Assignment Title:

Ticket Booking System: MSSQL Assignment 5

Submitted By:

Subrat Shukla, Python Batch 1

Date of Submission:

25-09-2024

Ticket Booking System: SQL Assignment 5

Task1: Database Design

1. Create the database named "TicketBookingSystem"

```
create database TicketBookingSystem;

use TicketBookingSystem;
```

```
Messages
Commands completed successfully.

Completion time: 2024-09-19T11:57:26.7251844+05:30
```

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

• Venu • Event • Customers • Booking

```
create table Venu(
    venue_id int primary key,
    venue_name varchar(255),
    address varchar(255)
);
```

```
Messages
Commands completed successfully.

Completion time: 2024-09-19T12:10:38.1909638+05:30
```

```
create table Event(
    event_id int primary key,
    event_name varchar(255),
    event_date date,
    event_time time,
    venue_id int,
    total_seats int,
    available_seats int,
    ticket_price decimal(10,2),
    event_type text,
    booking_id int,
    foreign key(venue_id) references Venu(venue_id)
);
```

Messages

```
Commands completed successfully.

Completion time: 2024-09-19T12:27:12.3893186+05:30
```

```sql
create table Customer(
    customer_id int primary key,
    customer_name varchar(255),
    email varchar(255),
    phone_number varchar(15),
    booking_id int
);
```

Messages

```
Commands completed successfully.

Completion time: 2024-09-19T12:36:42.4308838+05:30
```

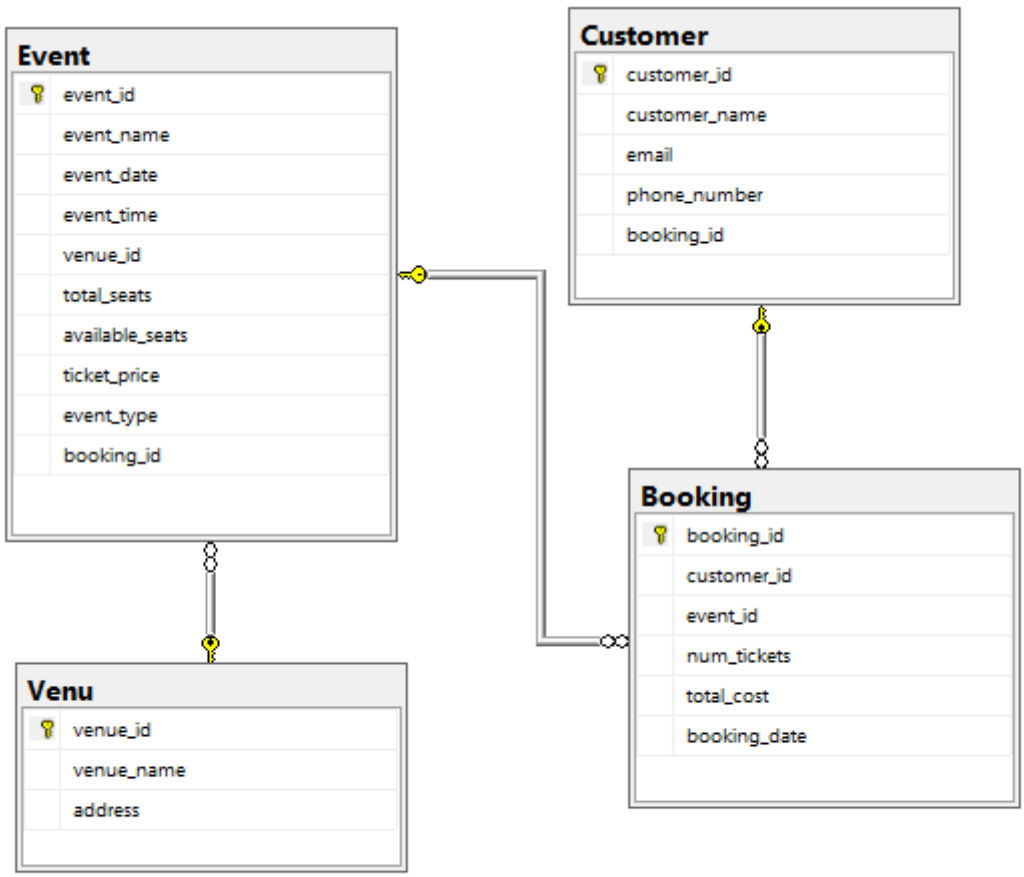
```sql
create table Booking(
    booking_id int primary key,
    customer_id int,
    event_id int,
    num_tickets int,
    total_cost decimal(10,2),
    booking_date date,
    foreign key(customer_id) references Customer(customer_id),
    foreign key(event_id) references Event(event_id)
);
```

Messages

```
Commands completed successfully.

Completion time: 2024-09-19T12:40:15.1388713+05:30
```

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



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

Venu Table:

Primary Key: venue_id :ensuring each venue has a unique identifier.

Event Table:

Primary Key: event_id uniquely identifies each event.

Foreign Key: venue_id references the primary key venue_id in the "Venu" table.

Customer Table:

Primary Key: customer_id uniquely identifies each customer.

Foreign Key: booking_id references the primary key booking_id in the "Booking" table, connecting customers to their bookings.

Booking Table:

Primary Key: booking_id uniquely identifies each booking.

Foreign Keys: customer_id references the primary key customer_id in the "Customer" table, establishing a link between bookings and customers.

event_id references the primary key event_id in the "Event" table, linking bookings to specific events.

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

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

```
insert into Venu(venue_id, venue_name, address) values
    (1, 'Jaipur Convention Center', 'JLN Jaipur'),
    (2, 'Mansarovar Sports Arena', 'Mansarovar Jaipur'),
    (3, 'Udaipur Music Hall', 'Gandhipur Udaipur'),
    (4, 'Trishul Event Plaza', 'Vaishali Nagar, Jodhpur'),
    (5, 'Rajmandir Hall', 'MI Road, Jaipur'),
    (6, 'Vellore Entertainment Complex', 'Katpadi, Vellore'),
    (7, 'Birla Auditorium', 'Raja Park, Ajmer'),
    (8, 'Erode Cultural Center', 'Gandhiji Road, Erode'),
    (9, 'RajMandir Convention Hall', 'City Park, Udaipur'),
    (10, 'JC Event Centre', 'Sitapura, Agra');

select *from Venu;
```

Results | Messages

	venue_id	venue_name	address
1	1	Jaipur Convention Center	JLN Jaipur
2	2	Mansarovar Sports Arena	Mansarovar Jaipur
3	3	Udaipur Music Hall	Gandhipur Udaipur
4	4	Trishul Event Plaza	Vaishali Nagar, Jodhpur
5	5	Rajmandir Hall	MI Road, Jaipur
6	6	Vellore Entertainment Complex	Katpadi, Vellore
7	7	Birla Auditorium	Raja Park, Ajmer
8	8	Erode Cultural Center	Gandhiji Road, Erode
9	9	RajMandir Convention Hall	City Park, Udaipur
10	10	JC Event Centre	Sitapura, Agra

```
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, 'Jaipur Film Festival', '2024-09-15', '18:00:00', 1, 500, 450, 15.00, 'Movie', 1),
(2, 'Jaipur Cricket Match', '2024-10-10', '15:30:00', 2, 1000, 800, 25.00, 'Sports', 2),
(3, 'Udaipur Music Concert', '2024-11-05', '20:00:00', 3, 800, 700, 20.00, 'Concert', 3),
(4, 'Trishul Dance Show', '2024-11-20', '19:30:00', 4, 600, 550, 18.00, 'Concert', 4),
(5, 'High Act', '2024-10-15', '18:00:00', 5, 1200, 1000, 10.00, 'Drama', 5),
(6, 'Vellore Movie Night', '2024-09-08', '21:00:00', 6, 300, 280, 12.00, 'Movie', 6),
(7, 'Cultural Festival', '2024-10-17', '17:00:00', 7, 700, 650, 22.00, 'Concert', 7),
(8, 'Erode Football Championship', '2024-12-12', '16:45:00', 8, 1500, 1300, 30.00, 'Sports', 8),
(9, 'Rajmandir Art Exhibition', '2024-11-30', '10:30:00', 9, 400, 380, 8.00, 'Concert', 9),
(10, 'JC Comedy Show', '2024-11-18', '19:00:00', 10, 250, 230, 15.00, 'Concert', 10);
```

Results | Messages

	event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	event_type	booking_id
1	1	Jaipur Film Festival	2024-09-15	18:00:00.0000000	1	500	450	15.00	Movie	1
2	2	Jaipur Cricket Match	2024-10-10	15:30:00.0000000	2	1000	800	25.00	Sports	2
3	3	Udaipur Music Concert	2024-11-05	20:00:00.0000000	3	800	700	20.00	Concert	3
4	4	Trishul Dance Show	2024-11-20	19:30:00.0000000	4	600	550	18.00	Concert	4
5	5	High Act	2024-10-15	18:00:00.0000000	5	1200	1000	10.00	Drama	5
6	6	Vellore Movie Night	2024-09-08	21:00:00.0000000	6	300	280	12.00	Movie	6
7	7	Cultural Festival	2024-10-17	17:00:00.0000000	7	700	650	22.00	Concert	7
8	8	Erode Football Championship	2024-12-12	16:45:00.0000000	8	1500	1300	30.00	Sports	8
9	9	Rajmandir Art Exhibition	2024-11-30	10:30:00.0000000	9	400	380	8.00	Concert	9
10	10	JC Comedy Show	2024-11-18	19:00:00.0000000	10	250	230	15.00	Concert	10

Query executed successfully. | DESKTOP-MKS6P3G (16.0 RTM) | DESKTOP-MKS6P3G\Shubh

```sql
insert into Customer(customer_id, customer_name, email, phone_number, booking_id)
values
(1, 'Kartik Kumar', 'kartik@email.com', '9876543210', 1),
(2, 'Priya Raghavan', 'priya@email.com', '8765432109', 2),
(3, 'Rajesh Sundaram', 'rajesh@email.com', '7654321098', 3),
(4, 'Ananya Balaji', 'ananya@email.com', '6543210987', 4),
(5, 'Vishal Mohan', 'vishal@email.com', '5432109876', 5),
(6, 'Nithya Venkat', 'nithya@email.com', '4321098765', 6),
(7, 'Arjun Kumar', 'arjun@email.com', '3210987654', 7),
(8, 'Divya Rajan', 'divya@email.com', '2109876543', 8),
(9, 'Suresh Ramalingam', 'suresh@gmail.com', '1098765432', 9),
(10, 'Shreya Anand', 'shreya@email.com', '9987654321', 10);

select *from Customer;
```

Results | Messages

	customer_id	customer_name	email	phone_number	booking_id
1	1	Kartik Kumar	kartik@email.com	9876543210	1
2	2	Priya Raghavan	priya@email.com	8765432109	2
3	3	Rajesh Sundaram	rajesh@email.com	7654321098	3
4	4	Ananya Balaji	ananya@email.com	6543210987	4
5	5	Vishal Mohan	vishal@email.com	5432109876	5
6	6	Nithya Venkat	nithya@email.com	4321098765	6
7	7	Arjun Kumar	arjun@email.com	3210987654	7
8	8	Divya Rajan	divya@email.com	2109876543	8
9	9	Suresh Ramalingam	suresh@gmail.com	1098765432	9
10	10	Shreya Anand	shreya@email.com	9987654321	10

Query executed successfully. | DESKTOP-MKS6P3G (16.0 RTM)

```sql
insert into Booking(booking_id, customer_id, event_id, num_tickets, total_cost, booking_date)
values
(1, 1, 1, 2, 30.00, '2024-09-01'),
(2, 2, 2, 4, 100.00, '2024-10-02'),
(3, 3, 3, 1, 20.00, '2024-11-01'),
(4, 4, 4, 3, 50.00, '2024-11-10'),
(5, 5, 5, 5, 50.00, '2024-10-05'),
(6, 6, 6, 2, 24.00, '2024-09-01'),
(7, 7, 7, 4, 88.00, '2024-10-11'),
(8, 8, 8, 6, 180.00, '2024-12-06'),
(9, 9, 9, 1, 8.00, '2024-11-15'),
(10, 10, 10, 3, 45.00, '2024-11-13');
```

Results | Messages

	booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
1	1	1	1	2	30.00	2024-09-01
2	2	2	2	4	100.00	2024-10-02
3	3	3	3	1	20.00	2024-11-01
4	4	4	4	3	50.00	2024-11-10
5	5	5	5	5	50.00	2024-10-05
6	6	6	6	2	24.00	2024-09-01
7	7	7	7	4	88.00	2024-10-11
8	8	8	8	6	180.00	2024-12-06
9	9	9	9	1	8.00	2024-11-15
10	10	10	10	3	45.00	2024-11-13

Query executed successfully.

2. Write a SQL query to list all Events.

```
select event_id, event_name from Event;
```

.00 %

Results | Messages

	event_id	event_name
1	1	Jaipur Film Festival
2	2	Jaipur Cricket Match
3	3	Udaipur Music Concert
4	4	Trishul Dance Show
5	5	High Act
6	6	Vellore Movie Night
7	7	Cultural Festival
8	8	Erode Football Championship
9	9	Rajmandir Art Exhibition
10	10	JC Comedy Show

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

```
--3
select event_name, event_type, available_seats AS available_tickets from Event;
```

%

Results | Messages

	event_name	event_type	available_tickets
	Jaipur Film Festival	Movie	450
	Jaipur Cricket Match	Sports	800
	Udaipur Music Concert	Concert	700
	Trishul Dance Show	Concert	550
	High Act	Drama	1000
	Vellore Movie Night	Movie	280
	Cultural Festival	Concert	650
	Erode Football Championship	Sports	1300
	Rajmandir Art Exhibition	Concert	380
)	JC Comedy Show	Concert	230
	World Cup Auction	Sports	1450

Query executed successfully.

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

```sql
--4
select event_id, event_name from Event where event_name like '%cup%';
```

100 %

Results | Messages

	event_id	event_name
1	11	World Cup Auction

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

```sql
--5
select event_id, event_name from Event where ticket_price between 1000 and 2500;
```

100 %

Results | Messages

	event_id	event_name
1	11	World Cup Auction
2	12	Thane Art Exhibition
3	13	Chennai Music Concert

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

```sql
--6
select event_id, event_name from Event where event_date between '2024-10-01' and '2024-11-15';
```

100 %

Results | Messages

	event_id	event_name
1	2	Jaipur Cricket Match
2	3	Udaipur Music Concert
3	5	High Act
4	7	Cultural Festival
5	11	World Cup Auction

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

```sql
--7
select event_name, available_seats As available_tickets from Event where event_type like '%concert%';
```

100 %

Results | Messages

	event_name	available_tickets
1	Udaipur Music Concert	700
2	Trishul Dance Show	550
3	Cultural Festival	650
4	Rajmandir Art Exhibition	380
5	JC Comedy Show	230

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

```
--8
select customer_id, customer_name from Customer order by customer_id Offset 5 rows fetch first 5 rows only;
```

100 %

Results | Messages

	customer_id	customer_name
1	6	Nithya Venkat
2	7	Arjun Kumar
3	8	Divya Rajan
4	9	Suresh Ramalingam
5	10	Shreya Anand

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

```
--9
select *from Booking where num_tickets>4;
```

100 %

Results | Messages

	booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
1	5	5	5	5	50.00	2024-10-05
2	8	8	8	6	180.00	2024-12-06

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

```
--10
select *from Customer where phone_number like '%000';
```

100 %

Results | Messages

	customer_id	customer_name	email	phone_number	booking_id
1	12	Lokesh Kumar	lokesh@email.com	9876543000	12

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

```
--11
select event_id, event_name, total_seats from Event where total_seats>15000 order by total_seats;
```

100 %

Results | Messages

	event_id	event_name	total_seats
1	13	Chennai Music Concert	16000
2	12	Thane Art Exhibition	18500

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

```
--12
select event_name from Event where event_name not like 'x%' and
event_name not like 'y%' and event_name not like 'z%';
```

100 %

Results | Messages

	event_name
1	Jaipur Film Festival
2	Jaipur Cricket Match
3	Udaipur Music Concert
4	Trishul Dance Show
5	High Act
6	Vellore Movie Night
7	Cultural Festival
8	Erode Football Championship
9	Rajmandir Art Exhibition
10	JC Comedy Show
11	World Cup Auction
12	Thane Art Exhibition
13	Chennai Music Concert

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

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

```
--1
select event_name, AVG(ticket_price) AS Average_Ticket_Price from Event group by event_name;
```

100 %

Results | Messages

	event_name	Average_Ticket_Price
1	Chennai Music Concert	1500.000000
2	Cultural Festival	22.000000
3	Erode Football Championship	30.000000
4	High Act	10.000000
5	Jaipur Cricket Match	25.000000
6	Jaipur Film Festival	15.000000
7	JC Comedy Show	15.000000
8	Rajmandir Art Exhibition	8.000000
9	Thane Art Exhibition	1500.000000
10	Trishul Dance Show	18.000000
11	Udaipur Music Concert	20.000000
12	Vellore Movie Night	12.000000
13	World Cup Auction	1500.000000

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

```
--2
select SUM(total_cost) AS Total_Revenue_Generated from Booking;
```

100 %

Results | Messages

	Total_Revenue_Generated
1	595.00

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

```
--3
select top 1 e.event_id, e.event_name, SuM(b.num_tickets) AS Total_Ticket_Sales
from Event e
JOIN Booking b ON e.event_id = b.event_id
Group By e.event_id, e.event_name
Order By Total_Ticket_Sales DESC;
```

100 %

Results | Messages

	event_id	event_name	Total_Ticket_Sales
1	8	Erode Football Championship	6

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

```
--4
select e.event_name, SuM(b.num_tickets) AS No_Tickets_Sold
from Event e
JOIN Booking b ON e.event_id = b.event_id
Group By e.event_name;
```

100 %

Results | Messages

	event_name	No_Tickets_Sold
1	Cultural Festival	4
2	Erode Football Championship	6
3	High Act	5
4	Jaipur Cricket Match	4
5	Jaipur Film Festival	2
6	JC Comedy Show	3
7	Rajmandir Art Exhibition	1
8	Trishul Dance Show	3
9	Udaipur Music Concert	1
10	Vellore Movie Night	2
11	World Cup Auction	1

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

```sql
--5
SELECT e.event_id, e.event_name FROM Event e
LEFT JOIN Booking b ON e.event_id = b.event_id
WHERE b.num_tickets IS NULL;
```

100 %

Results | Messages

	event_id	event_name
1	12	Thane Art Exhibition
2	13	Chennai Music Concert

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

```sql
--6
select top 1 c.customer_name, SuM(b.num_tickets) AS Tickets_Booked
from Customer c
JOIN Booking b ON c.customer_id = b.customer_id
Group By c.customer_name
Order By Tickets_Booked DESC;
```

100 %

Results | Messages

	customer_name	Tickets_Booked
1	Divya Rajan	6

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

```sql
--7
select MONTH(booking_date) AS booked_month, SUM(num_tickets) AS total_tickets
From Booking b
Join Event e ON e.event_id = b.event_id
Group by booking_date;
```

91 %

Results | Messages

	booked_month	total_tickets
1	9	4
2	9	1
3	10	4
4	10	5
5	10	4
6	11	1
7	11	3
8	11	3
9	11	1
10	12	6

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

```sql
--8
select e.event_id, e.event_name, AVG(e.ticket_price) As Avg_Ticket_price, v.venue_name
from Event e
JOIN venu v ON v.venue_id=e.venue_id
Group By e.event_id, e.event_name, v.venue_name;
```

100 %

Results | Messages

	event_id	event_name	Avg_Ticket_price	venue_name
1	1	Jaipur Film Festival	15.000000	Jaipur Convention Center
2	2	Jaipur Cricket Match	25.000000	Mansarovar Sports Arena
3	3	Udaipur Music Concert	20.000000	Udaipur Music Hall
4	4	Trishul Dance Show	18.000000	Trishul Event Plaza
5	5	High Act	10.000000	Rajmandir Hall
6	6	Vellore Movie Night	12.000000	Vellore Entertainment Complex
7	7	Cultural Festival	22.000000	Birla Auditorium
8	8	Erode Football Championship	30.000000	Erode Cultural Center
9	9	Rajmandir Art Exhibition	8.000000	RajMandir Convention Hall
10	10	JC Comedy Show	15.000000	JC Event Centre
11	11	World Cup Auction	1500.000000	Birla Auditorium
12	12	Thane Art Exhibition	1500.000000	Birla Auditorium
13	13	Chennai Music Concert	1500.000000	JC Event Centre

Query executed successfully. DESKTOP-MKS6P3G (16.0 RT

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

```sql
--9
SELECT CAST(e.event_type AS NVARCHAR(MAX)) AS event_type, SUM(b.num_tickets)
AS total_tickets_sold FROM Event e
JOIN Booking b ON e.event_id = b.event_id
GROUP BY CAST(e.event_type AS NVARCHAR(MAX));
```

100 %

Results | Messages

	event_type	total_tickets_sold
1	Concert	12
2	Drama	5
3	Movie	4
4	Sports	11

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 Event e
JOIN Booking b ON e.event_id = b.event_id
GROUP BY YEAR(e.event_date)
ORDER BY event_year;
```

100 %

Results | Messages

	event_year	total_revenue
1	2024	2095.00

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

```sql
--11
select c.customer_id, c.customer_name, COUNT(b.event_id)
AS Events_Booked From Customer c
Join Booking b ON c.customer_id=b.customer_id
Group by c.customer_id, c.customer_name
HAVING COUNT(b.event_id) > 1;
```

100 %

Results | Messages

	customer_id	customer_name	Events_Booked
1	1	Kartik Kumar	2

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

```sql
--12
select c.customer_name, SUM(total_cost) AS Total_Revenue
From Customer c
Join Booking b On b.customer_id=c.customer_id
Group By c.customer_name;
```

100 %

Results | Messages

	customer_name	Total_Revenue
1	Ananya Balaji	50.00
2	Arjun Kumar	88.00
3	Divya Rajan	180.00
4	Kartik Kumar	1530.00
5	Nithya Venkat	24.00
6	Priya Raghavan	100.00
7	Rajesh Sundaram	20.00
8	Shreya Anand	45.00
9	Suresh Ramalingam	8.00
10	Vishal Mohan	50.00

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

```sql
--13
select v.venue_name, CAST(e.event_type AS NVARCHAR(MAX))
As event_type, AVG(e.ticket_price) AS Average_ticket_price
From event e Join Venu v ON e.venue_id=v.venue_id
Group By v.venue_name, CAST(e.event_type AS NVARCHAR(MAX));
```

100 %

Results | Messages

	venue_name	event_type	Average_ticket_price
1	Birla Auditorium	Concert	22.000000
2	JC Event Centre	Concert	15.000000
3	RajMandir Convention Hall	Concert	8.000000
4	Trishul Event Plaza	Concert	18.000000
5	Udaipur Music Hall	Concert	20.000000
6	Rajmandir Hall	Drama	10.000000
7	Jaipur Convention Center	Movie	15.000000
8	Vellore Entertainment Complex	Movie	12.000000
9	Birla Auditorium	Sports	1500.000000
10	Erode Cultural Center	Sports	30.000000
11	JC Event Centre	Sports	1500.000000
12	Mansarovar Sports Arena	Sports	25.000000

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

```sql
--14
SELECT c.customer_name, SUM(b.num_tickets) AS Ticket_Purchased
FROM Booking b JOIN Customer c ON b.customer_id = c.customer_id
WHERE b.booking_date BETWEEN DATEADD(DAY, -30, GETDATE()) AND GETDATE()
GROUP BY c.customer_name;
```

100 %

Results | Messages

	customer_name	Ticket_Purchased
1	Kartik Kumar	3
2	Nithya Venkat	2

Tasks 4: Subquery and its types

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

```
--1
select v.venue_id, v.venue_name,
(Select AVG(ticket_price) From  event e where e.venue_id=v.venue_id)
As average_ticket_price from Venu v;
```

100 %

Results | Messages

	venue_id	venue_name	average_ticket_price
1	1	Jaipur Convention Center	15.000000
2	2	Mansarovar Sports Arena	25.000000
3	3	Udaipur Music Hall	20.000000
4	4	Trishul Event Plaza	18.000000
5	5	Rajmandir Hall	10.000000
6	6	Vellore Entertainment Complex	12.000000
7	7	Birla Auditorium	1007.333333
8	8	Erode Cultural Center	30.000000
9	9	RajMandir Convention Hall	8.000000
10	10	JC Event Centre	757.500000

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

```
--2
SELECT e.event_id, e.event_name FROM Event e
WHERE (SELECT SUM(b.num_tickets) FROM Booking b
WHERE b.event_id = e.event_id) > (0.5 * e.total_seats);
```

100 %

Results | Messages

	venue_id	venue_name	average_ticket_price
1	1	Jaipur Convention Center	15.000000
2	2	Mansarovar Sports Arena	25.000000
3	3	Udaipur Music Hall	20.000000
4	4	Trishul Event Plaza	18.000000
5	5	Rajmandir Hall	10.000000
6	6	Vellore Entertainment Complex	12.000000
7	7	Birla Auditorium	1007.333333
8	8	Erode Cultural Center	30.000000
9	9	RajMandir Convention Hall	8.000000
10	10	JC Event Centre	757.500000

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

```
--3
select e.event_name, e.event_id,(select sum(num_tickets)
From Booking b where b.event_id=e.event_id)
As Sold_tickets From Event e;
```

100 %

Results | Messages

	event_name	event_id	Sold_tickets
1	Jaipur Film Festival	1	2
2	Jaipur Cricket Match	2	4
3	Udaipur Music Concert	3	1
4	Trishul Dance Show	4	3
5	High Act	5	5
6	Vellore Movie Night	6	2
7	Cultural Festival	7	4
8	Erode Football Championship	8	6
9	Rajmandir Art Exhibition	9	1
10	JC Comedy Show	10	3
11	World Cup Auction	11	1
12	Thane Art Exhibition	12	NULL
13	Chennai Music Concert	13	NULL

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

```
--4
Select c.customer_id, c.customer_name From Customer c
where not exists(select *from Booking b where c.customer_id=b.customer_id);
```

100 %

Results | Messages

	customer_id	customer_name
1	12	Lokesh Kumar

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

```
--5
select e.event_name from event e where e.event_id NOT IN
(select distinct b.event_id From Booking b where e.event_id=b.event_id);
```

100 %

Results | Messages

	event_name
1	Thane Art Exhibition
2	Chennai Music Concert

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

```
--6
select CAST(e.event_type AS NVARCHAR(MAX)) As event_type,
SUM(b.num_tickets) AS Total_tickets_sold from
(select event_id, SUM(num_tickets) AS num_tickets from Booking
group by event_id) b JOIN Event e ON e.event_id=b.event_id
Group By CAST(e.event_type AS NVARCHAR(MAX));
```

100 %

Results | Messages

	event_type	Total_tickets_sold
1	Concert	12
2	Drama	5
3	Movie	4
4	Sports	11

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

```
--7
SELECT e.event_id, e.event_name, e.ticket_price
FROM Event e
WHERE e.ticket_price > (SELECT AVG(ticket_price) FROM Event);
```

100 %

Results | Messages

	event_id	event_name	ticket_price
1	11	World Cup Auction	1500.00
2	12	Thane Art Exhibition	1500.00
3	13	Chennai Music Concert	1500.00

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

```sql
--8
select c.customer_id, c.customer_name, (Select SUM(b.total_cost) AS TotalCost
From Booking b where b.customer_id = c.customer_id) As Total_Revenue
From Customer c;
```

100 %

Results | Messages

	customer_id	customer_name	Total_Revenue
1	1	Kartik Kumar	1530.00
2	2	Priya Raghavan	100.00
3	3	Rajesh Sundaram	20.00
4	4	Ananya Balaji	50.00
5	5	Vishal Mohan	50.00
6	6	Nithya Venkat	24.00
7	7	Arjun Kumar	88.00
8	8	Divya Rajan	180.00
9	9	Suresh Ramalingam	8.00
10	10	Shreya Anand	45.00
11	12	Lokesh Kumar	NULL

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

```sql
--9
select c.customer_id, c.customer_name From Customer c where c.customer_id
IN (Select b.customer_id from Booking b JOIN Event e ON
b.event_id=e.event_id where e.venue_id=4);
```

100 %

Results | Messages

	customer_id	customer_name
1	4	Ananya Balaji

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

```sql
--10
select event_id, SUM(num_tickets) AS Total_Tickets_Sold
From Booking group by event_id;
```

100 %

Results | Messages

	event_id	Total_Tickets_Sold
1	1	2
2	2	4
3	3	1
4	4	3
5	5	5
6	6	2
7	7	4
8	8	6
9	9	1
10	10	3
11	11	1

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

```sql
--11
select c.customer_id, c.customer_name, FORMAT(b.booking_date, 'MM')
AS Booking_Month From Customer c JOIN Booking b ON
c.customer_id=b.customer_id ORDER BY c.customer_id, FORMAT(b.booking_date, 'MM');
```

100 %

Results | Messages

	customer_id	customer_name	Booking_Month
1	1	Kartik Kumar	09
2	1	Kartik Kumar	09
3	2	Priya Raghavan	10
4	3	Rajesh Sundaram	11
5	4	Ananya Balaji	11
6	5	Vishal Mohan	10
7	6	Nithya Venkat	09
8	7	Arjun Kumar	10
9	8	Divya Rajan	12
10	9	Suresh Ramalingam	11
11	10	Shreya Anand	11

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

```
--12
select v.venue_name, (Select AVG(e.ticket_price) From Event e
Where e.venue_id=v.venue_id) As Average_Ticket_Price From Venu v;
```

100 %

Results | Messages

	venue_name	Average_Ticket_Price
1	Jaipur Convention Center	15.000000
2	Mansarovar Sports Arena	25.000000
3	Udaipur Music Hall	20.000000
4	Trishul Event Plaza	18.000000
5	Rajmandir Hall	10.000000
6	Vellore Entertainment Complex	12.000000
7	Birla Auditorium	1007.333333
8	Erode Cultural Center	30.000000
9	RajMandir Convention Hall	8.000000
10	JC Event Centre	757.500000