RAILWAY RESERVATION

SYSTEM

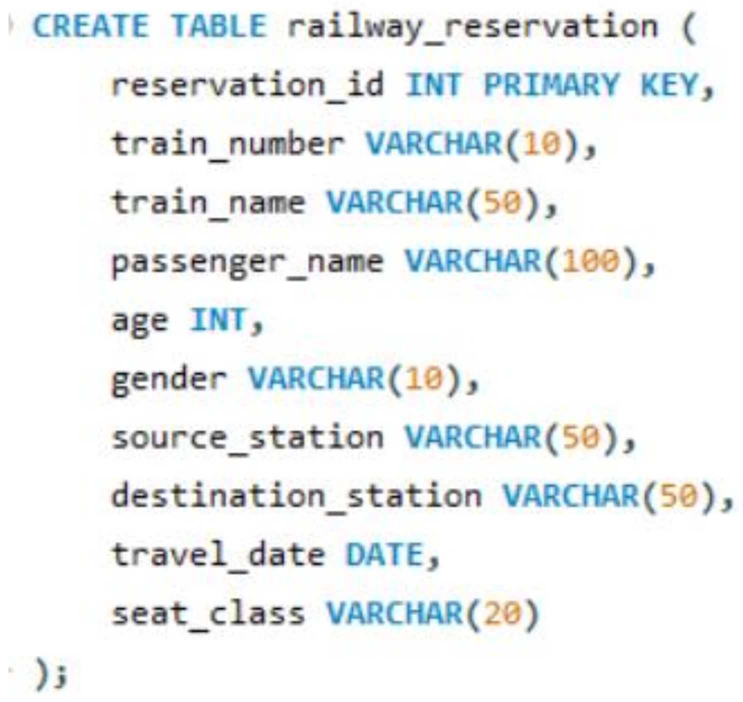
INTRODUCTION

A **Railway Reservation System** is a software platform designed to streamline the process of booking, managing, and canceling train tickets. It allows passengers to search for available trains based on their source, destination, and travel dates. The system displays seat availability and enables users to select their preferred seat class. It calculates fares based on route, distance, and class, ensuring accurate pricing. Passengers can make secure payments through integrated payment gateways. Additionally, the system manages passenger details, providing real-time information on bookings and cancellations. It offers a convenient interface for both online and mobile users, available 24/7. Railway authorities benefit from better resource management and operational efficiency. Ultimately, the system enhances user experience, reduces manual work, and improves overall train travel management.

OBJECTIVES

1. To enable efficient and hassle-free ticket booking for passengers.
2. To provide real-time updates on train schedules and seat availability.
3. To minimize manual errors and delays in the reservation process.
4. To offer a user-friendly interface for easy navigation and booking.
5. To ensure secure and convenient online payment options.
6. To improve customer service through streamlined cancellations and refunds.
7. To maintain a reliable system capable of handling high user traffic.
8. To generate detailed reports for better operational management.
9. To make the system accessible through multiple platforms and devices.

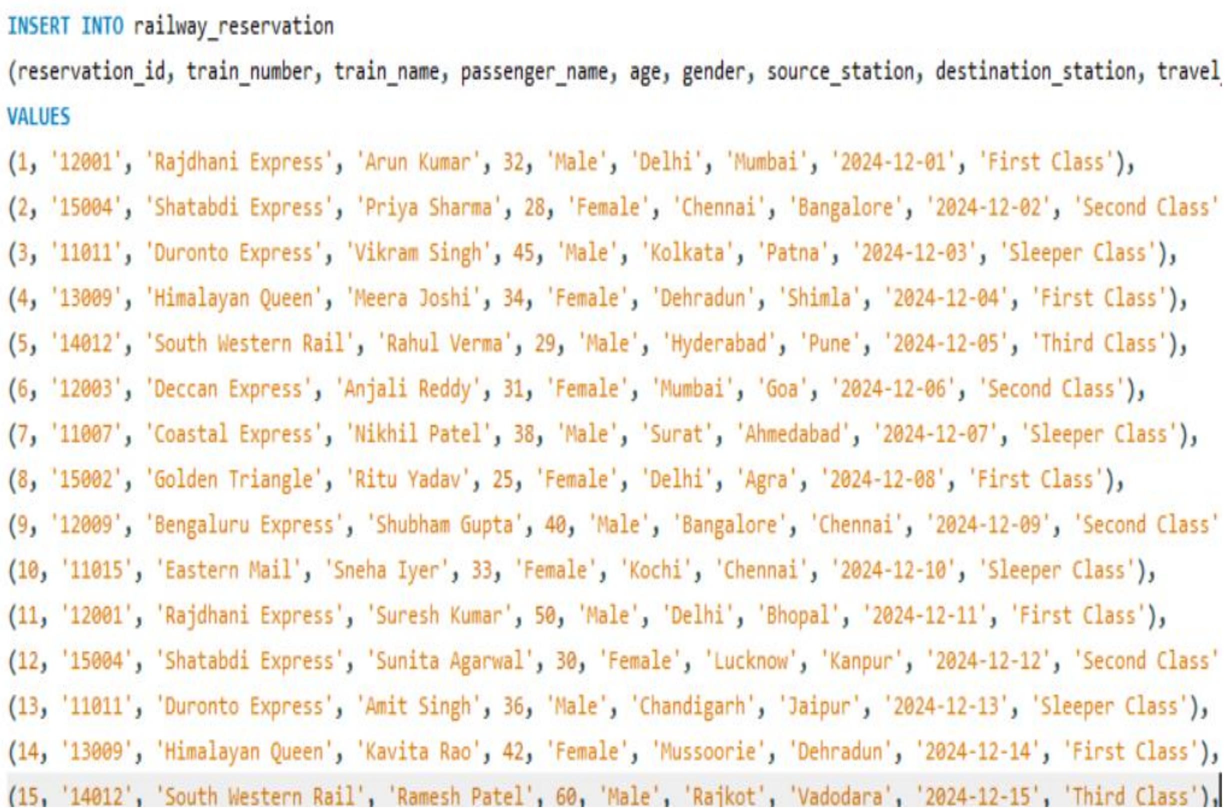
CREATION OF TABLE



```
CREATE TABLE railway_reservation (
    reservation_id INT PRIMARY KEY,
    train_number VARCHAR(10),
    train_name VARCHAR(50),
    passenger_name VARCHAR(100),
    age INT,
    gender VARCHAR(10),
    source_station VARCHAR(50),
    destination_station VARCHAR(50),
    travel_date DATE,
    seat_class VARCHAR(20)
);
```

The creation of a table involves defining its structure to store data in an organized manner. A table is a collection of rows and columns, where rows represent individual records, and columns define the attributes or fields of the data. While creating a table in a database, you specify the table name, column names, data types, and constraints like primary keys or not null. This ensures data integrity and consistency.

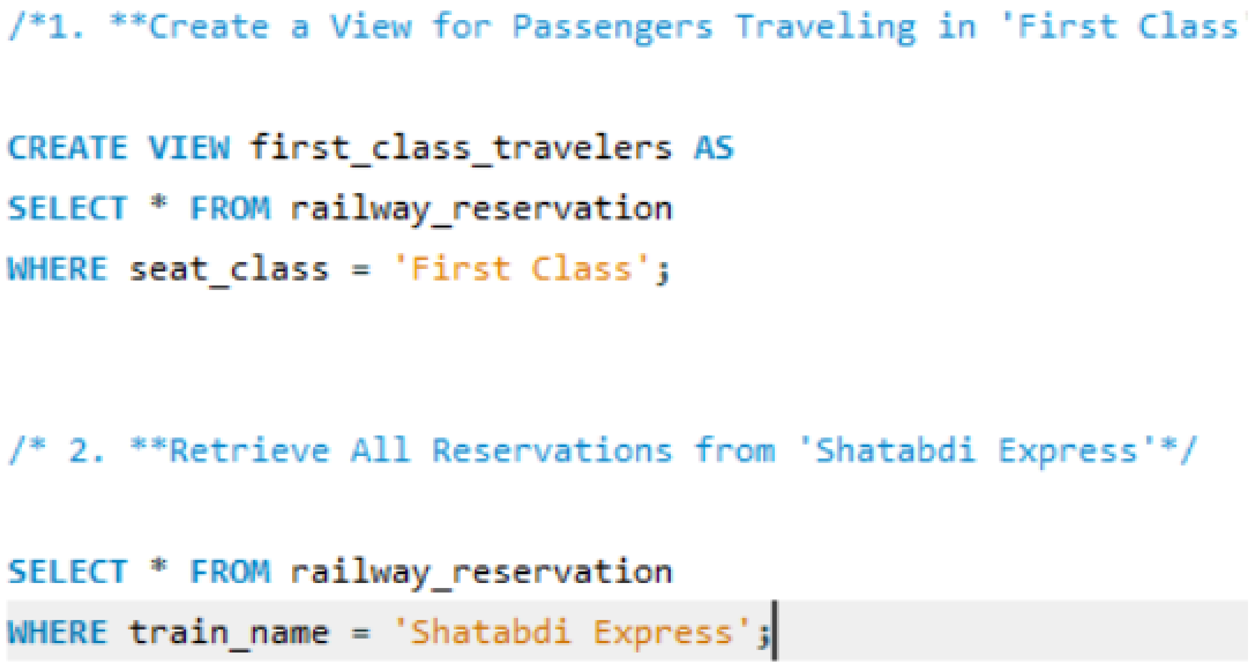
INSERTION OF VALUES



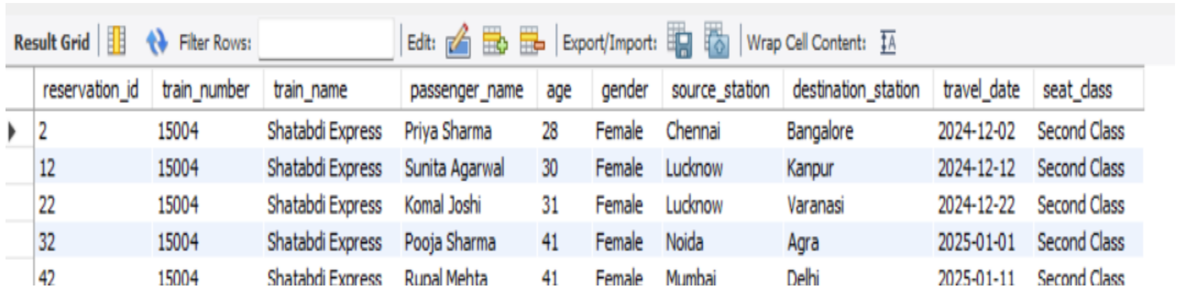
```
INSERT INTO railway_reservation
(reservation_id, train_number, train_name, passenger_name, age, gender, source_station, destination_station, travel
VALUES
(1, '12001', 'Rajdhani Express', 'Arun Kumar', 32, 'Male', 'Delhi', 'Mumbai', '2024-12-01', 'First Class'),
(2, '15004', 'Shatabdi Express', 'Priya Sharma', 28, 'Female', 'Chennai', 'Bangalore', '2024-12-02', 'Second Class'
(3, '11011', 'Duronto Express', 'Vikram Singh', 45, 'Male', 'Kolkata', 'Patna', '2024-12-03', 'Sleeper Class'),
(4, '13009', 'Himalayan Queen', 'Meera Joshi', 34, 'Female', 'Dehradun', 'Shimla', '2024-12-04', 'First Class'),
(5, '14012', 'South Western Rail', 'Rahul Verma', 29, 'Male', 'Hyderabad', 'Pune', '2024-12-05', 'Third Class'),
(6, '12003', 'Deccan Express', 'Anjali Reddy', 31, 'Female', 'Mumbai', 'Goa', '2024-12-06', 'Second Class'),
(7, '11007', 'Coastal Express', 'Nikhil Patel', 38, 'Male', 'Surat', 'Ahmedabad', '2024-12-07', 'Sleeper Class'),
(8, '15002', 'Golden Triangle', 'Ritu Yadav', 25, 'Female', 'Delhi', 'Agra', '2024-12-08', 'First Class'),
(9, '12009', 'Bengaluru Express', 'Shubham Gupta', 40, 'Male', 'Bangalore', 'Chennai', '2024-12-09', 'Second Class'
(10, '11015', 'Eastern Mail', 'Sneha Iyer', 33, 'Female', 'Kochi', 'Chennai', '2024-12-10', 'Sleeper Class'),
(11, '12001', 'Rajdhani Express', 'Suresh Kumar', 50, 'Male', 'Delhi', 'Bhopal', '2024-12-11', 'First Class'),
(12, '15004', 'Shatabdi Express', 'Sunita Agarwal', 30, 'Female', 'Lucknow', 'Kanpur', '2024-12-12', 'Second Class'
(13, '11011', 'Duronto Express', 'Amit Singh', 36, 'Male', 'Chandigarh', 'Jaipur', '2024-12-13', 'Sleeper Class'),
(14, '13009', 'Himalayan Queen', 'Kavita Rao', 42, 'Female', 'Mussoorie', 'Dehradun', '2024-12-14', 'First Class'),
(15, '14012', 'South Western Rail', 'Ramesh Patel', 60, 'Male', 'Rajkot', 'Vadodara', '2024-12-15', 'Third Class'),
```

The insertion of values involves adding data into a database table. SQL command, where the table name and column names are specified, followed by the values to be added. Each row represents a single record, and the data provided must match the structure and constraints of the table. Proper insertion ensures the database remains consistent and organized. This process is essential for populating a database with meaningful information.

RETRIEVE ALL RESERVATION FROM SHATABDI EXPRESS



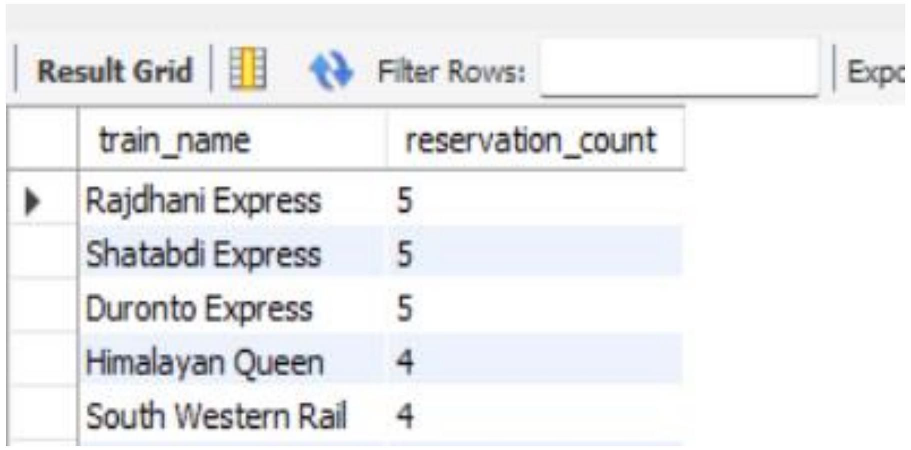
```
/*1. **Create a View for Passengers Traveling in 'First Class

CREATE VIEW first_class_travelers AS
SELECT * FROM railway_reservation
WHERE seat_class = 'First Class';



/* 2. **Retrieve All Reservations from 'Shatabdi Express'*/

SELECT * FROM railway_reservation
WHERE train_name = 'Shatabdi Express';
```



reservation_id	train_number	train_name	passenger_name	age	gender	source_station	destination_station	travel_date	seat_class
2	15004	Shatabdi Express	Priya Sharma	28	Female	Chennai	Bangalore	2024-12-02	Second Class
12	15004	Shatabdi Express	Sunita Agarwal	30	Female	Lucknow	Kanpur	2024-12-12	Second Class
22	15004	Shatabdi Express	Komal Joshi	31	Female	Lucknow	Varanasi	2024-12-22	Second Class
32	15004	Shatabdi Express	Pooja Sharma	41	Female	Noida	Agra	2025-01-01	Second Class
42	15004	Shatabdi Express	Rupal Mehta	41	Female	Mumbai	Delhi	2025-01-11	Second Class

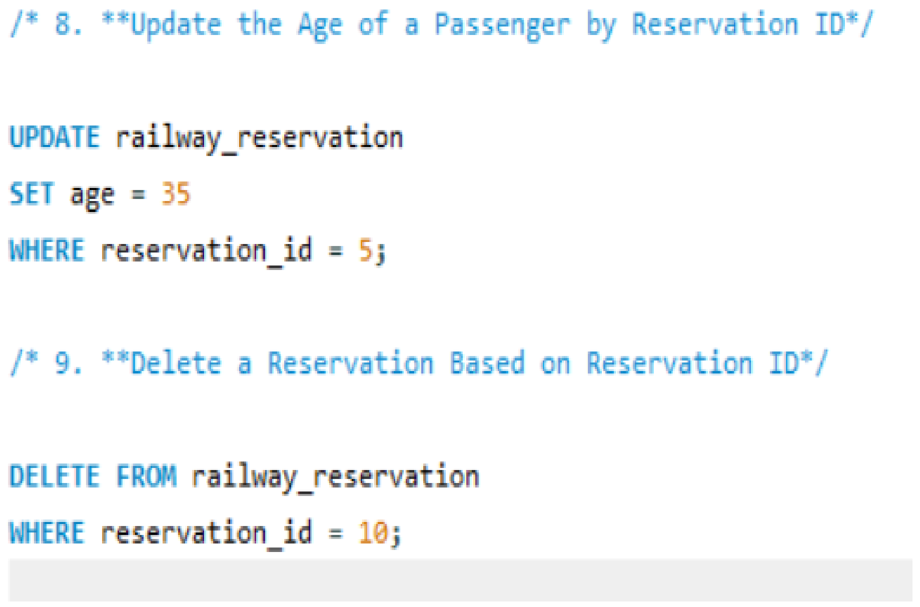
COUNT THE TOTAL NO OF RESERVATION FOR TRAIN

```
/* 3. **Count the Total Number of Reservations for Each Train*/

SELECT train_name, COUNT(*) AS reservation_count
FROM railway_reservation
GROUP BY train_name;


/* 4. **Find the Maximum Age of Passengers in the Reservation System*/

SELECT MAX(age) AS max_age FROM railway_reservation;
```


Result Grid | Filter Rows: | Expo

train_name	reservation_count
Rajdhani Express	5
Shatabdi Express	5
Duronto Express	5
Himalayan Queen	4
South Western Rail	4

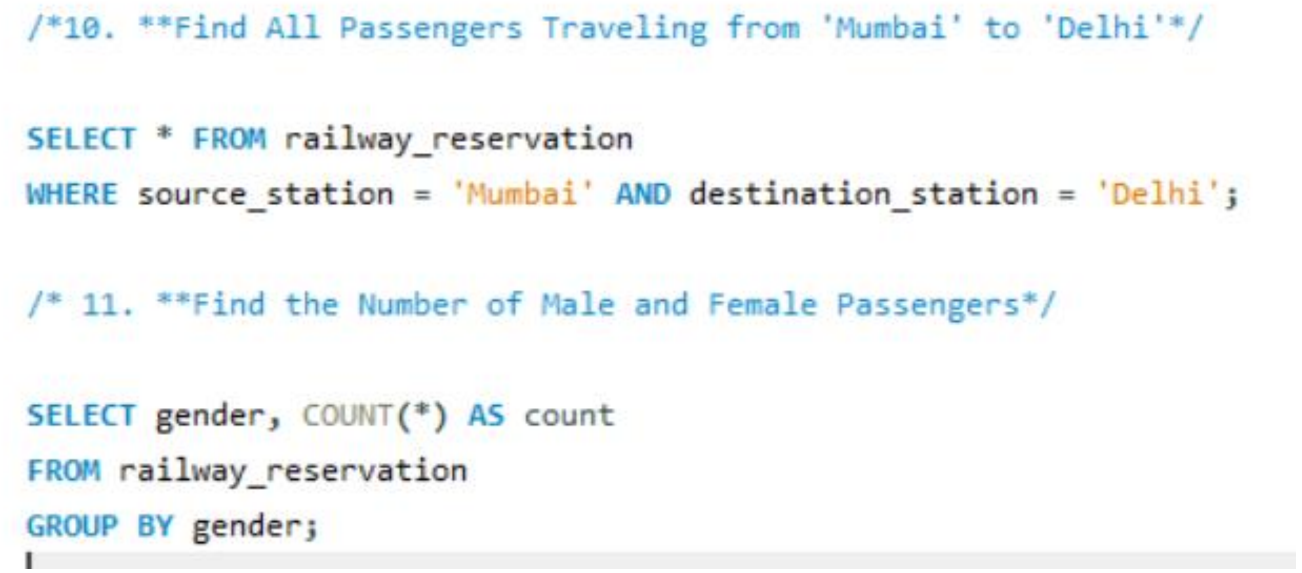
DELETE OPERATOR



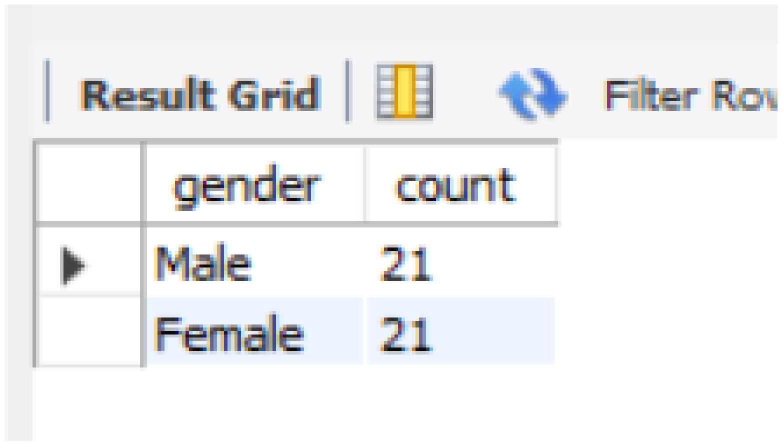
```
/* 8. **Update the Age of a Passenger by Reservation ID*/

UPDATE railway_reservation
SET age = 35
WHERE reservation_id = 5;

/* 9. **Delete a Reservation Based on Reservation ID*/

DELETE FROM railway_reservation
WHERE reservation_id = 10;
```

The SQL query deletes a record from the railway reservation table where the reservation id is equal to 10. The DELETE FROM command removes specific rows based on the condition provided in the WHERE clause. This ensures only the intended record is deleted without affecting other data.

FIND THE NUMBER OF MALE AND FEMALE PASSENGERS

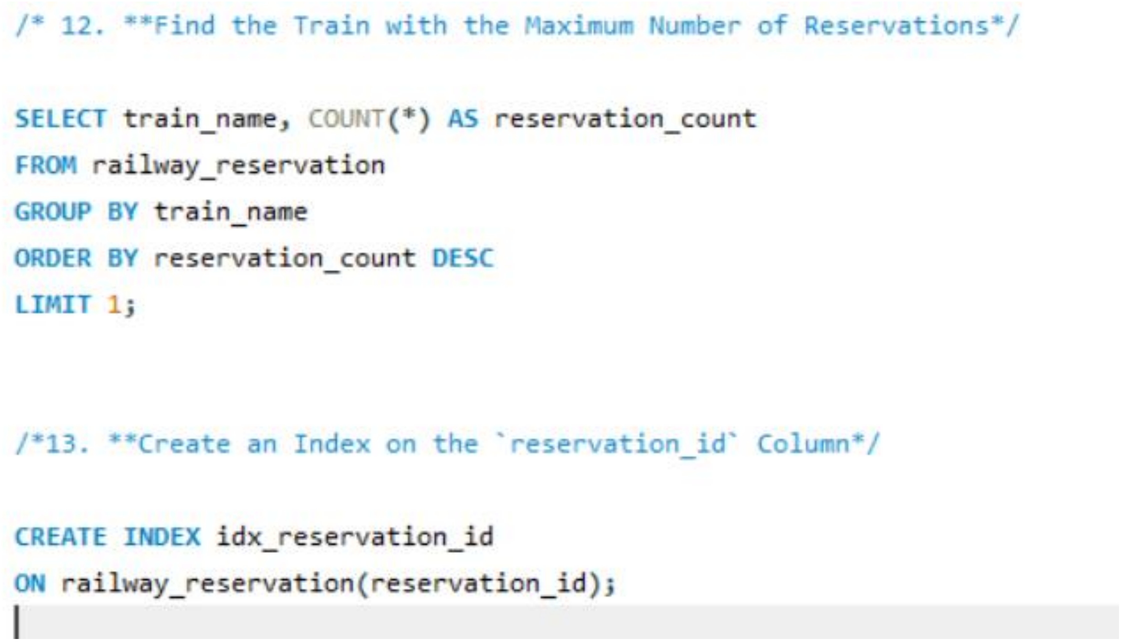


```sql
/*10. **Find All Passengers Traveling from 'Mumbai' to 'Delhi'*/

SELECT * FROM railway_reservation
WHERE source_station = 'Mumbai' AND destination_station = 'Delhi';

/* 11. **Find the Number of Male and Female Passengers*/

SELECT gender, COUNT(*) AS count
FROM railway_reservation
GROUP BY gender;
```

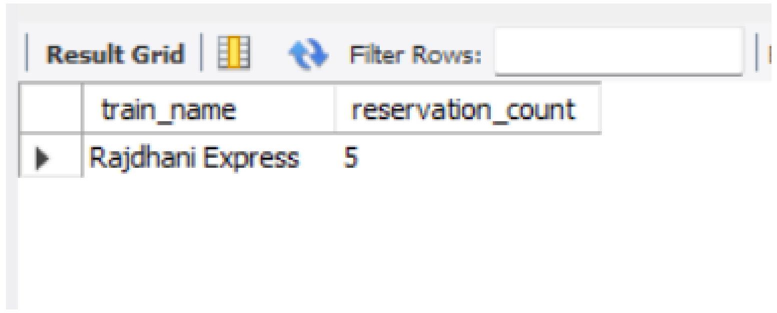


gender	count
Male	21
Female	21

FIND THE TRAIN WITH MAXIMUM NUMBER OF RESVERVATION

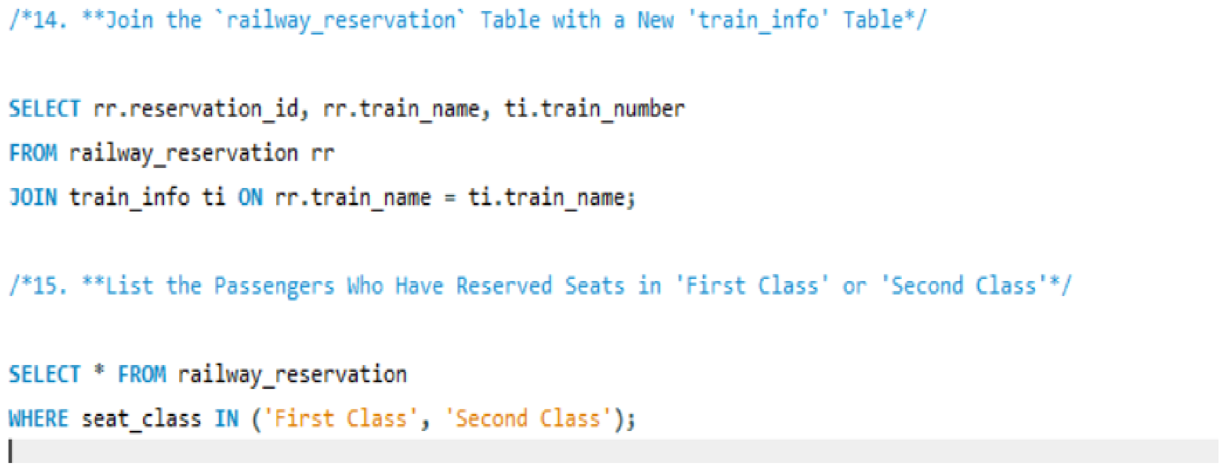


```
/* 12. **Find the Train with the Maximum Number of Reservations*/

SELECT train_name, COUNT(*) AS reservation_count
FROM railway_reservation
GROUP BY train_name
ORDER BY reservation_count DESC
LIMIT 1;


/*13. **Create an Index on the `reservation_id` Column*/

CREATE INDEX idx_reservation_id
ON railway_reservation(reservation_id);
```

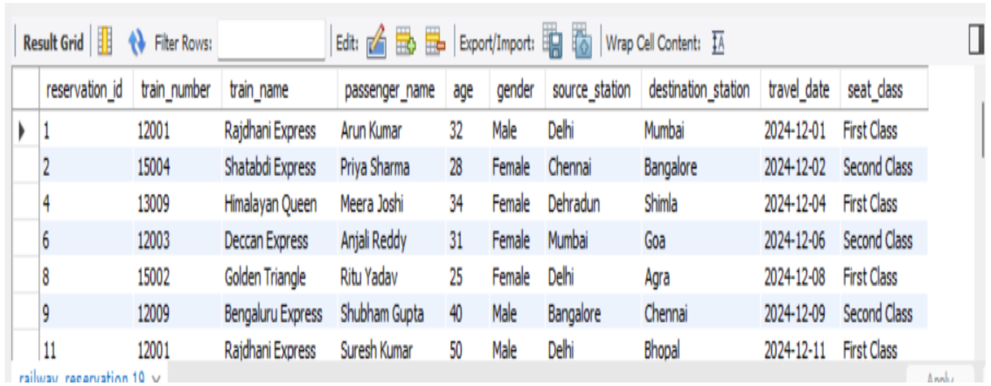


	train_name	reservation_count
▶	Rajdhani Express	5

LIST THE PASSENGERS WHO HAVE RESERVED SEATS IN FIRST CLASS OR SECOND CLASS

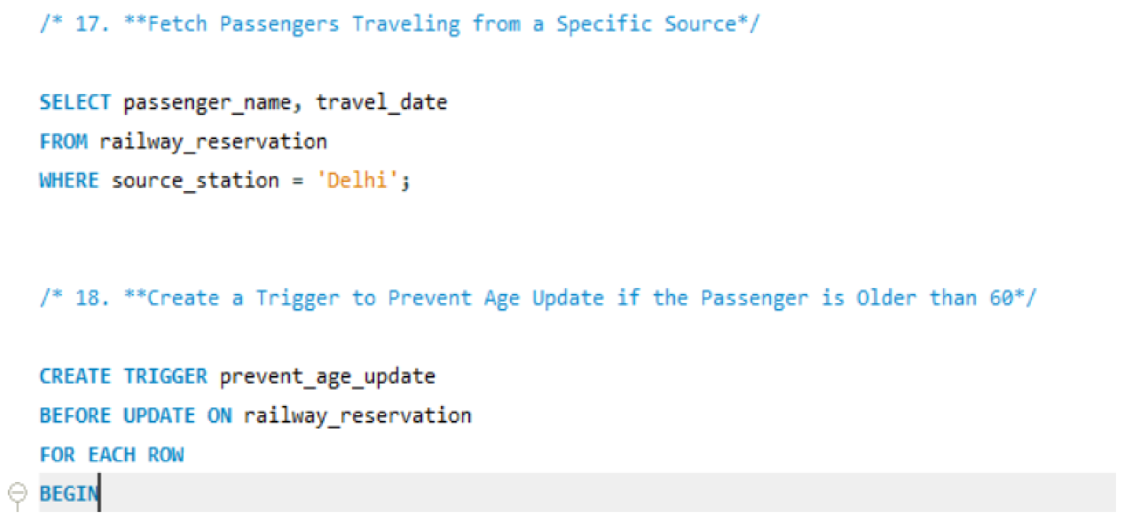


```
/*14. **Join the `railway_reservation` Table with a New 'train_info' Table*/

SELECT rr.reservation_id, rr.train_name, ti.train_number
FROM railway_reservation rr
JOIN train_info ti ON rr.train_name = ti.train_name;

/*15. **List the Passengers Who Have Reserved Seats in 'First Class' or 'Second Class'*/

SELECT * FROM railway_reservation
WHERE seat_class IN ('First Class', 'Second Class');
```

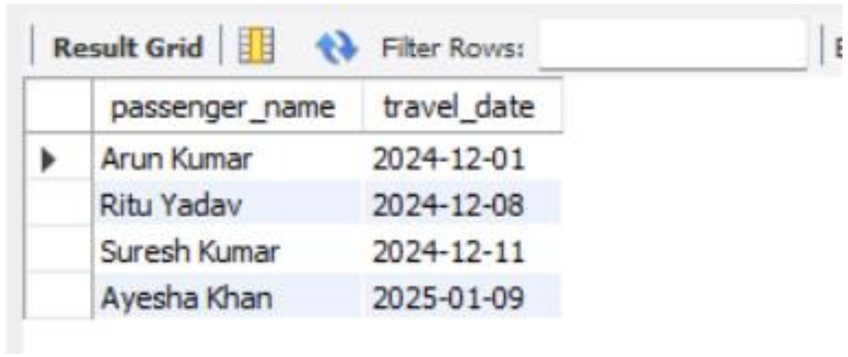


reservation_id	train_number	train_name	passenger_name	age	gender	source_station	destination_station	travel_date	seat_class
1	12001	Rajdhani Express	Arun Kumar	32	Male	Delhi	Mumbai	2024-12-01	First Class
2	15004	Shatabdi Express	Priya Sharma	28	Female	Chennai	Bangalore	2024-12-02	Second Class
4	13009	Himalayan Queen	Meera Joshi	34	Female	Dehradun	Shimla	2024-12-04	First Class
6	12003	Deccan Express	Anjali Reddy	31	Female	Mumbai	Goa	2024-12-06	Second Class
8	15002	Golden Triangle	Ritu Yadav	25	Female	Delhi	Agra	2024-12-08	First Class
9	12009	Bengaluru Express	Shubham Gupta	40	Male	Bangalore	Chennai	2024-12-09	Second Class
11	12001	Rajdhani Express	Suresh Kumar	50	Male	Delhi	Bhopal	2024-12-11	First Class

FETCH PASSENGERS TRAVELLING FROM A SPECIFIC SORUCE

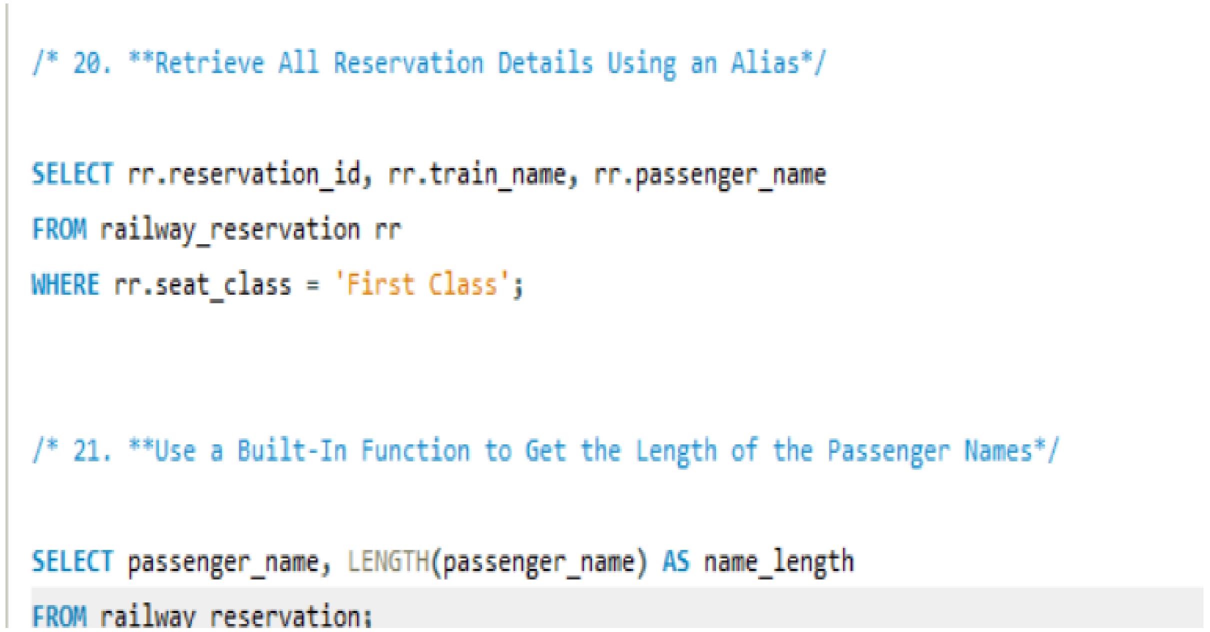


```
/* 17. **Fetch Passengers Traveling from a Specific Source*/

SELECT passenger_name, travel_date
FROM railway_reservation
WHERE source_station = 'Delhi';


/* 18. **Create a Trigger to Prevent Age Update if the Passenger is Older than 60*/

CREATE TRIGGER prevent_age_update
BEFORE UPDATE ON railway_reservation
FOR EACH ROW
BEGIN
```

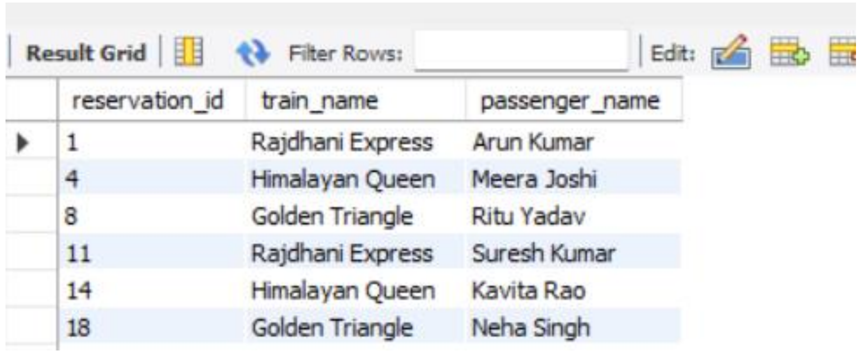


passenger_name	travel_date
Arun Kumar	2024-12-01
Ritu Yadav	2024-12-08
Suresh Kumar	2024-12-11
Ayesha Khan	2025-01-09

RETRIEVE ALL RESERVATION DETAILS USING AN ALIAS

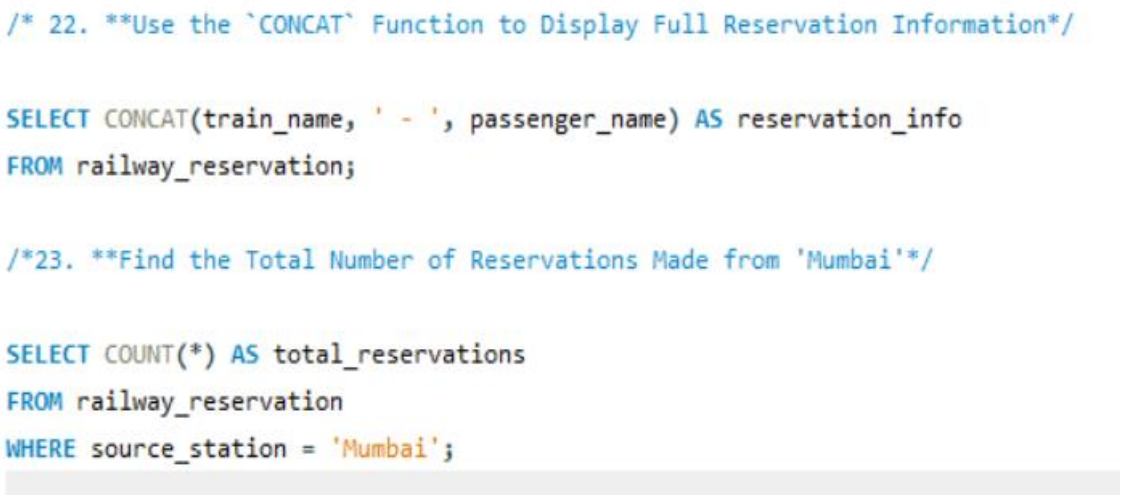


```
/* 20. **Retrieve All Reservation Details Using an Alias*/

SELECT rr.reservation_id, rr.train_name, rr.passenger_name
FROM railway_reservation rr
WHERE rr.seat_class = 'First Class';



/* 21. **Use a Built-In Function to Get the Length of the Passenger Names*/

SELECT passenger_name, LENGTH(passenger_name) AS name_length
FROM railway_reservation;
```

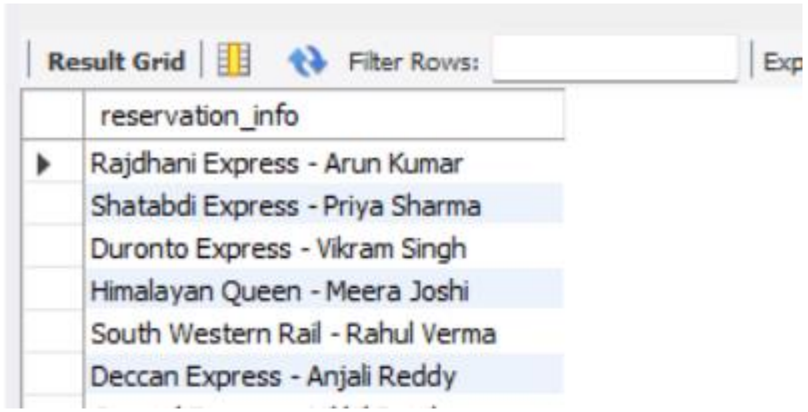


reservation_id	train_name	passenger_name
1	Rajdhani Express	Arun Kumar
4	Himalayan Queen	Meera Joshi
8	Golden Triangle	Ritu Yadav
11	Rajdhani Express	Suresh Kumar
14	Himalayan Queen	Kavita Rao
18	Golden Triangle	Neha Singh

USE THE CONCAT FUNCTION TO DISPLAY FULL RESERVATION INFORMATION

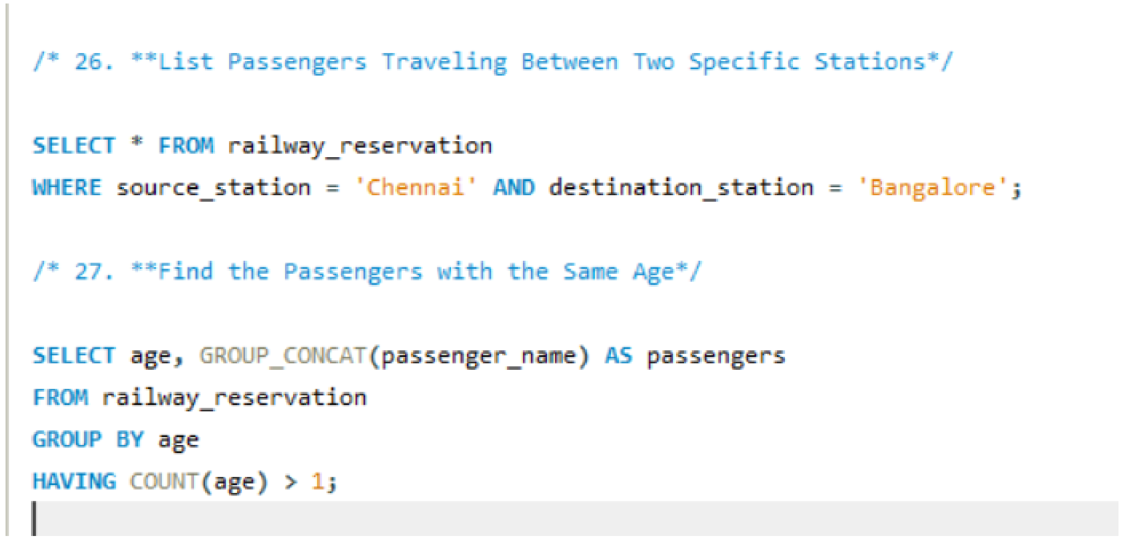


```
/* 22. **Use the `CONCAT` Function to Display Full Reservation Information*/

SELECT CONCAT(train_name, ' - ', passenger_name) AS reservation_info
FROM railway_reservation;

/*23. **Find the Total Number of Reservations Made from 'Mumbai'*/

SELECT COUNT(*) AS total_reservations
FROM railway_reservation
WHERE source_station = 'Mumbai';
```

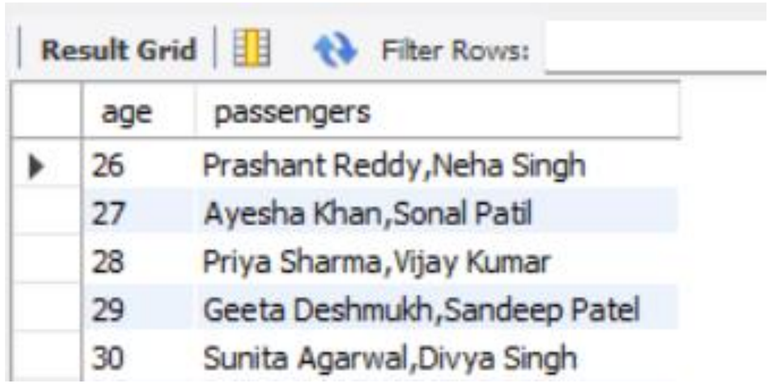


```
reservation_info
Rajdhani Express - Arun Kumar
Shatabdi Express - Priya Sharma
Duronto Express - Vikram Singh
Himalayan Queen - Meera Joshi
South Western Rail - Rahul Verma
Deccan Express - Anjali Reddy
```

FIND THE PASSENGERS WITH THE SAME AGE

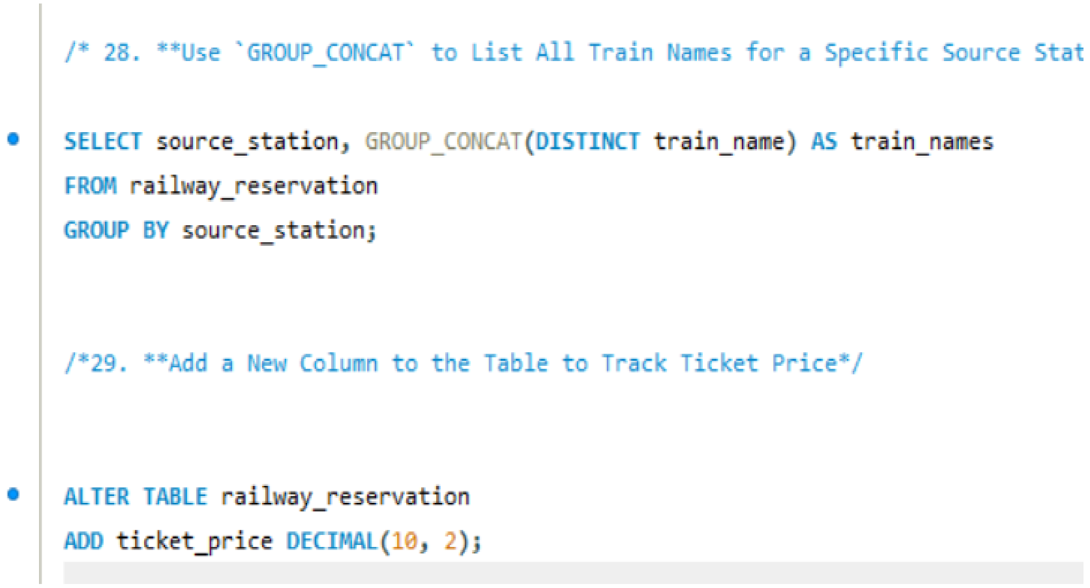


```
/* 26. **List Passengers Traveling Between Two Specific Stations*/

SELECT * FROM railway_reservation
WHERE source_station = 'Chennai' AND destination_station = 'Bangalore';

/* 27. **Find the Passengers with the Same Age*/

SELECT age, GROUP_CONCAT(passenger_name) AS passengers
FROM railway_reservation
GROUP BY age
HAVING COUNT(age) > 1;
```

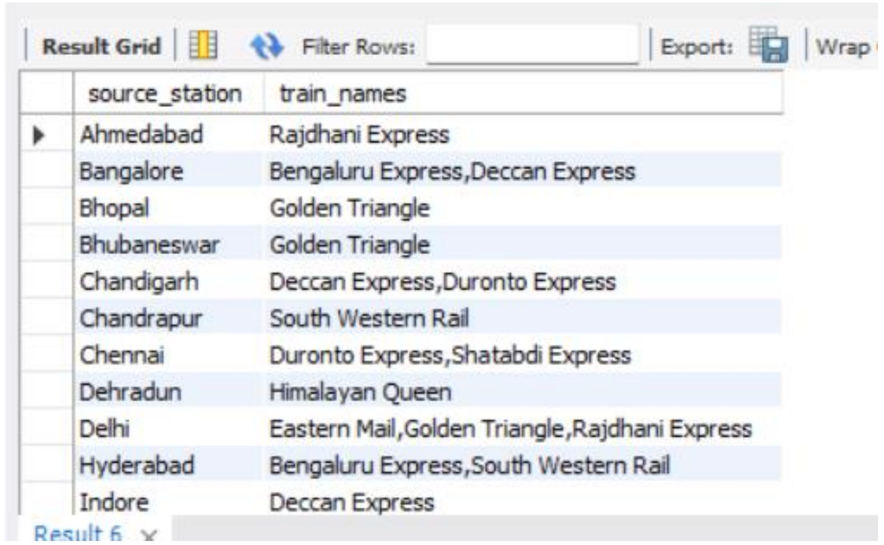


age	passengers
26	Prashant Reddy,Neha Singh
27	Ayesha Khan,Sonal Patil
28	Priya Sharma,Vijay Kumar
29	Geeta Deshmukh,Sandeep Patel
30	Sunita Agarwal,Divya Singh

USE GROUP CONCAT TO LIST ALL TRAINS NAMES

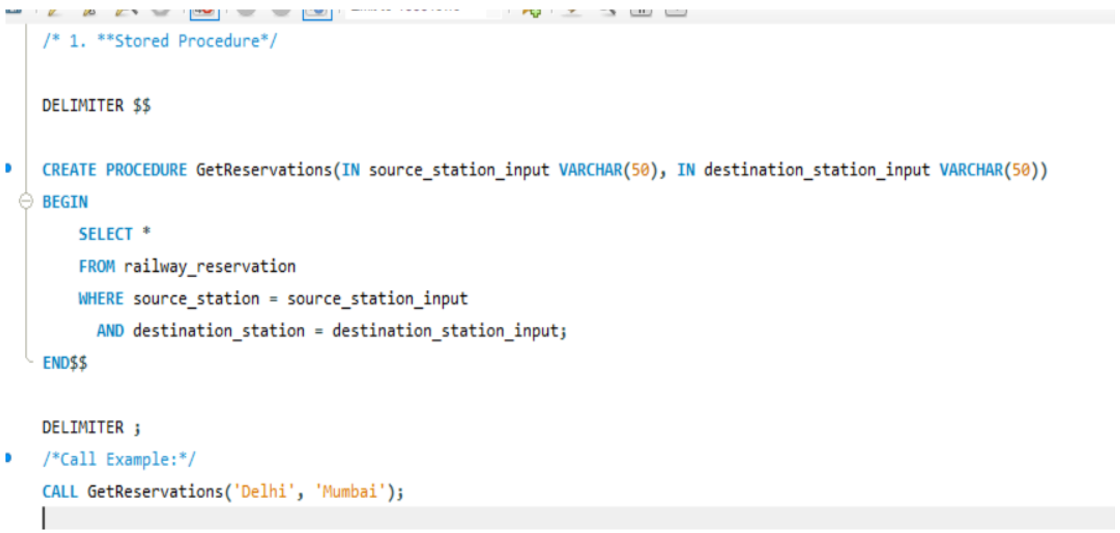


```
/* 28. **Use `GROUP_CONCAT` to List All Train Names for a Specific Source Stat

SELECT source_station, GROUP_CONCAT(DISTINCT train_name) AS train_names
FROM railway_reservation
GROUP BY source_station;


/*29. **Add a New Column to the Table to Track Ticket Price*/


ALTER TABLE railway_reservation
ADD ticket_price DECIMAL(10, 2);
```

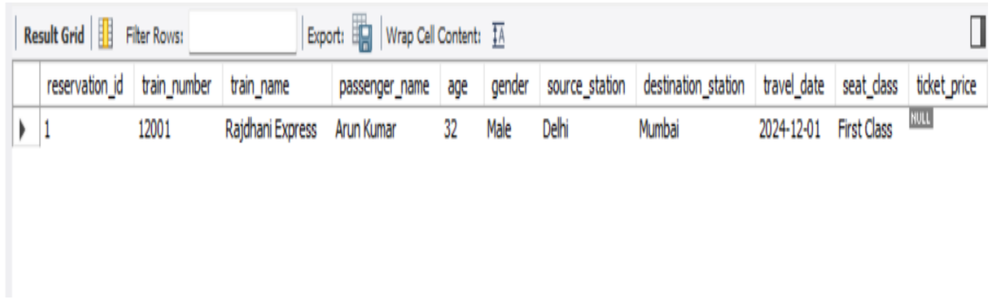


source_station	train_names
Ahmedabad	Rajdhani Express
Bangalore	Bengaluru Express,Deccan Express
Bhopal	Golden Triangle
Bhubaneswar	Golden Triangle
Chandigarh	Deccan Express,Duronto Express
Chandrapur	South Western Rail
Chennai	Duronto Express,Shatabdi Express
Dehradun	Himalayan Queen
Delhi	Eastern Mail,Golden Triangle,Rajdhani Express
Hyderabad	Bengaluru Express,South Western Rail
Indore	Deccan Express

STORED PROCEDURE



```
/* 1. **Stored Procedure*/

DELIMITER $$

CREATE PROCEDURE GetReservations(IN source_station_input VARCHAR(50), IN destination_station_input VARCHAR(50))
BEGIN
    SELECT *
    FROM railway_reservation
    WHERE source_station = source_station_input
      AND destination_station = destination_station_input;
END$$

DELIMITER ;
/*Call Example:*/
CALL GetReservations('Delhi', 'Mumbai');
```

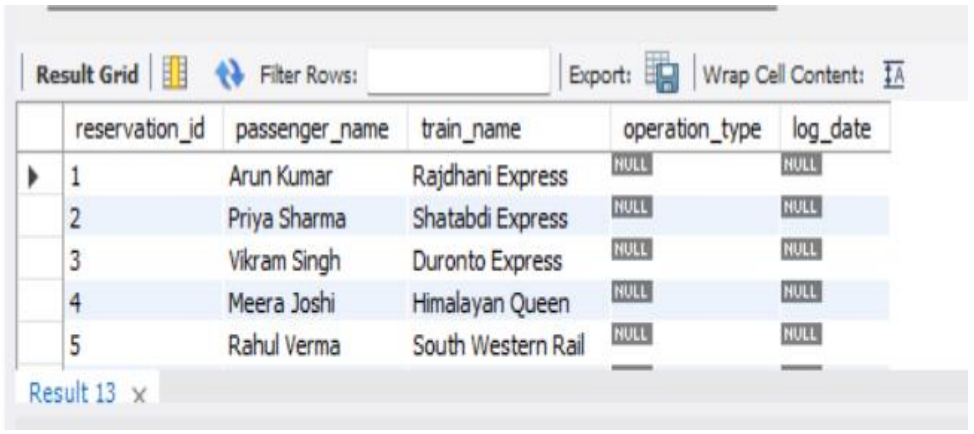


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

reservation_id	train_number	train_name	passenger_name	age	gender	source_station	destination_station	travel_date	seat_class	ticket_price
1	12001	Rajdhani Express	Arun Kumar	32	Male	Delhi	Mumbai	2024-12-01	First Class	NULL

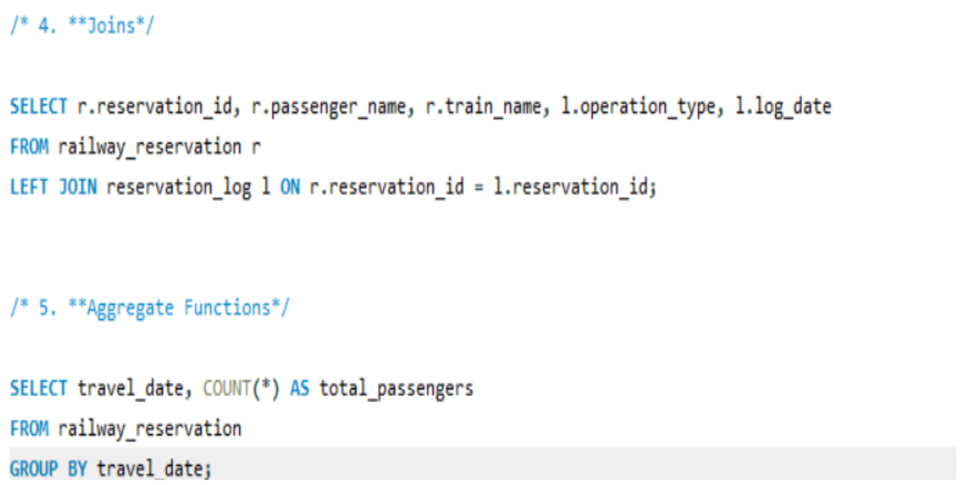
JOINS

```
/* 4. **Joins*/

SELECT r.reservation_id, r.passenger_name, r.train_name, l.operation_type, l.log_date
FROM railway_reservation r
LEFT JOIN reservation_log l ON r.reservation_id = l.reservation_id;


/* 5. **Aggregate Functions*/

SELECT travel_date, COUNT(*) AS total_passengers
FROM railway_reservation
GROUP BY travel_date;
```

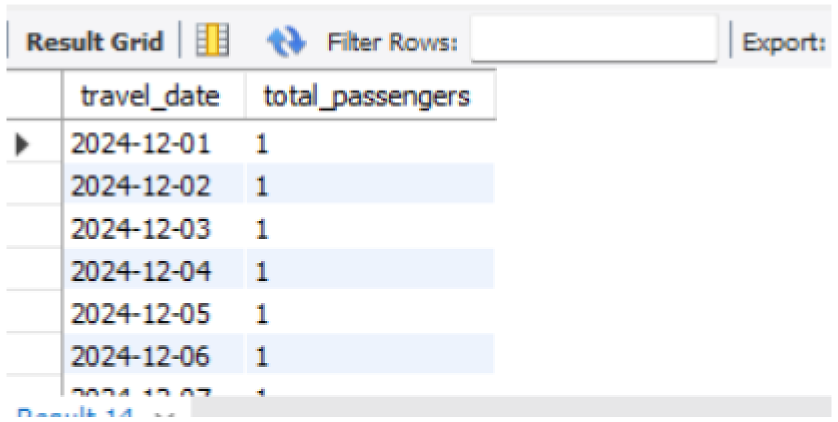


reservation_id	passenger_name	train_name	operation_type	log_date
1	Arun Kumar	Rajdhani Express	NULL	NULL
2	Priya Sharma	Shatabdi Express	NULL	NULL
3	Vikram Singh	Duronto Express	NULL	NULL
4	Meera Joshi	Himalayan Queen	NULL	NULL
5	Rahul Verma	South Western Rail	NULL	NULL

AGGREGATE

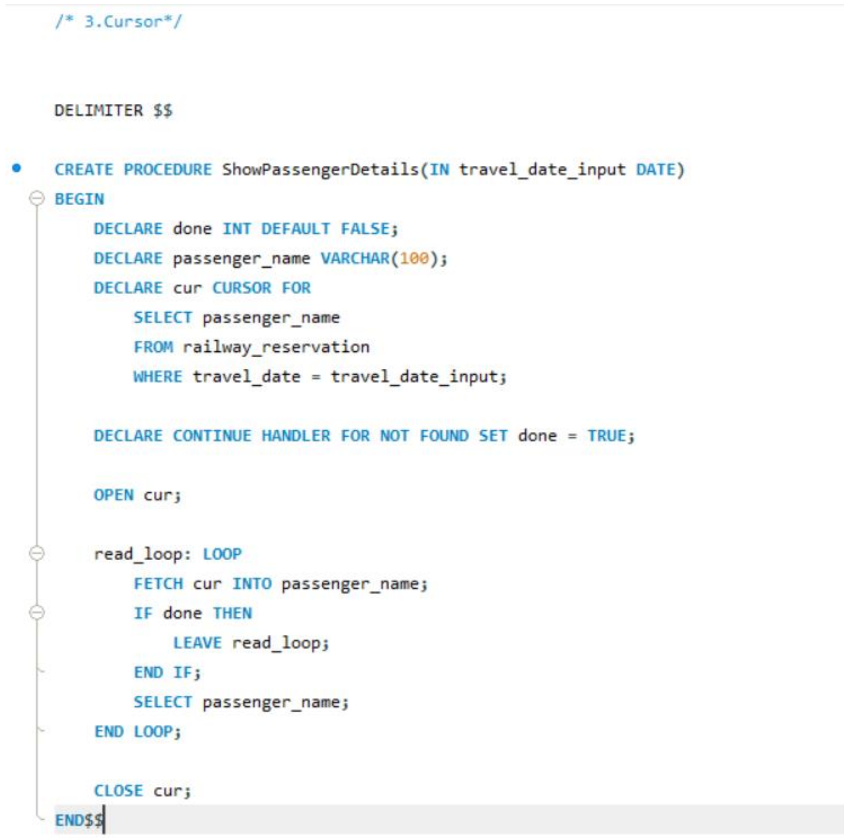


```
/* 4. **Joins*/

SELECT r.reservation_id, r.passenger_name, r.train_name, l.operation_type, l.log_date
FROM railway_reservation r
LEFT JOIN reservation_log l ON r.reservation_id = l.reservation_id;


/* 5. **Aggregate Functions*/

SELECT travel_date, COUNT(*) AS total_passengers
FROM railway_reservation
GROUP BY travel_date;
```



travel_date	total_passengers
2024-12-01	1
2024-12-02	1
2024-12-03	1
2024-12-04	1
2024-12-05	1
2024-12-06	1
2024-12-07	1

CURSOR



```
/* 3.Cursor*/


DELIMITER $$

CREATE PROCEDURE ShowPassengerDetails(IN travel_date_input DATE)
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE passenger_name VARCHAR(100);
    DECLARE cur CURSOR FOR
        SELECT passenger_name
        FROM railway_reservation
        WHERE travel_date = travel_date_input;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

    OPEN cur;

    read_loop: LOOP
        FETCH cur INTO passenger_name;
        IF done THEN
            LEAVE read_loop;
        END IF;
        SELECT passenger_name;
    END LOOP;

    CLOSE cur;
END$$
```

In SQL, a **cursor** is a database object used to **retrieve, manipulate, and iterate through a result set** row by row. Cursors are often employed in stored procedures or triggers when operations need to be performed sequentially on each row of a query result.

THANK YOU