**Aim** : To understand the use of sql subquery.

**Objective** :

This experiment aims to delve into SQL subqueries, exploring their functionality in retrieving data from interconnected tables, and executing queries based on specific conditions. It enhances proficiency in database querying techniques and comprehension of relational database concepts.

**Theory** :

SQL subqueries, also known as nested queries, facilitate the execution of queries within other queries, enabling complex data retrieval.

They are utilized extensively in relational databases for their flexibility in filtering, sorting, and aggregating data.

Subqueries can be integrated into various SQL statements, including SELECT, INSERT, UPDATE, or DELETE.

They assist in retrieving data from multiple interconnected tables using JOIN operations and applying conditions based on inner query results.

Subqueries can be categorized as correlated or non-correlated, with correlated subqueries referencing outer queries.

Proficiency in SQL subqueries optimizes query performance, enabling advanced data analysis and manipulation tasks in relational database systems.

**CODE**

**-- Create tables**

CREATE TABLE Supplier (

scode INT PRIMARY KEY,

sname VARCHAR(50),

scity VARCHAR(50),

turnov--er DECIMAL(15, 2)

);

CREATE TABLE Part (

pcode INT PRIMARY KEY,

weigh DECIMAL(10, 2),

color VARCHAR(50),

cost DECIMAL(10, 2),

sellingprice DECIMAL(10, 2)

);

CREATE TABLE Supplier\_Part (

scode INT,

pcode INT,

qty INT,

FOREIGN KEY (scode) REFERENCES Supplier(scode),

FOREIGN KEY (pcode) REFERENCES Part(pcode));

**-- Populate tables (Sample data)**

INSERT INTO Supplier VALUES (1, 'M Wasi', 'Bombay', 200000000.00);

INSERT INTO Supplier VALUES (2, 'Rajneesh Prajapati', 'Delhi', 600000000.00);

INSERT INTO Supplier VALUES (3, 'Moulik Verma', 'Dehradun', 700000000.00);

INSERT INTO Supplier VALUES (4, 'Ayush Maurya', 'Dehradun', NULL);

INSERT INTO Supplier VALUES (5, 'Eroze Barua', 'Bombay', NULL);

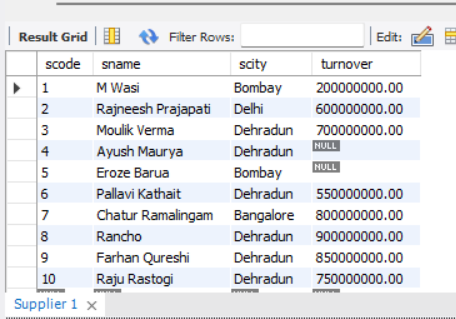
INSERT INTO Supplier VALUES (6, 'Pallavi Kathait', 'Dehradun', 550000000.00);

INSERT INTO Supplier VALUES (7, 'Chatur Ramalingam', 'Bangalore', 800000000.00);

INSERT INTO Supplier VALUES (8, 'Rancho', 'Dehradun', 900000000.00);

INSERT INTO Supplier VALUES (9, 'Farhan Qureshi', 'Dehradun', 850000000.00);

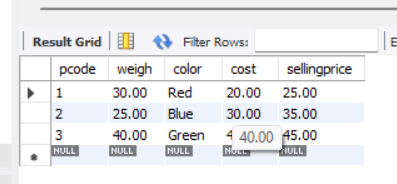
INSERT INTO Supplier VALUES (10, 'Raju Rastogi', 'Dehradun', 750000000.00);



INSERT INTO Part VALUES (1, 30.00, 'Red', 20.00, 25.00);

INSERT INTO Part VALUES (2, 25.00, 'Blue', 30.00, 35.00);

INSERT INTO Part VALUES (3, 40.00, 'Green', 40.00, 45.00);

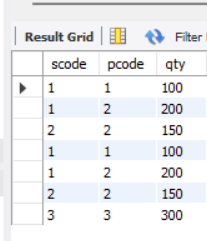


INSERT INTO Supplier\_Part VALUES (1, 1, 100);

INSERT INTO Supplier\_Part VALUES (1, 2, 200);

INSERT INTO Supplier\_Part VALUES (2, 2, 150);

INSERT INTO Supplier\_Part VALUES (3, 3, 300);



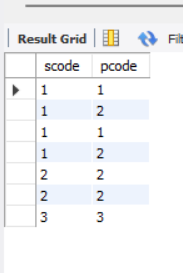
**-- Queries**

**-- 1. Get the supplier number and part number in ascending order of supplier number.**

SELECT sp.scode, sp.pcode

FROM Supplier\_Part sp

ORDER BY sp.scode;

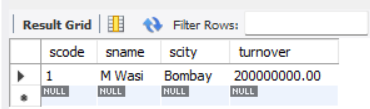


**-- 2. Get the details of supplier who operate from Bombay with turnover 50.**

SELECT \*

FROM Supplier

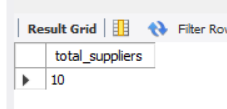
WHERE scity = 'Bombay' AND turnover = 200000000.00;



**-- 3. Get the total number of supplier.**

SELECT COUNT(\*) AS total\_suppliers

FROM Supplier;

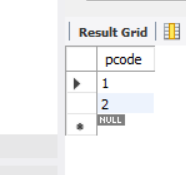


**-- 4. Get the part number weighing between 25 and 35.**

SELECT pcode

FROM Part

WHERE weigh BETWEEN 25 AND 35;

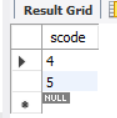


**-- 5. Get the supplier number whose turnover is null.**

SELECT scode

FROM Supplier

WHERE turnover IS NULL;

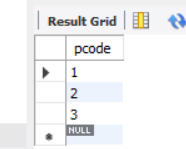


**-- 6. Get the part number that cost 20, 30 or 40 rupees.**

SELECT pcode

FROM Part

WHERE cost IN (20.00, 30.00, 40.00);

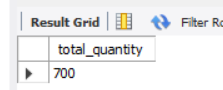


**-- 7. Get the total quantity of part 2 that is supplied.**

SELECT SUM(qty) AS total\_quantity

FROM Supplier\_Part

WHERE pcode = 2;



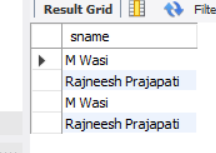
**-- 8. Get the name of supplier who supply part 2.**

SELECT s.sname

FROM Supplier s

JOIN Supplier\_Part sp ON s.scode = sp.scode

WHERE sp.pcode = 2;

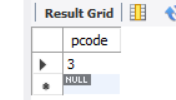


**-- 9. Get the part number whose cost is greater than the average cost.**

SELECT pcode

FROM Part

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



**-- 10. Get the supplier number and turnover in descending order of turnover.**

SELECT scode, turnover

FROM Supplier

ORDER BY turnover DESC;

