ASSIGNMENT-6

TITLE

To perform SQL queries that demonstrate **Join Queries** (Inner Join, Outer Join, Left Join, Right Join) and **TCL (Transaction Control Language) Commands**.

PROBLEM STATEMENT

Perform SQL operations to understand how to combine data from multiple tables using different types of join queries (Inner Join, Outer Join, Left Join, Right Join) and manage transactions using TCL commands.
This exercise will help in understanding relational database concepts, data retrieval across multiple tables, and transaction control for maintaining data integrity.

OBJECTIVES

- To understand the concept of joins in SQL.
- To learn how to perform **Inner Join** to retrieve matching records.
- To learn how to perform **Left Join** and **Right Join** to include unmatched records.
- To understand **Outer Join** for retrieving all records from both tables.
- To use **TCL commands** like COMMIT, ROLLBACK, and SAVEPOINT for transaction management.
- To observe the effect of transaction control on database operations.

PLATFORM REQUIRED

- **Operating System**: Windows/Linux
- **Software/Tools**: MySQL, Oracle, or any SQL environment (MySQL Workbench, XAMPP, SQL Command Line).

THEORY

Joins in SQL

A **JOIN** is used to combine rows from two or more tables based on a related column between them.

1. **Inner Join**
    - Returns only the rows with matching values in both tables.
2. SELECT s.Student_ID, s.Name, c.Course_Name

3. FROM Student s
4. INNER JOIN Course c ON s.Student_ID = c.Student_ID;
5. **Left Join (Left Outer Join)**
    - Returns all rows from the left table, and matching rows from the right table.
6. SELECT s.Student_ID, s.Name, c.Course_Name
7. FROM Student s
8. LEFT JOIN Course c ON s.Student_ID = c.Student_ID;
9. **Right Join (Right Outer Join)**
    - Returns all rows from the right table, and matching rows from the left table.
10. SELECT s.Student_ID, s.Name, c.Course_Name
11. FROM Student s
12. RIGHT JOIN Course c ON s.Student_ID = c.Student_ID;
13. **Full Outer Join**
    - Returns all rows when there is a match in one of the tables.
      (Note: In MySQL, FULL OUTER JOIN is not directly supported, but can be simulated using UNION.)
14. SELECT s.Student_ID, s.Name, c.Course_Name
15. FROM Student s
16. LEFT JOIN Course c ON s.Student_ID = c.Student_ID
17. UNION
18. SELECT s.Student_ID, s.Name, c.Course_Name
19. FROM Student s
20. RIGHT JOIN Course c ON s.Student_ID = c.Student_ID;

TCL (Transaction Control Language) Commands

- **COMMIT** → Saves the changes permanently in the database.
- **ROLLBACK** → Restores the database to the last committed state.
- **SAVEPOINT** → Sets a point within a transaction to which you can later roll back.

Example:

```
START TRANSACTION;

UPDATE Student SET Address = 'Delhi' WHERE Student_ID = 1;
```

```
SAVEPOINT sp1;
DELETE FROM Course WHERE Course_ID = 102;
ROLLBACK TO sp1;   -- Undo delete but keep update
COMMIT;            -- Save changes permanently
```

SCENARIO EXAMPLE

Tables:

Student Table

Student_ID	Name	Address
1	Amit	Pune
2	Priya	Mumbai
3	Rahul	Nagpur

Course Table

Course_ID	Student_ID	Course_Name
101	1	DBMS
102	2	AI
103	4	Java

- **Inner Join Example** → Matches only Amit and Priya.
- **Left Join Example** → Shows all Students, including Rahul (with NULL for Course).
- **Right Join Example** → Shows all Courses, including Java (with NULL for Student).

STEP-BY-STEP ALGORITHM

1. Open SQL environment and connect to the server.
2. Create two tables: **Student** and **Course**.
3. Insert sample data into both tables.
4. Perform **INNER JOIN** between Student and Course tables.
5. Perform **LEFT JOIN** and observe unmatched records.
6. Perform **RIGHT JOIN** and observe unmatched records.
7. Perform **FULL OUTER JOIN** (using UNION).

8. Start a transaction using START TRANSACTION;.
9. Perform an **UPDATE** or **DELETE** operation.
10. Use SAVEPOINT to mark a transaction point.
11. Use ROLLBACK to undo changes up to a savepoint.
12. Use COMMIT to save changes permanently.

QUESTIONS FOR PRACTICE

1. Write an SQL query to perform an **Inner Join** on Student and Course tables.
2. What is the difference between **Left Join** and **Right Join**?
3. How can you simulate a **Full Outer Join** in MySQL?
4. Write an SQL query to show all students, even if they are not enrolled in any course.
5. Explain the role of **COMMIT** and **ROLLBACK** in TCL.
6. What will happen if you do not use COMMIT after an update inside a transaction?
7. Write an example using SAVEPOINT and ROLLBACK.

CONCLUSION

In this lab, you have learned how to perform **Join Queries** (Inner, Left, Right, and Outer Joins) to combine data from multiple tables. You also explored **TCL commands** (COMMIT, ROLLBACK, SAVEPOINT) to manage database transactions. These concepts are essential for ensuring accurate data retrieval across related tables and maintaining **data integrity** during operations. Mastery of Joins and TCL provides the foundation for advanced query writing and robust database management in real-world applications.