

TICKET_BOOKING_SYSTEM

Work by -

PAVITHRA B

Pavithrabala2203@gmail.com

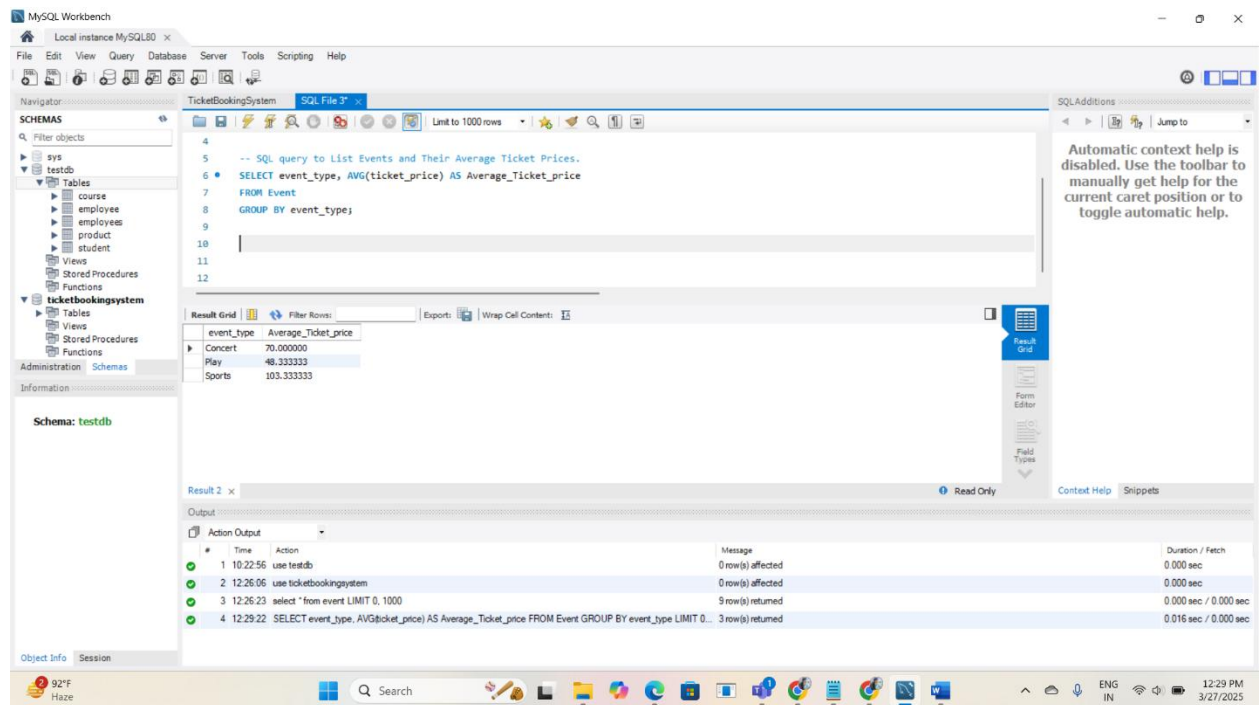
Task 3

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;
```



The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'Schemas' panel with 'testdb' selected. The main editor window contains the following SQL query:

```
-- 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;
```

The 'Result Grid' shows the output of the query:

event_type	Average_Ticket_price
Concert	70.000000
Play	48.333333
Sports	103.333333

The bottom panel shows the 'Action Output' with the following details:

#	Time	Action	Message	Duration / Fetch
1	10:22:56	use testdb	0 row(s) affected	0.000 sec
2	12:26:06	use ticketbookingsystem	0 row(s) affected	0.000 sec
3	12:26:23	select * from event LIMIT 0, 1000	9 row(s) returned	0.000 sec / 0.000 sec
4	12:29:22	SELECT event_type, AVG(ticket_price) AS Average_Ticket_price FROM Event GROUP BY event_type LIMIT 0, 1000	3 row(s) returned	0.016 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
-- 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;
```

The Results grid displays the following data:

event_name	total_revenue
Music Concert	150.00
Football Match	400.00
Broadway Show	195.00
Classic Play	100.00
Basketball Finals	600.00
Rock Night	80.00
Art Play	400.00
Jazz Festival	165.00

The Output tab shows the following messages:

#	Time	Action	Message	Duration / Fetch
1	21:37:47	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	Error Code: 1146. Table 'testdb.booking' doesn't exist	0.015 sec
2	21:38:26	use ticketbookingsystem	0 row(s) affected	0.000 sec
3	21:39:10	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	8 row(s) returned	0.015 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
-- 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;
```

The Result Grid shows the following data:

event_name	total_tickets_sold
Art Play	8

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	21:37:47	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	Error Code: 1146. Table 'testdb.booking' doesn't exist	0.016 sec
2	21:38:26	use ticketbookingsystem	0 row(s) affected	0.000 sec
3	21:39:10	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	8 row(s) returned	0.015 sec / 0.000 sec
4	21:43:22	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	1 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
-- 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;
```

The Results grid displays the following data:

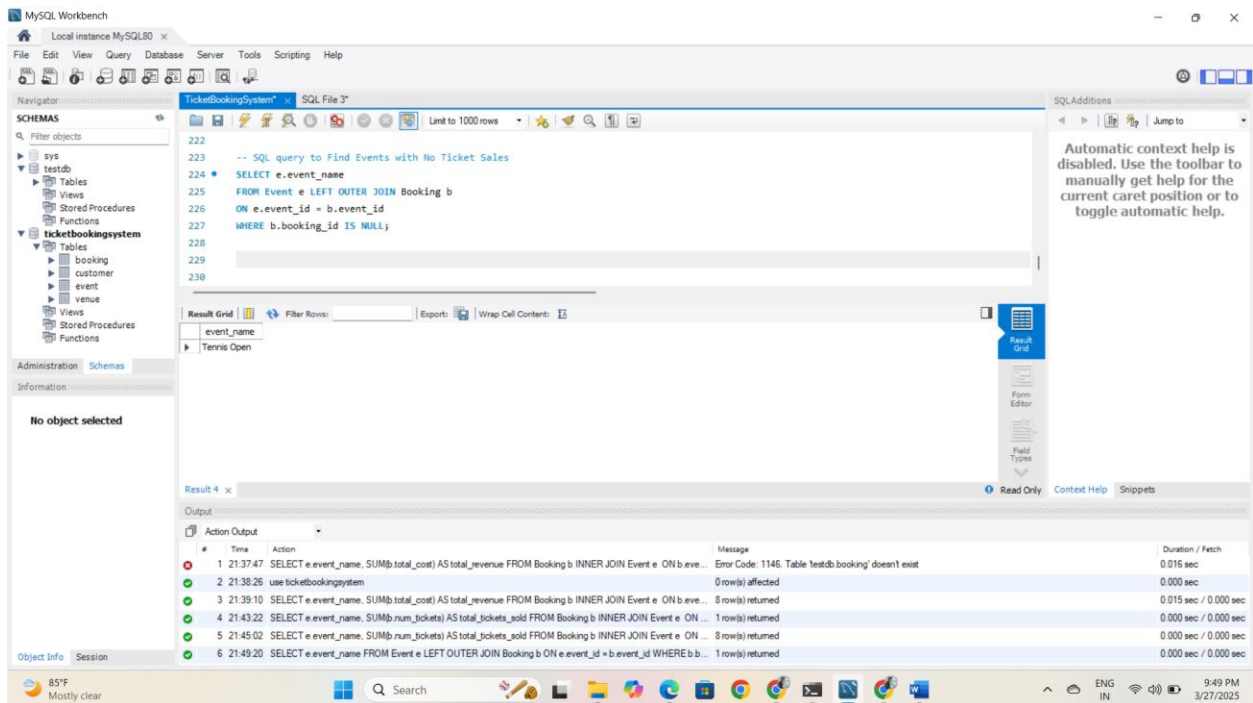
event_name	total_tickets_sold
Music Concert	2
Football Match	4
Broadway Show	3
Classic Play	2
Basketball Finals	5
Rock Night	1
Art Play	8
Jazz Festival	3

The Output tab shows the execution log with the following messages:

#	Time	Action	Message	Duration / Fetch
1	21:37:47	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	Error Code: 1146. Table 'testdb.booking' doesn't exist	0.016 sec
2	21:38:26	use ticketbookingsystem	0 row(s) affected	0.000 sec
3	21:39:10	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	8 row(s) returned	0.015 sec / 0.000 sec
4	21:43:22	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	1 row(s) returned	0.000 sec / 0.000 sec
5	21:45:02	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	8 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot displays the MySQL Workbench interface. The SQL Editor window contains the following query:

```
-- 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;
```

The Results window shows the output of the query:

customer_name	total_tickets
Charlie Brown	5

The bottom panel shows the Action Output window with the following log:

#	Time	Action	Message	Duration / Fetch
2	21:38:26	use ticketbookingsystem	0 row(s) affected	0.000 sec
3	21:39:10	SELECT e.event_name, SUM(b.total_cost) AS total_revenue FROM Booking b INNER JOIN Event e ON b.e...	8 row(s) returned	0.015 sec / 0.000 sec
4	21:43:22	SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b INNER JOIN Event e O...	1 row(s) returned	0.000 sec / 0.000 sec
5	21:45:02	SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b INNER JOIN Event e O...	8 row(s) returned	0.000 sec / 0.000 sec
6	21:45:20	SELECT e.event_name FROM Event e LEFT OUTER JOIN Booking b ON e.event_id = b.event_id WHERE b...	1 row(s) returned	0.000 sec / 0.000 sec
7	21:56:55	SELECT c.customer_name, SUM(b.num_tickets) AS total_tickets FROM Booking b LEFT OUTER JOIN Custo...	1 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
-- 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;
```

The Results Grid displays the following data:

event_name	month	total_tickets_sold
Basketball Finals	6	5
Football Match	7	4
Art Play	7	2
Music Concert	8	2
Art Play	8	4
Classic Play	9	2
Jazz Festival	9	3
Rock Night	10	1
Broadway Show	11	3
Art Play	12	2

The Output tab shows the execution log with the following messages:

#	Time	Action	Message	Duration / Fetch
7	21:56:55	SELECT e.customer_name, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b LEFT OUTER JOIN Cust...	1 row(s) returned	0.000 sec / 0.000 sec
8	21:59:38	SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b LEFT OUTER JOIN Eve...	8 row(s) returned	0.016 sec / 0.000 sec
9	22:02:53	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
10	22:03:36	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.016 sec / 0.000 sec
11	22:06:34	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.000 sec / 0.000 sec
12	22:14:49	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot displays the MySQL Workbench interface. The SQL Editor window shows a query to calculate the average ticket price for events in each venue. The Results window shows the output of the query, which is a table with two columns: venue_name and average_ticket_price. The Output window shows the execution log, including the query and its results.

SQL Query:

```
-- 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;
```

Result Grid:

venue_name	average_ticket_price
Grand Arena	75.000000
City Theatre	50.000000
Music Hall	100.000000
Grand Mall	80.000000
City Mall	65.000000
Lobus Park	120.000000
Homeslay	55.000000
High Town	90.000000
Dance Hall	30.000000

Output:

#	Time	Action	Message	Duration / Fetch
8	21:59:38	SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b LEFT OUTER JOIN Eve...	8 row(s) returned	0.016 sec / 0.000 sec
9	22:02:53	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
10	22:03:36	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.016 sec / 0.000 sec
11	22:08:34	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.000 sec / 0.000 sec
12	22:14:49	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	10 row(s) returned	0.000 sec / 0.000 sec
13	22:18:59	SELECT v.venue_name ,AVG(e.ticket_price) AS average_ticket_price FROM venue v INNER JOIN Event e ...	9 row(s) returned	0.016 sec / 0.000 sec

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;
```

The screenshot displays the MySQL Workbench interface. The SQL Editor window contains the following query:

```
-- 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;
```

The Results Grid shows the output of the query:

event_type	total_tickets_sold
Concert	6
Sports	9
Play	13

The Output window shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
9	22:02:53	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
10	22:03:36	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	10 row(s) returned	0.016 sec / 0.000 sec
11	22:08:34	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	10 row(s) returned	0.000 sec / 0.000 sec
12	22:14:49	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	10 row(s) returned	0.000 sec / 0.000 sec
13	22:18:59	SELECT v.venue_name, AVG(t.ticket_price) AS average_ticket_price FROM venue v INNER JOIN Event e ...	9 row(s) returned	0.016 sec / 0.000 sec
14	22:21:19	SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b INNER JOIN Event e ON ...	3 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot displays the MySQL Workbench interface. The SQL Editor window contains the following query:

```
-- 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;
```

The Results Grid shows the output of the query:

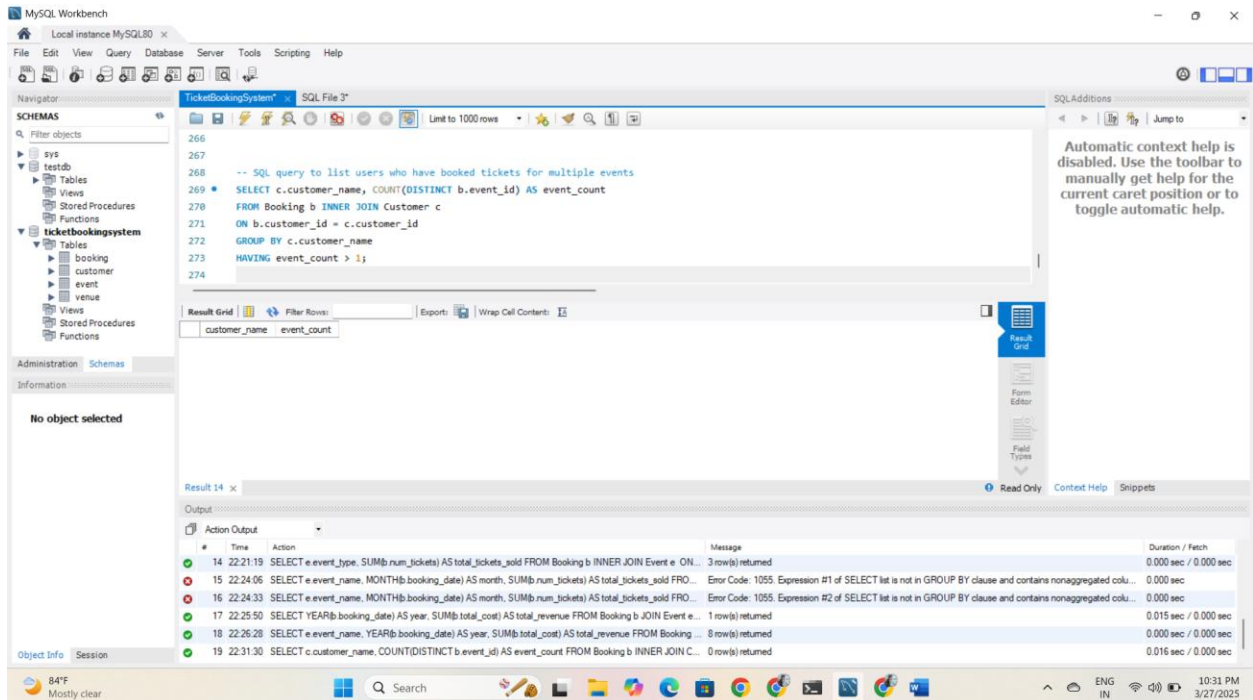
event_name	year	total_revenue
Music Concert	2025	150.00
Football Match	2025	400.00
Broadway Show	2025	195.00
Classic Play	2025	100.00
Basketball Finals	2025	600.00
Rock Night	2025	80.00
Art Play	2025	400.00
Jazz Festival	2025	165.00

The Output window shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
13	22:18:59	SELECT v.venue_name, AVG(t.ticket_price) AS average_ticket_price FROM venue v INNER JOIN Event e ...	9 row(s) returned	0.016 sec / 0.000 sec
14	22:21:19	SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold FROM Booking b INNER JOIN Event e ON ...	3 row(s) returned	0.000 sec / 0.000 sec
15	22:24:06	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
16	22:24:33	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FRO...	Error Code: 1055. Expression #2 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
17	22:25:50	SELECT YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking b JOIN Event e ...	1 row(s) returned	0.015 sec / 0.000 sec
18	22:26:28	SELECT e.event_name, YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking ...	8 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
-- 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;
```

The Results window displays the output of the query:

customer_name	total_revenue
John Doe	150.00
Jane Smith	400.00
Alice Johnson	195.00
Bob Williams	100.00
Charlie Brown	600.00
David Miller	80.00
Emma Davis	180.00
Frank Wilson	160.00
Grace Taylor	165.00
Henry Clark	60.00

The Output window shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
15	22:24:06	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
16	22:24:33	SELECT e.event_name, MONTH(b.booking_date) AS month, SUM(b.num_tickets) AS total_tickets_sold FROM ...	Error Code: 1055. Expression #2 of SELECT list is not in GROUP BY clause and contains nonaggregated colu...	0.000 sec
17	22:25:50	SELECT YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking b JOIN Event e ...	1 row(s) returned	0.015 sec / 0.000 sec
18	22:26:28	SELECT e.event_name, YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking ...	8 row(s) returned	0.000 sec / 0.000 sec
19	22:31:30	SELECT c.customer_name, COUNT(DISTINCT b.event_id) AS event_count FROM Booking b INNER JOIN C...	0 row(s) returned	0.016 sec / 0.000 sec
20	22:33:51	SELECT c.customer_name, SUM(b.total_cost) AS total_revenue FROM Booking b INNER JOIN Customer c O...	10 row(s) returned	0.000 sec / 0.000 sec

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;
```

The screenshot displays the MySQL Workbench interface. The 'SQL File 3*' tab contains the following query:

```
-- 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;
```

The 'Result Grid' shows the following data:

event_name	venue_name	avg_ticket_price
Music Concert	Grand Arena	75.000000
Classic Play	City Theatre	50.000000
Football Match	Music Hall	100.000000
Rock Night	Grand Mall	80.000000
Broadway Show	City Mall	65.000000
Basketball Finals	Lakes Park	120.000000
Jazz Festival	Homesday	55.000000
Tennis Open	High Town	90.000000
Art Play	Dance Hall	30.000000

The 'Output' pane shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
17	22:25:50	SELECT YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking b JOIN Event e...	1 row(s) returned	0.015 sec / 0.000 sec
18	22:26:28	SELECT e.event_name, YEAR(b.booking_date) AS year, SUM(b.total_cost) AS total_revenue FROM Booking ...	8 row(s) returned	0.000 sec / 0.000 sec
19	22:31:30	SELECT c.customer_name, COUNT(DISTINCT b.event_id) AS event_count FROM Booking b INNER JOIN C...	0 row(s) returned	0.016 sec / 0.000 sec
20	22:33:51	SELECT c.customer_name, SUM(b.total_cost) AS total_revenue FROM Booking b INNER JOIN Customer c O...	10 row(s) returned	0.000 sec / 0.000 sec
21	22:36:43	SELECT e.event_name, v.venue_name, AVG(e.ticket_price) AS avg_ticket_price FROM Event e INNER JOIN ...	9 row(s) returned	0.000 sec / 0.000 sec
22	22:37:09	SELECT e.event_name, v.venue_name, AVG(e.ticket_price) AS avg_ticket_price FROM Event e INNER JOIN ...	9 row(s) returned	0.016 sec / 0.000 sec

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;
```

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the query to list users and their total tickets. The Results Grid shows the output of the query. The Output panel shows the execution log.

SQL Query:

```
-- 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;
```

Results Grid:

customer_name	Total_Number_Of_Tickets
John Doe	2
Jane Smith	4
Alice Johnson	3
Bob Williams	2
Charlie Brown	5
David Miller	1
Emma Davis	2
Frank Wilson	4
Grace Taylor	3
Henry Clark	2

Output Panel:

#	Time	Action	Message	Duration / Fetch
22	22:37:09	SELECT e.event_name, v.venue_name, AVG(ticket_price) AS avg_ticket_price FROM Event e INNER JOIN booking b ON e.event_id = b.event_id	9 row(s) returned	0.016 sec / 0.000 sec
23	22:44:55	SELECT c.customer_name, SUM(b.num_tickets) FROM c.customer INNER JOIN b.booking ON c.customer_id = b.customer_id	Error Code: 1049. Unknown database 'c'	0.000 sec
24	22:45:16	SELECT c.customer_name, SUM(b.num_tickets) FROM c.customer INNER JOIN b.booking ON c.customer_id = b.customer_id	Error Code: 1049. Unknown database 'c'	0.000 sec
25	22:45:40	SELECT c.customer_name, SUM(b.num_tickets) FROM c.customer INNER JOIN b.booking ON c.customer_id = b.customer_id	Error Code: 1054. Unknown column 'b.booking_date' in 'where clause'	0.000 sec
26	22:45:51	SELECT c.customer_name, SUM(b.num_tickets) FROM c.customer INNER JOIN b.booking ON c.customer_id = b.customer_id	10 row(s) returned	0.000 sec / 0.000 sec
27	22:46:33	SELECT c.customer_name, SUM(b.num_tickets) AS Total_Number_Of_Tickets FROM c.customer INNER JOIN b.booking ON c.customer_id = b.customer_id	10 row(s) returned	0.000 sec / 0.000 sec