Ticket Booking System

Tasks 1: Creating Database:

```
CREATE DATABASE TicketBookingSystem;

USE TicketBookingSystem;

CREATE TABLE Venu (
   venue_id INT(5) PRIMARY KEY,
   venue_name VARCHAR(50),
   address VARCHAR(100)
);

CREATE TABLE Event (
   event_id INT(5) PRIMARY KEY,
   event_name VARCHAR(50),
   event_date DATE,
   event_time TIME,
   venue_id INT(5),
   total_seats INT(5),
   available_seats INT(3),
   ticket_price DECIMAL(10,2),
   event_type ENUM('Movie', 'Sports', 'Concert'),
   booking_id INT(5)
);

CREATE TABLE Customer (
   customer_id INT(5) PRIMARY KEY,
   customer_name VARCHAR(50),
   email VARCHAR(100),
   phone_number VARCHAR(20),
   booking_id INT(5)
);

CREATE TABLE Booking (
   booking_id INT(5) PRIMARY KEY,
   customer_id INT,
   event_id INT(6),
   num_tickets INT(10),
   total_cost DECIMAL(20,2),
   booking_date DATE
);


ALTER TABLE Event
   ADD FOREIGN KEY (venue_id) REFERENCES Venu(venue_id),
   ADD FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);

ALTER TABLE Customer
   ADD FOREIGN KEY (booking_id) REFERENCES Booking(booking_id);

ALTER TABLE Booking
   ADD FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
   ADD FOREIGN KEY (event_id) REFERENCES Event(event_id);
```

Output:

✔	68	16:27:01	CREATE DATABASE TicketBookingSystem	1 row(s) affected	0.015 sec
✔	69	16:27:01	USE TicketBookingSystem	0 row(s) affected	0.000 sec
⚠	70	16:27:01	CREATE TABLE Venu (venue_id INT(5) PRIMARY KEY, venue_name VARCHAR(50), address VARCHAR(100))	0 row(s) affected, 1 warning(s): 1681 Integer display width is deprecated and will be removed in a future release.	0.000 sec
⚠	71	16:27:01	CREATE TABLE Event (event_id INT(5) PRIMARY KEY, event_name VARCHAR(50), event_date DATE, event_time TIME, venue_id IN...	0 row(s) affected, 5 warning(s): 1681 Integer display width is deprecated and will be removed in a future release. 1681 Integer display width is deprecate...	0.016 sec
⚠	72	16:27:01	CREATE TABLE Customer (customer_id INT(5) PRIMARY KEY, customer_name VARCHAR(50), email VARCHAR(100), phone_number VA...	0 row(s) affected, 2 warning(s): 1681 Integer display width is deprecated and will be removed in a future release. 1681 Integer display width is deprecate...	0.015 sec
⚠	73	16:27:01	CREATE TABLE Booking (booking_id INT(5) PRIMARY KEY, customer_id INT, event_id INT(6), num_tickets INT(10), total_cost DECIMA...	0 row(s) affected, 3 warning(s): 1681 Integer display width is deprecated and will be removed in a future release. 1681 Integer display width is deprecate...	0.000 sec
✔	74	16:27:01	ALTER TABLE Event ADD FOREIGN KEY (venue_id) REFERENCES Venu(venue_id), ADD FOREIGN KEY (booking_id) REFERENCES Bookin...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
✔	75	16:27:01	ALTER TABLE Customer ADD FOREIGN KEY (booking_id) REFERENCES Booking(booking_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
✔	76	16:27:01	ALTER TABLE Booking ADD FOREIGN KEY (customer_id) REFERENCES Customer(customer_id), ADD FOREIGN KEY (event_id) REFERENCE...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec

Inserting Data to the Database:

```
-- Insert data into Venu table
INSERT INTO Venu (venue_id, venue_name, address) VALUES
(1, 'Wembley Stadium', 'London, UK'),
(2, 'Madison Square Garden', 'New York, USA'),
(3, 'Sydney Opera House', 'Sydney, Australia'),
(4, 'Staples Center', 'Los Angeles, USA'),
(5, 'Yankee Stadium', 'New York, USA'),
(6, 'The O2 Arena', 'London, UK'),
(7, 'Saitama Super Arena', 'Saitama, Japan'),
(8, 'Estadio Azteca', 'Mexico City, Mexico'),
(9, 'Allianz Arena', 'Munich, Germany'),
(10, 'Santiago Bernabeu Stadium', 'Madrid, Spain'),
(11, 'Maracanã Stadium', 'Rio de Janeiro, Brazil'),
(12, 'Rose Bowl Stadium', 'Los Angeles, USA'),
(13, 'Melbourne Cricket Ground', 'Melbourne, Australia'),
(14, 'Camp Nou', 'Barcelona, Spain'),
(15, 'AT&T Stadium', 'Dallas, USA'),
(16, 'National Stadium', 'Beijing, China'),
(17, 'Principality Stadium', 'Cardiff, UK'),
(18, 'Rogers Centre', 'Toronto, Canada'),
(19, 'Mercedes-Benz Stadium', 'Atlanta, USA'),
(20, 'Gelora Bung Karno Stadium', 'Jakarta, Indonesia'),
(21, 'Olympiastadion', 'Berlin, Germany'),
(22, 'FNB Stadium', 'Johannesburg, South Africa'),
(23, 'Nissan Stadium', 'Nashville, USA'),
(24, 'MCG', 'Melbourne, Australia'),
(25, 'Stade de France', 'Saint-Denis, France'),
(26, 'Bukit Jalil National Stadium', 'Kuala Lumpur, Malaysia'),
(27, 'Twickenham Stadium', 'London, UK'),
(28, 'King Fahd Stadium', 'Riyadh, Saudi Arabia'),
(29, 'AT&T Park', 'San Francisco, USA'),
(30, 'Singapore National Stadium', 'Singapore'),
(31, 'Optus Stadium', 'Perth, Australia'),
(32, 'Lambeau Field', 'Green Bay, USA'),
(33, 'Suncorp Stadium', 'Brisbane, Australia'),
(34, 'Estadio Monumental', 'Lima, Peru'),
(35, 'Fisht Olympic Stadium', 'Sochi, Russia'),
(36, 'Stamford Bridge', 'London, UK'),
(37, 'Croke Park', 'Dublin, Ireland'),
(38, 'Aviva Stadium', 'Dublin, Ireland'),
(39, 'Veltins-Arena', 'Gelsenkirchen, Germany'),
(40, 'Estadio BBVA', 'Monterrey, Mexico');

-- Insert data into Event table
INSERT INTO Event (event_id, event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price,
event_type, booking_id) VALUES
(1, 'UEFA Champions League Final', '2024-05-29', '20:00:00', 1, 90000, 85000, 200.00, 'Sports', NULL),
(2, 'Super Bowl', '2024-02-04', '18:30:00', 15, 70000, 68000, 500.00, 'Sports', NULL),
(3, 'Coachella Music Festival', '2024-04-12', '12:00:00', 2, 125000, 120000, 350.00, 'Concert', NULL),
(4, 'Wimbledon Men''s Final', '2024-07-14', '14:00:00', 17, 15000, 14000, 150.00, 'Sports', NULL),
(5, 'Academy Awards', '2024-02-25', '17:00:00', 5, 3000, 2800, 1000.00, 'Concert', NULL),
(6, 'World Cup Final', '2024-12-15', '20:00:00', 11, 100000, 95000, 300.00, 'Sports', NULL),
(7, 'Glastonbury Festival', '2024-06-28', '12:00:00', 1, 135000, 130000, 250.00, 'Concert', NULL),
```

```
(8, 'US Open Tennis Men''s Final', '2024-09-08', '16:00:00', 22, 22000, 20000, 180.00, 'Sports', NULL),
(9, 'Comic-Con International', '2024-07-18', '10:00:00', 29, 150000, 148000, 120.00, 'Concert', NULL),
(10, 'Summer Olympics Opening Ceremony', '2024-07-19', '20:00:00', 30, 80000, 78000, 400.00, 'Sports', NULL),
(11, 'Australian Open Women''s Final', '2024-01-27', '15:00:00', 31, 18000, 17000, 200.00, 'Sports', NULL),
(12, 'Bonnaroo Music Festival', '2024-06-13', '12:00:00', 3, 80000, 75000, 150.00, 'Concert', NULL),
(13, 'Tour de France Final Stage', '2024-07-21', '14:00:00', 21, 10000, 9500, 80.00, 'Sports', NULL),
(14, 'EDC Las Vegas', '2024-05-17', '18:00:00', 13, 160000, 155000, 300.00, 'Concert', NULL),
(15, 'FIFA World Cup Final', '2024-12-01', '20:00:00', 11, 100000, 95000, 300.00, 'Sports', NULL),
(16, 'San Diego Comic-Con', '2024-07-20', '09:00:00', 29, 150000, 148000, 120.00, 'Concert', NULL),
(17, 'Indian Premier League Final', '2024-05-12', '19:00:00', 12, 50000, 48000, 250.00, 'Sports', NULL),
(18, 'Eurovision Song Contest Final', '2024-05-18', '21:00:00', 20, 15000, 14000, 200.00, 'Concert', NULL),
(19, 'NBA Finals Game 7', '2024-06-20', '20:30:00', 4, 21000, 20000, 350.00, 'Sports', NULL),
(20, 'Burning Man Festival', '2024-08-25', '12:00:00', 32, 80000, 75000, 300.00, 'Concert', NULL);

-- Insert data into Customer table
INSERT INTO Customer (customer_id, customer_name, email, phone_number, booking_id) VALUES
(1, 'John Doe', 'john.doe@example.com', '+1234567890', NULL),
(2, 'Alice Smith', 'alice.smith@example.com', '+1987654321', NULL),
(3, 'Bob Johnson', 'bob.johnson@example.com', '+1122334455', NULL),
(4, 'Emily Brown', 'emily.brown@example.com', '+1443322110', NULL),
(5, 'Michael Wilson', 'michael.wilson@example.com', '+1555666777', NULL),
(6, 'Jennifer Davis', 'jennifer.davis@example.com', '+1777333444', NULL),
(7, 'Christopher Taylor', 'christopher.taylor@example.com', '+1888999900', NULL),
(8, 'Jessica Martinez', 'jessica.martinez@example.com', '+1999888777', NULL),
(9, 'David Anderson', 'david.anderson@example.com', '+1555222233', NULL),
(10, 'Sarah Garcia', 'sarah.garcia@example.com', '+1222111333', NULL),
(11, 'Matthew Hernandez', 'matthew.hernandez@example.com', '+1999111222', NULL),
(12, 'Laura Lopez', 'laura.lopez@example.com', '+1444777999', NULL),
(13, 'William Gonzalez', 'william.gonzalez@example.com', '+1333666999', NULL),
(14, 'Samantha Perez', 'samantha.perez@example.com', '+1888333666', NULL),
(15, 'Daniel Wilson', 'daniel.wilson@example.com', '+1222333444', NULL),
(16, 'Olivia Miller', 'olivia.miller@example.com', '+1333222111', NULL),
(17, 'James Brown', 'james.brown@example.com', '+1777555444', NULL),
(18, 'Charlotte Moore', 'charlotte.moore@example.com', '+1444555666', NULL),
(19, 'Benjamin Lee', 'benjamin.lee@example.com', '+1999777888', NULL),
(20, 'Amelia Taylor', 'amelia.taylor@example.com', '+1555999888', NULL),
(21, 'Ethan Jackson', 'ethan.jackson@example.com', '+1888555111', NULL),
(22, 'Mia White', 'mia.white@example.com', '+1222444111', NULL),
(23, 'Alexander Harris', 'alexander.harris@example.com', '+1888444222', NULL),
(24, 'Sophia Clark', 'sophia.clark@example.com', '+1999222333', NULL),
(25, 'Aiden Young', 'aiden.young@example.com', '+1222999333', NULL),
(26, 'Ava Scott', 'ava.scott@example.com', '+1444111999', NULL),
(27, 'Logan Green', 'logan.green@example.com', '+1777888999', NULL),
(28, 'Chloe Hall', 'chloe.hall@example.com', '+1999444555', NULL),
(29, 'Jackson King', 'jackson.king@example.com', '+1222888999', NULL),
(30, 'Penelope Adams', 'penelope.adams@example.com', '+1333444888', NULL),
(31, 'Lucas Wright', 'lucas.wright@example.com', '+1888333222', NULL),
(32, 'Madison Hughes', 'madison.hughes@example.com', '+1222333999', NULL),
(33, 'Evelyn Nelson', 'evelyn.nelson@example.com', '+1444666333', NULL),
(34, 'Jack Walker', 'jack.walker@example.com', '+1999666888', NULL),
(35, 'Grace Hill', 'grace.hill@example.com', '+1555444777', NULL),
(36, 'Gabriel Young', 'gabriel.young@example.com', '+1888777555', NULL),
(37, 'Lily Cook', 'lily.cook@example.com', '+1222444333', NULL),
(38, 'Owen Murphy', 'owen.murphy@example.com', '+1333999111', NULL),
(39, 'Zoe Carter', 'zoe.carter@example.com', '+1444888999', NULL),
(40, 'Connor Rivera', 'connor.rivera@example.com', '+1888444777', NULL);

-- Insert data into Booking table
INSERT INTO Booking (booking_id, customer_id, event_id, num_tickets, total_cost, booking_date) VALUES
(1, 1, 1, 2, 400.00, '2024-05-15'),
(2, 2, 2, 4, 2000.00, '2024-01-30'),
(3, 3, 3, 3, 1050.00, '2024-04-08'),
(4, 4, 4, 1, 150.00, '2024-07-10'),
```

```
(5, 5, 5, 2, 2000.00, '2024-02-20'),
(6, 6, 6, 2, 600.00, '2024-11-20'),
(7, 7, 7, 5, 1250.00, '2024-06-20'),
(8, 8, 8, 3, 540.00, '2024-08-30'),
(9, 9, 9, 6, 720.00, '2024-07-16'),
(10, 10, 10, 2, 800.00, '2024-07-15'),
(11, 11, 11, 4, 800.00, '2024-01-20'),
(12, 12, 12, 3, 450.00, '2024-06-10'),
(13, 13, 13, 2, 160.00, '2024-07-19'),
(14, 14, 14, 3, 900.00, '2024-05-10'),
(15, 15, 15, 4, 1200.00, '2024-11-28'),
(16, 16, 16, 5, 600.00, '2024-07-18'),
(17, 17, 17, 2, 500.00, '2024-05-08'),
(18, 18, 18, 4, 800.00, '2024-05-15'),
(19, 19, 19, 3, 1050.00, '2024-06-10'),
(20, 20, 20, 2, 600.00, '2024-08-20');
```

Output:

```
77  16:27:05  INSERT INTO Venu (venue_id, venue_name, address) VALUES (1, 'Wembley Stadium', 'London, UK'), (2, 'Madison Square Garden', 'New York, USA'),...  40 row(s) affected Records: 40  Duplicates: 0  Warnings: 0  0.016 sec
78  16:27:05  INSERT INTO Event (event_id, event_name, event_date, event_time, venue_id, total_seats, available_seats, ticket_price, event_type, booking_id) VA...  20 row(s) affected Records: 20  Duplicates: 0  Warnings: 0  0.000 sec
79  16:27:05  INSERT INTO Customer (customer_id, customer_name, email, phone_number, booking_id) VALUES (1, 'John Doe', 'john.doe@example.com', '+12345...  40 row(s) affected Records: 40  Duplicates: 0  Warnings: 0  0.000 sec
80  16:27:05  INSERT INTO Booking (booking_id, customer_id, event_id, num_tickets, total_cost, booking_date) VALUES (1, 1, 1, 2, 400.00, '2024-05-15'), (2, 2, 2, ...  20 row(s) affected Records: 20  Duplicates: 0  Warnings: 0  0.000 sec
```

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

-- Task 2

-- 2. Listing all Events:

SELECT * FROM Event;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	UEFA Champions League Final	2024-05-29	20:00:00	1	90000	85000	200.00	Sports	NULL
2	Super Bowl	2024-02-04	18:30:00	15	70000	68000	500.00	Sports	NULL
3	Coachella Music Festival	2024-04-12	12:00:00	2	125000	120000	350.00	Concert	NULL
4	Wimbledon Men's Final	2024-07-14	14:00:00	17	15000	14000	150.00	Sports	NULL
5	Academy Awards	2024-02-25	17:00:00	5	3000	2800	1000.00	Concert	NULL
6	World Cup Final	2024-12-15	20:00:00	11	100000	95000	300.00	Sports	NULL
7	Glastonbury Festival	2024-06-28	12:00:00	1	135000	130000	250.00	Concert	NULL
8	US Open Tennis Men's Final	2024-09-08	16:00:00	22	22000	20000	180.00	Sports	NULL
9	Comic-Con International	2024-07-18	10:00:00	29	150000	148000	120.00	Concert	NULL
10	Summer Olympics Opening Ceremony	2024-07-19	20:00:00	30	80000	78000	400.00	Sports	NULL
11	Australian Open Women's Final	2024-01-27	15:00:00	31	18000	17000	200.00	Sports	NULL
12	Bonnaroo Music Festival	2024-06-13	12:00:00	3	80000	75000	150.00	Concert	NULL
13	Tour de France Final Stage	2024-07-21	14:00:00	21	10000	9500	80.00	Sports	NULL
14	EDC Las Vegas	2024-05-17	18:00:00	13	160000	155000	300.00	Concert	NULL
15	FIFA World Cup Final	2024-12-01	20:00:00	11	100000	95000	300.00	Sports	NULL
16	San Diego Comic-Con	2024-07-20	09:00:00	29	150000	148000	120.00	Concert	NULL
17	Indian Premier League Final	2024-05-12	19:00:00	12	50000	48000	250.00	Sports	NULL
18	Eurovision Song Contest Final	2024-05-18	21:00:00	20	15000	14000	200.00	Concert	NULL
19	NBA Finals Game 7	2024-06-20	20:30:00	4	21000	20000	350.00	Sports	NULL
20	Burning Man Festival	2024-08-25	12:00:00	32	80000	75000	300.00	Concert	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 3. Selecting events with available tickets:

SELECT * FROM Event WHERE available_seats > 0;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	UEFA Champions League Final	2024-05-29	20:00:00	1	90000	85000	200.00	Sports	NULL
2	Super Bowl	2024-02-04	18:30:00	15	70000	68000	500.00	Sports	NULL
3	Coachella Music Festival	2024-04-12	12:00:00	2	125000	120000	350.00	Concert	NULL
4	Wimbledon Men's Final	2024-07-14	14:00:00	17	15000	14000	150.00	Sports	NULL
5	Academy Awards	2024-02-25	17:00:00	5	3000	2800	1000.00	Concert	NULL
6	World Cup Final	2024-12-15	20:00:00	11	100000	95000	300.00	Sports	NULL
7	Glastonbury Festival	2024-06-28	12:00:00	1	135000	130000	250.00	Concert	NULL
8	US Open Tennis Men's Final	2024-09-08	16:00:00	22	22000	20000	180.00	Sports	NULL
9	Comic-Con International	2024-07-18	10:00:00	29	150000	148000	120.00	Concert	NULL
10	Summer Olympics Opening Ceremony	2024-07-19	20:00:00	30	80000	78000	400.00	Sports	NULL
11	Australian Open Women's Final	2024-01-27	15:00:00	31	18000	17000	200.00	Sports	NULL
12	Bonnaroo Music Festival	2024-06-13	12:00:00	3	80000	75000	150.00	Concert	NULL
13	Tour de France Final Stage	2024-07-21	14:00:00	21	10000	9500	80.00	Sports	NULL
14	EDC Las Vegas	2024-05-17	18:00:00	13	160000	155000	300.00	Concert	NULL
15	FIFA World Cup Final	2024-12-01	20:00:00	11	100000	95000	300.00	Sports	NULL
16	San Diego Comic-Con	2024-07-20	09:00:00	29	150000	148000	120.00	Concert	NULL
17	Indian Premier League Final	2024-05-12	19:00:00	12	50000	48000	250.00	Sports	NULL
18	Eurovision Song Contest Final	2024-05-18	21:00:00	20	15000	14000	200.00	Concert	NULL
19	NBA Finals Game 7	2024-06-20	20:30:00	4	21000	20000	350.00	Sports	NULL
20	Burning Man Festival	2024-08-25	12:00:00	32	80000	75000	300.00	Concert	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 4. Selecting events with a partial match on event name with 'cup':
SELECT * FROM Event WHERE event_name LIKE '%cup%';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
6	World Cup Final	2024-12-15	20:00:00	11	100000	95000	300.00	Sports	NULL
15	FIFA World Cup Final	2024-12-01	20:00:00	11	100000	95000	300.00	Sports	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 5. Selecting events with ticket price range between 1000 to 2500:
SELECT * FROM Event WHERE ticket_price BETWEEN 1000 AND 2500;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
5	Academy Awards	2024-02-25	17:00:00	5	3000	2800	1000.00	Concert	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 6. Retrieving events with dates falling within a specific range:
SELECT * FROM Event WHERE event_date BETWEEN '2024-04-20' AND '2024-04-23';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 7. Retrieving events with available tickets and also have "Concert" in their name:
SELECT * FROM Event WHERE available_seats > 0 AND event_name LIKE '%Concert%';

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 8. Retrieving users in batches of 5, starting from the 6th user:
SELECT * FROM Customer LIMIT 5 OFFSET 5;

customer_id	customer_name	email	phone_number	booking_id
6	Jennifer Davis	jennifer.davis@example.com	+1777333444	NULL
7	Christopher Taylor	christopher.taylor@example.com	+1888999900	NULL
8	Jessica Martinez	jessica.martinez@example.com	+1999888777	NULL
9	David Anderson	david.anderson@example.com	+1555222233	NULL
10	Sarah Garcia	sarah.garcia@example.com	+1222111333	NULL
NULL	NULL	NULL	NULL	NULL

-- 9. Retrieving booking details containing booked no of tickets more than 4:
SELECT * FROM Booking WHERE num_tickets > 4;

booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
7	7	7	5	1250.00	2024-06-20
9	9	9	6	720.00	2024-07-16
16	16	16	5	600.00	2024-07-18
NULL	NULL	NULL	NULL	NULL	NULL

-- 10. Retrieving customer information whose phone number ends with '000':
SELECT * FROM Customer WHERE phone_number LIKE '%000';

customer_id	customer_name	email	phone_number	booking_id
NULL	NULL	NULL	NULL	NULL

-- 11. Retrieving events in order whose seat capacity is more than 15000:
SELECT * FROM Event WHERE total_seats > 15000 ORDER BY total_seats;

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
11	Australian Open Women's Final	2024-01-27	15:00:00	31	18000	17000	200.00	Sports	NULL
19	NBA Finals Game 7	2024-06-20	20:30:00	4	21000	20000	350.00	Sports	NULL
8	US Open Tennis Men's Final	2024-09-08	16:00:00	22	22000	20000	180.00	Sports	NULL
17	Indian Premier League Final	2024-05-12	19:00:00	12	50000	48000	250.00	Sports	NULL
2	Super Bowl	2024-02-04	18:30:00	15	70000	68000	500.00	Sports	NULL
10	Summer Olympics Opening Ceremony	2024-07-19	20:00:00	30	80000	78000	400.00	Sports	NULL
12	Bonnaroo Music Festival	2024-06-13	12:00:00	3	80000	75000	150.00	Concert	NULL
20	Burning Man Festival	2024-08-25	12:00:00	32	80000	75000	300.00	Concert	NULL
1	UEFA Champions League Final	2024-05-29	20:00:00	1	90000	85000	200.00	Sports	NULL
6	World Cup Final	2024-12-15	20:00:00	11	100000	95000	300.00	Sports	NULL
15	FIFA World Cup Final	2024-12-01	20:00:00	11	100000	95000	300.00	Sports	NULL
3	Coachella Music Festival	2024-04-12	12:00:00	2	125000	120000	350.00	Concert	NULL
7	Glastonbury Festival	2024-06-28	12:00:00	1	135000	130000	250.00	Concert	NULL
9	Comic-Con International	2024-07-18	10:00:00	29	150000	148000	120.00	Concert	NULL
16	San Diego Comic-Con	2024-07-20	09:00:00	29	150000	148000	120.00	Concert	NULL
14	EDC Las Vegas	2024-05-17	18:00:00	13	160000	155000	300.00	Concert	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

-- 12. Selecting events name not starting 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_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	UEFA Champions League Final	2024-05-29	20:00:00	1	90000	85000	200.00	Sports	NULL
2	Super Bowl	2024-02-04	18:30:00	15	70000	68000	500.00	Sports	NULL
3	Coachella Music Festival	2024-04-12	12:00:00	2	125000	120000	350.00	Concert	NULL
4	Wimbledon Men's Final	2024-07-14	14:00:00	17	15000	14000	150.00	Sports	NULL
5	Academy Awards	2024-02-25	17:00:00	5	3000	2800	1000.00	Concert	NULL
6	World Cup Final	2024-12-15	20:00:00	11	100000	95000	300.00	Sports	NULL
7	Glastonbury Festival	2024-06-28	12:00:00	1	135000	130000	250.00	Concert	NULL
8	US Open Tennis Men's Final	2024-09-08	16:00:00	22	22000	20000	180.00	Sports	NULL
9	Comic-Con International	2024-07-18	10:00:00	29	150000	148000	120.00	Concert	NULL
10	Summer Olympics Opening Ceremony	2024-07-19	20:00:00	30	80000	78000	400.00	Sports	NULL
11	Australian Open Women's Final	2024-01-27	15:00:00	31	18000	17000	200.00	Sports	NULL
12	Bonnaroo Music Festival	2024-06-13	12:00:00	3	80000	75000	150.00	Concert	NULL
13	Tour de France Final Stage	2024-07-21	14:00:00	21	10000	9500	80.00	Sports	NULL
14	EDC Las Vegas	2024-05-17	18:00:00	13	160000	155000	300.00	Concert	NULL
15	FIFA World Cup Final	2024-12-01	20:00:00	11	100000	95000	300.00	Sports	NULL
16	San Diego Comic-Con	2024-07-20	09:00:00	29	150000	148000	120.00	Concert	NULL
17	Indian Premier League Final	2024-05-12	19:00:00	12	50000	48000	250.00	Sports	NULL
18	Eurovision Song Contest Final	2024-05-18	21:00:00	20	15000	14000	200.00	Concert	NULL
19	NBA Finals Game 7	2024-06-20	20:30:00	4	21000	20000	350.00	Sports	NULL
20	Burning Man Festival	2024-08-25	12:00:00	32	80000	75000	300.00	Concert	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

-- 1. Listing Events and Their Average Ticket Prices:
SELECT event_id, event_name, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY event_id, event_name;

event_id	event_name	average_ticket_price
1	UEFA Champions League Final	200.000000
2	Super Bowl	500.000000
3	Coachella Music Festival	350.000000
4	Wimbledon Men's Final	150.000000
5	Academy Awards	1000.000000
6	World Cup	300.000000
7	Glastonbury Festival	250.000000
8	US Open Tennis Men's Final	180.000000
9	Comic-Con International	120.000000
10	Summer Olympics Opening Ceremony	400.000000
11	Australian Open Women's Final	200.000000
12	Bonnaroo Music Festival	150.000000
13	Tour de France Final Stage	80.000000
14	EDC Las Vegas	300.000000
15	FIFA World Cup Final	300.000000
16	San Diego Comic-Con	120.000000
17	Indian Premier League Final	250.000000
18	Eurovision Song Contest Final	200.000000
19	NBA Finals Game 7	350.000000
20	Burning Man Festival	300.000000

-- 2. Calculating the Total Revenue Generated by Events:
```
SELECT SUM(total_cost) AS total_revenue
FROM Booking;
```

total_revenue
16570.00

-- 3. Finding the Event with the Highest Ticket Sales:
```
SELECT event_id, SUM(num_tickets) AS total_tickets_sold
FROM Booking
GROUP BY event_id
ORDER BY total_tickets_sold DESC
LIMIT 1;
```

event_id	total_tickets_sold
9	6

-- 4. Calculating the Total Number of Tickets Sold for Each Event:
```
SELECT event_id, SUM(num_tickets) AS total_tickets_sold
FROM Booking
GROUP BY event_id;
```

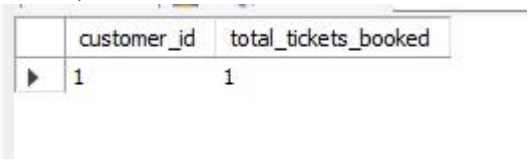
event_id	total_tickets_sold
1	2
2	4
3	3
4	1
5	2
6	2
7	5
8	3
9	6
10	2
11	4
12	3
13	2
14	3
15	4
16	5
17	2
18	4
19	3
20	2

-- 5. Finding Events with No Ticket Sales:
SELECT event_id, event_name
FROM Event
GROUP BY event_id, event_name
HAVING COUNT(booking_id) = 0;

event_id	event_name
1	UEFA Champions League Final
2	Super Bowl
3	Coachella Music Festival
4	Wimbledon Men's Final
5	Academy Awards
6	World Cup Final
7	Glastonbury Festival
8	US Open Tennis Men's Final
9	Comic-Con International
10	Summer Olympics Opening Ceremony
11	Australian Open Women's Final
12	Bonnaroo Music Festival
13	Tour de France Final Stage
14	EDC Las Vegas
15	FIFA World Cup Final
16	San Diego Comic-Con
17	Indian Premier League Final
18	Eurovision Song Contest Final
19	NBA Finals Game 7
20	Burning Man Festival
NULL	NULL

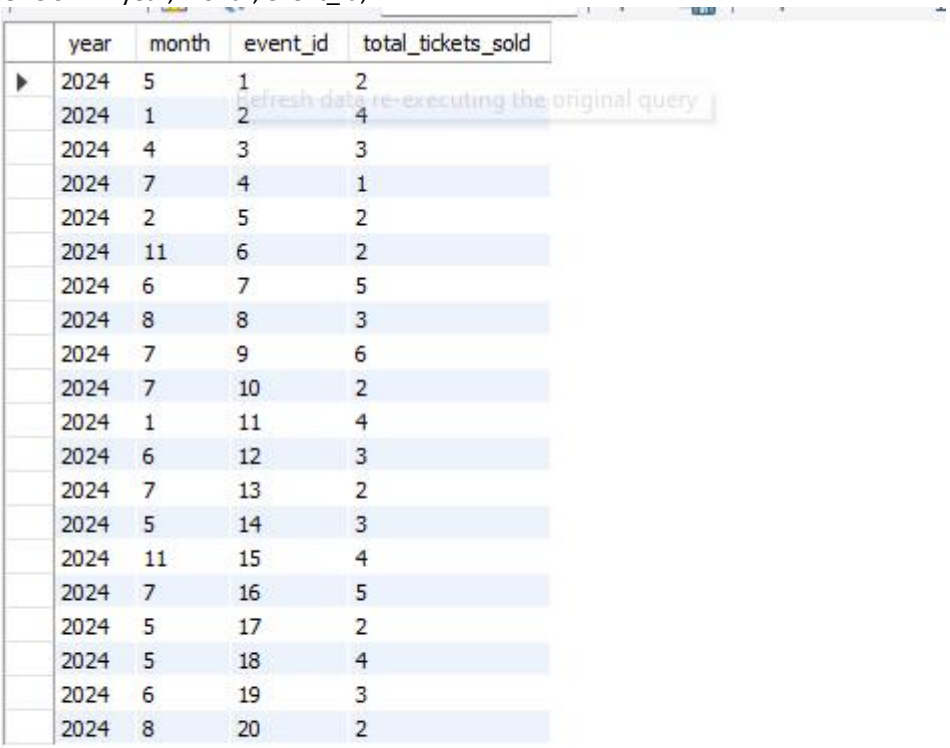
-- 6. Finding the User Who Has Booked the Most Tickets:
SELECT customer_id, COUNT(*) AS total_tickets_booked
FROM Booking
GROUP BY customer_id

```
ORDER BY total_tickets_booked DESC
LIMIT 1;
```



customer_id	total_tickets_booked
1	1

```
-- 7. Listing Events and the Total Number of Tickets Sold for Each Month:
SELECT YEAR(booking_date) AS year, MONTH(booking_date) AS month, event_id, SUM(num_tickets) AS total_tickets_sold
FROM Booking
GROUP BY year, month, event_id;
```



year	month	event_id	total_tickets_sold
2024	5	1	2
2024	1	2	4
2024	4	3	3
2024	7	4	1
2024	2	5	2
2024	11	6	2
2024	6	7	5
2024	8	8	3
2024	7	9	6
2024	7	10	2
2024	1	11	4
2024	6	12	3
2024	7	13	2
2024	5	14	3
2024	11	15	4
2024	7	16	5
2024	5	17	2
2024	5	18	4
2024	6	19	3
2024	8	20	2

```
-- 8. Calculating the Average Ticket Price for Events in Each Venue:
SELECT venue_id, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY venue_id;
```

	venue_id	average_ticket_price
▶	1	225.000000
	2	350.000000
	3	150.000000
	4	350.000000
	5	1000.000000
	11	300.000000
	12	250.000000
	13	30(250.000000
	15	500.000000
	17	150.000000
	20	200.000000
	21	80.000000
	22	180.000000
	29	120.000000
	30	400.000000
	31	200.000000
	32	300.000000

-- 9. Calculating the Total Number of Tickets Sold for Each Event Type:
SELECT e.event_type, SUM(b.num_tickets) AS total_tickets_sold
FROM Event e
JOIN Booking b ON e.event_id = b.event_id
GROUP BY e.event_type;

	event_type	total_tickets_sold
▶	Sports	29
	Concert	33

-- 10. Calculating the Total Revenue Generated by Events in Each Year:
SELECT YEAR(booking_date) AS year, SUM(total_cost) AS total_revenue
FROM Booking
GROUP BY year;

	year	total_revenue
▶	2024	16570.00

-- 11. Listing Users Who Have Booked Tickets for Multiple Events:
SELECT customer_id, COUNT(DISTINCT event_id) AS events_booked
FROM Booking
GROUP BY customer_id
HAVING events_booked > 1;

	customer_id	events_booked

-- 12. Calculating the Total Revenue Generated by Events for Each User:
SELECT customer_id, SUM(total_cost) AS total_revenue
FROM Booking
GROUP BY customer_id;

customer_id	total_revenue
1	400.00
2	2000.00
3	1050.00
4	150.00
5	2000.00
6	600.00
7	1250.00
8	540.00
9	720.00
10	800.00
11	800.00
12	450.00
13	160.00
14	900.00
15	1200.00
16	600.00
17	500.00
18	800.00
19	1050.00
20	600.00

-- 13. Calculating the Average Ticket Price for Events in Each Category and Venue:

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

venue_id	event_type	average_ticket_price
1	Sports	200.000000
15	Sports	500.000000
2	Concert	350.000000
17	Sports	150.000000
5	Concert	1000.000000
11	Sports	300.000000
1	Concert	250.000000
22	Sports	180.000000
29	Concert	120.000000
30	Sports	400.000000
31	Sports	200.000000
3	Concert	150.000000
21	Sports	80.000000
13	Concert	300.000000
12	Sports	250.000000
20	Concert	200.000000
4	Sports	350.000000
32	Concert	300.000000

-- 14. Listing Users and the Total Number of Tickets They've Purchased in the Last 30 Days:

```
SELECT customer_id, SUM(num_tickets) AS total_tickets_purchased
FROM Booking
WHERE booking_date >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
GROUP BY customer_id;
```

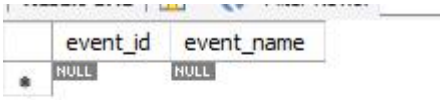
customer_id	total_tickets_purchased
1	2
3	3
4	1
6	2
7	5
8	3
9	6
10	2
12	3
13	2
14	3
15	4
16	5
17	2
18	4
19	3
20	2

Tasks 4: Subquery and its types

-- 1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
SELECT venue_id, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY venue_id;

venue_id	average_ticket_price
1	225.000000
2	350.000000
3	150.000000
4	350.000000
5	1000.000000
11	300.000000
12	250.000000
13	300.000000
15	500.000000
17	150.000000
20	200.000000
21	80.000000
22	180.000000
29	120.000000
30	400.000000
31	200.000000
32	300.000000

-- 2. Find Events with More Than 50% of Tickets Sold using subquery
SELECT event_id, event_name
FROM Event
WHERE (
  SELECT SUM(num_tickets)
  FROM Booking
  WHERE Booking.event_id = Event.event_id
) > 0.5 * total_seats;



event_id	event_name
NULL	NULL

-- 3. Calculate the Total Number of Tickets Sold for Each Event
SELECT event_id, SUM(num_tickets) AS total_tickets_sold
FROM Booking
GROUP BY event_id;

event_id	total_tickets_sold
1	2
2	4
3	3
4	1
5	2
6	2
7	5
8	3
9	6
10	2
11	4
12	3
13	2
14	3
15	4
16	5
17	2
18	4
19	3
20	2

-- 4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery
SELECT customer_id, customer_name
FROM Customer
WHERE NOT EXISTS (
  SELECT *
  FROM Booking
  WHERE Booking.customer_id = Customer.customer_id
);

customer_id	customer_name
21	Ethan Jackson
22	Mia White
23	Alexander Harris
24	Sophia Clark
25	Aiden Young
26	Ava Scott
27	Logan Green
28	Chloe Hall
29	Jackson King
30	Penelope Adams
31	Lucas Wright
32	Madison Hughes
33	Evelyn Nelson
34	Jack Walker
35	Grace Hill
36	Gabriel Young
37	Lily Cook
38	Owen Murphy
39	Zoe Carter
40	Connor Rivera
NULL	NULL

```
-- 5. List Events with No Ticket Sales Using a NOT IN Subquery
SELECT event_id, event_name
FROM Event
WHERE event_id NOT IN (
   SELECT DISTINCT event_id
   FROM Booking
);
```

event_id	event_name
NULL	NULL

```
-- 6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause
SELECT E.event_type, COALESCE(SUM(B.num_tickets), 0) AS total_tickets_sold
FROM (SELECT DISTINCT event_type FROM Event) AS E
LEFT JOIN (
   SELECT event_id, num_tickets
   FROM Booking
) AS B ON E.event_type = (
   SELECT event_type
   FROM Event
   WHERE Event.event_id = B.event_id
)
GROUP BY E.event_type;
```

event_type	total_tickets_sold
Sports	29
Concert	33

-- 7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause
```
SELECT event_id, event_name, ticket_price
FROM Event
WHERE ticket_price > (
   SELECT AVG(ticket_price)
   FROM Event
);
```

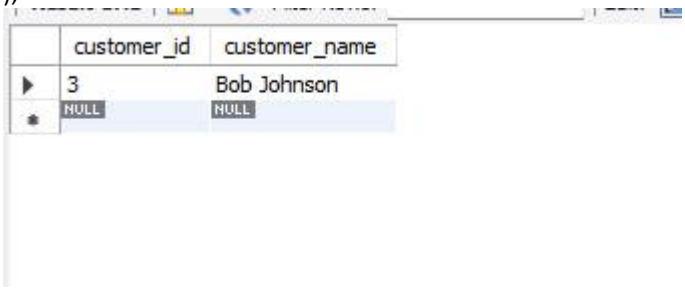
event_id	event_name	ticket_price
2	Super Bowl	500.00
3	Coachella Music Festival	350.00
5	Academy Awards	1000.00
6	World Cup Final	300.00
10	Summer Olympics Opening Ceremony	400.00
14	EDC Las Vegas	300.00
15	FIFA World Cup Final	300.00
19	NBA Finals Game 7	350.00
20	Burning Man Festival	300.00
NULL	NULL	NULL

-- 8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery
```
SELECT customer_id, customer_name,
   (SELECT SUM(total_cost) FROM Booking WHERE Booking.customer_id = Customer.customer_id) AS total_revenue
FROM Customer;
```

customer_id	customer_name	total_revenue
1	John Doe	400.00
2	Alice Smith	2000.00
3	Bob Johnson	1050.00
4	Emily Brown	150.00
5	Michael Wilson	2000.00
6	Jennifer Davis	600.00
7	Christopher Taylor	1250.00
8	Jessica Martinez	540.00
9	David Anderson	720.00
10	Sarah Garcia	800.00
11	Matthew Hernandez	800.00
12	Laura Lopez	450.00
13	William Gonzalez	160.00
14	Samantha Perez	900.00
15	Daniel Wilson	1200.00
16	Olivia Miller	600.00
17	James Brown	500.00
18	Charlotte Moore	800.00
19	Benjamin Lee	1050.00
20	Amelia Taylor	600.00

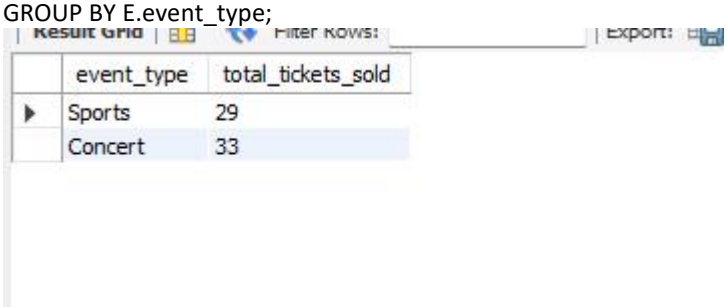
-- 9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause
```
SELECT customer_id, customer_name
FROM Customer
```

```
WHERE EXISTS (
  SELECT *
  FROM Booking
  WHERE Booking.customer_id = Customer.customer_id
  AND Booking.event_id IN (
    SELECT event_id
    FROM Event
    WHERE venue_id = 2
  )
);
```



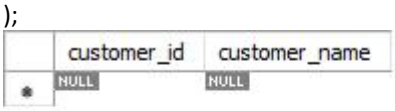
customer_id	customer_name
3	Bob Johnson
NULL	NULL

```
-- 10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY
SELECT E.event_type, SUM(B.num_tickets) AS total_tickets_sold
FROM Event AS E
LEFT JOIN Booking AS B ON E.event_id = B.event_id
GROUP BY E.event_type;
```



event_type	total_tickets_sold
Sports	29
Concert	33

```
-- 11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT
SELECT customer_id, customer_name
FROM Customer
WHERE EXISTS (
  SELECT *
  FROM Booking
  WHERE Booking.customer_id = Customer.customer_id
  AND DATE_FORMAT(Booking.booking_date, '%Y-%m') = '{given_month}'
);
```



customer_id	customer_name
NULL	NULL

```
-- 12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
SELECT venue_id, AVG(ticket_price) AS average_ticket_price
FROM Event
GROUP BY venue_id;
```

venue_id	average_ticket_price
1	225.000000
2	350.000000
3	150.000000
4	350.000000
5	1000.000000
11	300.000000
12	250.000000
13	300.000000
15	500.000000
17	150.000000
20	200.000000
21	80.000000
22	180.000000
29	120.000000
30	400.000000
31	200.000000
32	300.000000