DBMS: LAB 7
Advanced SQL
University Fest Management System

Name: Rithvik Matta
SRN: PES2UG23CS485
Sec: CSE H

Task 1: Nested Queries

Q1. List the names of participants who registered for events with a price above the average event price.

```
SELECT DISTINCT p.Name AS Participant_Name
FROM Participants p
JOIN Registration r ON p.SRN = r.SRN JOIN
Event e ON r.Event_ID = e.Event_ID WHERE
e.Price > (
   SELECT AVG(Price)
   FROM Event
);
```

```
+------------------+
| Participant_Name |
+------------------+
| Varun            |
| Neha             |
| Charlie          |
| Priya            |
| Meera            |
| Manoj            |
| Kabir            |
+------------------+
7 rows in set (0.00 sec)
```

Q2. Show events that have more registrations than the average number of registrations per event.

```
SELECT e.Event_name, COUNT(r.SRN) AS Total_Registrations FROM
Event e
JOIN Registration r ON e.Event_ID = r.Event_ID
GROUP BY e.Event_ID, e.Event_name HAVING
COUNT(r.SRN) > (
   SELECT AVG(reg_count) FROM (
      SELECT COUNT(SRN) AS reg_count
      FROM Registration
      GROUP BY Event_ID
   ) AS avg_reg
);
```

```
+----------------+---------------------+
| Event_name     | Total_Registrations |
+----------------+---------------------+
| Dance Battle   |                   2 |
| Drama Play     |                   2 |
| Music Night    |                   2 |
| Startup Pitch  |                   2 |
| Art Exhibition |                   2 |
+----------------+---------------------+
5 rows in set (0.00 sec)
```

Task 2: Correlated Queries

Q3. Find participants who have registered for the highest-priced event(s)

```
SELECT DISTINCT p.Name AS Participant_Name, e.Event_name, e.Price FROM
Participants p
JOIN Registration r ON p.SRN = r.SRN JOIN
Event e ON r.Event_ID = e.Event_ID WHERE
e.Price = (
   SELECT MAX(Price) FROM Event
);
```

```
+------------------+------------+---------+
| Participant_Name | Event_name | Price   |
+------------------+------------+---------+
| Varun            | FutureTech | 1000.00 |
+------------------+------------+---------+
1 row in set (0.00 sec)
```

Q4. Find stalls that offer items cheaper than the average price of that item across all stalls.

```
SELECT si.Stall_ID, si.Item_name, si.Price_per_unit
FROM Stall_items si
WHERE si.Price_per_unit < ( SELECT
   AVG(s2.Price_per_unit) FROM
   Stall_items s2
   WHERE s2.Item_name = si.Item_name
);
```

```
+----------+------------------+----------------+
| Stall_ID | Item_name        | Price_per_unit |
+----------+------------------+----------------+
| S9       | Mushroom Risotto |          40.00 |
+----------+------------------+----------------+
1 row in set (0.00 sec)
```

Task 3: Window Functions

Q5. Show the SRN and name of participants and how many times they purchased something, along with their rank based on total quantity bought

```
SELECT
    p.SRN,
    p.Name,
    COUNT(DISTINCT pur.Timestamp) AS Purchases_Made,
    SUM(pur.Quantity) AS Total_Quantity,
    RANK() OVER (ORDER BY SUM(pur.Quantity) DESC) AS Purchase_Rank FROM
Participants p
JOIN Purchased pur ON p.SRN = pur.SRN
GROUP BY p.SRN, p.Name;
```

SRN	Name	Purchases_Made	Total_Quantity	Purchase_Rank
P1017	Varun	5	13	1
P1005	Suresh	1	3	2
P1012	Nisha	1	3	2
P1001	Charlie	1	2	4
P1003	Vikram	1	2	4
P1007	Manoj	1	2	4
P1009	Arjun	1	2	4
P1011	Rahul	1	2	4
P1014	Kavya	1	2	4
P1002	Anita	1	1	10
P1004	Neha	1	1	10
P1006	Priya	1	1	10
P1008	Divya	1	1	10
P1010	Sneha	1	1	10
P1013	Aman	1	1	10

15 rows in set (0.00 sec)

Q6. Rank participants within their department based on the number of events they've registered for (should have 4 columns - department, name, total_number_of_events and rank).

```
SELECT
    p.Department,
    p.Name AS Participant_Name, COUNT(r.Event_ID)
    AS Total_Events,
    RANK() OVER (PARTITION BY p.Department ORDER BY COUNT(r.Event_ID)
DESC) AS Dept_Rank
FROM Participants p
```

JOIN Registration r ON p.SRN = r.SRN
GROUP BY p.Department, p.Name;

Department	Participant_Name	Total_Events	Dept_Rank
AIML	Arjun	1	1
Civil	Manoj	1	1
Civil	Meera	1	1
CSE	Varun	3	1
CSE	Charlie	1	2
CSE	Priya	1	2
CSE	Divya	1	2
CSE	Nisha	1	2
CSE	Kavya	1	2
ECE	Aman	1	1
ECE	Kabir	1	1
EEE	Suresh	1	1
ISE	Sneha	1	1
IT	Neha	1	1
ME	Rahul	1	1
ME	Alia	1	1

16 rows in set (0.00 sec)

Task 4: Full Text Search
Q7. Ranked search: Order events by relevance to the word “Tournament”

ALTER TABLE Event ADD FULLTEXT(event_name);

```
SELECT
    Event_ID,
    Event_name,
    MATCH(Event_name) AGAINST('Tournament' IN NATURAL LANGUAGE MODE) AS
Relevance
FROM Event
WHERE MATCH(Event_name) AGAINST('Tournament' IN NATURAL LANGUAGE MODE)
ORDER BY Relevance DESC;
```

```
+----------+------------------+--------------------+
| Event_ID | Event_name       | Relevance          |
+----------+------------------+--------------------+
| E99      | Coding Tournament | 1.2408697605133057 |
+----------+------------------+--------------------+
1 row in set (0.00 sec)
```

Q8. List stalls offering items whose names match "chicken" or "soup", and compute total stock per stall

```
SELECT
   s.Stall_ID,
   s.Name AS Stall_Name,
   SUM(si.Total_quantity) AS Total_Stock
FROM Stall s
JOIN Stall_items si ON s.Stall_ID = si.Stall_ID
WHERE MATCH(si.Item_desc) AGAINST('chicken soup' IN NATURAL LANGUAGE MODE)
GROUP BY s.Stall_ID, s.Name;
```

```
+----------+--------------+-------------+
| Stall_ID | Stall_Name   | Total_Stock |
+----------+--------------+-------------+
| S3       | SnacksCorner |          25 |
| S5       | CafeCorner   |          20 |
| S7       | BurgerJoint  |          18 |
+----------+--------------+-------------+
3 rows in set (0.00 sec)
```