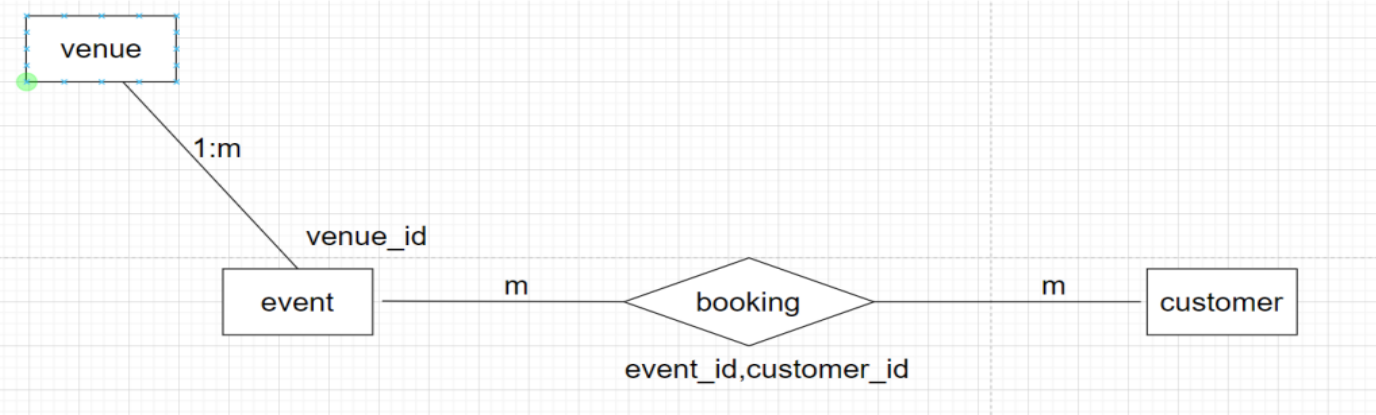
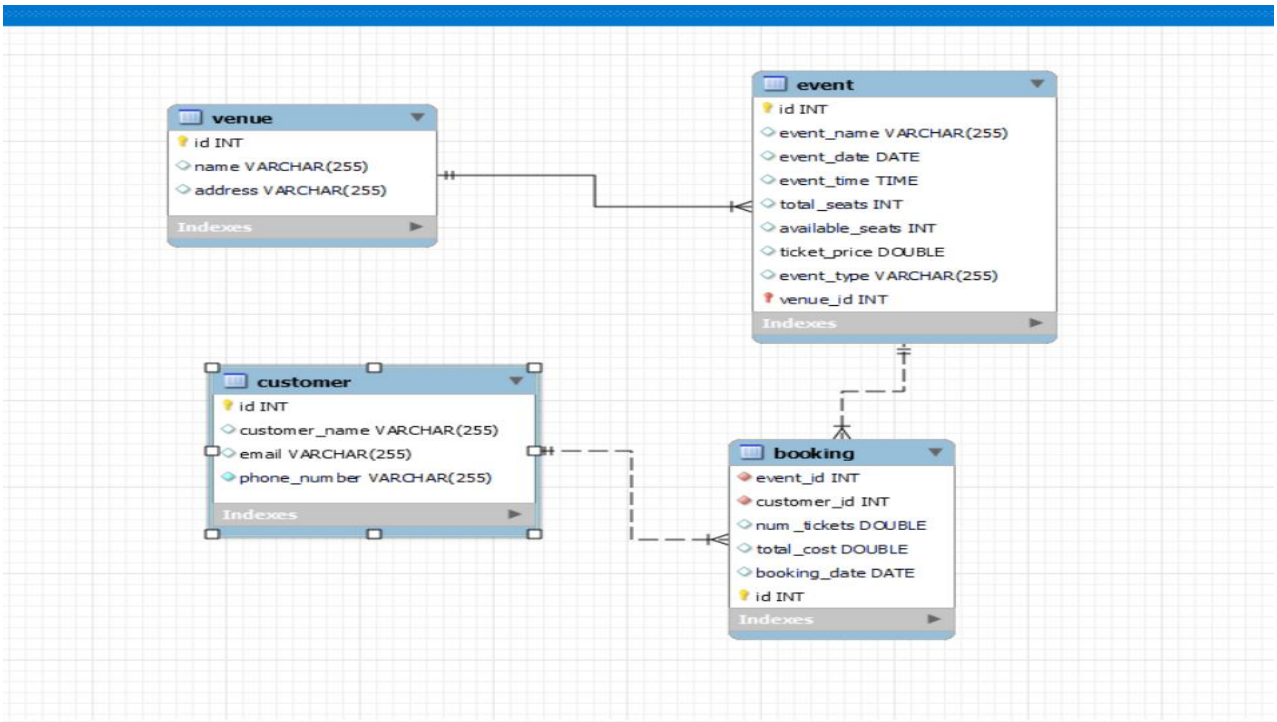
Ticket Booking System




```
CREATE DATABASE TicketBookingSystem;
USE TicketBookingSystem;

CREATE TABLE IF NOT EXISTS `TicketBookingSystem`.`venue` (
`id` INT NOT NULL AUTO_INCREMENT,
`name` VARCHAR(255) NULL,
`address` VARCHAR(255) NULL,
PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `TicketBookingSystem`.`event` (
`id` INT NOT NULL AUTO_INCREMENT,
`event_name` VARCHAR(255) NULL,
`event_date` DATE NULL,
`event_time` TIME NULL,
`total_seats` INT NULL,
`available_seats` INT NULL,
`ticket_price` DOUBLE NULL,
`event_type` VARCHAR(255) NULL,
`venue_id` INT NOT NULL,
PRIMARY KEY (`id`, `venue_id`),
INDEX `fk_event_venue1_idx` (`venue_id` ASC),
CONSTRAINT `fk_event_venue1`
FOREIGN KEY (`venue_id`)
REFERENCES `TicketBookingSystem`.`venue` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB

CREATE TABLE IF NOT EXISTS `TicketBookingSystem`.`customer` (
`id` INT NOT NULL AUTO_INCREMENT,
`customer_name` VARCHAR(255) NULL,
`email` VARCHAR(255) NULL,
```

```
`phone_number` VARCHAR(255) NOT NULL,
PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `TicketBookingSystem`.`booking` (
`event_id` INT NOT NULL,
`customer_id` INT NOT NULL,
`num_tickets` DOUBLE NULL,
`total_cost` DOUBLE NULL,
`booking_date` DATE NULL,
`id` INT NOT NULL AUTO_INCREMENT,
INDEX `fk_event_has_customer_customer1_idx` (`customer_id` ASC),
INDEX `fk_event_has_customer_event1_idx` (`event_id` ASC),
PRIMARY KEY (`id`),
CONSTRAINT `fk_event_has_customer_event1`
FOREIGN KEY (`event_id`)
REFERENCES `TicketBookingSystem`.`event` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_event_has_customer_customer1`
FOREIGN KEY (`customer_id`)
REFERENCES `TicketBookingSystem`.`customer` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

ER DIAGRAM



INSERTIONS...

```
insert into venue(name,address) values
```

('mumbai', 'marol andheri(w)'),
('chennai', 'IT Park'),
('pondicherry ', 'state beach');

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');

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);

insert into booking(event_id, customer_id, num_tickets, total_cost, booking_date) values
(1,1,2,640,'2021-09-12'),
(1,4,3,960,'2021-09-12'),
(2,1,3,10800,'2024-04-11'),
(2,3,5,18000,'2024-04-10'),
(3,5,10,34000,'2024-04-15'),
(4,2,4,32000,'2024-05-01');

```
mysql> select * from customer;
+----+--------------------+-------------------+--------------+
| id | customer_name      | email             | phone_number |
+----+--------------------+-------------------+--------------+
|  1 | harry potter       | harry@gmail.com   | 45454545     |
|  2 | ronald weasley     | ron@gmail.com     | 45454545     |
|  3 | hermione granger   | her@gmail.com     | 45454545     |
|  4 | draco malfoy       | drac@gmail.com    | 45454545     |
|  5 | ginni weasley      | ginni@gmail.com   | 45454545     |
|  6 | Severus Snape      | sev@gmail.com     | 45454846     |
|  7 | rubeus hagrid      | hagrid@gmail.com  | 45454525     |
|  8 | albus dumbuldore   | albus@gmail.com   | 45454515     |
|  9 | neville longbottom | neville@gmail.com | 45454565     |
| 10 | remus lupin        | remus@gmail.com   | 45454575     |
| 11 | sirius black       | sirius@gmail.com  | 45454555     |
+----+--------------------+-------------------+--------------+
11 rows in set (0.00 sec)
```

```
mysql> select * from venue;
+----+---------------+-------------------+
| id | name          | address           |
+----+---------------+-------------------+
|  1 | mumbai        | marol andheri(w)  |
|  2 | chennai       | IT Park           |
|  3 | pondicherry   | state beach       |
+----+---------------+-------------------+
3 rows in set (0.00 sec)
```

```
mysql> select * from booking;
+----------+-------------+-------------+------------+--------------+----+
| event_id | customer_id | num_tickets | total_cost | booking_date | id |
+----------+-------------+-------------+------------+--------------+----+
|        1 |           1 |           2 |        640 | 2021-09-12   |  1 |
|        1 |           3 |           5 |       3000 | 2024-03-15   |  2 |
|        1 |           4 |           3 |        960 | 2021-09-12   |  3 |
|        2 |           1 |           3 |      10800 | 2024-04-11   |  4 |
|        2 |           3 |           5 |      18000 | 2024-04-10   |  5 |
|        3 |           5 |          10 |      34000 | 2024-04-15   |  6 |
|        4 |           2 |           4 |      32000 | 2024-05-01   |  7 |
|        4 |           6 |           1 |       8000 | 2024-03-15   |  8 |
+----------+-------------+-------------+------------+--------------+----+
8 rows in set (0.00 sec)
```

```
mysql> select * from event;
+----+---------------------------------+------------+------------+-------------+-----------------+--------------+------------+----------+
| id | event_name                      | event_date | event_time | total_seats | available_seats | ticket_price | event_type | venue_id |
+----+---------------------------------+------------+------------+-------------+-----------------+--------------+------------+----------+
|  1 | Late Ms. Lata Mangeshkar Musical | 2021-09-12 | 20:00:00   |         320 |             270 |          600 | concert    |        3 |
|  2 | CSK vs RCB                      | 2024-04-11 | 19:30:00   |       23000 |               3 |         3600 | sports     |        2 |
|  3 | CSK vs RR                       | 2024-04-19 | 19:30:00   |       23000 |              10 |         3400 | sports     |        2 |
|  4 | MI vs KKR                       | 2024-05-01 | 15:30:00   |       28000 |             100 |         8000 | sports     |        1 |
+----+---------------------------------+------------+------------+-------------+-----------------+--------------+------------+----------+
4 rows in set (0.00 sec)
```

Task-2

2. Write SQL query to list all events.
=> select * from event;

3. Write a SQL query to select events with available tickets.
=> select * from event where available_seats>0;

4. Write a SQL query to select events name partial match with 'cup'.
=> select * from event where event_name LIKE '%cup%';

5. Write a SQL query to select events with ticket price range is between 1000 to 2500.
=> select * from event where ticket_price between 1000 AND 2500;

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

=> select * from event where event_date between '2021-09-01' AND '2023-12-12';

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 '%concert%';

8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.
=> select * from customer limit 5,5;

9. Write a SQL query to retrieve bookings details contains booked no of ticket more than 4.
=> select * from booking where num_tickets>4;

10. Write a SQL query to retrieve customer information whose phone number end with '000'
=> select * from customer where phone_number LIKE '%000';

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;

12. Write a SQL query to select events name not start with 'x', 'y', 'z'
=> select event_name from event where event_name not like 'x%' AND event_name not like 'y%' and event_name not like 'z%';

TASK 3

1. Write a SQL query to List Events and Their Average Ticket Prices.
=> select e.event_name, avg(b.total_cost) from event e, booking b where e.id=b.event_id group by e.event_name;

2. Write a SQL query to Calculate the Total Revenue Generated by Events.
=> select e.event_name, sum(b.total_cost) from event e JOIN booking b ON e.id=b.event_id group by e.event_name;

3. Write a SQL query to find the event with the highest ticket sales.
=> select e.event_name, SUM(b.num_tickets) as total_tickets_sold from event e JOIN booking b ON e.id=b.event_id group by e.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 e.event_name, SUM(num_tickets) from event e JOIN booking b ON e.id=b.event_id group by e.event_name;

5. Write a SQL query to Find Events with No Ticket Sales.
=> select * from event where id NOT IN (select e.id from event e JOIN booking b ON e.id=b.event_id)

6. Write a SQL query to Find the User Who Has Booked the Most Tickets.
=> select c.customer_name, SUM(num_tickets) as tickets_bought from customer c JOIN booking b ON c.id=b.customer_id group by c.customer_name order by tickets_bought DESC limit 1;

8. Write a SQL query to calculate the average Ticket Price for Events in Each Venue.
=> select v.name, avg(ticket_price) as average_price from event e, venue v where v.id=e.venue_id group by v.name;

9. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.
=> select e.event_type, SUM(num_tickets) from event e JOIN booking b ON e.id=b.event_id group by event_type;

11. Write a SQL query to list users who have booked tickets for multiple events.
=> SELECT * FROM customer where id in (select customer_id from booking group by customer_id having count(event_id)>1);12.

12. Write a SQL query to calculate the Total Revenue Generated by Events for Each User.
=> select c.customer_name, SUM(b.total_cost) from customer c, booking b where c.id=b.customer_id group by c.customer_name;

13. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.
=> select e.event_type, v.name, AVG(e.ticket_price) from event e, venue v where e.venue_id=v.id group by e.event_type, v.name;

TASK 4

1. Calculate the Average Ticket Price for Events in Each Venue.
=> select v.name, AVG(e.ticket_price) from venue v, event e where e.venue_id=v.id group by v.name;

2. Find Events with More Than 50% of Tickets Sold.
=> select * from event WHERE total_seats-available_seats>total_seats/2;

3. Calculate the Total Number of Tickets Sold for Each Event.
=> select event_name, SUM(total_seats-available_seats) as ticket_sold from event group by event_name;

4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.
=> select * from customer where id not in (select customer_id from booking);

5. List Events with No Ticket Sales Using a NOT IN Subquery.
=> select * from event where id not in (select event_id from booking);

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;

7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.
=> select * 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 c.customer_name, SUM(b.total_cost) from customer c, booking b where c.id=b.customer_id group by c.customer_name;

9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.
=> select * 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 name='mumbai')));

10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with
GROUP BY.
=> select event_type, SUM(total_seats-available_seats) from event group by event_type;

12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery
=> select v.name, AVG(e.ticket_price) from venue v JOIN event e ON e.venue_id=v.id group by v.name;