**EX NO.: 04 SUB QUERIES AND JOINS**

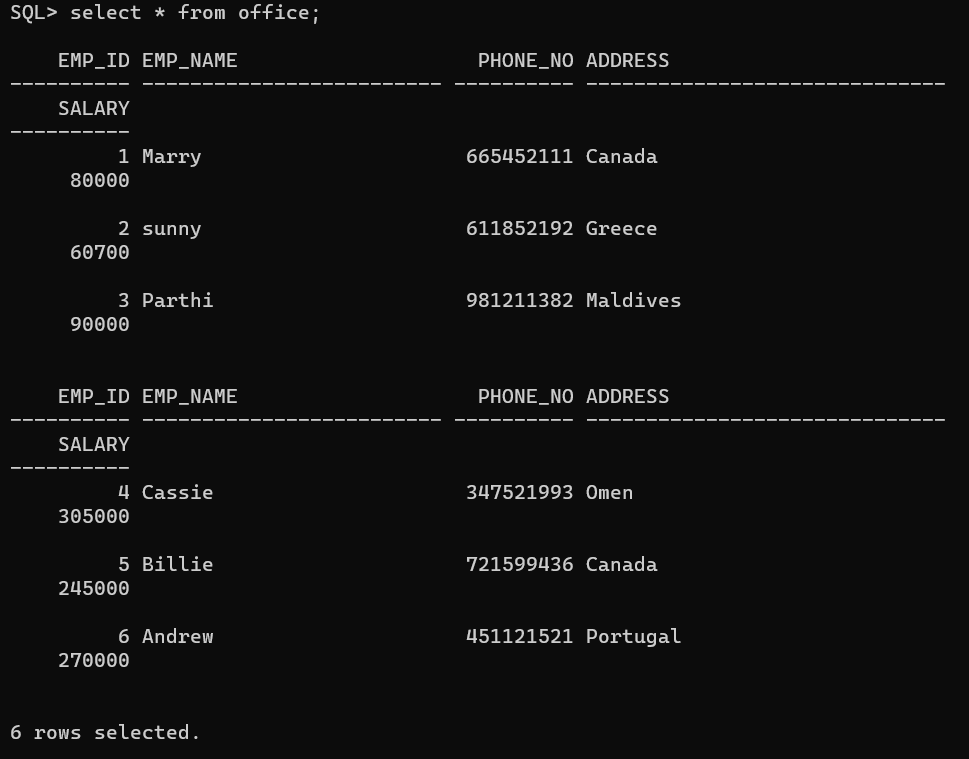
**AIM:**

To work with Sub queries and joins

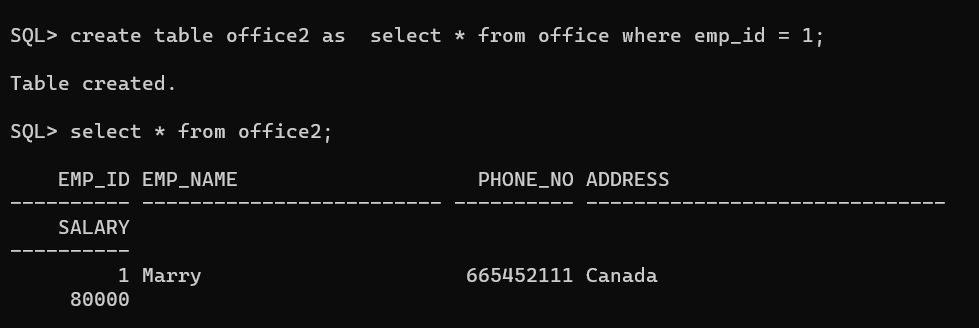
**SUB QUERIES:**

In SQL a Subquery can be simply defined as a query within another query. In other words we can say that a Subquery is a query that is embedded in WHERE clause of another SQL query.

**TABLE 1 – OFFICE:**



**TABLE 2 – OFFICE 2:**



**SUB QUERIES WITH SELECT STATEMENT:**

Subqueries are most frequently used with the SELECT statement.

**The basic syntax is as follows −**

SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

WHERE column\_name OPERATOR

(SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

[WHERE])

**OUTPUT:**



**SUB QUERIES WITH INSERT STATEMENT:**

Subqueries also can be used with INSERT statements. The INSERT statement uses the data returned from the subquery to insert into another table. The selected data in the subquery can be modified with any of the character, date or number functions.

**The basic syntax is as follows.**

INSERT INTO table\_name [ (column1 [, column2 ]) ]

SELECT [ \*|column1 [, column2 ]

FROM table1 [, table2 ]

[ WHERE VALUE OPERATOR ]

**OUTPUT:**



**SUB QUERIES WITH UPDATE STATEMENT:**

The subquery can be used in conjunction with the UPDATE statement. Either single or multiple columns in a table can be updated when using a subquery with the UPDATE statement.

**The basic syntax is as follows**

UPDATE table

SET column\_name = new\_value

[ WHERE OPERATOR [ VALUE ]

(SELECT COLUMN\_NAME

FROM TABLE\_NAME)

[ WHERE) ]

**OUTPUT:**



**SUB QUERIES WITH DELETE STATEMENT:**

The subquery can be used in conjunction with the DELETE statement like with any other statements mentioned above.

**The basic syntax is as follows.**

DELETE FROM TABLE\_NAME

[ WHERE OPERATOR [ VALUE ]

(SELECT COLUMN\_NAME

FROM TABLE\_NAME)

[ WHERE) ]

**OUTPUT:**



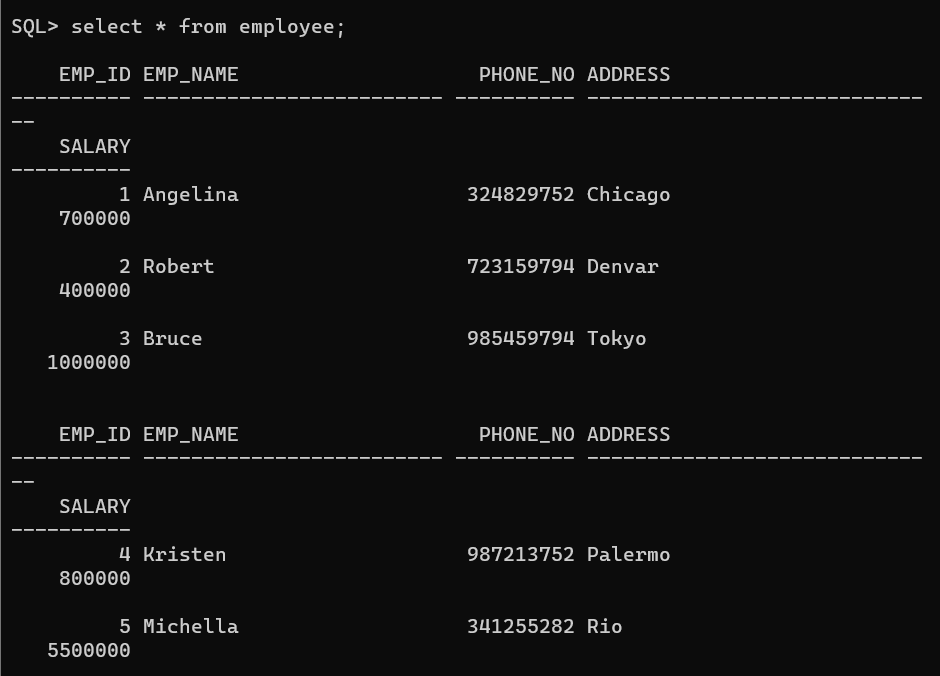
**JOINS:**

Different types of Joins are as follows:

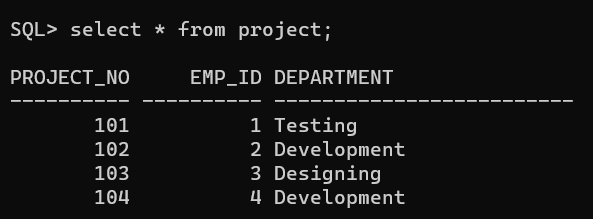
* INNER JOIN
* LEFT JOIN
* RIGHT JOIN
* FULL JOIN

Consider the two tables below:

**EMPLOYEE TABLE:**



**PROJECT TABLE:**



**A. INNER JOIN**

The INNER JOIN keyword selects all rows from both the tables as long as the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

**Syntax**:

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

INNER JOIN table2

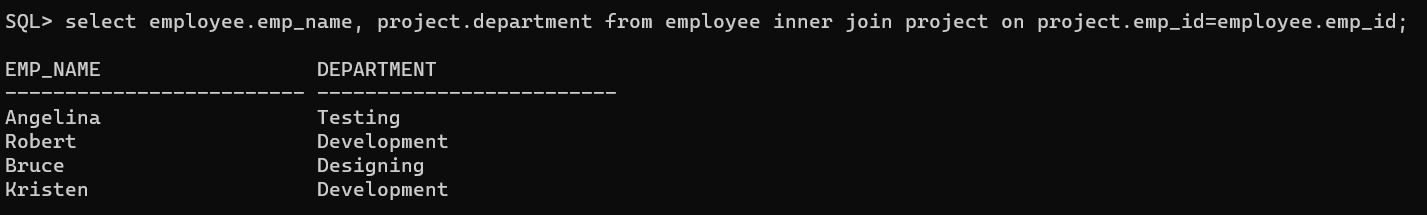
ON table1.matching\_column = table2.matching\_column;

**table1**: First table.

**table2**: Second table

**matching\_column**: Column common to both the tables.

**OUTPUT:**



**B. LEFT JOIN**

This join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain *null*. LEFT JOIN is also known as LEFT OUTER JOIN.

**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

LEFT JOIN table2

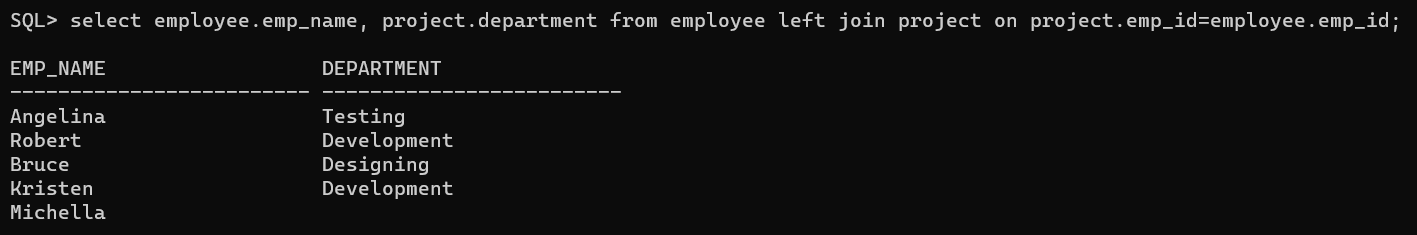
ON table1.matching\_column = table2.matching\_column;

**table1:** First table.

**table2:** Second table

**matching\_column:** Column common to both the tables.

**OUTPUT:**



**C. RIGHT JOIN**

RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain *null*. RIGHT JOIN is also known as RIGHT OUTER JOIN.

**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

RIGHT JOIN table2

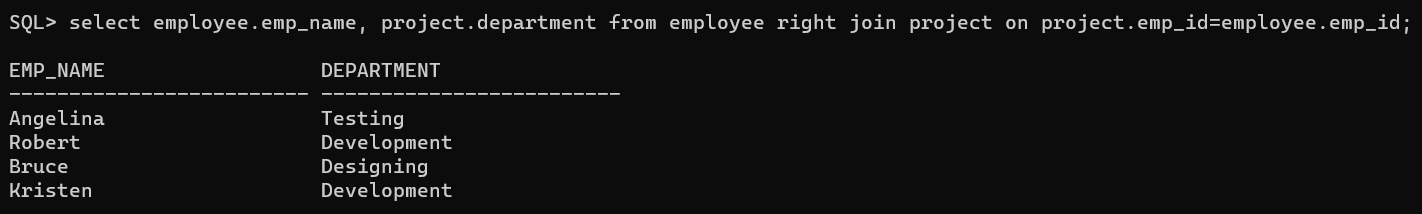
ON table1.matching\_column = table2.matching\_column;

**table1:** First table.

**table2:** Second table

**matching\_column:** Column common to both the tables.

**OUTPUT:**



**D. FULL JOIN**

FULL JOIN creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain *NULL* values.

**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

FULL JOIN table2

ON table1.matching\_column = table2.matching\_column;

**table1:** First table.

**table2:** Second table

**matching\_column:** Column common to both the tables.

**OUTPUT:**

