TICKET BOOKING SYSTEM

```
create database Ticket_Booking_System;

use Ticket_Booking_System;

create table venue(venueid int not null primary key,venuename varchar(50),address varchar(50));

create table eventtab(eventid int not null primary key,eventname varchar(50),eventdate date,eventtime time,venueid int not null,totseats int not null,availseats int,ticketprice decimal,eventtype varchar(50),bookingid int not null);

create table customer(customerid int not null primary key,customername varchar(50),email varchar(50),phone int(10),bookingid int not null);

create table booking(bookingid int not null primary key,customerid int not null ,eventid int not null,numtickets int,totcost int,bookingdate date);
```

3.DATA MANIPULATION LANGUAGE

```
INSERT INTO venue (venueid, venuename, address)

VALUES

   (1, 'Convention Center', '123 Main Street'),

   (2, 'City Hall Auditorium', '456 Oak Avenue'),

   (3, 'Grand Ballroom', '789 Pine Lane'),

   (4, 'Exhibition Hall', '101 Elm Street'),

   (5, 'Community Center', '202 Maple Drive'),

   (6, 'Event Plaza', '303 Cedar Court'),

   (7, 'Civic Center', '404 Birch Boulevard'),

   (8, 'Theater Royale', '505 Spruce Road'),

   (9, 'Harmony Hall', '606 Willow Way'),

   (10, 'Gala Gardens', '707 Juniper Junction');
```

```
INSERT INTO eventtab (eventid, eventname, eventdate, eventtime, venueid, totseats,
availseats, ticketprice, bookingid)
VALUES
   (1, 'Iron man', '2023-12-15', '10:00:00', 1, 500, 500, 50.00, 101),
   (2, 'Music Festival', '2023-11-25', '18:30:00', 3, 1000, 800, 75.00, 102),
   (3, 'Comedy Night', '2024-01-08', '20:00:00', 5, 300, 300, 30.00, 103),
   (4, 'Sports Gala', '2023-12-05', '15:00:00', 2, 800, 600, 40.00, 104),
   (5, 'Movie Mela', '2023-11-20', '12:00:00', 7, 200, 200, 25.00, 105),
   (6, 'Music Show', '2024-02-01', '19:00:00', 8, 400, 350, 60.00, 106),
   (7, 'Sport Summit', '2023-12-10', '09:30:00', 4, 600, 550, 55.00, 107),
   (8, 'Music Performance', '2023-11-30', '21:00:00', 6, 250, 200, 35.00, 108),
   (9, 'Movie Fair', '2024-02-15', '14:00:00', 10, 150, 150, 20.00, 109),
   (10, 'Film Screening', '2023-12-20', '17:45:00', 9, 180, 150, 25.00, 110);
   INSERT INTO booking(bookingid, customerid, eventid, numtickets, totcost, bookingdate)
VALUES
   (101, 201, 1, 2, 100, '2023-12-01'),
   (102, 202, 2, 4, 300, '2023-11-20'),
   (103, 203, 3, 1, 30, '2024-01-05'),
   (104, 204, 4, 3, 120, '2023-12-02'),
   (105, 205, 5, 5, 125, '2023-11-15'),
   (106, 206, 6, 2, 120, '2024-01-20'),
   (107, 207, 7, 1, 55, '2023-12-05'),
   (108, 208, 8, 3, 105, '2023-11-25'),
   (109, 209, 9, 4, 80, '2024-02-10'),
   (110, 210, 10, 2, 50, '2023-12-10');



  INSERT INTO customer (customerid, customername, email, phone, bookingid)
VALUES
   (201, 'Megan Anderson', 'megan.anderson@email.com',123456, 101),
   (202, 'Ryan Johnson', 'ryan.johnson@email.com',908765, 102),
   (203, 'Emily Davis', 'emily.davis@email.com',342589, 103),
   (204, 'Chris Smith', 'chris.smith@email.com',826457, 104),
   (205, 'Ava White', 'ava.white@email.com',749282, 105),
   (206, 'Daniel Brown', 'daniel.brown@email.com',343529, 106),
   (207, 'Sophie Lee', 'sophie.lee@email.com',484646, 107),
   (208, 'Jack Kim', 'jack.kim@email.com',520548, 108),
   (209, 'Emma Chen', 'emma.chen@email.com',517698, 109),
   (210, 'Noah Wang', 'noah.wang@email.com',987653, 110);


create table customer(customerid int not null primary key,customername varchar(50),email varchar(50),phone int,bookingid int not null);
alter table eventtab add foreign key(venueid)references venue(venueid);
alter table eventtab add foreign key(bookingid) references booking(bookingid);
alter table customer add foreign key(bookingid) references booking(bookingid);
alter table booking add  foreign key(customerid) references customer(customerid);
alter table booking add  foreign key(eventid) references eventtab(eventid);
```

2. select *from eventtab;

3. select*from eventtab where totseats>availseats;

4. select *from eventtab where eventname like '%cup%';

5. select *from eventtab where ticketprice between 1000 and 2000;

6. select*from eventtab where month(eventdate)>10 and month(eventdate)<12;

7. select * from eventtab where totseats>availseats and eventname like '%concert%';

8. select * from customer order by customerid limit 5 offset 5;

9. select *from booking where numtickets>4;

10. select * from customer where phone like '%000';

11. select * from eventtab where totseats>15000 order by totseats desc;

12. select * from eventtab where eventname not like 'x%' or 'y%' or 'z%';

4.AGGREGATE FUNCTIONS , GROUP BY, JOINS

1. select eventid,eventname,avg(ticketprice) from eventtab group by eventid ;

2. select sum(totcost) as total_revenue from booking;

3. select e.eventname,b.numtickets from eventtab e inner join booking b on e.eventid=b.eventid order by b.numtickets desc limit 1 ;

4. select e.eventid,e.eventname,sum(numtickets) as total_tickets from eventtab e inner join booking b on e.bookingid=b.bookingid group by e.eventid;

5. select e.eventid ,e.eventname ,b.numtickets from eventtab e inner join booking b on e.bookingid=b.bookingid where b.numtickets is null ;

6. select c.customerid,c.customername,b.numtickets from customer c join booking b on c.customerid=b.customerid order by b.numtickets desc limit 1;

7. select sum(numtickets)as ticket_sold_each_month from eventtab e inner join booking b on e.eventid=b.eventid where b.numtickets is not null group by month(e.eventdate);

8. select avg(ticketprice) from eventtab e inner join venue v on e.venueid=v.venueid group by e.venueid;

9. select e.eventtype,sum(numtickets) as tickets_sold_for_each_eventtype from booking b inner join eventtab e on b.eventid=e.eventid group by e.eventtype;

10. select year(e.eventdate),sum(totcost)as revenue_per_year from booking b inner join eventtab e on b.eventid=e.eventid group by year(e.eventdate);

11. select c.customerid,c.customername from customer c inner join booking b on c.bookingid=b.bookingid group by c.customerid having count(distinct b.eventid) > 1;

12. select b.customerid,sum(totcost) as total_revenue from booking b inner join eventtab e on b.eventid=e.eventid group by b.customerid;

13. select avg(ticketprice) from eventtab e inner join venue v on e.venueid=v.venueid group by e.eventtype and v.venueid;

14. select c.customerid,c.customername, sum(numtickets) as total_tickets from customer c inner join booking b on c.customerid=b.customerid inner join eventtab e on b.eventid=e.eventid where month(e.eventdate)=11 group by c.customerid;

5.SUBQUERY

```
1. select venueid, (select avg(ticketprice)
            from eventtab
            where eventtab.venueid = venue.venueid
              and eventtab.ticketprice>=1) as average_ticket_price
from venue;
```

2. SELECT eventid, eventname

```
FROM eventtab
WHERE (
   SELECT COUNT(*)
   FROM booking
   WHERE booking.eventid = eventtab.eventid
) > 0.5 * totseats;
3. select eventid,sum(numtickets) as total_tickets from booking where ( select eventid from eventtab where eventtab.eventid=booking.eventid)group by booking.eventid;
4. select customerid,customername from customer  where not exists ( select numtickets from booking where booking.customerid=customer.customerid);
5. SELECT eventid, eventname
FROM eventtab
WHERE eventid NOT IN (
   SELECT DISTINCT eventid
   FROM booking
);


7. select eventid,eventname from eventtab where ticketprice >=any(select avg(ticketprice) from eventtab);
8. SELECT customerid,
      (
         SELECT COALESCE(SUM(totcost), 0)
         FROM booking b
         WHERE b.customerid = c.customerid
```

```
     ) AS total_revenue
FROM customer c;
9. SELECT DISTINCT c.customerid, c.customername
FROM customer c
INNER JOIN booking b ON c.customerid = b.customerid
WHERE b.eventid IN (
   SELECT eventid
   FROM eventtab
   WHERE venueid = '3'
);
11. SELECT DISTINCT c.customerid, c.customername
FROM customer c
INNER JOIN booking b ON c.customerid = b.customerid
INNER JOIN eventtab e ON b.eventid = e.eventid
WHERE DATE_FORMAT(e.eventdate, '%y-%m-%d') = '11';


12. select venueid, (select avg(ticketprice)
             from eventtab
             where eventtab.venueid = venue.venueid
               and eventtab.ticketprice>=1) as average_ticket_price
from venue;
```