ASSIGNMENT-1 TICKET BOOKING SYSTEM

```
use ticket_booking;
show tables;
describe booking;
describe venue;
describe event;
describe customer;
insert into venue(venue_name,address) values
('mumbai', 'marol andheri(w)'),
('chennai', 'IT Park'),
('pondicherry ', 'state beach');
select * from venue;
insert into customer(customer_name,email,phone_number)
values
('harry potter','harry@gmail.com','45454545'),
('ronald weasley','ron@gmail.com','45454545'),
('hermione granger','her@gmail.com','45454545'),
('draco malfoy','drac@gmail.com','45454545'),
('ginni weasley','ginni@gmail.com','45454545');
select * from customer;
-- ALTER TABLE event AUTO_INCREMENT = 4;

insert into
event(event_name,event_date,event_time,total_seats,available_seats,ticket_price,event_type,venue_id
)
values
('Late Ms. Lata Mangeshkar Musical', '2021-09-12','20:00',320,270,600,'concert',3),
('CSK vs RCB', '2024-04-11','19:30',23000,3,3600,'sports',2),
('CSK vs RR', '2024-04-19','19:30',23000,10,3400,'sports',2),
('MI vs KKR', '2024-05-01','15:30',28000,100,8000,'sports',1);
select * from event;
insert into booking values
```

```
(4,1,2,640,'2021-09-12'),
(4,4,3,960,'2021-09-12'),
(5,1,3,10800,'2024-04-11'),
(5,3,5,18000,'2024-04-10'),
(6,5,10,34000,'2024-04-15'),
(7,2,4,32000,'2024-05-01');
-- Tasks 2: Select, Where, Between, AND, LIKE:
SELECT * FROM event;
/*
4       Late Ms. Lata Mangeshkar Musical       2021-09-12      20:00:00        320     270     600
        concert  3
5       CSK vs RCB      2024-04-11      19:30:00        23000   3       3600    sports  2
6       CSK vs RR       2024-04-19      19:30:00        23000   10      3400    sports  2
7       MI vs KKR       2024-05-01      15:30:00        28000   100     8000    sports  1

*/
-- 3. Write a SQL query to select events with available tickets.
SELECT event_name
From event
Where available_seats > 0;
/*
Late Ms. Lata Mangeshkar Musical
CSK vs RCB
CSK vs RR
MI vs KKR
*/
-- 4. Write a SQL query to select events name partial match with 'cup'.
Select event_name
from event
Where event_name LIKE '%vs%';
/*
CSK vs RCB
CSK vs RR
```

```
MI vs KKR
*/
-- 5. Write a SQL query to select events with ticket price range is between 1000 to 2500.
Select event_name
From event
Where ticket_price BETWEEN 1000 AND 3500;
/*
CSK vs RR
*/
-- 6. Write a SQL query to retrieve events with dates falling within a specific range.
SELECT *
FROM event
Where event_date BETWEEN '2024-09-01' AND '2024-12-31';
-- 7. Write a SQL query to retrieve events with available tickets that also have "Concert" in their
name.
SELECT *
From event
Where available_seats > 0 AND event_name LIKE '%vs%';
/*
5       CSK vs RCB     2024-04-11     19:30:00       23000   3      3600    sports   2
6       CSK vs RR      2024-04-19     19:30:00       23000   10     3400    sports   2
7       MI vs KKR      2024-05-01     15:30:00       28000   100    8000    sports   1
*/
-- 8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.
-- 9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.
SELECT *
FROM booking
WHERE num_tickets > 4;
/*
5       3       5       18000   2024-04-10
6       5       10      34000   2024-04-15
*/
-- 10. Write a SQL query to retrieve customer information whose phone number end with '000'
```

```
SELECT *
FROM customer
WHERE phone_number LIKE '%45';
/*
2	ronald weasley	ron@gmail.com45454545
3	hermione granger	her@gmail.com45454545
4	draco malfoy	drac@gmail.com	45454545
5	ginni weasley	ginni@gmail.com	45454545
*/
-- 11. Write a SQL query to retrieve the events in order whose seat capacity more than 15000.
SELECT *
FROM event
WHERE total_seats > 15000 ORDER BY total_seats DESC;
/*
7	MI vs KKR	2024-05-01	15:30:00	28000	100	8000	sports	1
5	CSK vs RCB	2024-04-11	19:30:00	23000	3	3600	sports	2
*/
-- 12. Write a SQL query to select events name not start with x', y. 't
SELECT *
FROM event
WHERE event_name NOT LIKE 'x%'
AND event_name NOT LIKE 'y%'
AND event_name NOT LIKE 't%';
/*
4	Late Ms. Lata Mangeshkar Musical	2021-09-12	20:00:00	320	270	600	concert	3
5	CSK vs RCB	2024-04-11	19:30:00	23000	3	3600	sports	2
6	CSK vs RR	2024-04-19	19:30:00	23000	10	3400	sports	2
7	MI vs KKR	2024-05-01	15:30:00	28000	100	8000	sports	1
*/
-- 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 avg_ticket_price
```

```
FROM event
GROUP BY event_name;
/*
CSK vs RCB	3600
CSK vs RR	3400
Late Ms. Lata Mangeshkar Musical	600
MI vs KKR	8000
*/
-- 2. Write a SQL query to Calculate the Total Revenue Generated by Events.
SELECT SUM(total_cost) AS total_revenue
FROM booking;
/*
96400
*/
-- 3. Write a SQL query to find the event with the highest ticket sales.
SELECT event_name, SUM(num_tickets) AS total_tickets_sold
FROM booking
JOIN event ON booking.event_id = event.event_id
GROUP BY event_name
ORDER BY total_tickets_sold DESC
LIMIT 1;
-- 4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event.
SELECT event_name, SUM(num_tickets) AS total_tickets_sold
FROM booking
JOIN event ON booking.event_id = event_id
GROUP BY event_name;
/*
CSK vs RCB	27
CSK vs RR	27
Late Ms. Lata Mangeshkar Musical	27
MI vs KKR	27
*/
```

```
-- 5. Write a SQL query to Find Events with No Ticket Sales.
SELECT event_name
FROM event
LEFT JOIN booking ON event.id = booking.event_id
WHERE booking.id IS NULL;
-- 6. Write a SQL query to Find the User Who Has Booked the Most Tickets.
SELECT customer_name, SUM(num_tickets) AS total_tickets_booked
FROM booking
JOIN customer ON booking.customer_id = customer.id
GROUP BY customer_name
ORDER BY total_tickets_booked DESC
LIMIT 1;
/*
ginni weasley    10
*/
-- 7. Write a SQL query to List Events and the total number of tickets sold for each month.
SELECT MONTH(booking_date) AS month,sum(num_tickets)
FROM booking
JOIN event ON booking.event_id = event.id
GROUP BY month;
/*
4        18
5        4
9        5
*/
-- 8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue.
SELECT venue_name, AVG(ticket_price) AS avg_ticket_price
FROM event
JOIN venue ON event.venue_id = venue.id
GROUP BY venue_name;
/*
chennai 3500
```

```
mumbai          8000
pondicherry     600*/
-- 9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.
SELECT event_type, SUM(num_tickets) AS total_tickets_sold
FROM booking
JOIN event ON booking.event_id = event.id
GROUP BY event_type;
/*
concert 5
sports  22
*/
-- 10. Write a SQL query to calculate the total Revenue Generated by Events in Each Year.
SELECT YEAR(booking_date) AS year, SUM(total_cost) AS total_revenue
FROM booking
GROUP BY year
ORDER BY year;
/*
2021    1600
2024    94800
*/
-- 11. Write a SQL query to list users who have booked tickets for multiple events.
SELECT customer_name, COUNT(DISTINCT event_id) AS num_events_booked
FROM booking
JOIN customer ON booking.customer_id = customer.id
GROUP BY customer_name
HAVING num_events_booked > 1;
/*
harry potter    2
*/
-- 12. Write a SQL query to calculate the Total Revenue Generated by Events for Each User.
SELECT customer_name, SUM(total_cost) AS total_revenue
FROM booking
```

```
JOIN customer ON booking.customer_id = customer.id
GROUP BY customer_name;
/*
draco malfoy	960
ginni weasley	34000
harry potter	11440
hermione granger		18000
ronald weasley	32000
*/
-- 13. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.
SELECT venue_name, event_type, AVG(ticket_price) AS avg_ticket_price
FROM event
JOIN venue ON event.venue_id = venue.id
GROUP BY venue_name, event_type;
/*
chennai	sports	3500
mumbai		sports	8000
pondicherry	concert	600
*/
-- task 4
-- 1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
/*
projection: ticket price of event
criteria: venue
*/
select v.venue_name, AVG(e.ticket_price) as Average_Ticket_price
from venue v JOIN event e ON v.id=e.venue_id
group by v.venue_name;
/*
venue_name Average_Ticket_price
chennai 3500
mumbai 8000
```

```
pondicherry 600
*/
-- Find Events with More Than 50% of Tickets Sold using subquery.
/*
Analysis: If (total_seats-available seats) > (total_seats/2) -- this event shd be part of RS
(320-270) > (320/2) -- this will not be displayed
*/
select *
from event
where (total_seats-available_seats) > (total_seats/2);
-- 3. Calculate the Total Number of Tickets Sold for Each Event.
/*
Analysis: tickets_sold = (total_seats-available seats)
*/
select event_name, SUM(total_seats-available_seats) as Tickets_Sold
from event
group by event_name;
-- 4. Find Customer Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery
/* Project : customer
condition: booking table */
select *
from customer
where id NOT IN (select distinct c.id
from customer c JOIN booking b ON c.id = b.customer_id);
/*
7 frodo baggins frodo@lotr.com 35454
*/
-- EXISTS and NOT-EXISTS
select *
from venue;
-- we want the above query to display results if and only if the below query returns atleast 1 record
select *
```

```sql
from event
where total_seats>27000; -- 1 row
select *
from venue
where EXISTS (select *
from event
where total_seats>29000);
-- EXISTS: for the outer query to run and show result, the inner query must return atleast 1 record.
-- 6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause
select event_name, SUM(total_seats-available_seats) as Total_tickets_sold
from event
group by event_name;
select dt.event_name, SUM(dt.total_seats-dt.available_seats) as Total_tickets_sold
from (select * from event) as dt
group by event_name;
-- Display events with number of tickets_sold. consider those events where venue is in given list ['mumbai','chennai']
select event_name, SUM(total_seats-available_seats) as Total_tickets_sold
from ( select event_name,total_seats,available_seats
from event e JOIN venue v ON e.venue_id=v.id
where venue_name IN ('mumbai','chennai')) as dt
group by event_name;
select event_name, SUM(total_seats-available_seats) as Total_tickets_sold
from event e JOIN venue v ON e.venue_id=v.id
where venue_name IN ('mumbai','chennai')
group by event_name;
-- NOT IN , EXISTS-NOT EXISTS , Query in From statement - drived table/virtual
-- Calculate the Total Revenue Generated by Events for Each Customer Using a Correlated Subquery
select c.customer_name,SUM(total_cost)
from booking b JOIN customer c ON b.customer_id = c.id
group by c.customer_name;
```