Lab 07 Manual

Database Systems.

Objective:

To explore concepts i.e., Joins and Sub-Queries.

Scope:

The student shall know the following:

- Workaround SQL Server.
- SQL Practice.
- Hands-on experience on the above-mentioned concepts.

Discussion:

1. **Joins**

   JOINS in SQL are commands which are used to combine rows from two or more tables, based on a related column between those tables. There are predominantly used when a user is trying to extract data from tables with joining condition.

   There are 4 different types of joins in SQL Server which are:

   - **Inner Join.**
   - **Outer Join (Left Outer Join, Right Outer Join and Full Outer Join).**
   - **Cross Join.**
   - **Self-Join.**

   Let's understand these Joins in details with examples.

   To understand joins, let's create 2 tables named "tblEmployee" and "tblDepartment" with some data as shown below.

```
CREATE TABLE tblEmployee (
       id INTEGER NOT NULL,
       firstName VARCHAR(255) NOT NULL,
       lastName VARCHAR(255) NOT NULL,
       salary INTEGER NOT NULL,
       departID INTEGER NULL
);

CREATE TABLE tblDepartment (
       id INTEGER NOT NULL,
       name VARCHAR(255) NOT NULL,
       location VARCHAR(255) NOT NULL
);
```

```
INSERT into tblEmployee VALUES (
       1, 'Sami', 'Ullah', 37000, 1
);

INSERT into tblEmployee VALUES (
       2, 'Hamid', 'Khan', 38000, 2
);

INSERT into tblEmployee VALUES (
       3, 'Aftab', 'Khan', 39000, 3
);

INSERT into tblEmployee VALUES (
       4, 'Abhishek', 'Raj', 35000, NULL
);
```

Try to insert at-least 25 records in tblEmployee

```
INSERT into tblDepartment VALUES (
       1, 'Structured Query Language', 'Lahore'
);

INSERT into tblDepartment VALUES (
       2, 'Finance and Management', 'Peshawar'
);

INSERT into tblDepartment VALUES (
       3, 'Marketing', 'Karachi'
);
```

Similarly, try to insert at-least 9 records in tblDepartment. One can take consider records from the table mentioned in the next page of this manual.

tblEmployee

	EmpId	FirstName	LastName	Salary	DepartmentId
1	1	Abhishek	Yadav	37000	1
2	2	Raj	Verma	47500	3
3	3	Tony	Bell	44000	2
4	4	Rahul	Sharma	27000	NULL
5	5	Vishal	Maurya	18500	4
6	6	Jyoti	Kakawat	28000	5
7	7	Vipin	Mehra	30000	2
8	8	Pradeep	Gupta	70000	1
9	9	Sumit	Jolly	55000	6
10	10	Jyoti	Mishra	20000	NULL
11	11	Karan	Pratap	25000	2
12	12	Aleem	Shaikh	35000	1
13	13	Kavita	Thakur	30000	4
14	14	Sufyan	Mukadam	40000	6
15	15	Ashish	Mehra	37000	NULL
16	16	Mehul	Joshi	65000	NULL
17	17	Payal	Yadav	15000	5
18	18	Swapnil	Patil	36000	5
19	19	Richa	Patel	35000	NULL
20	20	Mahesh	Singh	33000	5
21	21	Kaleem	Khan	8000	4
22	22	Heena	Shaikh	14000	NULL
23	23	Rahul	Kotian	26000	3
24	24	Naveen	Kapoor	37000	NULL
25	25	Sanjay	Sawant	65000	3

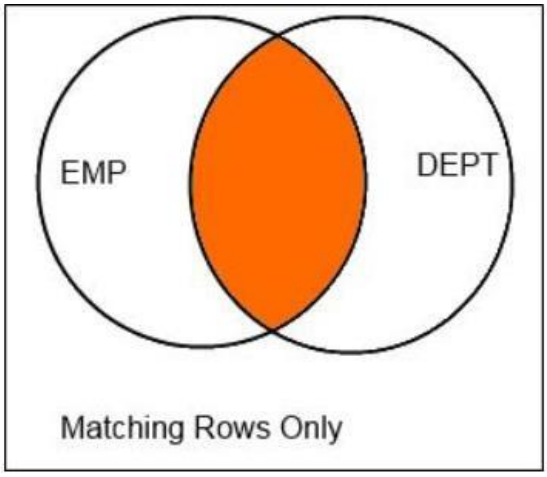
tblDepartment

	DeptId	DeptName	Location
1	1	SQL	Mumbai
2	2	Finance	Pune
3	3	SDM	Thane
4	4	MySQL	Chennai
5	5	DB2	Kolkata
6	6	Oracle	Delhi
7	7	Marketing	Delhi
8	8	Networking	Mumbai
9	9	HR	Mumbai

2. **Inner Join**

An Inner join requires matching records from both tables. You can use the "JOIN" or "INNER JOIN" keyword to join your tables. The "INNER" keyword is optional. In inner join, the record that is considered common among tables; is retrieved as you can see in diagram that two tables EMP and DEPT are joined while the common part is shaded which depicts the record that shall be retrieved.

SELECT <columns> from table1
JOIN table2
ON table1.KEY = table2.KEY;



Exercise:

```
SELECT
tblEmployee.id,
tblEmployee.firstName,
tblEmployee.lastName,
tblEmployee.salary,
tblDepartment.name,
tblDepartment.location
FROM tblEmployee
JOIN tblDepartment
ON tblEmployee.departID = tblDepartment.id;
```

Results | Messages

	id	firstName	lastName	salary	name	location
1	1	Sami	Ullah	37000	Structured Query Language	Lahore
2	2	Hamid	Khan	38000	Finance and Management	Peshawar
3	3	Aftab	Khan	39000	Marketing	Karachi

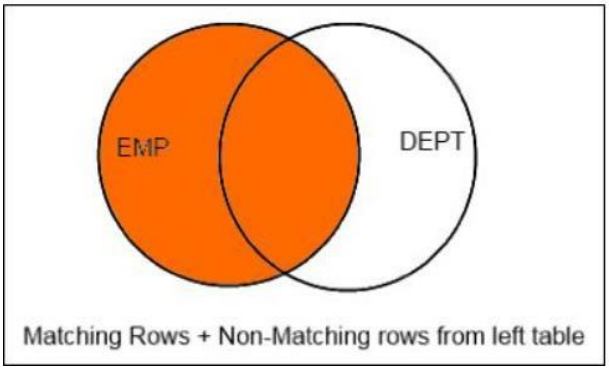
3. `Left Outer Join`

You can use "LEFT OUTER JOIN" keywords to do Left Joining. A left outer join returns a result containing both matching and non-matching rows from the left table. In other words, it returns all rows from the left table. If the join condition matches 0 records in 'Table B' then the join will still return a row in the Result Set, but with NULL in each column from 'Table B'.

SELECT <columns> from table1

LEFT JOIN table2

ON table1.KEY = table2.KEY;



Exercise

```
SELECT
tblEmployee.id,
tblEmployee.firstName,
tblEmployee.lastName,
tblEmployee.salary,
tblDepartment.name,
tblDepartment.location
FROM tblEmployee
LEFT JOIN tblDepartment
ON tblEmployee.departID = tblDepartment.id;
```

	id	firstName	lastNa...	salary	name	location
1	1	Sami	Ullah	37000	Structured Query Language	Lahore
2	2	Hamid	Khan	38000	Finance and Management	Peshawar
3	3	Aftab	Khan	39000	Marketing	Karachi
4	4	Abhishek	Raj	35000	NULL	NULL

4. `Right Outer Join`

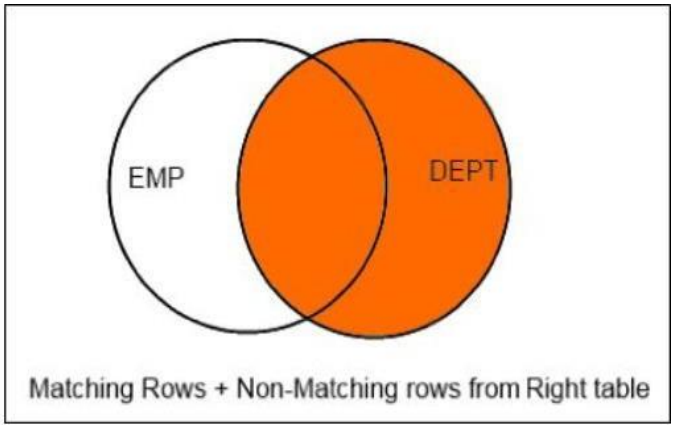
"RIGHT OUTER JOIN" keywords are used to do Right Joining.

A right outer join is nearly the same as a left outer join. A right outer join retrieves matching data from both of the tables + non-matching data from Table B.

SELECT <*columns*> from table1

OUTER JOIN table2

ON table1.KEY = table2.KEY;



Exercise:

```
SELECT
tblEmployee.id,
tblEmployee.firstName,
tblEmployee.lastName,
tblEmployee.salary,
tblDepartment.name,
tblDepartment.location
FROM tblEmployee
RIGHT JOIN tblDepartment
ON tblEmployee.departID = tblDepartment.id;
```

	id	firstNa...	lastNa...	salary	name	location
1	1	Sami	Ullah	37000	Structured Query Language	Lahore
2	2	Hamid	Khan	38000	Finance and Management	Peshawar
3	3	Aftab	Khan	39000	Marketing	Karachi
4	NULL	NULL	NULL	NULL	Social Media Production	Islamabad

5. **`Full Outer Join`**

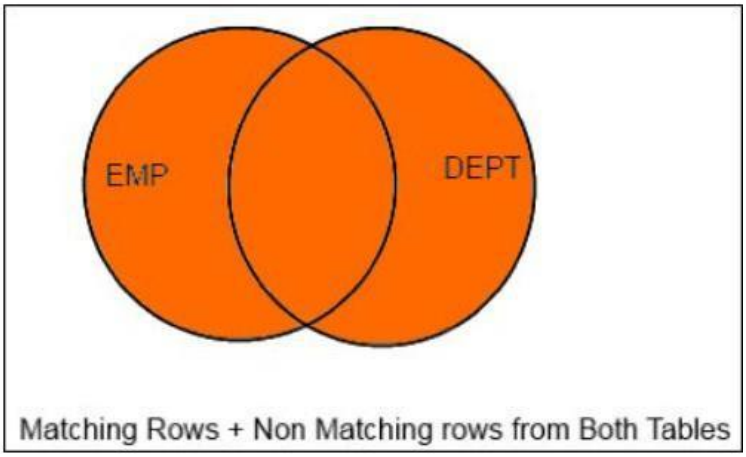
To do a Full outer join you can use the "FULL OUTER JOIN" keywords.

In a full outer join, the result set will display all the data from Table A and Table B. For every non-matching row in a full outer join the unmatched data has NULL.

SELECT <*columns*> from table1

FULL OUTER JOIN table2

ON table1.KEY = table2.KEY;



Exercise

```
SELECT
tblEmployee.id,
tblEmployee.firstName,
tblEmployee.lastName,
tblEmployee.salary,
tblDepartment.name,
tblDepartment.location
FROM tblEmployee
FULL JOIN tblDepartment
ON tblEmployee.departID = tblDepartment.id;
```

	id	firstName	lastNa...	salary	name	location
1	1	Sami	Ullah	37000	Structured Query Language	Lahore
2	2	Hamid	Khan	38000	Finance and Management	Peshawar
3	3	Aftab	Khan	39000	Marketing	Karachi
4	4	Abhishek	Raj	35000	NULL	NULL
5	NULL	NULL	NULL	NULL	Social Media Production	Islamabad

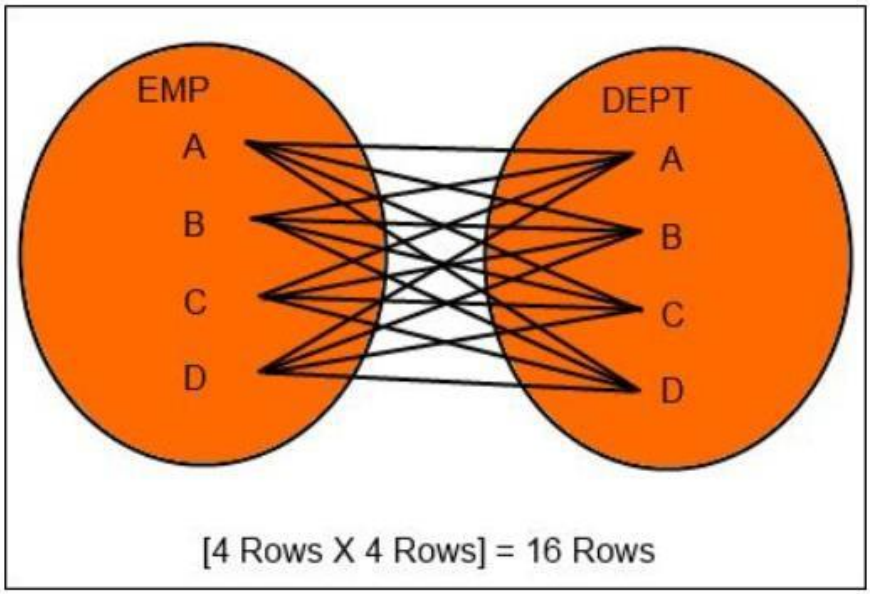
6. Cross Join

Use "CROSS JOIN" keywords to do cross joining.

A cross join returns the Cartesian product of rows from both tables in the join. In other words, it'll produce rows that combines rows from Table A with each row from Table B.

SELECT <*columns*> from table1

CROSS JOIN table2;



Exercise:

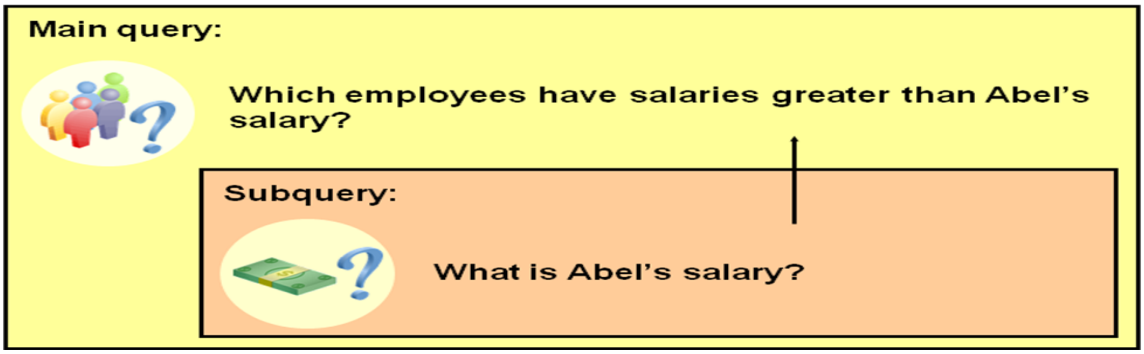
```
SELECT *from tblEmployee
CROSS JOIN tblDepartment;
```

	id	firstName	lastNa...	salary	depar...	id	name	location
1	1	Sami	Ullah	37000	1	1	Structured Query Language	Lahore
2	2	Hamid	Khan	38000	2	1	Structured Query Language	Lahore
3	3	Aftab	Khan	39000	3	1	Structured Query Language	Lahore
4	4	Abhishek	Raj	35000	NULL	1	Structured Query Language	Lahore
5	1	Sami	Ullah	37000	1	2	Finance and Management	Peshawar
6	2	Hamid	Khan	38000	2	2	Finance and Management	Peshawar
7	3	Aftab	Khan	39000	3	2	Finance and Management	Peshawar
8	4	Abhishek	Raj	35000	NULL	2	Finance and Management	Peshawar
9	1	Sami	Ullah	37000	1	3	Marketing	Karachi
10	2	Hamid	Khan	38000	2	3	Marketing	Karachi
11	3	Aftab	Khan	39000	3	3	Marketing	Karachi
12	4	Abhishek	Raj	35000	NULL	3	Marketing	Karachi
13	1	Sami	Ullah	37000	1	4	Social Media Production	Islamabad
14	2	Hamid	Khan	38000	2	4	Social Media Production	Islamabad
15	3	Aftab	Khan	39000	3	4	Social Media Production	Islamabad
16	4	Abhishek	Raj	35000	NULL	4	Social Media Production	Islamabad

7. Sub Queries:

Suppose you want to write a query to find out who earns a salary greater than Abel‘s salary. To solve this problem, you need two queries: one to find how much Abel earns, and a second query to find who earns more than that amount.

You can solve this problem by combining the two queries, placing one query inside the other query.



The inner query (or subquery) returns a value that is used by the outer query (or main query). Using a subquery is equivalent to performing two sequential queries and using the result of the first query as the search value in the second query.

Syntax (Sub-Query):

A subquery is a SELECT statement that is embedded in the clause of another SELECT statement. You can build powerful statements out of simple ones by using subqueries. They can be very useful when you need to select rows from a table with a condition that depends on the data in the table itself.

You can place the subquery in a number of SQL clauses, including the following:

- WHERE clause.
- HAVING clause.
- FROM clause.

SELECT <*columns*>

FROM table

WHERE expression *operator* (SELECT <*columns*> FROM table);

In the above syntax: *operator* includes a comparison condition such as >, =, or IN

Note: Comparison conditions fall into two classes: single-row operators (>, =, >=, <, <>, <=) and multiple-row operators (IN, ANY, ALL). The subquery is often referred to as a nested SELECT, sub-SELECT, or inner SELECT statement. The subquery generally executes first, and its output is used to complete the query condition for the main (or outer) query.

Exercise:

Using subquery, select record from *tblEmployee* whose salary is greater than the record whose *lastName* is **Raj.**

Please practice it.

```
104
105  SELECT *from dbo.tblEmployee
106  WHERE salary > (
107  SELECT salary from dbo.tblEmployee WHERE lastName = 'Raj'
108  );
```

100 %

Results | Messages

	id	firstNa...	lastNa...	salary	depart...
1	1	Sami	Ullah	37000	1
2	2	Hamid	Khan	38000	2
3	3	Aftab	Khan	39000	3

In the example, the inner query determines the salary of employee **Raj** that is 35000. The outer query takes the result of the inner query and uses this result to display all the employees who earn more than employee **Raj**.

Guidelines for Using Subqueries:

- A subquery must be enclosed in parentheses.
- Place the subquery on the right side of the comparison condition for readability however, the subquery can appear on either side of the comparison operator.
- Two classes of comparison conditions are used in subqueries: single-row operators and multiple-row operators.

Types of Subqueries:

- **Single-row subqueries:**

  Queries that return only one row from the inner SELECT statement.
- **Multiple-row subqueries:**

  Queries that return more than one row from the inner SELECT statement

Single-Row Subqueries

A single-row subquery is one that returns one row from the inner SELECT statement. This type of subquery uses a single-row operator.

Executing Single-Row Subqueries

A SELECT statement can be considered as a query block.

Exercise:

Using subqueries, retrieve records of the employees whose salary is greater than the employee whose last name is “Raj” and have same department.

```
SELECT *from tblEmployee
WHERE salary > (
	select salary from tblEmployee WHERE lastName = 'Raj'
)
AND departID = (
	select departID from tblEmployee WHERE lastName = 'Raj'
);
```

The above example retrieves all the records of the employees who are doing the same job as **Raj** is doing but have maximum salary than **Raj**.

Using Group functions in a Subquery

You can display data from a main query by using a group function in a subquery to return a single row. The subquery is in parentheses and is placed after the comparison condition.

Exercise:

Construct a query that extract the employee who has least salary in the whole data.

```
SELECT *from dbo.EMP
WHERE SAL = (SELECT MIN(SAL) from dbo.EMP);
```

The HAVING clause with sub-queries

You can use subqueries not only in the WHERE clause, but also in the HAVING clause.

Exercise:

The SQL statement in the following example displays all the departments that have a minimum salary less than that of department 10.

```
SELECT   DEPTNO, MIN(SAL) FROM dbo.EMP
GROUP BY DEPTNO
HAVING   MIN(SAL) <
(
	SELECT MIN(SAL) FROM dbo.EMP
	WHERE DEPTNO = 10
);
```

Moreover, you can use IN, ANY, ALL and NOT IN key words in sub-query condition.

Tasks:

1. Using joins, retrieve records from two tables i.e., employee and department. The resultant table shall be like the table mentioned below

	EMP_NUMBER	FULL_NAME	Salary	DEPT_NAME	DEPT_ADDRESS
1	1	Sami Ullah	37000	Structured Query Language	Lahore
2	2	Hamid Khan	38000	Finance and Management	Peshawar
3	3	Aftab Khan	39000	Marketing	Karachi
4	4	Abhishek Raj	35000	Not Found	Not Found

2. Construct a query that can result same as in the below diagram. The query shall be able to count total number of employees in single department.

Results Messages

	TOTAL_EMPS_IN_EACH_DEPARTMENT	DNAME
1	1	ACCOUNTING
2	4	RESEARCH

3. Using sub-queries and joins, extract department information about who has the least salary of employees.
4. Construct a query that can retrieve the department information that has lowest average salary? **Hint: use joins and sub query.**