TICKET BOOKING SYSTEM

Task 1: Database Design:

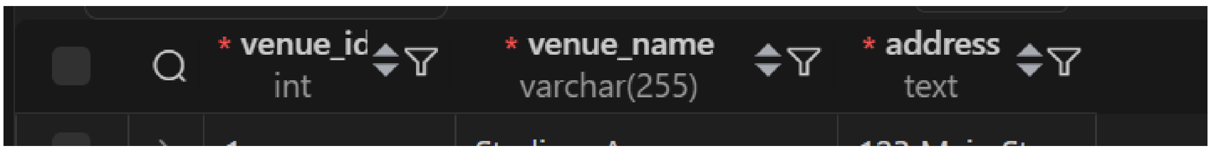
1. Create the database named "TicketBookingSystem"

```
CREATE DATABASE TicketBookingSystem;
USE TicketBookingSystem;
```

2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

- **Venu**

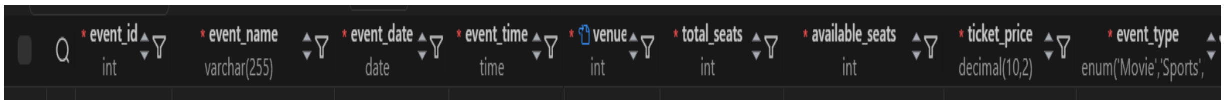
```
CREATE TABLE Venue (
    venue_id INT PRIMARY KEY AUTO_INCREMENT,
    venue_name VARCHAR(255) NOT NULL,
    address TEXT NOT NULL
);
```



* venue_id	* venue_name	* address
int	varchar(255)	text

- **Event**

```
CREATE TABLE Event (
    event_id INT PRIMARY KEY AUTO_INCREMENT,
    event_name VARCHAR(255) NOT NULL,
    event_date DATE NOT NULL,
    event_time TIME NOT NULL,
    venue_id INT NOT NULL,
    total_seats INT NOT NULL,
    available_seats INT NOT NULL,
    ticket_price DECIMAL(10,2) NOT NULL,
    event_type ENUM('Movie', 'Sports', 'Concert') NOT NULL,
    FOREIGN KEY (venue_id) REFERENCES Venue(venue_id)
);
```



* event_id	* event_name	* event_date	* event_time	* venue	* total_seats	* available_seats	* ticket_price	* event_type
int	varchar(255)	date	time	int	int	int	decimal(10,2)	enum('Movie','Sports',

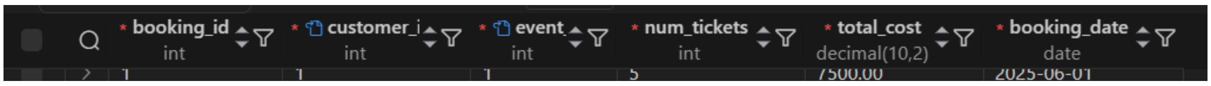
- **Customers**

```
CREATE TABLE Customer (
    customer_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    phone_number VARCHAR(20) UNIQUE NOT NULL
);
```

* customer_id int	* customer_name varchar(255)	* email varchar(255)	* phone_numbe varchar(20)

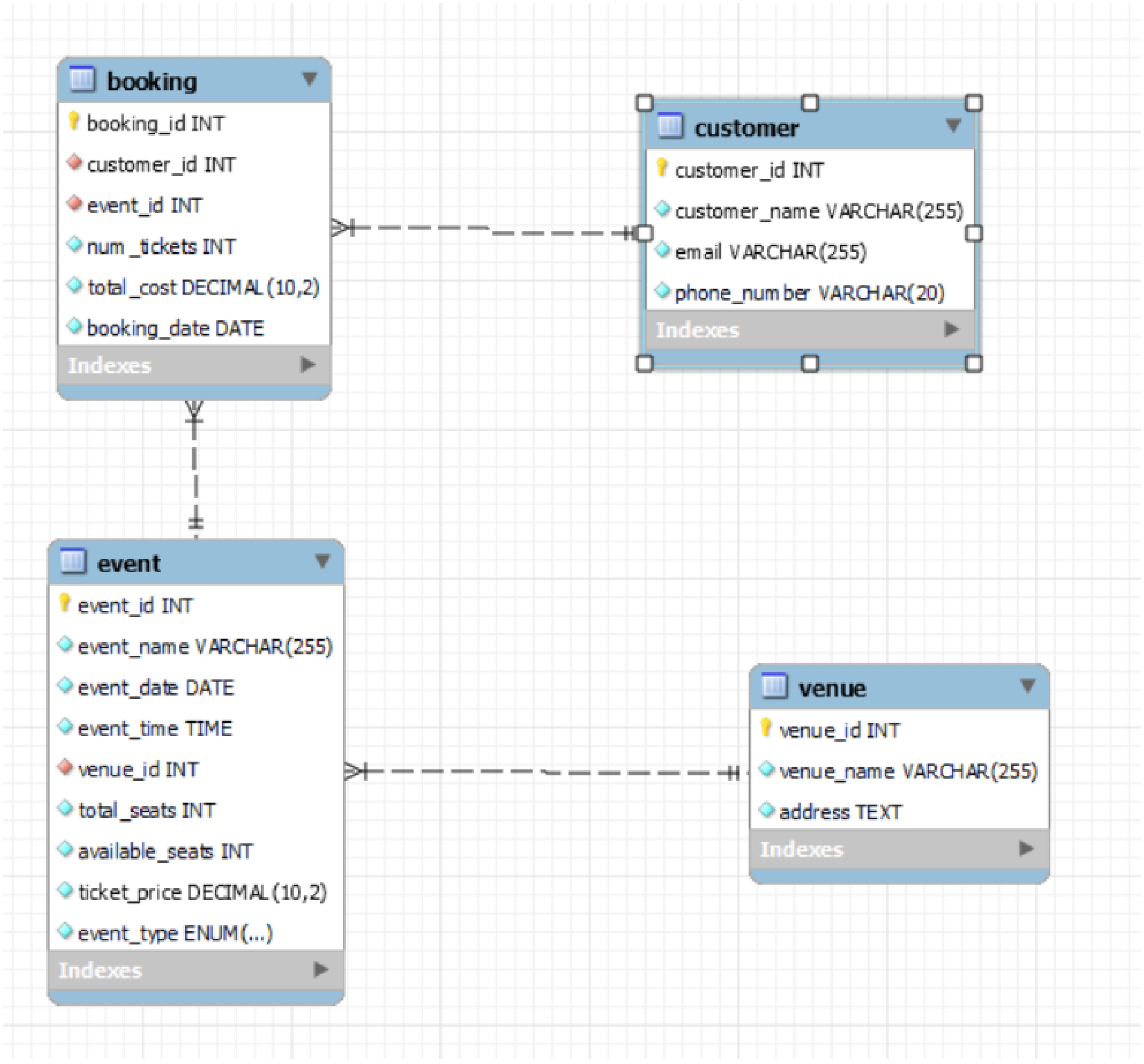
- **Booking**

```
CREATE TABLE Booking (
    booking_id INT(20) PRIMARY KEY NOT NULL AUTO_INCREMENT,
    customer_id INT NOT NULL,
    event_id INT NOT NULL,
    num_tickets INT NOT NULL,
    total_cost DECIMAL(10,2) NOT NULL,
    booking_date DATE NOT NULL ,
     FOREIGN KEY (customer_id) REFERENCES Customer(customer_id) ON
DELETE CASCADE,
     FOREIGN KEY (event_id) REFERENCES Event(event_id) ON DELETE
CASCADE
);
```



* booking_id int	* customer_i int	* event int	* num_tickets int	* total_cost decimal(10,2)	* booking_date date
1	1	1	5	7500.00	2025-06-01

3. Create an ERD (Entity Relationship Diagram) for the database



4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

All primary keys and foreign keys were properly defined in the SQL provided

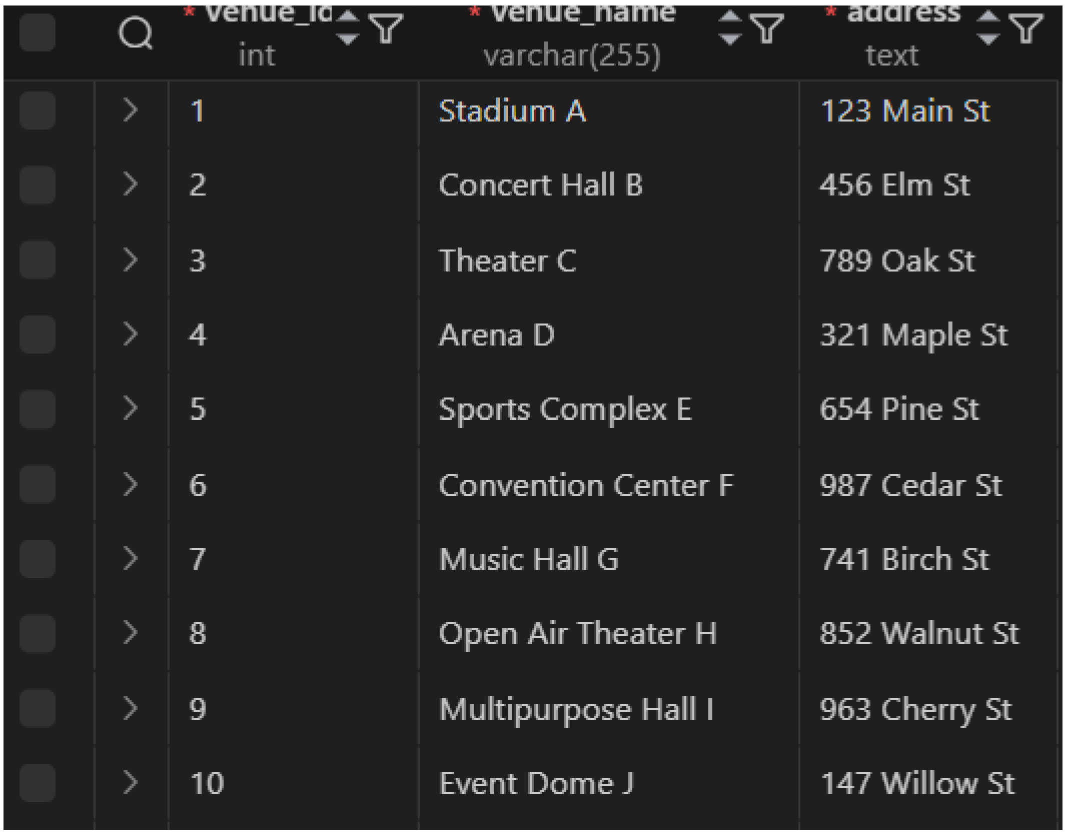
Tasks 2: Select, Where, Between, AND, LIKE:

1. Write a SQL query to insert at least 10 sample records into each table.

venue

```
INSERT INTO Venue (venue_name, address) VALUES
('Stadium A', '123 Main St'),
('Concert Hall B', '456 Elm St'),
('Theater C', '789 Oak St'),
```

('Arena D', '321 Maple St'),
('Sports Complex E', '654 Pine St'),
('Convention Center F', '987 Cedar St'),
('Music Hall G', '741 Birch St'),
('Open Air Theater H', '852 Walnut St');

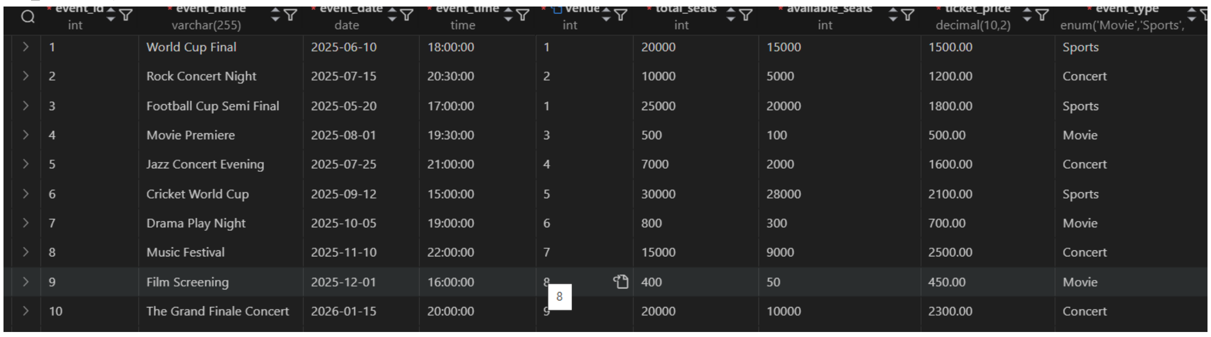


* venue_id int	* venue_name varchar(255)	* address text
1	Stadium A	123 Main St
2	Concert Hall B	456 Elm St
3	Theater C	789 Oak St
4	Arena D	321 Maple St
5	Sports Complex E	654 Pine St
6	Convention Center F	987 Cedar St
7	Music Hall G	741 Birch St
8	Open Air Theater H	852 Walnut St
9	Multipurpose Hall I	963 Cherry St
10	Event Dome J	147 Willow St

event

INSERT INTO Event (event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type) VALUES
('World Cup Final', '2025-06-10', '18:00:00', 1, 20000, 15000, 1500.00, 'Sports'),
('Rock Concert Night', '2025-07-15', '20:30:00', 2, 10000, 5000, 1200.00, 'Concert'),
('Football Cup Semi Final', '2025-05-20', '17:00:00', 1, 25000, 20000, 1800.00, 'Sports'),
('Movie Premiere', '2025-08-01', '19:30:00', 3, 500, 100, 500.00, 'Movie'),
('Jazz Concert Evening', '2025-07-25', '21:00:00', 4, 7000, 2000, 1600.00, 'Concert'),
('Cricket World Cup', '2025-09-12', '15:00:00', 5, 30000, 28000, 2100.00, 'Sports'),
('Drama Play Night', '2025-10-05', '19:00:00', 6, 800, 300, 700.00, 'Movie'),

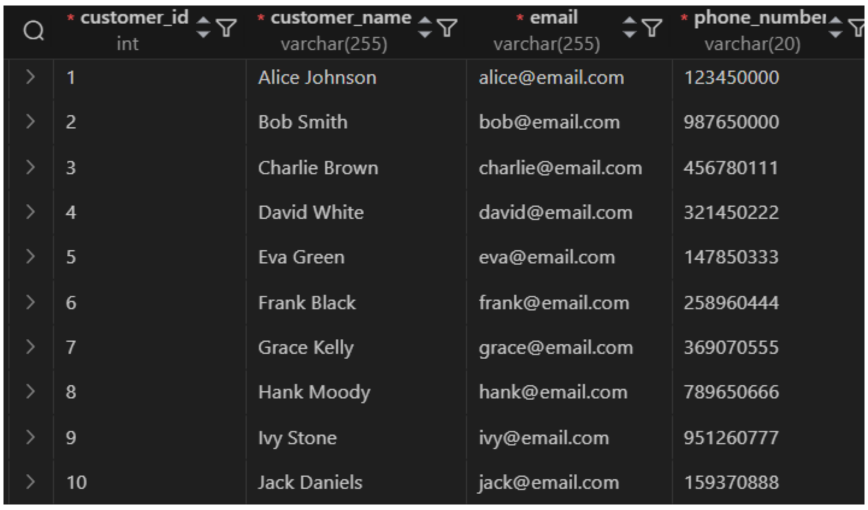
('Music Festival', '2025-11-10', '22:00:00', 7, 15000, 9000, 2500.00, 'Concert'),
('Film Screening', '2025-12-01', '16:00:00', 8, 400, 50, 450.00, 'Movie'),
('The Grand Finale Concert', '2026-01-15', '20:00:00', 9, 20000, 10000, 2300.00, 'Concert'),
('Basketball Finals', '2025-06-20', '19:00:00', 11, 15000, 12000, 1700.00, 'Sports'),



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
4	Movie Premiere	2025-08-01	19:30:00	3	500	100	500.00	Movie
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
7	Drama Play Night	2025-10-05	19:00:00	6	800	300	700.00	Movie
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert
9	Film Screening	2025-12-01	16:00:00	8	400	50	450.00	Movie
10	The Grand Finale Concert	2026-01-15	20:00:00	9	20000	10000	2300.00	Concert

customer

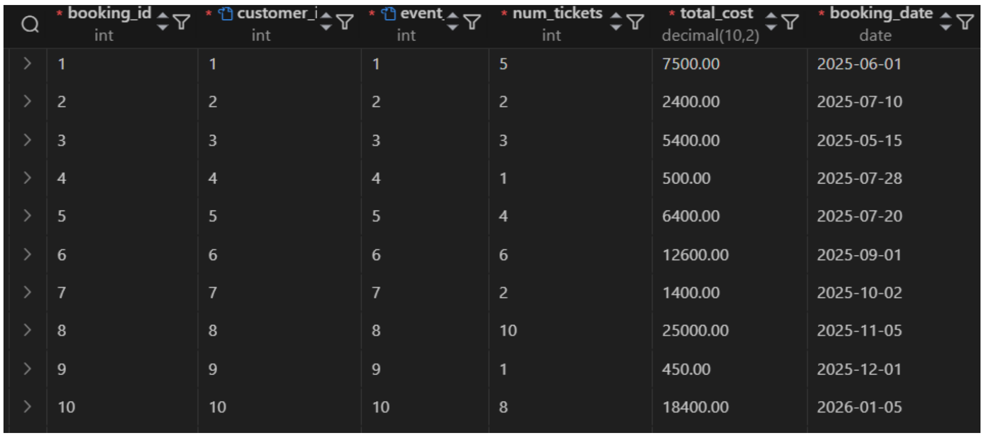
INSERT INTO Customer (customer_name, email, phone_number) VALUES
('Alice Johnson', 'alice@email.com', '123450000'),
('Bob Smith', 'bob@email.com', '987650000'),
('Charlie Brown', 'charlie@email.com', '456780111'),
('David White', 'david@email.com', '321450222'),
('Eva Green', 'eva@email.com', '147850333'),
('Frank Black', 'frank@email.com', '258960444'),
('Grace Kelly', 'grace@email.com', '369070555'),
('Hank Moody', 'hank@email.com', '789650666'),
('Ivy Stone', 'ivy@email.com', '951260777'),
('Jack Daniels', 'jack@email.com', '159370888');



customer_id int	customer_name varchar(255)	email varchar(255)	phone_number varchar(20)
1	Alice Johnson	alice@email.com	123450000
2	Bob Smith	bob@email.com	987650000
3	Charlie Brown	charlie@email.com	456780111
4	David White	david@email.com	321450222
5	Eva Green	eva@email.com	147850333
6	Frank Black	frank@email.com	258960444
7	Grace Kelly	grace@email.com	369070555
8	Hank Moody	hank@email.com	789650666
9	Ivy Stone	ivy@email.com	951260777
10	Jack Daniels	jack@email.com	159370888

booking

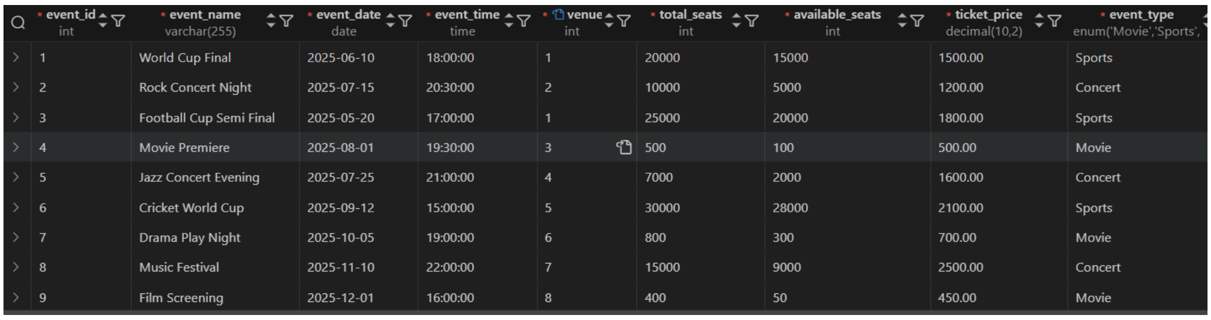
INSERT INTO Booking (customer_id, event_id, num_tickets, total_cost, booking_date) VALUES
(1, 1, 5, 7500.00, '2025-06-01'),
(2, 2, 2, 2400.00, '2025-07-10'),
(3, 3, 3, 5400.00, '2025-05-15'),
(4, 4, 1, 500.00, '2025-07-28'),
(5, 5, 4, 6400.00, '2025-07-20'),
(6, 6, 6, 12600.00, '2025-09-01'),
(7, 7, 2, 1400.00, '2025-10-02'),
(8, 8, 10, 25000.00, '2025-11-05'),
(9, 9, 1, 450.00, '2025-12-01'),
(10, 10, 8, 18400.00, '2026-01-05');



booking_id int	customer_i int	event int	num_tickets int	total_cost decimal(10,2)	booking_date date
1	1	1	5	7500.00	2025-06-01
2	2	2	2	2400.00	2025-07-10
3	3	3	3	5400.00	2025-05-15
4	4	4	1	500.00	2025-07-28
5	5	5	4	6400.00	2025-07-20
6	6	6	6	12600.00	2025-09-01
7	7	7	2	1400.00	2025-10-02
8	8	8	10	25000.00	2025-11-05
9	9	9	1	450.00	2025-12-01
10	10	10	8	18400.00	2026-01-05

2. Write a SQL query to list all Events.

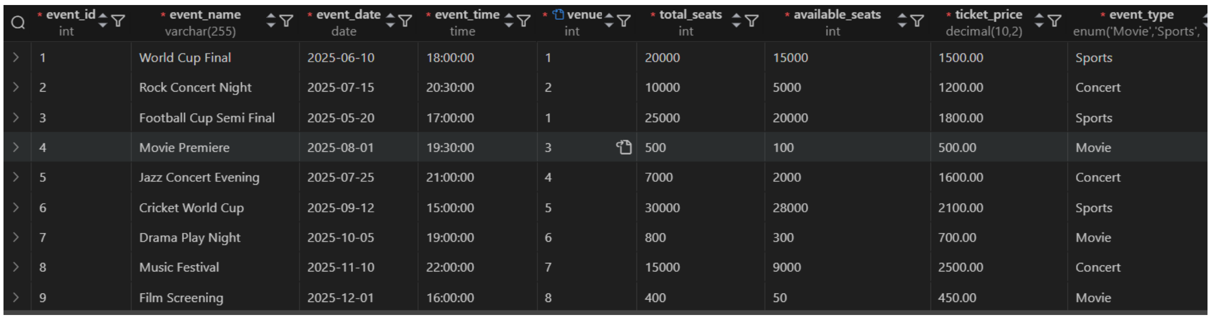
SELECT * FROM Event;



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
4	Movie Premiere	2025-08-01	19:30:00	3	500	100	500.00	Movie
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
7	Drama Play Night	2025-10-05	19:00:00	6	800	300	700.00	Movie
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert
9	Film Screening	2025-12-01	16:00:00	8	400	50	450.00	Movie

3. Write a SQL query to select events with available tickets

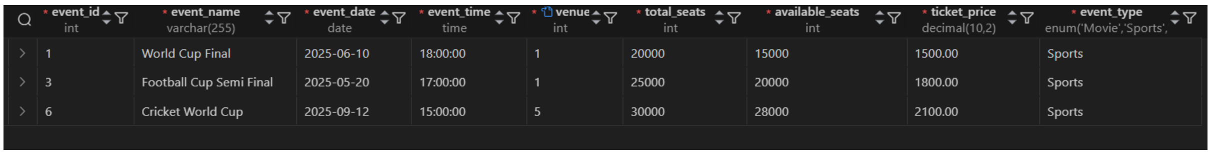
SELECT * FROM Event WHERE available_seats > 0;



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
4	Movie Premiere	2025-08-01	19:30:00	3	500	100	500.00	Movie
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
7	Drama Play Night	2025-10-05	19:00:00	6	800	300	700.00	Movie
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert
9	Film Screening	2025-12-01	16:00:00	8	400	50	450.00	Movie

4. Write a SQL query to select events name partial match with 'cup'.

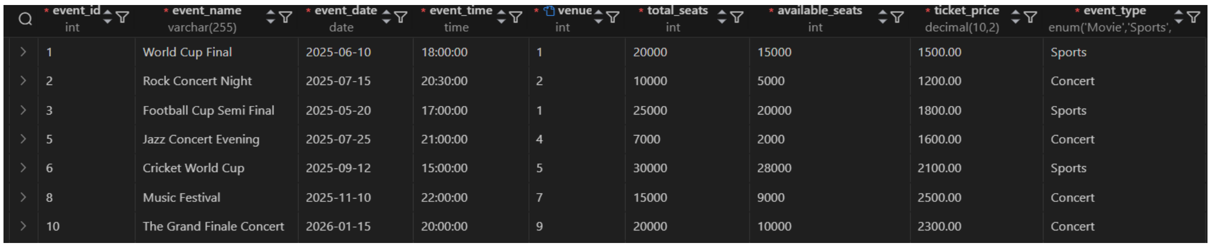
SELECT * FROM Event WHERE event_name LIKE '%cup%';



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports

5. Write a SQL query to select events with ticket price range is between 1000 to 2500.

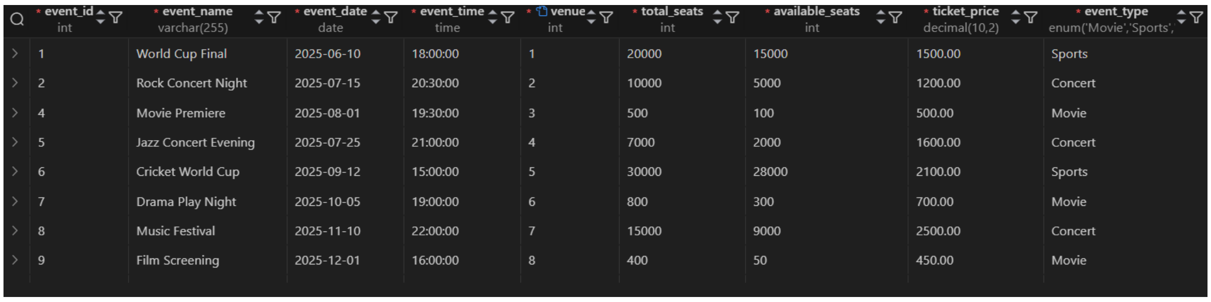
SELECT * FROM Event WHERE ticket_price BETWEEN 1000 AND 2500;



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert
10	The Grand Finale Concert	2026-01-15	20:00:00	9	20000	10000	2300.00	Concert

6. Write a SQL query to retrieve events with dates falling within a specific range.

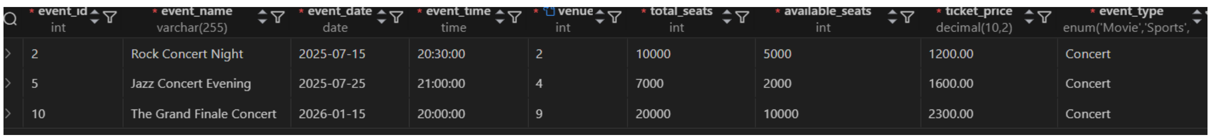
SELECT * FROM Event WHERE event_date BETWEEN '2025-06-01' AND '2025-12-31';



event_id int	event_name varchar(255)	event_date date	event_time time	venue int	total_seats int	available_seats int	ticket_price decimal(10,2)	event_type enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
4	Movie Premiere	2025-08-01	19:30:00	3	500	100	500.00	Movie
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
7	Drama Play Night	2025-10-05	19:00:00	6	800	300	700.00	Movie
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert
9	Film Screening	2025-12-01	16:00:00	8	400	50	450.00	Movie

7. Write a SQL query to retrieve events with available tickets that also have "Concert" in their name.

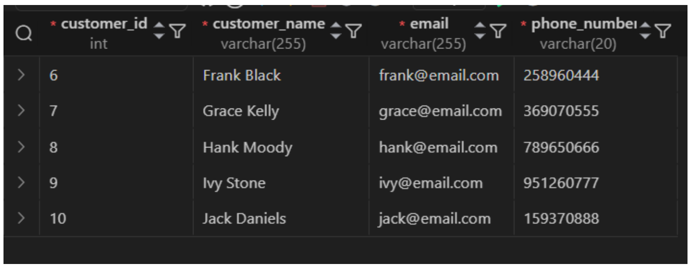
SELECT * FROM Event WHERE available_seats > 0 AND event_name LIKE '%Concert%';



* event_id int	* event_name varchar(255)	* event_date date	* event_time time	* venue int	* total_seats int	* available_seats int	* ticket_price decimal(10,2)	* event_type enum('Movie','Sports',
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
10	The Grand Finale Concert	2026-01-15	20:00:00	9	20000	10000	2300.00	Concert

8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.

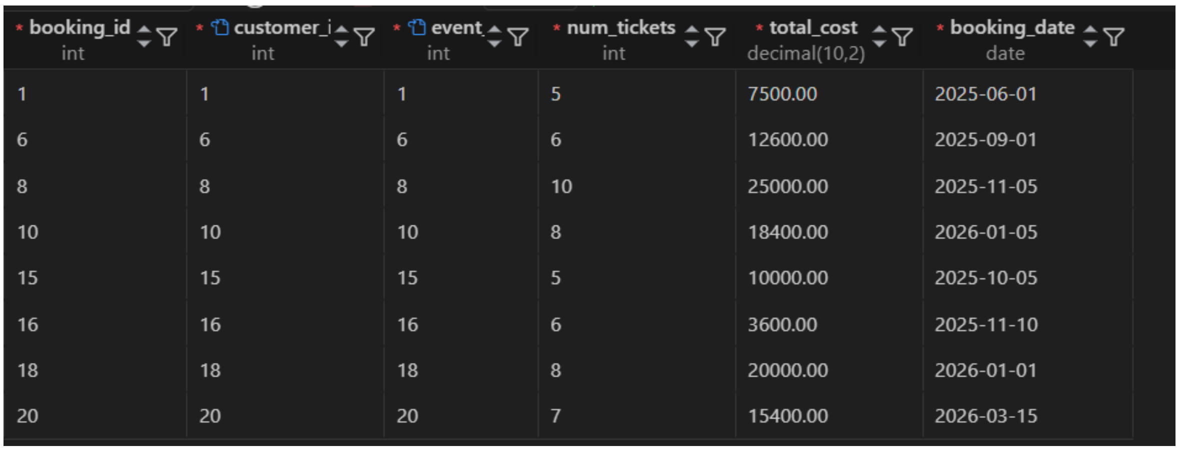
SELECT * FROM Customer LIMIT 5 OFFSET 5;



* customer_id int	* customer_name varchar(255)	* email varchar(255)	* phone_number varchar(20)
6	Frank Black	frank@email.com	258960444
7	Grace Kelly	grace@email.com	369070555
8	Hank Moody	hank@email.com	789650666
9	Ivy Stone	ivy@email.com	951260777
10	Jack Daniels	jack@email.com	159370888

9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.

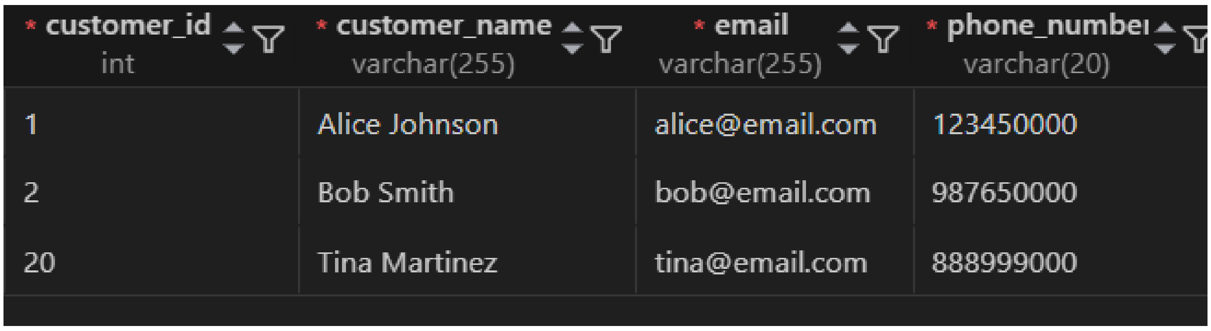
SELECT * FROM Booking WHERE num_tickets > 4;



* booking_id int	* customer_i int	* event int	* num_tickets int	* total_cost decimal(10,2)	* booking_date date
1	1	1	5	7500.00	2025-06-01
6	6	6	6	12600.00	2025-09-01
8	8	8	10	25000.00	2025-11-05
10	10	10	8	18400.00	2026-01-05
15	15	15	5	10000.00	2025-10-05
16	16	16	6	3600.00	2025-11-10
18	18	18	8	20000.00	2026-01-01
20	20	20	7	15400.00	2026-03-15

10. Write a SQL query to retrieve customer information whose phone number end with '000'

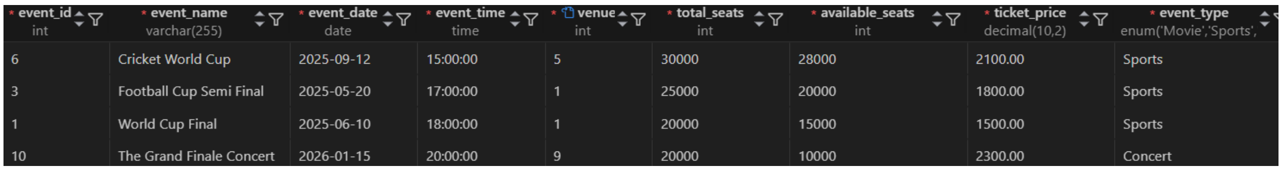
SELECT * FROM Customer WHERE phone_number LIKE '%000';



* customer_id int	* customer_name varchar(255)	* email varchar(255)	* phone_number varchar(20)
1	Alice Johnson	alice@email.com	123450000
2	Bob Smith	bob@email.com	987650000
20	Tina Martinez	tina@email.com	888999000

11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.

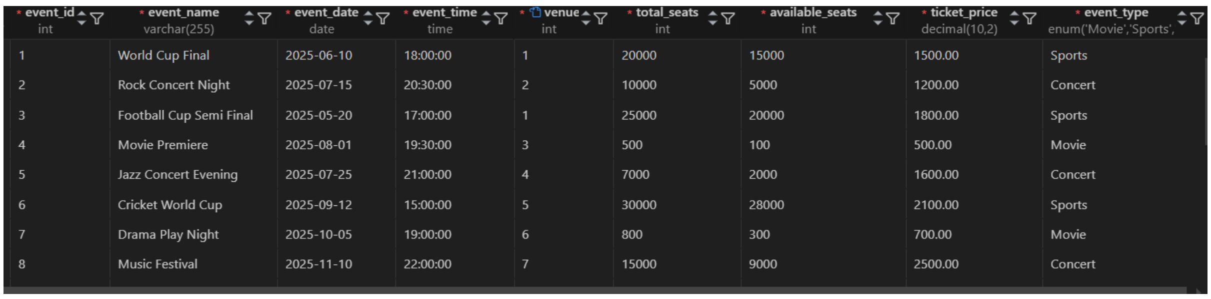
SELECT * FROM Event WHERE total_seats > 15000 ORDER BY total_seats DESC;



event_id	event_name	event_date	event_time	venue	total_seats	available_seats	ticket_price	event_type
int	varchar(255)	date	time	int	int	int	decimal(10,2)	enum('Movie','Sports',
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
10	The Grand Finale Concert	2026-01-15	20:00:00	9	20000	10000	2300.00	Concert

12. Write a SQL query to select events name not start with ‘x’, ‘y’, ‘z’

SELECT * FROM Event WHERE event_name NOT LIKE 'X%'
AND event_name NOT LIKE 'Y%'
AND event_name NOT LIKE 'Z%';

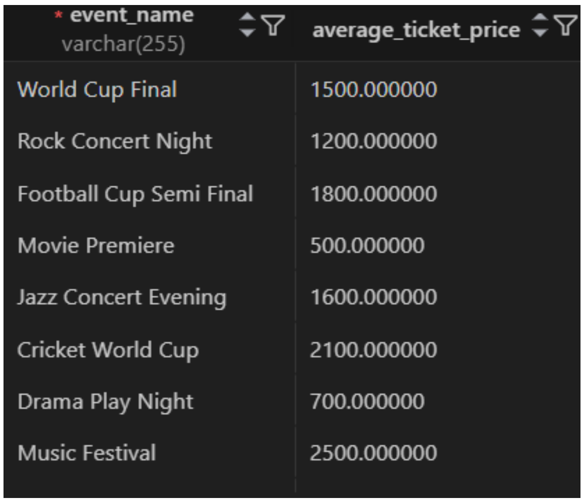


event_id	event_name	event_date	event_time	venue	total_seats	available_seats	ticket_price	event_type
int	varchar(255)	date	time	int	int	int	decimal(10,2)	enum('Movie','Sports',
1	World Cup Final	2025-06-10	18:00:00	1	20000	15000	1500.00	Sports
2	Rock Concert Night	2025-07-15	20:30:00	2	10000	5000	1200.00	Concert
3	Football Cup Semi Final	2025-05-20	17:00:00	1	25000	20000	1800.00	Sports
4	Movie Premiere	2025-08-01	19:30:00	3	500	100	500.00	Movie
5	Jazz Concert Evening	2025-07-25	21:00:00	4	7000	2000	1600.00	Concert
6	Cricket World Cup	2025-09-12	15:00:00	5	30000	28000	2100.00	Sports
7	Drama Play Night	2025-10-05	19:00:00	6	800	300	700.00	Movie
8	Music Festival	2025-11-10	22:00:00	7	15000	9000	2500.00	Concert

Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

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

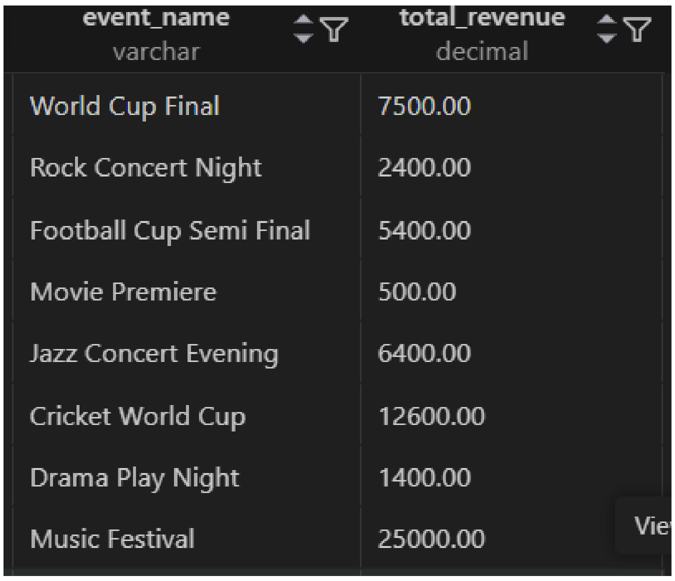
SELECT event_name, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY event_name;



* event_name varchar(255)	average_ticket_price
World Cup Final	1500.000000
Rock Concert Night	1200.000000
Football Cup Semi Final	1800.000000
Movie Premiere	500.000000
Jazz Concert Evening	1600.000000
Cricket World Cup	2100.000000
Drama Play Night	700.000000
Music Festival	2500.000000

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

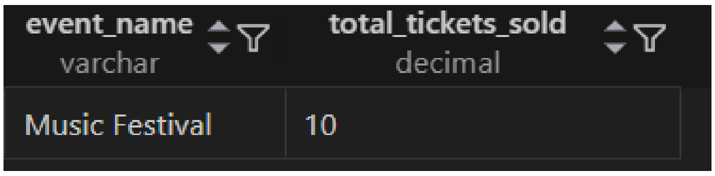
```
SELECT e.event_name, SUM(b.total_cost) AS total_revenue
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY e.event_name;
```



event_name varchar	total_revenue decimal
World Cup Final	7500.00
Rock Concert Night	2400.00
Football Cup Semi Final	5400.00
Movie Premiere	500.00
Jazz Concert Evening	6400.00
Cricket World Cup	12600.00
Drama Play Night	1400.00
Music Festival	25000.00

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

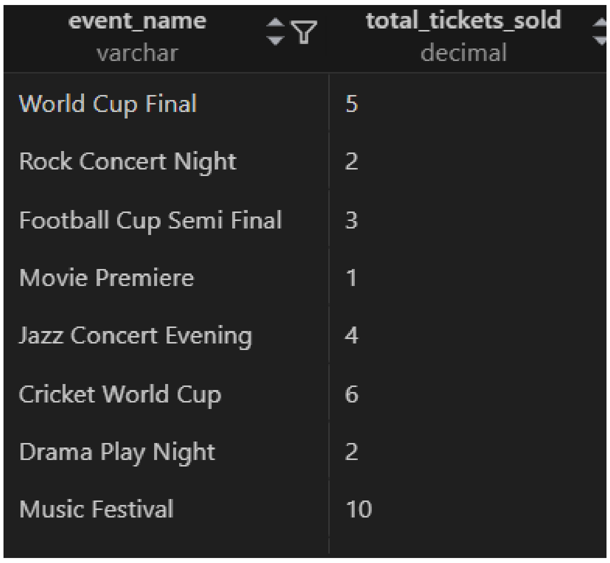
```
SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY e.event_name
ORDER BY total_tickets_sold DESC
LIMIT 1;
```



event_name varchar	total_tickets_sold decimal
Music Festival	10

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

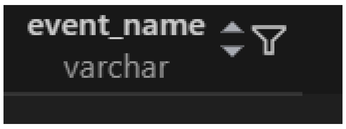
```
SELECT e.event_name, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY e.event_name;
```



event_name varchar	total_tickets_sold decimal
World Cup Final	5
Rock Concert Night	2
Football Cup Semi Final	3
Movie Premiere	1
Jazz Concert Evening	4
Cricket World Cup	6
Drama Play Night	2
Music Festival	10

5. Write a SQL query to Find Events with No Ticket Sales.

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

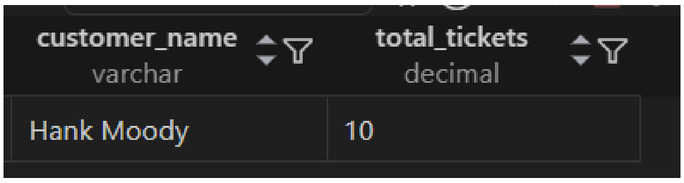


event_name varchar

6. Write a 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
JOIN Customer c ON b.customer_id = c.customer_id
GROUP BY c.customer_name
ORDER BY total_tickets DESC
```

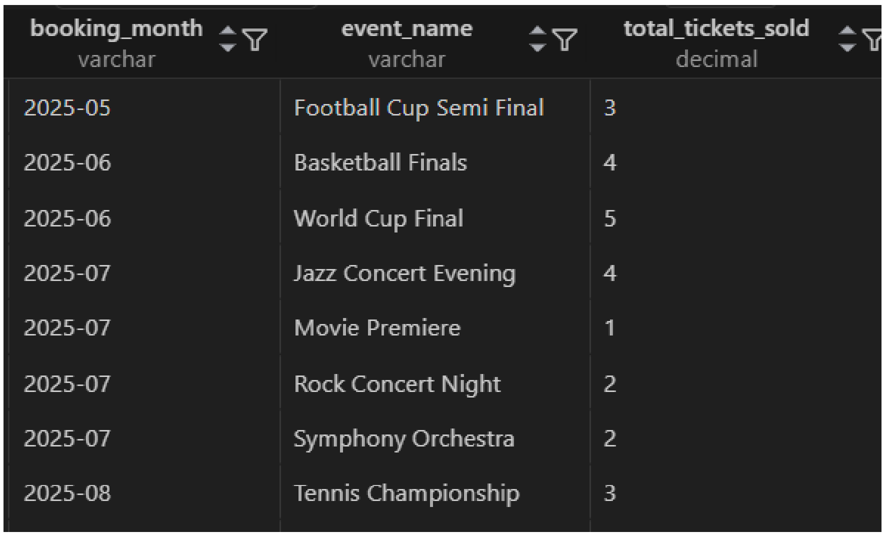
```
LIMIT 1;
```



customer_name varchar	total_tickets decimal
Hank Moody	10

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

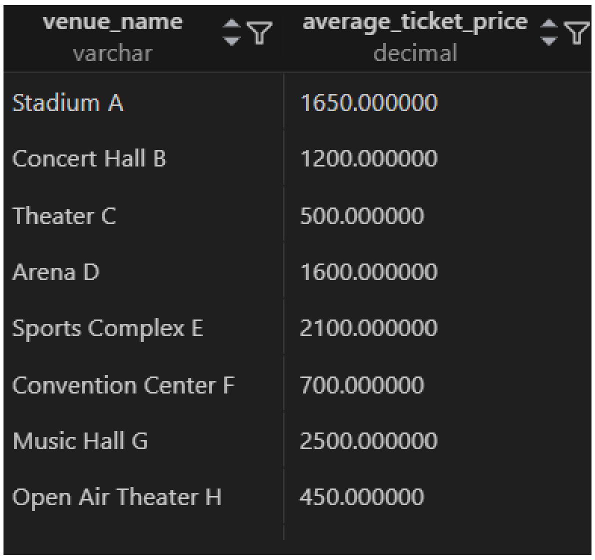
```
SELECT
   DATE_FORMAT(b.booking_date, '%Y-%m') AS booking_month,
   e.event_name,
   SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY booking_month, e.event_name
ORDER BY booking_month ASC;
```



booking_month varchar	event_name varchar	total_tickets_sold decimal
2025-05	Football Cup Semi Final	3
2025-06	Basketball Finals	4
2025-06	World Cup Final	5
2025-07	Jazz Concert Evening	4
2025-07	Movie Premiere	1
2025-07	Rock Concert Night	2
2025-07	Symphony Orchestra	2
2025-08	Tennis Championship	3

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

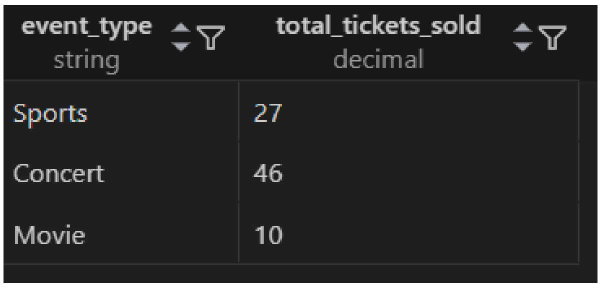
```
SELECT v.venue_name, AVG(e.ticket_price) AS average_ticket_price
FROM Event e
JOIN Venue v ON e.venue_id = v.venue_id
GROUP BY v.venue_name;
```



venue_name varchar	average_ticket_price decimal
Stadium A	1650.000000
Concert Hall B	1200.000000
Theater C	500.000000
Arena D	1600.000000
Sports Complex E	2100.000000
Convention Center F	700.000000
Music Hall G	2500.000000
Open Air Theater H	450.000000

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

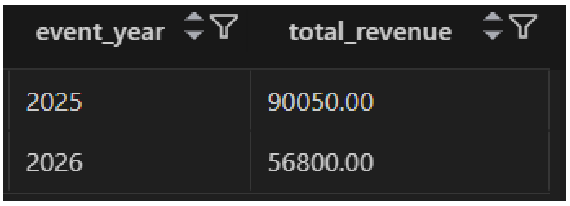
```
SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
GROUP BY e.event_type;
```



event_type string	total_tickets_sold decimal
Sports	27
Concert	46
Movie	10

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

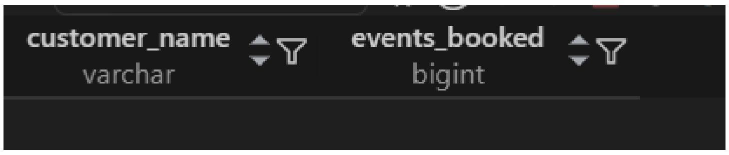
```
SELECT YEAR(b.booking_date) AS event_year, SUM(b.total_cost) AS
total_revenue
FROM Booking b
GROUP BY event_year
ORDER BY event_year;
```



event_year	total_revenue
2025	90050.00
2026	56800.00

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

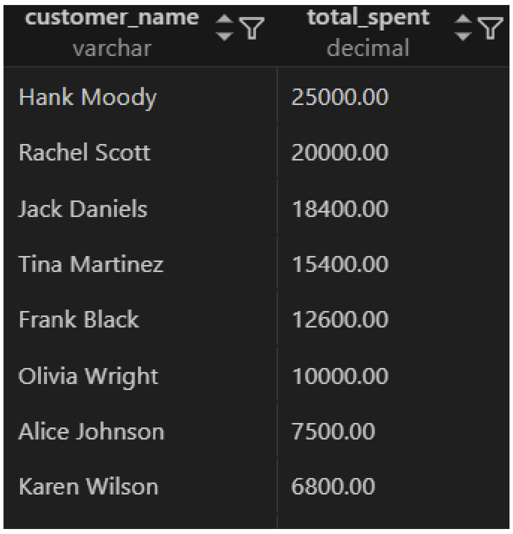
SELECT c.customer_name, COUNT(DISTINCT b.event_id) AS events_booked
FROM Booking b
JOIN Customer c ON b.customer_id = c.customer_id
GROUP BY c.customer_name
HAVING events_booked > 1;



customer_name varchar	events_booked bigint

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

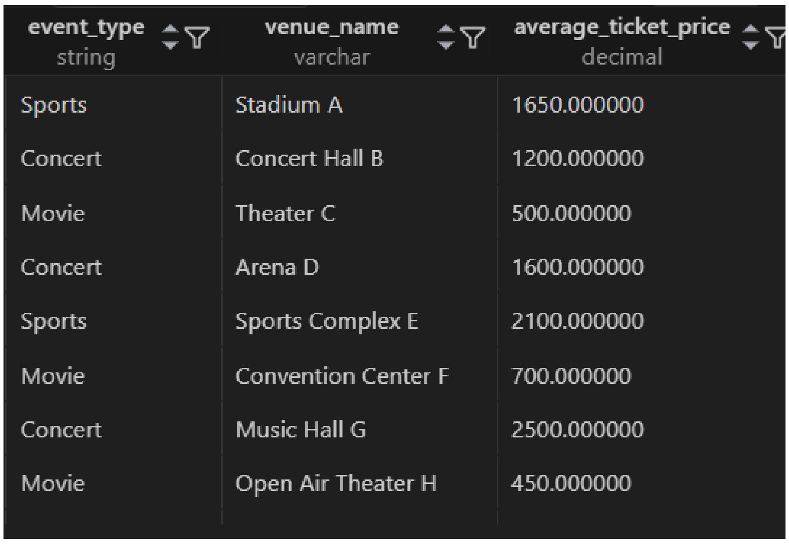
SELECT c.customer_name, SUM(b.total_cost) AS total_spent
FROM Booking b
JOIN Customer c ON b.customer_id = c.customer_id
GROUP BY c.customer_name
ORDER BY total_spent DESC;



customer_name varchar	total_spent decimal
Hank Moody	25000.00
Rachel Scott	20000.00
Jack Daniels	18400.00
Tina Martinez	15400.00
Frank Black	12600.00
Olivia Wright	10000.00
Alice Johnson	7500.00
Karen Wilson	6800.00

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

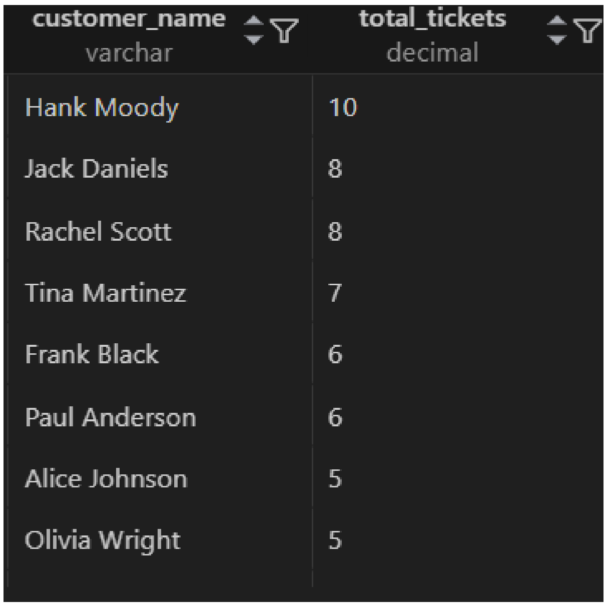
SELECT e.event_type, v.venue_name, AVG(e.ticket_price) AS average_ticket_price
FROM Event e
JOIN Venue v ON e.venue_id = v.venue_id
GROUP BY e.event_type, v.venue_name;



event_type string	venue_name varchar	average_ticket_price decimal
Sports	Stadium A	1650.000000
Concert	Concert Hall B	1200.000000
Movie	Theater C	500.000000
Concert	Arena D	1600.000000
Sports	Sports Complex E	2100.000000
Movie	Convention Center F	700.000000
Concert	Music Hall G	2500.000000
Movie	Open Air Theater H	450.000000

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

```
SELECT c.customer_name, SUM(b.num_tickets) AS total_tickets
FROM Booking b
JOIN Customer c ON b.customer_id = c.customer_id
WHERE b.booking_date >= CURDATE() - INTERVAL 30 DAY
GROUP BY c.customer_name
ORDER BY total_tickets DESC;
```

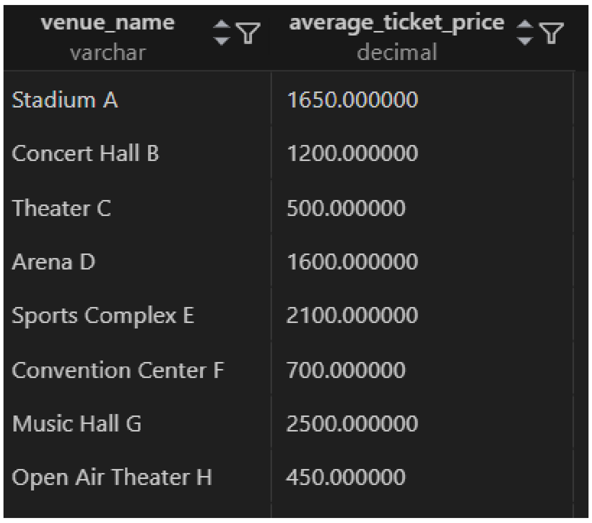


customer_name varchar	total_tickets decimal
Hank Moody	10
Jack Daniels	8
Rachel Scott	8
Tina Martinez	7
Frank Black	6
Paul Anderson	6
Alice Johnson	5
Olivia Wright	5

Tasks 4: Subquery and its types

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

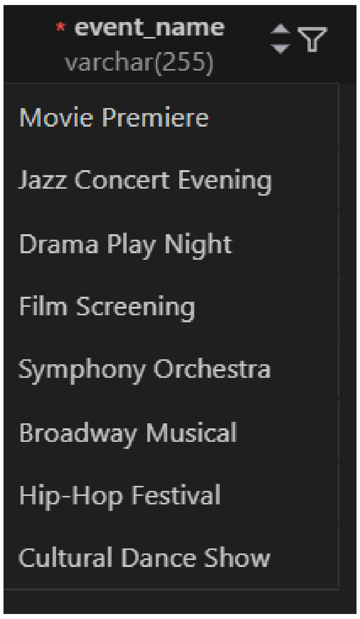
```
SELECT venue_name,
    (SELECT AVG(ticket_price) FROM Event e WHERE e.venue_id = v.venue_id) AS average_ticket_price
FROM Venue v;
```



venue_name varchar	average_ticket_price decimal
Stadium A	1650.000000
Concert Hall B	1200.000000
Theater C	500.000000
Arena D	1600.000000
Sports Complex E	2100.000000
Convention Center F	700.000000
Music Hall G	2500.000000
Open Air Theater H	450.000000

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

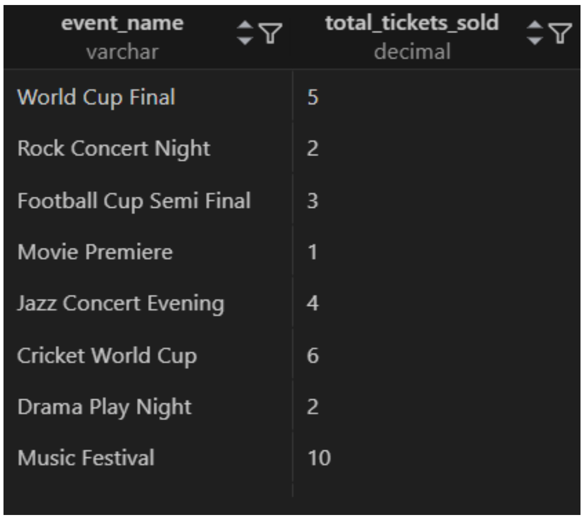
```
SELECT event_name
FROM Event
WHERE available_seats < (total_seats / 2);
```



* event_name varchar(255)
Movie Premiere
Jazz Concert Evening
Drama Play Night
Film Screening
Symphony Orchestra
Broadway Musical
Hip-Hop Festival
Cultural Dance Show

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

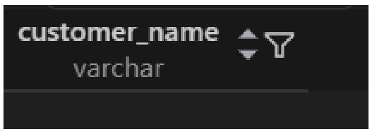
```
SELECT event_name,
    (SELECT SUM(num_tickets) FROM Booking b WHERE b.event_id = e.event_id) AS total_tickets_sold
FROM Event e;
```



event_name varchar	total_tickets_sold decimal
World Cup Final	5
Rock Concert Night	2
Football Cup Semi Final	3
Movie Premiere	1
Jazz Concert Evening	4
Cricket World Cup	6
Drama Play Night	2
Music Festival	10

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

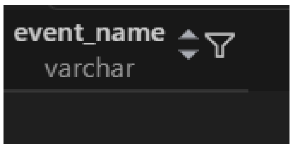
```
SELECT customer_name
FROM Customer c
WHERE NOT EXISTS (SELECT 1 FROM Booking b WHERE b.customer_id
= c.customer_id);
```



```
customer_name
varchar
```

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

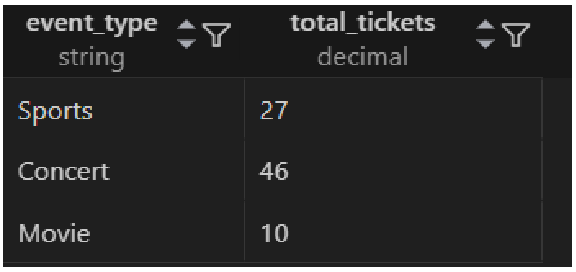
```
SELECT event_name
FROM Event
WHERE event_id NOT IN (SELECT DISTINCT event_id FROM Booking);
```



```
event_name
varchar
```

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

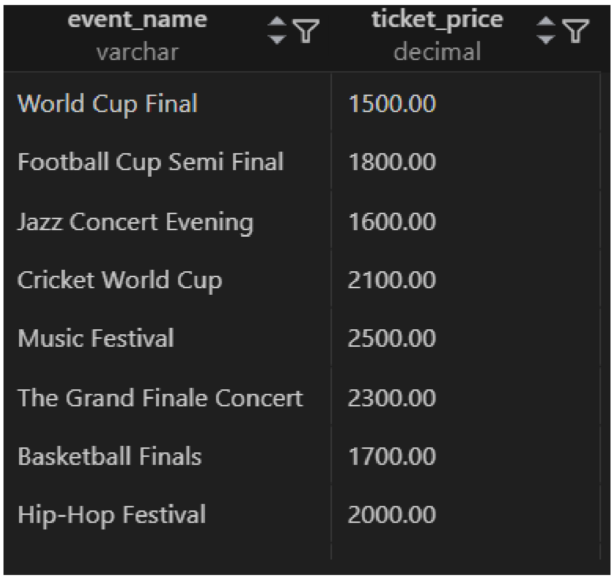
```
SELECT event_type, SUM(total_tickets_sold) AS total_tickets
FROM (
    SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold
    FROM Booking b
    JOIN Event e ON b.event_id = e.event_id
    GROUP BY e.event_type
) AS ticket_counts
GROUP BY event_type;
```



event_type string	total_tickets decimal
Sports	27
Concert	46
Movie	10

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

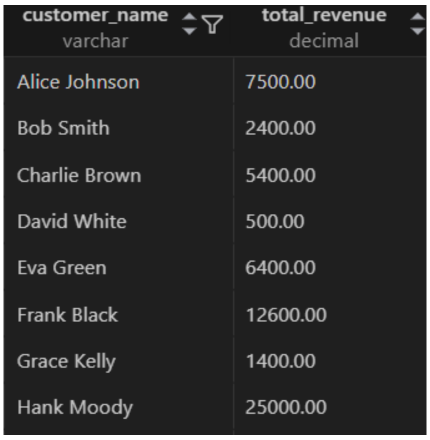
```
SELECT event_name, ticket_price
FROM Event
WHERE ticket_price > (SELECT AVG(ticket_price) FROM Event);
```



event_name varchar	ticket_price decimal
World Cup Final	1500.00
Football Cup Semi Final	1800.00
Jazz Concert Evening	1600.00
Cricket World Cup	2100.00
Music Festival	2500.00
The Grand Finale Concert	2300.00
Basketball Finals	1700.00
Hip-Hop Festival	2000.00

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

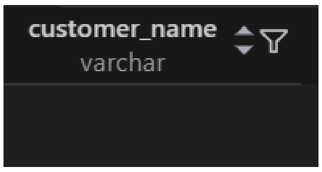
```
SELECT c.customer_name,
       (SELECT SUM(b.total_cost) FROM Booking b WHERE b.customer_id =
c.customer_id) AS total_revenue
FROM Customer c;
```



customer_name varchar	total_revenue decimal
Alice Johnson	7500.00
Bob Smith	2400.00
Charlie Brown	5400.00
David White	500.00
Eva Green	6400.00
Frank Black	12600.00
Grace Kelly	1400.00
Hank Moody	25000.00

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

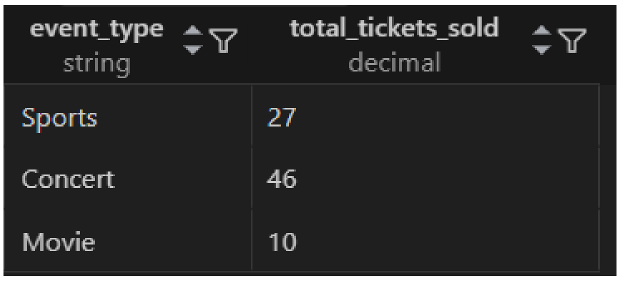
```
SELECT customer_name
FROM Customer
WHERE customer_id IN (
    SELECT DISTINCT b.customer_id
    FROM Booking b
    JOIN Event e ON b.event_id = e.event_id
        WHERE e.venue_id = (SELECT venue_id FROM Venue WHERE
venue_name = 'Some Venue Name')
);
```



customer_name varchar

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

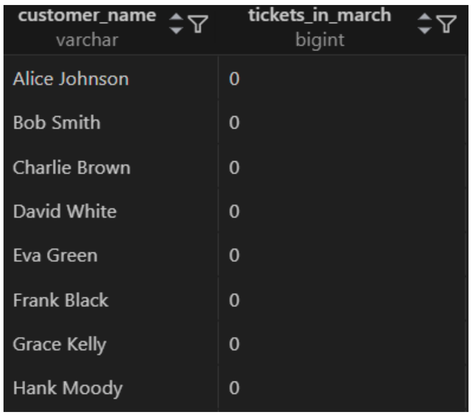
```
SELECT e.event_type,
       COALESCE(SUM(b.num_tickets), 0) AS total_tickets_sold
FROM Event e
LEFT JOIN Booking b ON e.event_id = b.event_id
GROUP BY e.event_type;
```



event_type string	total_tickets_sold decimal
Sports	27
Concert	46
Movie	10

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

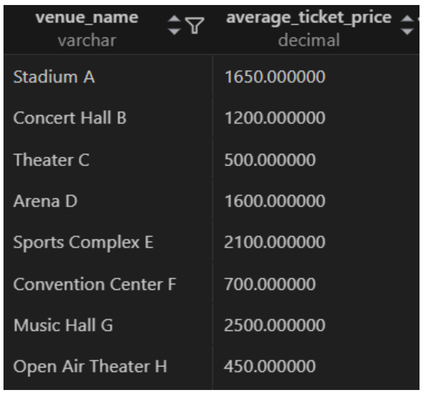
```
SELECT customer_name,
       (SELECT COUNT(*) FROM Booking b WHERE b.customer_id = c.customer_id AND
         DATE_FORMAT(b.booking_date, '%Y-%m') = '2025-03') AS tickets_in_march
FROM Customer c;
```



customer_name varchar	tickets_in_march bigint
Alice Johnson	0
Bob Smith	0
Charlie Brown	0
David White	0
Eva Green	0
Frank Black	0
Grace Kelly	0
Hank Moody	0

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

```
SELECT venue_name, (SELECT AVG(ticket_price) FROM Event e WHERE e.venue_id = v.venue_id) AS average_ticket_price
FROM Venue v;
```



venue_name varchar	average_ticket_price decimal
Stadium A	1650.000000
Concert Hall B	1200.000000
Theater C	500.000000
Arena D	1600.000000
Sports Complex E	2100.000000
Convention Center F	700.000000
Music Hall G	2500.000000
Open Air Theater H	450.000000