Assignment 1

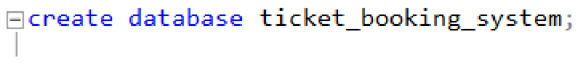
Name – Rohan Mahajan

Project - Ticket Booking System

Tasks 1: Database Design:

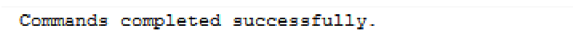
1. Create the database named "TicketBookingSystem"

Query =>



```
create database ticket_booking_system;
```

Result =>

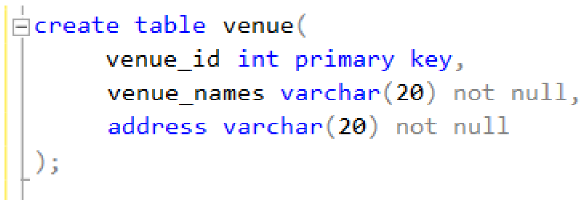


```
Commands completed successfully.
```

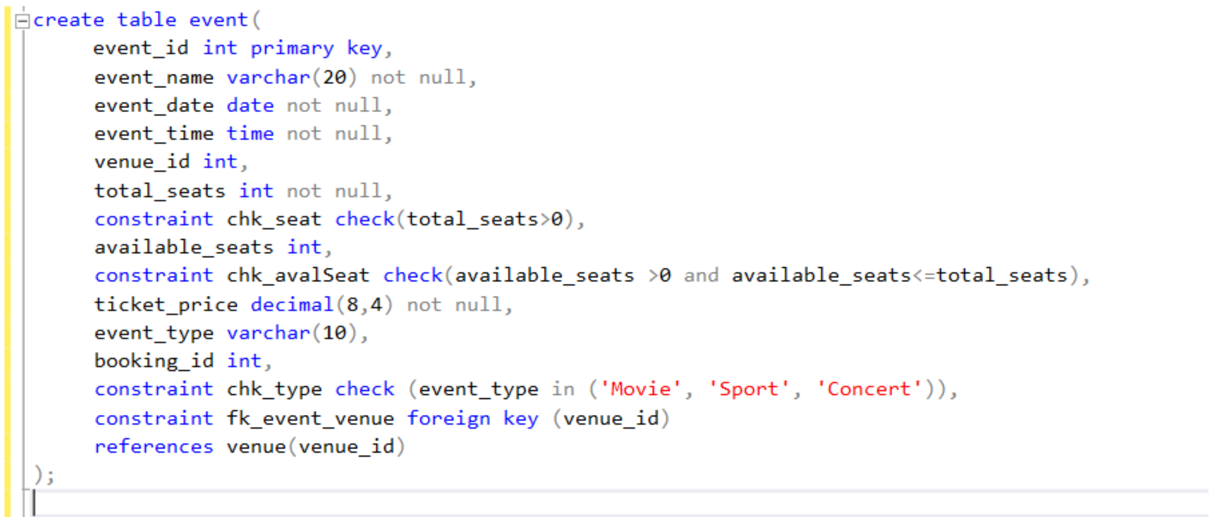
2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

- **Venu**
- **Event**
- **Customers**
- **Booking**

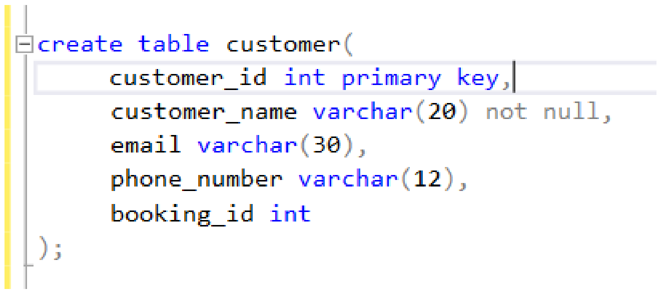
Query =>



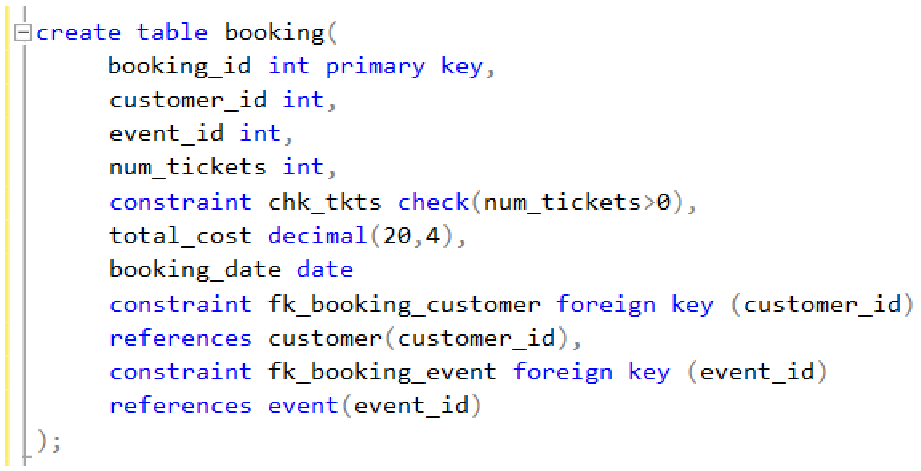
```
create table venue(
	venue_id int primary key,
	venue_names varchar(20) not null,
	address varchar(20) not null
);
```



```sql
create table event(
	event_id int primary key,
	event_name varchar(20) not null,
	event_date date not null,
	event_time time not null,
	venue_id int,
	total_seats int not null,
	constraint chk_seat check(total_seats>0),
	available_seats int,
	constraint chk_avalSeat check(available_seats >0 and available_seats<=total_seats),
	ticket_price decimal(8,4) not null,
	event_type varchar(10),
	booking_id int,
	constraint chk_type check (event_type in ('Movie', 'Sport', 'Concert')),
	constraint fk_event_venue foreign key (venue_id)
	references venue(venue_id)
);
```

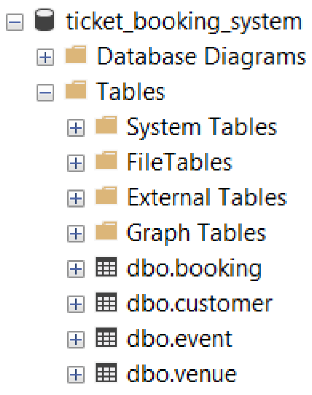


```sql
create table customer(
	customer_id int primary key,
	customer_name varchar(20) not null,
	email varchar(30),
	phone_number varchar(12),
	booking_id int
);
```



```sql
create table booking(
	booking_id int primary key,
	customer_id int,
	event_id int,
	num_tickets int,
	constraint chk_tkts check(num_tickets>0),
	total_cost decimal(20,4),
	booking_date date
	constraint fk_booking_customer foreign key (customer_id)
	references customer(customer_id),
	constraint fk_booking_event foreign key (event_id)
	references event(event_id)
);
```

Result =>

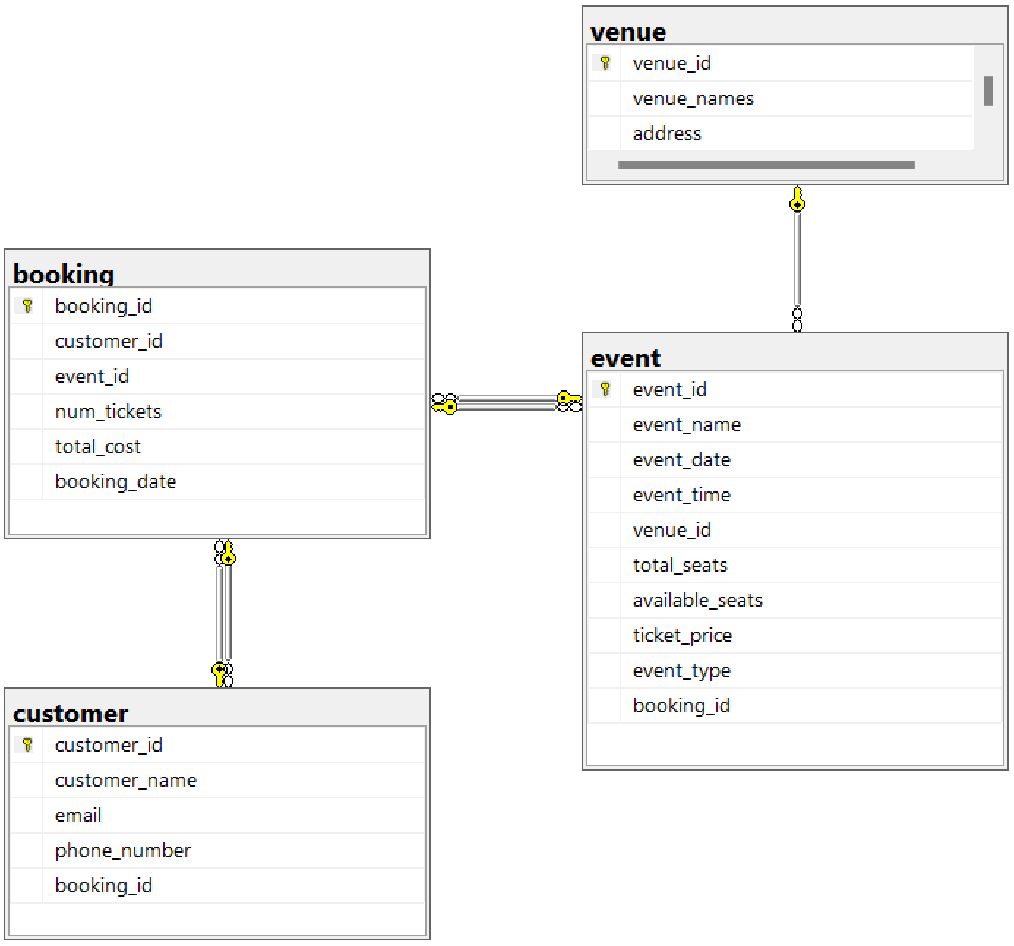


```
⊟ ticket_booking_system
    ⊞ Database Diagrams
    ⊟ Tables
        ⊞ System Tables
        ⊞ FileTables
        ⊞ External Tables
        ⊞ Graph Tables
        ⊞ dbo.booking
        ⊞ dbo.customer
        ⊞ dbo.event
        ⊞ dbo.venue
```

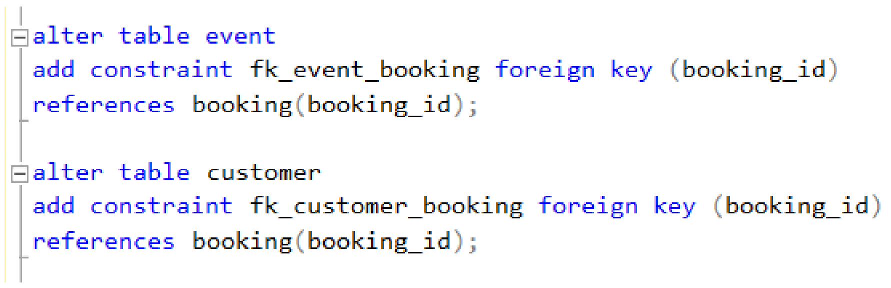
3. Create an ERD (Entity Relationship Diagram) for the database.

Result =>

P.T.O

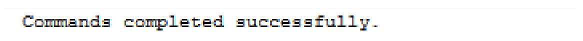


4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.
Query =>



```sql
alter table event
add constraint fk_event_booking foreign key (booking_id)
references booking(booking_id);

alter table customer
add constraint fk_customer_booking foreign key (booking_id)
references booking(booking_id);
```

Result =>

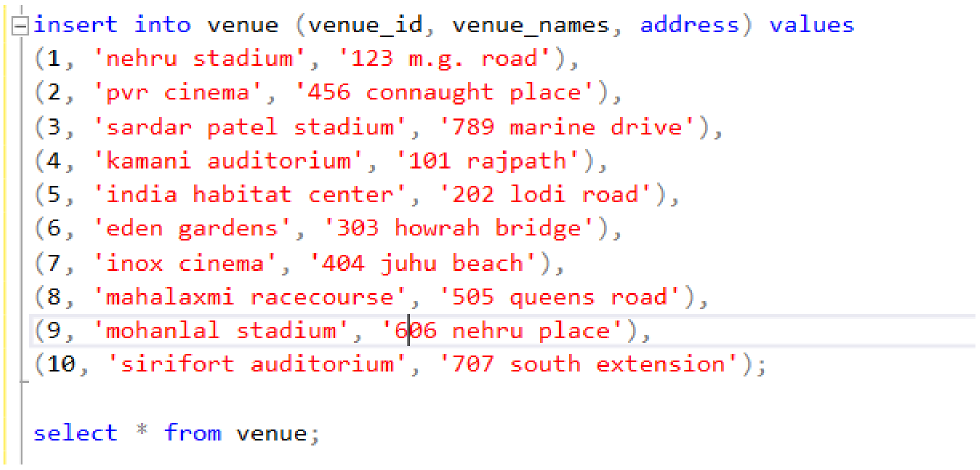


```
Commands completed successfully.
```

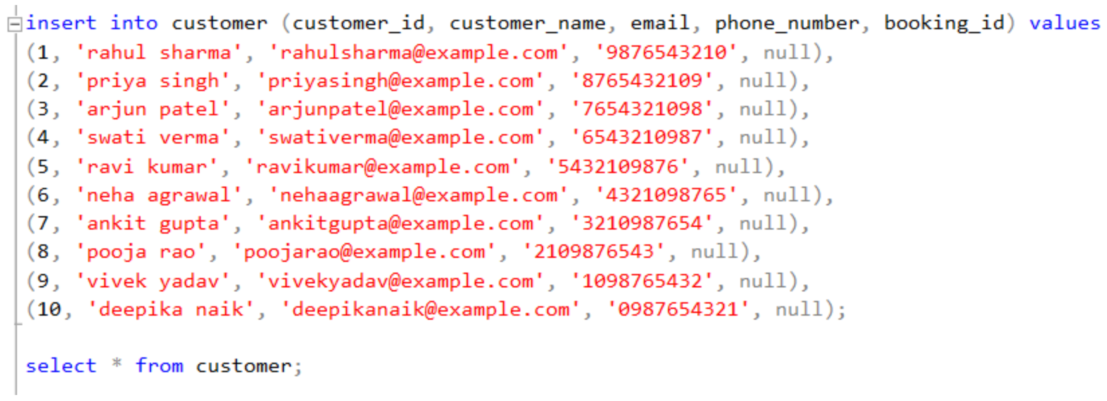
Tasks 2: Select, Where, Between, AND, LIKE:

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

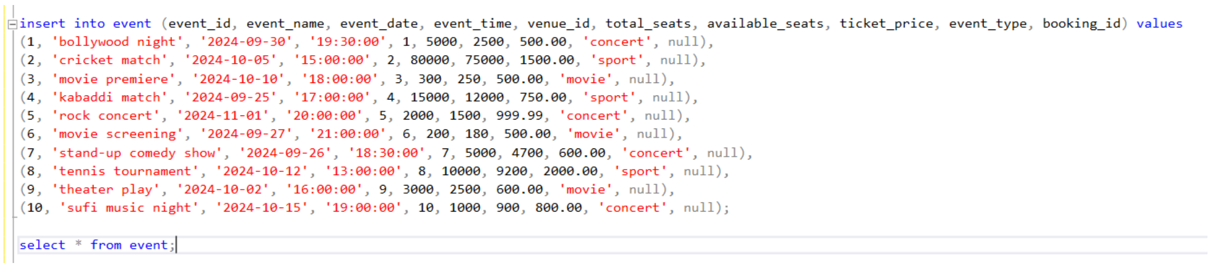
Query=>



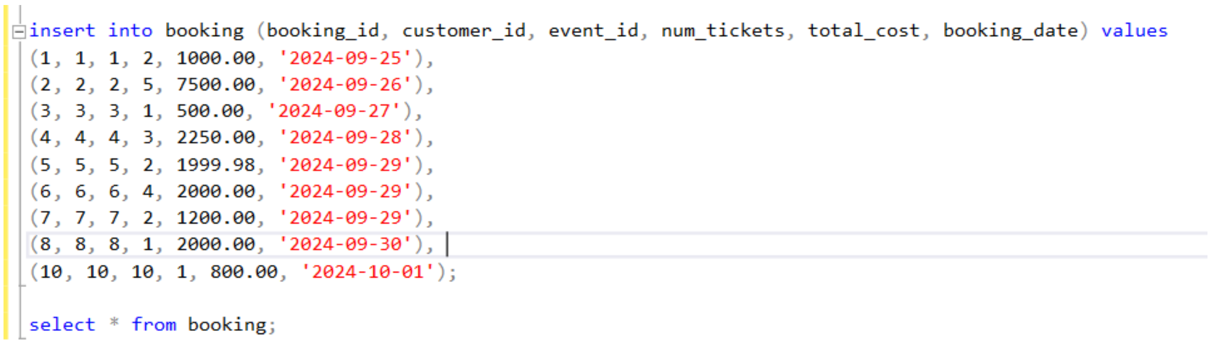
```sql
insert into venue (venue_id, venue_names, address) values
(1, 'nehru stadium', '123 m.g. road'),
(2, 'pvr cinema', '456 connaught place'),
(3, 'sardar patel stadium', '789 marine drive'),
(4, 'kamani auditorium', '101 rajpath'),
(5, 'india habitat center', '202 lodi road'),
(6, 'eden gardens', '303 howrah bridge'),
(7, 'inox cinema', '404 juhu beach'),
(8, 'mahalaxmi racecourse', '505 queens road'),
(9, 'mohanlal stadium', '606 nehru place'),
(10, 'sirifort auditorium', '707 south extension');

select * from venue;
```



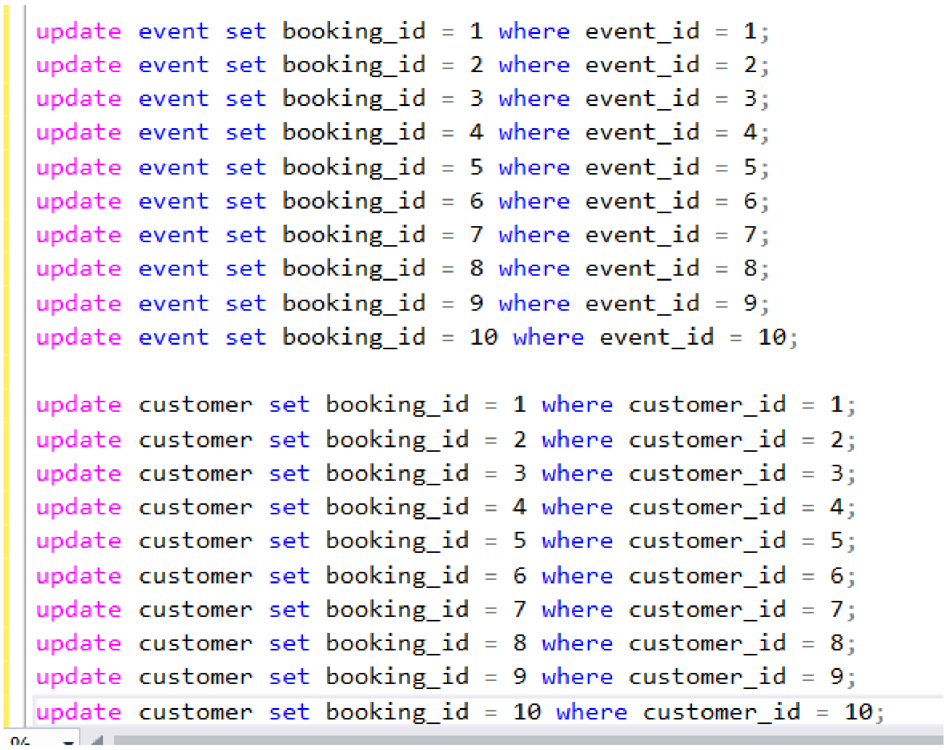
```sql
insert into customer (customer_id, customer_name, email, phone_number, booking_id) values
(1, 'rahul sharma', 'rahulsharma@example.com', '9876543210', null),
(2, 'priya singh', 'priyasingh@example.com', '8765432109', null),
(3, 'arjun patel', 'arjunpatel@example.com', '7654321098', null),
(4, 'swati verma', 'swativerma@example.com', '6543210987', null),
(5, 'ravi kumar', 'ravikumar@example.com', '5432109876', null),
(6, 'neha agrawal', 'nehaagrawal@example.com', '4321098765', null),
(7, 'ankit gupta', 'ankitgupta@example.com', '3210987654', null),
(8, 'pooja rao', 'poojarao@example.com', '2109876543', null),
(9, 'vivek yadav', 'vivekyadav@example.com', '1098765432', null),
(10, 'deepika naik', 'deepikanaik@example.com', '0987654321', null);

select * from customer;
```



```sql
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, 'bollywood night', '2024-09-30', '19:30:00', 1, 5000, 2500, 500.00, 'concert', null),
(2, 'cricket match', '2024-10-05', '15:00:00', 2, 80000, 75000, 1500.00, 'sport', null),
(3, 'movie premiere', '2024-10-10', '18:00:00', 3, 300, 250, 500.00, 'movie', null),
(4, 'kabaddi match', '2024-09-25', '17:00:00', 4, 15000, 12000, 750.00, 'sport', null),
(5, 'rock concert', '2024-11-01', '20:00:00', 5, 2000, 1500, 999.99, 'concert', null),
(6, 'movie screening', '2024-09-27', '21:00:00', 6, 200, 180, 500.00, 'movie', null),
(7, 'stand-up comedy show', '2024-09-26', '18:30:00', 7, 5000, 4700, 600.00, 'concert', null),
(8, 'tennis tournament', '2024-10-12', '13:00:00', 8, 10000, 9200, 2000.00, 'sport', null),
(9, 'theater play', '2024-10-02', '16:00:00', 9, 3000, 2500, 600.00, 'movie', null),
(10, 'sufi music night', '2024-10-15', '19:00:00', 10, 1000, 900, 800.00, 'concert', null);

select * from event;
```

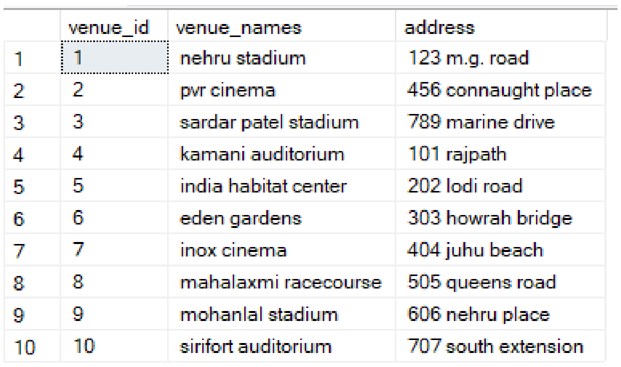


```sql
insert into booking (booking_id, customer_id, event_id, num_tickets, total_cost, booking_date) values
(1, 1, 1, 2, 1000.00, '2024-09-25'),
(2, 2, 2, 5, 7500.00, '2024-09-26'),
(3, 3, 3, 1, 500.00, '2024-09-27'),
(4, 4, 4, 3, 2250.00, '2024-09-28'),
(5, 5, 5, 2, 1999.98, '2024-09-29'),
(6, 6, 6, 4, 2000.00, '2024-09-29'),
(7, 7, 7, 2, 1200.00, '2024-09-29'),
(8, 8, 8, 1, 2000.00, '2024-09-30'),
(10, 10, 10, 1, 800.00, '2024-10-01');

select * from booking;
```

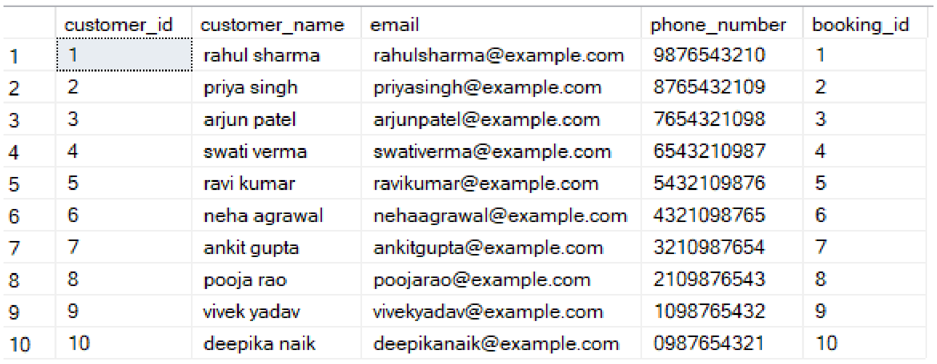


```sql
update event set booking_id = 1 where event_id = 1;
update event set booking_id = 2 where event_id = 2;
update event set booking_id = 3 where event_id = 3;
update event set booking_id = 4 where event_id = 4;
update event set booking_id = 5 where event_id = 5;
update event set booking_id = 6 where event_id = 6;
update event set booking_id = 7 where event_id = 7;
update event set booking_id = 8 where event_id = 8;
update event set booking_id = 9 where event_id = 9;
update event set booking_id = 10 where event_id = 10;

update customer set booking_id = 1 where customer_id = 1;
update customer set booking_id = 2 where customer_id = 2;
update customer set booking_id = 3 where customer_id = 3;
update customer set booking_id = 4 where customer_id = 4;
update customer set booking_id = 5 where customer_id = 5;
update customer set booking_id = 6 where customer_id = 6;
update customer set booking_id = 7 where customer_id = 7;
update customer set booking_id = 8 where customer_id = 8;
update customer set booking_id = 9 where customer_id = 9;
update customer set booking_id = 10 where customer_id = 10;
```

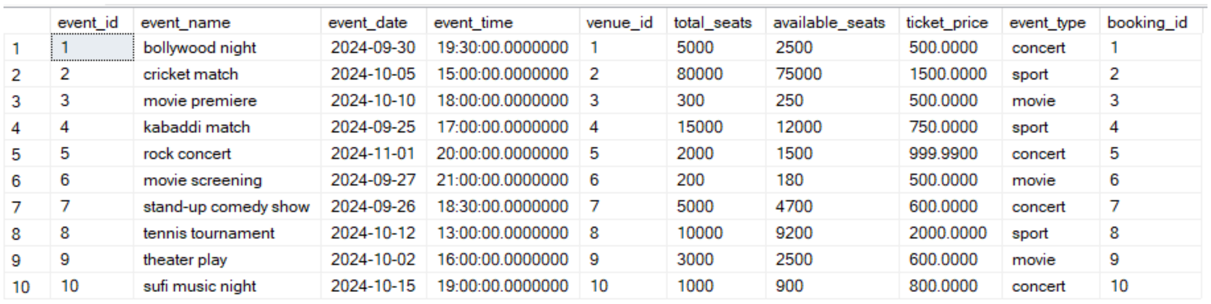
Result=>



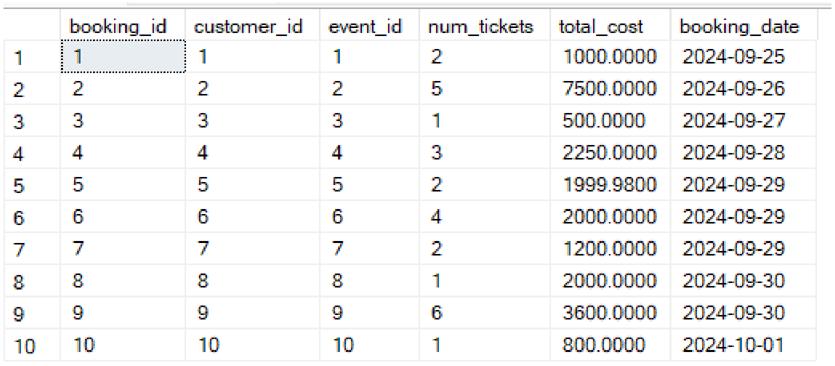
	venue_id	venue_names	address
1	1	nehru stadium	123 m.g. road
2	2	pvr cinema	456 connaught place
3	3	sardar patel stadium	789 marine drive
4	4	kamani auditorium	101 rajpath
5	5	india habitat center	202 lodi road
6	6	eden gardens	303 howrah bridge
7	7	inox cinema	404 juhu beach
8	8	mahalaxmi racecourse	505 queens road
9	9	mohanlal stadium	606 nehru place
10	10	sirifort auditorium	707 south extension



	customer_id	customer_name	email	phone_number	booking_id
1	1	rahul sharma	rahulsharma@example.com	9876543210	1
2	2	priya singh	priyasingh@example.com	8765432109	2
3	3	arjun patel	arjunpatel@example.com	7654321098	3
4	4	swati verma	swativerma@example.com	6543210987	4
5	5	ravi kumar	ravikumar@example.com	5432109876	5
6	6	neha agrawal	nehaagrawal@example.com	4321098765	6
7	7	ankit gupta	ankitgupta@example.com	3210987654	7
8	8	pooja rao	poojarao@example.com	2109876543	8
9	9	vivek yadav	vivekyadav@example.com	1098765432	9
10	10	deepika naik	deepikanaik@example.com	0987654321	10



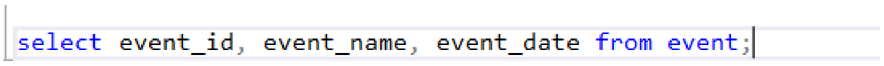
	event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	bollywood night	2024-09-30	19:30:00.0000000	1	5000	2500	500.0000	concert	1
2	2	cricket match	2024-10-05	15:00:00.0000000	2	80000	75000	1500.0000	sport	2
3	3	movie premiere	2024-10-10	18:00:00.0000000	3	300	250	500.0000	movie	3
4	4	kabaddi match	2024-09-25	17:00:00.0000000	4	15000	12000	750.0000	sport	4
5	5	rock concert	2024-11-01	20:00:00.0000000	5	2000	1500	999.9900	concert	5
6	6	movie screening	2024-09-27	21:00:00.0000000	6	200	180	500.0000	movie	6
7	7	stand-up comedy show	2024-09-26	18:30:00.0000000	7	5000	4700	600.0000	concert	7
8	8	tennis tournament	2024-10-12	13:00:00.0000000	8	10000	9200	2000.0000	sport	8
9	9	theater play	2024-10-02	16:00:00.0000000	9	3000	2500	600.0000	movie	9
10	10	sufi music night	2024-10-15	19:00:00.0000000	10	1000	900	800.0000	concert	10



	booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
1	1	1	1	2	1000.0000	2024-09-25
2	2	2	2	5	7500.0000	2024-09-26
3	3	3	3	1	500.0000	2024-09-27
4	4	4	4	3	2250.0000	2024-09-28
5	5	5	5	2	1999.9800	2024-09-29
6	6	6	6	4	2000.0000	2024-09-29
7	7	7	7	2	1200.0000	2024-09-29
8	8	8	8	1	2000.0000	2024-09-30
9	9	9	9	6	3600.0000	2024-09-30
10	10	10	10	1	800.0000	2024-10-01

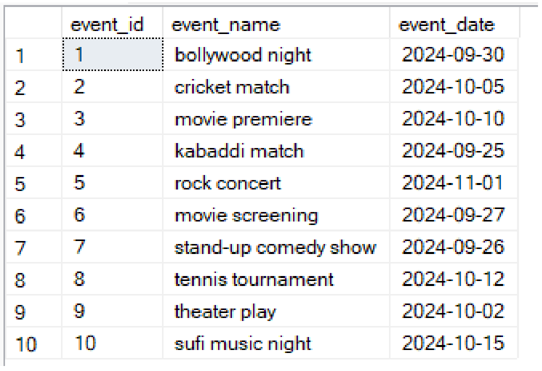
2. Write a SQL query to list all Events.

Query=>



```
select event_id, event_name, event_date from event;
```

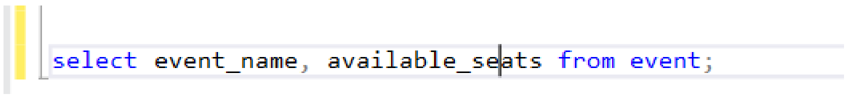
Result=>



	event_id	event_name	event_date
1	1	bollywood night	2024-09-30
2	2	cricket match	2024-10-05
3	3	movie premiere	2024-10-10
4	4	kabaddi match	2024-09-25
5	5	rock concert	2024-11-01
6	6	movie screening	2024-09-27
7	7	stand-up comedy show	2024-09-26
8	8	tennis tournament	2024-10-12
9	9	theater play	2024-10-02
10	10	sufi music night	2024-10-15

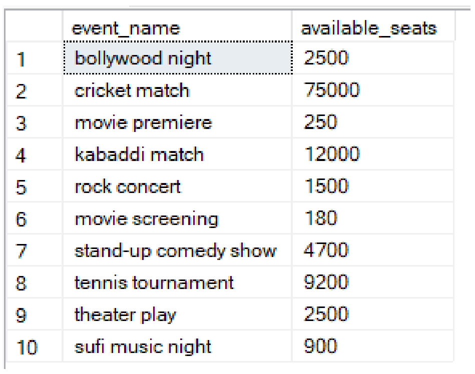
3. Write a SQL query to select events with available tickets.

Query=>



```
select event_name, available_seats from event;
```

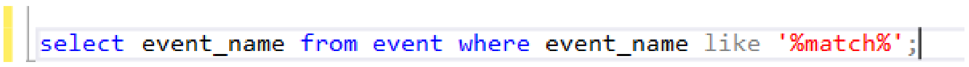
Result=>



	event_name	available_seats
1	bollywood night	2500
2	cricket match	75000
3	movie premiere	250
4	kabaddi match	12000
5	rock concert	1500
6	movie screening	180
7	stand-up comedy show	4700
8	tennis tournament	9200
9	theater play	2500
10	sufi music night	900

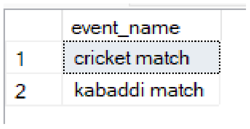
4. Write a SQL query to select events name partial match with 'match'.

Query=>



```
select event_name from event where event_name like '%match%';
```

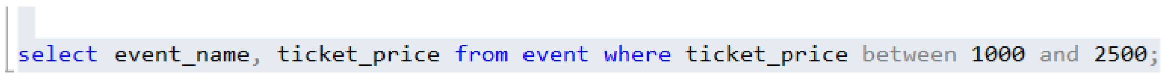
Result=>



	event_name
1	cricket match
2	kabaddi match

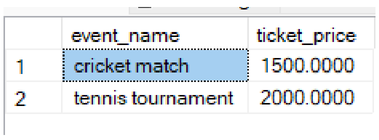
5. Write a SQL query to select events with ticket price range is between 1000 to 2500.

Query=>



```
select event_name, ticket_price from event where ticket_price between 1000 and 2500;
```

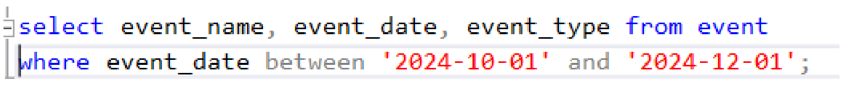
Result=>



	event_name	ticket_price
1	cricket match	1500.0000
2	tennis tournament	2000.0000

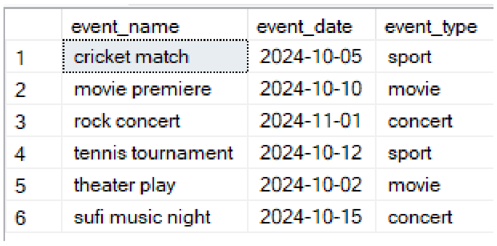
6. Write a SQL query to retrieve events with dates falling within a specific range.

Query=>



```
select event_name, event_date, event_type from event
where event_date between '2024-10-01' and '2024-12-01';
```

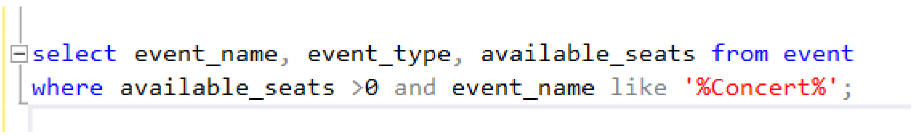
Result=>



	event_name	event_date	event_type
1	cricket match	2024-10-05	sport
2	movie premiere	2024-10-10	movie
3	rock concert	2024-11-01	concert
4	tennis tournament	2024-10-12	sport
5	theater play	2024-10-02	movie
6	sufi music night	2024-10-15	concert

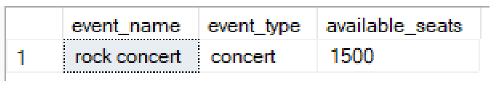
7. Write a SQL query to retrieve events with available tickets that also have "Concert" in their name.

Query=>



```
select event_name, event_type, available_seats from event
where available_seats >0 and event_name like '%Concert%';
```

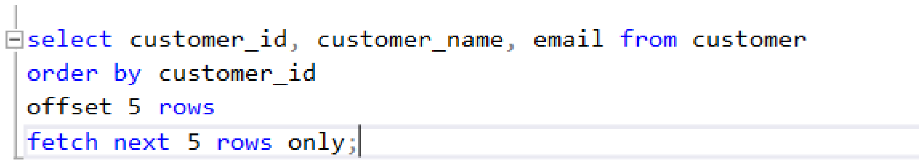
Result=>



	event_name	event_type	available_seats
1	rock concert	concert	1500

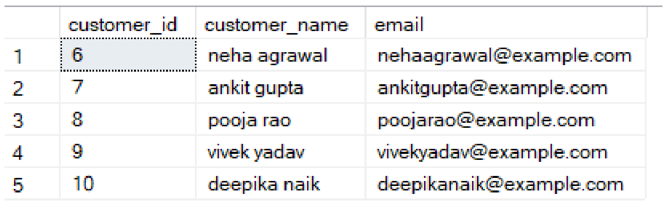
8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.

Query=>



```
select customer_id, customer_name, email from customer
order by customer_id
offset 5 rows
fetch next 5 rows only;
```

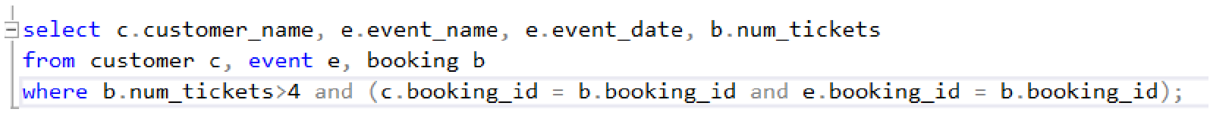
Result=>



	customer_id	customer_name	email
1	6	neha agrawal	nehaagrawal@example.com
2	7	ankit gupta	ankitgupta@example.com
3	8	pooja rao	poojarao@example.com
4	9	vivek yadav	vivekyadav@example.com
5	10	deepika naik	deepikanaik@example.com

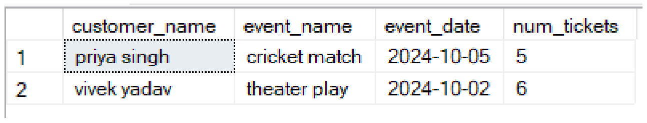
9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.

Query=>



```
select c.customer_name, e.event_name, e.event_date, b.num_tickets
from customer c, event e, booking b
where b.num_tickets>4 and (c.booking_id = b.booking_id and e.booking_id = b.booking_id);
```

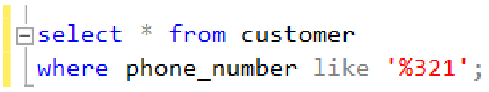
Result=>



	customer_name	event_name	event_date	num_tickets
1	priya singh	cricket match	2024-10-05	5
2	vivek yadav	theater play	2024-10-02	6

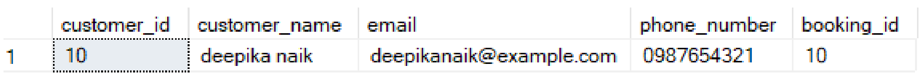
10. Write a SQL query to retrieve customer information whose phone number end with ‘000’

Query=>



```
select * from customer
where phone_number like '%321';
```

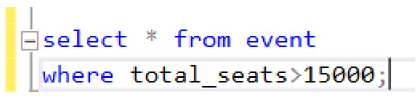
Result=>



	customer_id	customer_name	email	phone_number	booking_id
1	10	deepika naik	deepikanaik@example.com	0987654321	10

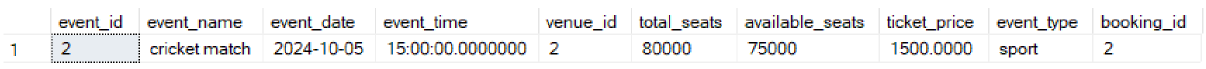
11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.

Query=>



```
select * from event
where total_seats>15000;
```

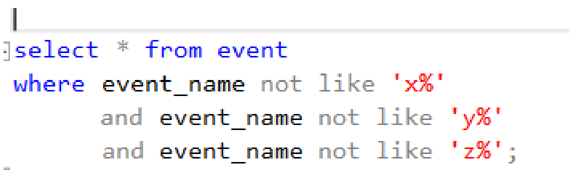
Result=>



	event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	2	cricket match	2024-10-05	15:00:00.0000000	2	80000	75000	1500.0000	sport	2

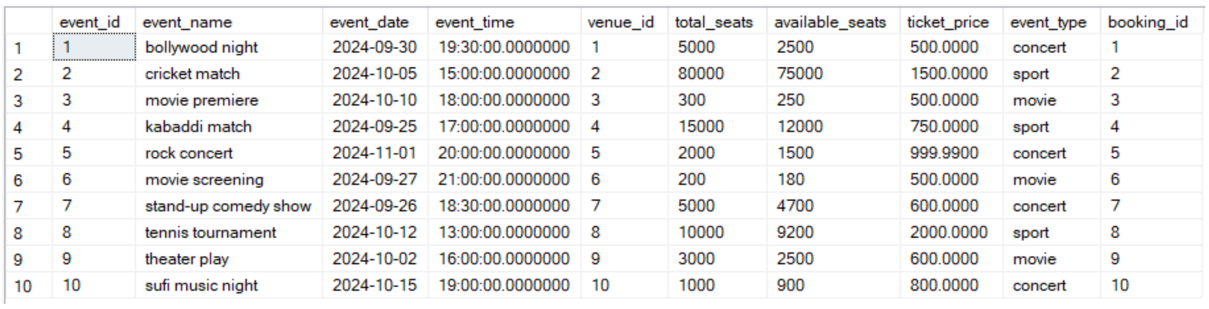
12. Write a SQL query to select events name not start with ‘x’, ‘y’, ‘z’

Query=>



```
select * from event
where event_name not like 'x%'
      and event_name not like 'y%'
      and event_name not like 'z%';
```

Result=>

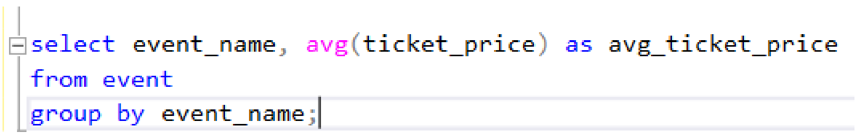


	event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	bollywood night	2024-09-30	19:30:00.0000000	1	5000	2500	500.0000	concert	1
2	2	cricket match	2024-10-05	15:00:00.0000000	2	80000	75000	1500.0000	sport	2
3	3	movie premiere	2024-10-10	18:00:00.0000000	3	300	250	500.0000	movie	3
4	4	kabaddi match	2024-09-25	17:00:00.0000000	4	15000	12000	750.0000	sport	4
5	5	rock concert	2024-11-01	20:00:00.0000000	5	2000	1500	999.9900	concert	5
6	6	movie screening	2024-09-27	21:00:00.0000000	6	200	180	500.0000	movie	6
7	7	stand-up comedy show	2024-09-26	18:30:00.0000000	7	5000	4700	600.0000	concert	7
8	8	tennis tournament	2024-10-12	13:00:00.0000000	8	10000	9200	2000.0000	sport	8
9	9	theater play	2024-10-02	16:00:00.0000000	9	3000	2500	600.0000	movie	9
10	10	sufi music night	2024-10-15	19:00:00.0000000	10	1000	900	800.0000	concert	10

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

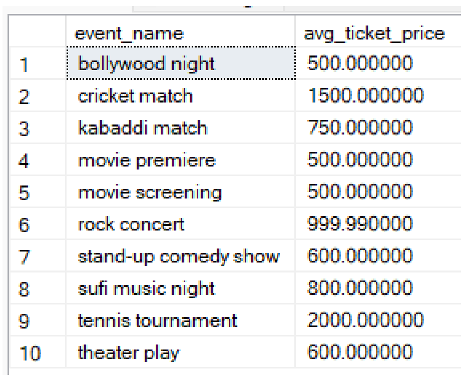
1. Write a SQL query to List Events and Their Average Ticket Prices.

Query =>



```sql
select event_name, avg(ticket_price) as avg_ticket_price
from event
group by event_name;
```

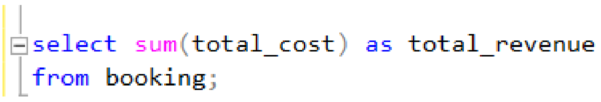
Result=>



	event_name	avg_ticket_price
1	bollywood night	500.000000
2	cricket match	1500.000000
3	kabaddi match	750.000000
4	movie premiere	500.000000
5	movie screening	500.000000
6	rock concert	999.990000
7	stand-up comedy show	600.000000
8	sufi music night	800.000000
9	tennis tournament	2000.000000
10	theater play	600.000000

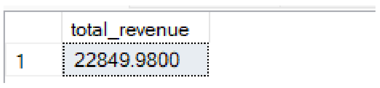
2. Write a SQL query to Calculate the Total Revenue Generated by Events.

Query =>



```sql
select sum(total_cost) as total_revenue
from booking;
```

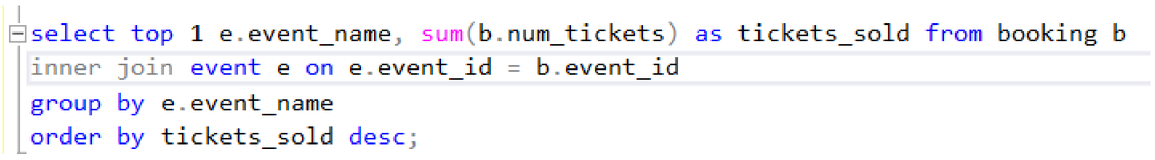
Result=>



	total_revenue
1	22849.9800

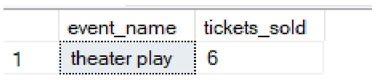
3. Write a SQL query to find the event with the highest ticket sales.

Query =>



```sql
select top 1 e.event_name, sum(b.num_tickets) as tickets_sold from booking b
inner join event e on e.event_id = b.event_id
group by e.event_name
order by tickets_sold desc;
```

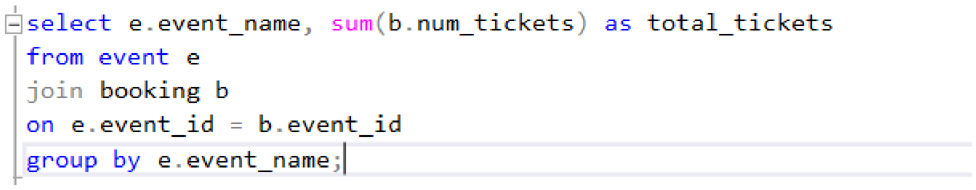
Result=>



	event_name	tickets_sold
1	theater play	6

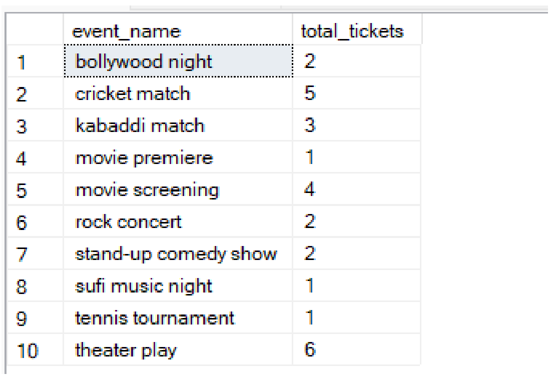
4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event.

Query =>



```sql
select e.event_name, sum(b.num_tickets) as total_tickets
from event e
join booking b
on e.event_id = b.event_id
group by e.event_name;
```

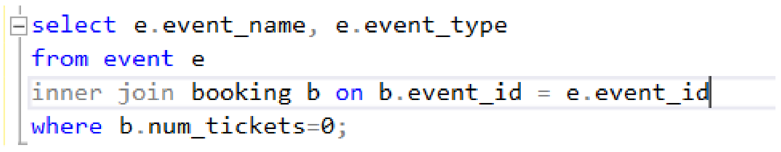
Result=>



	event_name	total_tickets
1	bollywood night	2
2	cricket match	5
3	kabaddi match	3
4	movie premiere	1
5	movie screening	4
6	rock concert	2
7	stand-up comedy show	2
8	sufi music night	1
9	tennis tournament	1
10	theater play	6

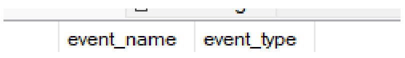
5. Write a SQL query to Find Events with No Ticket Sales.

Query =>



```sql
select e.event_name, e.event_type
from event e
inner join booking b on b.event_id = e.event_id
where b.num_tickets=0;
```

Result=>

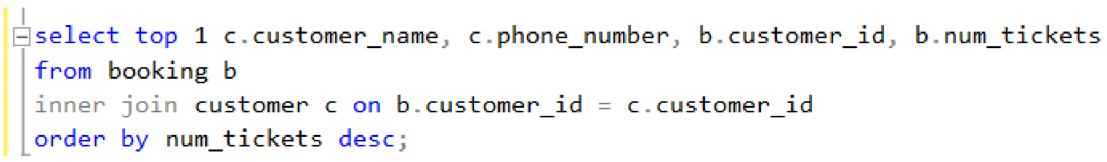


event_name	event_type

***Empty table as, each event has ticket sales greater than zero.**

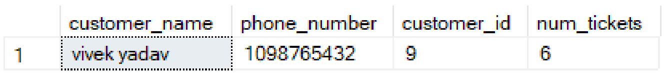
6. Write a SQL query to Find the User Who Has Booked the Most Tickets.

Query =>



```sql
select top 1 c.customer_name, c.phone_number, b.customer_id, b.num_tickets
from booking b
inner join customer c on b.customer_id = c.customer_id
order by num_tickets desc;
```

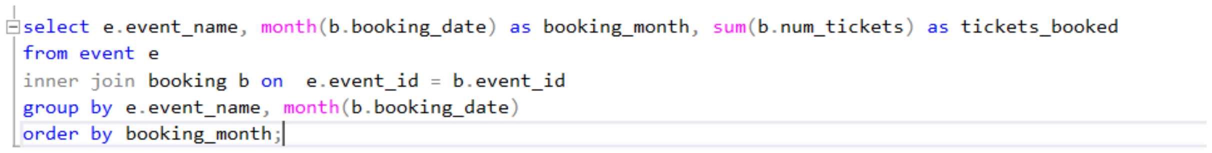
Result=>



	customer_name	phone_number	customer_id	num_tickets
1	vivek yadav	1098765432	9	6

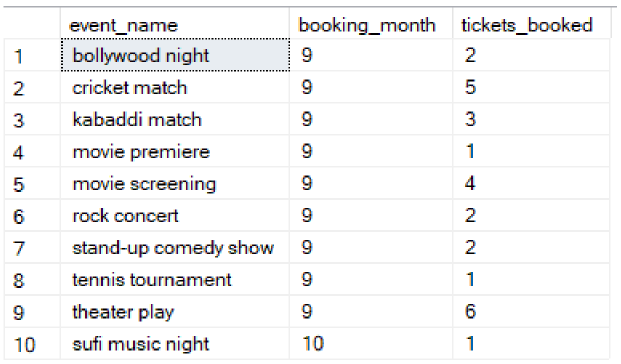
7. Write a SQL query to List Events and the total number of tickets sold for each month.

Query =>



```sql
select e.event_name, month(b.booking_date) as booking_month, sum(b.num_tickets) as tickets_booked
from event e
inner join booking b on  e.event_id = b.event_id
group by e.event_name, month(b.booking_date)
order by booking_month;
```

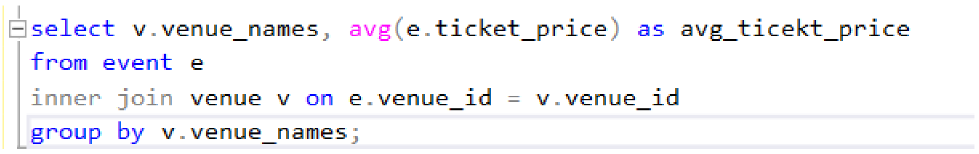
Result=>



	event_name	booking_month	tickets_booked
1	bollywood night	9	2
2	cricket match	9	5
3	kabaddi match	9	3
4	movie premiere	9	1
5	movie screening	9	4
6	rock concert	9	2
7	stand-up comedy show	9	2
8	tennis tournament	9	1
9	theater play	9	6
10	sufi music night	10	1

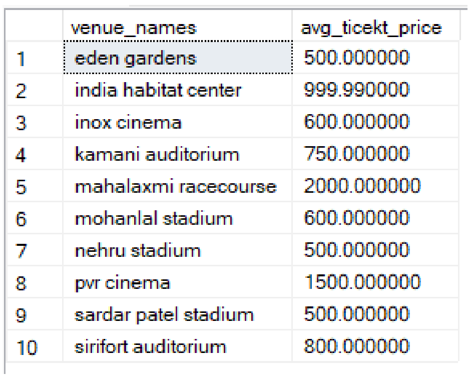
8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue.

Query =>



```sql
select v.venue_names, avg(e.ticket_price) as avg_ticekt_price
from event e
inner join venue v on e.venue_id = v.venue_id
group by v.venue_names;
```

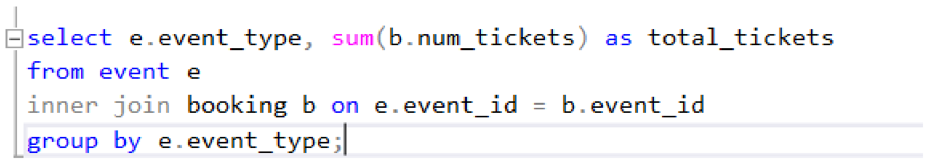
Result=>



	venue_names	avg_ticekt_price
1	eden gardens	500.000000
2	india habitat center	999.990000
3	inox cinema	600.000000
4	kamani auditorium	750.000000
5	mahalaxmi racecourse	2000.000000
6	mohanlal stadium	600.000000
7	nehru stadium	500.000000
8	pvr cinema	1500.000000
9	sardar patel stadium	500.000000
10	sirifort auditorium	800.000000

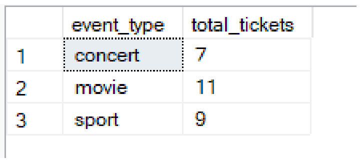
9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.

Query =>



```sql
select e.event_type, sum(b.num_tickets) as total_tickets
from event e
inner join booking b on e.event_id = b.event_id
group by e.event_type;
```

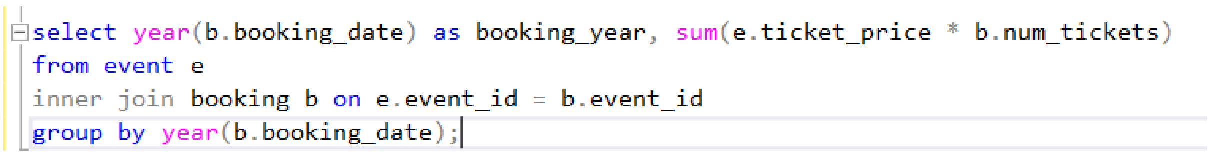
Result=>



	event_type	total_tickets
1	concert	7
2	movie	11
3	sport	9

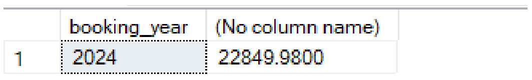
10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.

Query =>



```
select year(b.booking_date) as booking_year, sum(e.ticket_price * b.num_tickets)
from event e
inner join booking b on e.event_id = b.event_id
group by year(b.booking_date);
```

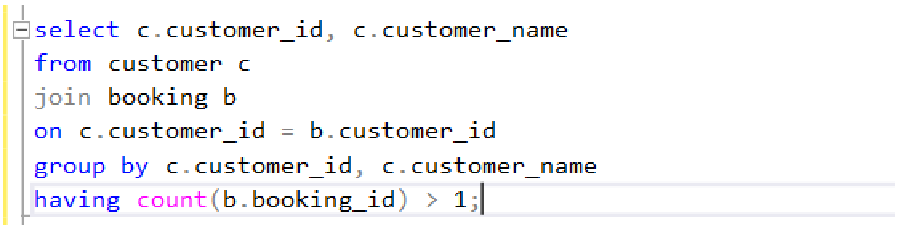
Result=>



	booking_year	(No column name)
1	2024	22849.9800

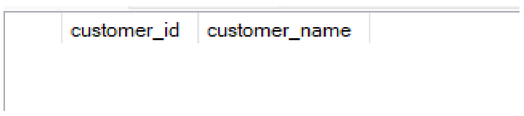
11. Write a SQL query to list users who have booked tickets for multiple events.

Query =>



```
select c.customer_id, c.customer_name
from customer c
join booking b
on c.customer_id = b.customer_id
group by c.customer_id, c.customer_name
having count(b.booking_id) > 1;
```

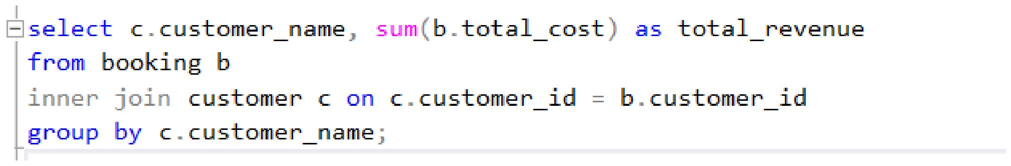
Result=>



	customer_id	customer_name

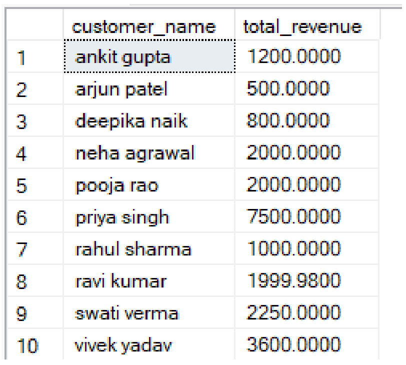
12. Write a SQL query to calculate the Total Revenue Generated by Events for Each User.

Query =>



```
select c.customer_name, sum(b.total_cost) as total_revenue
from booking b
inner join customer c on c.customer_id = b.customer_id
group by c.customer_name;
```

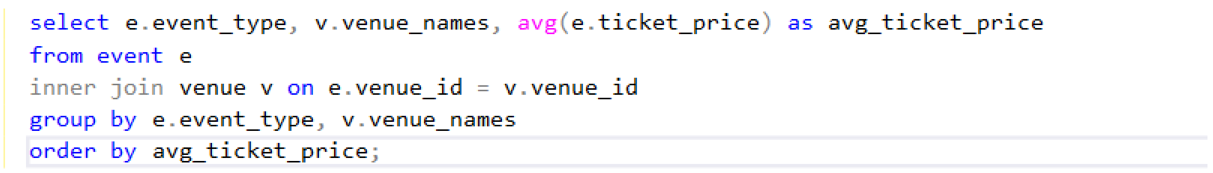
Result=>



	customer_name	total_revenue
1	ankit gupta	1200.0000
2	arjun patel	500.0000
3	deepika naik	800.0000
4	neha agrawal	2000.0000
5	pooja rao	2000.0000
6	priya singh	7500.0000
7	rahul sharma	1000.0000
8	ravi kumar	1999.9800
9	swati verma	2250.0000
10	vivek yadav	3600.0000

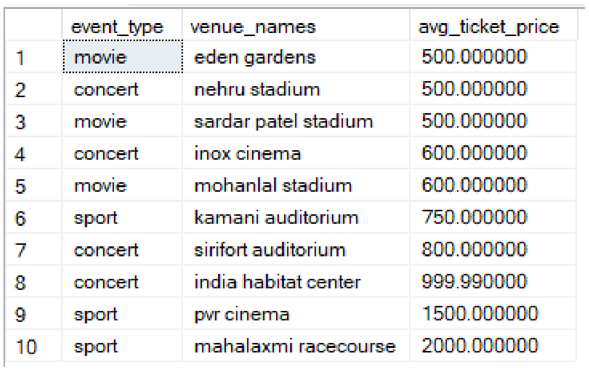
13. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.

Query =>



```
select e.event_type, v.venue_names, avg(e.ticket_price) as avg_ticket_price
from event e
inner join venue v on e.venue_id = v.venue_id
group by e.event_type, v.venue_names
order by avg_ticket_price;
```

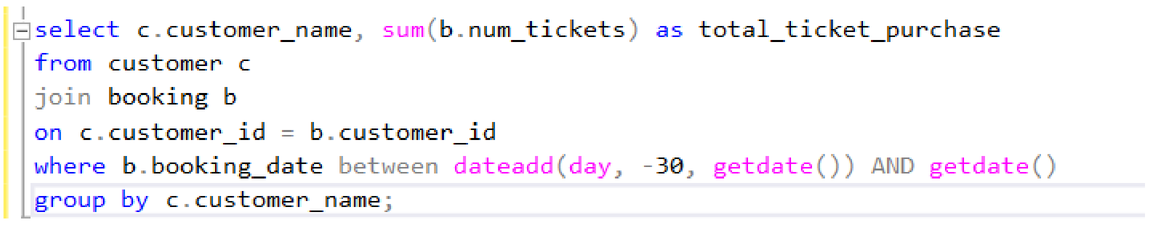
Result=>



	event_type	venue_names	avg_ticket_price
1	movie	eden gardens	500.000000
2	concert	nehru stadium	500.000000
3	movie	sardar patel stadium	500.000000
4	concert	inox cinema	600.000000
5	movie	mohanlal stadium	600.000000
6	sport	kamani auditorium	750.000000
7	concert	sirifort auditorium	800.000000
8	concert	india habitat center	999.990000
9	sport	pvr cinema	1500.000000
10	sport	mahalaxmi racecourse	2000.000000

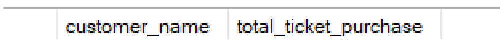
14. Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30 Days.

Query =>



```
select c.customer_name, sum(b.num_tickets) as total_ticket_purchase
from customer c
join booking b
on c.customer_id = b.customer_id
where b.booking_date between dateadd(day, -30, getdate()) AND getdate()
group by c.customer_name;
```

Result=>



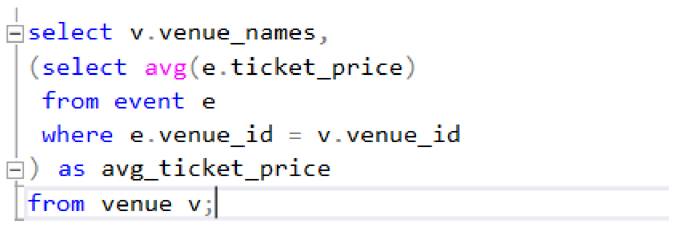
customer_name	total_ticket_purchase

***No records, as the booking dates are of future**

Tasks 4: Subquery and its types

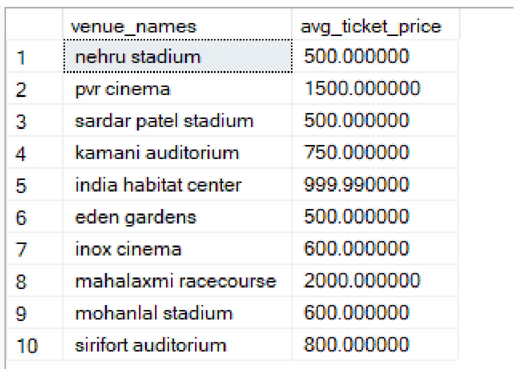
1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery.

Query=>



```sql
select v.venue_names,
(select avg(e.ticket_price)
 from event e
 where e.venue_id = v.venue_id
) as avg_ticket_price
from venue v;
```

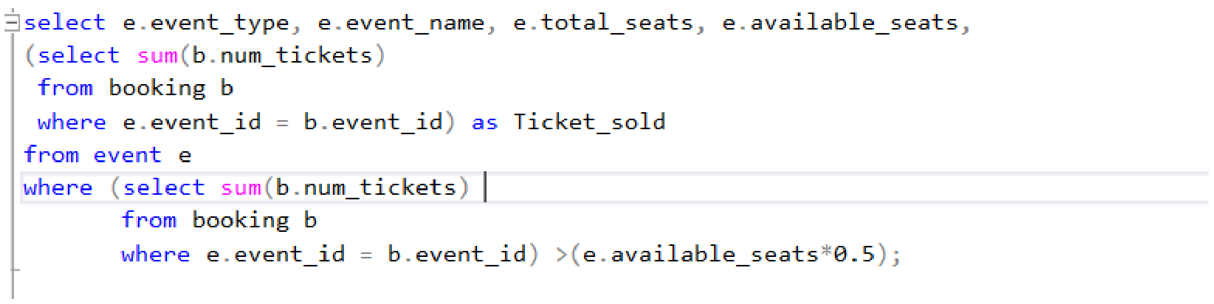
Result=>



	venue_names	avg_ticket_price
1	nehru stadium	500.000000
2	pvr cinema	1500.000000
3	sardar patel stadium	500.000000
4	kamani auditorium	750.000000
5	india habitat center	999.990000
6	eden gardens	500.000000
7	inox cinema	600.000000
8	mahalaxmi racecourse	2000.000000
9	mohanlal stadium	600.000000
10	sirifort auditorium	800.000000

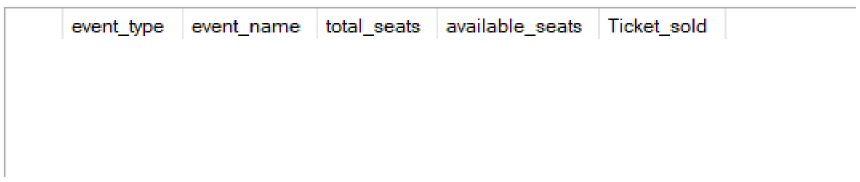
2. Find Events with More Than 50% of Tickets Sold using subquery.

Query=>



```sql
select e.event_type, e.event_name, e.total_seats, e.available_seats,
(select sum(b.num_tickets)
 from booking b
 where e.event_id = b.event_id) as Ticket_sold
from event e
where (select sum(b.num_tickets)
       from booking b
       where e.event_id = b.event_id) >(e.available_seats*0.5);
```

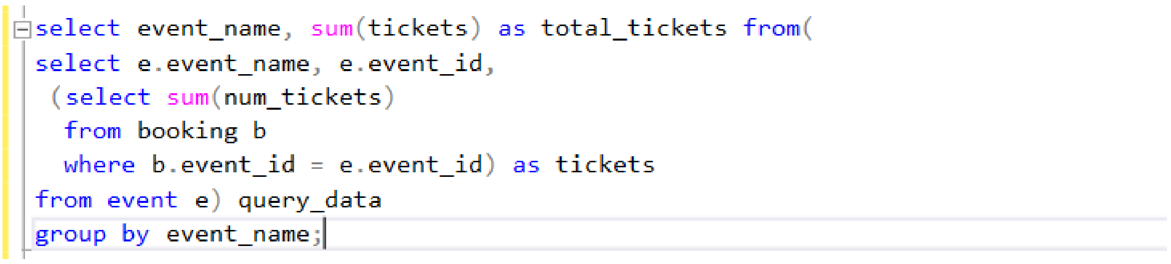
Result=>



	event_type	event_name	total_seats	available_seats	Ticket_sold

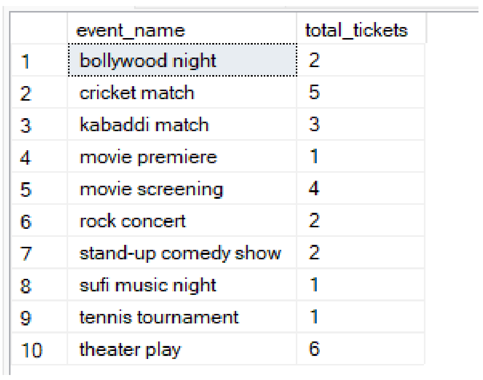
3. Calculate the Total Number of Tickets Sold for Each Event.

Query=>



```
select event_name, sum(tickets) as total_tickets from(
select e.event_name, e.event_id,
 (select sum(num_tickets)
  from booking b
  where b.event_id = e.event_id) as tickets
from event e) query_data
group by event_name;
```

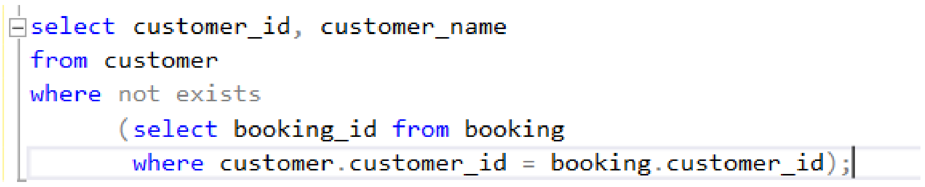
Result=>



	event_name	total_tickets
1	bollywood night	2
2	cricket match	5
3	kabaddi match	3
4	movie premiere	1
5	movie screening	4
6	rock concert	2
7	stand-up comedy show	2
8	sufi music night	1
9	tennis tournament	1
10	theater play	6

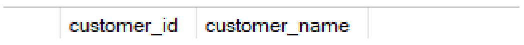
4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.

Query=>



```
select customer_id, customer_name
from customer
where not exists
      (select booking_id from booking
       where customer.customer_id = booking.customer_id);
```

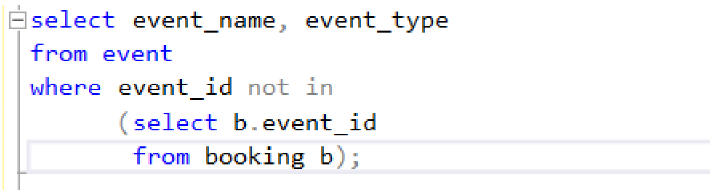
Result=>



	customer_id	customer_name

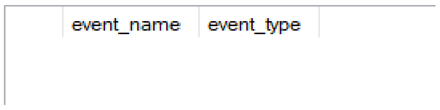
5. List Events with No Ticket Sales Using a NOT IN Subquery.

Query=>



```
select event_name, event_type
from event
where event_id not in
      (select b.event_id
       from booking b);
```

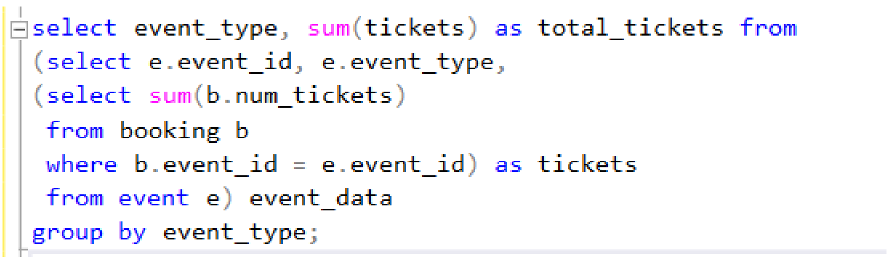
Result=>



	event_name	event_type

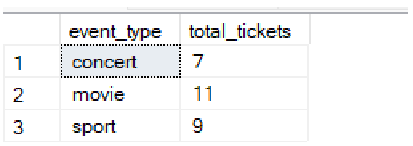
6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause.

Query=>



```
select event_type, sum(tickets) as total_tickets from
(select e.event_id, e.event_type,
(select sum(b.num_tickets)
 from booking b
 where b.event_id = e.event_id) as tickets
 from event e) event_data
group by event_type;
```

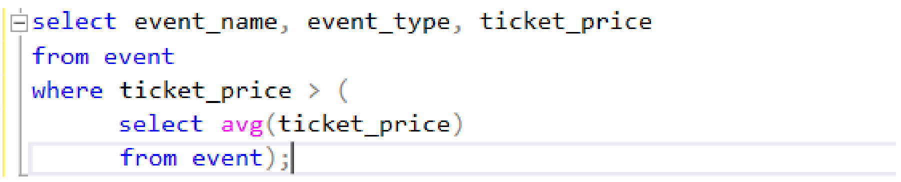
Result=>



	event_type	total_tickets
1	concert	7
2	movie	11
3	sport	9

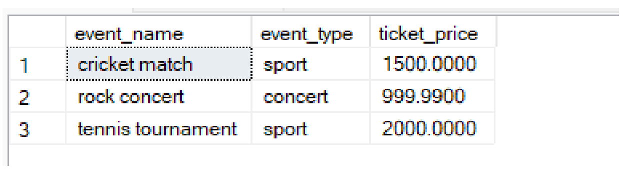
7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.

Query=>



```
select event_name, event_type, ticket_price
from event
where ticket_price > (
      select avg(ticket_price)
      from event);
```

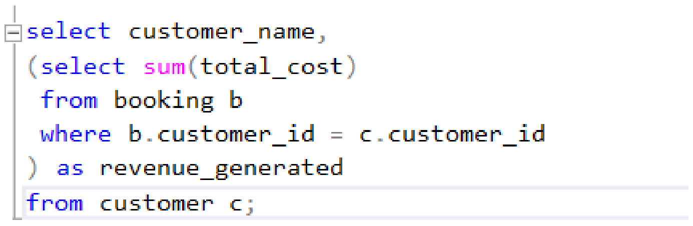
Result=>



	event_name	event_type	ticket_price
1	cricket match	sport	1500.0000
2	rock concert	concert	999.9900
3	tennis tournament	sport	2000.0000

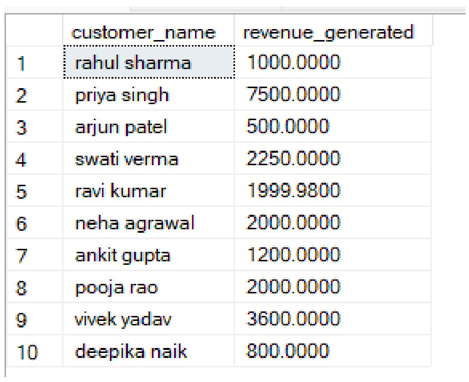
8. Calculate the Total Revenue Generated by Events for Each User Using a Correlated Subquery.

Query=>



```
select customer_name,
(select sum(total_cost)
 from booking b
 where b.customer_id = c.customer_id
) as revenue_generated
from customer c;
```

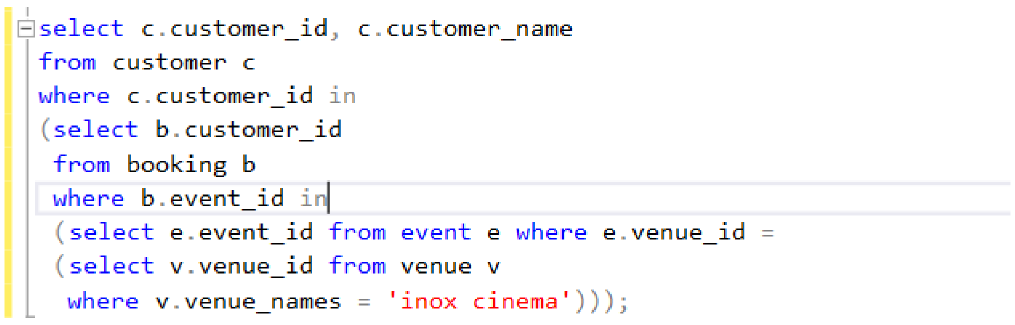
Result=>



	customer_name	revenue_generated
1	rahul sharma	1000.0000
2	priya singh	7500.0000
3	arjun patel	500.0000
4	swati verma	2250.0000
5	ravi kumar	1999.9800
6	neha agrawal	2000.0000
7	ankit gupta	1200.0000
8	pooja rao	2000.0000
9	vivek yadav	3600.0000
10	deepika naik	800.0000

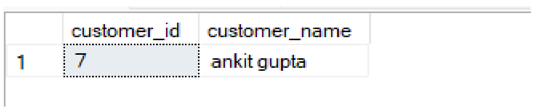
9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.

Query=>



```
select c.customer_id, c.customer_name
from customer c
where c.customer_id in
(select b.customer_id
 from booking b
 where b.event_id in
 (select e.event_id from event e where e.venue_id =
 (select v.venue_id from venue v
  where v.venue_names = 'inox cinema')));
```

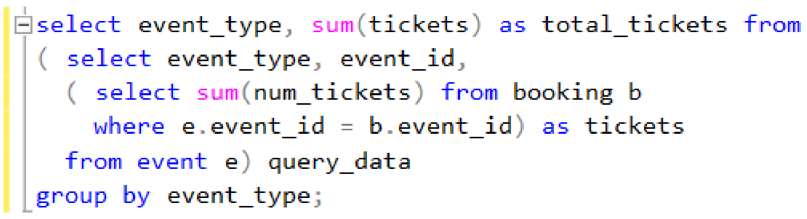
Result=>



	customer_id	customer_name
1	7	ankit gupta

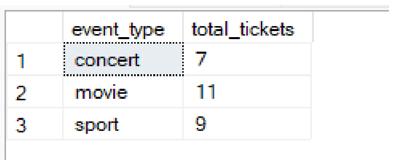
10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY.

Query=>



```
select event_type, sum(tickets) as total_tickets from
( select event_type, event_id,
  ( select sum(num_tickets) from booking b
    where e.event_id = b.event_id) as tickets
  from event e) query_data
group by event_type;
```

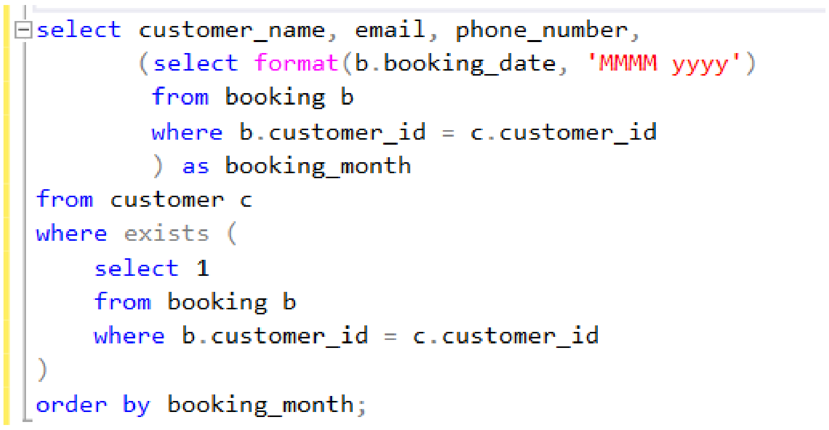
Result=>



	event_type	total_tickets
1	concert	7
2	movie	11
3	sport	9

11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT.

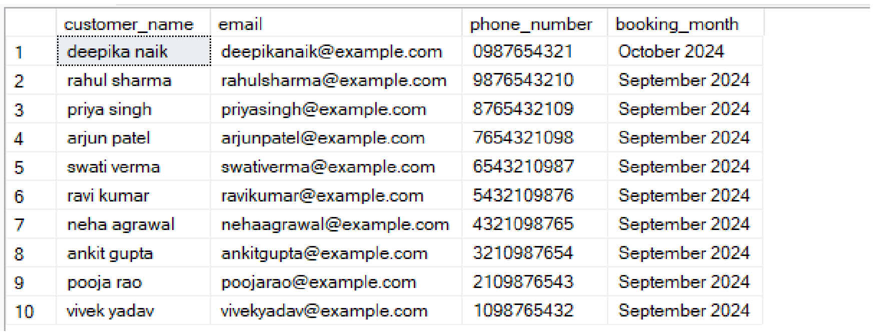
Query=>



```
select customer_name, email, phone_number,
       (select format(b.booking_date, 'MMMM yyyy')
        from booking b
        where b.customer_id = c.customer_id
        ) as booking_month
from customer c
where exists (
    select 1
    from booking b
    where b.customer_id = c.customer_id
)
order by booking_month;
```

***since there is no DATE_FORMAT in ms sql server, so we use format**

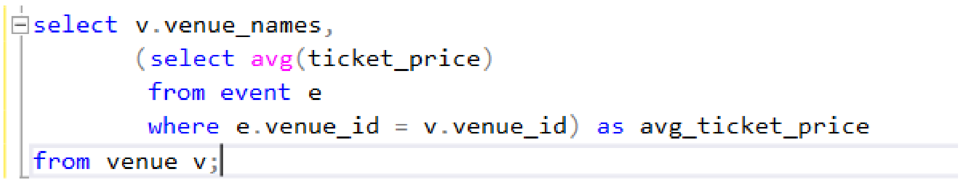
Result=>



	customer_name	email	phone_number	booking_month
1	deepika naik	deepikanaik@example.com	0987654321	October 2024
2	rahul sharma	rahulsharma@example.com	9876543210	September 2024
3	priya singh	priyasingh@example.com	8765432109	September 2024
4	arjun patel	arjunpatel@example.com	7654321098	September 2024
5	swati verma	swativerma@example.com	6543210987	September 2024
6	ravi kumar	ravikumar@example.com	5432109876	September 2024
7	neha agrawal	nehaagrawal@example.com	4321098765	September 2024
8	ankit gupta	ankitgupta@example.com	3210987654	September 2024
9	pooja rao	poojarao@example.com	2109876543	September 2024
10	vivek yadav	vivekyadav@example.com	1098765432	September 2024

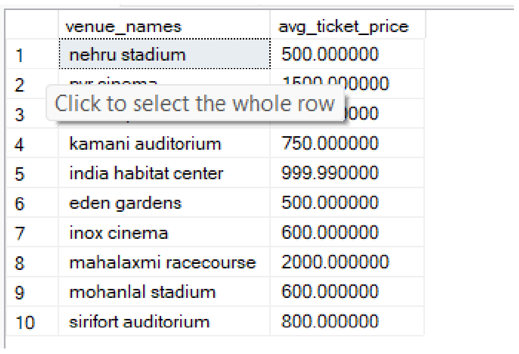
12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery

Query=>



```
select v.venue_names,
       (select avg(ticket_price)
        from event e
        where e.venue_id = v.venue_id) as avg_ticket_price
from venue v;
```

Result=>



	venue_names	avg_ticket_price
1	nehru stadium	500.000000
2	pvr cinema	1500.000000
3	[illegible]	[illegible]000
4	kamani auditorium	750.000000
5	india habitat center	999.990000
6	eden gardens	500.000000
7	inox cinema	600.000000
8	mahalaxmi racecourse	2000.000000
9	mohanlal stadium	600.000000
10	sirifort auditorium	800.000000