



School of Computer Science, UPES, Dehradun.

A

LABORATORY FILE

On

# DATABASE MANAGEMENT SYSTEM (DBMS) LAB

B.TECH. -III Semester

**Submitted by:**

Name: Mayank Negi

SAP ID: 500120279

Roll No: R2142230039

Batch: 2

## **Experiment 06**

### **To understand and use SQL Sub-Query**

#### **AIM:**

To understand the use of SQL subquery.

#### **Problem Statement:**

1. Create the tables
2. Populate the tables with sample data
3. Perform some SQL Queries

#### **THEORY:**

Structured query language (SQL) is a programming language for storing and processing information in a relational database. A relational database stores information in tabular form, with rows and columns representing different data attributes and the various relationships between the data values.

#### **COMMAND USED:**

1. DROP TABLE: Deletes a table and all its data.
2. PRIMARY KEY: Defines a column or set of columns as the unique identifier for rows in a table.
3. FOREIGN KEY: Establishes a relationship between columns in different tables.
4. ORDER BY: Sorts the result set by one or more columns.
5. WHERE: Filters records based on specific conditions.
6. GROUP BY: Groups rows that have the same values into summary rows.
7. HAVING: Filters records after the GROUP BY clause.
8. JOIN: Combines rows from two or more tables based on a related column.
9. CREATE INDEX: Creates an index on a table to speed up searches.

**RESULTS:**

```
1      -- Ayush Vashishth
2      -- 500119331
3
4  •   CREATE DATABASE Supplier;
5  •   USE Supplier;
6      -- Supplier Table
7  •   CREATE TABLE Supplier (
8      scode INT PRIMARY KEY,
9      sname VARCHAR(50),
10     scity VARCHAR(50),
11     turnover DECIMAL(10, 2)
12 );
13     -- Part Table
14 •   CREATE TABLE Part (
15     pcode INT PRIMARY KEY,
16     weigh DECIMAL(10, 2),
17     color VARCHAR(20),
18     cost DECIMAL(10, 2),
19     sellingprice DECIMAL(10, 2)
20 );
21     -- Supplier_Part Table (Many-to-Many relationship)
22 •   CREATE TABLE Supplier_Part (
23     scode INT,
24     pcode INT,
25     qty INT,
26     PRIMARY KEY (scode, pcode),
```

```
27 FOREIGN KEY (scode) REFERENCES Supplier(scode),
28 FOREIGN KEY (pcode) REFERENCES Part(pcode)
29 );
30 -- Insert data into Supplier
31 • INSERT INTO Supplier VALUES
32 (1, 'Supplier1', 'Bombay', 50.00),
33 (2, 'Supplier2', 'Delhi', 75.00),
34 (3, 'Supplier3', 'Bombay', NULL);
35 -- Insert data into Part
36 • INSERT INTO Part VALUES
37 (1, 30, 'Red', 20.00, 25.00),
38 (2, 40, 'Blue', 30.00, 35.00),
39 (3, 35, 'Green', 40.00, 50.00);
40 -- Insert data into Supplier_Part
41 • INSERT INTO Supplier_Part VALUES
42 (1, 1, 100),
43 (1, 2, 200),
44 (2, 3, 150),
45 (3, 2, 120);
46
47 • SELECT scode, pcode
48 FROM Supplier_Part
49 ORDER BY scode ASC;
50
51 • SELECT *
52 FROM Supplier
```

---

```
53 WHERE scity = 'Bombay' AND turnover = 50;
54
55 • SELECT COUNT(*) AS total_suppliers
56 FROM Supplier;
57
58 • SELECT pcode
59 FROM Part
60 WHERE weigh BETWEEN 25 AND 35;
61
62 • SELECT scode
63 FROM Supplier
64 WHERE turnover IS NULL;
65
66 • SELECT pcode
67 FROM Part
68 WHERE cost IN (20, 30, 40);
69
70 • SELECT SUM(qty) AS total_quantity
71 FROM Supplier_Part
72 WHERE pcode = 2;
73
74 • SELECT sname
75 FROM Supplier
76 WHERE scode IN (SELECT scode
77 FROM Supplier_Part
78 WHERE pcode = 2);
```

---

```
80 • SELECT pcode
81 FROM Part
82 WHERE cost > (SELECT AVG(cost) FROM Part);
83
84 • SELECT scode, turnover
85 FROM Supplier
86 ORDER BY turnover DESC;
```

```
1 • SELECT * FROM supplier.part;
```


Result Grid

Filter Rows:


Edit:

	pcode	weigh	color	cost	sellingprice
▶	1	30.00	Red	20.00	25.00
	2	40.00	Blue	30.00	35.00
	3	35.00	Green	40.00	50.00
●	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM supplier.supplier;
```

Result Grid				
Filter Rows: <input type="text"/>				
Edit: 				
	scode	sname	scity	turnover
▶	1	Supplier 1	Bombay	50.00
	2	Supplier2	Delhi	75.00
	3	Supplier3	Bombay	NULL
✱	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM supplier.supplier_part;
```

Result Grid			
Filter Rows: <input type="text"/>			
Edit: 			
	scode	pcode	qty
▶	1	1	100
	1	2	200
	2	3	150
	3	2	120
✱	NULL	NULL	NULL

**Conclusion:**

In this experiment, the use of SQL subqueries was explored to retrieve specific data from relational databases. Through the creation and population of tables, subqueries were used to filter and organize data based on various conditions, such as joins, groupings, and orderings. Subqueries enhance query flexibility by allowing complex data extraction within a single statement. This experiment provided hands-on experience with key SQL commands like DROP, PRIMARY KEY, FOREIGN KEY, ORDER BY, WHERE, GROUP BY, and JOIN, deepening the understanding of relational database management and data querying techniques.