create database ticketbookingsystem;

use ticketbookingsystem;

SET @OLD\_FOREIGN\_KEY\_CHECKS = @@FOREIGN\_KEY\_CHECKS;

SET FOREIGN\_KEY\_CHECKS = 0;

create table venue(

venue\_id int primary key,

venue\_name varchar(100),

address varchar(100)

);

create table event(

event\_id int primary key,

event\_name varchar(100),

event\_date date,

event\_time time,

venue\_id int null ,

foreign key(venue\_id) references venue(venue\_id),

total\_seats int,

available\_Seats int,

ticket\_price int,

event\_type varchar(100),

booking\_id int null,

foreign key(booking\_id) references booking(booking\_id)

);

create table customer(

customer\_id int primary key,

customer\_name varchar(100),

email varchar(100),

phone\_number int,

booking\_id int,

foreign key(booking\_id) references booking(booking\_id)

);

create table booking(

booking\_id int primary key,

customer\_id int null,

foreign key(customer\_id) references customer(customer\_id),

num\_tickets int,

total\_cost int,

booking\_date date

);

-- 1.

-- Insert into the "venue" table

INSERT INTO venue (venue\_id, venue\_name, address)

VALUES

(1, 'Concert Hall', '123 Main Street'),

(2, 'Conference Center', '456 Oak Avenue'),

(3, 'Sports Stadium', '789 Maple Lane'),

(4, 'Comedy Club', '101 Pine Road'),

(5, 'Art Gallery', '202 Artist Street');

-- Insert into the "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

(101, 'Music Festival', '2023-12-15', '19:00:00', 1, 2000, 1500, 50, 'Music', 201),

(102, 'Tech Summit', '2023-11-25', '10:00:00', 2, 500, 300, 100, 'Conference', 202),

(103, 'Football Match', '2023-11-30', '15:30:00', 3, 10000, 8000, 30, 'Sports', NULL),

(104, 'Stand-up Comedy Show', '2023-12-05', '20:30:00', 4, 800, 400, 25, 'Comedy', NULL),

(105, 'Art Exhibition', '2023-11-20', '11:00:00', 5, 300, 200, 20, 'Exhibition', 203);

-- Insert into the "customer" table

INSERT INTO customer (customer\_id, customer\_name, email, phone\_number, booking\_id)

VALUES

(501, 'Alice Johnson', 'alice.johnson@email.com', 11122233, 301),

(502, 'Bob Smith', 'bob.smith@email.com', 44455566, 302),

(503, 'Charlie Davis', 'charlie.davis@email.com', 77788899, 303),

(504, 'David Brown', 'david.brown@email.com', 12345678, NULL),

(505, 'Eva White', 'eva.white@email.com', 98765432, NULL);

-- Insert into the "booking" table

INSERT INTO booking (booking\_id, customer\_id, num\_tickets, total\_cost, booking\_date)

VALUES

(201, 501, 3, 150, '2023-11-22'),

(202, 502, 2, 200, '2023-11-10'),

(203, 503, 1, 20, '2023-11-18'),

(204, NULL, 5, 250, '2023-12-01'),

(205, NULL, 4, 100, '2023-11-15');

select \* from customer;

-- 2.

SELECT \* FROM event;

-- 3.

SELECT \* FROM event WHERE available\_seats > 0;

-- 4.

SELECT \* FROM event WHERE event\_name LIKE '%cup%';

-- 5.

SELECT \* FROM event WHERE ticket\_price BETWEEN 1000 AND 2500;

-- 6.

SELECT \* FROM event WHERE event\_date BETWEEN '2023-11-01' AND '2023-12-31';

-- 7.

SELECT \* FROM event WHERE available\_seats > 0 AND event\_name LIKE '%Concert%';

-- 8.

SELECT \* FROM customer OFFSET 5 LIMIT 5;

-- 9.

SELECT \* FROM booking WHERE num\_tickets > 4;

-- 10.

SELECT \* FROM customer WHERE phone\_number LIKE '%000';

-- 11.

SELECT \* FROM event WHERE total\_seats > 15000 ORDER BY total\_seats;

-- 12.

SELECT \* FROM event WHERE NOT (event\_name LIKE 'x%' OR event\_name LIKE 'y%' OR event\_name LIKE 'z%');

-- ---------------------------AGGREGATE FUNCTIONS, GROUP BY , AND JOIN-----------------------------------------------------------------

-- 1

SELECT event\_id, event\_name, AVG(ticket\_price) AS average\_ticket\_price

FROM event

GROUP BY event\_id, event\_name;

-- 2

SELECT event\_id, event\_name, SUM(total\_cost) AS total\_revenue

FROM booking

JOIN event ON booking.event\_id = event.event\_id

GROUP BY event\_id, event\_name;

-- 3

SELECT event\_id, event\_name, SUM(num\_tickets) AS total\_tickets\_sold

FROM booking

JOIN event ON booking.event\_id = event.event\_id

GROUP BY event\_id, event\_name

ORDER BY total\_tickets\_sold DESC

LIMIT 1;

-- 4

SELECT event\_id, event\_name, SUM(num\_tickets) AS total\_tickets\_sold

FROM booking

JOIN event ON booking.event\_id = event.event\_id

GROUP BY event\_id, event\_name;

-- 5

SELECT event\_id, event\_name

FROM event

WHERE event\_id NOT IN (SELECT DISTINCT event\_id FROM booking);

-- 6

SELECT customer\_id, customer\_name, SUM(num\_tickets) AS total\_tickets\_purchased

FROM booking

JOIN customer ON booking.customer\_id = customer.customer\_id

GROUP BY customer\_id, customer\_name

ORDER BY total\_tickets\_purchased DESC

LIMIT 1;

-- 7

SELECT EXTRACT(MONTH FROM booking\_date) AS month, event\_id, event\_name, SUM(num\_tickets) AS total\_tickets\_sold

FROM booking

JOIN event ON booking.event\_id = event.event\_id

GROUP BY month, event\_id, event\_name;

-- 8

SELECT venue\_id, AVG(ticket\_price) AS average\_ticket\_price

FROM event

GROUP BY venue\_id;

-- 9

SELECT event\_type, SUM(num\_tickets) AS total\_tickets\_sold

FROM booking

JOIN event ON booking.event\_id = event.event\_id

GROUP BY event\_type;

-- 10

SELECT EXTRACT(YEAR FROM booking\_date) AS year, SUM(total\_cost) AS total\_revenue

FROM booking

GROUP BY year;

-- 11

SELECT customer\_id, customer\_name

FROM customer

WHERE customer\_id IN (SELECT customer\_id FROM booking GROUP BY customer\_id HAVING COUNT(DISTINCT event\_id) > 1);

-- 12

SELECT customer\_id, customer\_name, SUM(total\_cost) AS total\_revenue

FROM booking

JOIN customer ON booking.customer\_id = customer.customer\_id

GROUP BY customer\_id, customer\_name;

-- 13

SELECT venue\_id, event\_type, AVG(ticket\_price) AS average\_ticket\_price

FROM event

GROUP BY venue\_id, event\_type;

-- 14

SELECT customer\_id, customer\_name, SUM(num\_tickets) AS total\_tickets\_purchased

FROM booking

JOIN customer ON booking.customer\_id = customer.customer\_id

WHERE booking\_date >= CURRENT\_DATE - INTERVAL '30' DAY

GROUP BY customer\_id, customer\_name;

-- -----------------------------------------------SUBQUERY---------------------------------------------------------------------------------

-- 1

SELECT venue\_id, AVG(ticket\_price) AS average\_ticket\_price

FROM event

WHERE venue\_id IN (SELECT DISTINCT venue\_id FROM event)

GROUP BY venue\_id;

-- 2

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;

-- 3

SELECT event\_id, event\_name, (SELECT SUM(num\_tickets) FROM booking WHERE booking.event\_id = event.event\_id) AS total\_tickets\_sold

FROM event;

-- 4

SELECT customer\_id, customer\_name

FROM customer

WHERE NOT EXISTS (SELECT 1 FROM booking WHERE booking.customer\_id = customer.customer\_id);

-- 5

SELECT event\_id, event\_name

FROM event

WHERE event\_id NOT IN (SELECT DISTINCT event\_id FROM booking);

-- 6

SELECT event\_type, SUM(total\_tickets\_sold) AS total\_tickets\_sold

FROM (SELECT event\_type, event\_id, (SELECT SUM(num\_tickets) FROM booking WHERE booking.event\_id = event.event\_id) AS total\_tickets\_sold FROM event) AS subquery

GROUP BY event\_type;

-- 7

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

FROM event

WHERE ticket\_price > (SELECT AVG(ticket\_price) FROM event);

-- 8

SELECT customer\_id, customer\_name, (SELECT SUM(total\_cost) FROM booking WHERE booking.customer\_id = customer.customer\_id) AS total\_revenue

FROM customer;

-- 9

SELECT customer\_id, customer\_name

FROM customer

WHERE customer\_id IN (SELECT DISTINCT customer\_id FROM booking WHERE booking.event\_id IN (SELECT event\_id FROM event WHERE venue\_id = 1));

-- 10

SELECT event\_type, SUM(total\_tickets\_sold) AS total\_tickets\_sold

FROM (SELECT event\_type, (SELECT SUM(num\_tickets) FROM booking WHERE booking.event\_id = event.event\_id) AS total\_tickets\_sold FROM event) AS subquery

GROUP BY event\_type;

-- 11

SELECT customer\_id, customer\_name

FROM customer

WHERE customer\_id IN (SELECT DISTINCT customer\_id FROM booking WHERE DATE\_FORMAT(booking\_date, '%Y-%m') = '2023-11');

-- 12

SELECT venue\_id, AVG(ticket\_price) AS average\_ticket\_price

FROM event

WHERE venue\_id IN (SELECT DISTINCT venue\_id FROM event)

GROUP BY venue\_id;