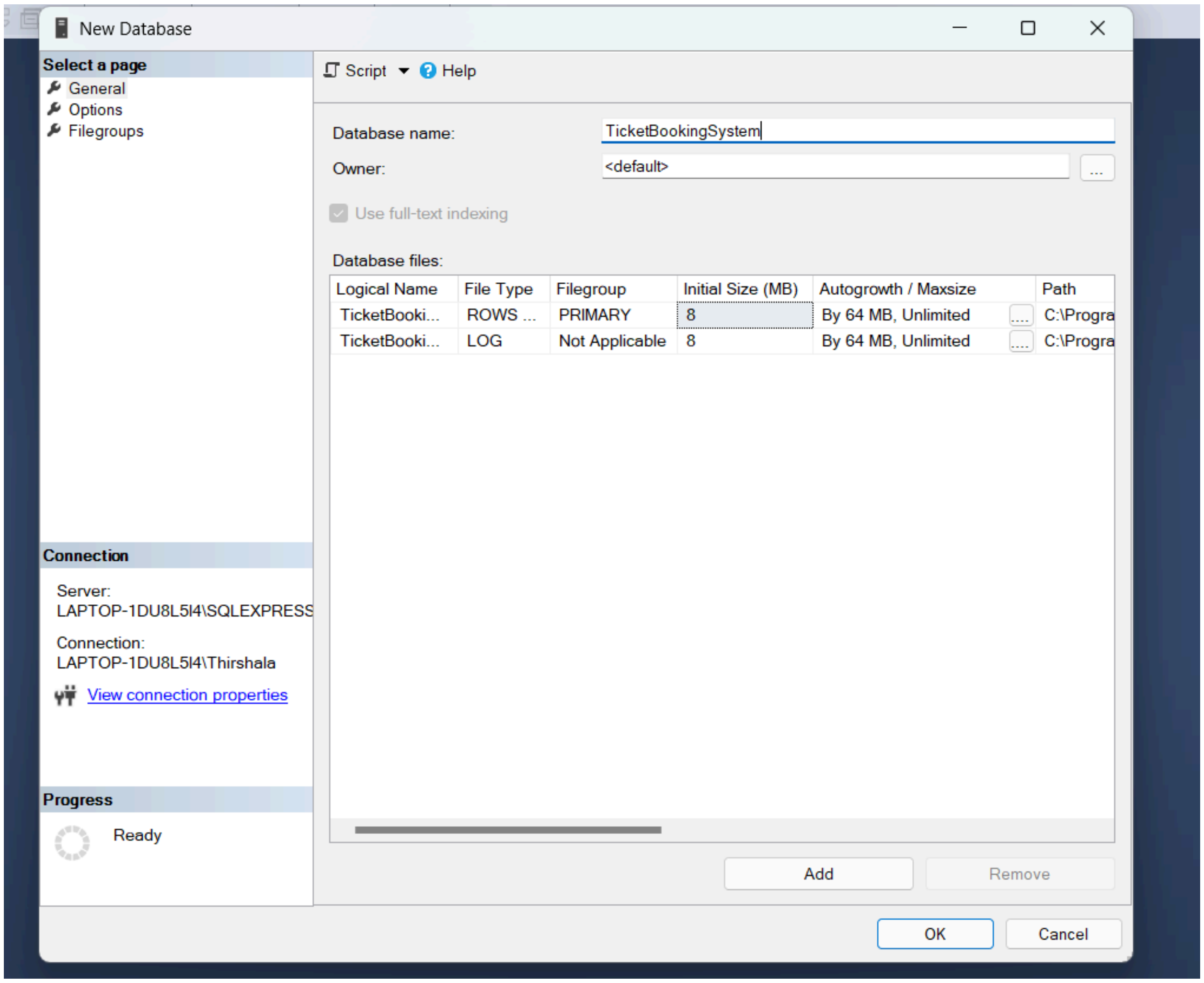
ASSIGNMENT
TICKET BOOKING SYSTEM

Name:Nidya Thirshala M

Tasks 1: Database Design:
1. Create a database named "TicketBookingSystem".



Database name: TicketBookingSystem
Owner: <default>
Use full-text indexing

Database files:

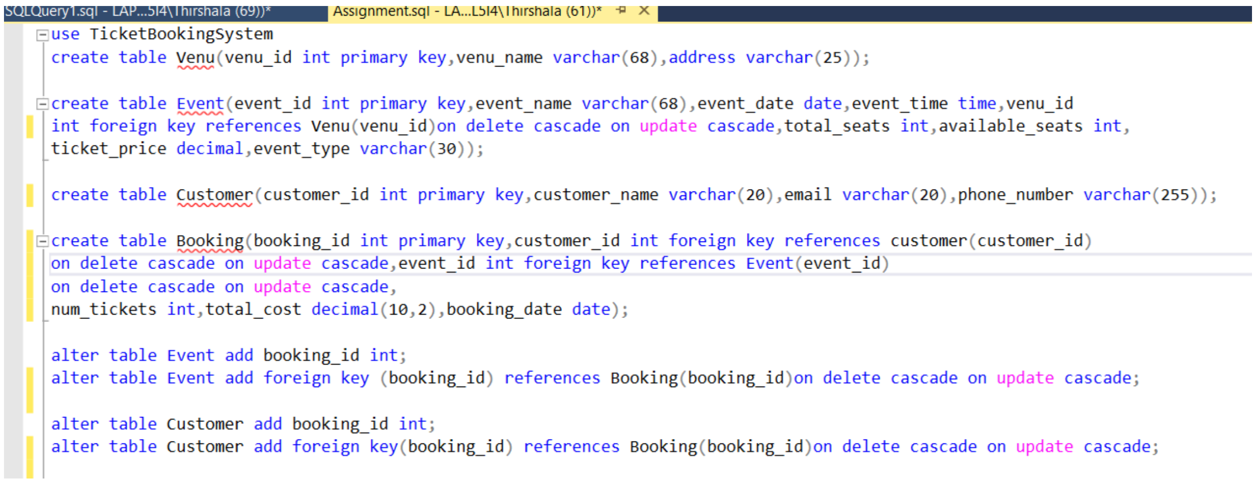
Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Path
TicketBooki...	ROWS ...	PRIMARY	8	By 64 MB, Unlimited	C:\Progra
TicketBooki...	LOG	Not Applicable	8	By 64 MB, Unlimited	C:\Progra

Server:
LAPTOP-1DU8L5I4\SQLEXPRESS

Connection:
LAPTOP-1DU8L5I4\Thirshala

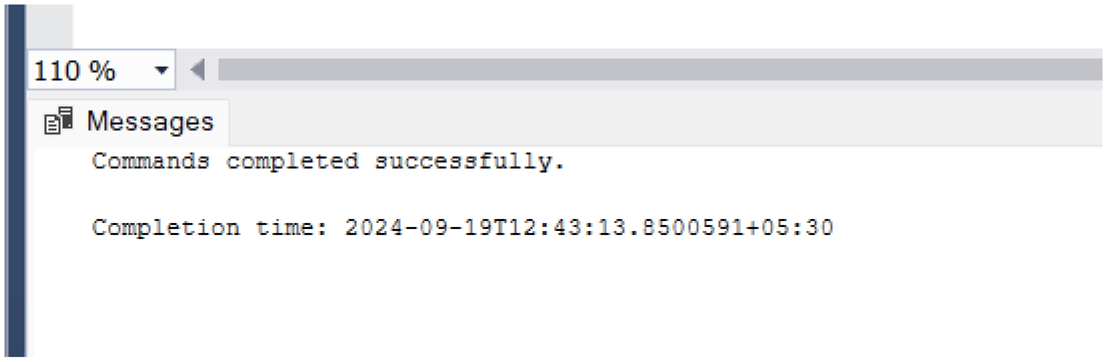
2. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
• Venu
• Event
• Customers

• **Booking**


SQLQuery1.sql - LAP...5I4\Thirshala (69))*   Assignment.sql - LA...L5I4\Thirshala (61))*

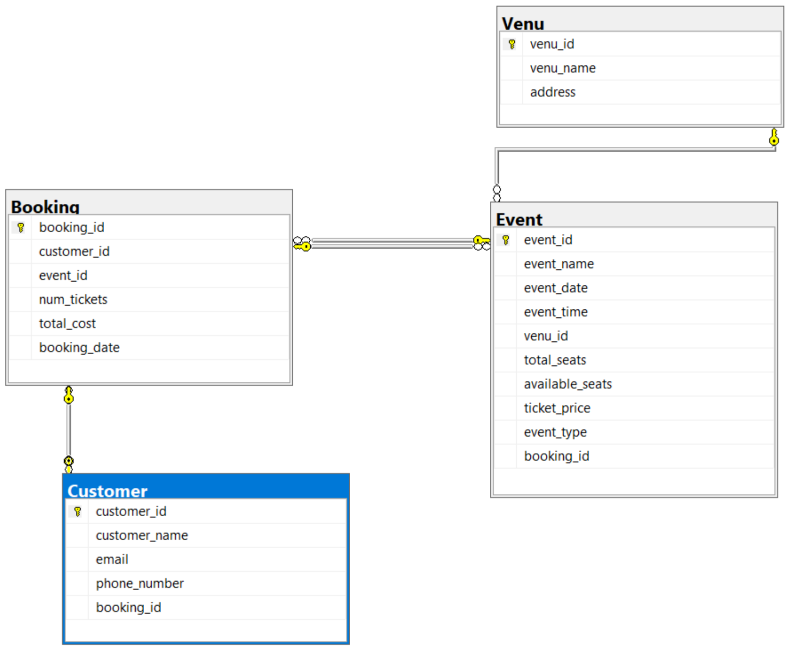
```
use TicketBookingSystem
create table Venu(venu_id int primary key,venu_name varchar(68),address varchar(25));

create table Event(event_id int primary key,event_name varchar(68),event_date date,event_time time,venu_id
int foreign key references Venu(venu_id)on delete cascade on update cascade,total_seats int,available_seats int,
ticket_price decimal,event_type varchar(30));

create table Customer(customer_id int primary key,customer_name varchar(20),email varchar(20),phone_number varchar(255));

create table Booking(booking_id int primary key,customer_id int foreign key references customer(customer_id)
on delete cascade on update cascade,event_id int foreign key references Event(event_id)
on delete cascade on update cascade,
num_tickets int,total_cost decimal(10,2),booking_date date);

alter table Event add booking_id int;
alter table Event add foreign key (booking_id) references Booking(booking_id)on delete cascade on update cascade;

alter table Customer add booking_id int;
alter table Customer add foreign key(booking_id) references Booking(booking_id)on delete cascade on update cascade;
```

OUTPUT:



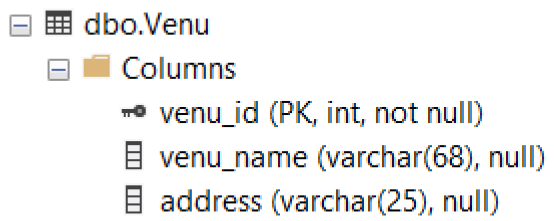
```
Commands completed successfully.

Completion time: 2024-09-19T12:43:13.8500591+05:30
```

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

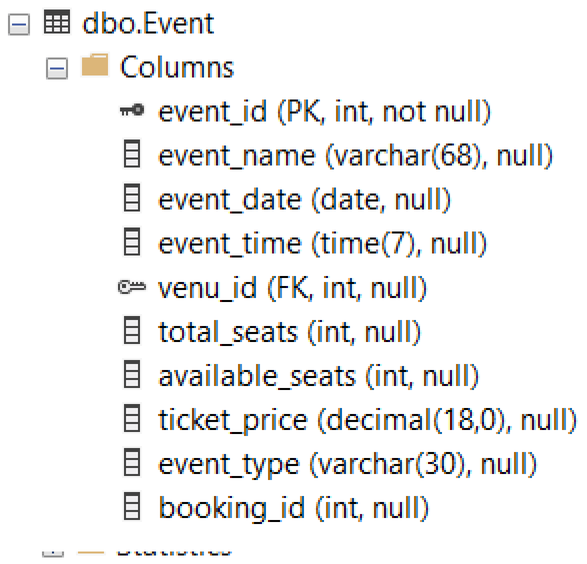


4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

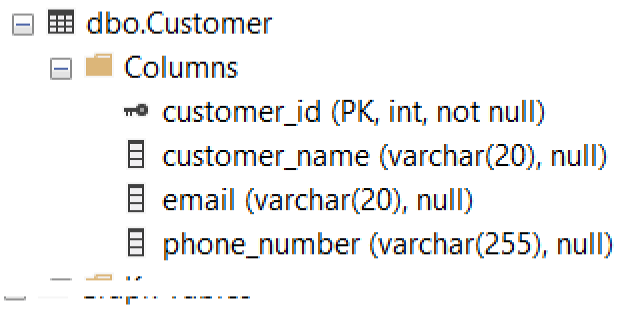
All primary key and Foreign key are created while creating the tables .



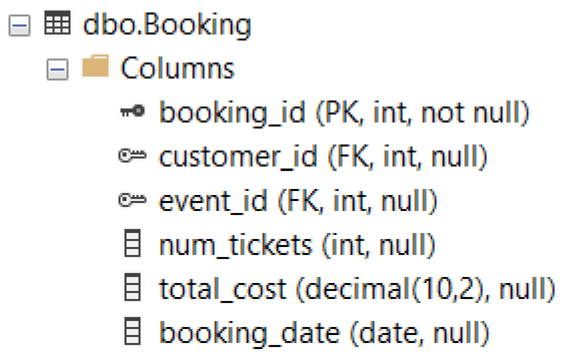
```
dbo.Venu
  Columns
    venu_id (PK, int, not null)
    venu_name (varchar(68), null)
    address (varchar(25), null)
```



```
dbo.Event
  Columns
    event_id (PK, int, not null)
    event_name (varchar(68), null)
    event_date (date, null)
    event_time (time(7), null)
    venu_id (FK, int, null)
    total_seats (int, null)
    available_seats (int, null)
    ticket_price (decimal(18,0), null)
    event_type (varchar(30), null)
    booking_id (int, null)
```



```
dbo.Customer
  Columns
    customer_id (PK, int, not null)
    customer_name (varchar(20), null)
    email (varchar(20), null)
    phone_number (varchar(255), null)
```

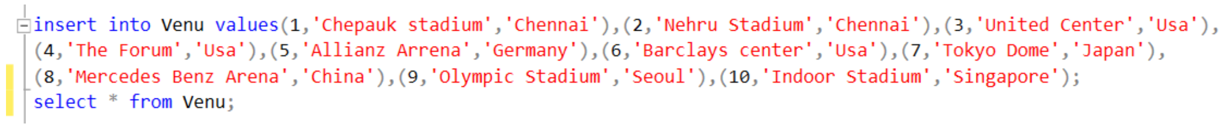


```
dbo.Booking
  Columns
    booking_id (PK, int, not null)
    customer_id (FK, int, null)
    event_id (FK, int, null)
    num_tickets (int, null)
    total_cost (decimal(10,2), null)
    booking_date (date, null)
```

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

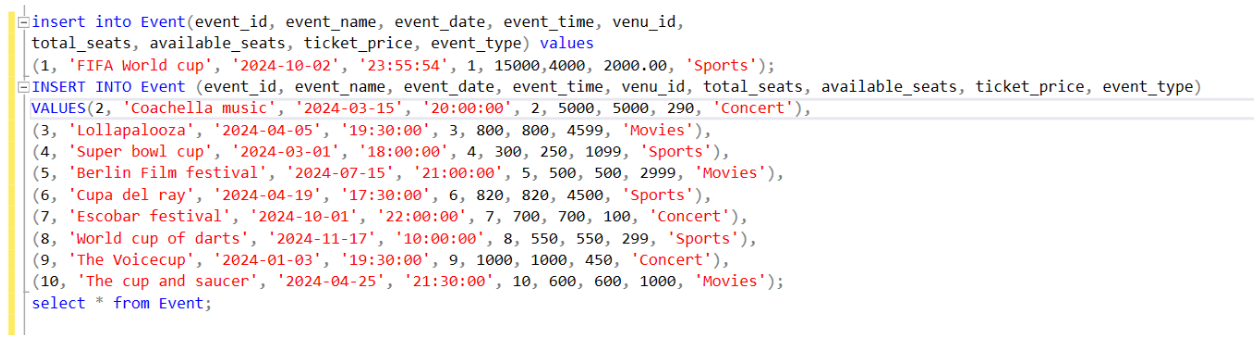
1. Write a SQL query to insert at least 10 sample records into each table.

Venu Table:



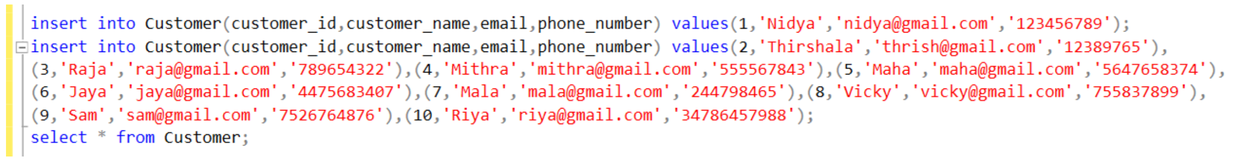
```sql
insert into Venu values(1,'Chepauk stadium','Chennai'),(2,'Nehru Stadium','Chennai'),(3,'United Center','Usa'),
(4,'The Forum','Usa'),(5,'Allianz Arrena','Germany'),(6,'Barclays center','Usa'),(7,'Tokyo Dome','Japan'),
(8,'Mercedes Benz Arena','China'),(9,'Olympic Stadium','Seoul'),(10,'Indoor Stadium','Singapore');
select * from Venu;
```

Event Table:



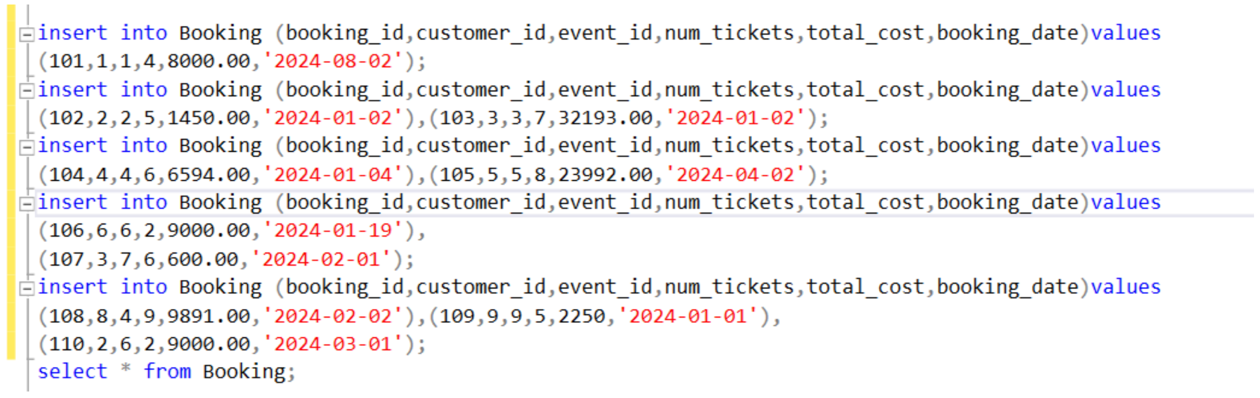
```sql
insert into Event(event_id, event_name, event_date, event_time, venu_id,
total_seats, available_seats, ticket_price, event_type) values
(1, 'FIFA World cup', '2024-10-02', '23:55:54', 1, 15000,4000, 2000.00, 'Sports');
INSERT INTO Event (event_id, event_name, event_date, event_time, venu_id, total_seats, available_seats, ticket_price, event_type)
VALUES(2, 'Coachella music', '2024-03-15', '20:00:00', 2, 5000, 5000, 290, 'Concert'),
(3, 'Lollapalooza', '2024-04-05', '19:30:00', 3, 800, 800, 4599, 'Movies'),
(4, 'Super bowl cup', '2024-03-01', '18:00:00', 4, 300, 250, 1099, 'Sports'),
(5, 'Berlin Film festival', '2024-07-15', '21:00:00', 5, 500, 500, 2999, 'Movies'),
(6, 'Cupa del ray', '2024-04-19', '17:30:00', 6, 820, 820, 4500, 'Sports'),
(7, 'Escobar festival', '2024-10-01', '22:00:00', 7, 700, 700, 100, 'Concert'),
(8, 'World cup of darts', '2024-11-17', '10:00:00', 8, 550, 550, 299, 'Sports'),
(9, 'The Voicecup', '2024-01-03', '19:30:00', 9, 1000, 1000, 450, 'Concert'),
(10, 'The cup and saucer', '2024-04-25', '21:30:00', 10, 600, 600, 1000, 'Movies');
select * from Event;
```

Customer Table:



```sql
insert into Customer(customer_id,customer_name,email,phone_number) values(1,'Nidya','nidya@gmail.com','123456789');
insert into Customer(customer_id,customer_name,email,phone_number) values(2,'Thirshala','thrish@gmail.com','12389765'),
(3,'Raja','raja@gmail.com','789654322'),(4,'Mithra','mithra@gmail.com','555567843'),(5,'Maha','maha@gmail.com','5647658374'),
(6,'Jaya','jaya@gmail.com','4475683407'),(7,'Mala','mala@gmail.com','244798465'),(8,'Vicky','vicky@gmail.com','755837899'),
(9,'Sam','sam@gmail.com','7526764876'),(10,'Riya','riya@gmail.com','34786457988');
select * from Customer;
```

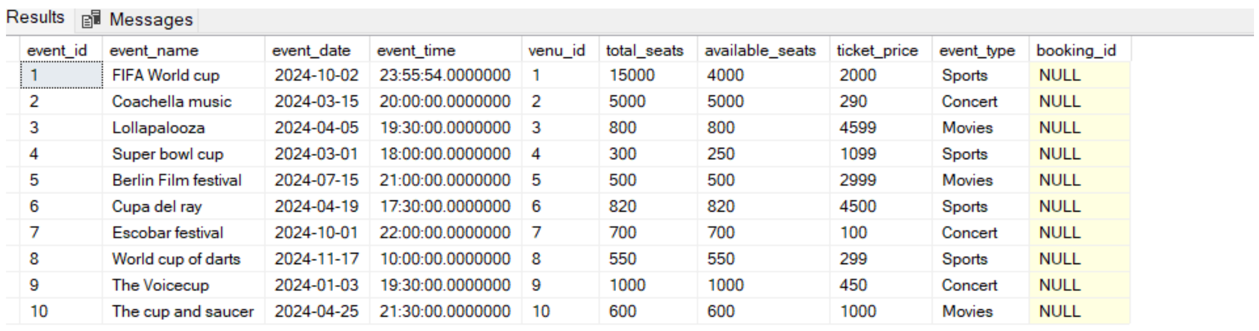
Booking Table:



```sql
insert into Booking (booking_id,customer_id,event_id,num_tickets,total_cost,booking_date)values
(101,1,1,4,8000.00,'2024-08-02');
insert into Booking (booking_id,customer_id,event_id,num_tickets,total_cost,booking_date)values
(102,2,2,5,1450.00,'2024-01-02'),(103,3,3,7,32193.00,'2024-01-02');
insert into Booking (booking_id,customer_id,event_id,num_tickets,total_cost,booking_date)values
(104,4,4,6,6594.00,'2024-01-04'),(105,5,5,8,23992.00,'2024-04-02');
insert into Booking (booking_id,customer_id,event_id,num_tickets,total_cost,booking_date)values
(106,6,6,2,9000.00,'2024-01-19'),
(107,3,7,6,600.00,'2024-02-01');
insert into Booking (booking_id,customer_id,event_id,num_tickets,total_cost,booking_date)values
(108,8,4,9,9891.00,'2024-02-02'),(109,9,9,5,2250,'2024-01-01'),
(110,2,6,2,9000.00,'2024-03-01');
select * from Booking;
```

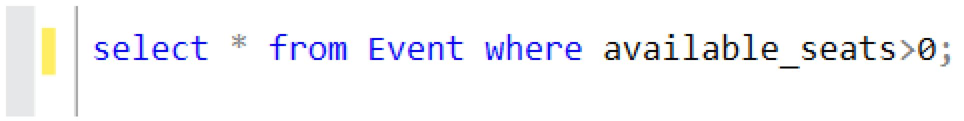
2. Write a SQL query to list all Events.

Select * from Events;

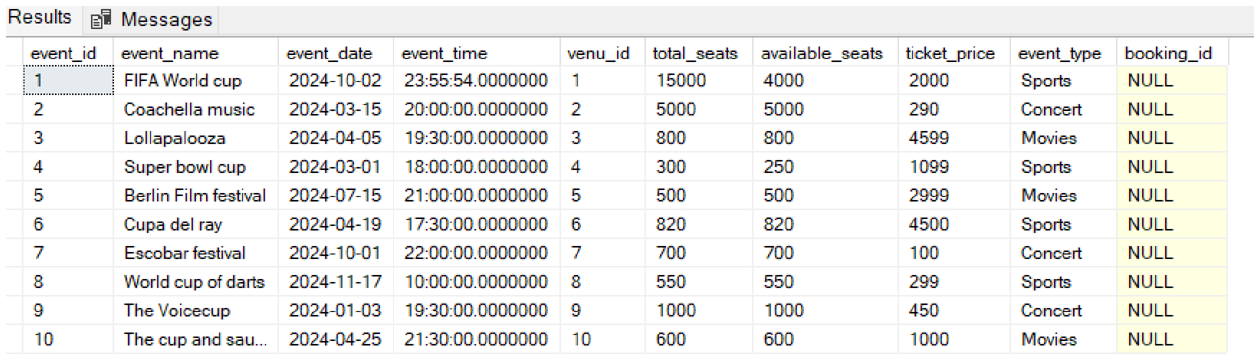


event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
2	Coachella music	2024-03-15	20:00:00.0000000	2	5000	5000	290	Concert	NULL
3	Lollapalooza	2024-04-05	19:30:00.0000000	3	800	800	4599	Movies	NULL
4	Super bowl cup	2024-03-01	18:00:00.0000000	4	300	250	1099	Sports	NULL
5	Berlin Film festival	2024-07-15	21:00:00.0000000	5	500	500	2999	Movies	NULL
6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
7	Escobar festival	2024-10-01	22:00:00.0000000	7	700	700	100	Concert	NULL
8	World cup of darts	2024-11-17	10:00:00.0000000	8	550	550	299	Sports	NULL
9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL
10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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

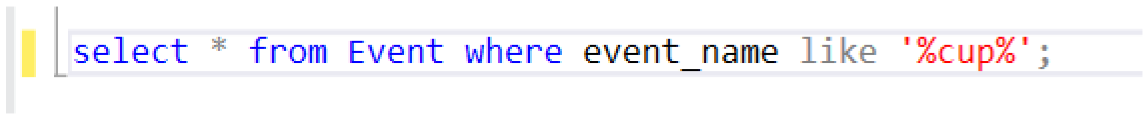


```sql
select * from Event where available_seats>0;
```

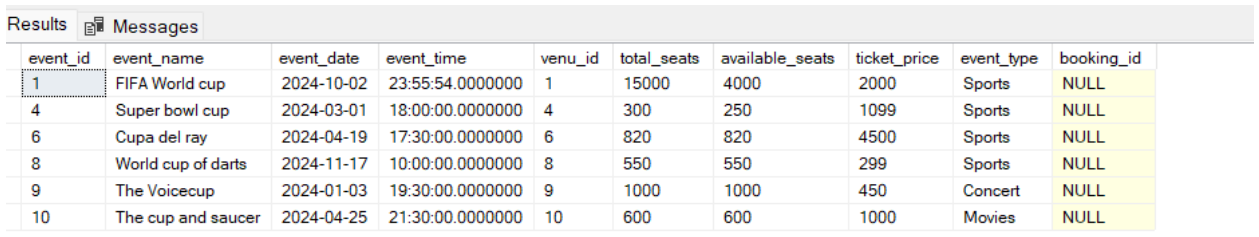


event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
2	Coachella music	2024-03-15	20:00:00.0000000	2	5000	5000	290	Concert	NULL
3	Lollapalooza	2024-04-05	19:30:00.0000000	3	800	800	4599	Movies	NULL
4	Super bowl cup	2024-03-01	18:00:00.0000000	4	300	250	1099	Sports	NULL
5	Berlin Film festival	2024-07-15	21:00:00.0000000	5	500	500	2999	Movies	NULL
6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
7	Escobar festival	2024-10-01	22:00:00.0000000	7	700	700	100	Concert	NULL
8	World cup of darts	2024-11-17	10:00:00.0000000	8	550	550	299	Sports	NULL
9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL
10	The cup and sau...	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

4. Write a SQL query to select events name partial match with ‘cup’.

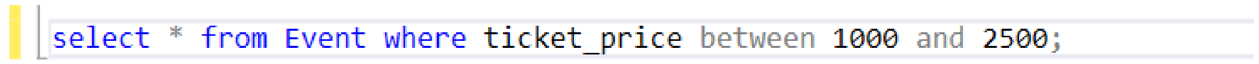


```sql
select * from Event where event_name like '%cup%';
```

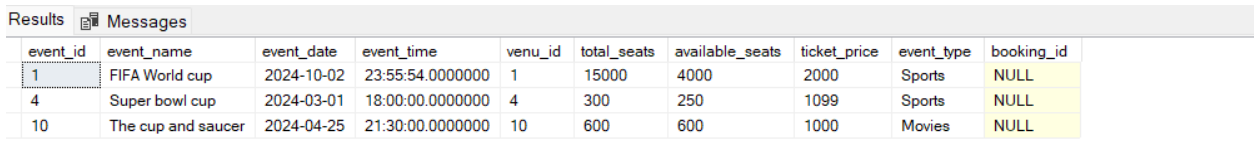


event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
4	Super bowl cup	2024-03-01	18:00:00.0000000	4	300	250	1099	Sports	NULL
6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
8	World cup of darts	2024-11-17	10:00:00.0000000	8	550	550	299	Sports	NULL
9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL
10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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



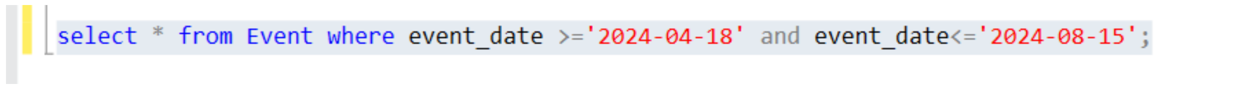
```sql
select * from Event where ticket_price between 1000 and 2500;
```



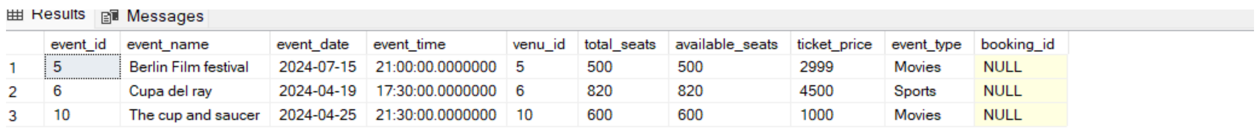
Results | Messages

event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
4	Super bowl cup	2024-03-01	18:00:00.0000000	4	300	250	1099	Sports	NULL
10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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



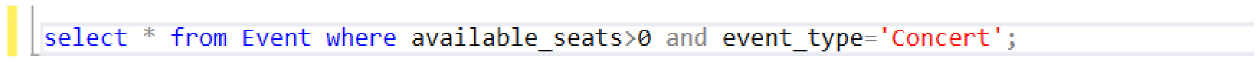
```sql
select * from Event where event_date >='2024-04-18' and event_date<='2024-08-15';
```



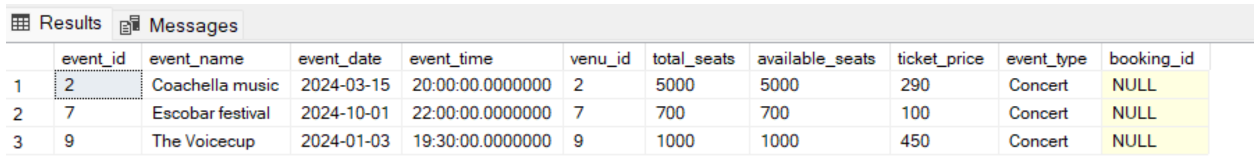
Results | Messages

	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	5	Berlin Film festival	2024-07-15	21:00:00.0000000	5	500	500	2999	Movies	NULL
2	6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
3	10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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



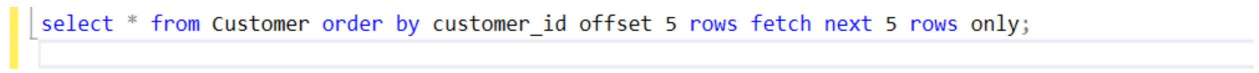
```sql
select * from Event where available_seats>0 and event_type='Concert';
```



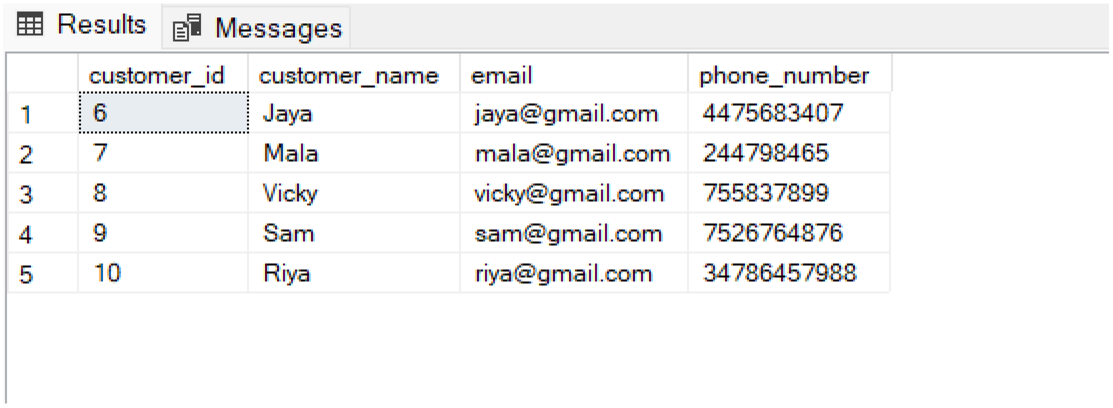
Results | Messages

	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	2	Coachella music	2024-03-15	20:00:00.0000000	2	5000	5000	290	Concert	NULL
2	7	Escobar festival	2024-10-01	22:00:00.0000000	7	700	700	100	Concert	NULL
3	9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL

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

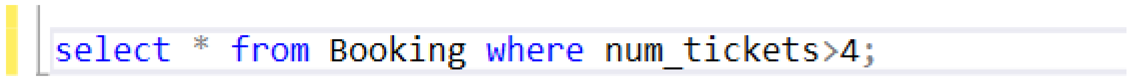


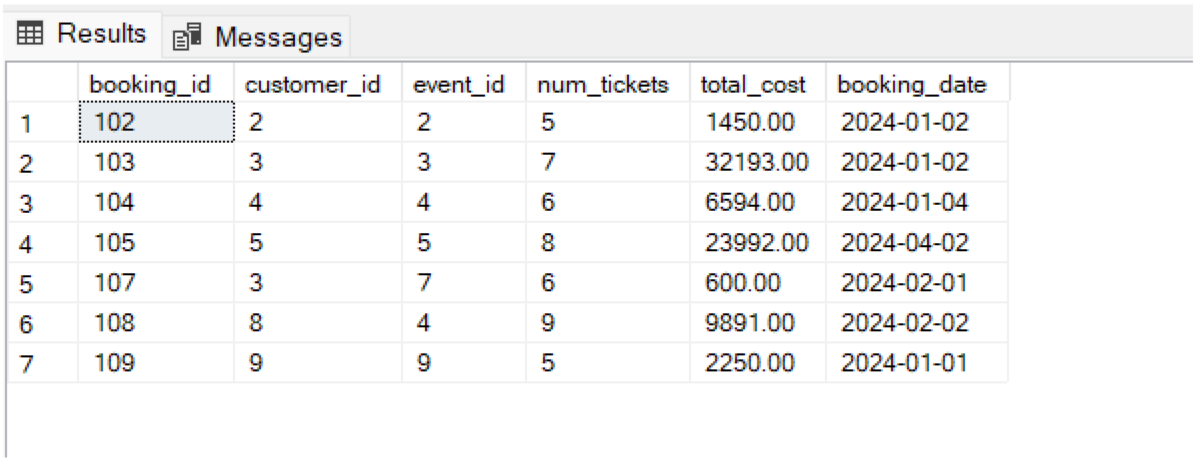
```sql
select * from Customer order by customer_id offset 5 rows fetch next 5 rows only;
```


Results Messages

	customer_id	customer_name	email	phone_number
1	6	Jaya	jaya@gmail.com	4475683407
2	7	Mala	mala@gmail.com	244798465
3	8	Vicky	vicky@gmail.com	755837899
4	9	Sam	sam@gmail.com	7526764876
5	10	Riya	riya@gmail.com	34786457988

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

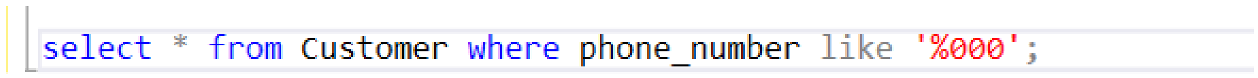

```sql
select * from Booking where num_tickets>4;
```

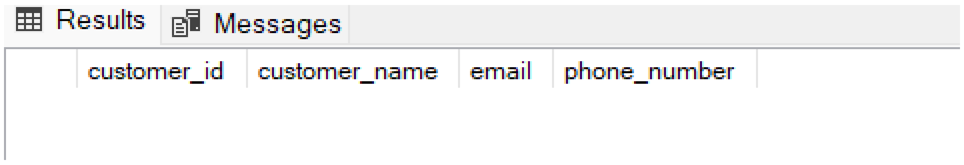

Results Messages

	booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
1	102	2	2	5	1450.00	2024-01-02
2	103	3	3	7	32193.00	2024-01-02
3	104	4	4	6	6594.00	2024-01-04
4	105	5	5	8	23992.00	2024-04-02
5	107	3	7	6	600.00	2024-02-01
6	108	8	4	9	9891.00	2024-02-02
7	109	9	9	5	2250.00	2024-01-01

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

There is no phone_number that ends with “**000**”.

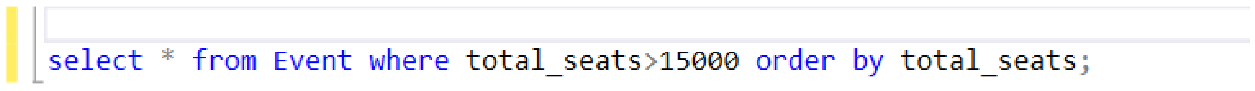

```sql
select * from Customer where phone_number like '%000';
```


Results Messages

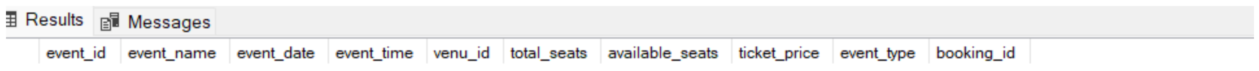
	customer_id	customer_name	email	phone_number

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

I have no event which have total_seats more than “15000”.



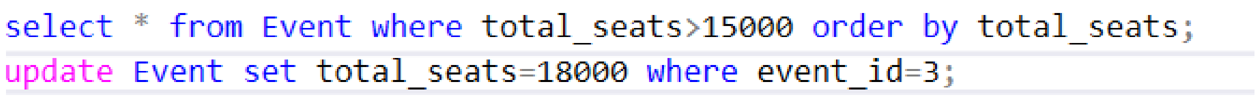
```
select * from Event where total_seats>15000 order by total_seats;
```



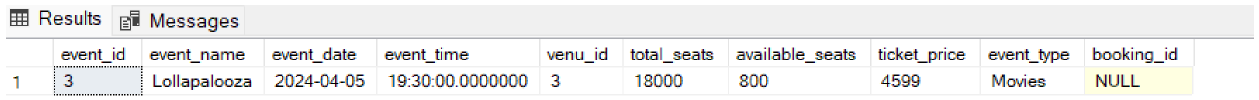
Results | Messages

event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id

After updating the Event table:



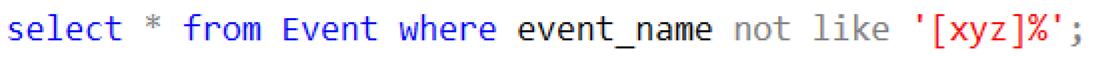
```
select * from Event where total_seats>15000 order by total_seats;
update Event set total_seats=18000 where event_id=3;
```



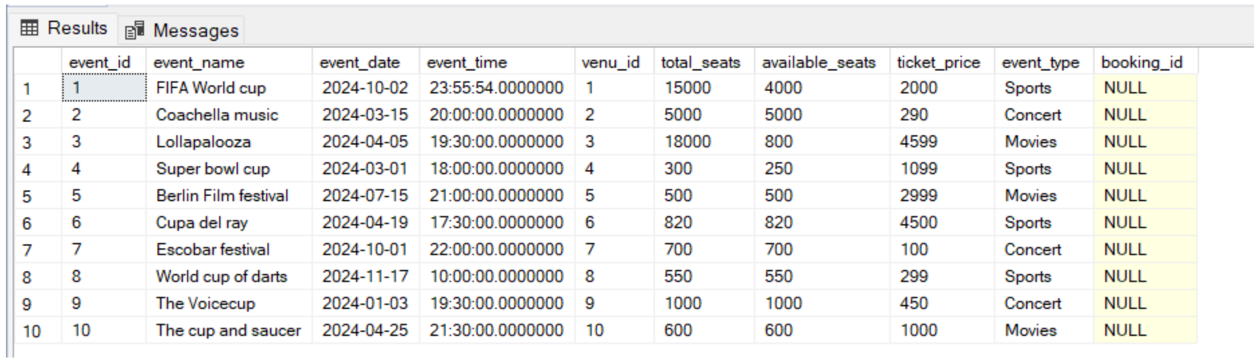
Results | Messages

	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	3	Lollapalooza	2024-04-05	19:30:00.0000000	3	18000	800	4599	Movies	NULL

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



```
select * from Event where event_name not like '[xyz]%';
```

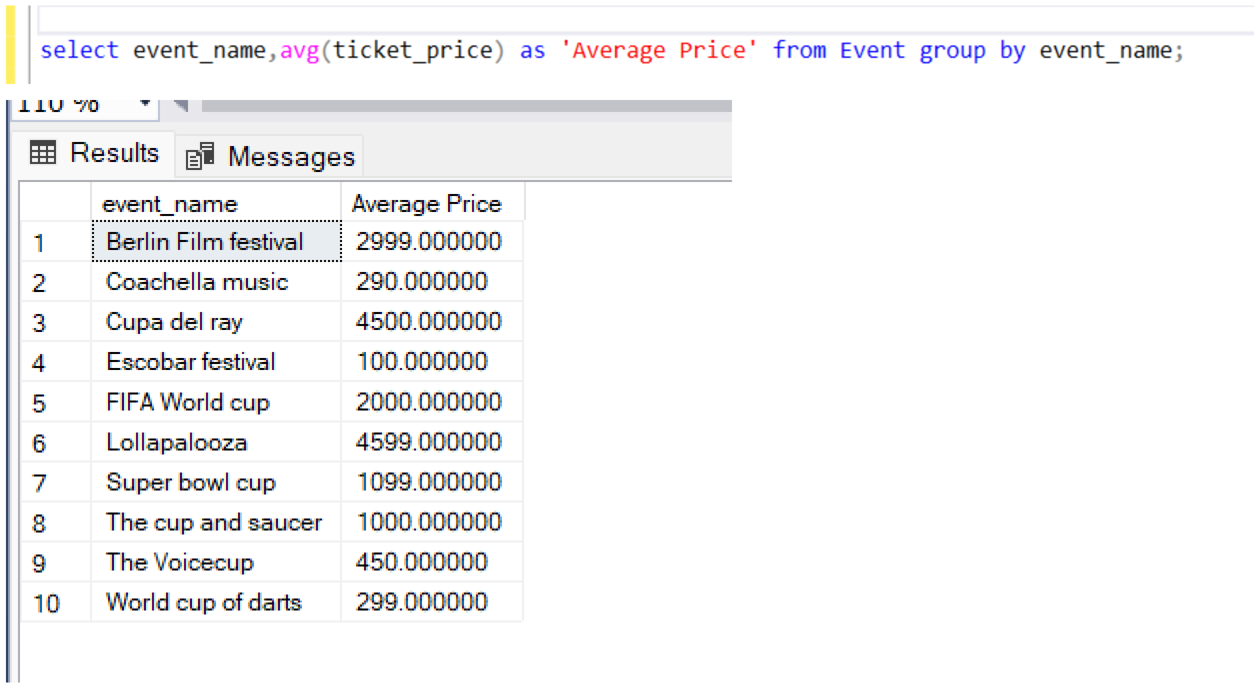


Results | Messages

	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
2	2	Coachella music	2024-03-15	20:00:00.0000000	2	5000	5000	290	Concert	NULL
3	3	Lollapalooza	2024-04-05	19:30:00.0000000	3	18000	800	4599	Movies	NULL
4	4	Super bowl cup	2024-03-01	18:00:00.0000000	4	300	250	1099	Sports	NULL
5	5	Berlin Film festival	2024-07-15	21:00:00.0000000	5	500	500	2999	Movies	NULL
6	6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
7	7	Escobar festival	2024-10-01	22:00:00.0000000	7	700	700	100	Concert	NULL
8	8	World cup of darts	2024-11-17	10:00:00.0000000	8	550	550	299	Sports	NULL
9	9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL
10	10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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

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



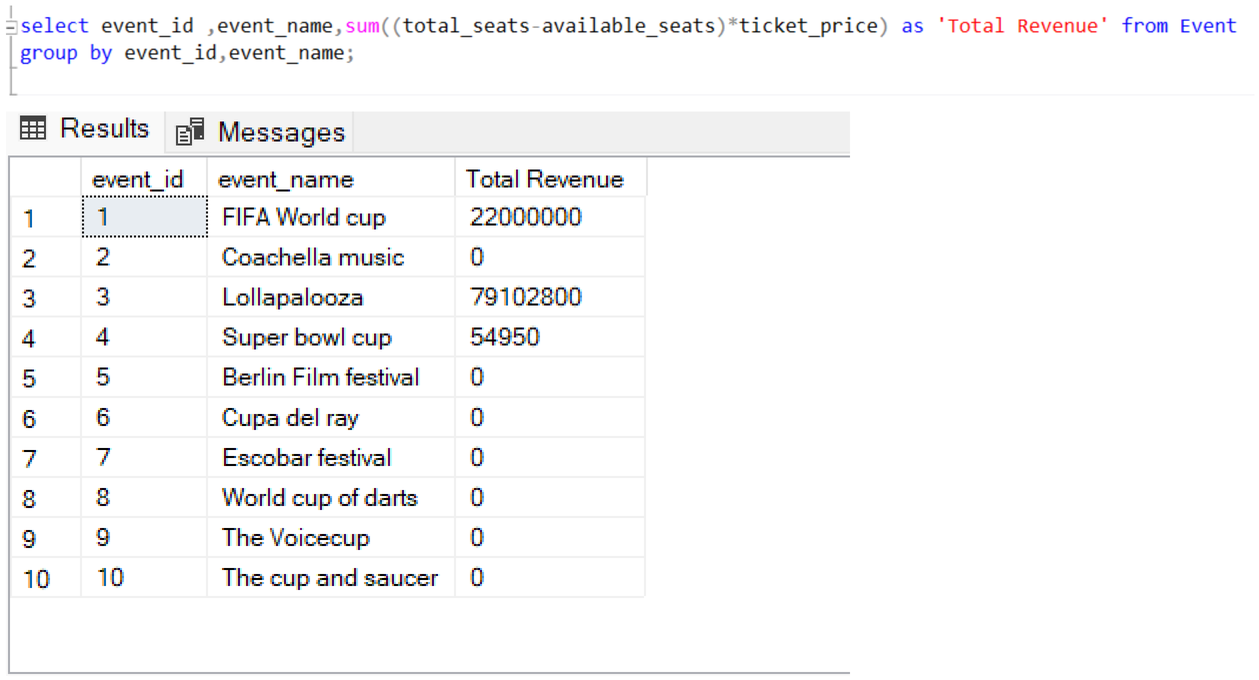
```
select event_name,avg(ticket_price) as 'Average Price' from Event group by event_name;
```

110 %

Results | Messages

	event_name	Average Price
1	Berlin Film festival	2999.000000
2	Coachella music	290.000000
3	Cupa del ray	4500.000000
4	Escobar festival	100.000000
5	FIFA World cup	2000.000000
6	Lollapalooza	4599.000000
7	Super bowl cup	1099.000000
8	The cup and saucer	1000.000000
9	The Voicecup	450.000000
10	World cup of darts	299.000000

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

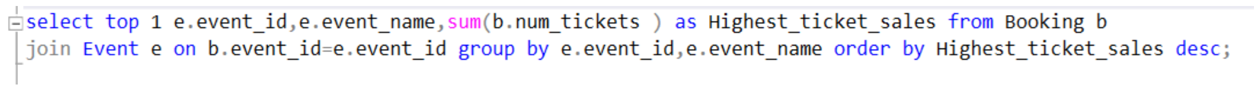


```
select event_id ,event_name,sum((total_seats-available_seats)*ticket_price) as 'Total Revenue' from Event
group by event_id,event_name;
```

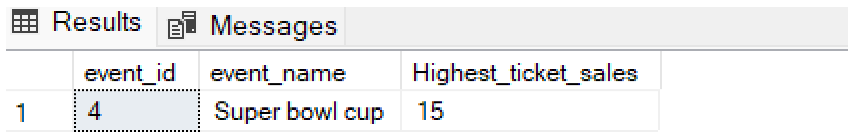
Results | Messages

	event_id	event_name	Total Revenue
1	1	FIFA World cup	22000000
2	2	Coachella music	0
3	3	Lollapalooza	79102800
4	4	Super bowl cup	54950
5	5	Berlin Film festival	0
6	6	Cupa del ray	0
7	7	Escobar festival	0
8	8	World cup of darts	0
9	9	The Voicecup	0
10	10	The cup and saucer	0

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

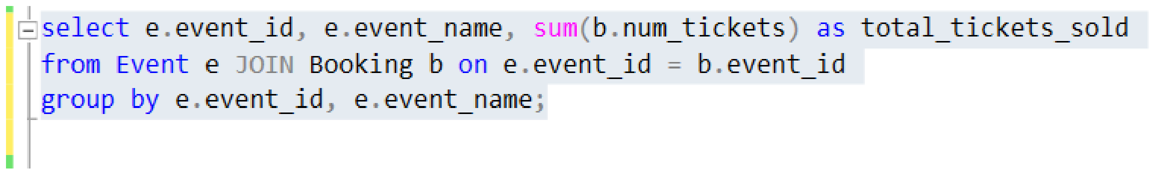


```sql
select top 1 e.event_id,e.event_name,sum(b.num_tickets ) as Highest_ticket_sales from Booking b
join Event e on b.event_id=e.event_id group by e.event_id,e.event_name order by Highest_ticket_sales desc;
```

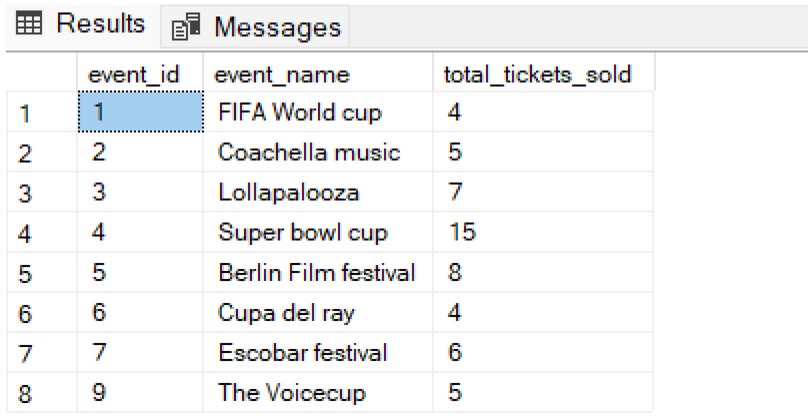


	event_id	event_name	Highest_ticket_sales
1	4	Super bowl cup	15

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

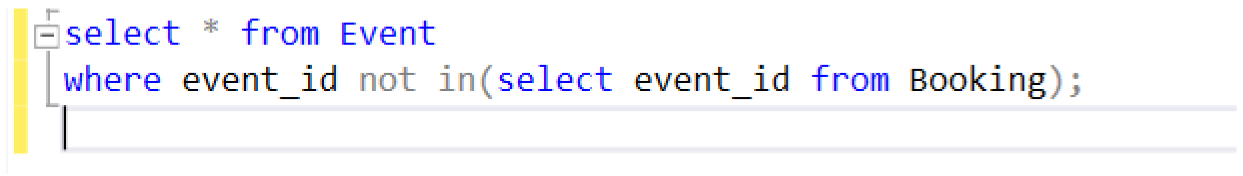


```sql
select e.event_id, e.event_name, sum(b.num_tickets) as total_tickets_sold
from Event e JOIN Booking b on e.event_id = b.event_id
group by e.event_id, e.event_name;
```

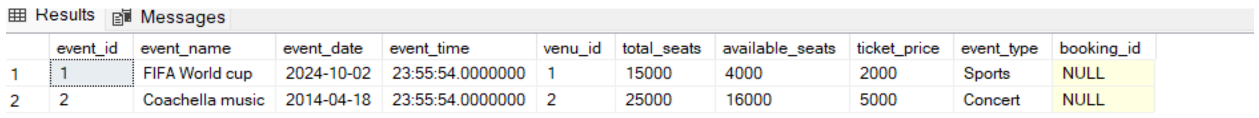


	event_id	event_name	total_tickets_sold
1	1	FIFA World cup	4
2	2	Coachella music	5
3	3	Lollapalooza	7
4	4	Super bowl cup	15
5	5	Berlin Film festival	8
6	6	Cupa del ray	4
7	7	Escobar festival	6
8	9	The Voicecup	5

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



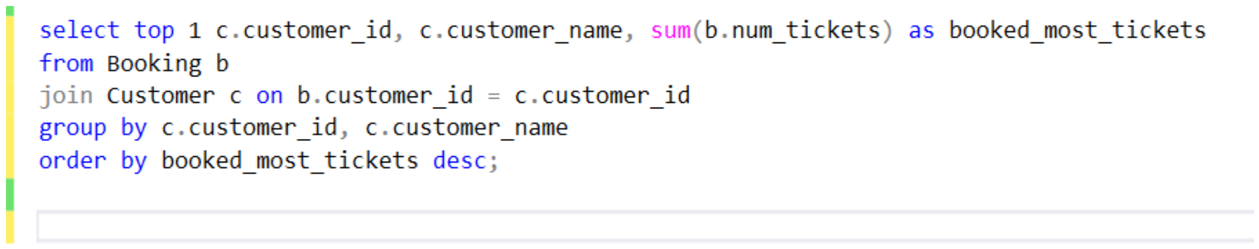
```sql
select * from Event
where event_id not in(select event_id from Booking);
```



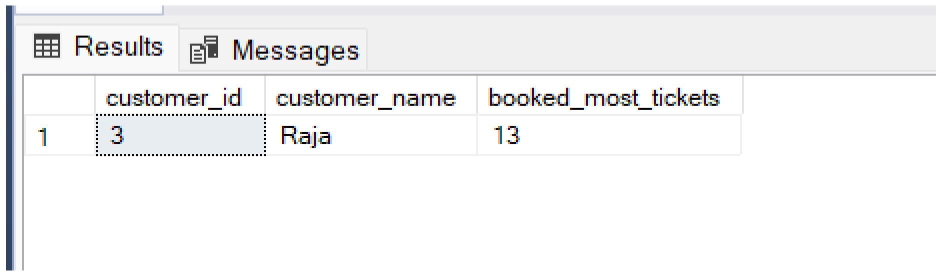
Results | Messages

	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
2	2	Coachella music	2014-04-18	23:55:54.0000000	2	25000	16000	5000	Concert	NULL

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



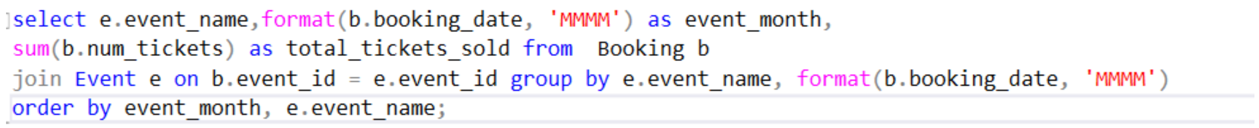
```
select top 1 c.customer_id, c.customer_name, sum(b.num_tickets) as booked_most_tickets
from Booking b
join Customer c on b.customer_id = c.customer_id
group by c.customer_id, c.customer_name
order by booked_most_tickets desc;
```



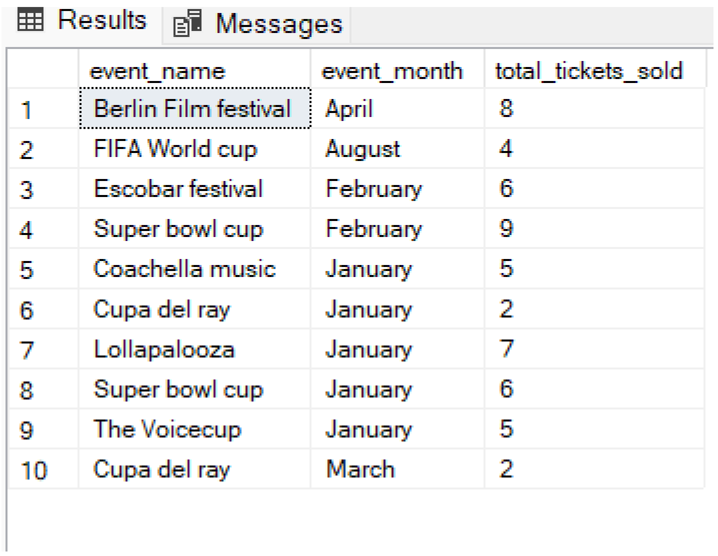
Results | Messages

	customer_id	customer_name	booked_most_tickets
1	3	Raja	13

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



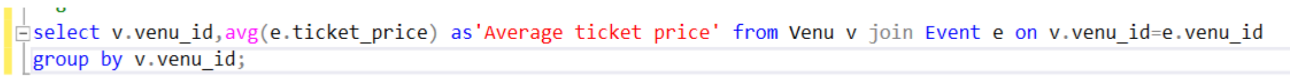
```
select e.event_name,format(b.booking_date, 'MMMM') as event_month,
sum(b.num_tickets) as total_tickets_sold from  Booking b
join Event e on b.event_id = e.event_id group by e.event_name, format(b.booking_date, 'MMMM')
order by event_month, e.event_name;
```



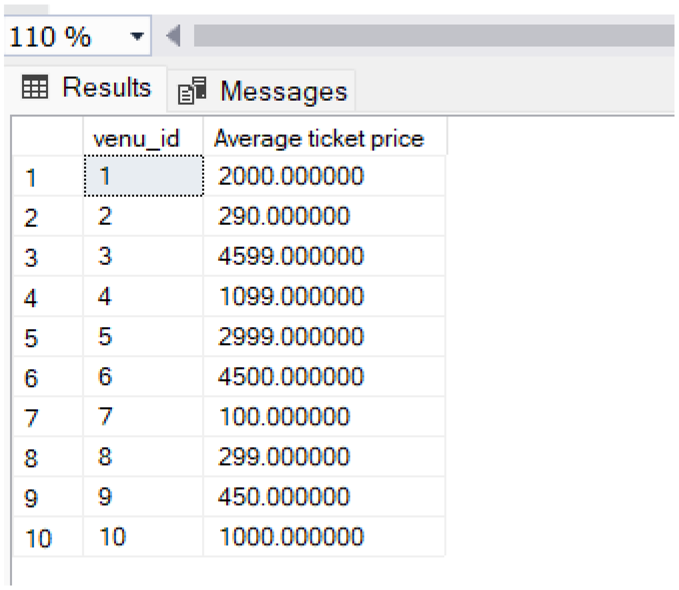
Results | Messages

	event_name	event_month	total_tickets_sold
1	Berlin Film festival	April	8
2	FIFA World cup	August	4
3	Escobar festival	February	6
4	Super bowl cup	February	9
5	Coachella music	January	5
6	Cupa del ray	January	2
7	Lollapalooza	January	7
8	Super bowl cup	January	6
9	The Voicecup	January	5
10	Cupa del ray	March	2

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

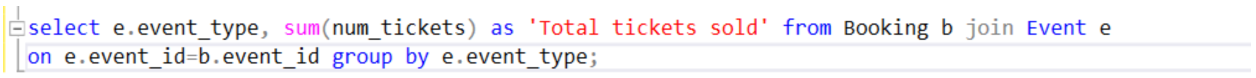


```
select v.venu_id,avg(e.ticket_price) as'Average ticket price' from Venu v join Event e on v.venu_id=e.venu_id
group by v.venu_id;
```

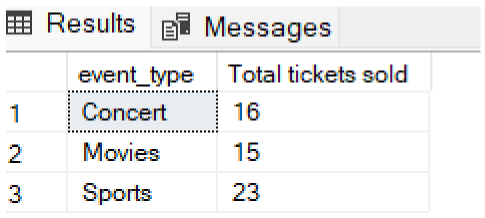


	venu_id	Average ticket price
1	1	2000.000000
2	2	290.000000
3	3	4599.000000
4	4	1099.000000
5	5	2999.000000
6	6	4500.000000
7	7	100.000000
8	8	299.000000
9	9	450.000000
10	10	1000.000000

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

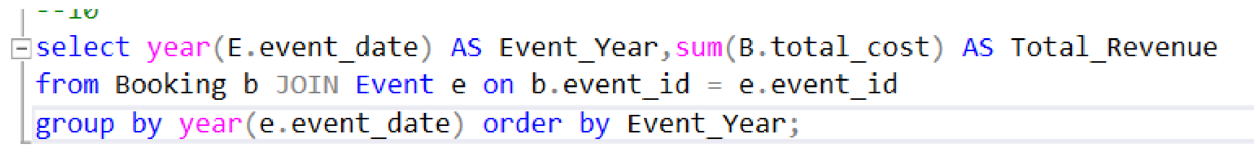


```
select e.event_type, sum(num_tickets) as 'Total tickets sold' from Booking b join Event e
on e.event_id=b.event_id group by e.event_type;
```

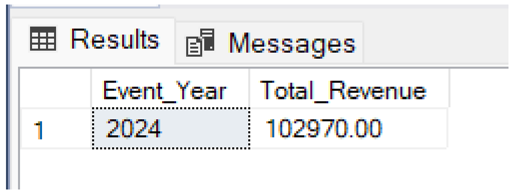


	event_type	Total tickets sold
1	Concert	16
2	Movies	15
3	Sports	23

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

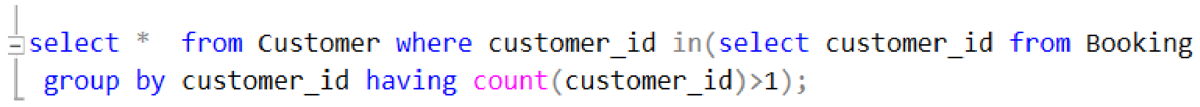


```sql
--10
select year(E.event_date) AS Event_Year,sum(B.total_cost) AS Total_Revenue
from Booking b JOIN Event e on b.event_id = e.event_id
group by year(e.event_date) order by Event_Year;
```

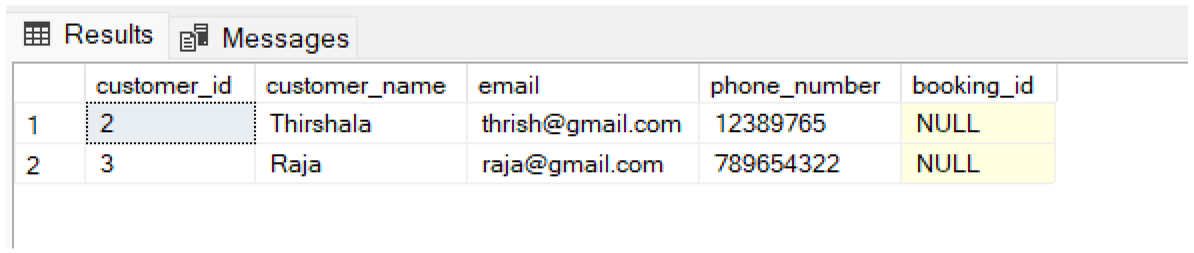


	Event_Year	Total_Revenue
1	2024	102970.00

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

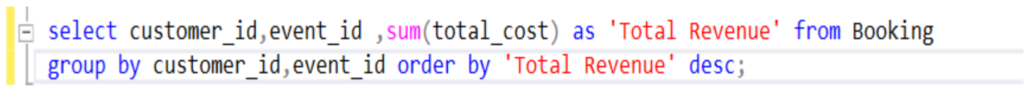


```sql
select *  from Customer where customer_id in(select customer_id from Booking
 group by customer_id having count(customer_id)>1);
```

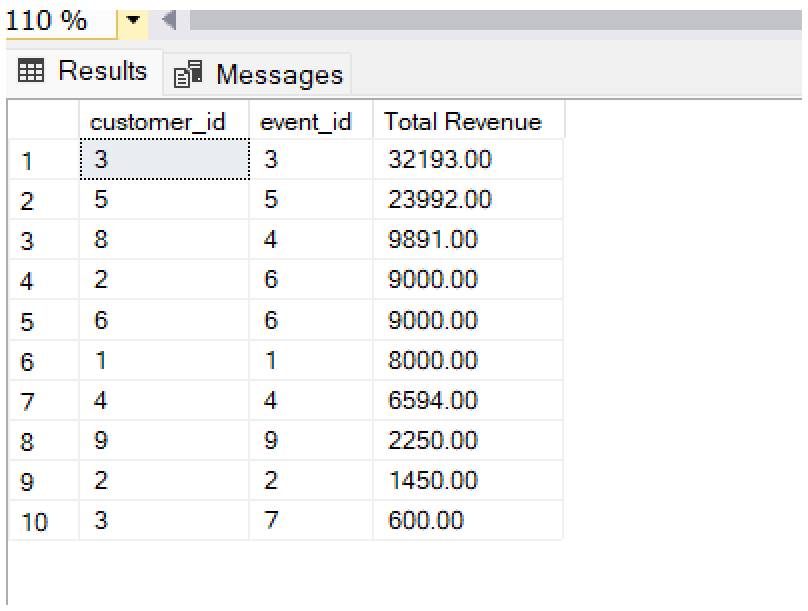


	customer_id	customer_name	email	phone_number	booking_id
1	2	Thirshala	thrish@gmail.com	12389765	NULL
2	3	Raja	raja@gmail.com	789654322	NULL

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



```
select customer_id,event_id ,sum(total_cost) as 'Total Revenue' from Booking
group by customer_id,event_id order by 'Total Revenue' desc;
```

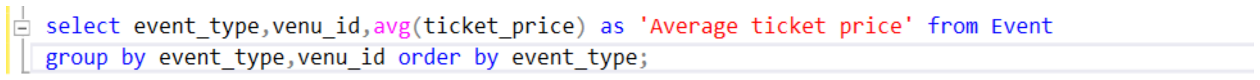


110 %

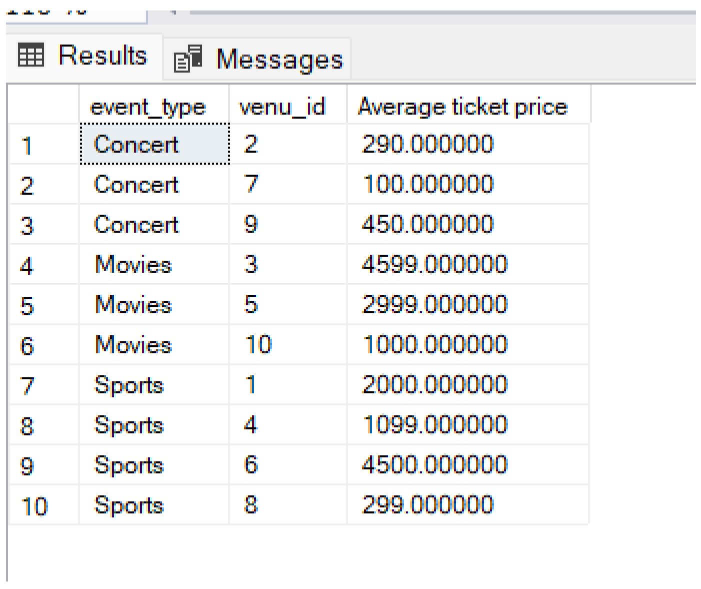
Results Messages

	customer_id	event_id	Total Revenue
1	3	3	32193.00
2	5	5	23992.00
3	8	4	9891.00
4	2	6	9000.00
5	6	6	9000.00
6	1	1	8000.00
7	4	4	6594.00
8	9	9	2250.00
9	2	2	1450.00
10	3	7	600.00

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



```
select event_type,venu_id,avg(ticket_price) as 'Average ticket price' from Event
group by event_type,venu_id order by event_type;
```

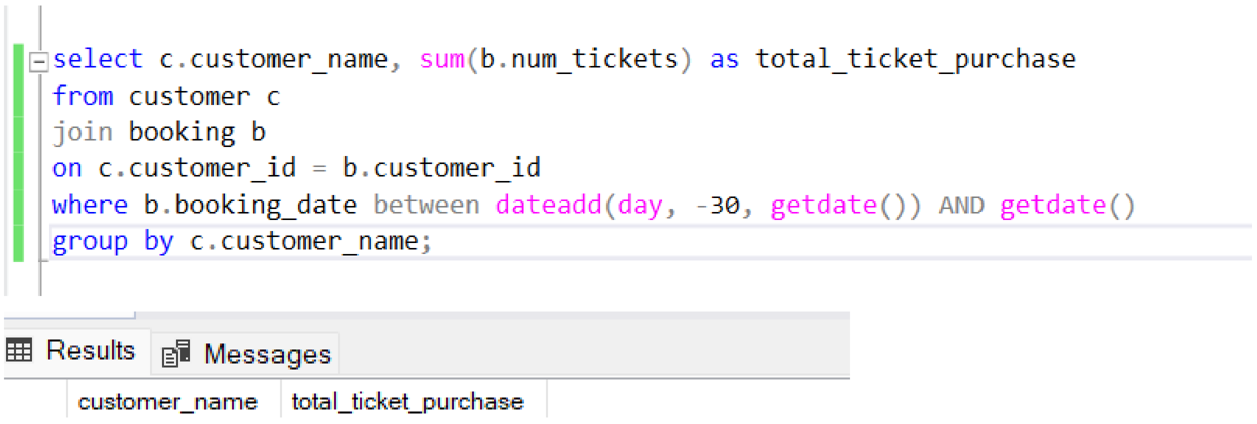


Results Messages

	event_type	venu_id	Average ticket price
1	Concert	2	290.000000
2	Concert	7	100.000000
3	Concert	9	450.000000
4	Movies	3	4599.000000
5	Movies	5	2999.000000
6	Movies	10	1000.000000
7	Sports	1	2000.000000
8	Sports	4	1099.000000
9	Sports	6	4500.000000
10	Sports	8	299.000000

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

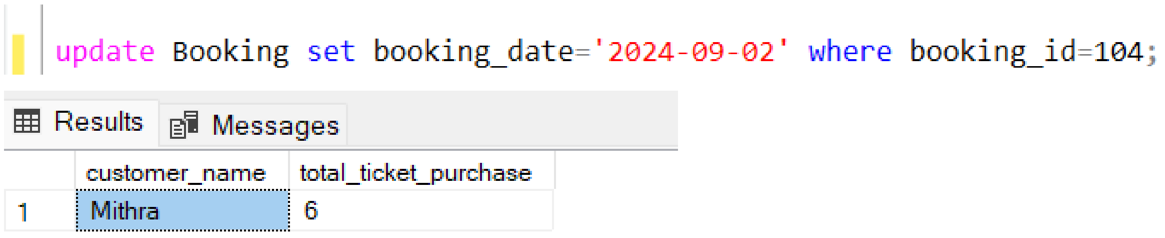
There is no record to list the users purchased the tickets from the past 30 days



```
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;
```

customer_name	total_ticket_purchase

After updating a table to see some result:

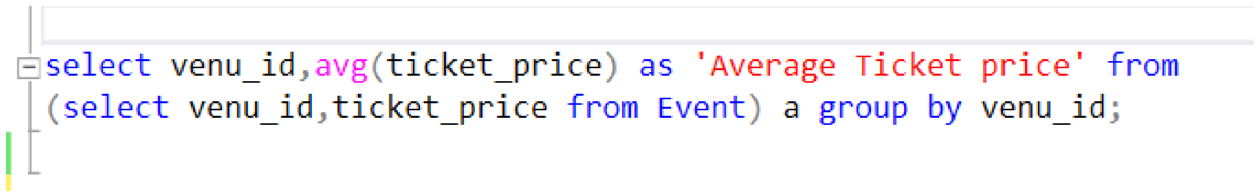


```
update Booking set booking_date='2024-09-02' where booking_id=104;
```

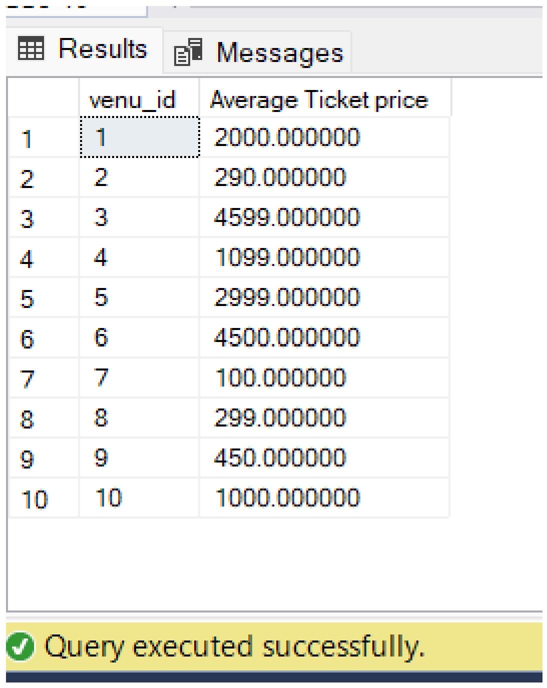
	customer_name	total_ticket_purchase
1	Mithra	6

Tasks 4: Subquery and its types

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



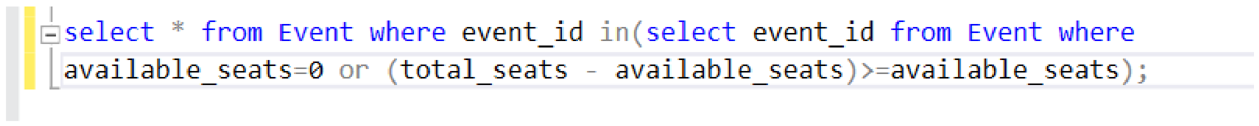
```
select venu_id,avg(ticket_price) as 'Average Ticket price' from
(select venu_id,ticket_price from Event) a group by venu_id;
```



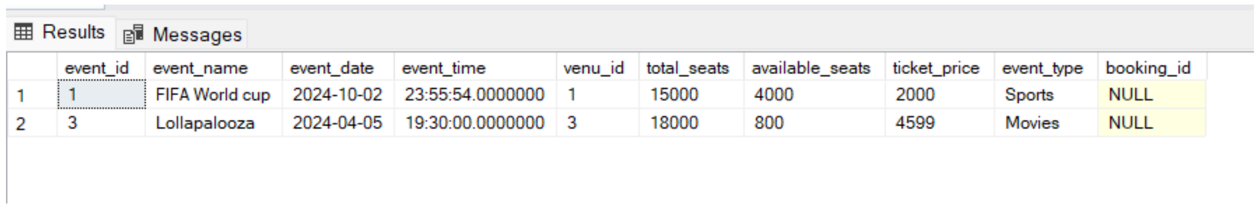
	venu_id	Average Ticket price
1	1	2000.000000
2	2	290.000000
3	3	4599.000000
4	4	1099.000000
5	5	2999.000000
6	6	4500.000000
7	7	100.000000
8	8	299.000000
9	9	450.000000
10	10	1000.000000

Query executed successfully.

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

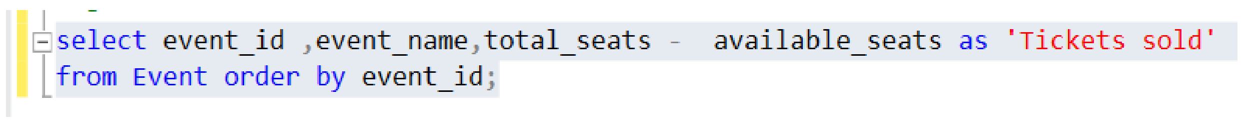


```
select * from Event where event_id in(select event_id from Event where
available_seats=0 or (total_seats - available_seats)>=available_seats);
```

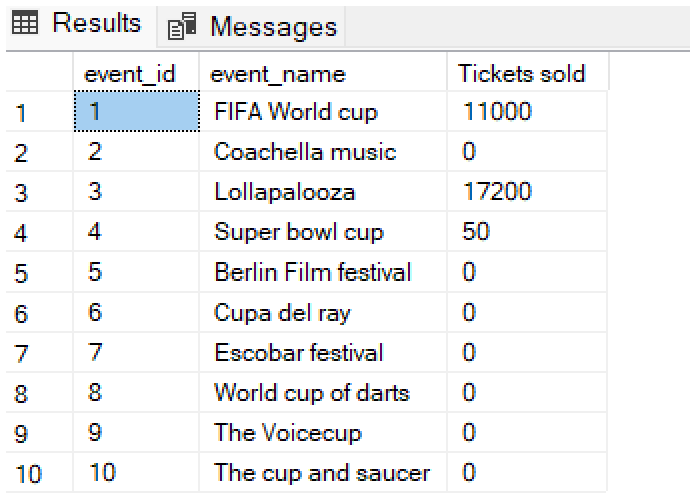


	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	FIFA World cup	2024-10-02	23:55:54.0000000	1	15000	4000	2000	Sports	NULL
2	3	Lollapalooza	2024-04-05	19:30:00.0000000	3	18000	800	4599	Movies	NULL

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



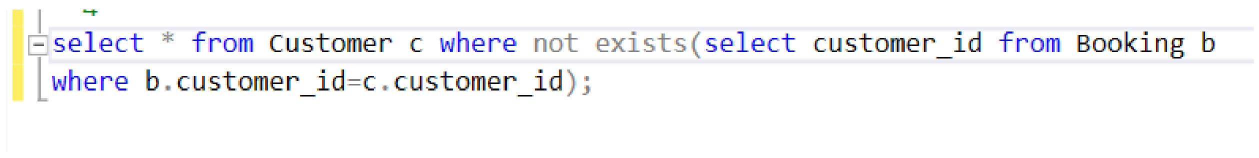
```
select event_id ,event_name,total_seats -  available_seats as 'Tickets sold'
from Event order by event_id;
```



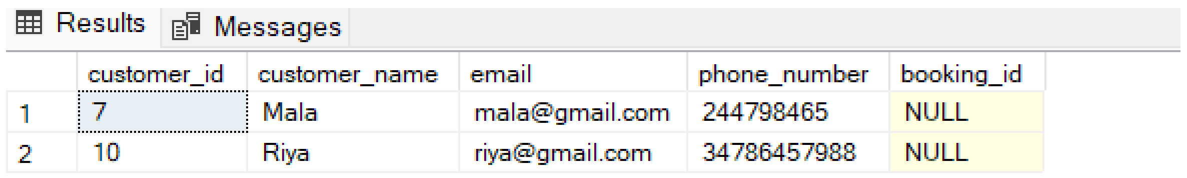
Results | Messages

	event_id	event_name	Tickets sold
1	1	FIFA World cup	11000
2	2	Coachella music	0
3	3	Lollapalooza	17200
4	4	Super bowl cup	50
5	5	Berlin Film festival	0
6	6	Cupa del ray	0
7	7	Escobar festival	0
8	8	World cup of darts	0
9	9	The Voicecup	0
10	10	The cup and saucer	0

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



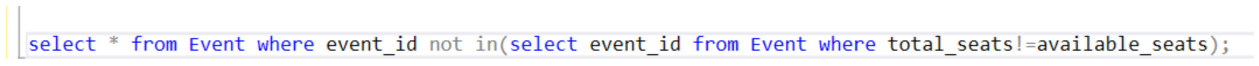
```
select * from Customer c where not exists(select customer_id from Booking b
where b.customer_id=c.customer_id);
```



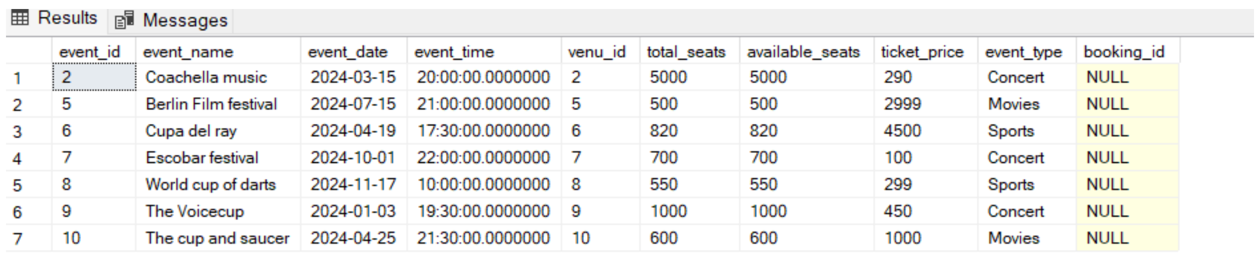
Results | Messages

	customer_id	customer_name	email	phone_number	booking_id
1	7	Mala	mala@gmail.com	244798465	NULL
2	10	Riya	riya@gmail.com	34786457988	NULL

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



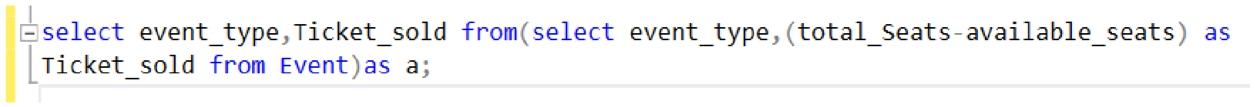
```
select * from Event where event_id not in(select event_id from Event where total_seats!=available_seats);
```



Results | Messages

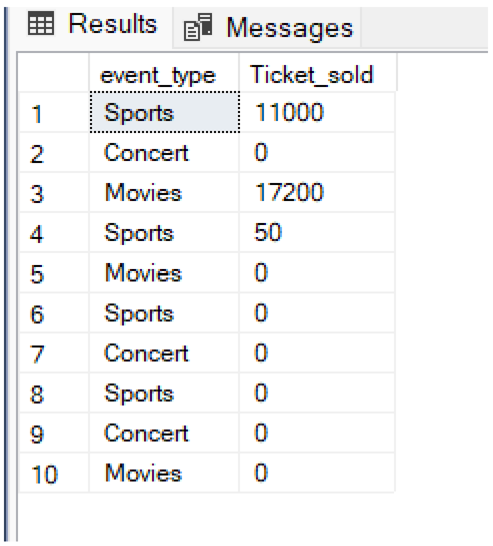
	event_id	event_name	event_date	event_time	venu_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	2	Coachella music	2024-03-15	20:00:00.0000000	2	5000	5000	290	Concert	NULL
2	5	Berlin Film festival	2024-07-15	21:00:00.0000000	5	500	500	2999	Movies	NULL
3	6	Cupa del ray	2024-04-19	17:30:00.0000000	6	820	820	4500	Sports	NULL
4	7	Escobar festival	2024-10-01	22:00:00.0000000	7	700	700	100	Concert	NULL
5	8	World cup of darts	2024-11-17	10:00:00.0000000	8	550	550	299	Sports	NULL
6	9	The Voicecup	2024-01-03	19:30:00.0000000	9	1000	1000	450	Concert	NULL
7	10	The cup and saucer	2024-04-25	21:30:00.0000000	10	600	600	1000	Movies	NULL

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



```sql
select event_type,Ticket_sold from(select event_type,(total_Seats-available_seats) as
Ticket_sold from Event)as a;
```

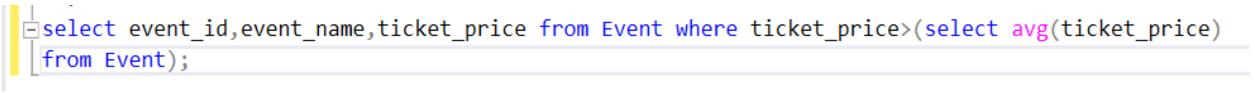
In SQL, when we use a subquery in the FROM clause, you must give it an alias (as a).



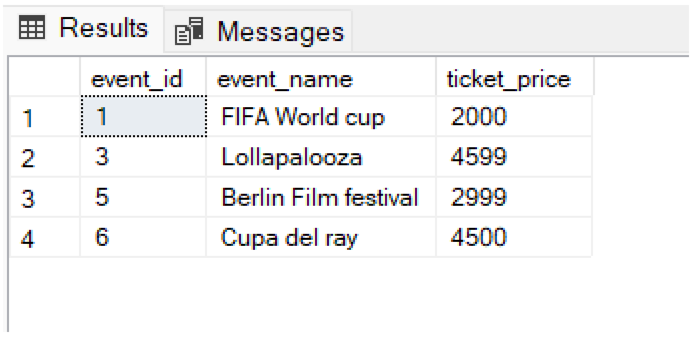
Results | Messages

	event_type	Ticket_sold
1	Sports	11000
2	Concert	0
3	Movies	17200
4	Sports	50
5	Movies	0
6	Sports	0
7	Concert	0
8	Sports	0
9	Concert	0
10	Movies	0

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



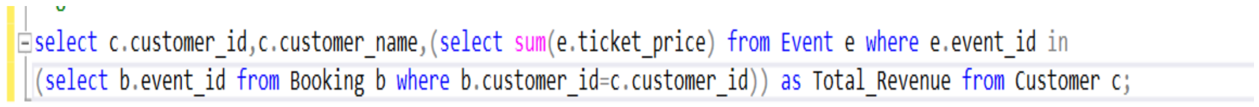
```sql
select event_id,event_name,ticket_price from Event where ticket_price>(select avg(ticket_price)
from Event);
```



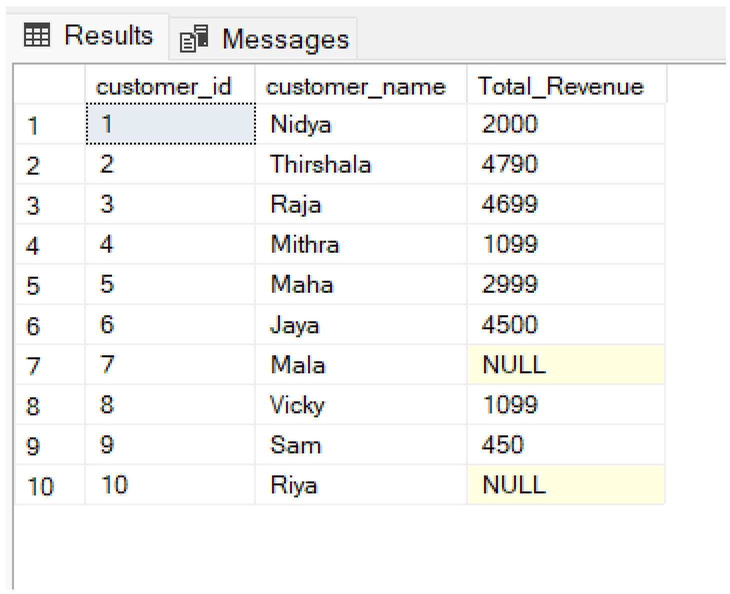
Results | Messages

	event_id	event_name	ticket_price
1	1	FIFA World cup	2000
2	3	Lollapalooza	4599
3	5	Berlin Film festival	2999
4	6	Cupa del ray	4500

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



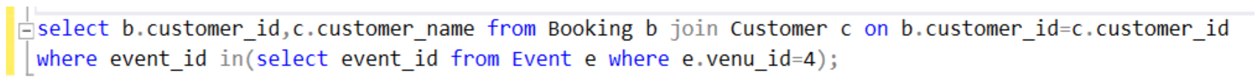
```sql
select c.customer_id,c.customer_name,(select sum(e.ticket_price) from Event e where e.event_id in
(select b.event_id from Booking b where b.customer_id=c.customer_id)) as Total_Revenue from Customer c;
```



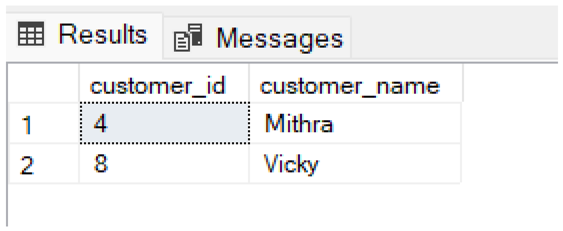
Results | Messages

	customer_id	customer_name	Total_Revenue
1	1	Nidya	2000
2	2	Thirshala	4790
3	3	Raja	4699
4	4	Mithra	1099
5	5	Maha	2999
6	6	Jaya	4500
7	7	Mala	NULL
8	8	Vicky	1099
9	9	Sam	450
10	10	Riya	NULL

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



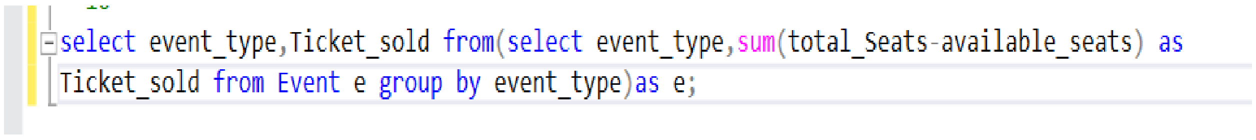
```sql
select b.customer_id,c.customer_name from Booking b join Customer c on b.customer_id=c.customer_id
where event_id in(select event_id from Event e where e.venu_id=4);
```



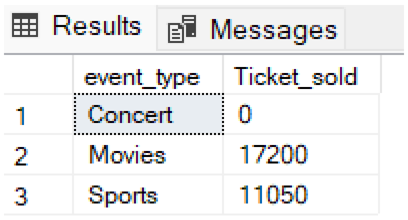
Results | Messages

	customer_id	customer_name
1	4	Mithra
2	8	Vicky

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



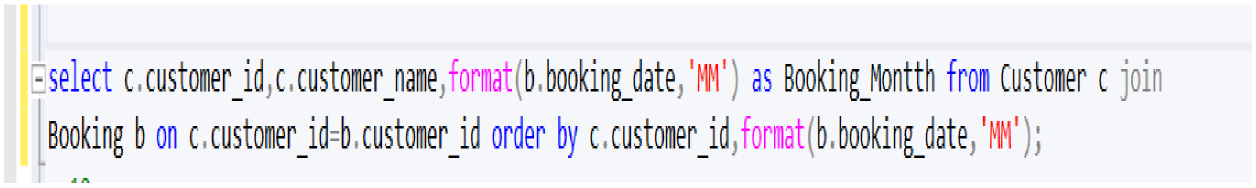
```sql
select event_type,Ticket_sold from(select event_type,sum(total_Seats-available_seats) as
Ticket_sold from Event e group by event_type)as e;
```



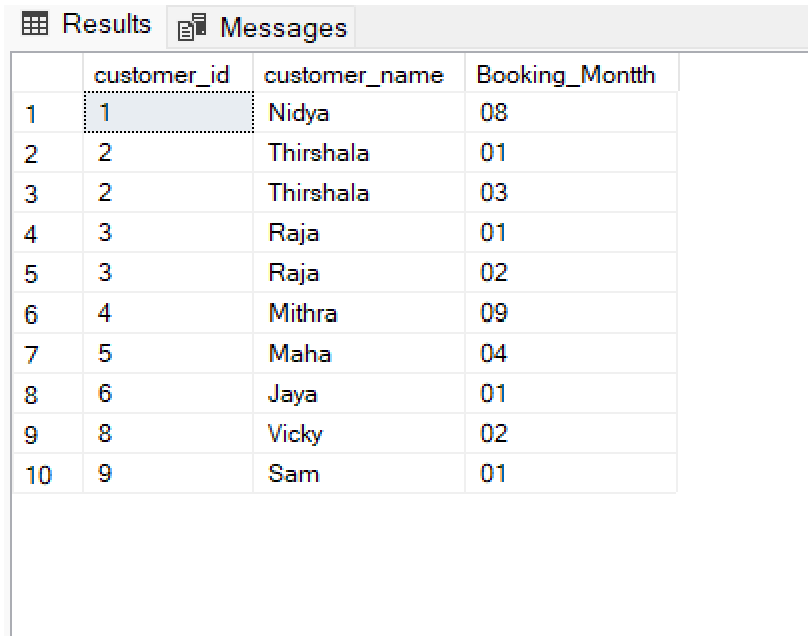
Results | Messages

	event_type	Ticket_sold
1	Concert	0
2	Movies	17200
3	Sports	11050

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



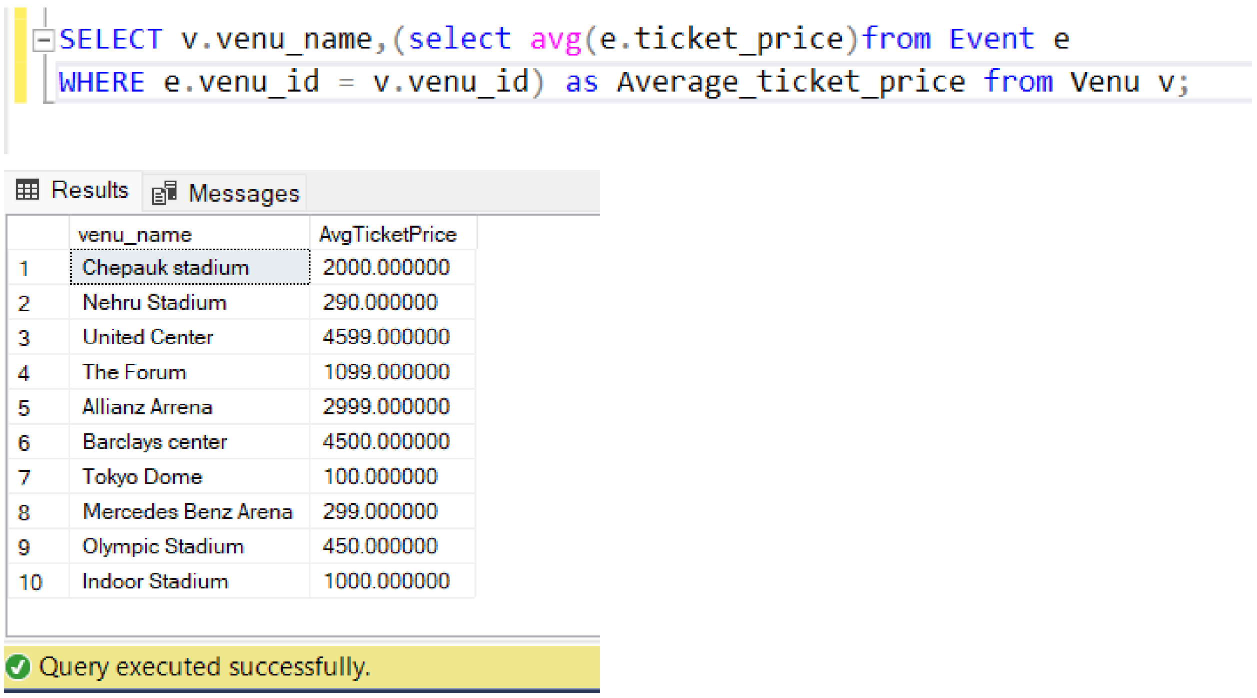
```sql
select c.customer_id,c.customer_name,format(b.booking_date,'MM') as Booking_Montth from Customer c join
Booking b on c.customer_id=b.customer_id order by c.customer_id,format(b.booking_date,'MM');
```



Results | Messages

	customer_id	customer_name	Booking_Montth
1	1	Nidya	08
2	2	Thirshala	01
3	2	Thirshala	03
4	3	Raja	01
5	3	Raja	02
6	4	Mithra	09
7	5	Maha	04
8	6	Jaya	01
9	8	Vicky	02
10	9	Sam	01

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



```
SELECT v.venu_name,(select avg(e.ticket_price)from Event e
WHERE e.venu_id = v.venu_id) as Average_ticket_price from Venu v;
```

Results Messages

	venu_name	AvgTicketPrice
1	Chepauk stadium	2000.000000
2	Nehru Stadium	290.000000
3	United Center	4599.000000
4	The Forum	1099.000000
5	Allianz Arrena	2999.000000
6	Barclays center	4500.000000
7	Tokyo Dome	100.000000
8	Mercedes Benz Arena	299.000000
9	Olympic Stadium	450.000000
10	Indoor Stadium	1000.000000

Query executed successfully.