School of Computer Science, UPES, Dehradun.

A

LABORATORY FILE

On

DATABASE MANAGEMENT SYSTEM (DBMS) LAB

B.TECH. -III Semester

AUG. – NOV.- 2024.

Submitted by:
Name: Mehul Sangwan
SAP ID: 500120255
Roll No: R2142230033
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

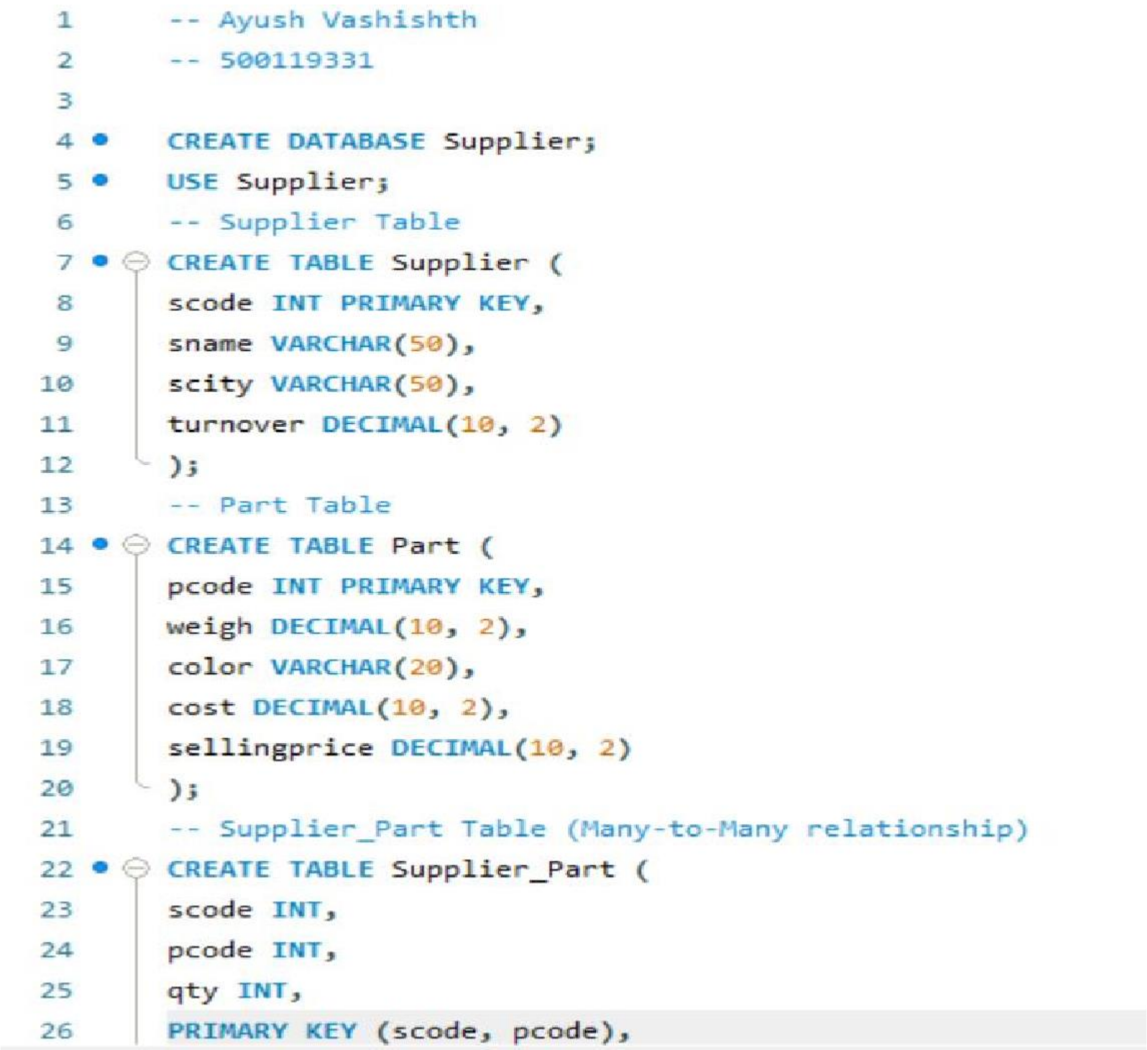
THEORY:

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.

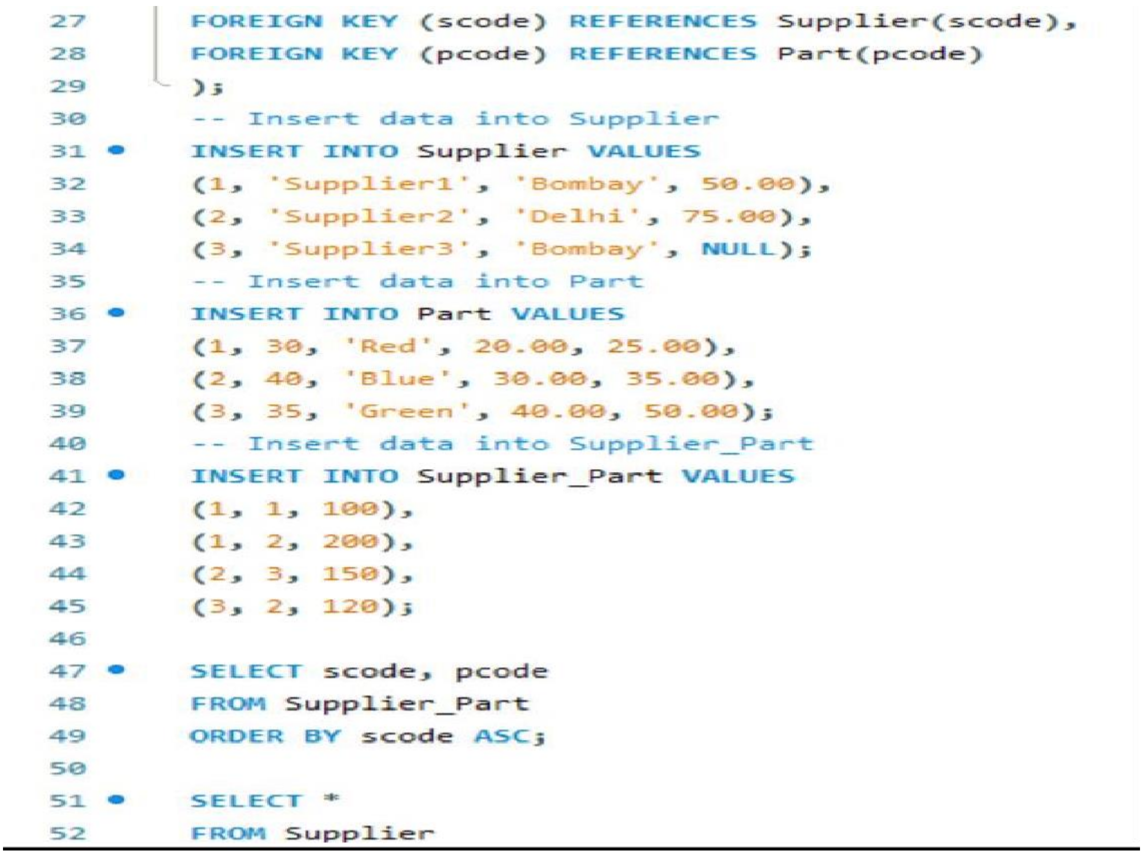
COMMAND USED:

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.

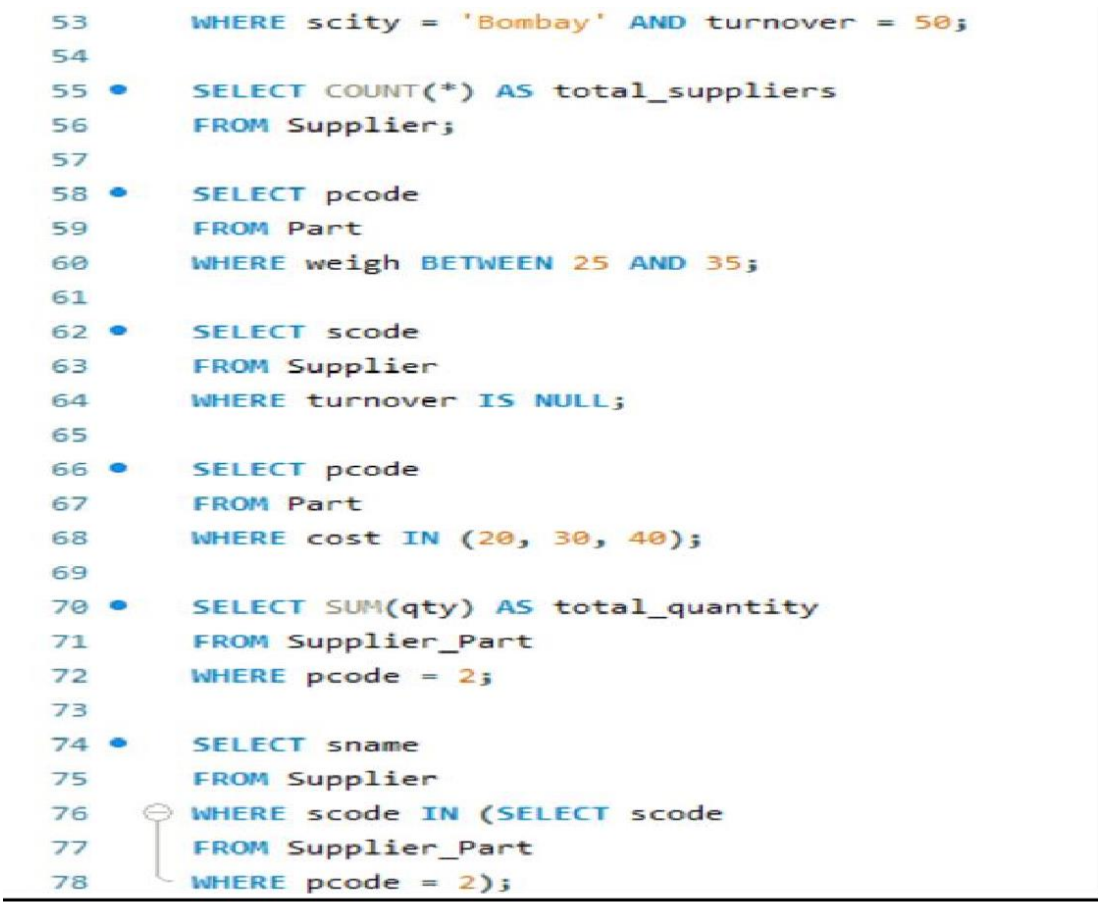
RESULTS:



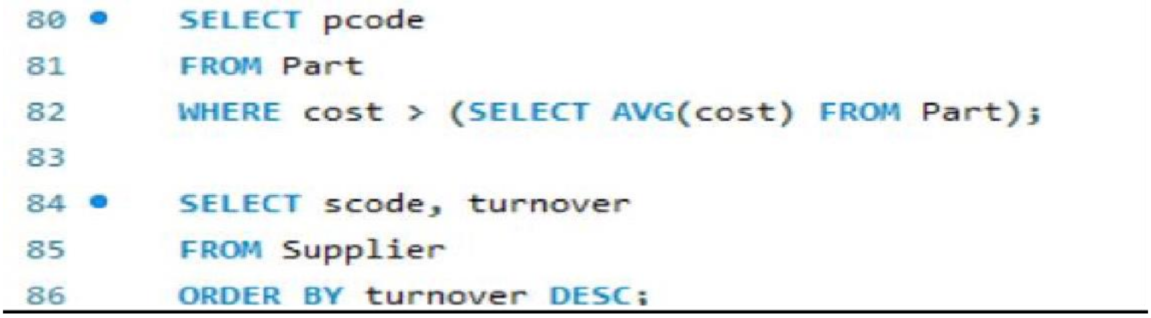
```
-- Ayush Vashishth
-- 500119331

CREATE DATABASE Supplier;
USE Supplier;
-- Supplier Table
CREATE TABLE Supplier (
scode INT PRIMARY KEY,
sname VARCHAR(50),
scity VARCHAR(50),
turnover DECIMAL(10, 2)
);
-- Part Table
CREATE TABLE Part (
pcode INT PRIMARY KEY,
weigh DECIMAL(10, 2),
color VARCHAR(20),
cost DECIMAL(10, 2),
sellingprice DECIMAL(10, 2)
);
-- Supplier_Part Table (Many-to-Many relationship)
CREATE TABLE Supplier_Part (
scode INT,
pcode INT,
qty INT,
PRIMARY KEY (scode, pcode),
```



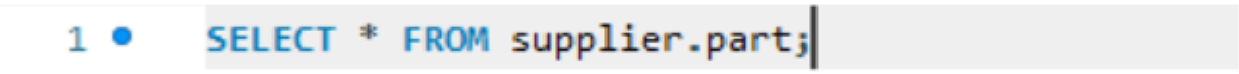
```sql
FOREIGN KEY (scode) REFERENCES Supplier(scode),
FOREIGN KEY (pcode) REFERENCES Part(pcode)
);
-- Insert data into Supplier
INSERT INTO Supplier VALUES
(1, 'Supplier1', 'Bombay', 50.00),
(2, 'Supplier2', 'Delhi', 75.00),
(3, 'Supplier3', 'Bombay', NULL);
-- Insert data into Part
INSERT INTO Part VALUES
(1, 30, 'Red', 20.00, 25.00),
(2, 40, 'Blue', 30.00, 35.00),
(3, 35, 'Green', 40.00, 50.00);
-- Insert data into Supplier_Part
INSERT INTO Supplier_Part VALUES
(1, 1, 100),
(1, 2, 200),
(2, 3, 150),
(3, 2, 120);

SELECT scode, pcode
FROM Supplier_Part
ORDER BY scode ASC;

SELECT *
FROM Supplier
```



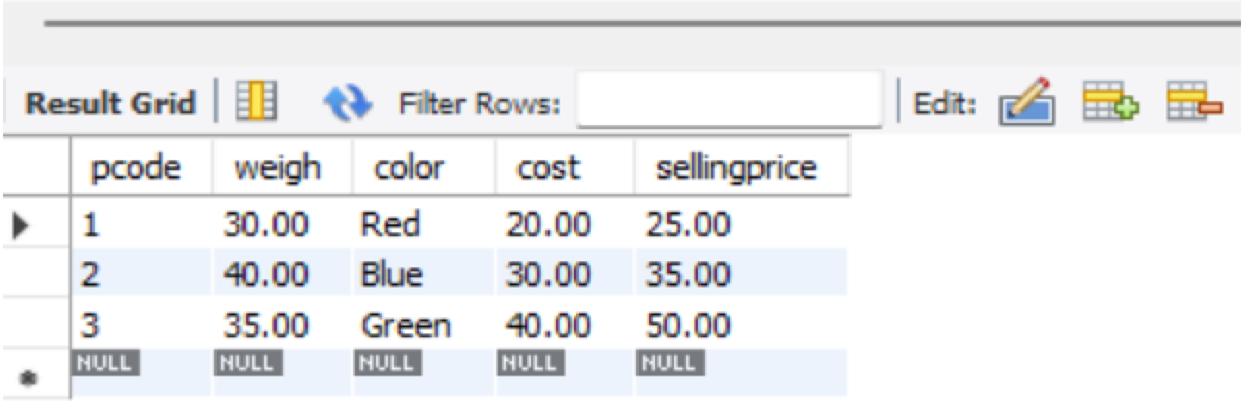
```sql
WHERE scity = 'Bombay' AND turnover = 50;

SELECT COUNT(*) AS total_suppliers
FROM Supplier;

SELECT pcode
FROM Part
WHERE weigh BETWEEN 25 AND 35;

SELECT scode
FROM Supplier
WHERE turnover IS NULL;

SELECT pcode
FROM Part
WHERE cost IN (20, 30, 40);

SELECT SUM(qty) AS total_quantity
FROM Supplier_Part
WHERE pcode = 2;

SELECT sname
FROM Supplier
WHERE scode IN (SELECT scode
FROM Supplier_Part
WHERE pcode = 2);
```



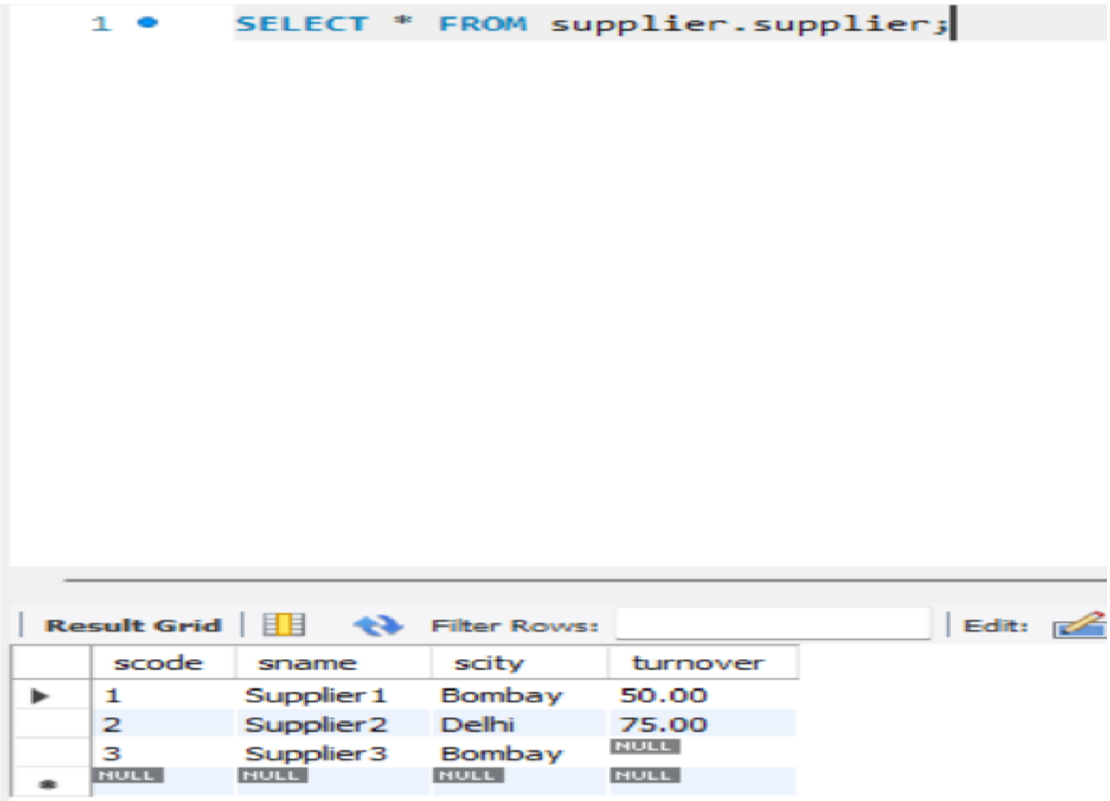
```
SELECT pcode
FROM Part
WHERE cost > (SELECT AVG(cost) FROM Part);

SELECT scode, turnover
FROM Supplier
ORDER BY turnover DESC;
```



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

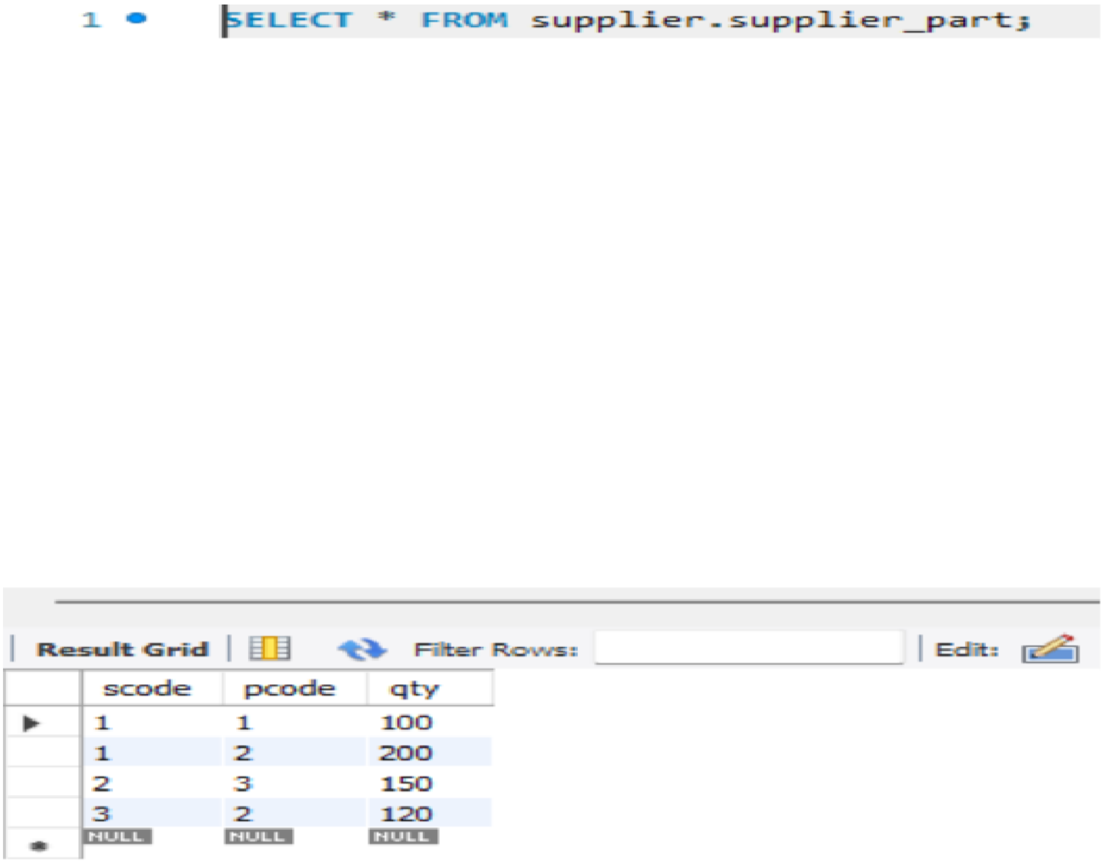


pcode	weigh	color	cost	sellingprice
1	30.00	Red	20.00	25.00
2	40.00	Blue	30.00	35.00
3	35.00	Green	40.00	50.00
NULL	NULL	NULL	NULL	NULL



```
SELECT * FROM supplier.supplier;
```

	scode	sname	scity	turnover
▶	1	Supplier1	Bombay	50.00
	2	Supplier2	Delhi	75.00
	3	Supplier3	Bombay	NULL
●	NULL	NULL	NULL	NULL



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

	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.