Ex. No: 05	SUB QUERIES AND CORRELATED SUB QUERIES
Date	13.02.2024

Objective:

To execute the given queries using sub queries and correlated sub queries.

Description:

Sub Queries:

A subquery, also known as an inner query or nested query, is a query that is embedded within another query. It allows you to retrieve data from one table or set of tables based on the results of another query. The subquery is executed first, and its results are then used by the outer query to perform further operations.

Subqueries can be used in various scenarios, such as filtering, joining, or aggregating data. They provide a powerful way to write complex queries and achieve more precise and targeted results. With subqueries, you can break down a problem into smaller, more manageable parts and solve them step by step.

Example of a subquery:

```
SELECT name FROM Customers
WHERE customer_id IN (SELECT customer_id FROM Orders WHERE YEAR(order_date) = 2023)
```

Correlated Subqueries:

A correlated subquery is a special type of subquery where the inner query references the outer query. In other words, the inner query depends on the outer query for its values. This correlation allows the subquery to be evaluated once for each row processed by the outer query. Correlated subqueries are useful when you need to perform calculations or filtering based on values from the outer query. They enable you to compare values between the inner and outer queries and make decisions based on the results.

One common use case for correlated subqueries is finding records that meet specific conditions relative to other records in the same table.

For example, you might use a correlated subquery to find all employees whose salary is higher than the average salary in their department.
Correlated subqueries can be a powerful tool, but they can also impact performance if not used carefully. The database engine needs to execute the subquery multiple times, which can lead to increased resource usage. It's important to optimize and evaluate the performance implications when using correlated subqueries in your queries.

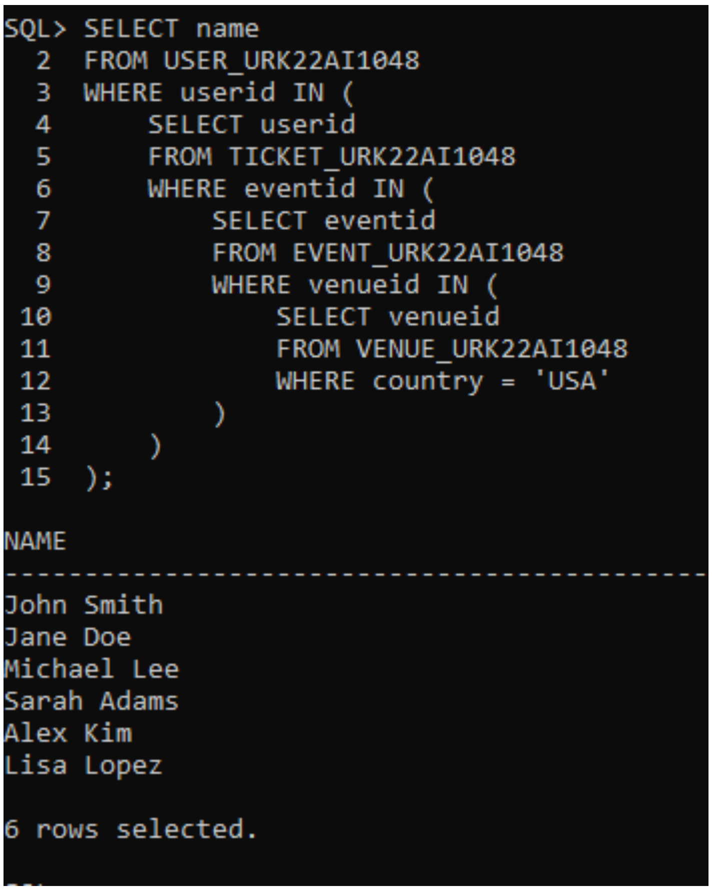
Example of a correlated subquery:

```
SELECT product_id, priceFROM Products p1
WHERE price > (SELECT AVG(price) FROM Products p2)
```

Questions:

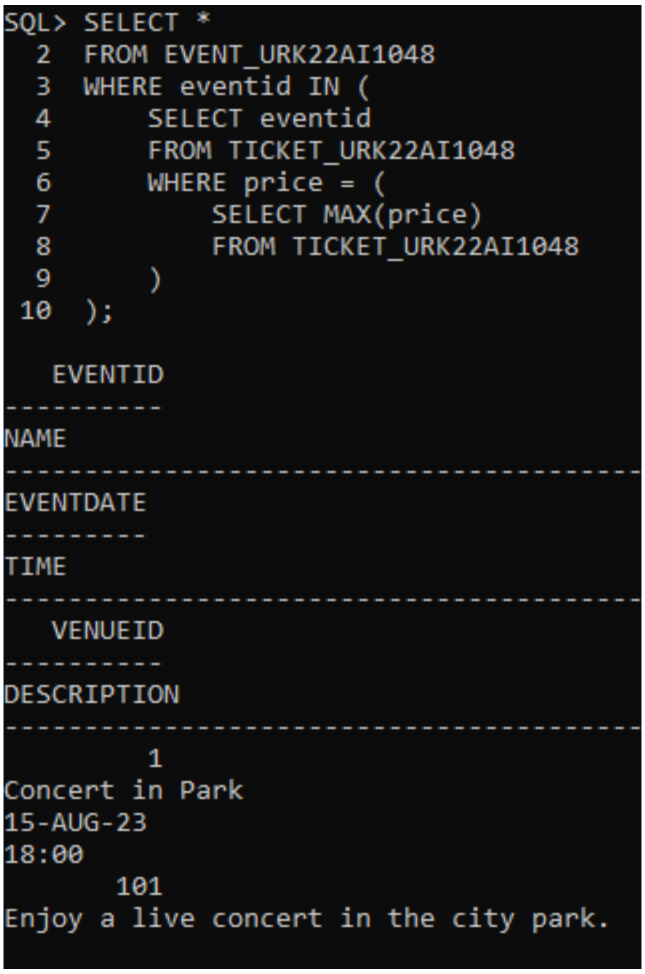
Sub Query

1. List all users who have made reservations for events that are taking place in a specific venue (e.g.,"USA").



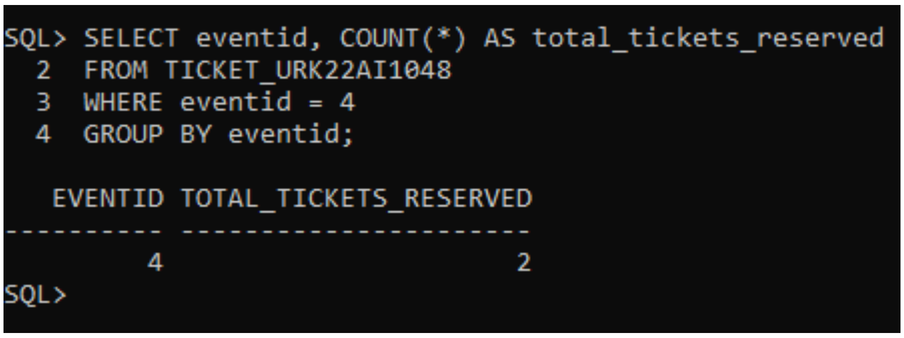
```
SQL> SELECT name
  2  FROM USER_URK22AI1048
  3  WHERE userid IN (
  4      SELECT userid
  5      FROM TICKET_URK22AI1048
  6      WHERE eventid IN (
  7          SELECT eventid
  8          FROM EVENT_URK22AI1048
  9          WHERE venueid IN (
 10              SELECT venueid
 11              FROM VENUE_URK22AI1048
 12              WHERE country = 'USA'
 13          )
 14      )
 15  );

NAME
--------------------------------------------
John Smith
Jane Doe
Michael Lee
Sarah Adams
Alex Kim
Lisa Lopez

6 rows selected.
```

2. Find the events with the highest ticket prices.



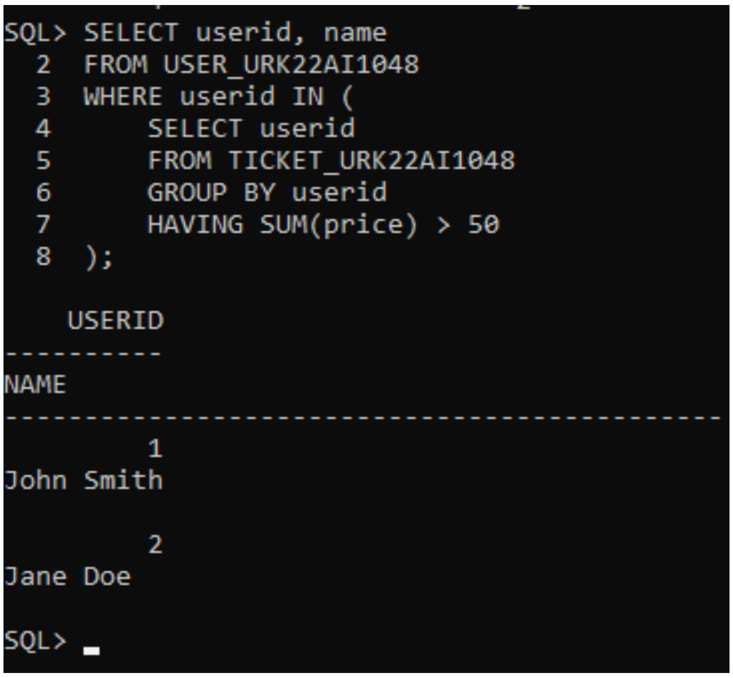
```
SQL> SELECT *
  2  FROM EVENT_URK22AI1048
  3  WHERE eventid IN (
  4      SELECT eventid
  5      FROM TICKET_URK22AI1048
  6      WHERE price = (
  7          SELECT MAX(price)
  8          FROM TICKET_URK22AI1048
  9      )
 10  );

   EVENTID
----------
NAME
----------------------------------------
EVENTDATE
---------
TIME
----------------------------------------
   VENUEID
----------
DESCRIPTION
----------------------------------------
         1
Concert in Park
15-AUG-23
18:00
       101
Enjoy a live concert in the city park.
```

3. Find the total number of tickets reserved for a specific event.



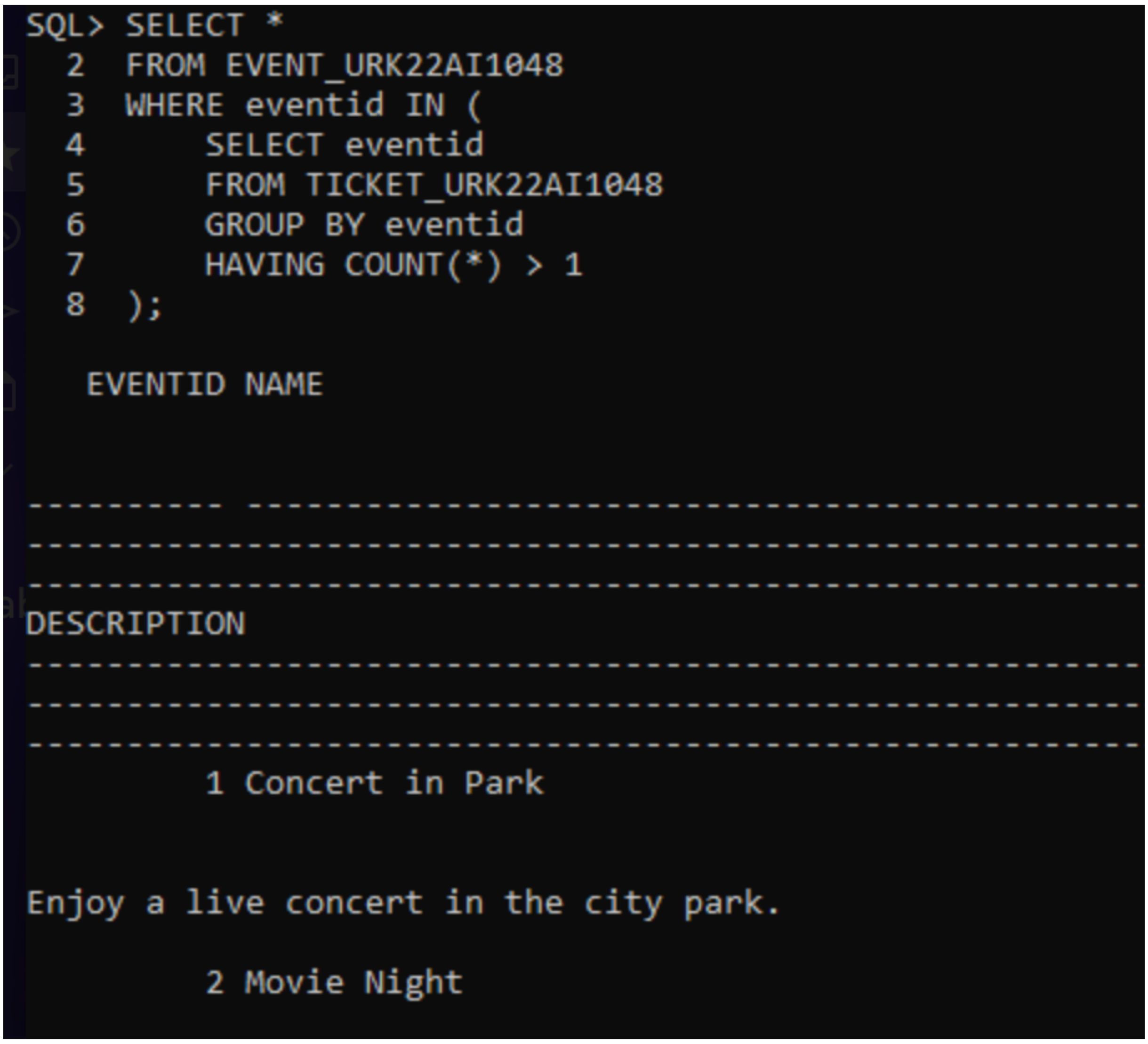
```
SQL> SELECT eventid, COUNT(*) AS total_tickets_reserved
  2  FROM TICKET_URK22AI1048
  3  WHERE eventid = 4
  4  GROUP BY eventid;

   EVENTID TOTAL_TICKETS_RESERVED
---------- ----------------------
         4                      2
SQL>
```

4. List the users who have made reservations with a total cost exceeding a certain amount (e.g.,50)



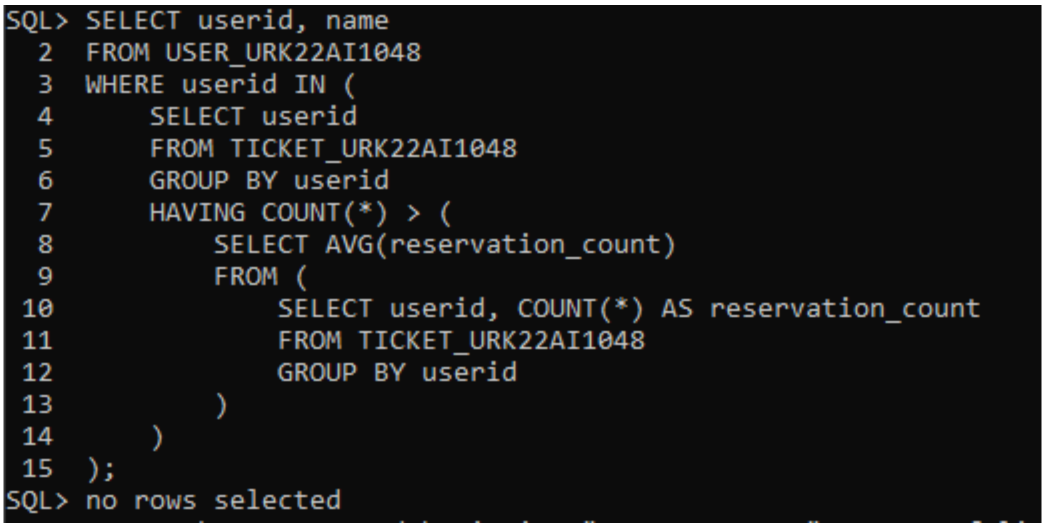
```
SQL> SELECT userid, name
  FROM USER_URK22AI1048
  WHERE userid IN (
      SELECT userid
      FROM TICKET_URK22AI1048
      GROUP BY userid
      HAVING SUM(price) > 50
  );

    USERID
----------
NAME
---------------------------------------------
         1
John Smith

         2
Jane Doe

SQL>
```

5. Retrieve the events where the number of reservations exceeds a certain threshold.



```
SQL> SELECT *
  FROM EVENT_URK22AI1048
  WHERE eventid IN (
      SELECT eventid
      FROM TICKET_URK22AI1048
      GROUP BY eventid
      HAVING COUNT(*) > 1
  );

   EVENTID NAME


---------- ------------------------------------------
----------------------------------------------------
----------------------------------------------------
DESCRIPTION
----------------------------------------------------
----------------------------------------------------
----------------------------------------------------
         1 Concert in Park


Enjoy a live concert in the city park.

         2 Movie Night
```

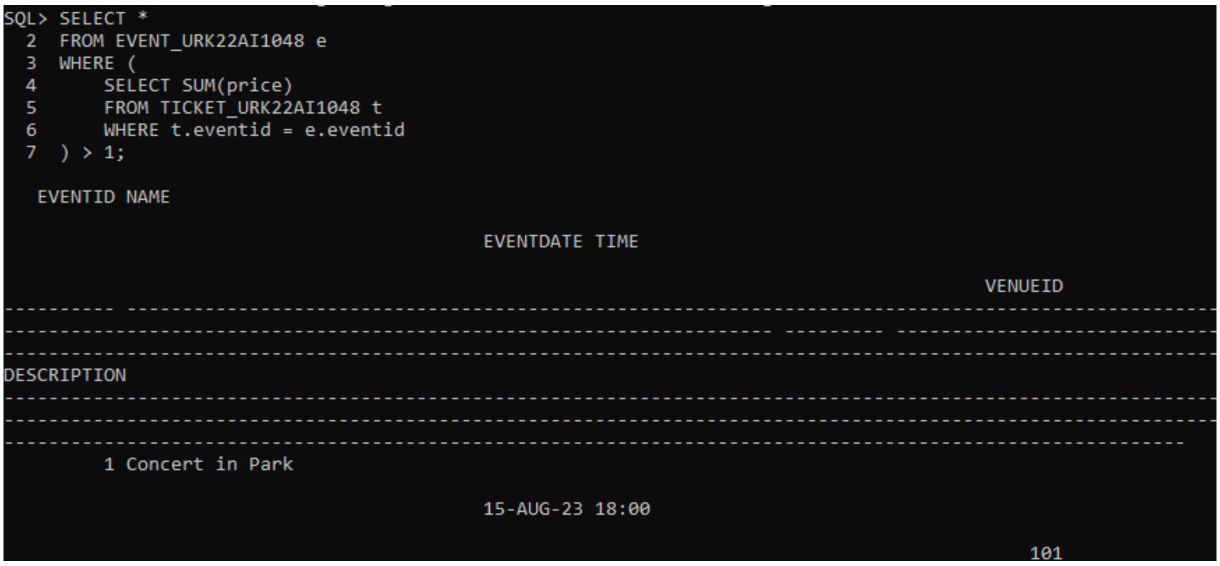
6. Find all users who have made more reservations than the average number of reservations across all users.



```
SQL> SELECT userid, name
  FROM USER_URK22AI1048
  WHERE userid IN (
      SELECT userid
      FROM TICKET_URK22AI1048
      GROUP BY userid
      HAVING COUNT(*) > (
          SELECT AVG(reservation_count)
          FROM (
              SELECT userid, COUNT(*) AS reservation_count
              FROM TICKET_URK22AI1048
              GROUP BY userid
          )
      )
  );
SQL> no rows selected
```

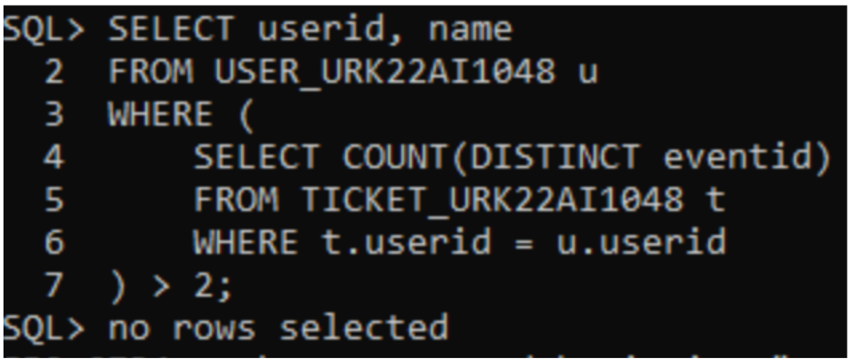
Correlated Sub Query

7. List all events where the total ticket price of reservations exceeds a certain amount.



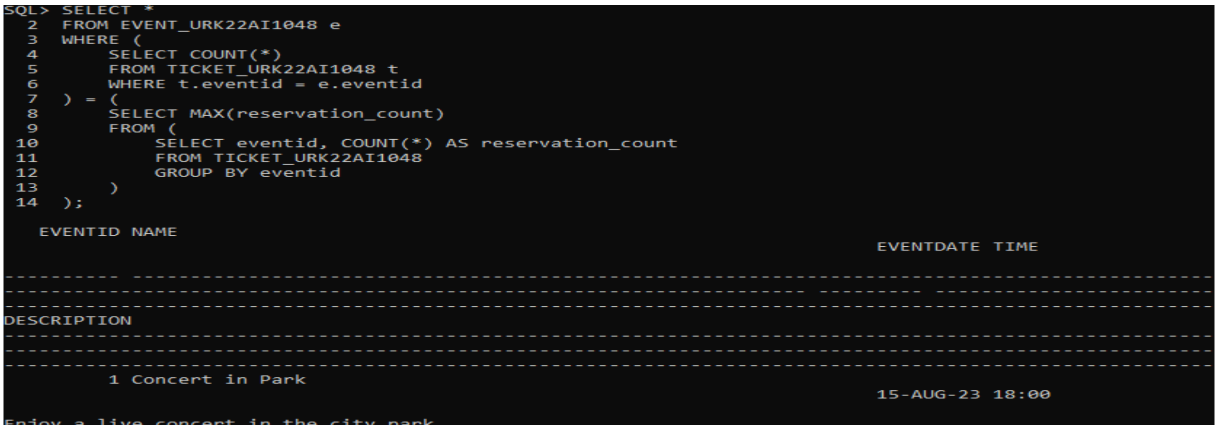
```
SQL> SELECT *
  FROM EVENT_URK22AI1048 e
  WHERE (
      SELECT SUM(price)
      FROM TICKET_URK22AI1048 t
      WHERE t.eventid = e.eventid
  ) > 1;

   EVENTID NAME

                                             EVENTDATE TIME

                                                                                VENUEID
---------- ------------------------------------------------------------------------------
------------------------------------------------------------ --------- ------------------
------------------------------------------------------------------------------------------
DESCRIPTION
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------
         1 Concert in Park

                                             15-AUG-23 18:00

                                                                                    101
```

8. Find the users who have made reservations for more than one event.



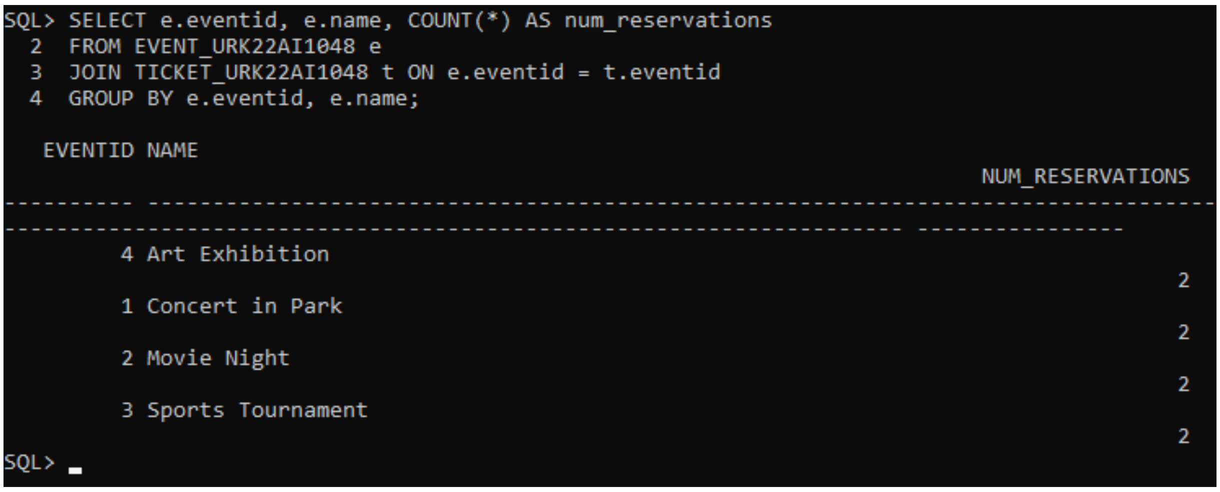
```
SQL> SELECT userid, name
  FROM USER_URK22AI1048 u
  WHERE (
      SELECT COUNT(DISTINCT eventid)
      FROM TICKET_URK22AI1048 t
      WHERE t.userid = u.userid
  ) > 2;
SQL> no rows selected
```

9. Retrieve the events with the highest number of reservations.



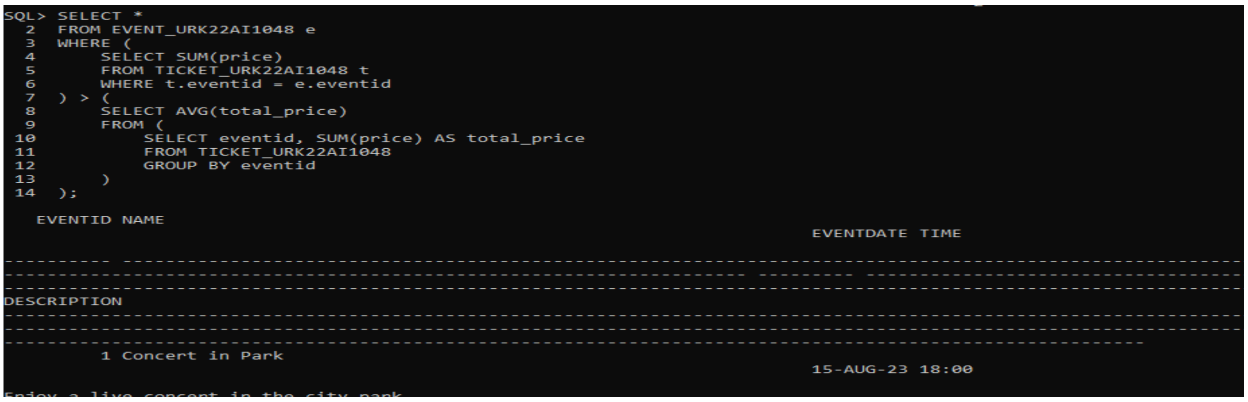
```
SQL> SELECT *
     FROM EVENT_URK22AI1048 e
     WHERE (
         SELECT COUNT(*)
         FROM TICKET_URK22AI1048 t
         WHERE t.eventid = e.eventid
     ) = (
         SELECT MAX(reservation_count)
         FROM (
             SELECT eventid, COUNT(*) AS reservation_count
             FROM TICKET_URK22AI1048
             GROUP BY eventid
         )
     );

   EVENTID NAME
                                                                          EVENTDATE TIME

---------- --------------------------------------------------------------------------------------------
-------------------------------------------------------------------- --------- ----------------------
----------------------------------------------------------------------------------------------------
DESCRIPTION
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
         1 Concert in Park
                                                                          15-AUG-23 18:00

Enjoy a live concert in the city park
```

10. For each event, find the number of reservations made by users



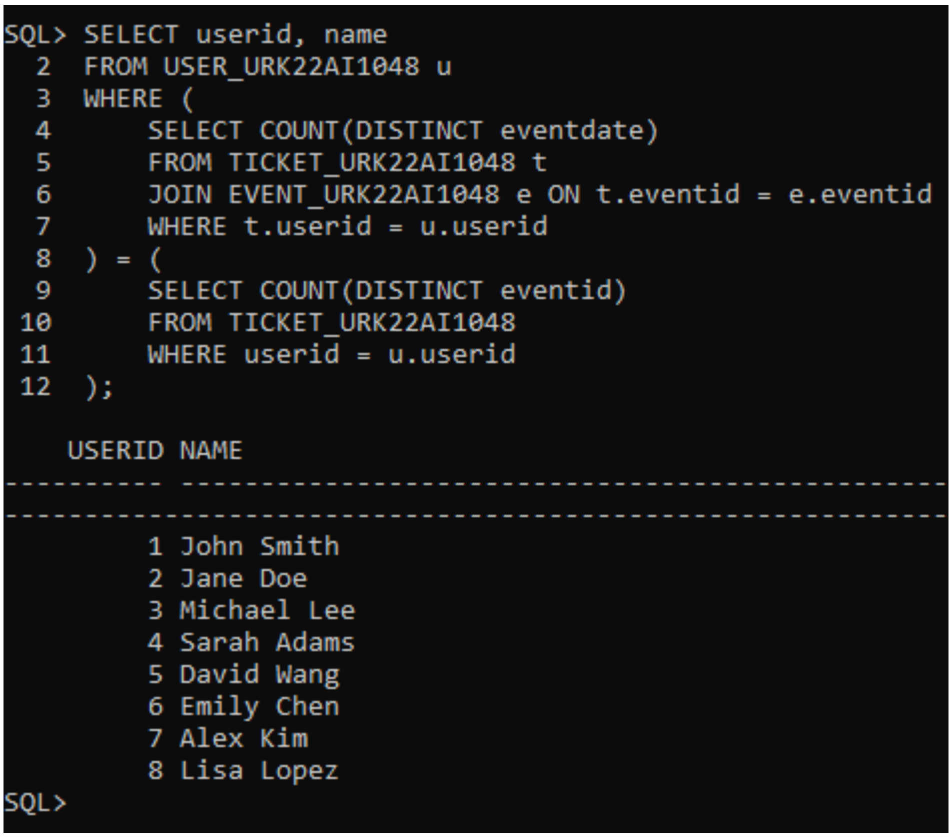
```
SQL> SELECT e.eventid, e.name, COUNT(*) AS num_reservations
     FROM EVENT_URK22AI1048 e
     JOIN TICKET_URK22AI1048 t ON e.eventid = t.eventid
     GROUP BY e.eventid, e.name;

   EVENTID NAME
                                                                          NUM_RESERVATIONS
---------- -----------------------------------------------------------------------------------
------------------------------------------------------------------- ----------------
         4 Art Exhibition
                                                                                        2

         1 Concert in Park
                                                                                        2

         2 Movie Night
                                                                                        2

         3 Sports Tournament
                                                                                        2

SQL> _
```

11. Find the events for which the total ticket price of reservations exceeds the average total ticket price for all events.



```
SQL> SELECT *
     FROM EVENT_URK22AI1048 e
     WHERE (
         SELECT SUM(price)
         FROM TICKET_URK22AI1048 t
         WHERE t.eventid = e.eventid
     ) > (
         SELECT AVG(total_price)
         FROM (
             SELECT eventid, SUM(price) AS total_price
             FROM TICKET_URK22AI1048
             GROUP BY eventid
         )
     );

   EVENTID NAME
                                                                          EVENTDATE TIME

---------- --------------------------------------------------------------------------------------------
-------------------------------------------------------------------- --------- ----------------------
----------------------------------------------------------------------------------------------------
DESCRIPTION
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
         1 Concert in Park
                                                                          15-AUG-23 18:00

Enjoy a live concert in the city park
```

12. List users who have made reservations for multiple events on the same day.



```
SQL> SELECT userid, name
FROM USER_URK22AI1048 u
WHERE (
    SELECT COUNT(DISTINCT eventdate)
    FROM TICKET_URK22AI1048 t
    JOIN EVENT_URK22AI1048 e ON t.eventid = e.eventid
    WHERE t.userid = u.userid
) = (
    SELECT COUNT(DISTINCT eventid)
    FROM TICKET_URK22AI1048
    WHERE userid = u.userid
);

    USERID NAME
---------- --------------------------------------------------
--------------------------------------------------------------
         1 John Smith
         2 Jane Doe
         3 Michael Lee
         4 Sarah Adams
         5 David Wang
         6 Emily Chen
         7 Alex Kim
         8 Lisa Lopez
SQL>
```

Result:

The given queries using sub queries and correlated sub queries were executed.