Task 5 **Implementation of different types of Joins**

Aim: To Perform the advanced query processing and test its heuristics using the designing of optimizing complex queries and their equivalence queries.

Procedures:

SQL JOIN

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

Types of JOIN

1. Simple Join
    1. Equi-join :
    2. NonEqui-join :
2. Self-Join Query
3. Outer Join
    1. Inner Join
    2. Left Outer Join
    3. Right Outer Join
    4. Full Outer Join

The join is actually performed by the 'where' clause which combines specified rows of tables.

Simple Join:

It is the most common type of join. It retrieves the rows from 2 tables having a common column and is further classified into 2 joins.

1. **Equi-join :**

A join, which is based on **equalities**, is called equi-join.

2. **Non Equi-join:**

It specifies the relationship between columns belonging to different tables by making use of relational **operators other than'='.**

Self join:

Joining of a table to itself is known as self-join. It joins one row in a table to

another. It can compare each row of the table to itself and also with other rows of the same table.

Outer Join:

- It extends the result of a simple join. An outer join returns all the rows returned by simplejoin as well as those rows from one table that do not match any row from the table.
- The symbol(+) represents outer join.

Outer Join

- Inner Join
- Left Join
- Right Join
- Full Outer Join

- (INNER) JOIN: Returns records that have matching values in both tables
- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
- RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
- FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

Questions:

Aim : To Perform the advanced query processing and test its heuristics using designing of optimal correlated and nested sub queries such as finding summary statistics.

1. To retrieve all cricket boards and their teams.

2. To list all matches along with the teams and their captains.

3. To count the number of matches played by each team.

4. To find all the players who are part of the team named "Nellai Royal Kings".

5. To find all the matches played by a specific player.

Table Names:

- CricketBoard(BoardID, BoardName, ...)
- Team(TeamID, TeamName, BoardID, CaptainID, ...)
- Player(PlayerID, PlayerName, TeamID, ...)
- Match(MatchID, Team1ID, Team2ID, MatchDate, Location, ...)
- Match_Player(MatchID, PlayerID) ← junction table (if many-to-many participation)

1. Retrieve all cricket boards and their teams

-- Step 1: Select BoardName from CricketBoard table

-- Step 2: Select TeamName from Team table

-- Step 3: Join both tables using BoardID (foreign key in Team)

-- Step 4: Display board names with their corresponding teams

SELECT cb.BoardName, t.TeamName FROM CricketBoard cb JOIN Team t ON cb.BoardID = t.BoardID;

2. List all matches along with the teams and their captains

-- Step 1: Select MatchID from Match table

-- Step 2: Join with Team table to get Team1 and Team2

-- Step 3: From Team table, get CaptainID for each team

-- Step 4: Join with Player table to get captain names

-- Step 5: Display match, team names, and their captains

SELECT m.MatchID, t1.TeamName AS Team1, p1.PlayerName AS Captain1, t2.TeamName AS Team2, p2.PlayerName AS Captain2 FROM Match m JOIN Team t1 ON m.Team1ID = t1.TeamID

JOIN Player p1 ON t1.CaptainID = p1.PlayerID

JOIN Team t2 ON m.Team2ID = t2.TeamID

JOIN Player p2 ON t2.CaptainID = p2.PlayerID;

3. Count the number of matches played by each team

-- Step 1: Select TeamName from Team table

-- Step 2: Join with Match table on TeamID appearing in Team1ID or Team2ID

-- Step 3: Count the number of matches per team using COUNT()

-- Step 4: Group results by TeamName

SELECT t.TeamName, COUNT(m.MatchID) AS MatchesPlayed

FROM Team t JOIN Match m ON t.TeamID IN (m.Team1ID, m.Team2ID) GROUP BY t.TeamName;

---

4. Find all the players who are part of the team named "Nellai Royal Kings"

-- Step 1: Select TeamID from Team table where TeamName = 'Nellai Royal Kings'

-- Step 2: Join Player table on TeamID

-- Step 3: Retrieve PlayerName of all players in that team

SELECT p.PlayerName FROM Player p JOIN Team t ON p.TeamID = t.TeamID

WHERE t.TeamName = 'Nellai Royal Kings';

5. Find all the matches played by a specific player (example: 'MS Dhoni')

If a MatchPlayer mapping table exists:

-- *Step 1: Input player name (e.g., 'MS Dhoni')*

-- Step 2: Find PlayerID from Player table

-- Step 3: Use MatchPlayer to check all matches the player participated in

-- Step 4: Join with Match table to retrieve match details

SELECT DISTINCT m.MatchID, m.Date, m.Venue FROM Match m JOIN MatchPlayer mp ON m.MatchID = mp.MatchID JOIN Player p ON mp.PlayerID = p.PlayerID WHERE p.PlayerName = 'MS Dhoni';

6. If only team-based participation is tracked:

-- *Step 1: Input player name (e.g., 'MS Dhoni')*

-- Step 2: Find the TeamID of this player

-- Step 3: Retrieve all matches where TeamID = Team1ID or Team2ID

-- Step 4: Display match details

SELECT DISTINCT m.MatchID, m.Date, m.Venue FROM Match m JOIN Team t ON t.TeamID IN (m.Team1ID, m.Team2ID) JOIN Player p ON p.TeamID = t.TeamID

WHERE p.PlayerName = 'MS Dhoni';