**ExperimentNo. 1.7**

**Student Name:** Gaurav Kumar **UID:** 22MCC20177

**Branch:** MCA**–**CCD **Section/Group:** MCD-1/A

**Semester:** III **Date of Performance:** 15th Oct 23

**Subject Name:** Business Analytics **Subject Code:** 22CAH-703

1. **Aim/Overview of the practical:**
   1. Execute the JOIN in SQL by taking a table as an example. Explain the differences between an INNER JOIN and a LEFT JOIN in SQL.
   2. Describe the concept of a self-join in SQL. Provide an example scenario where a self-join would be useful.
2. **Code for practical: (a)**
3. To perform the join operation first create at least 2 tables.

A screenshot of a computer

Description automatically generated

1. To perform join on 2 tables we can use **JOIN** keyword to join and **ON** keyword to specify relation on which the join is perform.

A screenshot of a computer

Description automatically generated

**INNER JOIN:** An INNER JOIN returns only the rows where there is a match in both tables. In this case, it will retrieve rows where students have corresponding records in the "student\_marks" table. **Note:** **join** and **Inner join** both are same.

**LEFT JOIN:** A LEFT JOIN returns all rows from the left table (the "student" table) and the matched rows from the right table (the "student\_marks" table). If there's no match in the right table for a row in the left table, the result will include NULL values for the right table's columns.

**A screenshot of a computer

Description automatically generated**

**Code for practical: (b)**

**SELF JOIN:** A self-join in SQL is a specific type of join operation where a table is joined with itself. In other words, it's a way to combine rows from a single table based on a related column within that same table. Self-joins are particularly useful when you have hierarchical or recursive data structures or when you want to establish relationships between rows in the same table.

**Example Scenario:** Let's consider an example scenario where a self-join would be useful: representing an organizational hierarchy. You have an "employees" table that includes data about employees and their managers. The table might have the following structure:

**A screenshot of a computer

Description automatically generated**

To visualize the organizational hierarchy, you can use a self-join to connect employees with their managers. For example, to retrieve the names of employees and their managers, you can write a self-join query like this:

A screenshot of a computer

Description automatically generated

In this example, self-join helps establish relationships between employees and their managers, making it easier to represent and understand the organizational hierarchy within the company. Self-joins are particularly valuable for handling hierarchical or recursive data structures and for querying data with relationships between rows within the same table.