```
show databases;
use ticket_booking_system;

show tables;

-- insertions
insert into venue(venue_name,address) values
('mumbai', 'marol andheri(w)'),
('chennai', 'IT Park'),
('pondicherry ', 'state beach');

insert into
event(event_name,event_date,event_time,total_seats,available_seats,ticket_price,event_type,venue_id1)

values
('Late Ms. Lata Mangeshkar Musical', '2021-09-12','20:00',320,270,600,'concert',3),
('CSK vs RCB', '2024-04-11','19:30',23000,3,3600,'sports',2),
('CSK vs RR', '2024-04-19','19:30',23000,10,3400,'sports',2),
('MI vs KKR', '2024-05-01','15:30',28000,100,8000,'sports',1);

insert into customer(customer_name,email,phone_number)
values
('harry potter','harry@gmail.com','45454545'),
('ronald weasley','ron@gmail.com','45454545'),
('hermione granger','her@gmail.com','45454545'),
('draco malfoy','drac@gmail.com','45454545'),
('ginni weasley','ginni@gmail.com','45454545');

insert into event_has_customer (event_id,customer_id,num_tickets,total_cost,booking_date)
values
(4,1,2,640,'2021-09-12'),
(4,4,3,960,'2021-09-12'),
(5,1,3,10800,'2024-04-11'),
(5,3,5,18000,'2024-04-10'),
(6,5,10,34000,'2024-04-15'),
(7,2,4,32000,'2024-05-01');

-- Task 2 Select , Where , Between And Like :
-- 1. Write a sql query to insert at least 10 sample records into each table;
INSERT INTO venue (venue_name, address) VALUES
('delhi', 'Connaught Place'),
('bangalore', 'Indiranagar'),
('kolkata', 'Park Street'),
('hyderabad', 'HITEC City'),
('pune', 'Magarpatta City'),
('jaipur', 'Pink City'),
('ahmedabad', 'Gandhinagar'),
('lucknow', 'Hazratganj'),
('goa', 'Calangute Beach'),
('kochi', 'Fort Kochi');



INSERT INTO event (event_name, event_date, event_time, total_seats, available_seats, ticket_price,
event_type, venue_id1)
```

```
VALUES
('Sunburn Music Festival', '2024-12-28', '18:00', 5000, 4500, 2500, 'music', 1),
('IPL Final', '2024-06-05', '19:00', 35000, 2000, 5000, 'sports', 2),
('Indian Grand Prix', '2024-11-10', '14:00', 15000, 8000, 3500, 'sports', 3),
('TEDx Mumbai', '2024-09-15', '10:00', 1000, 900, 1200, 'conference', 1),
('Comic Con Chennai', '2024-08-20', '11:30', 5000, 4800, 800, 'entertainment', 2),
('International Yoga Day', '2024-06-21', '07:00', 2000, 1500, 300, 'health', 3),
('Food Festival', '2024-07-05', '17:00', 3000, 2800, 500, 'food', 1),
('Tech Expo Bangalore', '2024-10-30', '09:00', 5000, 4500, 1500, 'technology', 2),
('Fashion Show Mumbai', '2024-08-10', '20:00', 1500, 1200, 2000, 'fashion', 1),
('Cricket League Pune', '2024-07-25', '19:30', 20000, 18000, 3000, 'sports', 4);



INSERT INTO customer (customer_name, email, phone_number)
VALUES
('Albus Dumbledore', 'albus@gmail.com', '12345678'),
('Severus Snape', 'severus@gmail.com', '23456789'),
('Sirius Black', 'sirius@gmail.com', '34567890'),
('Remus Lupin', 'remus@gmail.com', '45678901'),
('Neville Longbottom', 'neville@gmail.com', '56789012'),
('Luna Lovegood', 'luna@gmail.com', '67890123'),
('Ginny Weasley', 'gweasley@gmail.com', '78901234'),
('Fred Weasley', 'fred@gmail.com', '89012345'),
('George Weasley', 'george@gmail.com', '90123456'),
('Bellatrix Lestrange', 'bellatrix@gmail.com', '01234567');



INSERT INTO event_has_customer (event_id, customer_id, num_tickets, total_cost, booking_date)
VALUES
(6, 3, 2, 6800, '2024-05-20'),
(7, 4, 2, 7200, '2024-05-05'),
(8, 5, 1, 1200, '2024-06-10'),
(9, 6, 3, 9600, '2024-06-15'),
(10, 7, 4, 16000, '2024-07-01'),
(11, 8, 5, 3000, '2024-07-10'),
(12, 9, 2, 600, '2024-08-05'),
(13, 10, 3, 1800, '2024-08-20'),
(14, 1, 1, 5000, '2024-09-01'),
(15, 2, 2, 2000, '2024-09-15');



-- 2. Write a sql query to list all events
select * from event;
/* Output
4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
5 Late Ms. Lata Mangeshkar Musical 2021-09-12 20:00:00 320 270 600 concert 3
6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2
*/
```

```
-- 3. Write a sql query to select events with available tickets
select *
from event
where available_seats > 0;
/* Output
4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
5 Late Ms. Lata Mangeshkar Musical 2021-09-12 20:00:00 320 270 600 concert 3
6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2

*/



-- 4. Write a sql query to select events name partial match with 'cup'
select *
from event
where event_name LIKE '%cup%';
/* Output
No output
*/



-- 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 3500;
/* Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type
venue_id1
 7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2

*/



-- 6. Write a sql query to retrieve events with dates filling with a specific range
select *
from event
where event_date between '2022-12-11' and '2024-07-09';
/* Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type
venue_id1
 4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
 6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
 7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2

*/
```

```
-- 7. Write a sql query to retrieve events with avaialable tickets that also have "Concert" in their name
select *
from event
where available_seats>0 and
 event_type like '%concert%';
/* Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type
venue_id1
 5 Late Ms. Lata Mangeshkar Musical 2021-09-12 20:00:00 320 270 600 concert 3
*/



-- 8. Write a sql query to retrieve users in batches of 5, starting from the 6th user;
select *
from customer
order by customer_id
limit 5,5;
/* Output
 customer_id customer_name email phone_number
 6 severus snape sev@gmail.com 56556
*/



-- 9. Write a sql query to retrieve booking details contains booked no. of tickets more than 4
select *
from event_has_customer ec
where ec.num_tickets>4;
/* Output
 event_id customer_id num_tickets total_cost booking_date
 5 3 5 18000 2024-04-10
 6 5 10 34000 2024-04-15
*/



-- 10. Write a sql query to retrieve customer information whose phone number end with '000'
select *
from customer
where phone_number like '%000';
/* Output
No output
*/



-- 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;
/* Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type
venue_id1
 6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
 7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2
 4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1

*/




-- 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%';
/* Output
 4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
 5 Late Ms. Lata Mangeshkar Musical 2021-09-12 20:00:00 320 270 600 concert 3
 6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
 7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2
*/




-- Task 3 : Aggregate Functions , Having , Order By , Group By and Joins :
-- 1. Write a sql query to list all events and their average ticket prices
select event_name,avg(ticket_price) as average_ticket_price
from event
group by event_name;
/* Output
 event_name average_ticket_price
 CSK vs RCB 3600.0000
 CSK vs RR 3400.0000
 Late Ms. Lata Mangeshkar Musical 600.0000
 MI vs KKR 8000.0000
*/




-- 2. Write a sql query to calculate the total revenue genrated by events
select sum(total_seats * ticket_price) as total_revenue
from event;
/* Output
 total_revenue
 385192000
```

```
*/



-- 3. Write a sql query to find the event with highest ticket sales
use ticket_booking_system;
select event_name,MAX((total_seats  - available_seats) *  ticket_price) as total_sales
from event
group by event_name
order by total_sales DESC
limit 0,1;
/* Output
 total_revenue
 385192000
*/




-- 4. Write a sql query to calculate total number of tickets sold for each event
select event_name, total_seats  - available_seats as total_tickets_sold
from event
group by event_name;
/* Output
 event_name total_tickets_sold
 CSK vs RCB 22997
 CSK vs RR 22990
 Late Ms. Lata Mangeshkar Musical 50
 MI vs KKR 27900
*/




-- 5. Write a SQL query to Find Events with No Ticket Sales.
select *
from event
where event_id NOT IN (select e.event_id
from event e, event_has_customer b where e.event_id = b.event_id);
/* Output
No Output
*/




-- 6. Write a SQL query to Find Events with No Ticket Sales.
select customer_id, SUM(num_tickets) AS total_tickets_booked
from event_has_customer
group by customer_id
order by total_tickets_booked DESC
limit 1;
/* Output
```

```
 customer_id total_tickets_booked
 5 10
*/



-- 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_id1 = v.venue_id
group by v.venue_name;
/* Output
 venue_name average_ticket_price
 chennai 3500.0000
 mumbai 8000.0000
 pondicherry  600.0000
*/



-- 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 event e
join event_has_customer b ON e.event_id = b.event_id
group by e.event_type;
/* Output
 event_type total_tickets_sold
 concert 8
 sports 19
*/



-- 11 .Write a sql query to list users who have booked tickets for multiple events
select customer_id
from event_has_customer
group by customer_id
having count(distinct event_id) > 1;
/* Output
 customer_id
 1
*/



-- 12. Write a sql query to calculate the total revenue generated by events for each user
select b.customer_id, SUM(b.num_tickets * e.ticket_price) AS total_revenue
from event_has_customer b
left join event e ON b.event_id = e.event_id
```

```
group by b.customer_id;
/* Output
 customer_id total_revenue
 1 17800
 2 13600
 3 3000
 4 24000
 5 36000
*/



-- 13. Write a sql query to calculate the average ticket price for events in each category and venue
select  v.venue_name, AVG(e.ticket_price) AS average_ticket_price
from event e
join venue v ON e.venue_id1 = v.venue_id
group by v.venue_name;
/* Output
 venue_name average_ticket_price
 chennai 3500.0000
 mumbai 8000.0000
 pondicherry  600.0000
*/



-- 14. Write a sql query to list users and the total number of tickets they have purchased in the last 30 days
select customer_id, SUM(num_tickets) AS total_tickets_purchased
from event_has_customer
where booking_date >= '2024-0205'
group by customer_id;
/* Output
 customer_id total_tickets_purchased
 1 3
 2 4
 3 5
 5 10
*/



/*
Tasks 4: Subquery and its types
1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery.
2. Find Events with More Than 50% of Tickets Sold using subquery.
3. Calculate the Total Number of Tickets Sold for Each Event.
4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.
5. List Events with No Ticket Sales Using a NOT IN Subquery.
6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM
```

Clause.
7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.
8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.
9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.
10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.
11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT.
12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
*

```
-- Task 4
-- 1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
/*
projection: ticket price of event
criteria: venue
*/
select v.venue_name, AVG(e.ticket_price) as Average_Ticket_price
from venue v JOIN event e ON v.venue_id=e.venue_id1
group by v.venue_name;
/* Output
venue_name, Average_Ticket_price
# venue_name Average_Ticket_price
chennai 3500.0000
mumbai 8000.0000
pondicherry  600.0000

*/




 -- 2. Find Events with More Than 50% of Tickets Sold using subquery.
select *
from event
where (total_seats-available_seats)>(total_seats/2);
/* Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type
venue_id1
 4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
 6 CSK vs RCB 2024-04-11 19:30:00 23000 3 3600 sports 2
 7 CSK vs RR 2024-04-19 19:30:00 23000 10 3400 sports 2

*/




-- 3. Calculate the Total Number of Tickets Sold for Each Event.
select event_name,sum(total_seats-available_seats)
from event
group by event_name;
```

```
/* Output
 event_name sum(total_seats-available_seats)
 CSK vs RCB 22997
 CSK vs RR 22990
 Late Ms. Lata Mangeshkar Musical 50
 MI vs KKR 27900
*/



-- 4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.
select *
from customer
where customer_id NOT IN (select distinct c.customer_id
from customer c JOIN event_has_customer b ON c.customer_id = b.customer_id);
/* Output
 customer_id customer_name email phone_number
 6 severus snape sev@gmail.com 56556

*/



-- 5.
select *
from event
where event_id NOT IN (
    select distinct event_id
    from event_has_customer
);
/*
Output
No output
*/



-- 6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM
Clause.
select event_type,sum(num_tickets) as total_tickets_sold
from event_has_customer
join event on event_has_customer.event_id = event.event_id
group by event_type;
/* Output
 event_type total_tickets_sold
 sports 19
 concert 8
*/
```

```
-- 7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.
select *
from event
where ticket_price > (
	select avg(ticket_price)
	from event);

/*Output
 event_id event_name event_date event_time total_seats available_seats ticket_price event_type venue_id1
 4 MI vs KKR 2024-05-01 15:30:00 28000 100 8000 sports 1
*/



-- 8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.
select c.customer_name,sum(ec.num_tickets * e.ticket_price) as total_revenue
from customer c
join event_has_customer ec on c.customer_id = ec.customer_id
join event e on ec.event_id = e.event_id
group by c.customer_name;
/* Output
 customer_name total_revenue
 draco malfoy 24000
 ginni weasley 36000
 harry potter 17800
 hermione granger 3000
 ronald weasley 13600
*/



-- 9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.
select distinct c.customer_name
from customer c
join event_has_customer ec on c.customer_id = ec.customer_id
join event e on ec.event_id = e.event_id
where e.venue_id1 = (
	select venue_id
	from venue
	where venue_name = 'mumbai');
/* Output
 customer_name
 harry potter
 draco malfoy
*/
/* Output
```

```
*/

-- 10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.
select e.event_type,sum(ec.num_tickets) as total_tickets_sold
from event_has_customer ec
join event e on ec.event_id = e.event_id
group by e.event_type;
/* Output
event_type total_tickets_sold
concert 8
sports 19
*/

-- 12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
select v.venue_name,avg(e.ticket_price) as average_ticket_price
from venue v
join event e on v.venue_id = e.venue_id1
group by v.venue_name;

/* Output
venue_name average_ticket_price
chennai 3500.0000
mumbai 8000.0000
pondicherry  600.0000
*/
```