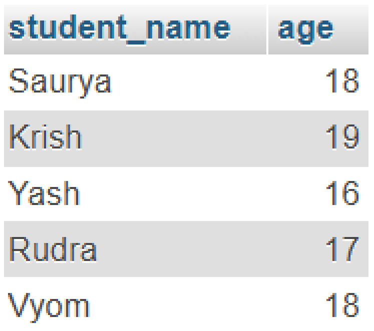
Ans: Clauses act as modifiers to query which help access specific data have more control. For example – WHERE helps to filter data, GROUP BY helps to create groups of same kind of data.

Lab Exercises:

1) Write SQL queries to retrieve specific columns (student_name and age) from the students table.

Ans:

SELECT student_name, age FROM students;

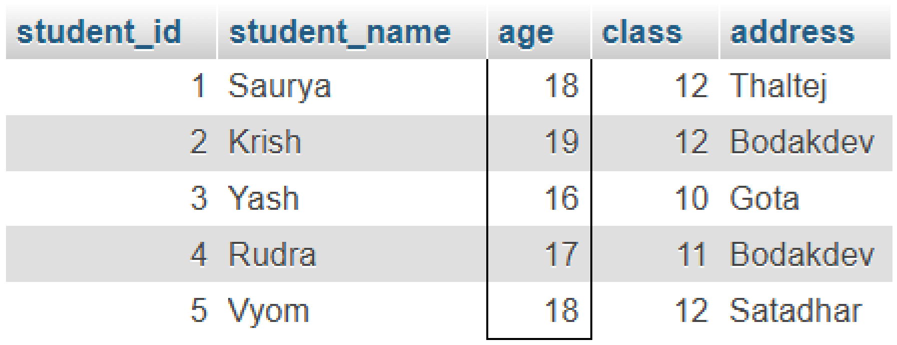


student_name	age
Saurya	18
Krish	19
Yash	16
Rudra	17
Vyom	18

2) Write SQL queries to retrieve all students whose age is greater than 10.

Ans:

SELECT * FROM students WHERE age > 10;



student_id	student_name	age	class	address
1	Saurya	18	12	Thaltej
2	Krish	19	12	Bodakdev
3	Yash	16	10	Gota
4	Rudra	17	11	Bodakdev
5	Vyom	18	12	Satadhar

3. SQL Constraints

Theory Questions:

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

Ans: Constraints help to limit the type of data we can enter into a table. Different types of constraints are: -

- NOT NULL – The value cannot be left empty the user has to compulsory provide a value. For example – employee ID, employee salary, etc
- UNIQUE – The value cannot be repeated i. e. once the value is entered in the column no other record can have the same value in that column
- PRIMARY KEY – It is a combination of NOT NULL and UNIQUE constraints. It is also used to create relations between tables.
- FOREIGN KEY – It helps to create relation with another table.
- CHECK – It allow the value to entered only if the given condition is true
- DEFAULT – It assigns a default value to the column i. e. if no value is assigned to the column while entering a record, then the value set to the default value.

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

Ans:

PRIMARY KEY	FOREIGN KEY
It contain unique values	It can contain duplicate values
It cannot contain null values	It can contain null values
It is used to identify the records in a table uniquely.	It is used to make a relation between two tables

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

Ans: NOT NULL ensures that the value is entered and not left blank. UNIQUE ensures that value entered is not present in any other record entered before.

Lab Exercise:

1) Create a table teachers with the following columns: teacher_id (Primary Key), teacher_name (NOT NULL), subject (NOT NULL), and email (UNIQUE).

Ans:

```
CREATE TABLE teachers

(

   teacher_id INT PRIMARY KEY,

   teacher_name VARCHAR(25) NOT NULL,

   subject TEXT NOT NULL,

   email TEXT UNIQUE

);
```

2) Implement a FOREIGN KEY constraint to relate the teacher_id from the teachers table with the students table.

Ans:

```
ALTER TABLE students ADD (teacher_id int, FOREIGN KEY (teacher_id)
REFERENCES teachers(teacher_id));
```

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

Theory Questions:

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

Ans: DDL commands are used on table to create, alter, drop and truncate the table. These are used to modify the structure of a table

2) Explain the CREATE command and its syntax.

Ans: CREATE command is used when you need to create anything in SQL like database, table, procedure, trigger, view. Its syntax is: -

```
CREATE TABLE <table_name>
```

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

Ans: The purpose of specifying data types and constrains is to limit the data that can be entered into a table so that the data is not inaccurate, misleading or ambiguous.

Lab Exercise:

1) Create a table courses with columns: course_id, course_name, and course_credits. Set the course_id as the primary key.

Ans:

```
CREATE TABLE courses (
   course_id int,
   course_name text,
   course_credits int,
   PRIMARY KEY (course_id)
);
```

2) Use the CREATE command to create a database university_db.

Ans:

```
CREATE DATABASE university_db;
```

5. ALTER Command

Theory Questions:

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

Ans: ALTER command is used whenever we need to makes changes to structure of the table like changing the datatypes or constrains.

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

Ans :

To add – ALTER TABLE <table_name> ADD (<column_name> <datatype> <constrain>);

To modify - ALTER TABLE <table_name> MODIFY (<column_name> <new datatype> <new constrain>);

To drop – ALTER TABLE <table_name> DROP <column_name>;

Lab Exercises:

1) Modify the courses table by adding a column course_duration using the ALTER command.

Ans:

ALTER TABLE courses ADD (course_duration int);

2) Drop the course_credits column from the courses table.

Ans:

ALTER TABLE courses DROP course_credits;

6. DROP Command

Theory Questions:

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

Ans: The function of DROP command is to delete any column or a complete table or database

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

Ans: Dropping a table means to completely remove the data of that table from the database.

Lab Exercise:

1) Drop the teachers table from the school_db database.

Ans:

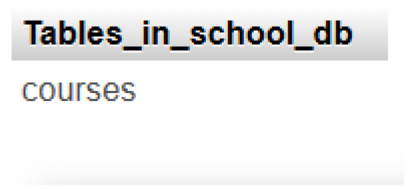
DROP TABLE teachers;

2) Drop the students table from the school_db database and verify that the table has been removed.

Ans:

DROP TABLE students;

SHOW tables;



```
Tables_in_school_db
courses
```

7. Data Manipulation Language (DML)

Theory Questions:

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

Ans:

INSERT – It is used add new data in the table

UPDATE – It is used to modify the existing data of the table

DELETE – It is used to delete data from the table

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

Ans: WHERE clause in important while using UPDATE and DELETE commands as it helps to modify specific data from the table without it the UPDATE or DELETE would apply to the whole column.

Lab Exercises:

1) Insert three records into the courses table using the INSERT command.

Ans:

```
INSERT INTO courses VALUES(1, 'Python', 4), (2, 'Java', 5), (3, 'PHP', 6);
```

2) Update the course duration of a specific course using the UPDATE command.

Ans:

```
UPDATE courses SET course_duration = 3 WHERE course_name = 'PHP';
```

3) Delete a course with a specific course_id from the courses table using the DELETE command.

Ans:

```
DELETE FROM courses WHERE course_id = 3;
```

8. Data Query Language

Theory Questions:

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

Ans: SELECT statement is used to filter specific data from the table in a query it used to filter out the column which aren't necessary for that query.

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

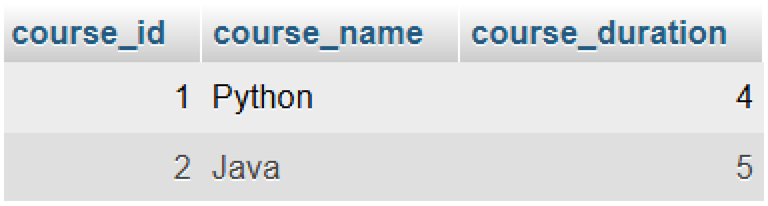
Ans: Using WHERE helps us to filter specific records from the table and ORDER BY helps to sort the data (by default ORDER BY sorts data in ascending order)

Lab Exercises:

1) Retrieve all courses from the courses table using the SELECT statement.

Ans:

SELECT * FROM courses;

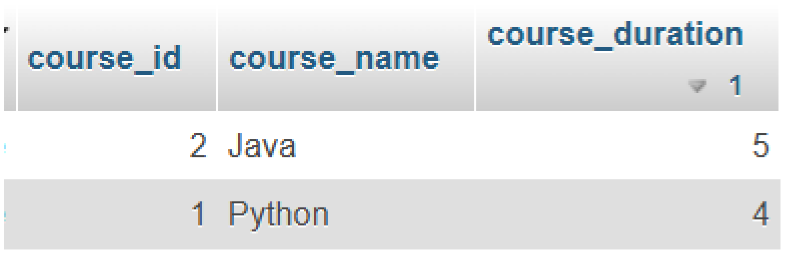


course_id	course_name	course_duration
1	Python	4
2	Java	5

2) Sort the courses based on course_duration in descending order using ORDER BY.

Ans:

SELECT * FROM courses ORDER BY course_duration DESC;

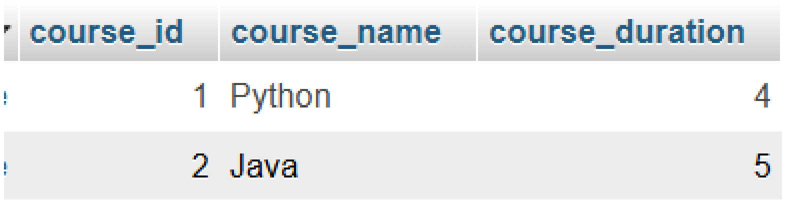


course_id	course_name	course_duration ▾ 1
2	Java	5
1	Python	4

3) Limit the results of the SELECT query to show only the top two courses using LIMIT.

Ans:

SELECT * FROM courses LIMIT 2;



course_id	course_name	course_duration
1	Python	4
2	Java	5

9. Data Control Language (DCL)

Theory Questions:

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

Ans: GRANT is used to give access of specific commands to other users. REVOKE is used to take back the access from the users.

2) How do you manage privileges using these commands?

Ans: Using GRANT we can give privileges to other users to use specific commands and when we want to take back the privileges we use the REVOKE command.

Lab Exercises:

1) Create two new users user1 and user2 and grant user1 permission to SELECT from the courses table.

Ans:

```
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password1';

CREATE USER 'user2'@'localhost' IDENTIFIED BY 'password2';

GRANT SELECT ON school.courses TO 'user1'@'localhost';
```

2) Revoke the INSERT permission from user1 and give it to user2.

Ans:

```
REVOKE INSERT ON your_database.courses FROM 'user1'@'localhost';

GRANT INSERT ON your_database.courses TO 'user2'@'localhost';
```

10, Transaction Control Language (TCL)

Theory Questions:

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

Ans: COMMIT is used to permanently store changes made and ROLLBACK is used to undo the last change made.

2) Explain how transactions are managed in SQL databases.

Ans: In SQL transactions are managed using COMMIT, ROLLBACK and SAVEPOINT commands. COMMIT is used to permanently apply changes, SAVEPOINT is used to create a reference for ROLLBACK so all the changes after the SAVEPOINT can be undone and ROLLBACK is used to undo the last change made.

Lab Exercises:

1) Insert a few rows into the courses table and use COMMIT to save the changes.

Ans:

```
START TRANSACTION;

INSERT INTO courses VALUES (3, 'PHP', 6), (4, '.NET', 7);

COMMIT;
```

2) Insert additional rows, then use ROLLBACK to undo the last insert operation.

Ans:

```
START TRANSACTION;

INSERT INTO courses VALUES

(5, 'Machine Learning', 4),

(6, 'Cloud Computing', 6);

ROLLBACK;
```

3) Create a SAVEPOINT before updating the courses table, and use it to roll back specific changes.

Ans:

```
START TRANSACTION;

SAVEPOINT before_changes;

INSERT INTO courses VALUES (3, 'PHP', 6), (4, '.NET', 7);

UPDATE courses SET course_duration = 10 WHERE course_id = 5;

ROLLBACK TO SAVEPOINT before_changes;
```

11, SQL Joins

Theory Questions:

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

Ans: JOIN is used to merge two or more tables who have a relation.

INNER JOIN – It returns all common data from both of the tables

LEFT JOIN – It returns all common data from both of the tables and all the data of the left table.

RIGHT JOIN – It returns all common data from both of the tables and all the data of the right table.

FULL OUTER JOIN – It returns all data except from the common data of the both tables.

2) How are JOIN used to combine data from multiple tables?

Ans: The syntax to use JOINs is:-

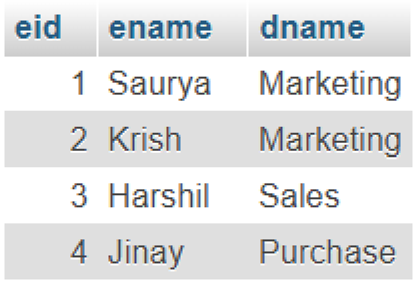
SELECT <column_names (from both tables)> FROM <table1_name> <JOIN_TYPE> <table2_name> ON <table1_primary_key> = <table2_foreign_key> ;

Lab Exercises:

1) Create two tables: departments and employees. Perform an INNER JOIN to display employees along with their respective departments.

Ans:

```
CREATE TABLE departments(
   did INT PRIMARY KEY,
    dname TEXT NOT NULL
);

CREATE TABLE employees(
    eid INT PRIMARY KEY,
    ename TEXT,
    did INT,
    FOREIGN KEY(did) REFERENCES departments(did)
);

INSERT INTO departments VALUES (1, 'Purchase'), (2, 'Sales'), (3,
'Finance'), (4, 'Marketing'), (5, 'Legal');

INSERT INTO employees VALUES (1, 'Saurya', 4), (2, 'Krish', 4), (3,
'Harshil', 2), (4, 'Jinay', 1);

SELECT employees.eid, employees.ename, departments.dname FROM
employees JOIN departments ON employees.did = departments.did;
```

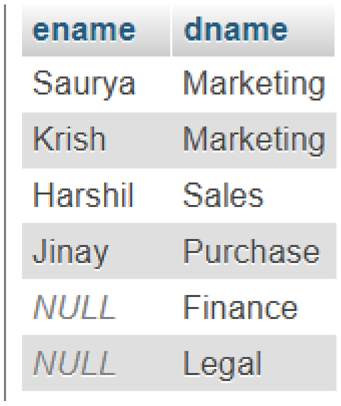


eid	ename	dname
1	Saurya	Marketing
2	Krish	Marketing
3	Harshil	Sales
4	Jinay	Purchase

2) Use a LEFT JOIN to show all departments, even those without employees.

Ans:

SELECT employees.ename, departments.dname FROM departments LEFT JOIN employees ON employees.did = departments.did;



ename	dname
Saurya	Marketing
Krish	Marketing
Harshil	Sales
Jinay	Purchase
NULL	Finance
NULL	Legal

12. SQL Group By

Theory Questions:

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

Ans: GROUP BY clause is used make a group of common data. Syntax to GROUP BY example to calculate average salary of different departments: -

SELECT departments, AVG (salary) FROM employees GROUP BY department;

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

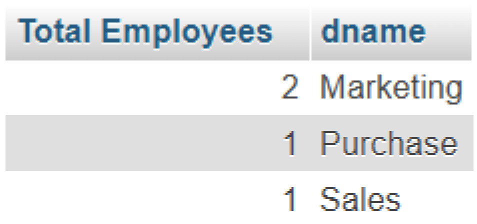
Ans: GROUP BY is used to group data whereas ORDER BY is used to sort data in ascending or descending order.

Lab Exercise:

1) Group employees by department and count the number of employees in each department using GROUP BY.

Ans:

SELECT COUNT(employees.eid) AS "Total Employees", departments.dname FROM employees JOIN departments on employees.did = departments.did GROUP BY departments.dname;

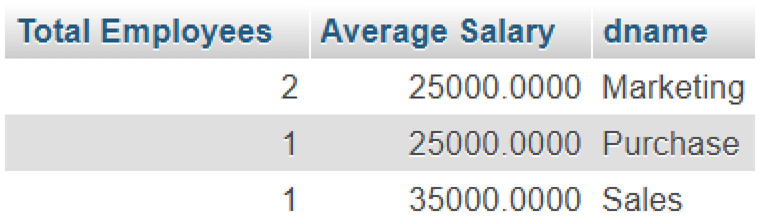


Total Employees	dname
2	Marketing
1	Purchase
1	Sales

2) Use the AVG aggregate function to find the average salary of employees in each department.

Ans:

SELECT COUNT(employees.eid) AS "Total Employees", AVG(salary) as "Average Salary", departments.dname FROM employees JOIN departments on employees.did = departments.did GROUP BY departments.dname;



Total Employees	Average Salary	dname
2	25000.0000	Marketing
1	25000.0000	Purchase
1	35000.0000	Sales

13. SQL Stored Procedure

Theory Questions:

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

Ans: Stored procedure is like a function when you call it is forms specific queries which written while creating the procedure. It is different from a standard SQL query as it only needs to be defined once and then you use it again anytime you want to whereas you need to write the query again every time you want to use it.

2) Explain the advantages of using stored procedures.

Ans: Procedures can be used again and again after defining them once so with help of a procedure you won’t need to rewrite the same query every time.

Lab Exercise:

1) Write a stored procedure to retrieve all employees from the employees table based on department.

Ans:

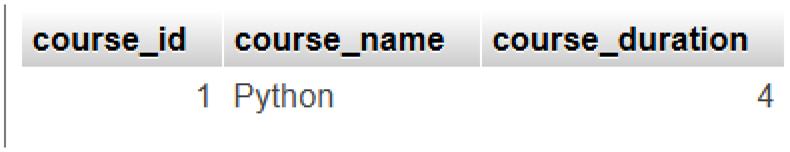
```
DELIMITER $$

CREATE PROCEDURE find_emp(dep_id int)

BEGIN

SELECT employees.eid, employees.ename, employees.salary,
departments.did FROM employees JOIN departments on employees.did
= departments.did HAVING did = dep_id;

END;

CALL find_emp(1);
```

2) Write a stored procedure that accepts course_id as input and returns the course details.

Ans:

```
DELIMITER $$
```

```
CREATE PROCEDURE get_course(id int)
BEGIN
        SELECT * FROM courses WHERE course_id = id;
end;
CALL get_course(1);
```



```
course_id  course_name  course_duration
        1  Python                     4
```

14. SQL View

Theory Questions:

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

Ans: View is a virtual table which has specific data from one or more tables. It is different from table as views aren't stored in the database they are created through queries.

2) Explain the advantages of using views In SQL databases.

Ans: Advantages of using views in SQL databases are: -

- As it can combine two or more tables, it is easier to run queries on those as we won't write join queries which are complex
- As we can choose which columns to show from which tables we can make sensitive data inaccessible.
- For frequently used queries we can create a view so that every time we won't need to rewrite the query.

Lab Exercise:

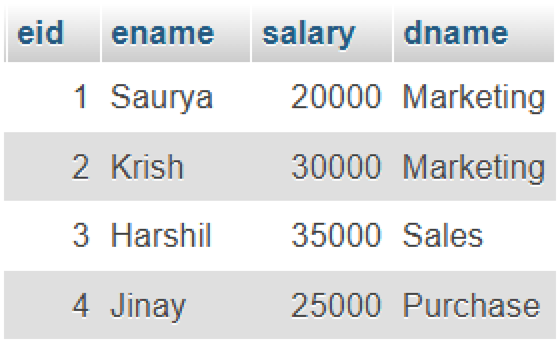
1) Create a view to show all employees along with their department names.

Ans:

CREATE VIEW emp_deparments AS

SELECT employees.eid, employees.ename, employees.salary, departments.dname FROM employees JOIN departments on employees.did = departments.did;

SELECT * FROM emp_deparments;



eid	ename	salary	dname
1	Saurya	20000	Marketing
2	Krish	30000	Marketing
3	Harshil	35000	Sales
4	Jinay	25000	Purchase

2) Modify the view to exclude employees whose salaries are below $50,000.

Ans:

CREATE OR REPLACE VIEW emp_deparments AS

SELECT employees.eid, employees.ename, employees.salary, departments.dname FROM employees JOIN departments on employees.did = departments.did

WHERE employees.salary < 50000;

15. SQL Triggers

Theory Questions:

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

Ans: Triggers are special type of stored procedure that automatically execute when an event occurs, triggers can be defined for only DML commands. Trigger are of 2 types

- Before Triggers – These are execute before the triggering DML operation occurs, allowing you to modify the data being inserted, updated or deleted before it is committed to the database.
- After Triggers – These triggers execute after the DML operation, allowing for complete control over how data is modified.

2) Explain the difference between INSERT, UPDATE and DELETE trigger.

Ans:

INSERT Trigger	UPDATE Trigger	DELETE Trigger
Runs whenever a new record in inserted into a table	Runs whenever an existing record is updated in a table	Runs whenever a record is delete from the table
Accesses the newly inserted record	Can access both the old data and the updated one	Access the old record that is deleted

Lab Exercise:

1) Create a trigger to automatically log changes to the employees table when a new employee is added.

Ans:

```
CREATE TABLE employees_history(
   dep_id int,
   emp_id int,

   name text,

   salary int,

   time_changed timestamp,

   action_performed text

);

DELIMITER $$

CREATE TRIGGER insert_trigger AFTER INSERT ON employees FOR EACH
ROW

BEGIN

        INSERT INTO employees_history(dep_id, emp_id, name, salary,
action_performed) VALUES(new.did, new.eid, new.ename, new.salary,
'Record Inserted');

END;
```

2) Create a trigger to update the last_modified timestamp whenever an employee record in updated.

Ans:

```
CREATE TABLE emp_update_history(

   eid int,

   enmae text,

   salary int,

   last_modified timestamp,

   did int
);

DELIMITER $$

CREATE TRIGGER update_trig AFTER UPDATE ON employees FOR EACH ROW

BEGIN

	INSERT INTO emp_update_history(eid, ename, salary, did) VALUES(new.eid, new.ename, new.salary, new.did);

END;
```

16. Introduction to PL/SQL
Theory Questions:

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

Ans: PL/SQL is a programming language that helps us to combine SQL with some of procedural programming constructs. It introduces new features like conditionals and loops into SQL.

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

Ans: Benefits of using PL/SQL are:

- Integration with SQL
- High performance
- High productivity
- Portability

Lab Exercises:

1) Write a PL/SQL block to print the total number of employees from the employees table.

Ans:

```
DECLARE
   Employees_total NUMBER;
BEGIN
SELECT COUNT(*) INTO employees_total FROM employees;
DBMS_OUTPUT.PUT_LINE('Total Number of Employees: ' ||
employees_total);
END;
```

2) Create a PL/SQL block that calculates the total sales from an orders table.

Ans:

```
DECLARE
   total_sales NUMBER;
BEGIN
   SELECT SUM(order_amount) INTO total_sales FROM orders;
   IF v_total_sales IS NULL THEN
      v_total_sales := 0;
   END IF;
   DBMS_OUTPUT.PUT_LINE('Total Sales: ' || total_sales);
END;
```

17. PL/SQL Control Structures

Theory Questions:

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

Ans: Control statements are used to change flow of execution of the PL/SQL block.

- IF-THEN is used to run a specific block only if the condition given is true.
- LOOP is used to repeat a task over and over again.

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

Ans: Control structures in PL/SQL like IF-THEN, LOOPS allow developers to create complex, dynamic queries by using programming logic in a PL/SQL block, effectively handling scenarios where a simple SQL statement wouldn't be enough.

Lab Exercises:

1) Write a PL/SQL block using an IF-THEN condition to check the department of an employee.

Ans:

```
DECLARE
   v_employee_id   NUMBER := 101;
   v_department_id NUMBER;
BEGIN
   SELECT department_id INTO v_department_id
   FROM employees
   WHERE employee_id = v_employee_id;
   IF v_department_id = 10 THEN
      DBMS_OUTPUT.PUT_LINE('Employee ' || v_employee_id || ' works in the HR department.');
   ELSIF v_department_id = 20 THEN
      DBMS_OUTPUT.PUT_LINE('Employee ' || v_employee_id || ' works in the Sales department.');
   ELSE
      DBMS_OUTPUT.PUT_LINE('Employee ' || v_employee_id || ' works in another department.');
   END IF;
END;
```

2) Use a FOR LOOP to iterate through employee records and display their names.

Ans:

```
DECLARE
   CURSOR emp_cursor IS
      SELECT employee_id, first_name, last_name FROM employees;
BEGIN
   FOR emp_rec IN emp_cursor LOOP
      DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_rec.employee_id ||
                           ', Name: ' || emp_rec.first_name || ' ' || emp_rec.last_name);
   END LOOP;
END;
```

18. SQL Cursors

Theory Questions:

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

Ans: Cursor is a pointer to a result set returned by a SELECT statement, allowing to access the data one row at a time.

Implicit cursor is automatically created when SELECT statement is executed. Explicit cursor is defined by the user.

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

Ans: We use an explicit cursor over an implicit one when we need to process multiple rows returned by the SELECT statement individually.

Explicit cursor gives more control over the data retrieval process by allowing you to fetch each row one at a time within a loop, which implicit cursor can't do.

Lab Exercise:

1) Write a PL/SQL block using an explicit cursor to retrieve and display employee details.

Ans:

```
DECLARE
   CURSOR emp_cursor IS
      SELECT employee_id, first_name, last_name, department_id, salary
```

```
        FROM employees;

    emp_rec emp_cursor%ROWTYPE;
BEGIN
    OPEN emp_cursor;

    LOOP
        FETCH emp_cursor INTO emp_rec;
        EXIT WHEN emp_cursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Employee ID: ' ||
emp_rec.employee_id ||
                            ', Name: ' || emp_rec.first_name || ' ' ||
emp_rec.last_name ||
                            ', Department ID: ' || emp_rec.department_id ||
                            ', Salary: ' || emp_rec.salary);
    END LOOP;
    CLOSE emp_cursor;
END;
```

2) Create a cursor to retrieve all courses and display them one by one.

Ans:

```
DECLARE
    CURSOR course_cursor IS
```

```
        SELECT course_id, course_name, instructor FROM courses;

    course_rec course_cursor%ROWTYPE;
BEGIN
    OPEN course_cursor;

    LOOP
        FETCH course_cursor INTO course_rec;
        EXIT WHEN course_cursor%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Course ID: ' || course_rec.course_id ||
                             ', Course Name: ' || course_rec.course_name ||
                             ', Instructor: ' || course_rec.instructor);
    END LOOP;

    CLOSE course_cursor;
END;
```

19. Rollback and Commit Savepoint

Theory Questions:

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

Ans: SAVEPOINT marks a place within a large transaction allowing you to rollback to that point in transaction, effectively undoing changes made the after the savepoint.

The COMMIT and ROLLBACK statement releases all savepoints names established within a transactions.

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

Ans: When doing a series of operations within a transaction where an error might occur in one step, you can use a savepoint to rollback to that problematic step while keeping the changes made in previous steps.

Lab Exercises:

1) Perform a transaction where you create a savepoint, insert records, then rollback to the savepoint.

Ans:

```
INSERT INTO employees (employee_id, first_name, last_name, department_id, salary)
VALUES (5001, 'John', 'Doe', 10, 50000);

SAVEPOINT before_insertion;

INSERT INTO employees (employee_id, first_name, last_name, department_id, salary)
VALUES (5002, 'Jane', 'Smith', 20, 60000);
```

```
INSERT INTO employees (employee_id, first_name, last_name,
department_id, salary)
VALUES (5003, 'Alice', 'Johnson', 30, 55000);

ROLLBACK TO before_insertion;
COMMIT;
```

2) Commit part of a transaction after using a savepoint and then rollback the remaining changes.

Ans:

```
INSERT INTO employees (employee_id, first_name, last_name,
department_id, salary)
VALUES (6001, 'Tom', 'Anderson', 10, 55000);

SAVEPOINT before_more_inserts;

INSERT INTO employees (employee_id, first_name, last_name,
department_id, salary)
VALUES (6002, 'Emma', 'Brown', 20, 60000);

INSERT INTO employees (employee_id, first_name, last_name,
department_id, salary)
VALUES (6003, 'Liam', 'Wilson', 30, 65000);
```

```
COMMIT;

ROLLBACK TO before_more_inserts;
```