ONLINE BUS RESERVATION SYSTEM

A PROJECT REPORT

Submitted by

RAVI PRAKASH YADAV [RA2211003010231]

ANIKET KUMAR [RA2211003010236]

Under the Guidance of

Dr. B Prakash

(Assistant Professor, Department of Computing Technologies)

in partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES,

COLLEGE OF ENGINEERING & TECHNOLOGY,

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY,

KATTANKULATHUR – 603203

APRIL 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

Certified that 21CSE253T Internet of Things Mini Project Report titled **"ONLINE BUS RESERVATION SYSTEM"** is the bonafide work of **RAVI PRAKASH YADAV [RA2211003010231], ANIKET KUMAR[RA2211003010236]** who carried out the project work under mysupervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlieroccasion for this or any other candidate.

Faculty Incharge
Dr. B Prakash
Associate Professor
Department of Computing Technologies
SRMIST, KTR

Dr. M. PUSHPALATHA
HEAD OF THE DEPARTMENT
Department of Computing Technologies

ABSTRACT

The Online Bus Reservation System (OBRS) redefines bus travel with its intuitive interface, real-time updates on availability and pricing, secure payment gateway, and robust management tools. It enhances passenger convenience and operator efficiency, transforming the bus transportation sector. OBRS simplifies booking processes, provides accurate information, and ensures seamless transactions, fostering a harmonious relationship between travelers and service providers while revolutionizing the dynamics of bus travel in the digital age.

TABLE OF CONTENTS

INTRODUCTION

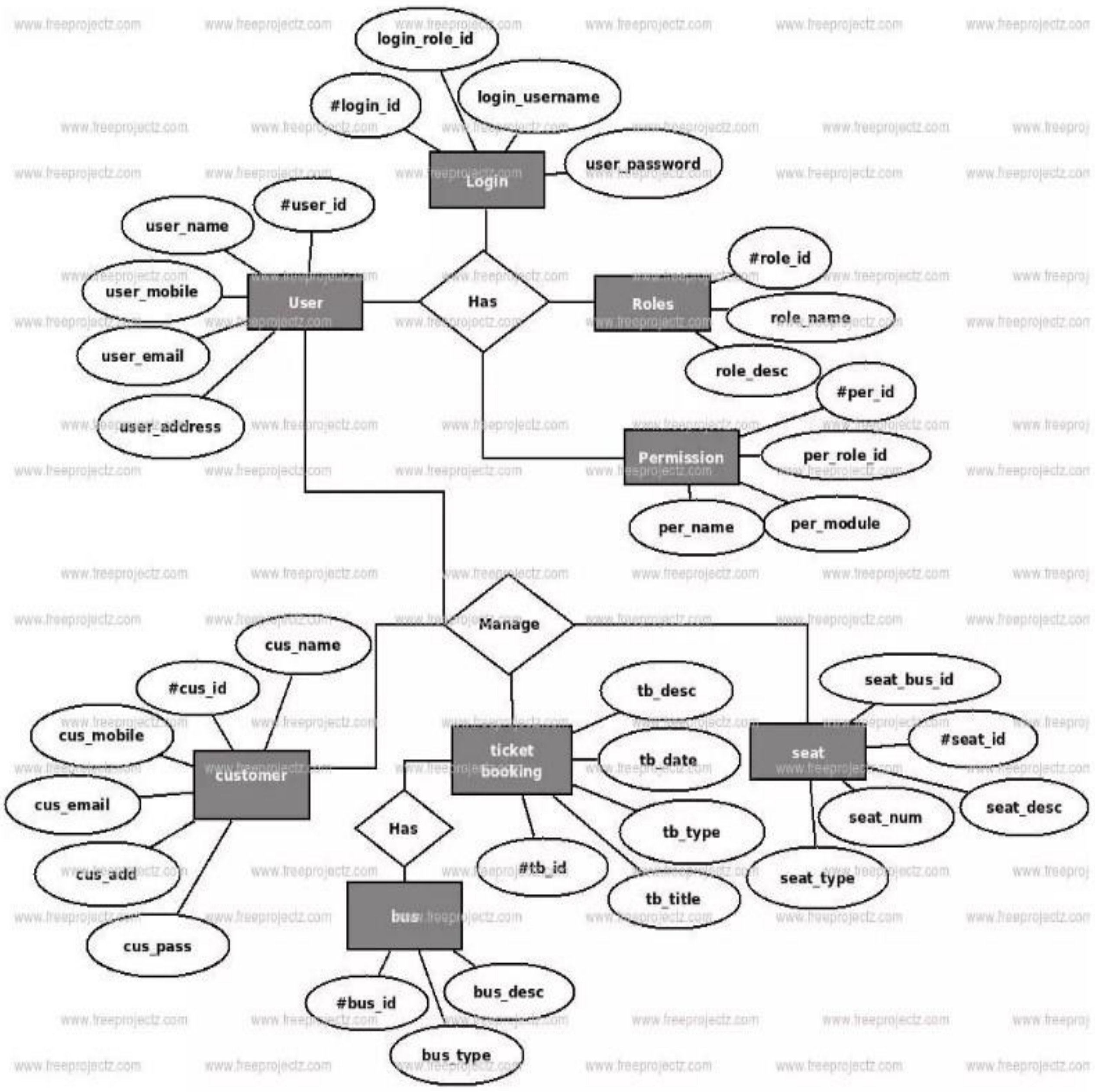
The Online Bus Reservation System (OBRS) represents a transformative leap in the realm of bus transportation, harnessing the power of technology to streamline booking processes and enhance the overall travel experience for passengers. In an era characterized by digitalization and connectivity, OBRS emerges as a pivotal solution to address the evolving needs and expectations of modern travelers while optimizing operational efficiency for bus operators.

Traditionally, the process of booking bus tickets has been plagued by inefficiencies, inconvenience, and often, uncertainty regarding seat availability and pricing. Travelers often faced challenges such as long queues at ticket counters, limited access to information on routes and schedules, and the risk of last-minute cancellations or overbooking.

In response to these challenges, OBRS offers a comprehensive and user-friendly platform that revolutionizes the way bus tickets are reserved and managed. By leveraging intuitive interfaces, real-time updates, secure payment gateways, and advanced management tools, OBRS empowers both passengers and operators with greater flexibility, transparency, and control over the booking process.

In essence, the Online Bus Reservation System (OBRS) represents a paradigm shift in the way bus travel is bridging the gap between traditional practices and modern expectations. As technology continues to evolve and reshape the transportation landscape, OBRS stands at the forefront, driving innovation, efficiency, and convenience in the bus transportation sector.

ER DIAGRAM


login_role_id
#login_id
login_username
user_password
Login
#user_id
user_name
user_mobile
user_email
user_address
User
Has
Roles
#role_id
role_name
role_desc
#per_id
Permission
per_role_id
per_name
per_module
Manage
cus_name
#cus_id
cus_mobile
customer
cus_email
cus_add
cus_pass
tb_desc
ticket booking
tb_date
tb_type
#tb_id
tb_title
Has
bus
#bus_id
bus_desc
bus_type
seat_bus_id
seat
#seat_id
seat_num
seat_desc
seat_type
www.freeprojectz.com

INFORMATION OF ENTITIES

In total we have eight entities and information of each entity is mentioned below:-

1. **Passengers:** Individuals who intend to travel by bus and utilize the OBRS platform to search for routes, check availability, make bookings, and manage their travel itineraries.

2. **Bus_Operator:** Companies or organizations that own and operate buses, providing transportation services to passengers. Bus operators utilize the OBRS platform to manage their fleet, publish schedules, allocate seats, and monitor bookings.

3. **Admin/User/Administrators:** Personnel responsible for overseeing and managing the OBRS platform. This includes system administrators who maintain the technical infrastructure and user administrators who handle user accounts, permissions, and support.

4. **Buses :** The physical vehicles used for transportation, categorized based on factors such as capacity, amenities, and route coverage. Each bus entity within the system is associated with specific attributes such as seating capacity, amenities available, and operational status.

5. **Routes :** The predefined travel itineraries followed by buses, encompassing origin and destination points, intermediate stops, and schedules. Route entities include details such as distance, duration, frequency, and associated fares.

6. Bookings : Reservations made by passengers to secure seats on specific buses and routes. Booking entities contain information such as passenger details, travel dates, seat assignments, and payment status.

RELATIONSHIP BETWEEN ENTITIES

1 Passenger-Reservation:

A passenger can have multiple reservations, indicating they've booked seats on different buses for various journeys.
Each reservation is linked to one passenger, representing the individual who made the booking.

2 Passenger-Payment:

A passenger makes payments to confirm their reservations.
Each payment is associated with one reservation, indicating which booking it corresponds to.
A passenger may have multiple payments if they've made multiple reservations.

3 Bus-Seat:

Each bus has multiple seats available for booking.
Seats belong to a specific bus, indicating their physical location within that vehicle.

4 Reservation-Seat:

A reservation links a passenger with a specific seat on a particular bus for a defined journey.
Each reservation is associated with one or more seats, representing the seats booked by the passenger(s) for that journey.

5 Reservation-Bus:

A reservation is made for a specific journey on a particular bus.
Each reservation is linked to one bus, indicating the vehicle the passenger(s) will travel on.

6 Reservation-Route:

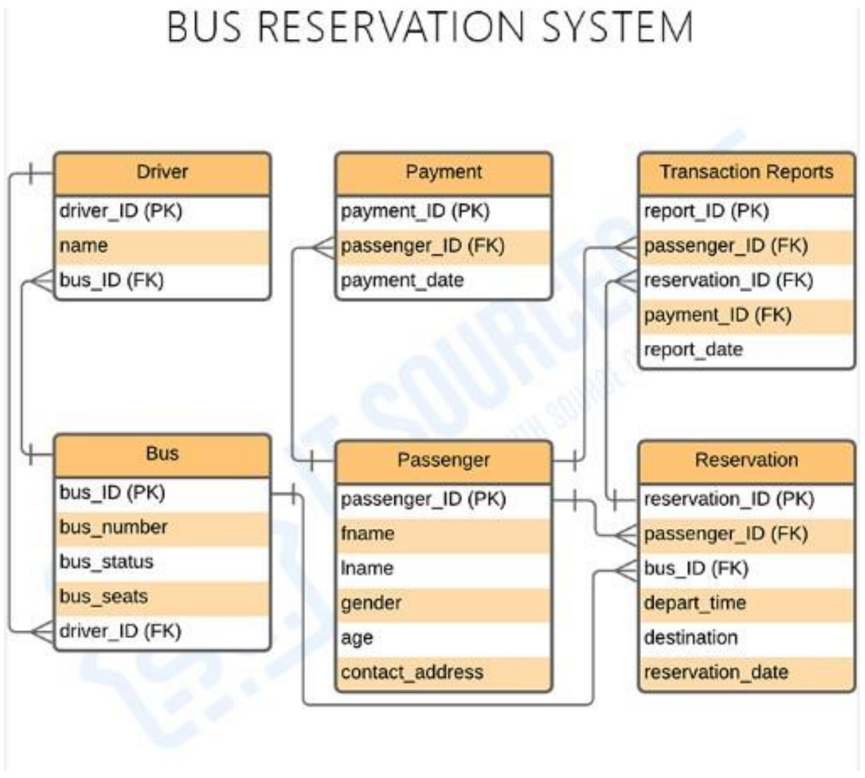
A reservation is made for a journey along a specific route.
Each reservation is associated with one route, indicating the path the bus will take during the journey.

7 Reservation-Schedule:

A reservation is made for a journey at a specific time as per the schedule. Each reservation is linked to one schedule, indicating the departure and arrival times for the journey.

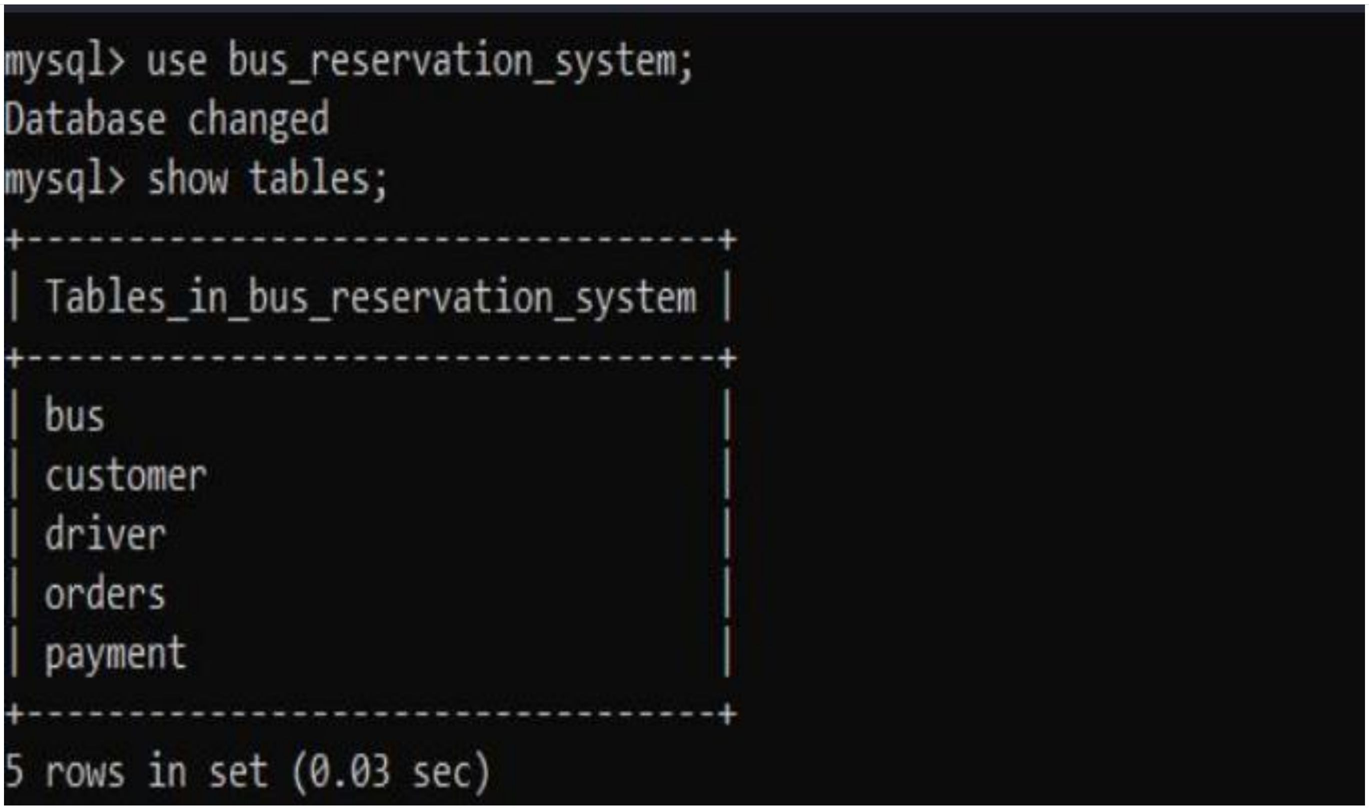
8 Route-Schedule:

Each route has multiple schedules defining the departure and arrival times for buses traveling along that route. Schedules are associated with a specific route, indicating when buses depart and arrive at various stops along the route.



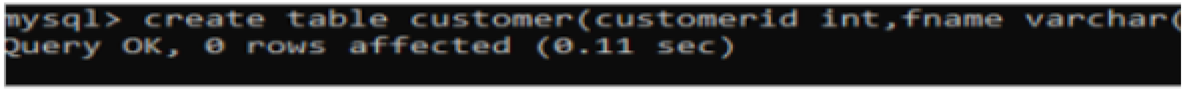
CREATION OF DATABASE TABLES FOR THE PROJECT

- **CREATE TABLE BUS (bus_id (Primary Key), bus_number,capacity,type,**



```
mysql> use bus_reservation_system;
Database changed
mysql> show tables;
+----------------------------------+
| Tables_in_bus_reservation_system |
+----------------------------------+
| bus                              |
| customer                         |
| driver                           |
| orders                           |
| payment                          |
+----------------------------------+
5 rows in set (0.03 sec)
```

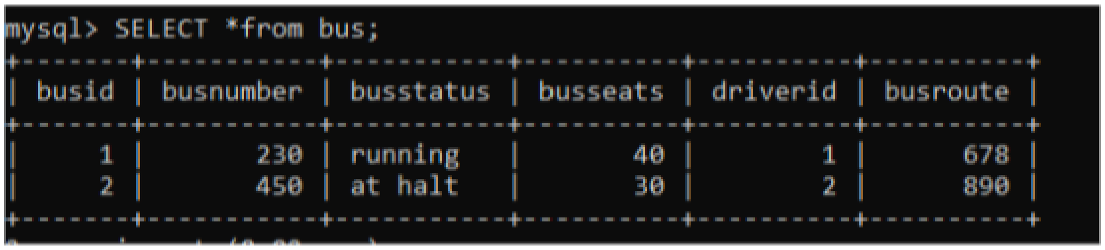
Create the tables DEPT and EMP as described below



```
mysql> create table customer(customerid int,fname varchar(
Query OK, 0 rows affected (0.11 sec)
```

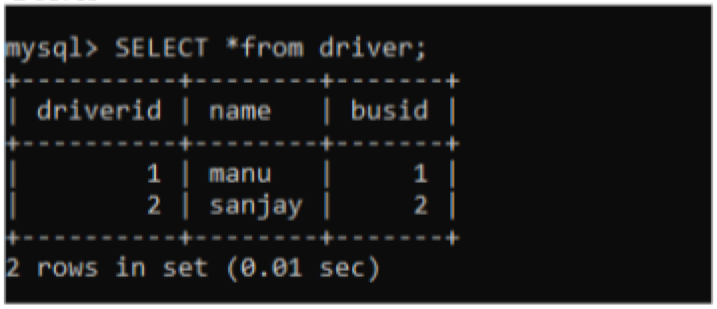
DEPT

bus



```
mysql> SELECT *from bus;
+-------+-----------+-----------+----------+----------+----------+
| busid | busnumber | busstatus | busseats | driverid | busroute |
+-------+-----------+-----------+----------+----------+----------+
|     1 |       230 | running   |       40 |        1 |      678 |
|     2 |       450 | at halt   |       30 |        2 |      890 |
+-------+-----------+-----------+----------+----------+----------+
```

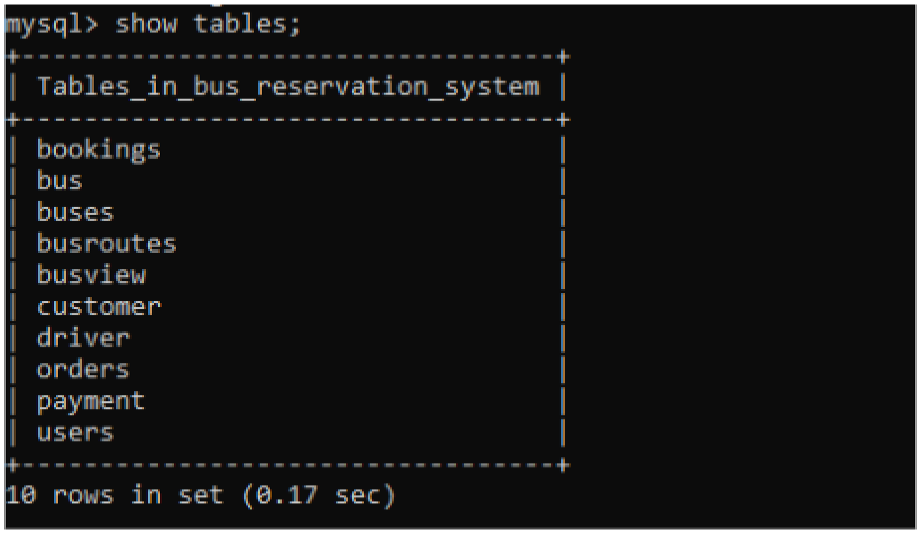
Driver



```
mysql> SELECT *from driver;
+----------+--------+-------+
| driverid | name   | busid |
+----------+--------+-------+
|        1 | manu   |     1 |
|        2 | sanjay |     2 |
+----------+--------+-------+
2 rows in set (0.01 sec)
```

Q3) List name of the tables created by the user

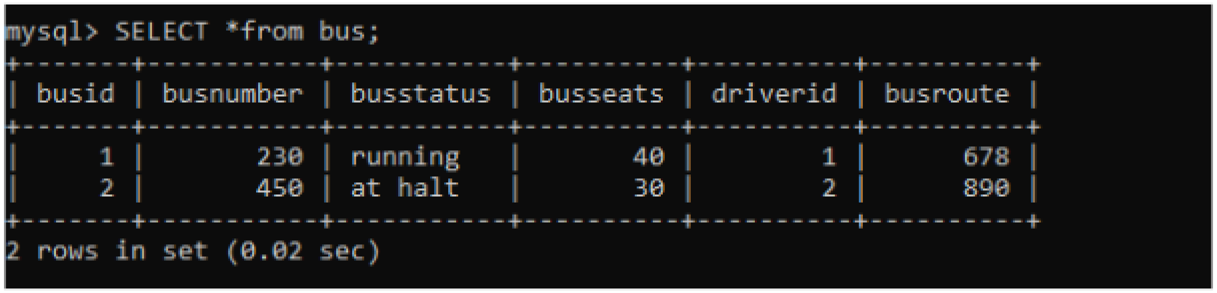
SQL>select * show table;



```
mysql> show tables;
+----------------------------------+
| Tables_in_bus_reservation_system |
+----------------------------------+
| bookings                         |
| bus                              |
| buses                            |
| busroutes                        |
| busview                          |
| customer                         |
| driver                           |
| orders                           |
| payment                          |
| users                            |
+----------------------------------+
10 rows in set (0.17 sec)
```

Q4) Describe tables owned by the user

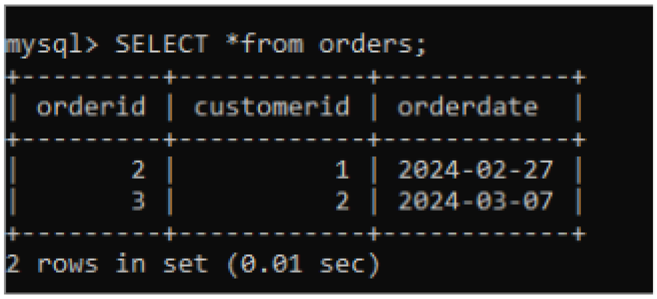
SQL> SELECT * FROM bus_tables;



```
mysql> SELECT *from bus;
+-------+-----------+-----------+----------+----------+----------+
| busid | busnumber | busstatus | busseats | driverid | busroute |
+-------+-----------+-----------+----------+----------+----------+
|     1 |       230 | running   |       40 |        1 |      678 |
|     2 |       450 | at halt   |       30 |        2 |      890 |
+-------+-----------+-----------+----------+----------+----------+
2 rows in set (0.02 sec)
```

Q5) View distinct object types owned by the user

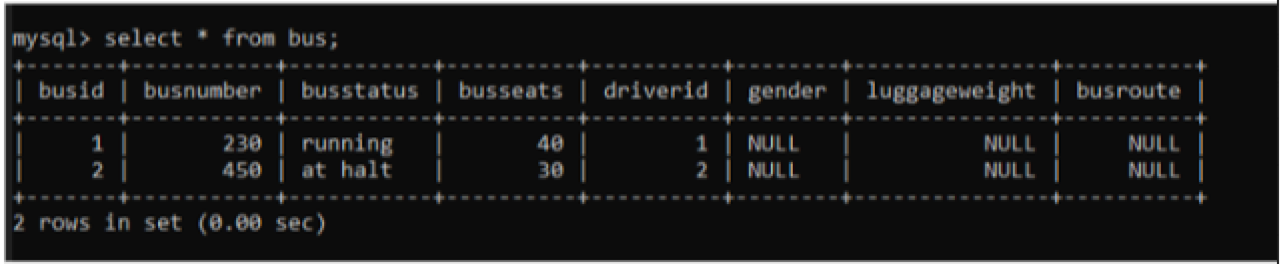
SQL> SELECT DISTINCT order_id FROM customer_id;



```
mysql> SELECT *from orders;
+---------+------------+------------+
| orderid | customerid | orderdate  |
+---------+------------+------------+
|       2 |          1 | 2024-02-27 |
|       3 |          2 | 2024-03-07 |
+---------+------------+------------+
2 rows in set (0.01 sec)
```

- *Drop Column*

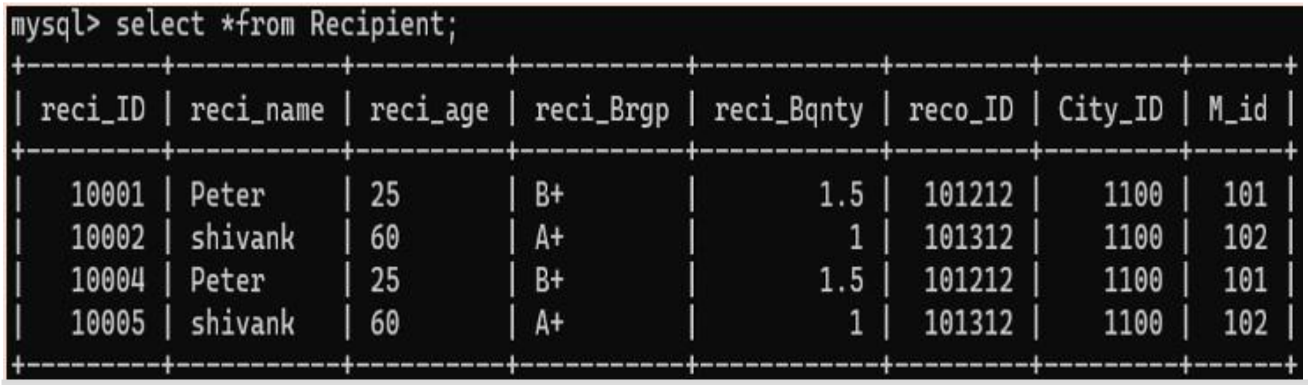
  ALTER TABLE bus_table column
  ADD BUS_ID VARCHER(10) NOT NULL;
  ALTER TABLE BUS_ID DROP COLUMN BUS_ID;



```
mysql> select * from bus;
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
| busid | busnumber | busstatus | busseats | driverid | gender | luggageweight  | busroute |
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
|     1 |       230 | running   |       40 |        1 | NULL   |           NULL |     NULL |
|     2 |       450 | at halt   |       30 |        2 | NULL   |           NULL |     NULL |
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
2 rows in set (0.00 sec)
```

- *Modify Column*

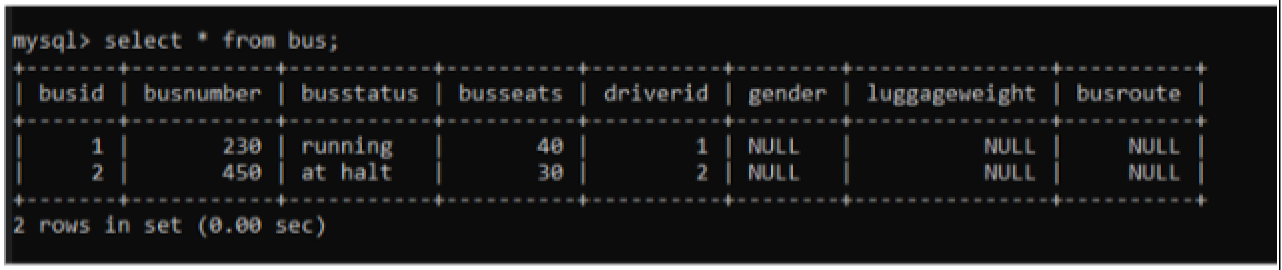
  ALTER TABLE *table* MODIFY(*column data type* [DEFAULT *expr*]
  [, *column data type*]...);



```
mysql> select *from Recipient;
+---------+-----------+----------+-----------+------------+---------+---------+------+
| reci_ID | reci_name | reci_age | reci_Brgp | reci_Bqnty | reco_ID | City_ID | M_id |
+---------+-----------+----------+-----------+------------+---------+---------+------+
|   10001 | Peter     | 25       | B+        |        1.5 |  101212 |    1100 |  101 |
|   10002 | shivank   | 60       | A+        |          1 |  101312 |    1100 |  102 |
|   10004 | Peter     | 25       | B+        |        1.5 |  101212 |    1100 |  101 |
|   10005 | shivank   | 60       | A+        |          1 |  101312 |    1100 |  102 |
+---------+-----------+----------+-----------+------------+---------+---------+------+
```

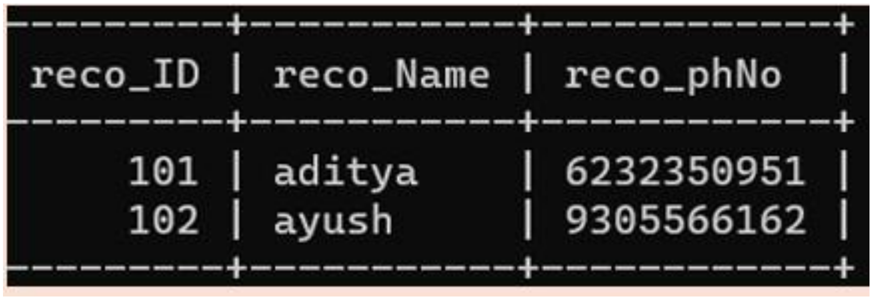
Q7) Drop the column BUS_ID from the table BUS TABLE

SQL >



```
mysql> select * from bus;
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
| busid | busnumber | busstatus | busseats | driverid | gender | luggageweight  | busroute |
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
|     1 |       230 | running   |       40 |        1 | NULL   |           NULL |     NULL |
|     2 |       450 | at halt   |       30 |        2 | NULL   |           NULL |     NULL |
+-------+-----------+-----------+----------+----------+--------+----------------+----------+
2 rows in set (0.00 sec)
```

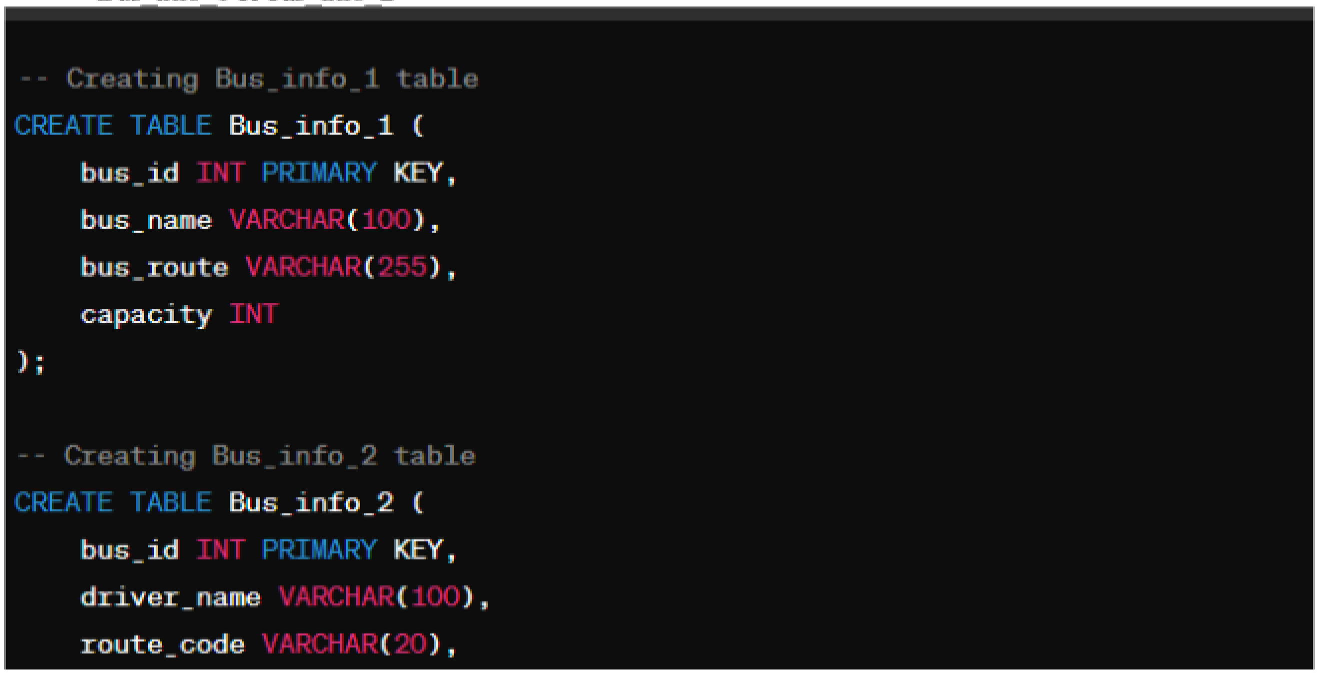
CREATE TABLE Recording_Staff (reco_ID int NOT NULL PRIMARY KEY, reco_Name varchar(100) NOT NULL, reco_phNo bigint);



```
---------+-----------+------------+
reco_ID | reco_Name | reco_phNo  |
---------+-----------+------------+
    101 | aditya    | 6232350951 |
    102 | ayush     | 9305566162 |
---------+-----------+------------+
```

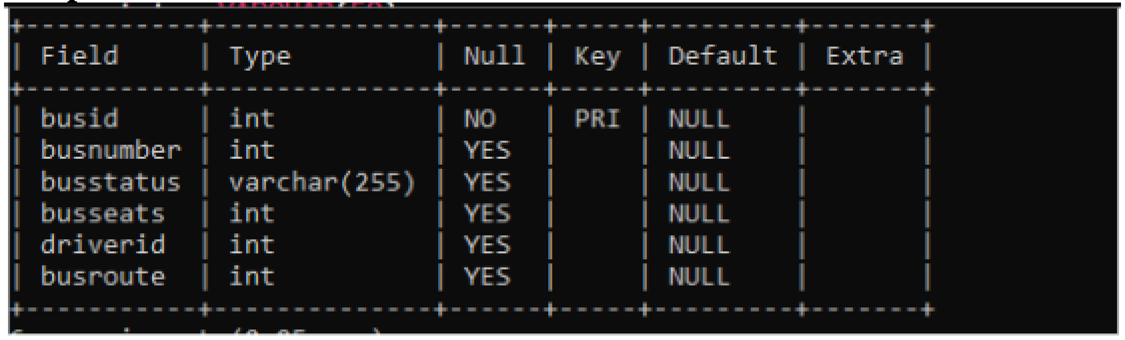
COMPLEX QURIES

Q1) Create the following tables :
Bus_info_1 & bus_info_2



```
-- Creating Bus_info_1 table
CREATE TABLE Bus_info_1 (
    bus_id INT PRIMARY KEY,
    bus_name VARCHAR(100),
    bus_route VARCHAR(255),
    capacity INT
);

-- Creating Bus_info_2 table
CREATE TABLE Bus_info_2 (
    bus_id INT PRIMARY KEY,
    driver_name VARCHAR(100),
    route_code VARCHAR(20),
```

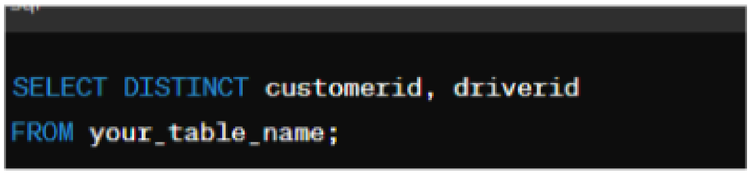
Output:



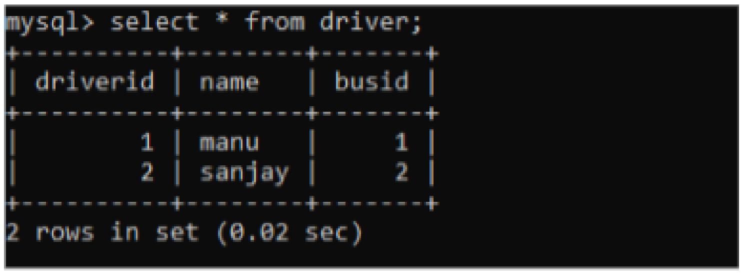
```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| busid     | int          | NO   | PRI | NULL    |       |
| busnumber | int          | YES  |     | NULL    |       |
| busstatus | varchar(255) | YES  |     | NULL    |       |
| busseats  | int          | YES  |     | NULL    |       |
| driverid  | int          | YES  |     | NULL    |       |
| busroute  | int          | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

Q2) List the names of distinct customersid driverid

SQL>



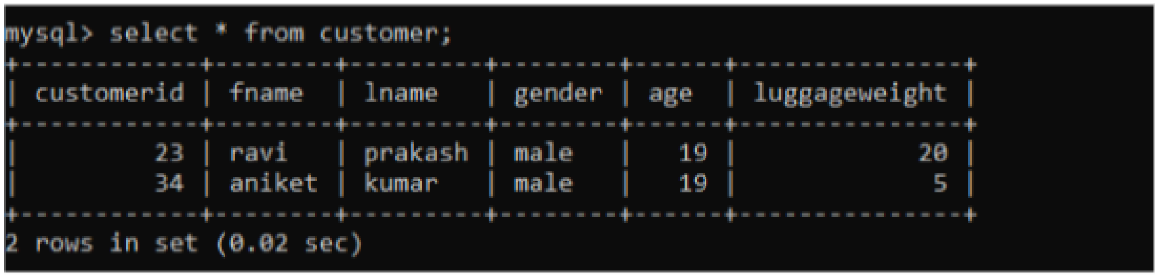
```
SELECT DISTINCT customerid, driverid
FROM your_table_name;
```



```
mysql> select * from driver;
+----------+--------+-------+
| driverid | name   | busid |
+----------+--------+-------+
|        1 | manu   |     1 |
|        2 | sanjay |     2 |
+----------+--------+-------+
2 rows in set (0.02 sec)
```

Q3) List the names of customers (with duplicates) who have either loan or account

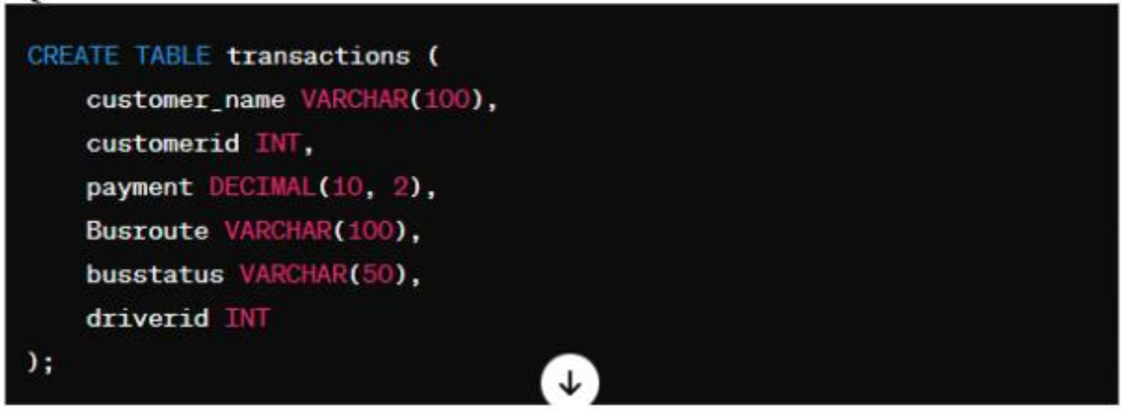
```
SQL> SELECT customer_name
     FROM customers;
```



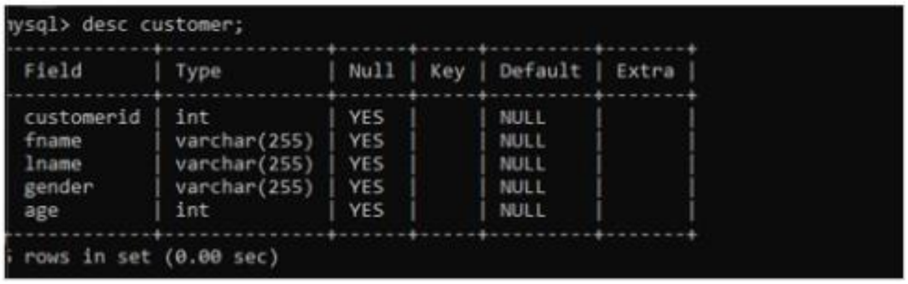
```
mysql> select * from customer;
+------------+--------+---------+--------+------+---------------+
| customerid | fname  | lname   | gender | age  | luggageweight |
+------------+--------+---------+--------+------+---------------+
|         23 | ravi   | prakash | male   |   19 |            20 |
|         34 | aniket | kumar   | male   |   19 |             5 |
+------------+--------+---------+--------+------+---------------+
2 rows in set (0.02 sec)
```

Q8) **list customer name; customerid; payment; Busroute; busstatus; driverid;**

SQL>



```
CREATE TABLE transactions (
    customer_name VARCHAR(100),
    customerid INT,
    payment DECIMAL(10, 2),
    Busroute VARCHAR(100),
    busstatus VARCHAR(50),
    driverid INT
);
```

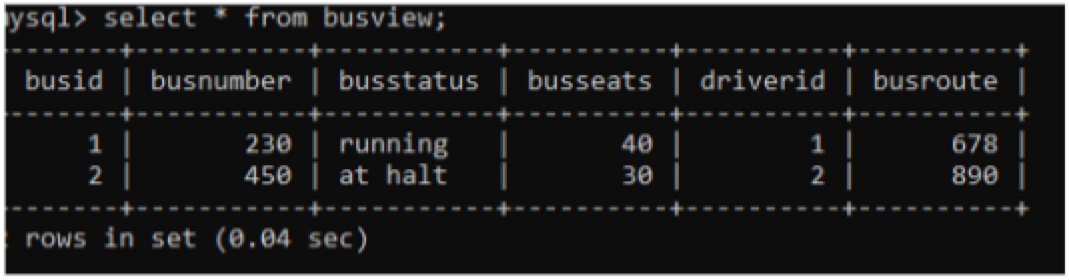


```
ysql> desc customer;
+-----------+--------------+------+-----+---------+-------+
 Field      | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
 customerid | int          | YES  |     | NULL    |       |
 fname      | varchar(255) | YES  |     | NULL    |       |
 lname      | varchar(255) | YES  |     | NULL    |       |
 gender     | varchar(255) | YES  |     | NULL    |       |
 age        | int          | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
 rows in set (0.00 sec)
```

Execute the following query and then try to delete the row with dept no 20. Now write in words that you understand

SQL> create view busroute AS

select * from busid where driver



```
ysql> select * from busview;
-------+-----------+-----------+----------+----------+----------+
 busid | busnumber | busstatus | busseats | driverid | busroute |
-------+-----------+-----------+----------+----------+----------+
     1 |       230 | running   |       40 |        1 |      678 |
     2 |       450 | at halt   |       30 |        2 |      890 |
-------+-----------+-----------+----------+----------+----------+
 rows in set (0.04 sec)
```

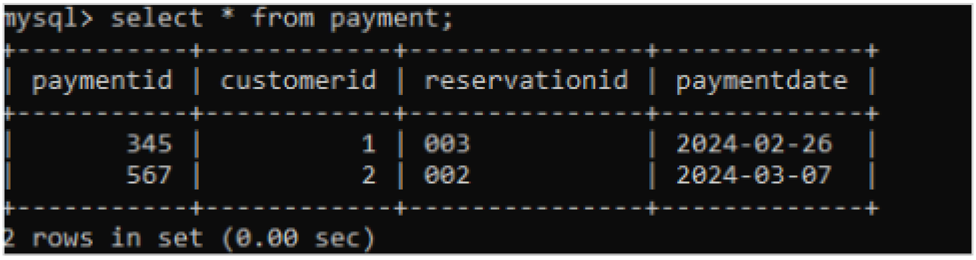
Q5) List the names of payment coustomer

SQL>

```
CREATE TABLE employees (
   employee_id INT PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   email VARCHAR(100),
   hire_date DATE,
   salary DECIMAL(10, 2)
);
```

;

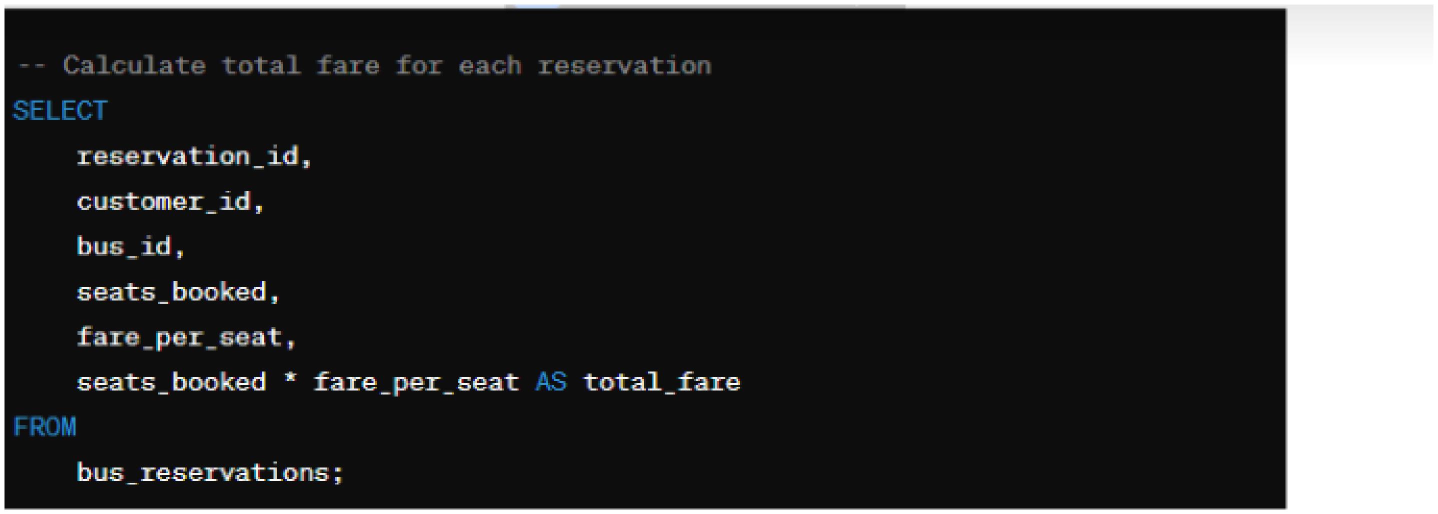
OUTPUT:-



```
mysql> select * from payment;
+-----------+------------+---------------+-------------+
| paymentid | customerid | reservationid | paymentdate |
+-----------+------------+---------------+-------------+
|       345 |          1 | 003           | 2024-02-26  |
|       567 |          2 | 002           | 2024-03-07  |
+-----------+------------+---------------+-------------+
2 rows in set (0.00 sec)
```

J

ADDING A CONSTRAINT



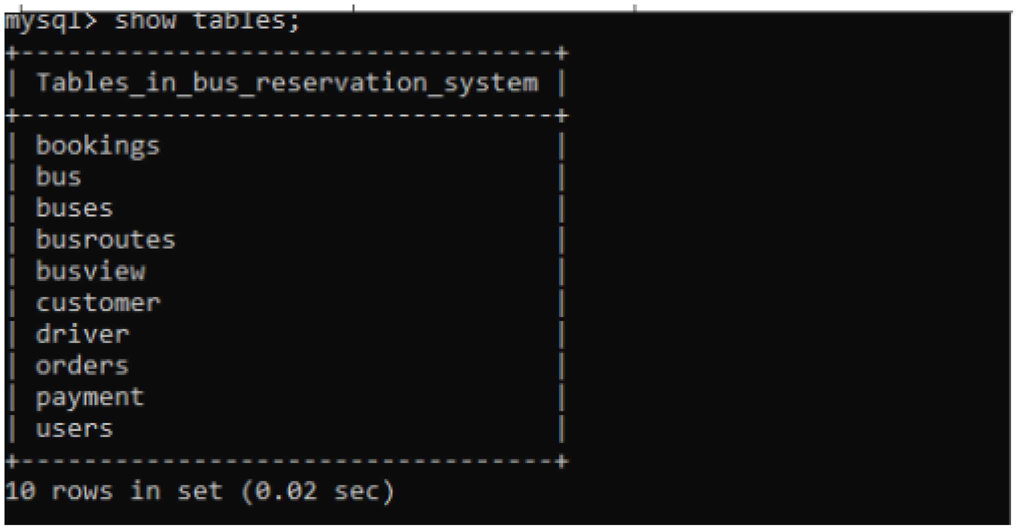
```
-- Calculate total fare for each reservation
SELECT
    reservation_id,
    customer_id,
    bus_id,
    seats_booked,
    fare_per_seat,
    seats_booked * fare_per_seat AS total_fare
FROM
    bus_reservations;
```

Q8. Miscellaneous Functions

Functions	Value Returned	Input	Output
Uid	User id	Select uid from dual;	
User	User name	Select user from dual;	
Vsize(n)	Storage size of v	Select vsize('hello') from dual;	
NVL(exp1,exp2)	Returns exp1 if not null, otherwise returns exp2.	Select nvl(comm,50) from emp where empno=7369;	

Q5. Character Functions

Functions	Value Returned	Input
initcap(char)	First letter of each word capitalized	Select initcap('database management') from dual;
lower(char)	Lower case	Select lower('WELCOME') from dual;
upper(char)	Upper case	Select upper('srmist') from dual;
ltrim(char, set)	Initial characters removed up to the character not in set.	Select ltrim('lordourgod','lord') from dual;
rtrim(char, set)	Final characters removed after the last character not in set.	Select rtrim('godlovesyou','you') from dual;
translate(char, from, to)	Translate 'from' by 'to' in char.	Select translate('jack','j','b') from dual;
replace(char, search, repl)	Replace 'search' string by 'repl' string in 'char'.	Select replace('jack and jue','j','bl') from dual;
substr(char, m, n)	Substring of 'char' at 'm' of size 'n' char long.	Select substr('wages of sin is death',10,3) from dual;



```
mysql> show tables;
+--------------------------------+
| Tables_in_bus_reservation_system |
+--------------------------------+
| bookings                       |
| bus                            |
| buses                          |
| busroutes                      |
| busview                        |
| customer                       |
| driver                         |
| orders                         |
| payment                        |
| users                          |
+--------------------------------+
10 rows in set (0.02 sec)
```

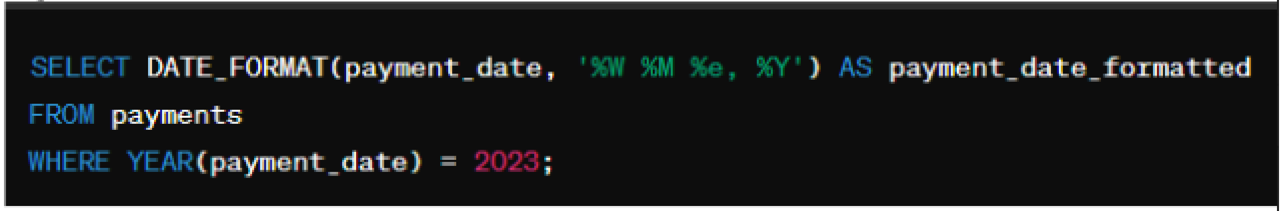
SCALAR FUNCTIONS

Q1) List the date of payment date who registered in 2023 in a format like 'WEDNESDAY JANUARY 12, 1983'

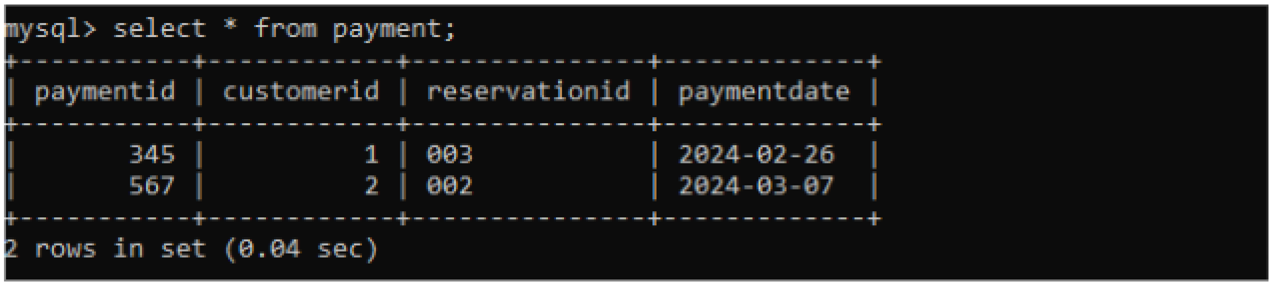
(Hint: DAY : Day of the week, MONTH : Name of the month, DD: Day of the month, and YYYY : Year)

OUTPUT:-

SQL>



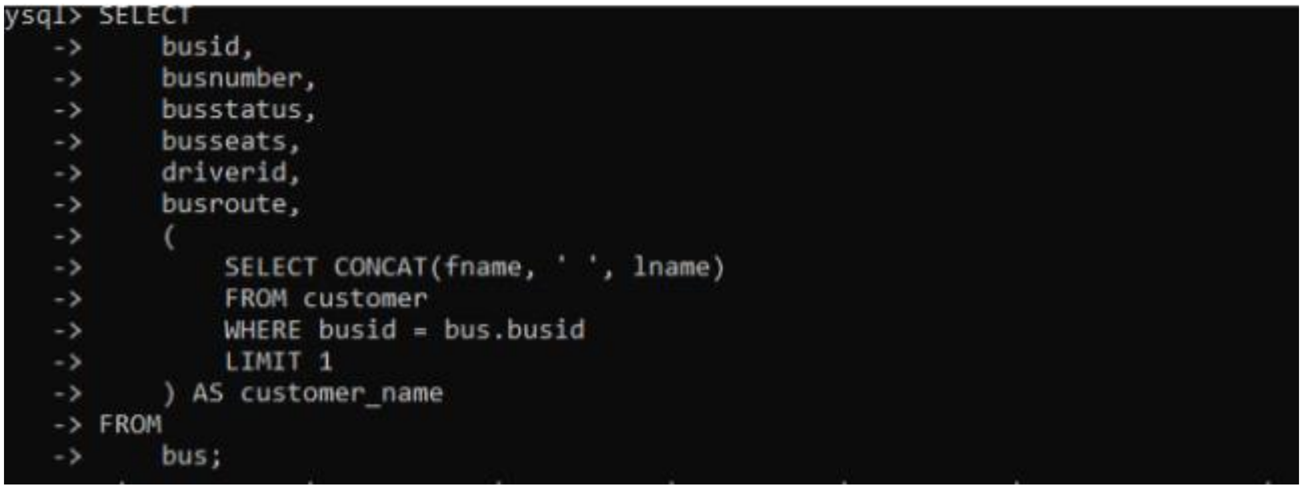
```
SELECT DATE_FORMAT(payment_date, '%W %M %e, %Y') AS payment_date_formatted
FROM payments
WHERE YEAR(payment_date) = 2023;
```



```
mysql> select * from payment;
+-----------+------------+---------------+-------------+
| paymentid | customerid | reservationid | paymentdate |
+-----------+------------+---------------+-------------+
|       345 |          1 | 003           | 2024-02-26  |
|       567 |          2 | 002           | 2024-03-07  |
+-----------+------------+---------------+-------------+
2 rows in set (0.04 sec)
```

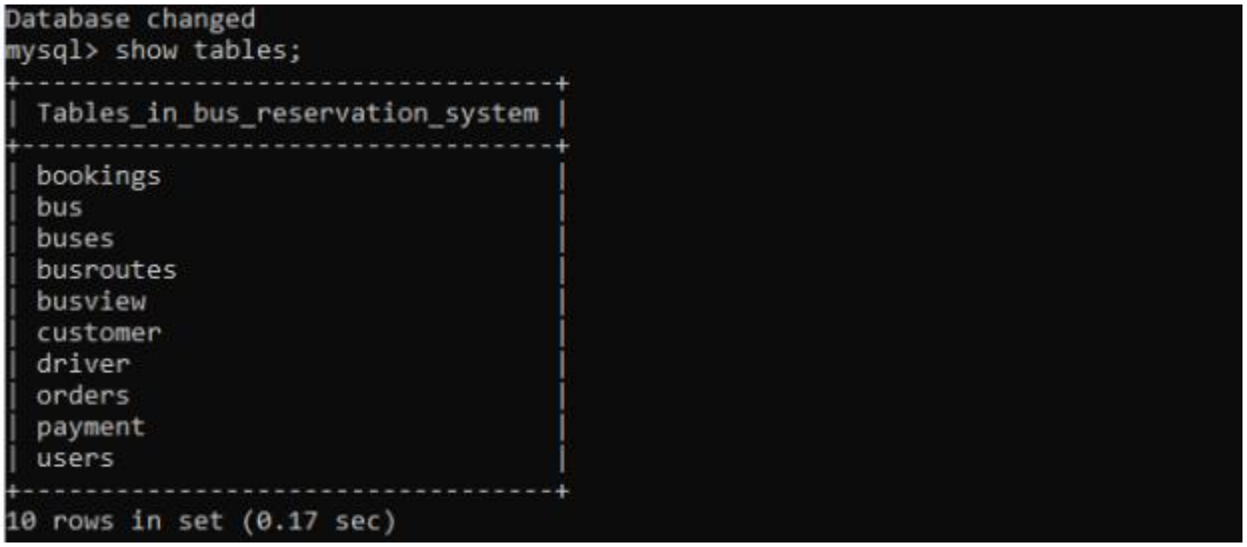
BASIC SELECT STATEMENTS

Update all the records of bus table



```
ysql> SELECT
    ->     busid,
    ->     busnumber,
    ->     busstatus,
    ->     busseats,
    ->     driverid,
    ->     busroute,
    ->     (
    ->         SELECT CONCAT(fname, ' ', lname)
    ->         FROM customer
    ->         WHERE busid = bus.busid
    ->         LIMIT 1
    ->     ) AS customer_name
    -> FROM
    ->     bus;
```

OUTPUT:-



```
Database changed
mysql> show tables;
+--------------------------------+
| Tables_in_bus_reservation_system |
+--------------------------------+
| bookings                       |
| bus                            |
| buses                          |
| busroutes                      |
| busview                        |
| customer                       |
| driver                         |
| orders                         |
| payment                        |
| users                          |
+--------------------------------+
10 rows in set (0.17 sec)
```

`,

IMPLEMENTATION OF TRIGGERS

```
 rigger to update seat availability after a reservation is made
CREATE TRIGGER update_seat_availability
AFTER INSERT ON reservations
FOR EACH ROW
BEGIN
   UPDATE buses
   SET available_seats = available_seats - 1
   WHERE bus_id = NEW.bus_id;
END;

-- Trigger to manage waitlist when a reservation is canceled
CREATE TRIGGER manage_waitlist
AFTER DELETE ON reservations
FOR EACH ROW
BEGIN
   DECLARE waitlist_count INT;
   SELECT COUNT(*) INTO waitlist_count FROM waitlist WHERE bus_id =
OLD.bus_id;
   IF waitlist_count > 0 THEN
      DELETE FROM waitlist WHERE bus_id = OLD.bus_id LIMIT 1; -- Release seat
for the next customer in waitlist
   END IF;
END;

-- Trigger to verify payment after reservation is made
CREATE TRIGGER verify_payment
BEFORE INSERT ON reservations
FOR EACH ROW
BEGIN
   IF NEW.payment_status != 'completed' THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Payment verification
failed';
   END IF;
END;

-- Trigger to send notification after reservation is made
```

```
CREATE TRIGGER send_notification
AFTER INSERT ON reservations
FOR EACH ROW
BEGIN
   INSERT INTO notifications (user_id, message)
   VALUES (NEW.user_id, 'Your reservation for bus ' || NEW.bus_id || ' has been
confirmed.');
END;

-- Trigger to enforce data integrity checks
CREATE TRIGGER check_departure_date
BEFORE INSERT ON reservations
FOR EACH ROW
BEGIN
   IF NEW.departure_date < CURDATE() THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Departure date cannot be
in the past';
   END IF;
END;

-- Trigger to capture reservation data for reporting
CREATE TRIGGER capture_reservation_data
AFTER INSERT ON reservations
FOR EACH ROW
BEGIN
   INSERT INTO reservation_logs (reservation_id, user_id, bus_id, booking_time)
   VALUES (NEW.reservation_id, NEW.user_id, NEW.bus_id, NOW());
END;

-- Trigger to award loyalty points
CREATE TRIGGER award_loyalty_points
AFTER INSERT ON reservations
FOR EACH ROW
BEGIN
   DECLARE reservation_count INT;
   SELECT COUNT(*) INTO reservation_count FROM reservations WHERE user_id =
NEW.user_id;
   IF reservation_count >= 10 THEN
```

```
      UPDATE users SET loyalty_points = loyalty_points + 100 WHERE user_id =
NEW.user_id;
   END IF;
END;
```

IMPLEMENTATION OF CURSORS

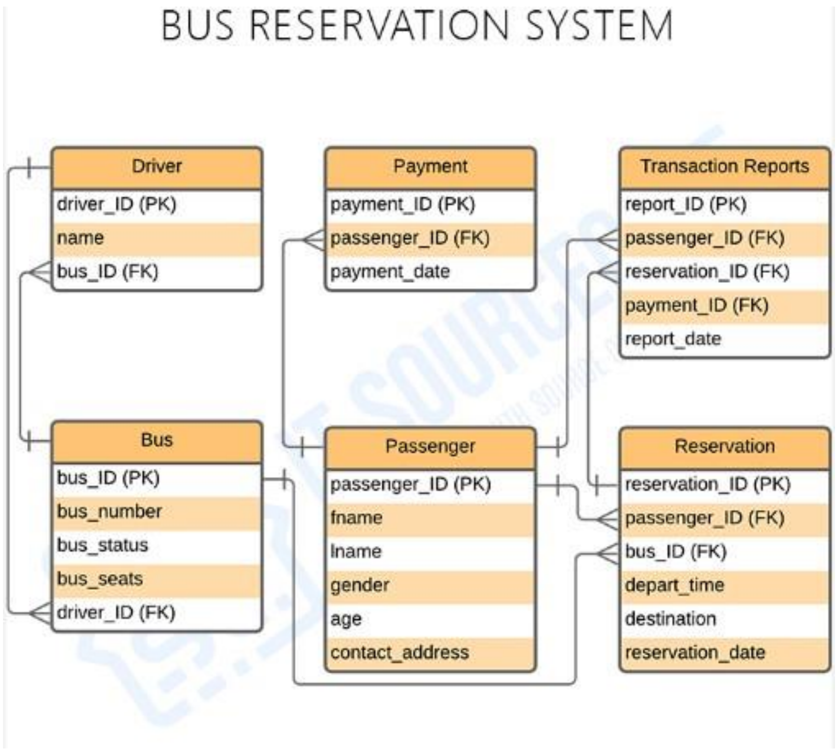
Declare the cursor

```
DECLARE seat_cursor CURSOR FOR
SELECT bus_id, available_seats FROM buses;

-- Open the cursor
OPEN seat_cursor;

-- Fetch the data from the cursor
FETCH seat_cursor INTO @bus_id, @available_seats;

-- Loop through the cursor result set
WHILE @@FETCH_STATUS = 0 DO
   -- Check seat availability and manage waitlist
   IF @available_seats <= 0 THEN
      -- Insert into waitlist
      INSERT INTO waitlist (bus_id, user_id, timestamp)
      VALUES (@bus_id, @user_id, NOW());
   ELSE
      -- Update available seats
      UPDATE buses
      SET available_seats = available_seats - 1
      WHERE bus_id = @bus_id;
      -- Send notification
      INSERT INTO notifications (user_id, message)
      VALUES (@user_id, CONCAT('Your reservation for bus ', @bus_id, ' has been
confirmed.'));
   END IF;

   -- Fetch the next row from the cursor
   FETCH seat_cursor INTO @bus_id, @available_seats;
END WHILE;

-- Close the cursor
CLOSE seat_cursor;
```

NORMALIZATION ONLINE BUS RESERVATION SYSTEM

-- 1. Buses (bus_id PK, bus_name, departure_time, arrival_time, route_id FK)
2. Routes (route_id PK, origin, destination, distance)
3. Users (user_id PK, username, email, password)
4. Seats (seat_id PK, bus_id FK, seat_number, availability_status)
5. Reservations (reservation_id PK, user_id FK, bus_id FK, reservation_date)
6. Payments (payment_id PK, reservation_id FK, amount, payment_date, payment_status)
7. Waitlist (waitlist_id PK, user_id FK, bus_id FK, timestamp)
8. Notifications (notification_id PK, user_id FK, message, timestamp)

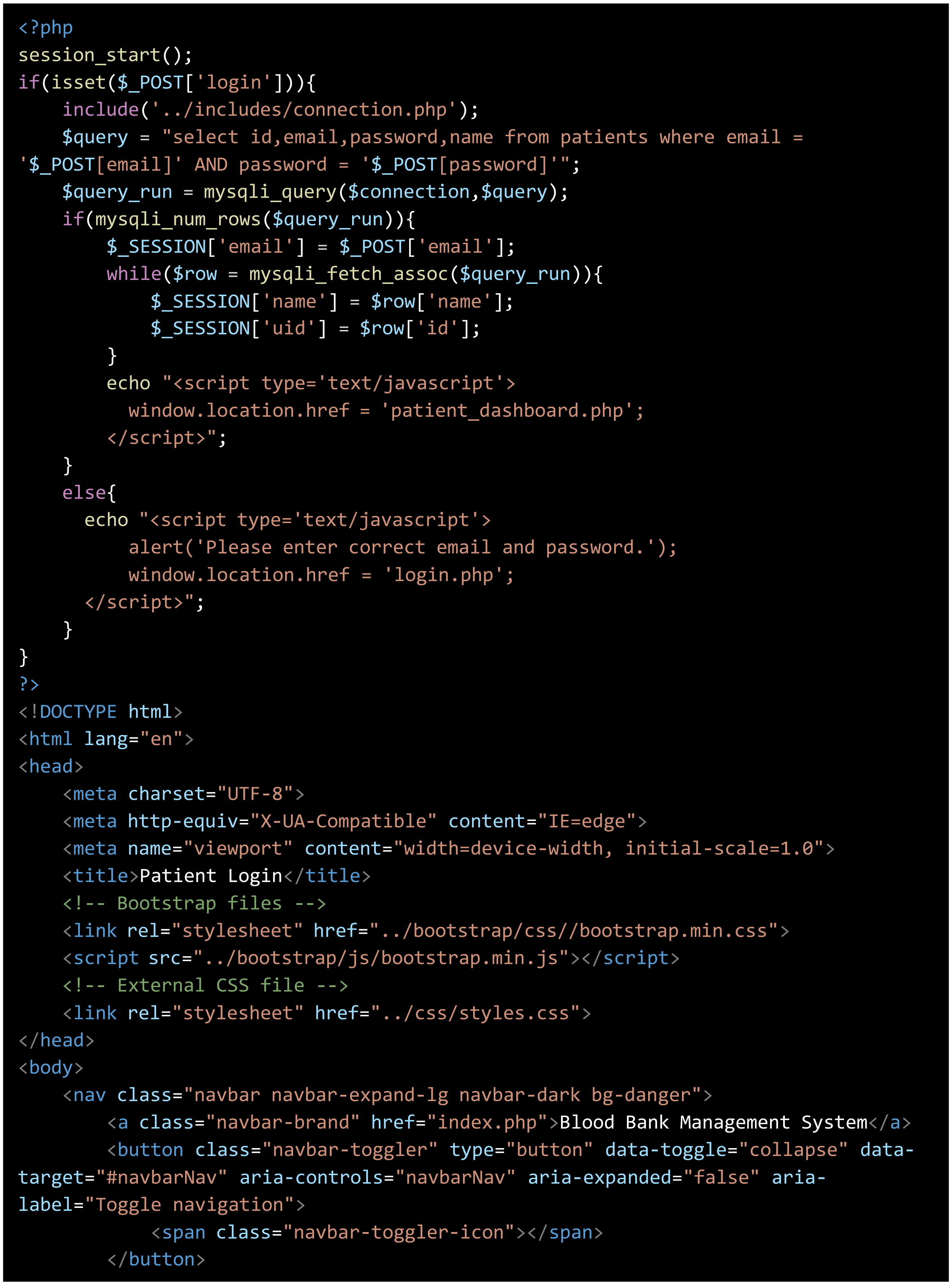
RELATION SCHEMA AFTER NORMALIZATION



CODE

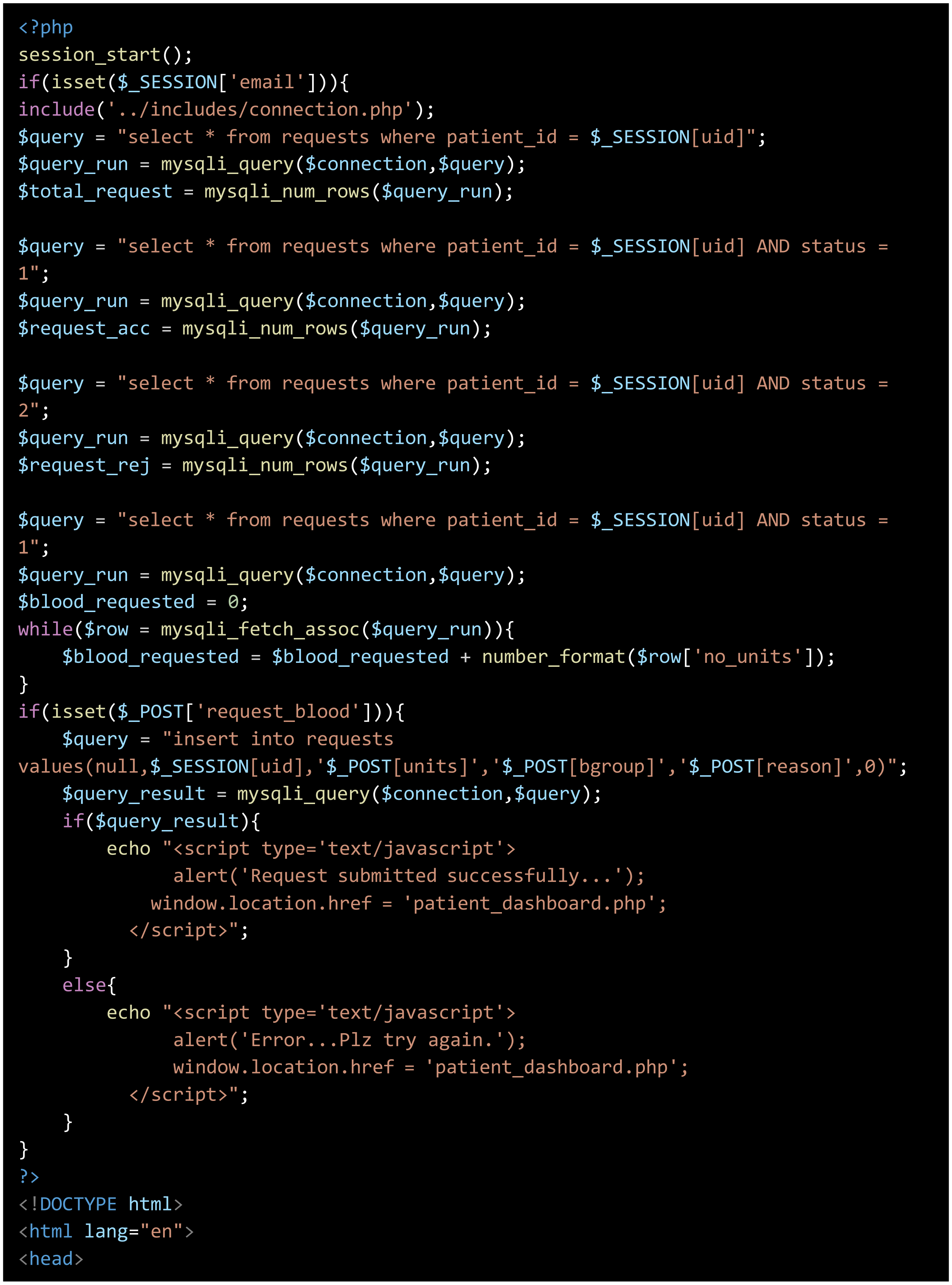
```
<?php

session_start();

if(isset($_POST['login'])){

    include('../includes/connection.php');

    $query = "select id,email,password,name from patients where email = 
'$_POST[email]' AND password = '$_POST[password]'";

    $query_run = mysqli_query($connection,$query);

    if(mysqli_num_rows($query_run)){

        $_SESSION['email'] = $_POST['email'];

        while($row = mysqli_fetch_assoc($query_run)){

            $_SESSION['name'] = $row['name'];

            $_SESSION['uid'] = $row['id'];

        }

        echo "<script type='text/javascript'>

          window.location.href = 'patient_dashboard.php';

        </script>";

    }

    else{

      echo "<script type='text/javascript'>

          alert('Please enter correct email and password.');

          window.location.href = 'login.php';

      </script>";

    }

}

?>

<!DOCTYPE html>
```

```
<html>
    <body>
        <div class="row">
            <div class="col-md-6 m-auto">
            <br><center><h4><u>List of all Donors</u></h4><br></center>
            <table class="table">
                <thead>
                    <th>S.No</th>
                    <th>Donor ID</th>
                    <th>Donor Name</th>
                    <th>Donor Email</th>
                    <th>Mobile No</th>
                    <th>Action</th>
                </thead>
                <?php
                    session_start();
                    include('../includes/connection.php');
                    $query = "select * from donors";
                    $query_run = mysqli_query($connection,$query);
                    $sno = 1;
                    while($row = mysqli_fetch_assoc($query_run)){
                        ?>
                        <tr>
                            <td><?php echo $sno; ?></td>
                            <td><?php echo $row['id']; ?></td>
                            <td><?php echo $row['name']; ?></td>
                            <td><?php echo $row['email']; ?></td>
                            <td><?php echo $row['mobile']; ?></td>
                            <td><a class="btn btn-sm btn-success"
href="edit_donor.php?did=<?php echo $row['id']; ?>">Edit</a> <a class="btn btn-sm
btn-danger" href="delete_donor.php?did=<?php echo $row['id']; ?>">Delete</a></td>
                        </tr>
                        <?php
                        $sno++;
                    }
                ?>
            </table>
            </div>
        </div>
    </body>
</html>
<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Patient Login</title>
```



```
<?php
session_start();
if(isset($_POST['login'])){
    include('../includes/connection.php');
    $query = "select id,email,password,name from patients where email =
'$_POST[email]' AND password = '$_POST[password]'";
    $query_run = mysqli_query($connection,$query);
    if(mysqli_num_rows($query_run)){
        $_SESSION['email'] = $_POST['email'];
        while($row = mysqli_fetch_assoc($query_run)){
            $_SESSION['name'] = $row['name'];
            $_SESSION['uid'] = $row['id'];
        }
        echo "<script type='text/javascript'>
          window.location.href = 'patient_dashboard.php';
        </script>";
    }
    else{
      echo "<script type='text/javascript'>
          alert('Please enter correct email and password.');
          window.location.href = 'login.php';
      </script>";
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Patient Login</title>
    <!-- Bootstrap files -->
    <link rel="stylesheet" href="../bootstrap/css//bootstrap.min.css">
    <script src="../bootstrap/js/bootstrap.min.js"></script>
    <!-- External CSS file -->
    <link rel="stylesheet" href="../css/styles.css">
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-danger">
        <a class="navbar-brand" href="index.php">Blood Bank Management System</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
```

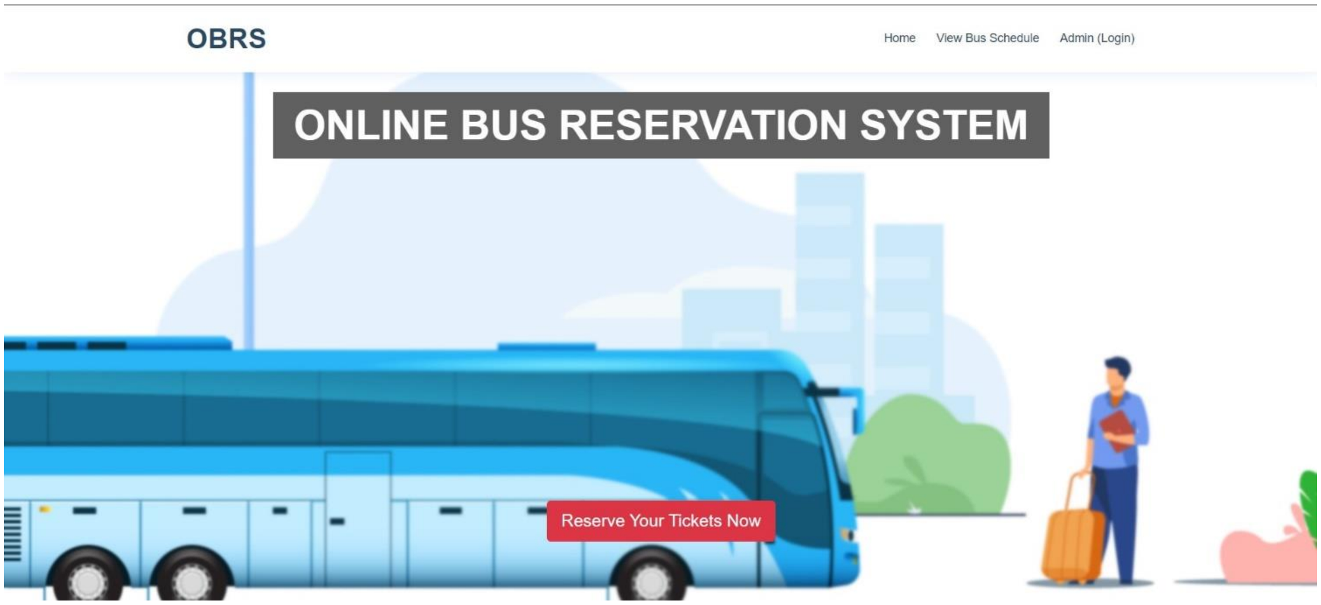
```php
<?php
session_start();
if(isset($_SESSION['email'])){
include('../includes/connection.php');
$query = "select * from requests where patient_id = $_SESSION[uid]";
$query_run = mysqli_query($connection,$query);
$total_request = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
1";
$query_run = mysqli_query($connection,$query);
$request_acc = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
2";
$query_run = mysqli_query($connection,$query);
$request_rej = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
1";
$query_run = mysqli_query($connection,$query);
$blood_requested = 0;
while($row = mysqli_fetch_assoc($query_run)){
    $blood_requested = $blood_requested + number_format($row['no_units']);
}
if(isset($_POST['request_blood'])){
    $query = "insert into requests
values(null,$_SESSION[uid],'$_POST[units]','$_POST[bgroup]','$_POST[reason]',0)";
    $query_result = mysqli_query($connection,$query);
    if($query_result){
        echo "<script type='text/javascript'>
              alert('Request submitted successfully...');
            window.location.href = 'patient_dashboard.php';
          </script>";
    }
    else{
        echo "<script type='text/javascript'>
              alert('Error...Plz try again.');
              window.location.href = 'patient_dashboard.php';
          </script>";
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
```



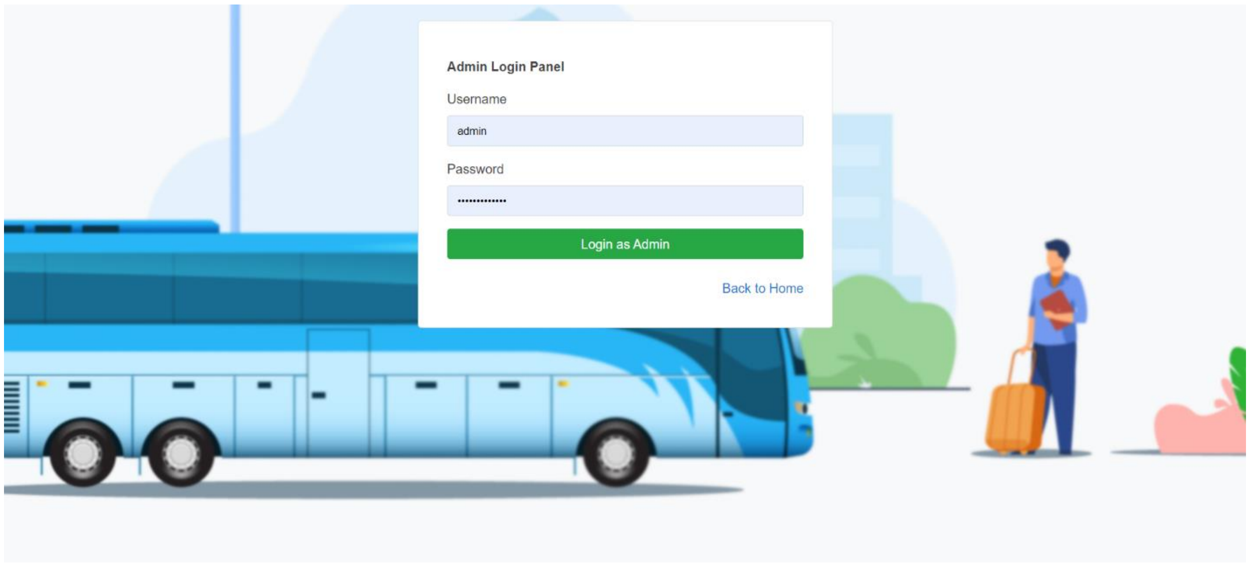
```
<?php
session_start();
if(isset($_SESSION['email'])){
include('../includes/connection.php');
$query = "select * from requests where patient_id = $_SESSION[uid]";
$query_run = mysqli_query($connection,$query);
$total_request = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
1";
$query_run = mysqli_query($connection,$query);
$request_acc = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
2";
$query_run = mysqli_query($connection,$query);
$request_rej = mysqli_num_rows($query_run);

$query = "select * from requests where patient_id = $_SESSION[uid] AND status =
1";
$query_run = mysqli_query($connection,$query);
$blood_requested = 0;
while($row = mysqli_fetch_assoc($query_run)){
    $blood_requested = $blood_requested + number_format($row['no_units']);
}
if(isset($_POST['request_blood'])){
    $query = "insert into requests
values(null,$_SESSION[uid],'$_POST[units]','$_POST[bgroup]','$_POST[reason]',0)";
    $query_result = mysqli_query($connection,$query);
    if($query_result){
        echo "<script type='text/javascript'>
              alert('Request submitted successfully...');
            window.location.href = 'patient_dashboard.php';
          </script>";
    }
    else{
        echo "<script type='text/javascript'>
              alert('Error...Plz try again.');
              window.location.href = 'patient_dashboard.php';
          </script>";
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
```

Screenshots

Home Page



Admin Login Page


Admin Login Panel

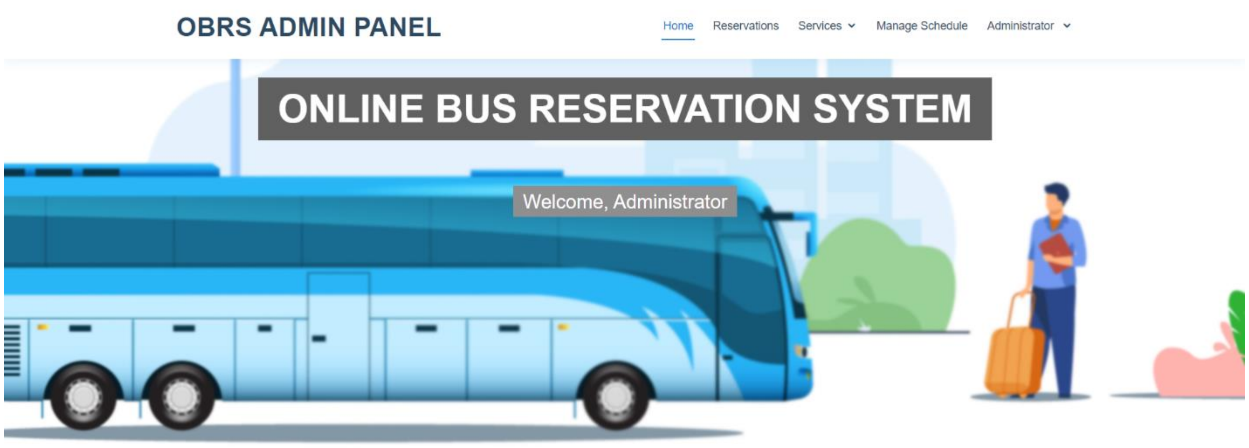
Username

admin

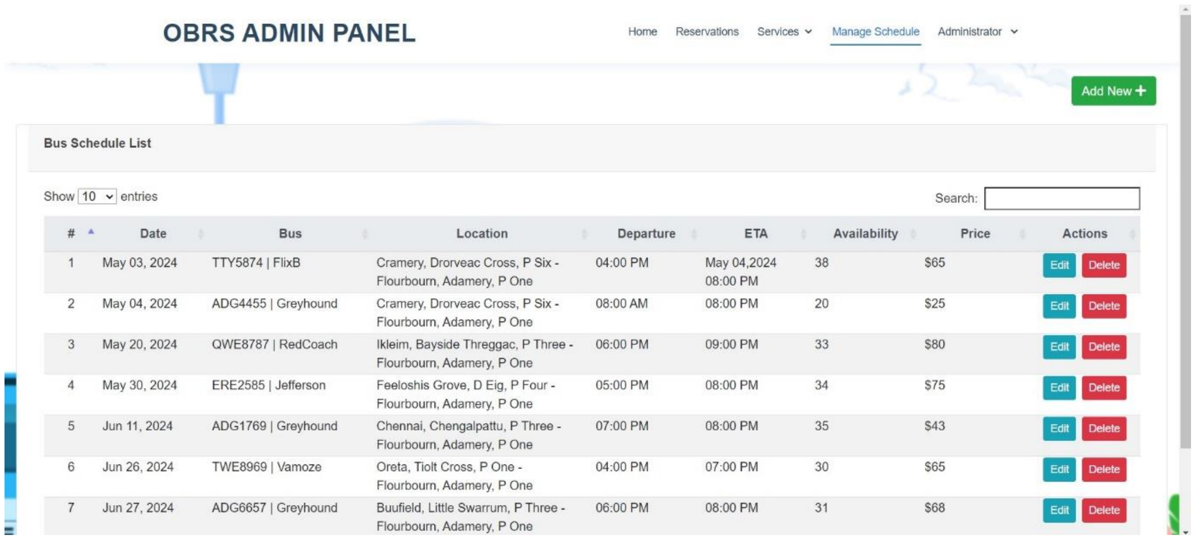
Password

••••••••••••

Admin Dashboard Page



Schedule



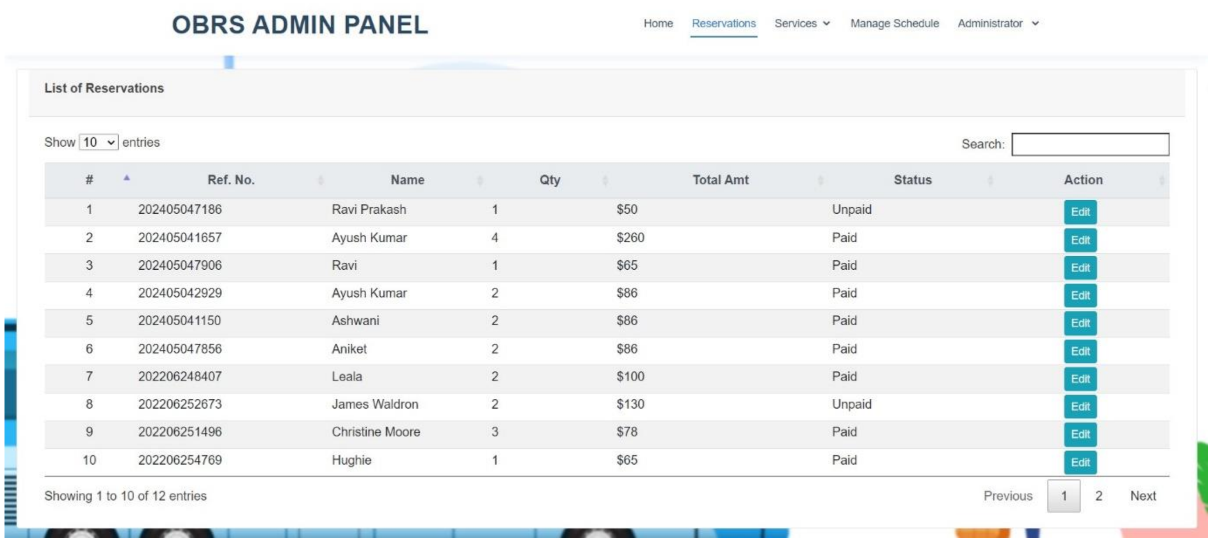
Bus Schedule List

Show 10 entries

Search:

#	Date	Bus	Location	Departure	ETA	Availability	Price	Actions
1	May 03, 2024	TTY5874 | FlixB	Cramery, Drorveac Cross, P Six - Flourbourn, Adamery, P One	04:00 PM	May 04,2024 08:00 PM	38	$65	Edit Delete
2	May 04, 2024	ADG4455 | Greyhound	Cramery, Drorveac Cross, P Six - Flourbourn, Adamery, P One	08:00 AM	08:00 PM	20	$25	Edit Delete
3	May 20, 2024	QWE8787 | RedCoach	Ikleim, Bayside Threggac, P Three - Flourbourn, Adamery, P One	06:00 PM	09:00 PM	33	$80	Edit Delete
4	May 30, 2024	ERE2585 | Jefferson	Feeloshis Grove, D Eig, P Four - Flourbourn, Adamery, P One	05:00 PM	08:00 PM	34	$75	Edit Delete
5	Jun 11, 2024	ADG1769 | Greyhound	Chennai, Chengalpattu, P Three - Flourbourn, Adamery, P One	07:00 PM	08:00 PM	35	$43	Edit Delete
6	Jun 26, 2024	TWE8969 | Vamoze	Oreta, Tiolt Cross, P One - Flourbourn, Adamery, P One	04:00 PM	07:00 PM	30	$65	Edit Delete
7	Jun 27, 2024	ADG6657 | Greyhound	Buufield, Little Swarrum, P Three - Flourbourn, Adamery, P One	06:00 PM	08:00 PM	31	$68	Edit Delete

Reservation



List of Reservations

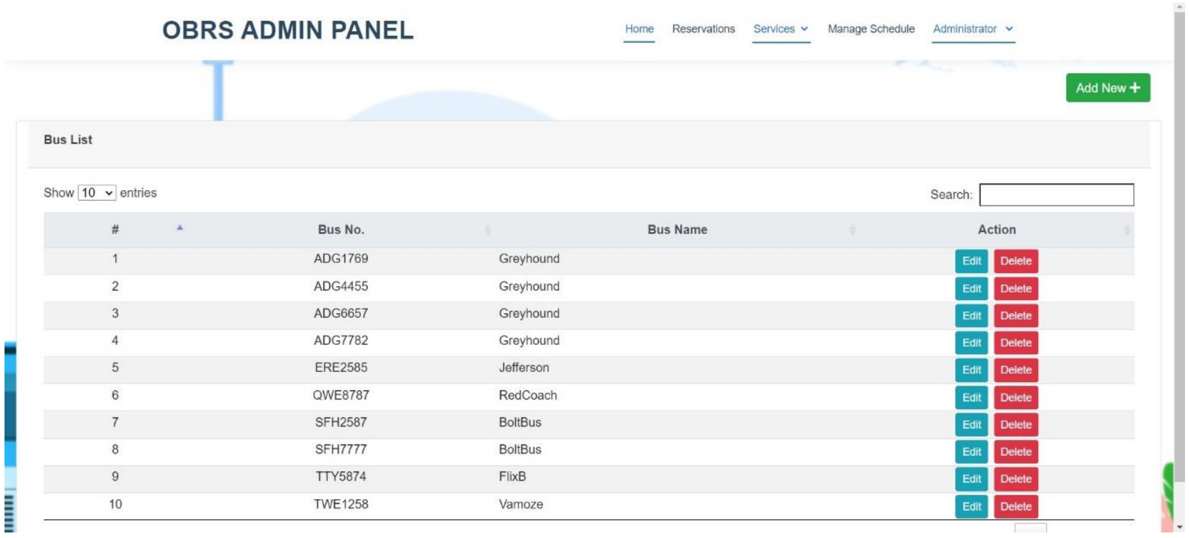
Show 10 entries Search:

#	Ref. No.	Name	Qty	Total Amt	Status	Action
1	202405047186	Ravi Prakash	1	$50	Unpaid	Edit
2	202405041657	Ayush Kumar	4	$260	Paid	Edit
3	202405047906	Ravi	1	$65	Paid	Edit
4	202405042929	Ayush Kumar	2	$86	Paid	Edit
5	202405041150	Ashwani	2	$86	Paid	Edit
6	202405047856	Aniket	2	$86	Paid	Edit
7	202206248407	Leala	2	$100	Paid	Edit
8	202206252673	James Waldron	2	$130	Unpaid	Edit
9	202206251496	Christine Moore	3	$78	Paid	Edit
10	202206254769	Hughie	1	$65	Paid	Edit

Showing 1 to 10 of 12 entries

Previous 1 2 Next

List of Bus



Bus List

Show 10 entries Search:

#	Bus No.	Bus Name	Action
1	ADG1769	Greyhound	Edit Delete
2	ADG4455	Greyhound	Edit Delete
3	ADG6657	Greyhound	Edit Delete
4	ADG7782	Greyhound	Edit Delete
5	ERE2585	Jefferson	Edit Delete
6	QWE8787	RedCoach	Edit Delete
7	SFH2587	BoltBus	Edit Delete
8	SFH7777	BoltBus	Edit Delete
9	TTY5874	FlixB	Edit Delete
10	TWE1258	Vamoze	Edit Delete

Languages used

1. HTML
2. CSS
3. JavaScript
4. jQuery
5. PHP
6. MySQL

Software used

1. Text editor (any)
2. Web browser (any)
3. Xampp local serve

CONCLUSION

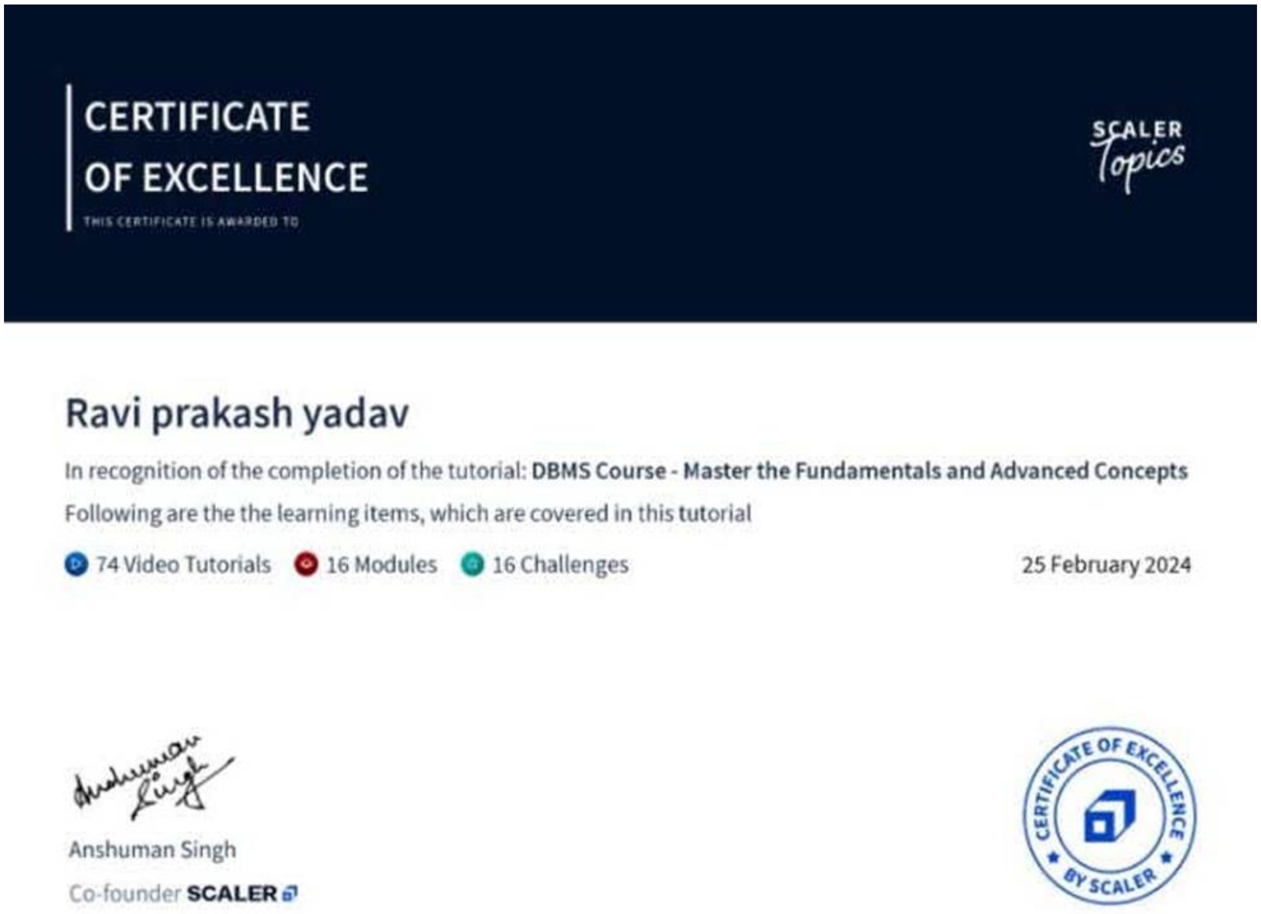
In conclusion, the development of the Online Bus Reservation System has successfully met its objectives by providing users with a convenient platform to search for buses, reserve seats, and manage bookings online. Through robust implementation of features such as user authentication, real-time seat availability updates, and an intuitive user interface, the system offers an efficient and seamless booking experience. Despite encountering challenges during development, including technical complexities and time constraints, the team's dedication and problem-solving skills led to the successful delivery of a reliable and user-friendly solution.

Looking ahead, future enhancements could focus on integrating additional functionalities such as mobile app support, payment gateways, and enhanced administrative tools. By continually refining and expanding the system's capabilities, it has the potential to further streamline bus travel and make it more accessible and enjoyable for users.

FUTURE WORK

Future work for the Online Bus Reservation System includes mobile app development for Android and iOS platforms, payment gateway integration, real-time bus tracking, feedback and rating systems, enhanced administrative tools, integration with travel agencies, data analytics, and accessibility features to improve functionality, usability, and scalability.

<u>ONLINE CERTIFICATION COURSE</u>



CERTIFICATE

OF EXCELLENCE

THIS CERTIFICATE IS AWARDED TO

SCALER Topics

Ravi prakash yadav

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

● 74 Video Tutorials ● 16 Modules ● 16 Challenges

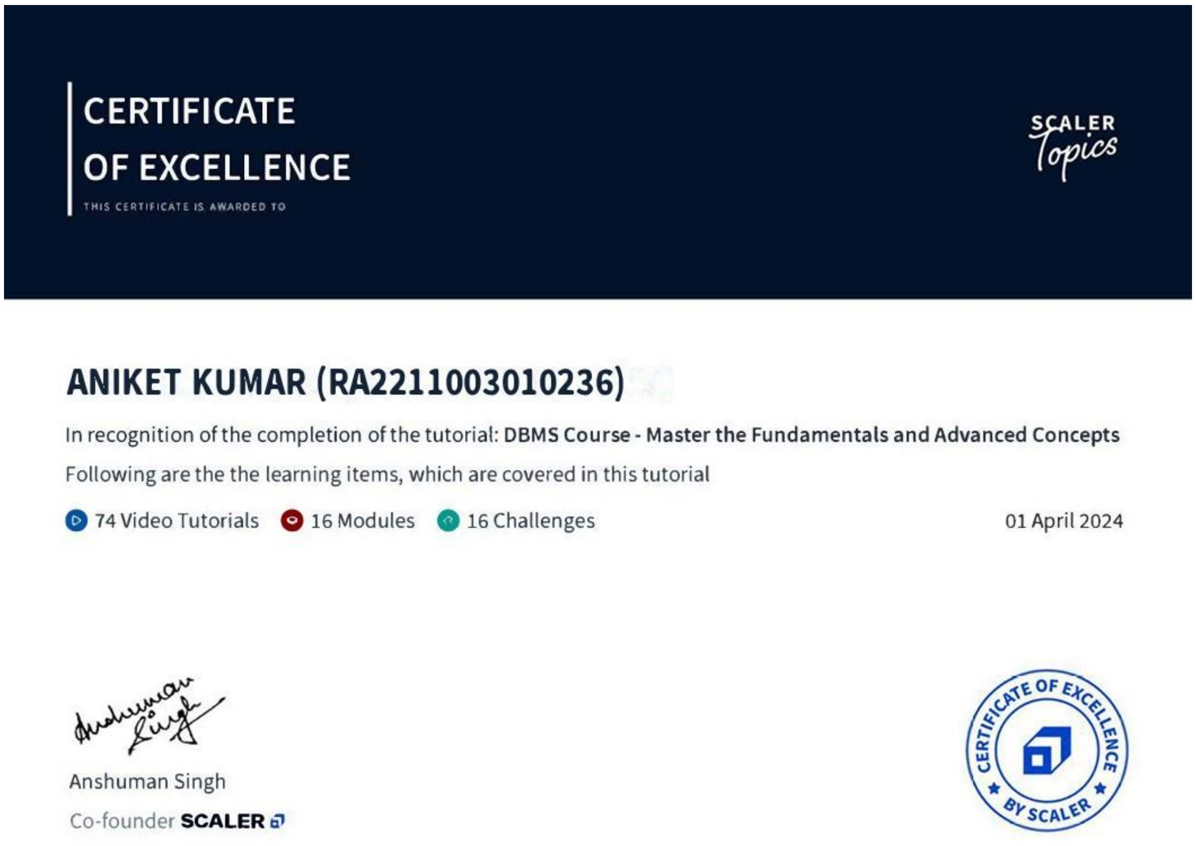
25 February 2024

[illegible]

Anshuman Singh

Co-founder **SCALER**

CERTIFICATE OF EXCELLENCE ★ BY SCALER ★



CERTIFICATE
OF EXCELLENCE
THIS CERTIFICATE IS AWARDED TO

SCALER Topics

ANIKET KUMAR (RA2211003010236)

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 01 April 2024

[illegible]

Anshuman Singh
Co-founder **SCALER**

CERTIFICATE OF EXCELLENCE ★ BY SCALER ★