Assignment

Ticket Booking System(SQL)

Name=Aman Gurung

Task 1.

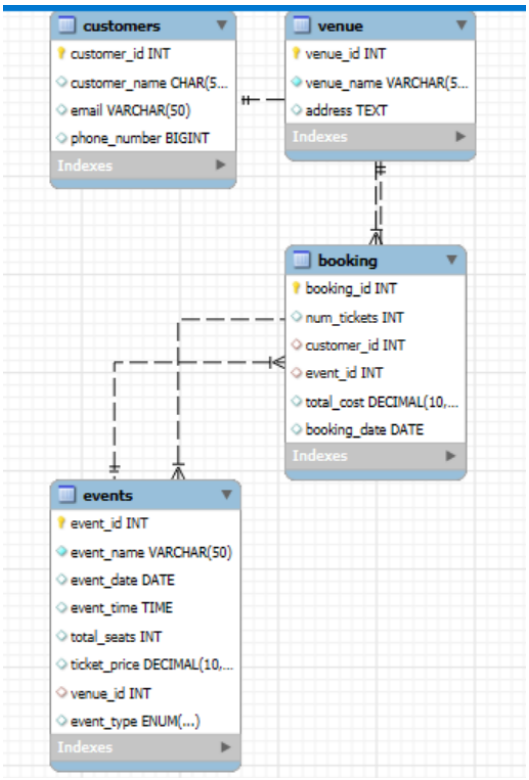
```
-- 1. Create the database named "TicketBookingSystem"
create database TicketBookingSystem;
use Ticketbookingsystem
```


```
/* 2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and
relationships.
• Venu
• Event
• Customers
• Booking */
```

```
create table venue (
venue_id int primary key,
venue_name varchar(50) not null,
address text
);

create table events (
event_id int primary key,
event_name varchar(50) not null,
event_date date,
event_time time,
total_seats int,
```

```
ticket_price decimal(10,2),
venue_id int,
event_type enum('Movie', 'Sports', 'Concert'),
foreign key (venue_id) references venue(venue_id));

create table customers (
customer_id int primary key,
customer_name char(50),
email varchar(50) unique,
phone_number bigint
);

create table booking (
booking_id int primary key,
num_tickets int,
customer_id int,
event_id int,
total_cost decimal(10,2),
booking_date date,
foreign key (event_id) references events(event_id),
foreign key (customer_id) references customers(customer_id));
```

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



```sql
-- 4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.
-- asnwer included in answer 2
```

```sql
create table venue (
venue_id int primary key,
venue_name varchar(50) not null,
address text
);
-- venue_id is primary key
```

```sql
create table events (
event_id int primary key,
event_name varchar(50) not null,
event_date date,
event_time time,
total_seats int,
ticket_price decimal(10,2),
venue_id int,
event_type enum('Movie', 'Sports', 'Concert'),
foreign key (venue_id) references venue(venue_id));
-- event_id is primary key and venue_id is the foreign key
```

```sql
create table customers (
customer_id int primary key,
customer_name char(50),
email varchar(50) unique,
phone_number bigint
);
-- customer_id is primmary key
```

```sql
create table booking (
booking_id int primary key,
num_tickets int,
customer_id int,
event_id int,
total_cost decimal(10,2),
booking_date date,
foreign key (event_id) references events(event_id),
foreign key (customer_id) references customers(customer_id));
-- has booking_id as primary key and event_id and customer_id as foreign key
```

Task 2.

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

insert into venue values
(101, 'Stadium A', 'Civil Lines'),
(102, 'Concert Hall B', 'Manish Nagar'),
(103, 'Arena C', 'IT Park'),
(104, 'Ground D', 'Kalmeshwar Road'),
(105, 'Theater E', 'Shankar Nagar'),
(106, 'Park F', 'Walkers Street'),
(107, 'Stadium G', 'Varora'),
(108, 'Club H', 'JP Street'),
(109, 'Expo Center I', 'Mankapur'),
(110, 'Auditorium J', 'Dragonn Palace');

insert into events values

(1, 'Cricket Cup', '2025-04-15', '18:00:00', 20000, 1500.00, 101, 'Sports'),
(2, 'Rock Concert', '2025-05-01', '20:00:00', 12000, 2500.00, 102, 'Concert'),
(3, 'Football Local Cup', '2025-04-20', '19:30:00', 18000, 1800.00, 103, 'Sports'),
(4, 'Drama Night', '2025-04-25', '21:00:00', 3000, 800.00, 104, 'Movie'),
(5, 'Jazz Concert', '2025-04-30', '20:30:00', 5000, 2200.00, 105, 'Concert'),
(6, 'Movie Gala', '2025-05-03', '19:00:00', 4000, 1000.00, 106, 'Movie'),
(7, 'Boxing Show', '2025-05-05', '18:30:00', 6000, 1300.00, 107, 'Sports'),
(8, 'Music Fest', '2025-05-07', '20:00:00', 10000, 2100.00, 108, 'Concert'),
(9, 'Comedy Night', '2025-05-09', '19:00:00', 3500, 900.00, 109, 'Movie'),
(10, 'Tech Expo', '2025-05-11', '11:00:00', 15000, 2000.00, 110, 'Movie');

insert into customers values
(1, 'Aman', 'aman52@google.com', 987650000),
(2, 'Daksha', 'daksha322@google.com', 987651091),
(3, 'Vedant', 'vedant222@yahoo.com', 987652112),
(4, 'Sanjit', 'Sanjit786@yahoo.com', 987653033),
(5, 'Megha', 'megha21@google.com', 987654444),
(6, 'Satyam', 'satyam331@bing.com', 987655235),
(7, 'Gracey', 'gracey21@google.com', 987656696),
(8, 'Praneet', 'praneet2020@bing.com', 987657557),
(9, 'Vaibhav', 'vaibhav5012@google.com', 987656888),
(10, 'Shreya', 'shreya212@google.com', 9876590010);

insert into booking values
(1, 5, 1, 1, 7500.00, '2025-04-01'),
(2, 2, 2, 2, 5000.00, '2025-04-02'),
(3, 6, 3, 3, 10800.00, '2025-04-03'),
(4, 3, 4, 4, 2400.00, '2025-04-04'),
(5, 1, 5, 5, 2200.00, '2025-04-05'),
(6, 4, 6, 6, 4000.00, '2025-04-06'),
(7, 7, 7, 7, 9100.00, '2025-04-07'),
(8, 2, 8, 8, 4200.00, '2025-04-08'),
(9, 5, 9, 9, 4500.00, '2025-04-09'),
(10, 9, 10, 10, 18000.00, '2025-04-10');

update booking set booking_date = '2025-01-15' where booking_id = 1;
update booking set booking_date = '2025-02-20' where booking_id = 2;
update booking set booking_date = '2025-03-10' where booking_id = 3;
update booking set booking_date = '2024-12-25' where booking_id = 4;
update booking set booking_date = '2024-11-05' where booking_id = 5;
update booking set booking_date = '2025-04-01' where booking_id = 6;
update booking set booking_date = '2025-04-05' where booking_id = 7;
update booking set booking_date = '2025-03-25' where booking_id = 8;
update booking set booking_date = '2025-02-28' where booking_id = 9;
update booking set booking_date = '2025-01-02' where booking_id = 10;

```sql
-- 2. Write a SQL query to list all Events.
select * from events;
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type
1	Cricket Cup	2025-04-15	18:00:00	20000	1500.00	101	Sports
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert
3	Football Local Cup	2025-04-20	19:30:00	18000	1800.00	103	Sports
4	Drama Night	2025-04-25	21:00:00	3000	800.00	104	Movie
5	Jazz Concert	2025-04-30	20:30:00	5000	2200.00	105	Concert
6	Movie Gala	2025-05-03	19:00:00	4000	1000.00	106	Movie
7	Boxing Show	2025-05-05	18:30:00	6000	1300.00	107	Sports
8	Music Fest	2025-05-07	20:00:00	10000	6000	108	Concert
9	Comedy Night	2025-05-09	19:00:00	3500	900.00	109	Movie
10	Tech Expo	2025-05-11	11:00:00	15000	2000.00	110	Movie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 3. Write a SQL query to select events with available tickets.
select *,total_seats as available_tickets from events where total_seats >0;
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type	available_tickets
1	Cricket Cup	2025-04-15	18:00:00	20000	1500.00	101	Sports	20000
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert	12000
3	Football Local Cup	2025-04-20	19:30:00	18000	1800.00	103	Sports	18000
4	Drama Night	2025-04-25	21:00:00	3000	800.00	104	Movie	3000
5	Jazz Concert	2025-04-30	20:30:00	5000	2200.00	105	Concert	5000
6	Movie Gala	2025-05-03	19:00:00	4000	1000.00	106	Movie	4000
7	Boxing Show	2025-05-05	18:30:00	6000	1300.00	107	Sports	6000
8	Music Fest	2025-05-07	20:00:00	10000	2100.00	108	Concert	10000
9	Comedy Night	2025-05-09	19:00:00	3500	900.00	109	Movie	3500
10	Tech Expo	2025-05-11	11:00:00	15000	2000.00	110	Movie	15000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 4. Write a SQL query to select events name partial match with ‘cup’.
select event_name from events where event_name like '%cup%';
```

event_name
Cricket Cup
Football Local Cup

```sql
-- 5. Write a SQL query to select events with ticket price range is between 1000 to 2500.
select * from events where ticket_price>=1000 and ticket_price<=2500;
-- or
select * from events where ticket_price between 1000 and 2500;
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type
1	Cricket Cup	2025-04-15	18:00:00	20000	1500.00	101	Sports
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert
3	Football Local Cup	2025-04-20	19:30:00	18000	1800.00	103	Sports
5	Jazz Concert	2025-04-30	20:30:00	5000	2200.00	105	Concert
6	Movie Gala	2025-05-03	19:00:00	4000	1000.00	106	Movie
7	Boxing Show	2025-05-05	18:30:00	6000	1300.00	107	Sports
8	Music Fest	2025-05-07	20:00:00	10000	2100.00	108	Concert
10	Tech Expo	2025-05-11	11:00:00	15000	2000.00	110	Movie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 6. Write a SQL query to retrieve events with dates falling within a specific range.
select * from events where event_date between '2025-05-01' and '2025-05-11';
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert
6	Movie Gala	2025-05-03	19:00:00	4000	1000.00	106	Movie
7	Boxing Show	2025-05-05	18:30:00	6000	1300.00	107	Sports
8	Music Fest	2025-05-07	20:00:00	10000	2100.00	108	Concert
9	Comedy Night	2025-05-09	19:00:00	3500	900.00	109	Movie
10	Tech Expo	2025-05-11	11:00:00	15000	2000.00	110	Movie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 7. Write a SQL query to retrieve events with available tickets that also have "Concert" in their name.
select *,total_seats as available_tickets  from events
where total_seats >0 and  event_name like '%concert%' ;
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type	available_tickets
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert	12000
5	Jazz Concert	2025-04-30	20:30:00	5000	2200.00	105	Concert	5000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.
select * from customers
limit 5 offset 5
```

customer_id	customer_name	email	phone_number
6	Satyam	satyam331@bing.com	987655235
7	Gracey	gracey21@google.com	987656696
8	Praneet	praneet2020@bing.com	987657557
9	Vaibhav	vaibhav5012@google.com	987656888
10	Shreya	shreya212@google.com	9876590010
NULL	NULL	NULL	NULL


```sql
-- 8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.
select * from customers
limit 5 offset 5
```

customer_id	customer_name	email	phone_number
6	Satyam	satyam331@bing.com	987655235
7	Gracey	gracey21@google.com	987656696
8	Praneet	praneet2020@bing.com	987657557
9	Vaibhav	vaibhav5012@google.com	987656888
10	Shreya	shreya212@google.com	9876590010
NULL	NULL	NULL	NULL

```sql
-- 9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.
select * from booking;
select * from booking where num_tickets > 4;
```

booking_id	num_tickets	customer_id	event_id	total_cost	booking_date
1	5	1	1	7500.00	2025-01-15
3	6	3	3	10800.00	2025-03-10
7	7	7	7	9100.00	2025-04-05
9	5	9	9	4500.00	2025-02-28
10	9	10	10	18000.00	2025-01-02
NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 10. Write a SQL query to retrieve customer information whose phone number end with '000'
select * from customers
where phone_number Like '%000';
```

customer_id	customer_name	email	phone_number
1	Aman	aman52@google.com	987650000
NULL	NULL	NULL	NULL

```sql
-- 11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.
select * from events
where total_seats > 15000;
```

event_id	event_name	event_date	event_time	total_seats	ticket_price	venue_id	event_type
1	Cricket Cup	2025-04-15	18:00:00	20000	1500.00	101	Sports
3	Football Local Cup	2025-04-20	19:30:00	18000	1800.00	103	Sports
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- 12. Write a SQL query to select events name not start with ‘x’, ‘y’, ‘z’
select * from events
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	total_seats	ticket_price	venue_id	event_type
1	Cricket Cup	2025-04-15	18:00:00	20000	1500.00	101	Sports
2	Rock Concert	2025-05-01	20:00:00	12000	2500.00	102	Concert
3	Football Local Cup	2025-04-20	19:30:00	18000	1800.00	103	Sports
4	Drama Night	2025-04-25	21:00:00	3000	800.00	104	Movie
5	Jazz Concert	2025-04-30	20:30:00	5000	2200.00	105	Concert
6	Movie Gala	2025-05-03	19:00:00	4000	1000.00	106	Movie
7	Boxing Show	2025-05-05	18:30:00	6000	1300.00	107	Sports
8	Music Fest	2025-05-07	20:00:00	10000	2100.00	108	Concert
9	Comedy Night	2025-05-09	19:00:00	3500	900.00	109	Movie
10	Tech Expo	2025-05-11	11:00:00	15000	2000.00	110	Movie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Task 3.

```sql
-- 1. Write a SQL query to List Events and Their Average Ticket Prices.
select event_name,avg(ticket_price) as avg_ticketprice
from events
group by event_name
```

event_name	avg_ticketprice
Cricket Cup	1500.000000
Rock Concert	2500.000000
Football Local Cup	1800.000000
Drama Night	800.000000
Jazz Concert	2200.000000
Movie Gala	1000.000000
Boxing Show	1300.000000
Music Fest	2100.000000
Comedy Night	900.000000
Tech Expo	2000.000000

```sql
-- 2. Write a SQL query to Calculate the Total Revenue Generated by Events.
select event_id,sum(total_cost) as totalrevenue
from booking
group by event_id
```

	event_id	totalrevenue
▶	1	7500.00
	2	5000.00
	3	10800.00
	4	2400.00
	5	2200.00
	6	4000.00
	7	9100.00
	8	4200.00
	9	4500.00
	10	18000.00

```sql
-- 3. Write a SQL query to find the event with the highest ticket sales.
select e.event_name,max(num_tickets) as  most_tickets
from events e
join booking b on e.event_id = b.event_id
group by event_name
order by most_tickets desc
limit 1
```

	event_name	most_tickets
▶	Tech Expo	9

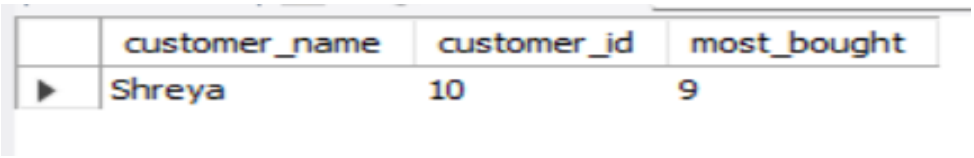
```sql
-- 4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event.
select e.event_name,sum(num_tickets) as  total_sold
from events e,booking b
where e.event_id = b.event_id
group by e.event_id
```

	event_name	total_sold
▶	Cricket Cup	5
	Rock Concert	2
	Football Local Cup	6
	Drama Night	3
	Jazz Concert	1
	Movie Gala	4
	Boxing Show	7
	Music Fest	2
	Comedy Night	5
	Tech Expo	9

```sql
-- 5. Write a SQL query to Find Events with No Ticket Sales.
select event_id, event_name
from events
where event_id not in (select distinct event_id from booking);
```

	event_id	event_name
*	NULL	NULL

```sql
-- 6. Write a SQL query to Find the User Who Has Booked the Most Tickets.
select c.customer_name,b.customer_id ,max(num_tickets) as most_bought
from booking b
join customers c on c.customer_id=b.customer_id
group by customer_id
order by most_bought desc
limit 1
```



	customer_name	customer_id	most_bought
▶	Shreya	10	9

```sql
-- 7. Write a SQL query to List Events and the total number of tickets sold for each month.
select e.event_name,month(b.booking_date) as booking_month,sum(b.num_tickets)as total_tickets
from events e
join booking b on e.event_id = b.event_id
group by e.event_name,booking_month
order by booking_month
```

event_name	booking_month	total_tickets
Cricket Cup	1	5
Tech Expo	1	9
Rock Concert	2	2
Comedy Night	2	5
Football Local Cup	3	6
Music Fest	3	2
Movie Gala	4	4
Boxing Show	4	7
Jazz Concert	11	1
Drama Night	12	3

```sql
-- 8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue.
 select venue_name,avg(ticket_price) as avgticketprice
 from events e
 join venue v on e.venue_id=v.venue_id
 group by venue_name
 order by avgticketprice
```

venue_name	avgticketprice
Ground D	800.000000
Expo Center I	900.000000
Park F	1000.000000
Stadium G	1300.000000
Stadium A	1500.000000
Arena C	1800.000000
Auditorium J	2000.000000
Club H	2100.000000
Theater E	2200.000000
Concert Hall B	2500.000000

```sql
 -- 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
from events e
join booking b on e.event_id = b.event_id
group by e.event_type;
```

event_type	total_tickets
Sports	18
Concert	5
Movie	21

```sql
-- 10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.
 select year(booking_date) as year,sum(total_cost) as revenue
 from booking
 group by year
```

	year	revenue
▶	2025	63100.00
	2024	4600.00

```sql
-- 11. Write a SQL query to list users who have booked tickets for multiple events.
select c.customer_name,count(distinct event_id) as event_count
from customers c
join booking b on c.customer_id = b.customer_id
group by c.customer_id
having event_count > 1;
```

	customer_name	event_count

```sql
-- 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_revenue
from customers c
join booking b on c.customer_id=b.customer_id
group by c.customer_name,c.customer_id
```

customer_name	total_revenue
Aman	7500.00
Daksha	5000.00
Vedant	10800.00
Sanjit	2400.00
Megha	2200.00
Satyam	4000.00
Gracey	9100.00
Praneet	4200.00
Vaibhav	4500.00
Shreya	18000.00

```sql
-- 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(ticket_price)as avgticketprice
from events e
join venue v on e.venue_id=v.venue_id
group by e.event_type,v.venue_id
```

event_type	venue_name	avgticketprice
Sports	Stadium A	1500.000000
Concert	Concert Hall B	2500.000000
Sports	Arena C	1800.000000
Movie	Ground D	800.000000
Concert	Theater E	2200.000000
Movie	Park F	1000.000000
Sports	Stadium G	1300.000000
Concert	Club H	2100.000000
Movie	Expo Center I	900.000000
Movie	Auditorium J	2000.000000

```sql
/*14.Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30
Days.*/
select c.customer_name,sum(num_tickets)as last30days_tickets
from customers c
join booking b on c.customer_id=b.customer_id
where b.booking_date>=curdate() - interval 30 day
group by c.customer_name
```

customer_name	last30days_tickets
Satyam	4
Gracey	7
Praneet	2

Task 4.

```sql
-- 1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery.
select v.venue_name,(select avg(ticket_price)
                     from events e
                     where e.venue_id=v.venue_id)as avgticketprice
from venue v
```

venue_name	avgticketprice
Stadium A	1500.000000
Concert Hall B	2500.000000
Arena C	1800.000000
Ground D	800.000000
Theater E	2200.000000
Park F	1000.000000
Stadium G	1300.000000
Club H	2100.000000
Expo Center I	900.000000
Auditorium J	2000.000000

```sql
-- 2.Find Events with More Than 50% of Tickets Sold using subquery.
select event_name
from events
where event_id IN (
    select e.event_id
    from events e
    join booking b on e.event_id = b.event_id
    group by e.event_id, e.total_seats
    having sum(b.num_tickets) > 0.5 * e.total_seats
);
```

	event_name

```sql
-- 3.Calculate the Total Number of Tickets Sold for Each Event.
select e.event_name,(select sum(num_tickets)
                     from booking b
                     where e.event_id=b.event_id) as total_ticketssold
from events e
```

	event_name	total_ticketssold
▶	Cricket Cup	5
	Rock Concert	2
	Football Local Cup	6
	Drama Night	3
	Jazz Concert	1
	Movie Gala	4
	Boxing Show	7
	Music Fest	2
	Comedy Night	5
	Tech Expo	9

```sql
-- 4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.
select customer_name
from customers c
where not exists (select * from booking b where b.customer_id=c.customer_id);
-- every customer booked a ticket
```

	customer_name

```sql
-- 5. List Events with No Ticket Sales Using a NOT IN Subquery.
select event_name
from events
where event_id not in ( select distinct event_id
                        from booking);
-- every event has sold some tickets
```

	event_name

```sql
-- 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) as tickets_sold
from (
      select e.event_type,b.num_tickets as total_tickets
      from booking b
      join events e on e.event_id=b.event_id)as  sub
group by event_type;
```

	event_type	tickets_sold
▶	Sports	18
	Concert	5
	Movie	21

```sql
/* 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 events
where ticket_price > (
                    select avg(ticket_price)
                    from events);
```

event_name	ticket_price
Rock Concert	2500.00
Football Local Cup	1800.00
Jazz Concert	2200.00
Music Fest	2100.00
Tech Expo	2000.00

```sql
-- 8.Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.
select c.customer_name,(select sum(total_cost)
                    from booking b
                    where c.customer_id=b.customer_id) as Total_revenue
from customers c
```

customer_name	Total_revenue
Aman	7500.00
Daksha	5000.00
Vedant	10800.00
Sanjit	2400.00
Megha	2200.00
Satyam	4000.00
Gracey	9100.00
Praneet	4200.00
Vaibhav	4500.00
Shreya	18000.00

```sql
-- 9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.
select customer_name
from customers
where customer_id in (
	select b.customer_id
	from booking b
	join events e on e.event_id=b.event_id
	where venue_id=102);
```

	customer_name
▶	Daksha

```sql
-- 10.Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.
select event_type ,sum(total_tickets) as Total_ticketssold
from(select e.event_type,b.num_tickets as total_tickets
	from booking b
	join events e on e.event_id = b.event_id) as sub
group by event_type
```

	event_type	Total_ticketssold
▶	Sports	18
	Concert	5
	Movie	21

```sql
-- 11.Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT.
select c.customer_name,date_format(b.booking_date,'%Y-%m')as  booking_month
from customers c
join booking b on c.customer_id=b.customer_id
```

customer_name	booking_month
Aman	2025-01
Daksha	2025-02
Vedant	2025-03
Sanjit	2024-12
Megha	2024-11
Satyam	2025-04
Gracey	2025-04
Praneet	2025-03
Vaibhav	2025-02
Shreya	2025-01

```sql
-- 12.Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
select v.venue_name,(select avg(e.ticket_price)
                     from events e
                     where v.venue_id=e.venue_id)  as avgticketprice
from venue v
```

venue_name	avgticketprice
Stadium A	1500.000000
Concert Hall B	2500.000000
Arena C	1800.000000
Ground D	800.000000
Theater E	2200.000000
Park F	1000.000000
Stadium G	1300.000000
Club H	2100.000000
Expo Center I	900.000000
Auditorium J	2000.000000