TASK 4 – TICKET BOOKING SYSTEM

1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery.

SELECT v.venue\_id, v.venue\_name,(SELECT AVG(e.ticket\_price) FROM Event e

-> WHERE e.venue\_id = v.venue\_id) AS average\_ticket\_price FROM Venu v;



2. Find Events with More Than 50% of Tickets Sold using subquery.

mysql> SELECT event\_id, event\_name, total\_seats, available\_seats

-> FROM Event

-> WHERE (total\_seats - available\_seats) <= (

-> SELECT 0.5 \* e.total\_seats

-> FROM Event e

-> WHERE e.event\_id = Event.event\_id

-> );



3. Calculate the Total Number of Tickets Sold for Each Event.

SELECT event\_id, event\_name, (SELECT total\_seats - available\_seats FROM Event e WHERE e.event\_id = Event.event\_id) AS tickets\_sold

FROM Event;



4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.

mysql> SELECT c.customer\_id, c.customer\_name

-> FROM Customers c

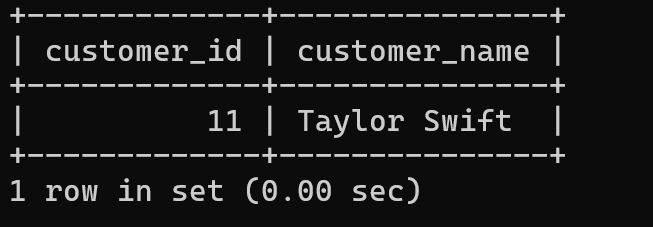
-> WHERE NOT EXISTS (

-> SELECT b.customer\_id

-> FROM Booking b

-> WHERE b.customer\_id = c.customer\_id

-> );



5. List Events with No Ticket Sales Using a NOT IN Subquery.

mysql> SELECT e.event\_id, e.event\_name, e.total\_seats, e.available\_seats

-> FROM Event e

-> WHERE e.event\_id NOT IN (

-> SELECT b.event\_id

-> FROM Booking b

-> );

6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause.

mysql> SELECT e.event\_type, SUM(tickets\_sold) AS total\_tickets\_sold

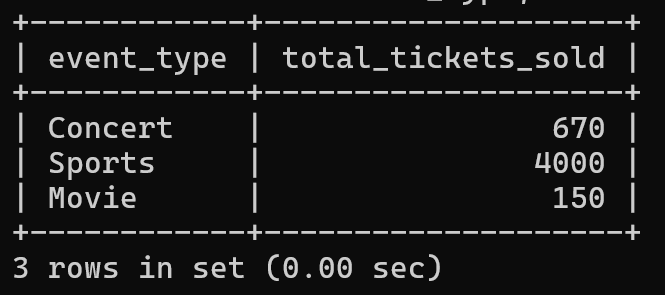
-> FROM (

-> SELECT event\_id, event\_type, (total\_seats - available\_seats) AS tickets\_sold

-> FROM Event )

-> AS e

-> GROUP BY e.event\_type;



7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.

mysql> SELECT event\_id, event\_name, ticket\_price

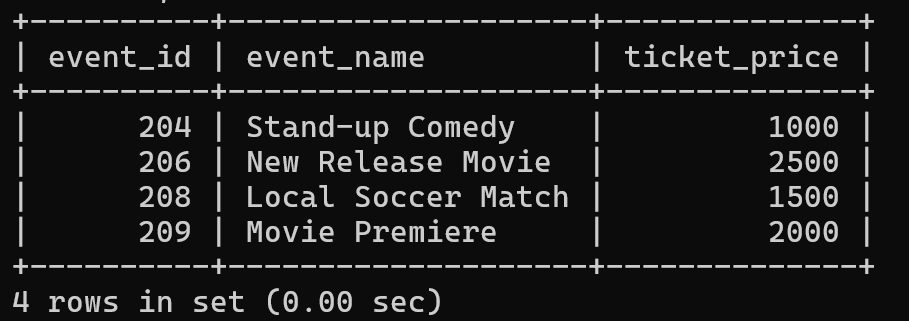
-> FROM Event

-> WHERE ticket\_price > (

-> SELECT AVG(ticket\_price)

-> FROM Event

-> );



8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.

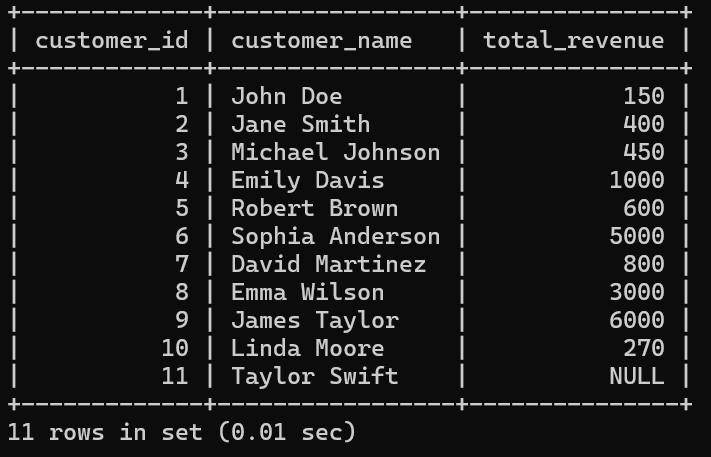
mysql> SELECT c.customer\_id, c.customer\_name, (SELECT SUM(b.num\_tickets \* e.ticket\_price)

-> FROM Booking b

-> INNER JOIN Event e ON b.event\_id = e.event\_id

-> WHERE b.customer\_id = c.customer\_id ) AS total\_revenue

-> FROM Customers c;



9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.

mysql> SELECT c.customer\_id, c.customer\_name, c.email, c.phone\_number

-> FROM Customers c

-> WHERE c.customer\_id IN (

-> SELECT b.customer\_id

-> FROM Booking b

-> WHERE b.event\_id IN (

-> SELECT e.event\_id

-> FROM Event e

-> WHERE e.venue\_id = 101

-> )

-> );



10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.

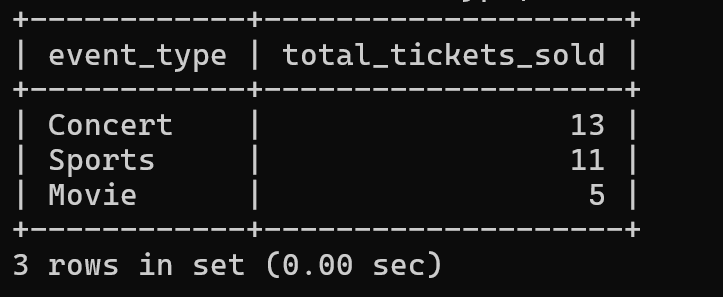
mysql> SELECT e.event\_type,

-> SUM(b.num\_tickets) AS total\_tickets\_sold

-> FROM Event e

-> LEFT JOIN Booking b ON e.event\_id = b.event\_id

-> GROUP BY e.event\_type;



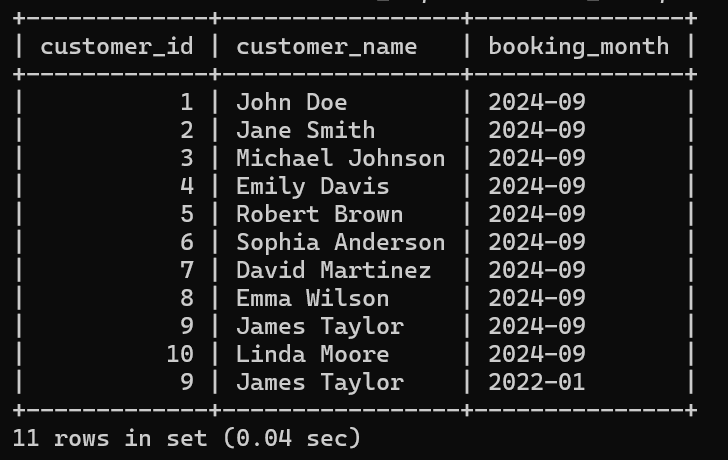
11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE\_FORMAT.

mysql> SELECT c.customer\_id, c.customer\_name, DATE\_FORMAT(b.booking\_date, '%Y-%m') AS booking\_month

-> FROM Customers c

-> Join Booking b ON c.customer\_id = b.customer\_id

-> GROUP BY c.customer\_id, c.customer\_name, DATE\_FORMAT(b.booking\_date, '%Y-%m');



12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery

mysql> SELECT v.venue\_name,

-> (SELECT AVG(e.ticket\_price)

-> FROM Event e

-> WHERE e.venue\_id = v.venue\_id) AS average\_ticket\_price

-> From venu v;

