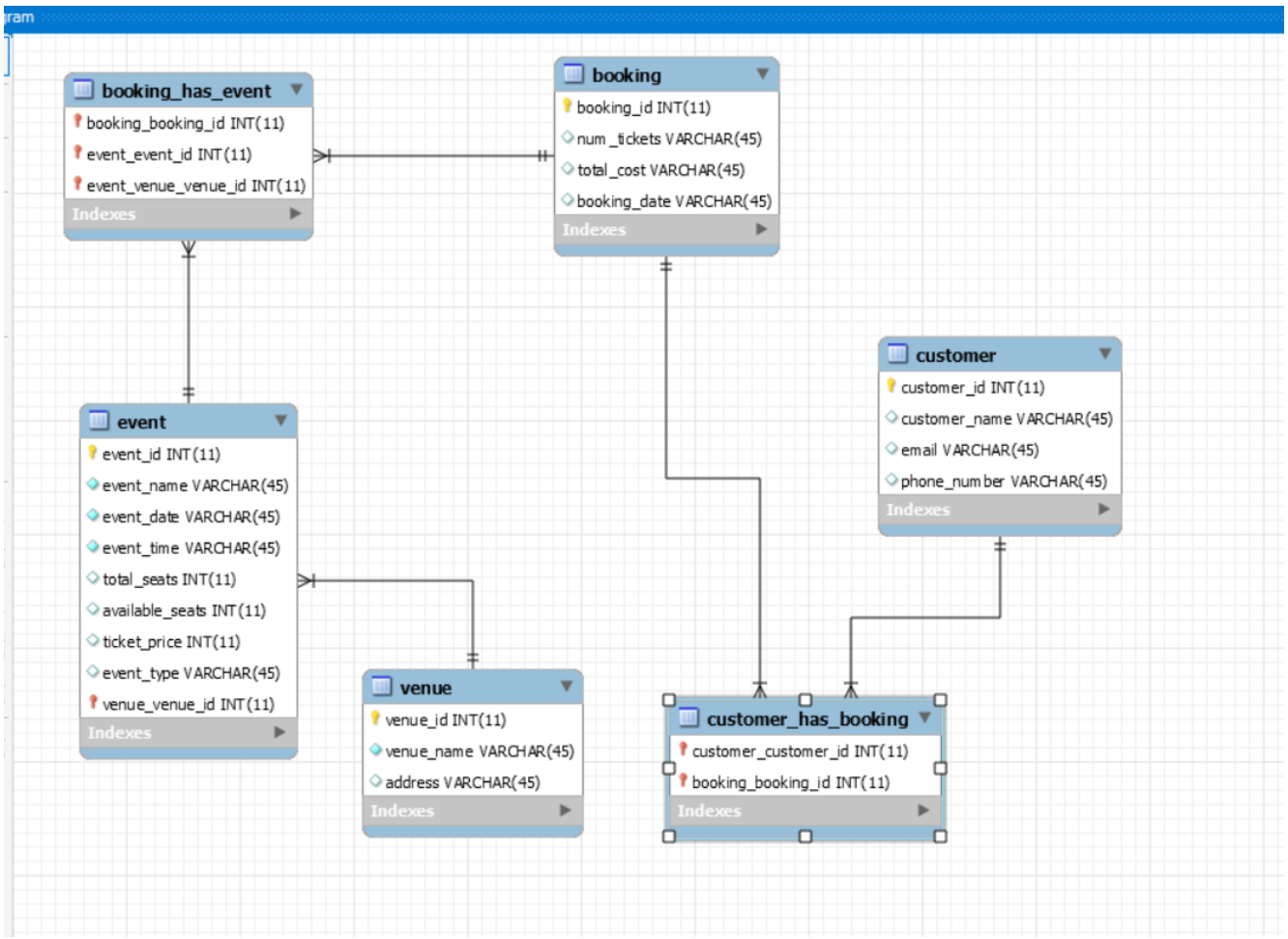
TICKET_BOOKING_DB



use ticket_db;

select * from venue;

select * from customer;

select* from booking;

select * from event;

/*task 2*/

/*Write a SQL query to list all Events.*/

```
select * from event;
```

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

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

/*Write a SQL query to retrieve events with dates falling within a specific range */

```
select event_name from event where event_date between '2021-09-12' and '2024-04-19';
```

/*Write a SQL query to retrieve customers in batches of 5, starting from the 6th user. */

```
select * from customer limit 1,3;
```

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

```
select * from customer where phone_number = '%000';
```

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

```
select total_seats from event where total_seats>1000 order by total_seats;
```

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

```
select event_name from event where event_name LIKE '(c,l)%';



/* Multi Table Queries using Manual Mapping Technique

practice

-- display list of events hosted by venue 'chennai'.*/

select e.event_name,v.venue_name from

event e join venue v On v.venue_id = e.venue_venue_id

having venue_name like 'chennai';

/*select customers that have booked tickes for event 'csk v rcb'  game with id=5; */

select c.customer_name from customer c join booking b ON c.customer_id=b.customer_customer_id where booking_id='7';

/*  display event details that have booking num_tickets > 1000*/

select e.event_name from event e,booking b where num_tickets>2 and e.event_id=b.event_id;

/* Display the names of venues visited by customer with email 'harry@gmail.com'*/
```

```
select v.venue_name,v.address,c.customer_name
from venue v,booking b,event e,customer c
where v.id=e.venue_id AND
e.id = b.event_id AND
b.customer_id = c.id AND
c.email='harry@gmail.com';



/*task 3*/


/*
. Write a SQL query to List Venues and Their Average Ticket Prices.*/


select avg(e.ticket_price),e.event_name from event e,venue v where
e.venue_venue_id=v.venue_id;


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


select avg(e.ticket_price),e.event_name from event e,venue v where
e.venue_venue_id=v.venue_id group by venue_name;


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


select SUM((total_seats - available_seats) *  ticket_price) #We can perform
arithmetic ops in select statement
from event;
```

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

```
select event_name,total_seats-available_seats as ticket_sold from event group by event_name;
```

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

```
select event_name from event where total_seats = available_seats;
```

/* Write a SQL query to list customer who have booked tickets for multiple events.*/

```
select c.customer_name , count(c.id) as events_booked
from event e,customer c, booking b
where e.id = b.event_id AND
b.customer_id = c.id
group by c.customer_name
having events_booked>1;
```

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

```
 select c.customer_name, SUM(b.num_tickets) as Number_Of_tickets
from  event e  JOIN booking b  ON e.id = b.event_id JOIN  customer c ON c.id = b.customer_id where b.booking_date between DATE_SUB('2024-04-30',INTERVAL 30 DAY) and  '2024-04-30'
group by c.customer_name;
```

```
/* display list of events hosted by venue 'chennai'.*/

select e.event_id,e.event_name,e.event_date,e.event_time,e.total_seats
from event e,venue v
where v.venue_id = e.venue_venue_id AND v.venue_name='chennai';

 /*select customers that have booked tickes for event 'csk v rcb'  game with id=5; */

select c.customer_name,email,phone_number
from customer c, booking b
where c.customer_id = b.customer_customer_id AND b.event_id=1;



 /* NESTED QUERY */

 /* practice*/

 select event_id,event_name from event where
 venue_venue_id IN (select venue_id from venue where venue_name ='chennai');

 select customer_id,customer_name from customer where
```

```
 customer_id IN (select customer_customer_id from booking where event_id
IN (select event_id from event where venue_venue_id IN (select venue_id from
venue where venue_name='chennai')));



 select event_id,event_name from event where event_type = 'sports'

 and event_id IN (select event_id from booking where num_tickets > 1);


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


select * from event;

select event_id,avg(ticket_price) from event where venue_venue_id IN (select
venue_id from venue) group by venue_venue_id;



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



select event_name from event where event_id IN

(select event_id from event where (total_seats-
available_seats)>(total_seats/2));


 /* 3. Find Events having ticket price more than average ticket price of all events

avg price-----> sub query


select event_name from event where ticket_price >

 (select avg(ticket_price) from event);
```

```
 /* 4. Find Customers Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.*/

 select customer_name
 from customer
 where NOT EXISTS (select distinct c.customer_name
from customer c join booking b ON b.customer_customer_id = c.customer_id);

/*Q. Names of Customers who have visited venue 'chennai' using all three techniques(Nested Query).*/

select id,customer_name
from customer
where id IN (select customer_id
 from booking where event_id IN (select id
from event  where venue_id IN (select id

from venue where venue_name='chennai')));

-- Task 4: Subquery and its types

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

select venue_id,AVG(ticket_price) as Avg_Price
from event
```

```
where venue_id IN (select id from venue)
group by venue_id;

/* 2. Find Events with More Than 50% of Tickets Sold using subquery.*/
select event_name
from event
where id IN ( select id
                    from event
        where (total_seats - available_seats) > (total_seats/2));

/* 3. Find Events having ticket price more than average ticket price of all events
*/

select event_name
from event
where ticket_price >  (select avg(ticket_price) from event);

 /* 4. Find Customers Who Have Not Booked Any Tickets Using a NOT EXISTS
Subquery */

 select customer_name
 from customer
 where NOT EXISTS (select distinct c.customer_name
                          from customer c join booking b ON b.customer_id =
c.id);
```

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

```
select event_name from event where total_seats = available_seats;
```

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

```
select count(total_seats),event_id from event group by event_id;
select*from event;
```

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

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

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

```
select (total_seats-available_seats)*ticket_price from event group by event_id;
```

/*9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHEREClause.*/

```
select event_id,customer_customer_id from booking where event_id in
```

```
 (select event_id from event where venue_venue_id=2);


/*10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery withGROUP BY .*/


 select sum(total_seats-available_seats),event_id from event group by event_id;


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

DATE_FORMAT.*/

 select customer_customer_id,DATE_FORMAT(booking_date, '%Y-%m') from booking;


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


 SELECT venue_venue_id, AVG(ticket_price) AS avg_ticket_price

FROM event

GROUP BY venue_venue_id;
```