TICKET_BOOKING_SYSTEM

Work by -

PAVITHRA B

Pavithrabala2203@gmail.com

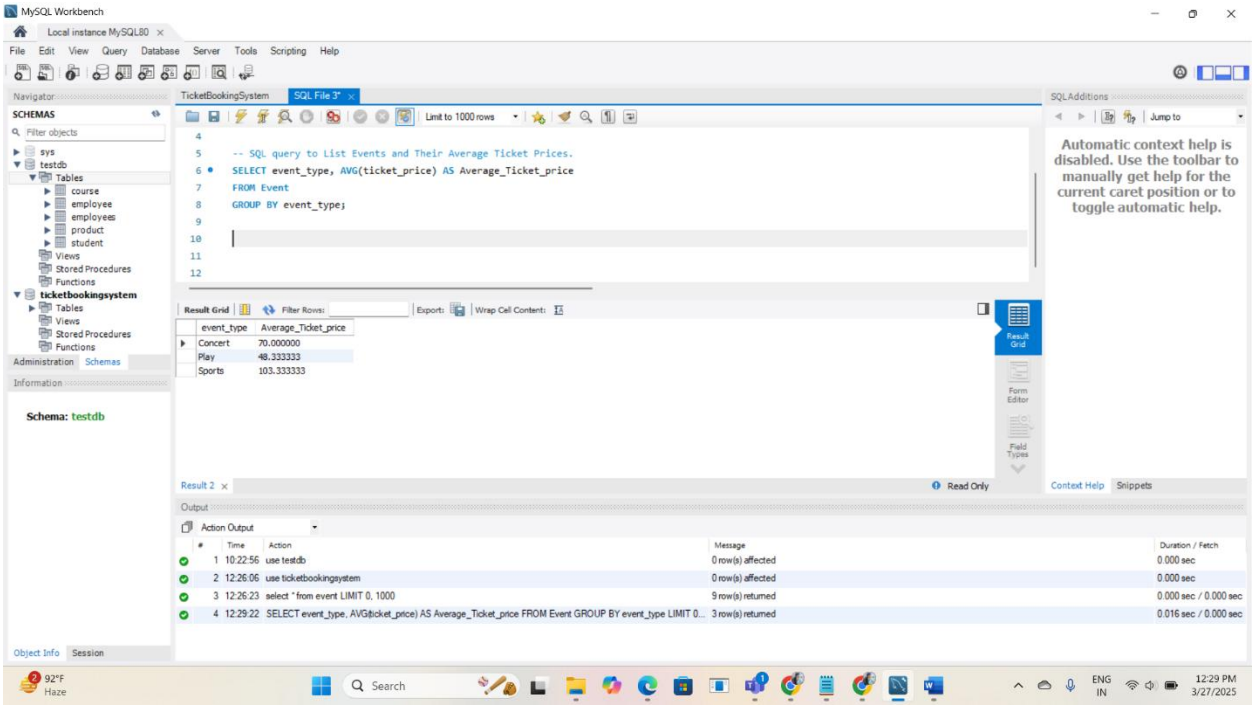
Task 3

1. Write a SQL query to List Events and Their Average Ticket Prices.
2. Write a SQL query to Calculate the Total Revenue Generated by Events.
3. Write a SQL query to find the event with the highest ticket sales.
4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event.
5. Write a SQL query to Find Events with No Ticket Sales.
6. Write a SQL query to Find the User Who Has Booked the Most Tickets.
7. Write a SQL query to List Events and the total number of tickets sold for each month.
8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue.
9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.
10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.
11. Write a SQL query to list users who have booked tickets for multiple events.
12. Write a SQL query to calculate the Total Revenue Generated by Events for Each User.
13. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.
14. Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30 Days.

Screenshots Of Task 3 :

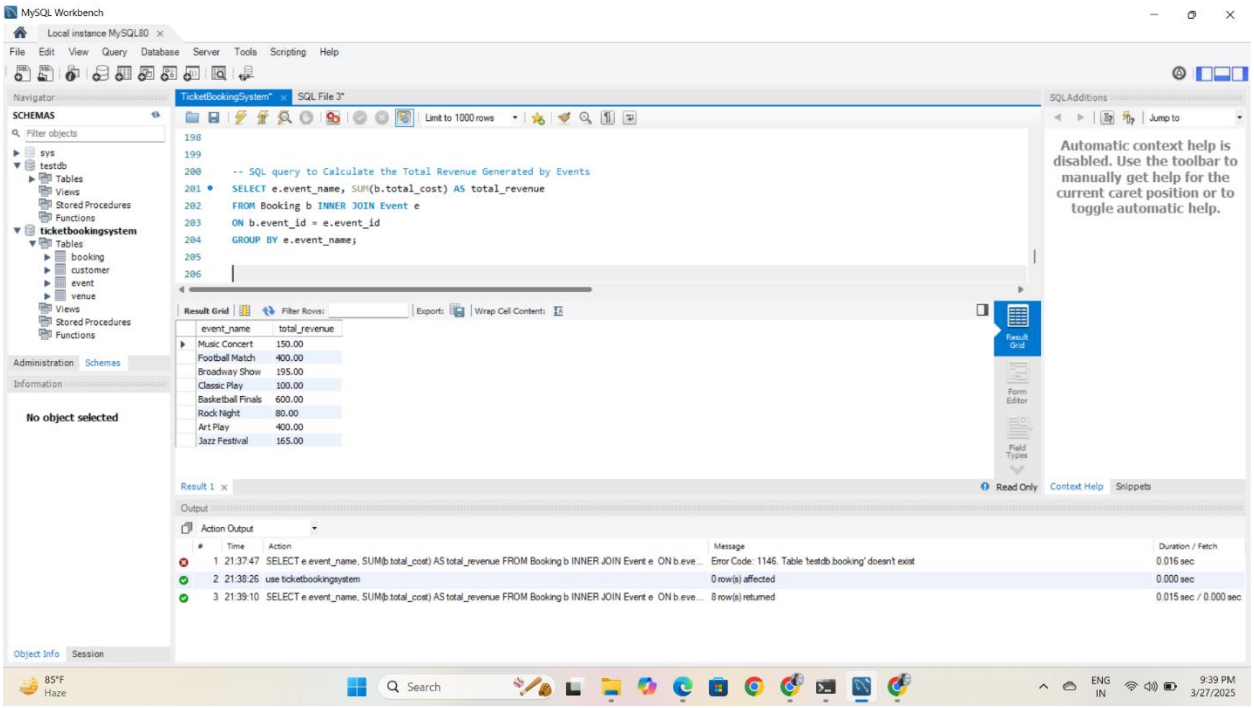
1. SQL query to List Events and Their Average Ticket Prices

```
SELECT event_type, AVG(ticket_price) AS Average_Ticket_price
FROM Event
GROUP BY event_type;
```



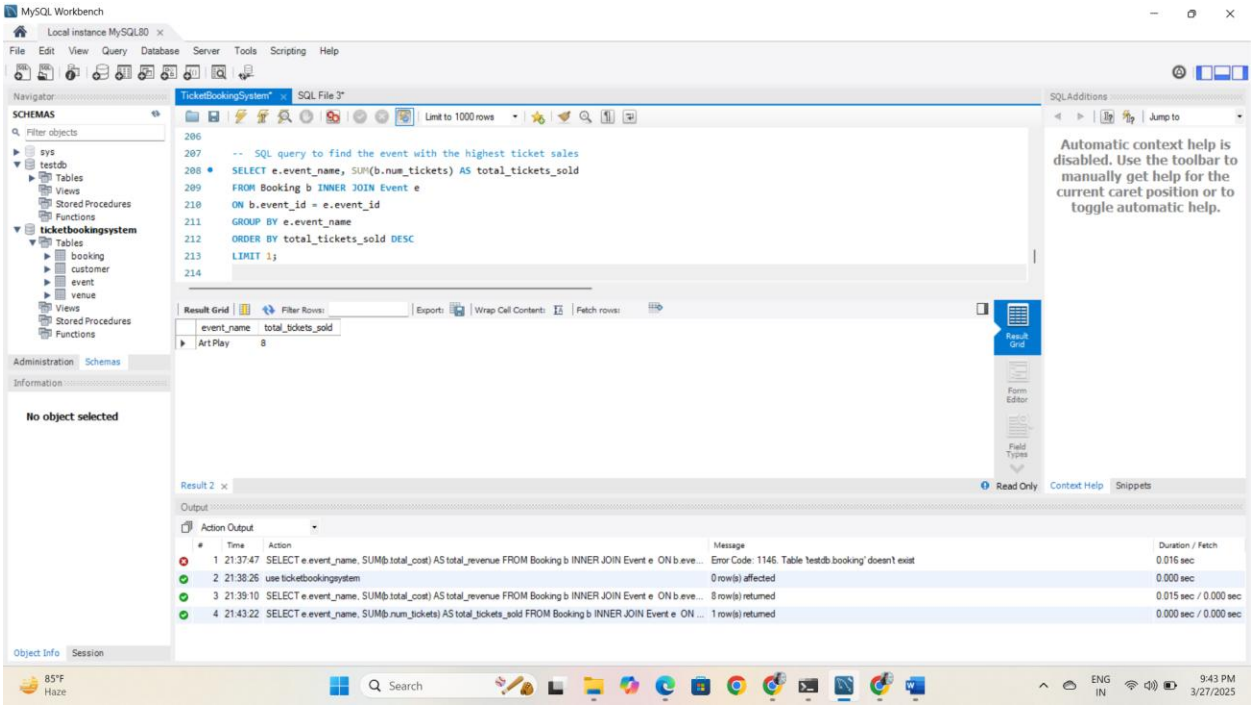
2. SQL query to Calculate the Total Revenue Generated by Events

```
SELECT e.event_name, SUM(b.total_cost) AS total_revenue
FROM Booking b INNER JOIN Event e
ON b.event_id = e.event_id
GROUP BY e.event_name;
```



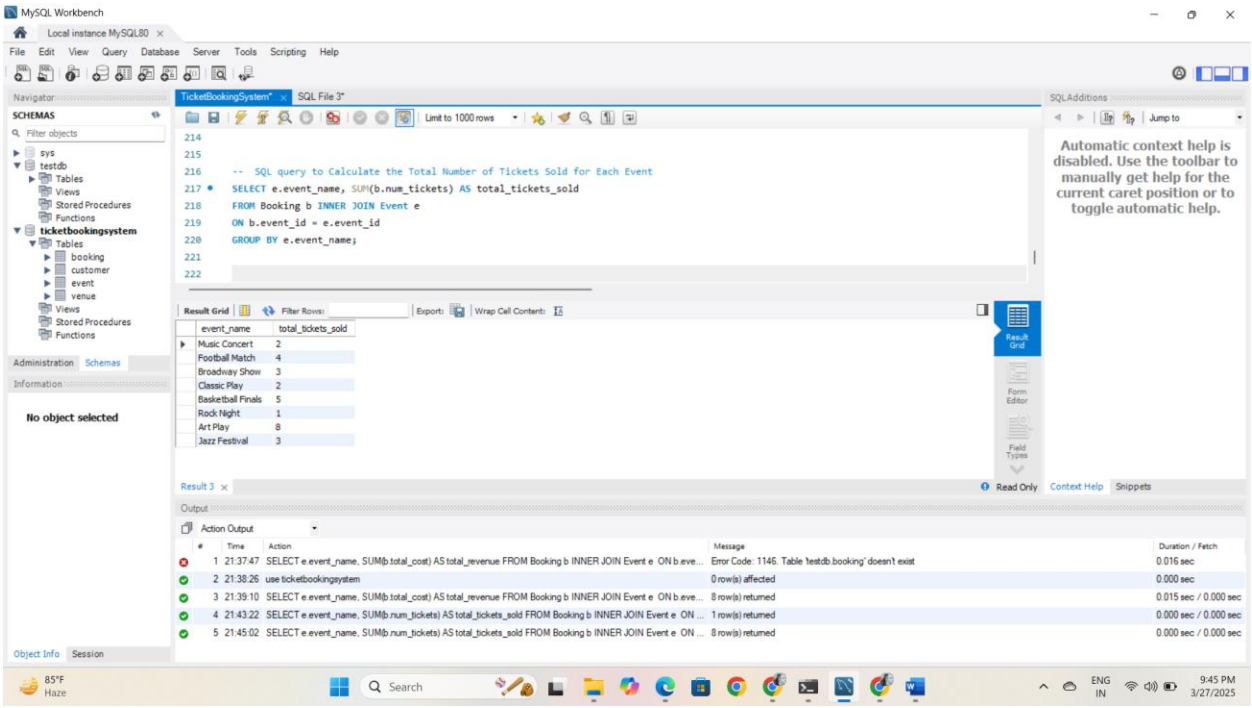
3. SQL query to find the event with the highest ticket sales

```
SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b INNER JOIN Event e
ON b.event_id = e.event_id
GROUP BY e.event_name
ORDER BY total_tickets_sold DESC
LIMIT 1;
```



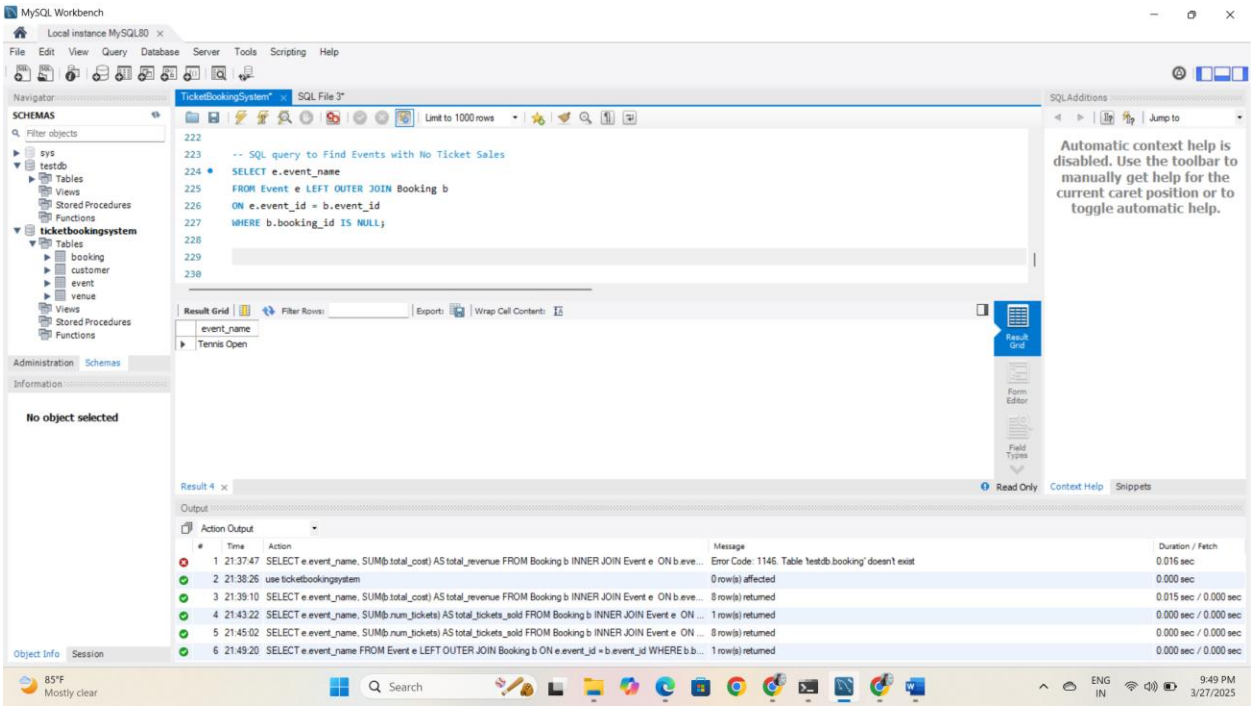
4. SQL query to Calculate the Total Number of Tickets Sold for Each Event

```
SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b INNER JOIN Event e
ON b.event_id = e.event_id
GROUP BY e.event_name;
```



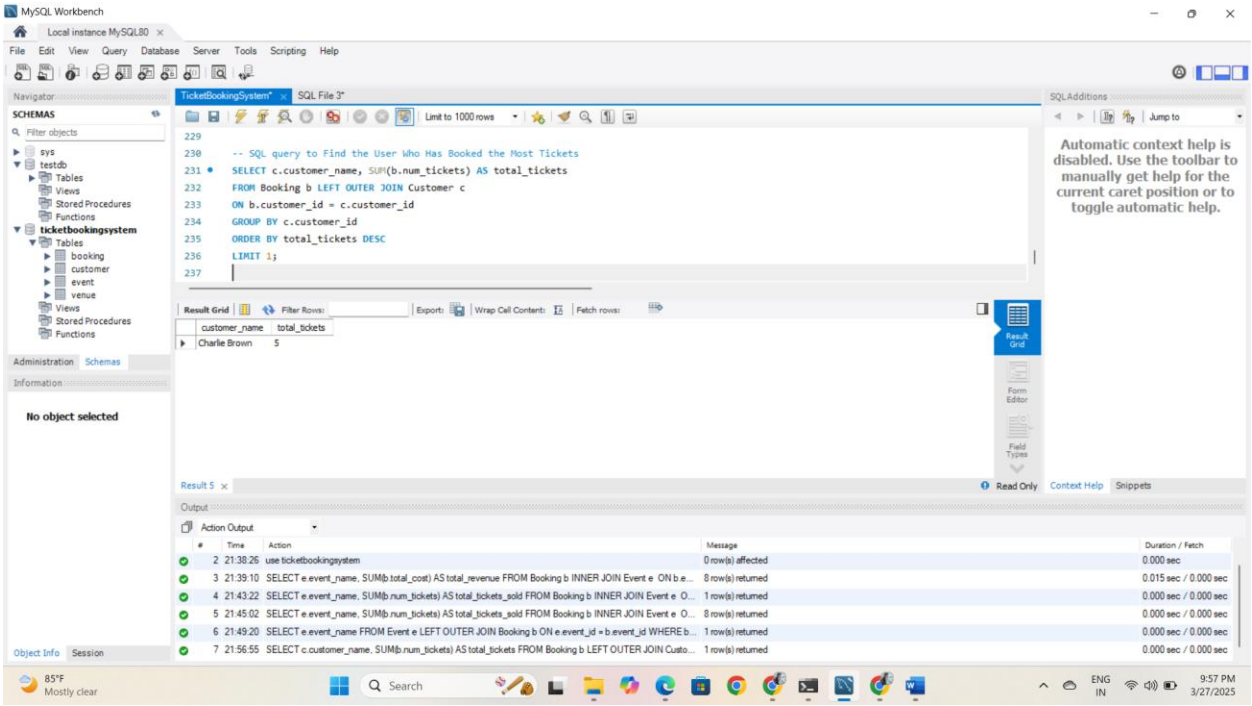
5. SQL query to Find Events with No Ticket Sales

```
SELECT e.event_name
FROM Event e LEFT OUTER JOIN Booking b
ON e.event_id = b.event_id
WHERE b.booking_id IS NULL;
```



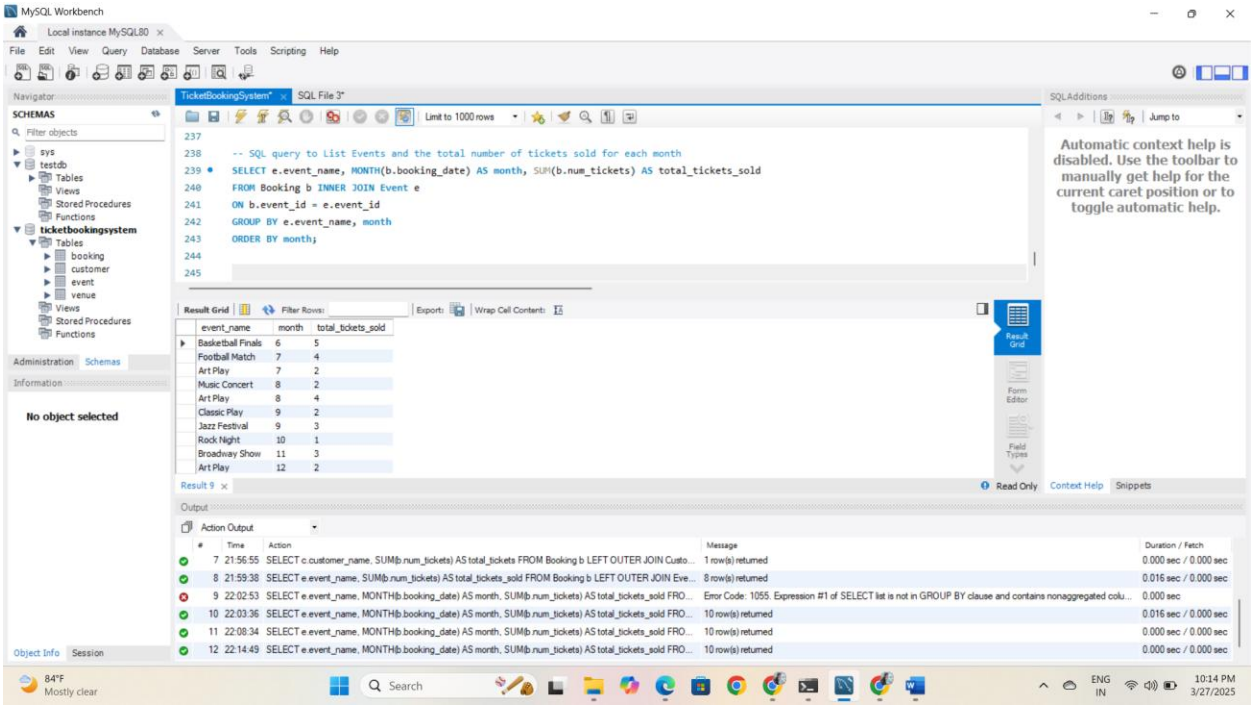
6. SQL query to Find the User Who Has Booked the Most Tickets

```
SELECT c.customer_name, SUM(b.num_tickets) AS total_tickets
FROM Booking b LEFT OUTER JOIN Customer c
ON b.customer_id = c.customer_id
GROUP BY c.customer_id
ORDER BY total_tickets DESC
LIMIT 1;
```



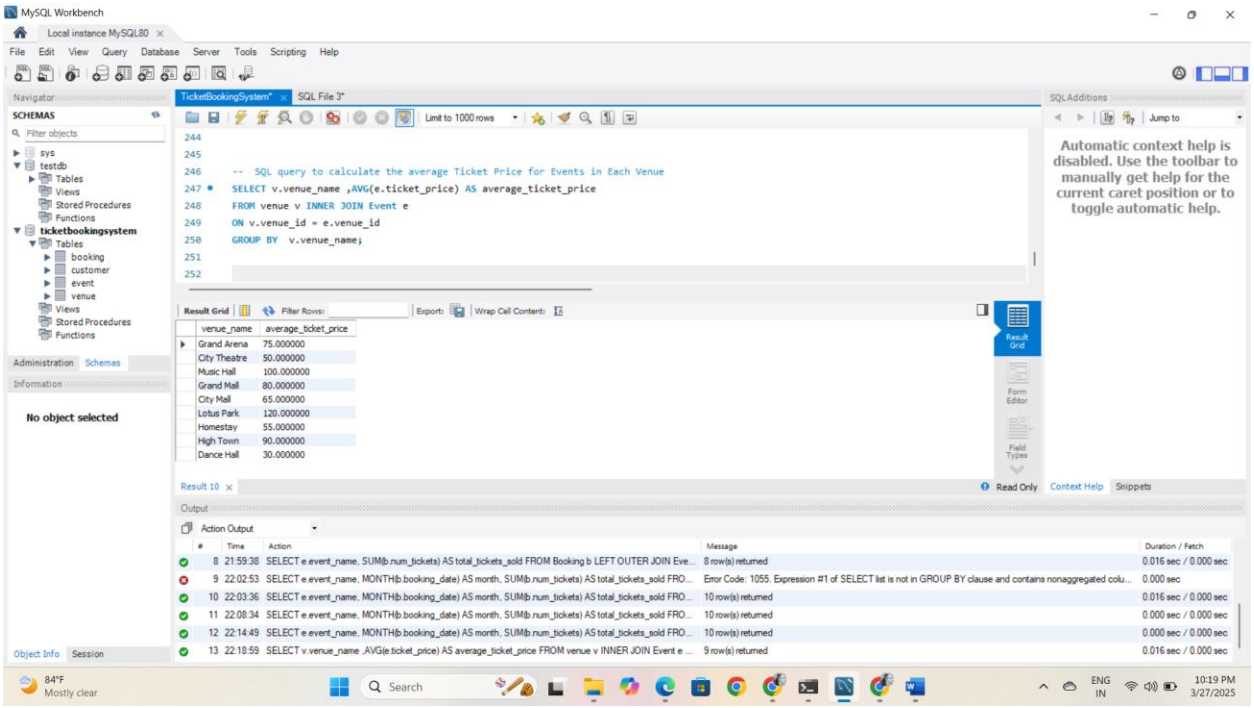
7. SQL query to List Events and the total number of tickets sold for each month

```
SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS
total_tickets_sold
FROM Booking b INNER JOIN Event e
ON b.event_id = e.event_id
GROUP BY e.event_name,month
ORDER BY month;
```



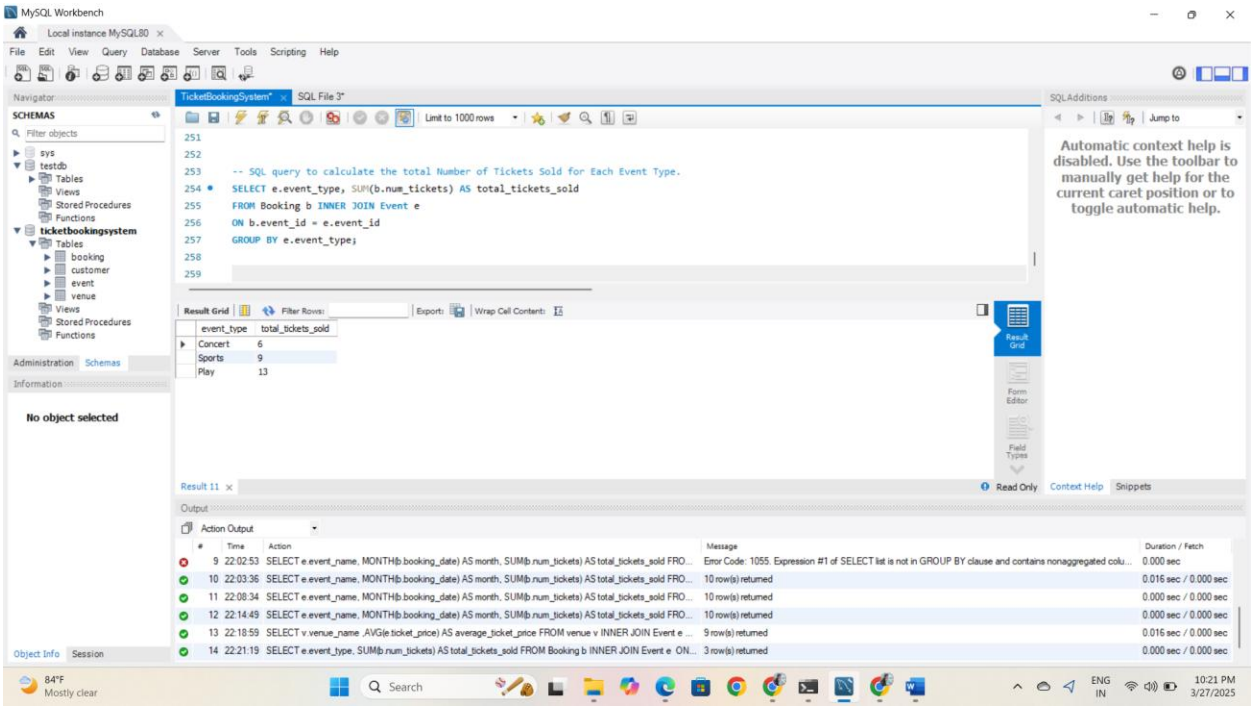
8.SQL query to calculate the average Ticket Price for Events in Each Venue

```
SELECT v.venue_name ,AVG(e.ticket_price) AS average_ticket_price
FROM venue v INNER JOIN Event e
ON v.venue_id = e.venue_id
GROUP BY  v.venue_name;
```



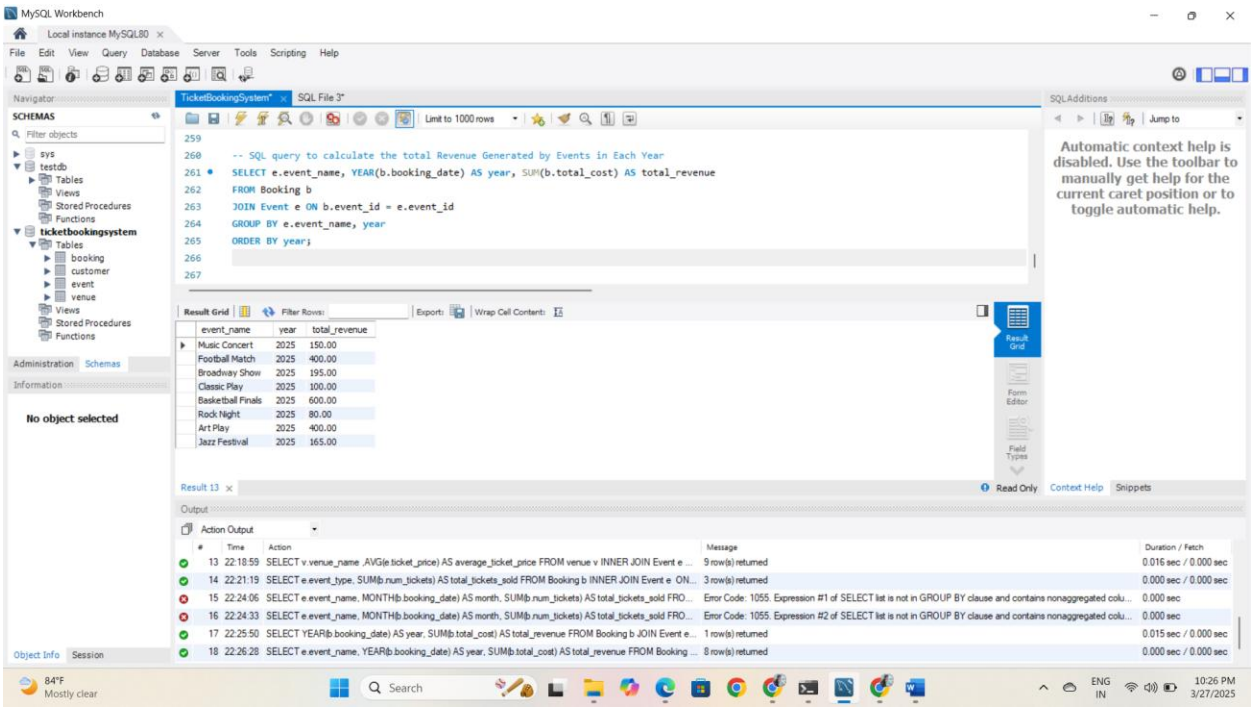
9.SQL query to calculate the total Number of Tickets Sold for Each Event Type

```
SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b INNER JOIN Event e
ON b.event_id = e.event_id
GROUP BY e.event_type;
```



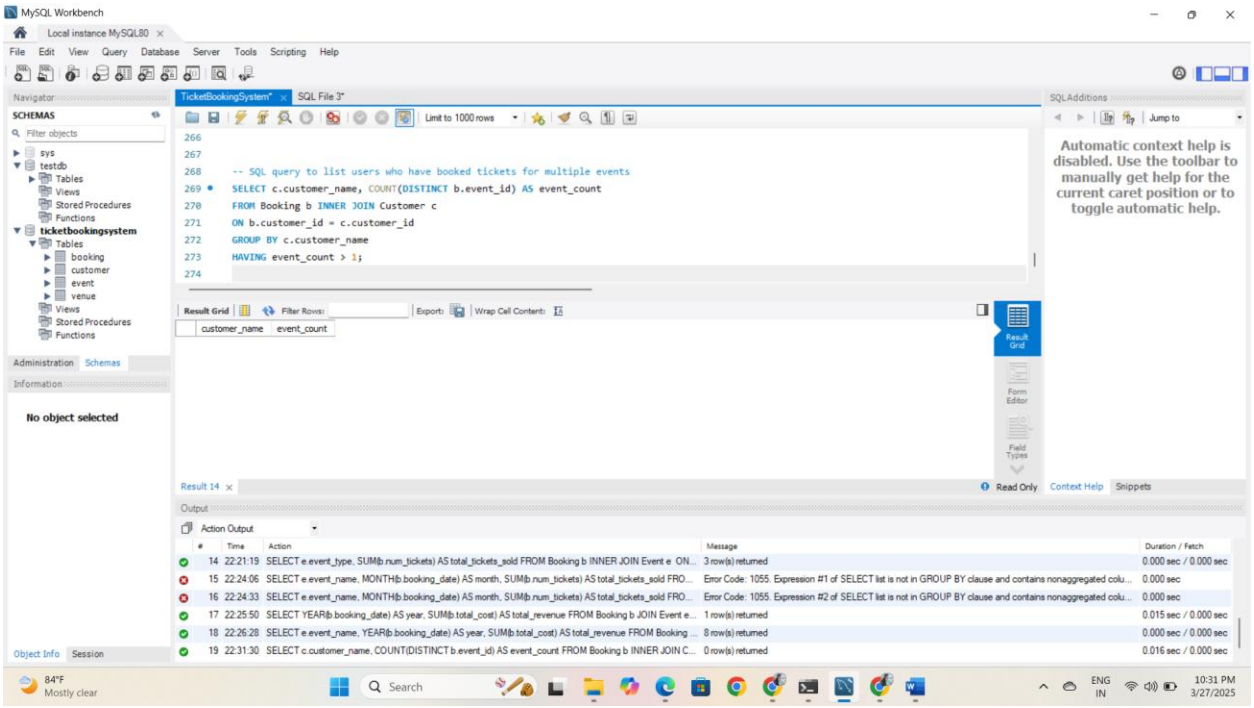
10.SQL query to calculate the total Revenue Generated by Events in Each Year

```
SELECT e.event_name, YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY e.event_name, year
ORDER BY year;
```



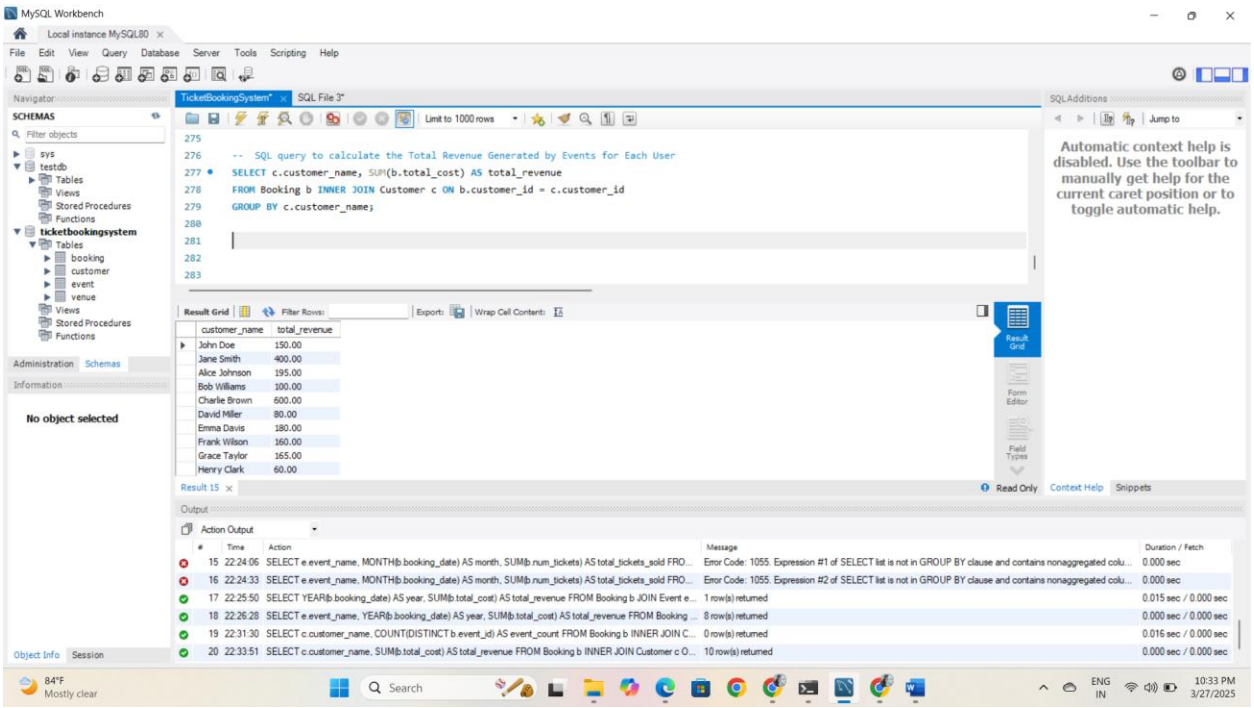
11.SQL query to list users who have booked tickets for multiple events

```
SELECT c.customer_name, COUNT(DISTINCT b.event_id) AS event_count
FROM Booking b INNER JOIN Customer c
ON b.customer_id = c.customer_id
GROUP BY c.customer_name
HAVING event_count > 1;
```



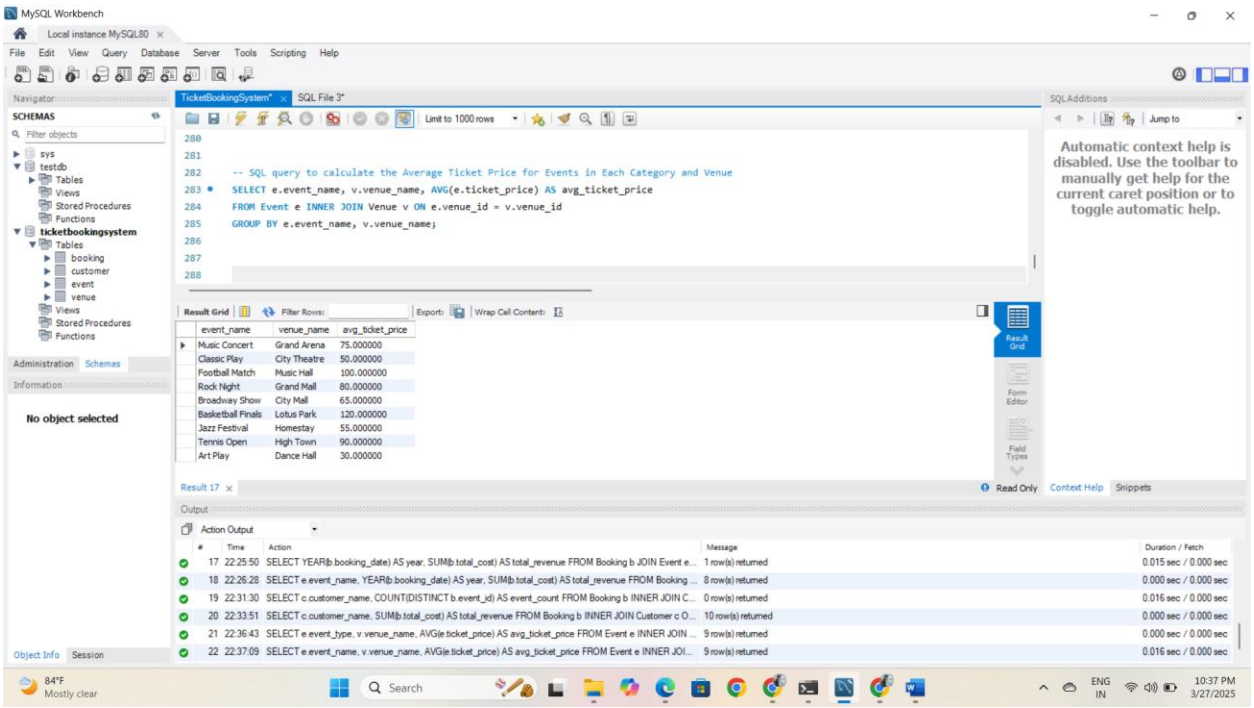
12. SQL query to calculate the Total Revenue Generated by Events for Each User

```
SELECT c.customer_name, SUM(b.total_cost) AS total_revenue

FROM Booking b INNER JOIN Customer c ON b.customer_id = c.customer_id

GROUP BY c.customer_name;
```



13.SQL query to calculate the Average Ticket Price for Events in Each Category and Venue

```
SELECT e.event_name, v.venue_name, AVG(e.ticket_price) AS avg_ticket_price
FROM Event e INNER JOIN Venue v ON e.venue_id = v.venue_id
GROUP BY e.event_name, v.venue_name;
```



14.SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30 Days

```
SELECT c.customer_name, SUM(b.num_tickets) AS Total_Number_Of_Tickets

FROM customer c INNER JOIN booking b

ON c.customer_id =b.customer_id

WHERE b.booking_date >= current_date-30

GROUP BY c.customer_name;
```