

Ex no: 6	Querying Using ANY, ALL, IN, EXISTS, NOT EXISTS, UNION, INTERSECT
Date:	

Aim:

To write SQL queries using ANY, ALL, IN, EXISTS, NOT EXISTS, UNION, and INTERSECT operators.

Procedure:

1. Use the ANY operator to compare a value to any value in a list.
2. Use the ALL operator to compare a value to all values in a list.
3. Use the IN operator to check if a value exists in a list.
4. Use the EXISTS operator to check the existence of a subquery result.
5. Use the NOT EXISTS operator to check the non-existence of a subquery result.
6. Use the UNION operator to combine the results of two queries.
7. Use the INTERSECT operator to return the common results of two queries.
8. Document the SQL queries used for each operator.
9. Execute the queries in the SQL environment.
10. Validate the results by checking the retrieved data.

Queries:

1. Use the ANY operator to compare a value to any value in a list.

Query:

```
CREATE TABLE EMPLOYEE (
  empId INTEGER PRIMARY KEY, Fname TEXT NOT NULL, Lname TEXT NOT NULL,
  dept TEXT NOT NULL, salary int, department_id int, manager_id int );
```

-- insert

```
INSERT INTO EMPLOYEE VALUES (1, 'Clark','S', 'Sales',1233455,10,201);
INSERT INTO EMPLOYEE VALUES (2, 'Dave', 'N','Accounting',20000,20,202);
INSERT INTO EMPLOYEE VALUES (3, 'Ava','V', 'Sales',30000,30,203);
```

-- fetch

```
SELECT empId , salary
FROM EMPLOYEE
WHERE salary > ALL (SELECT salary FROM EMPLOYEE WHERE department_id = 20);
```

Output:

```
+-----+-----+
| empId | salary |
+-----+-----+
|      1 | 1233455 |
|      3 | 30000  |
+-----+-----+
```

2. Use the ALL operator to compare a value to all values in a list.**Query:**

```
SELECT empId , salary
FROM EMPLOYEE
WHERE salary > ALL (SELECT salary FROM EMPLOYEE WHERE department_id = 20);
```

Output:

```
+-----+-----+
| empId | salary |
+-----+-----+
|      1 | 1233455 |
|      3 | 300000 |
+-----+-----+
```

3. Use the IN operator to check if a value exists in a list.**Query:**

```
SELECT * FROM EMPLOYEE;
SELECT empId, department_id
FROM EMPLOYEE
WHERE department_id IN (10, 20, 30);
```

Output:

```
+-----+-----+
| empId | department_id |
+-----+-----+
|      1 | 10 |
|      2 | 20 |
|      3 | 30 |
+-----+-----+
```

4. Use the EXISTS operator to check the existence of a subquery result.**Query:**

```
SELECT empId, Fname, Lname
FROM EMPLOYEE
WHERE EXISTS (SELECT 1 FROM EMPLOYEE e WHERE e.manager_id = EMPLOYEE.empId);
```

5. Use the NOT EXISTS operator to check the non-existence of a subquery result.**Query:**

```
SELECT empId, Fname, Lname
FROM EMPLOYEE e1
WHERE NOT EXISTS (
SELECT 1 FROM EMPLOYEE e2
```

```
WHERE e2.manager_id = e1.empId
);
```

Output:

```
+-----+-----+-----+
| empId | Fname | Lname |
+-----+-----+-----+
|      1 | Clark | S      |
|      2 | Dave  | N      |
|      3 | Ava   | V      |
+-----+-----+-----+
```

6. Use the UNION operator to combine the results of two queries.**Query:**

```
SELECT empId, Fname, Lname, dept
FROM EMPLOYEE
WHERE dept = 'Sales'
UNION ALL
SELECT empId, Fname, Lname, dept
FROM EMPLOYEE
WHERE dept = 'Accounting';
```

Output:

```
+-----+-----+-----+-----+
| empId | Fname | Lname | dept      |
+-----+-----+-----+-----+
|      1 | Clark | S      | Sales     |
|      3 | Ava   | V      | Sales     |
|      2 | Dave  | N      | Accounting|
+-----+-----+-----+-----+
```

7. Use the INTERSECT operator to return the common results of two queries.**Query:**

```
SELECT empId, Fname, Lname, dept
FROM EMPLOYEE
WHERE dept = 'Sales'
INTERSECT
SELECT empId, Fname, Lname, dept
FROM EMPLOYEE
WHERE dept = 'Accounting';
```

Algorithm	15	
Program	30	
Execution	30	
Output & Result	15	
Viva / Record	10	
Total	100	

Result:

Thus, we created SQL queries for quering using ANY, ALL, IN, EXISTS, NOT EXISTS, UNION, INTERSECT successfully and the output is verified.