CS2202: Mini-IRCTC

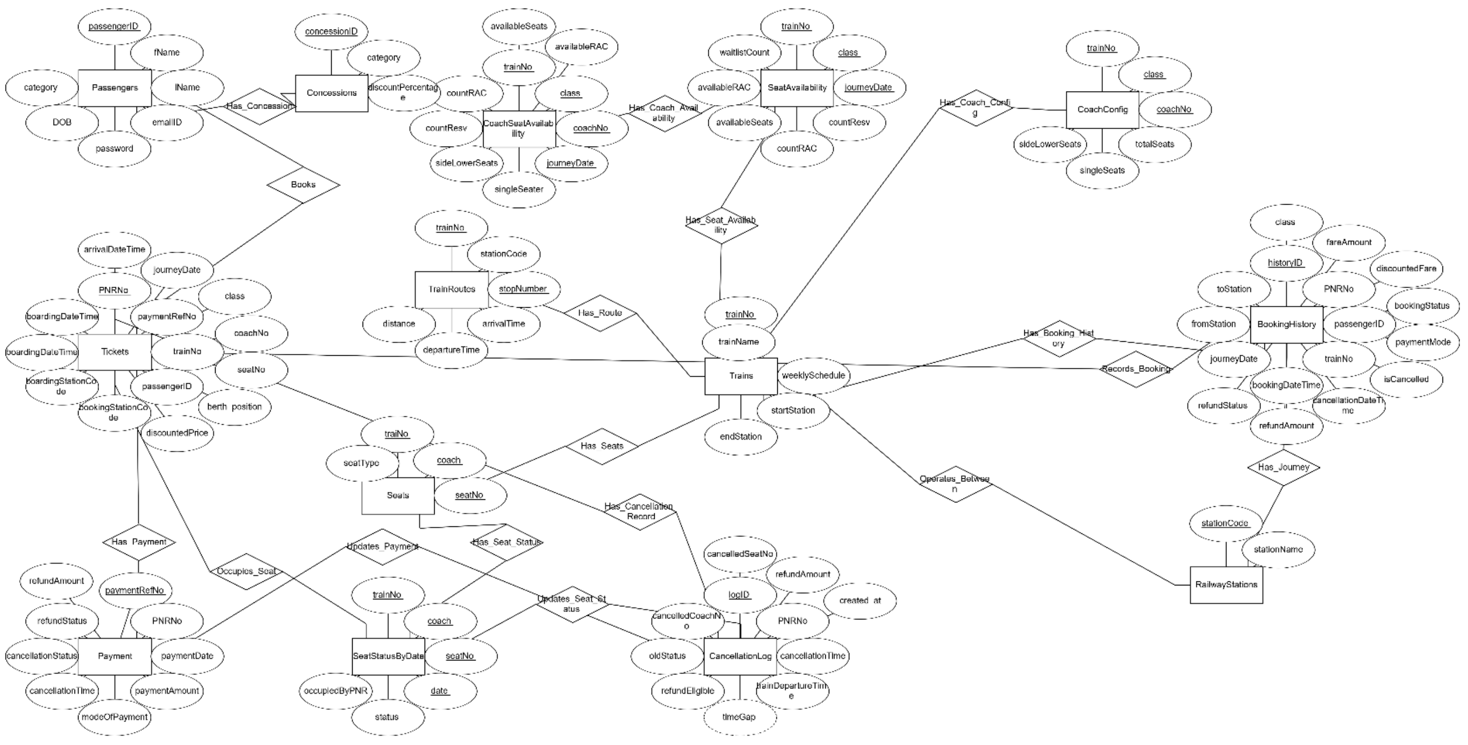
Project Overview

This project implements a comprehensive Indian Railway Ticket Reservation System (IRCTC) that allows passengers to book, modify, and cancel train tickets. The system manages train schedules, seat availability, payments, and various other aspects of railway ticket booking.

Drive Link containing CSV Files, Video Demo, Readme file, .SQL file, ER diagram, Relational Schema :
https://drive.google.com/drive/folders/1yPFskGxgu_Gwmni4K4Oqfh2zkFO1-rjg?usp=sharing
GitHub Link (README.md) : https://github.com/arbitcoper/Mini_IRCTC/blob/main/README.md

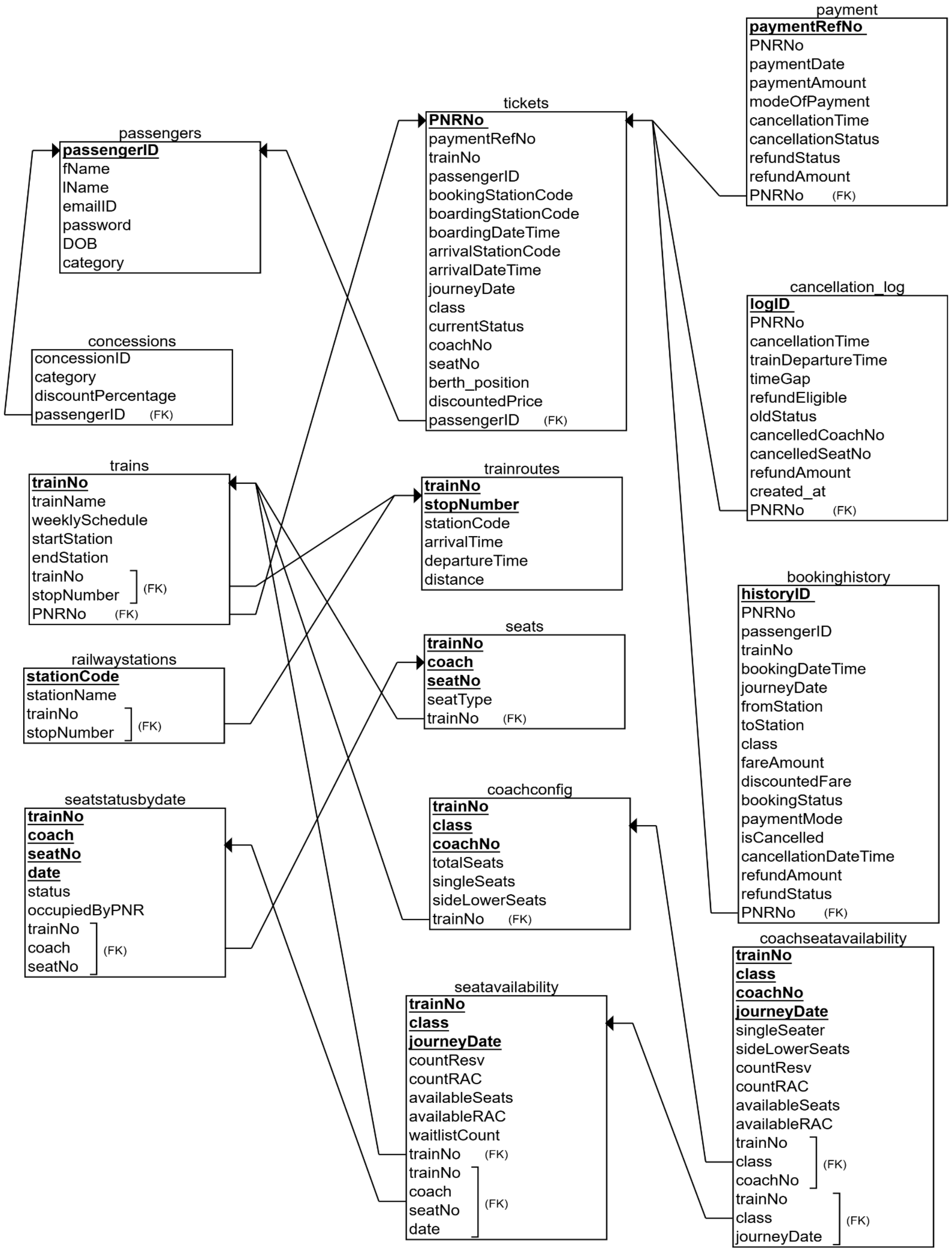
Team Details:
Sabbidi Neha Reddy – 2301AI42
Ankesh Kumar – 2301CS06
Krishan Kumawat – 2301CS25
Raushan Raj – 2301CS82

ER Diagram



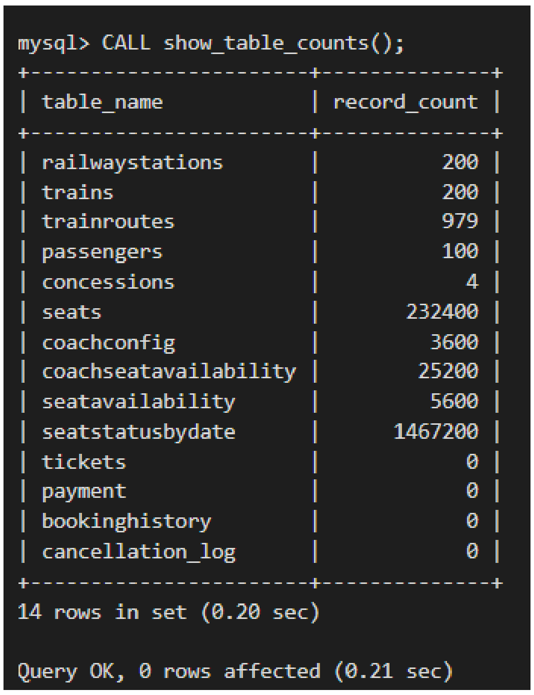
CS2202: Mini-IRCTC

Relational schema



CS2202: Mini-IRCTC

Sample data summary

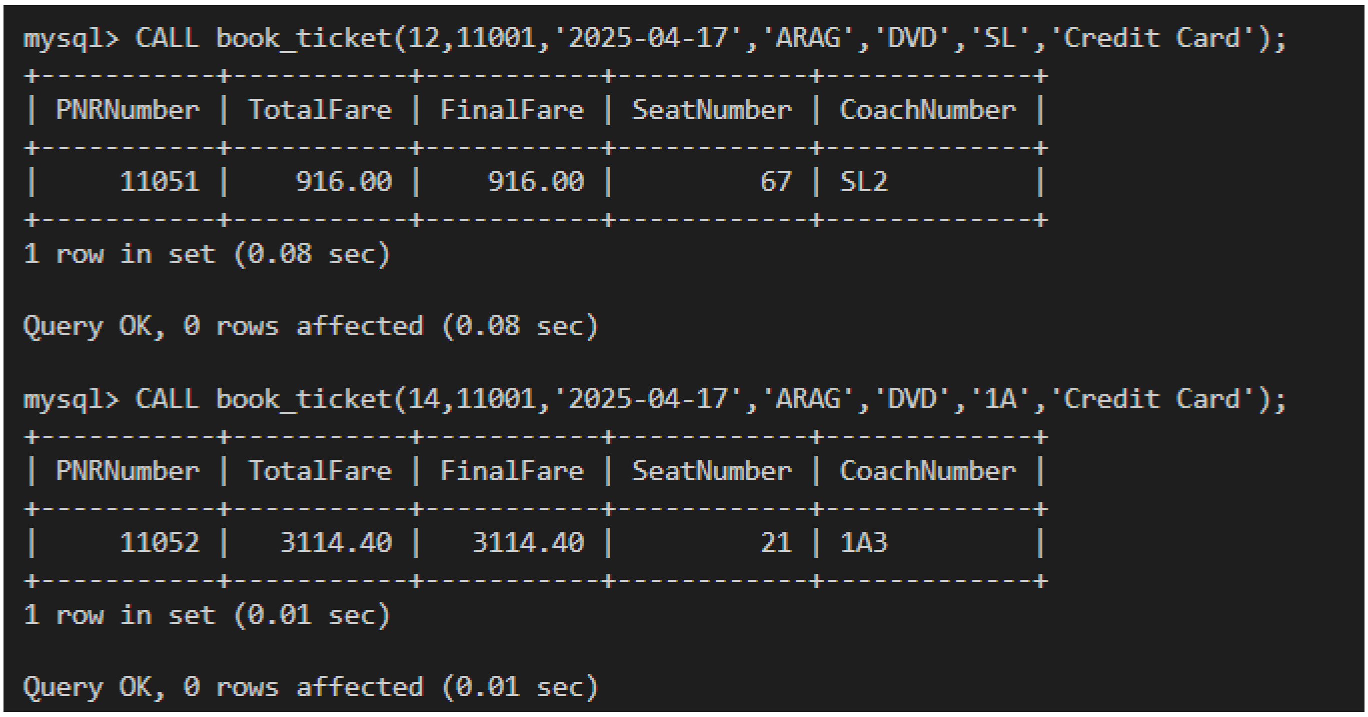


```
mysql> CALL show_table_counts();
+-----------------------+--------------+
| table_name            | record_count |
+-----------------------+--------------+
| railwaystations       |          200 |
| trains                |          200 |
| trainroutes           |          979 |
| passengers            |          100 |
| concessions           |            4 |
| seats                 |       232400 |
| coachconfig           |         3600 |
| coachseatavailability |        25200 |
| seatavailability      |         5600 |
| seatstatusbydate      |      1467200 |
| tickets               |            0 |
| payment               |            0 |
| bookinghistory        |            0 |
| cancellation_log      |            0 |
+-----------------------+--------------+
14 rows in set (0.20 sec)

Query OK, 0 rows affected (0.21 sec)
```

SQL Queries

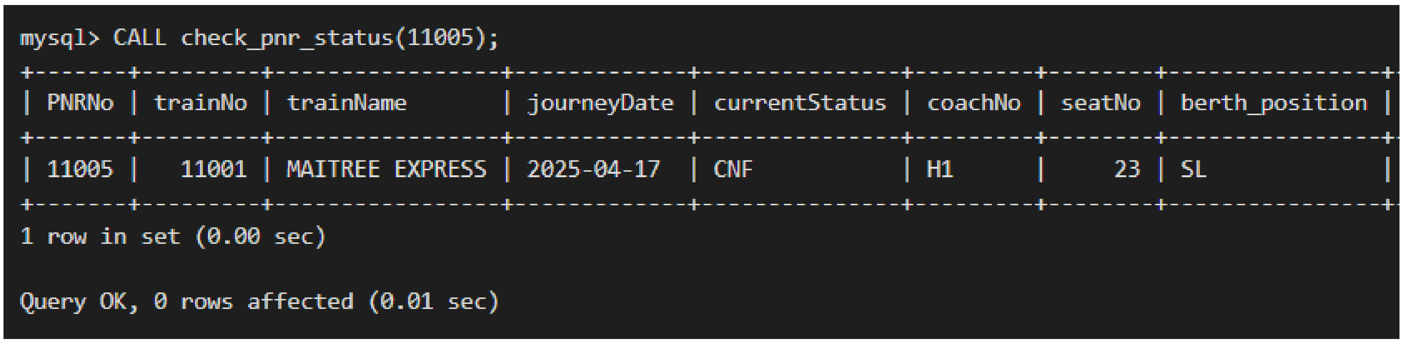
CS2202: Mini-IRCTC

Booking a ticket:

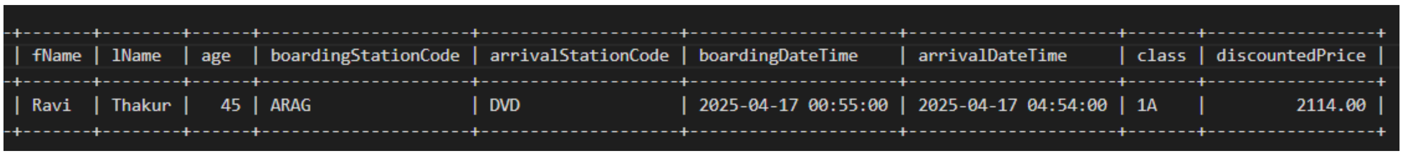


```
mysql> CALL book_ticket(12,11001,'2025-04-17','ARAG','DVD','SL','Credit Card');
+-----------+-----------+-----------+------------+-------------+
| PNRNumber | TotalFare | FinalFare | SeatNumber | CoachNumber |
+-----------+-----------+-----------+------------+-------------+
|     11051 |    916.00 |    916.00 |         67 | SL2         |
+-----------+-----------+-----------+------------+-------------+
1 row in set (0.08 sec)

Query OK, 0 rows affected (0.08 sec)

mysql> CALL book_ticket(14,11001,'2025-04-17','ARAG','DVD','1A','Credit Card');
+-----------+-----------+-----------+------------+-------------+
| PNRNumber | TotalFare | FinalFare | SeatNumber | CoachNumber |
+-----------+-----------+-----------+------------+-------------+
|     11052 |   3114.40 |   3114.40 |         21 | 1A3         |
+-----------+-----------+-----------+------------+-------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

PNR Enquiry:



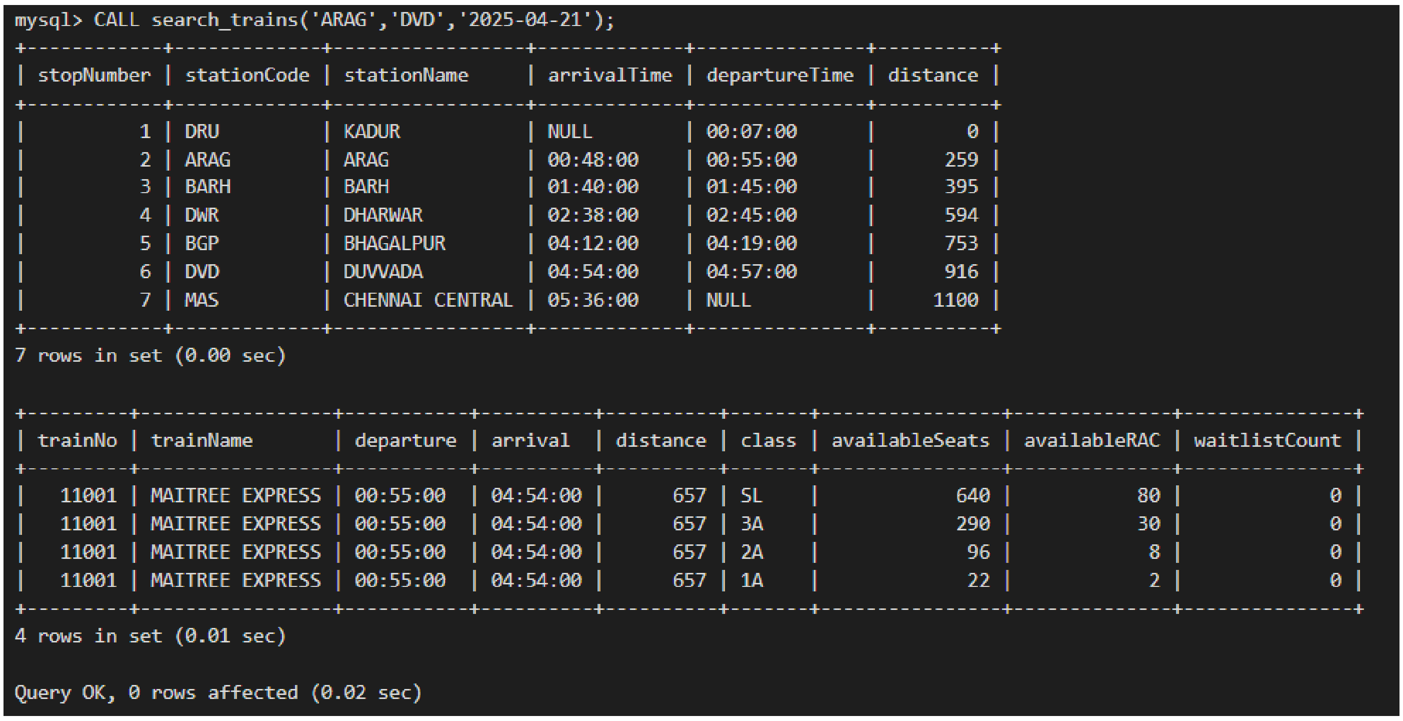
```
mysql> CALL check_pnr_status(11005);
+-------+---------+-----------------+-------------+---------------+---------+--------+----------------+
| PNRNo | trainNo | trainName       | journeyDate | currentStatus | coachNo | seatNo | berth_position |
+-------+---------+-----------------+-------------+---------------+---------+--------+----------------+
| 11005 |   11001 | MAITREE EXPRESS | 2025-04-17  | CNF           | H1      |     23 | SL             |
+-------+---------+-----------------+-------------+---------------+---------+--------+----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```



```
+-------+--------+-----+----------------------+--------------------+---------------------+---------------------+-------+-----------------+
| fName | lName  | age | boardingStationCode  | arrivalStationCode | boardingDateTime    | arrivalDateTime     | class | discountedPrice |
+-------+--------+-----+----------------------+--------------------+---------------------+---------------------+-------+-----------------+
| Ravi  | Thakur |  45 | ARAG                 | DVD                | 2025-04-17 00:55:00 | 2025-04-17 04:54:00 | 1A    |         2114.00 |
+-------+--------+-----+----------------------+--------------------+---------------------+---------------------+-------+-----------------+
```

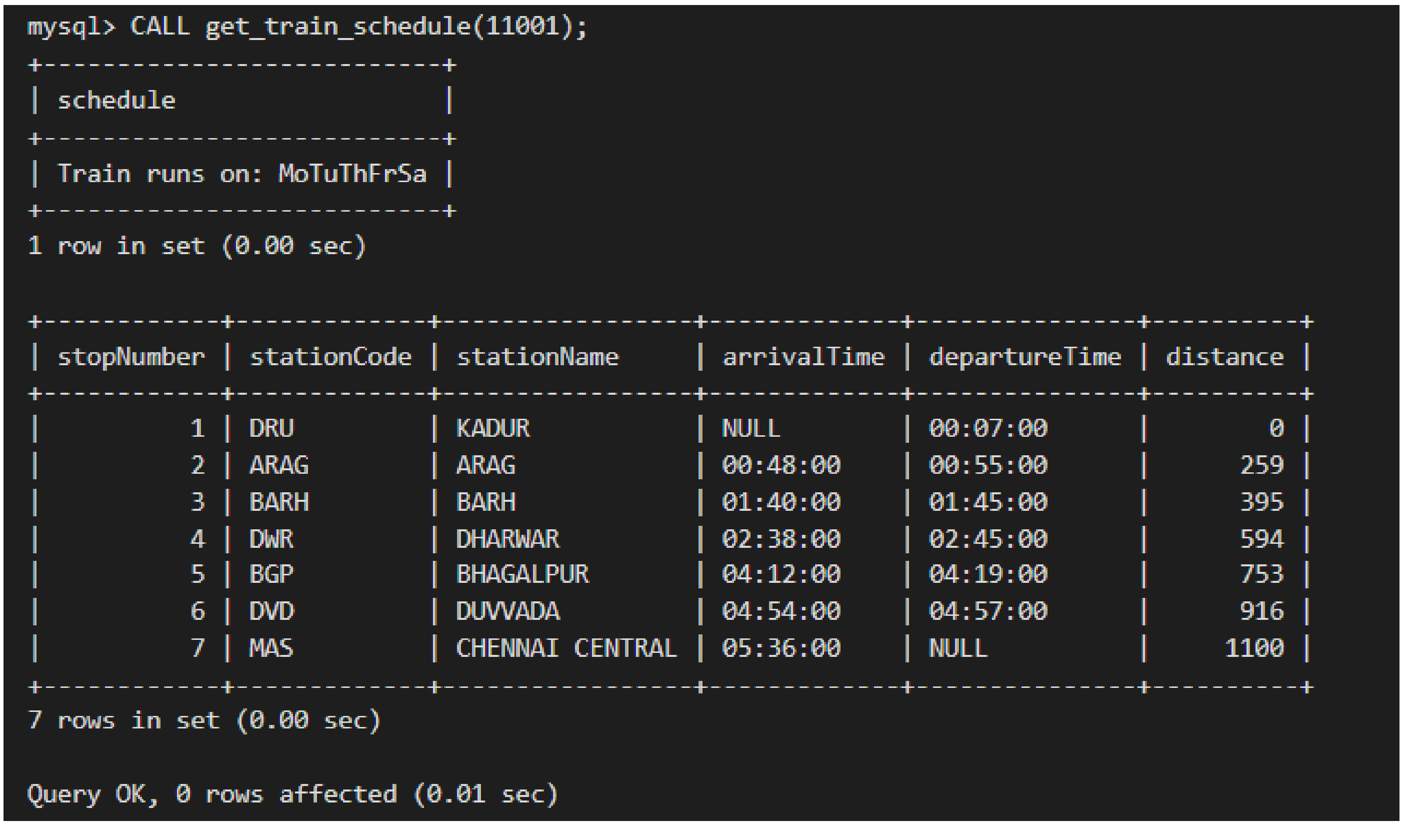
Search for a train:

CS2202: Mini-IRCTC



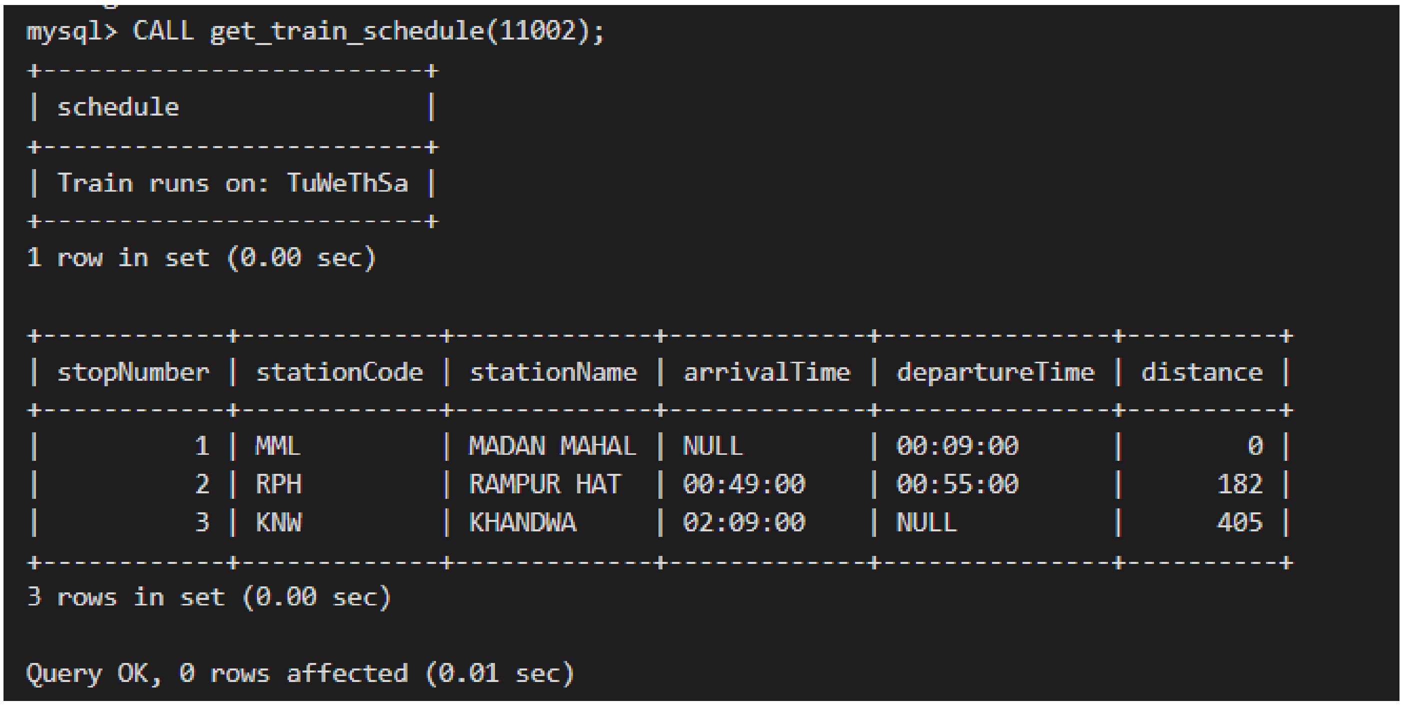
```
mysql> CALL search_trains('ARAG','DVD','2025-04-21');
+------------+-------------+-----------------+-------------+---------------+----------+
| stopNumber | stationCode | stationName     | arrivalTime | departureTime | distance |
+------------+-------------+-----------------+-------------+---------------+----------+
|          1 | DRU         | KADUR           | NULL        | 00:07:00      |        0 |
|          2 | ARAG        | ARAG            | 00:48:00    | 00:55:00      |      259 |
|          3 | BARH        | BARH            | 01:40:00    | 01:45:00      |      395 |
|          4 | DWR         | DHARWAR         | 02:38:00    | 02:45:00      |      594 |
|          5 | BGP         | BHAGALPUR       | 04:12:00    | 04:19:00      |      753 |
|          6 | DVD         | DUVVADA         | 04:54:00    | 04:57:00      |      916 |
|          7 | MAS         | CHENNAI CENTRAL | 05:36:00    | NULL          |     1100 |
+------------+-------------+-----------------+-------------+---------------+----------+
7 rows in set (0.00 sec)

+---------+-----------------+-----------+----------+----------+-------+----------------+--------------+---------------+
| trainNo | trainName       | departure | arrival  | distance | class | availableSeats | availableRAC | waitlistCount |
+---------+-----------------+-----------+----------+----------+-------+----------------+--------------+---------------+
|   11001 | MAITREE EXPRESS | 00:55:00  | 04:54:00 |      657 | SL    |            640 |           80 |             0 |
|   11001 | MAITREE EXPRESS | 00:55:00  | 04:54:00 |      657 | 3A    |            290 |           30 |             0 |
|   11001 | MAITREE EXPRESS | 00:55:00  | 04:54:00 |      657 | 2A    |             96 |            8 |             0 |
|   11001 | MAITREE EXPRESS | 00:55:00  | 04:54:00 |      657 | 1A    |             22 |            2 |             0 |
+---------+-----------------+-----------+----------+----------+-------+----------------+--------------+---------------+
4 rows in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)
```

Get Train Schedule:



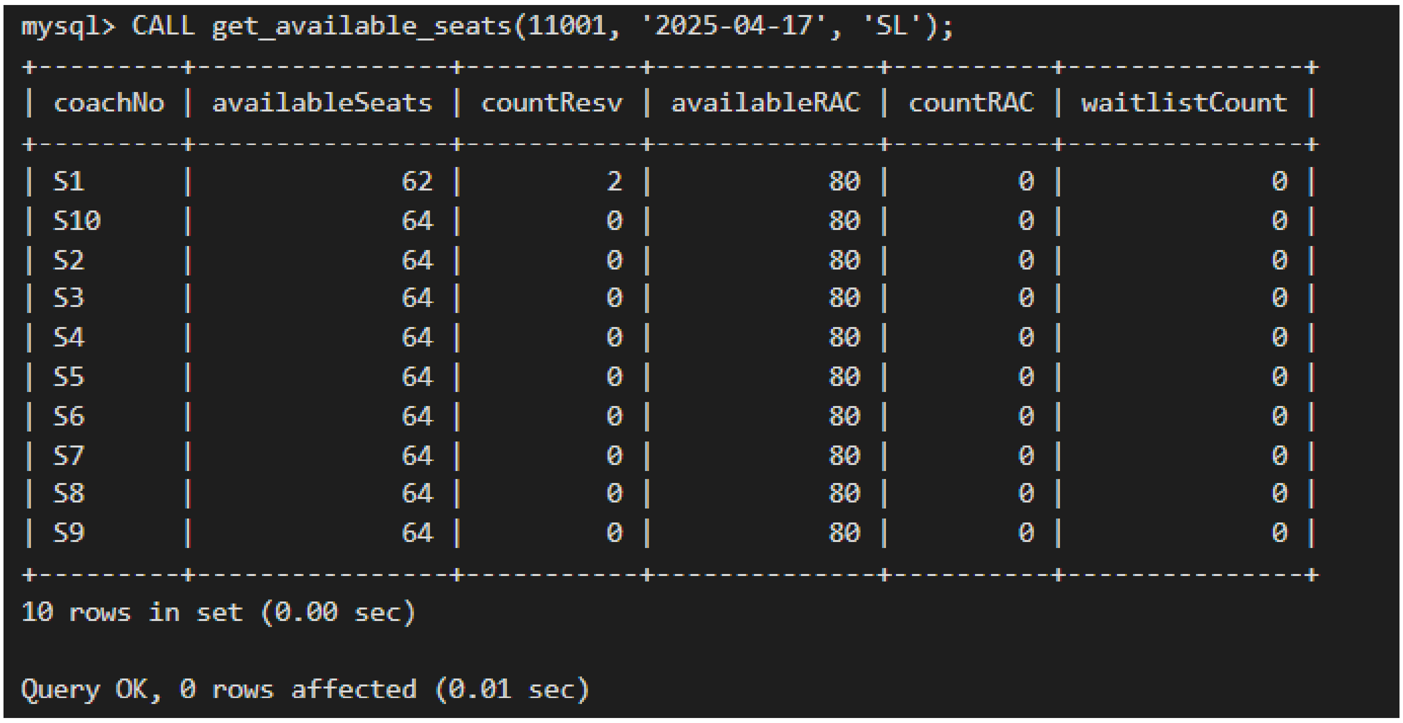
```
mysql> CALL get_train_schedule(11001);
+--------------------------+
| schedule                 |
+--------------------------+
| Train runs on: MoTuThFrSa |
+--------------------------+
1 row in set (0.00 sec)

+------------+-------------+-----------------+-------------+---------------+----------+
| stopNumber | stationCode | stationName     | arrivalTime | departureTime | distance |
+------------+-------------+-----------------+-------------+---------------+----------+
|          1 | DRU         | KADUR           | NULL        | 00:07:00      |        0 |
|          2 | ARAG        | ARAG            | 00:48:00    | 00:55:00      |      259 |
|          3 | BARH        | BARH            | 01:40:00    | 01:45:00      |      395 |
|          4 | DWR         | DHARWAR         | 02:38:00    | 02:45:00      |      594 |
|          5 | BGP         | BHAGALPUR       | 04:12:00    | 04:19:00      |      753 |
|          6 | DVD         | DUVVADA         | 04:54:00    | 04:57:00      |      916 |
|          7 | MAS         | CHENNAI CENTRAL | 05:36:00    | NULL          |     1100 |
+------------+-------------+-----------------+-------------+---------------+----------+
7 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

CS2202: Mini-IRCTC



```
mysql> CALL get_train_schedule(11002);
+-------------------------+
| schedule                |
+-------------------------+
| Train runs on: TuWeThSa |
+-------------------------+
1 row in set (0.00 sec)

+------------+-------------+-------------+-------------+---------------+----------+
| stopNumber | stationCode | stationName | arrivalTime | departureTime | distance |
+------------+-------------+-------------+-------------+---------------+----------+
|          1 | MML         | MADAN MAHAL | NULL        | 00:09:00      |        0 |
|          2 | RPH         | RAMPUR HAT  | 00:49:00    | 00:55:00      |      182 |
|          3 | KNW         | KHANDWA     | 02:09:00    | NULL          |      405 |
+------------+-------------+-------------+-------------+---------------+----------+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

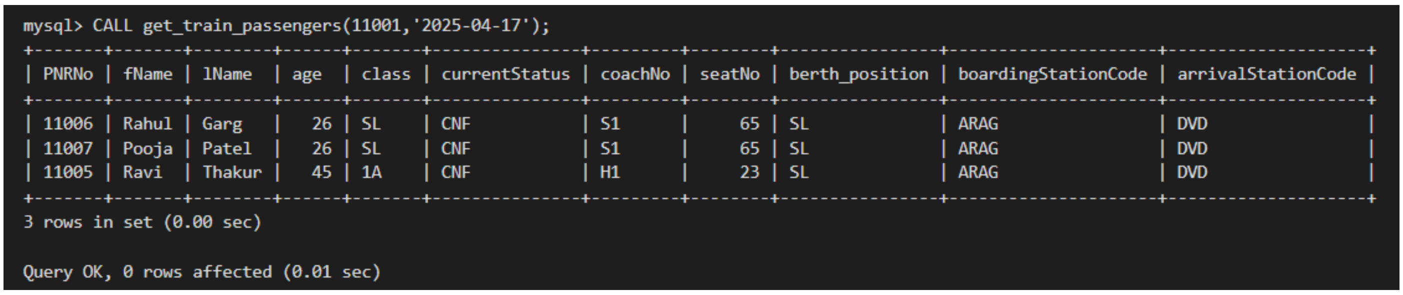
Available Seats:



```
mysql> CALL get_available_seats(11001, '2025-04-17', 'SL');
+---------+----------------+-----------+--------------+----------+---------------+
| coachNo | availableSeats | countResv | availableRAC | countRAC | waitlistCount |
+---------+----------------+-----------+--------------+----------+---------------+
| S1      |             62 |         2 |           80 |        0 |             0 |
| S10     |             64 |         0 |           80 |        0 |             0 |
| S2      |             64 |         0 |           80 |        0 |             0 |
| S3      |             64 |         0 |           80 |        0 |             0 |
| S4      |             64 |         0 |           80 |        0 |             0 |
| S5      |             64 |         0 |           80 |        0 |             0 |
| S6      |             64 |         0 |           80 |        0 |             0 |
| S7      |             64 |         0 |           80 |        0 |             0 |
| S8      |             64 |         0 |           80 |        0 |             0 |
| S9      |             64 |         0 |           80 |        0 |             0 |
+---------+----------------+-----------+--------------+----------+---------------+
10 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

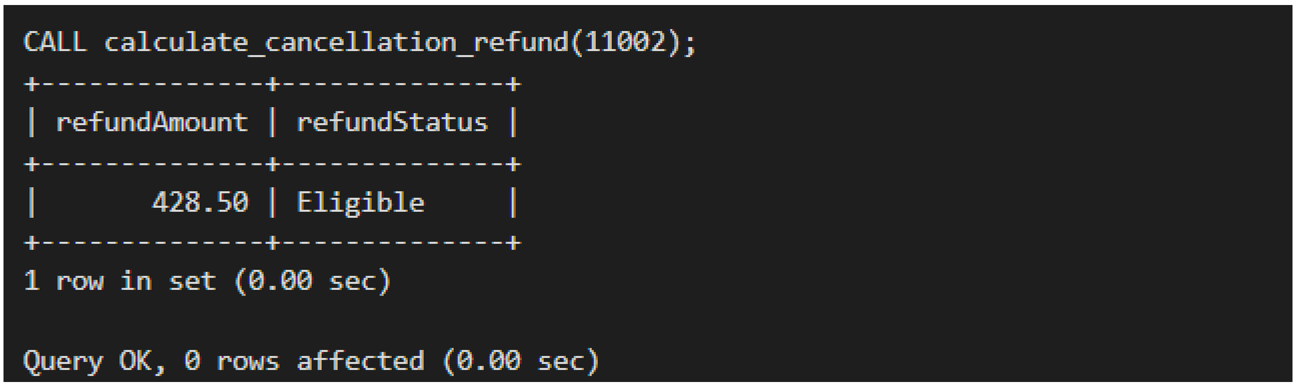
Train Passenger List:

CS2202: Mini-IRCTC



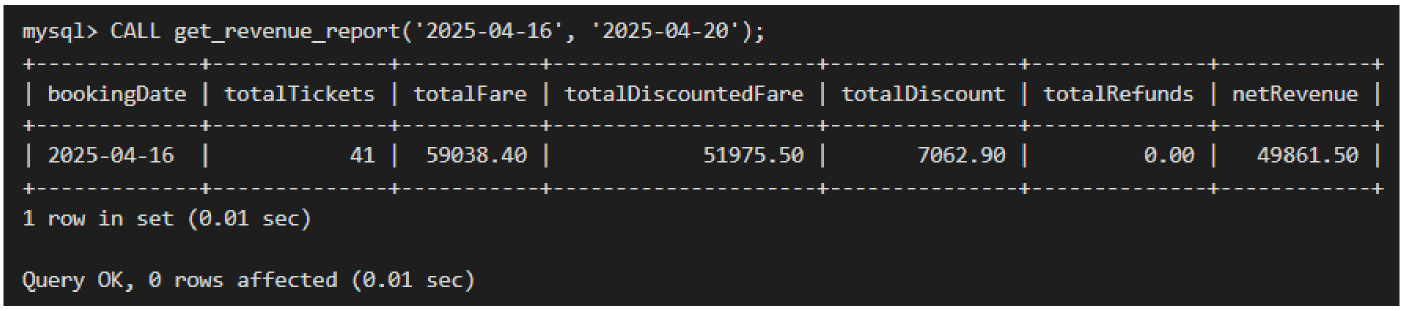
```
mysql> CALL get_train_passengers(11001,'2025-04-17');
+-------+-------+--------+-----+-------+---------------+---------+--------+----------------+----------------------+--------------------+
| PNRNo | fName | lName  | age | class | currentStatus | coachNo | seatNo | berth_position | boardingStationCode  | arrivalStationCode |
+-------+-------+--------+-----+-------+---------------+---------+--------+----------------+----------------------+--------------------+
| 11006 | Rahul | Garg   |  26 | SL    | CNF           | S1      |     65 | SL             | ARAG                 | DVD                |
| 11007 | Pooja | Patel  |  26 | SL    | CNF           | S1      |     65 | SL             | ARAG                 | DVD                |
| 11005 | Ravi  | Thakur |  45 | 1A    | CNF           | H1      |     23 | SL             | ARAG                 | DVD                |
+-------+-------+--------+-----+-------+---------------+---------+--------+----------------+----------------------+--------------------+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

Cancellation Refund Status:



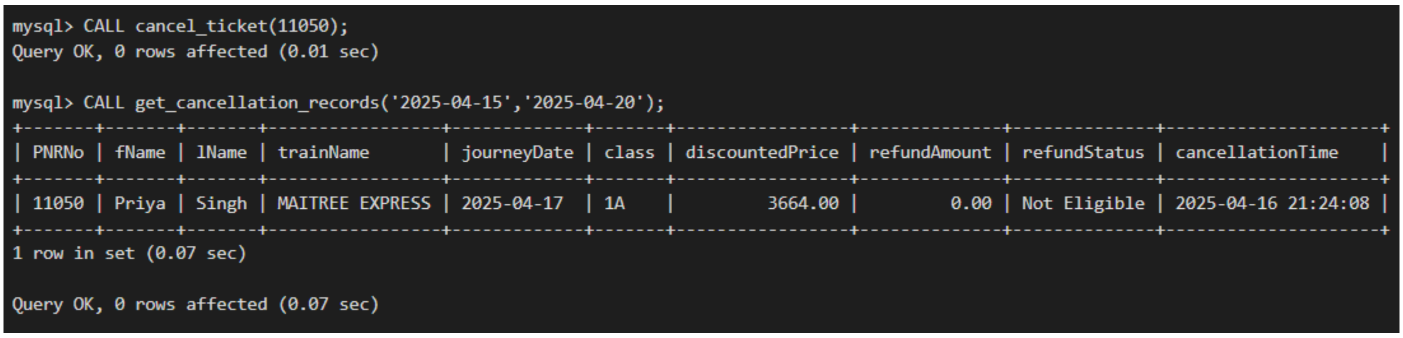
```
CALL calculate_cancellation_refund(11002);
+--------------+--------------+
| refundAmount | refundStatus |
+--------------+--------------+
|       428.50 | Eligible     |
+--------------+--------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Revenue Report:



```
mysql> CALL get_revenue_report('2025-04-16', '2025-04-20');
+-------------+--------------+-----------+--------------------+---------------+--------------+------------+
| bookingDate | totalTickets | totalFare | totalDiscountedFare | totalDiscount | totalRefunds | netRevenue |
+-------------+--------------+-----------+--------------------+---------------+--------------+------------+
| 2025-04-16  |           41 |  59038.40 |            51975.50 |       7062.90 |         0.00 |   49861.50 |
+-------------+--------------+-----------+--------------------+---------------+--------------+------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

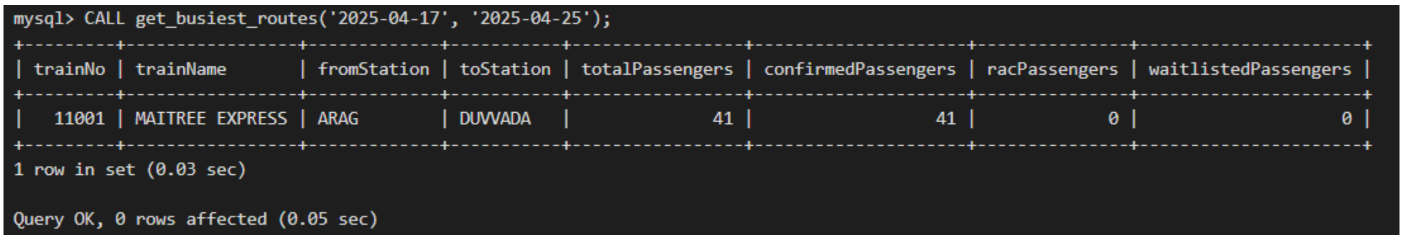
Cancelling a ticket:



```
mysql> CALL cancel_ticket(11050);
Query OK, 0 rows affected (0.01 sec)

mysql> CALL get_cancellation_records('2025-04-15','2025-04-20');
+-------+-------+-------+-----------------+-------------+-------+-----------------+--------------+--------------+---------------------+
| PNRNo | fName | lName | trainName       | journeyDate | class | discountedPrice | refundAmount | refundStatus | cancellationTime    |
+-------+-------+-------+-----------------+-------------+-------+-----------------+--------------+--------------+---------------------+
| 11050 | Priya | Singh | MAITREE EXPRESS | 2025-04-17  | 1A    |         3664.00 |         0.00 | Not Eligible | 2025-04-16 21:24:08 |
+-------+-------+-------+-----------------+-------------+-------+-----------------+--------------+--------------+---------------------+
1 row in set (0.07 sec)

Query OK, 0 rows affected (0.07 sec)
```

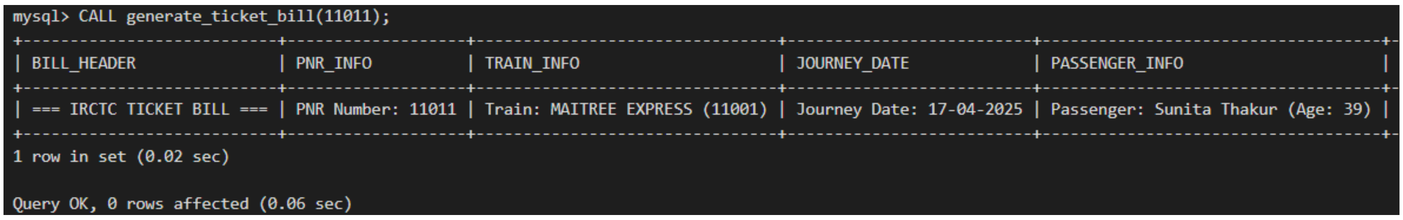
Busiest Route based on number of passengers:

CS2202: Mini-IRCTC

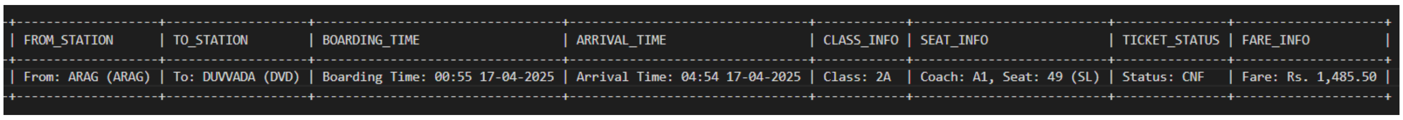


```
mysql> CALL get_busiest_routes('2025-04-17', '2025-04-25');
+---------+----------------+-------------+-----------+-----------------+---------------------+---------------+-----------------------+
| trainNo | trainName      | fromStation | toStation | totalPassengers | confirmedPassengers | racPassengers | waitlistedPassengers |
+---------+----------------+-------------+-----------+-----------------+---------------------+---------------+-----------------------+
|   11001 | MAITREE EXPRESS | ARAG       | DUVVADA   |              41 |                  41 |             0 |                     0 |
+---------+----------------+-------------+-----------+-----------------+---------------------+---------------+-----------------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.05 sec)
```

Ticket Bill:



```
mysql> CALL generate_ticket_bill(11011);
+-------------------------+-------------------+-------------------------------+------------------------+--------------------------------------+
| BILL_HEADER             | PNR_INFO          | TRAIN_INFO                    | JOURNEY_DATE           | PASSENGER_INFO                       |
+-------------------------+-------------------+-------------------------------+------------------------+--------------------------------------+
| === IRCTC TICKET BILL === | PNR Number: 11011 | Train: MAITREE EXPRESS (11001) | Journey Date: 17-04-2025 | Passenger: Sunita Thakur (Age: 39) |
+-------------------------+-------------------+-------------------------------+------------------------+--------------------------------------+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.06 sec)
```



```
+------------------+------------------+--------------------------------+-------------------------------+------------+------------------------+---------------+-------------------+
| FROM_STATION     | TO_STATION       | BOARDING_TIME                  | ARRIVAL_TIME                  | CLASS_INFO | SEAT_INFO              | TICKET_STATUS | FARE_INFO         |
+------------------+------------------+--------------------------------+-------------------------------+------------+------------------------+---------------+-------------------+
| From: ARAG (ARAG) | To: DUVVADA (DVD) | Boarding Time: 00:55 17-04-2025 | Arrival Time: 04:54 17-04-2025 | Class: 2A  | Coach: A1, Seat: 49 (SL) | Status: CNF   | Fare: Rs. 1,485.50 |
+------------------+------------------+--------------------------------+-------------------------------+------------+------------------------+---------------+-------------------+
```

Functions, Procedures & Triggers

Procedures

1. book_ticket

```
CREATE PROCEDURE book_ticket(
    IN p_passenger_id INT,
    IN p_train_id INT,
    IN p_journey_date DATE,
    IN p_source_station_id INT,
    IN p_destination_station_id INT,
    IN p_class_id INT
)
BEGIN
    DECLARE v_seat_available BOOLEAN;
    DECLARE v_pnr_number VARCHAR(20);
    DECLARE v_fare DECIMAL(10,2);
    DECLARE v_distance INT;

    -- Check seat availability
```

```
   SELECT CheckSeatAvailability(p_train_id, p_journey_date, p_class_id) INTO
v_seat_available;

   IF v_seat_available THEN
      -- Get distance between stations
      SELECT distance INTO v_distance FROM routes
      WHERE source_station_id = p_source_station_id
      AND destination_station_id = p_destination_station_id;

      -- Calculate fare
      SET v_fare = CalculateFare(v_distance, p_class_id, 'ADULT');

      -- Generate PNR
      SET v_pnr_number = CONCAT(DATE_FORMAT(p_journey_date, '%y%m%d'),
                      LPAD(p_train_id, 4, '0'),
                      LPAD(FLOOR(RAND() * 10000), 4, '0'));

      -- Insert ticket
      INSERT INTO tickets (pnr_number, passenger_id, train_id, journey_date,
                 source_station_id, destination_station_id, class_id,
                 fare, status, booking_date)
      VALUES (v_pnr_number, p_passenger_id, p_train_id, p_journey_date,
            p_source_station_id, p_destination_station_id, p_class_id,
            v_fare, 'CNF', CURDATE());

      -- Update seat availability
      UPDATE seat_availability
      SET available_seats = available_seats - 1
      WHERE train_id = p_train_id
      AND journey_date = p_journey_date
      AND class_id = p_class_id;
   ELSE
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'No seats available for the selected train and class';
   END IF;
END;
```

2. calculate_cancellation_refund

```
CREATE PROCEDURE calculate_cancellation_refund(
    IN p_ticket_id INT,
    IN p_cancellation_date DATE
)
BEGIN
    DECLARE v_journey_date DATE;
    DECLARE v_fare DECIMAL(10,2);
    DECLARE v_status VARCHAR(20);
    DECLARE v_refund_amount DECIMAL(10,2);
    DECLARE v_days_before_journey INT;

    -- Get ticket details
    SELECT journey_date, fare, status
    INTO v_journey_date, v_fare, v_status
    FROM tickets
    WHERE ticket_id = p_ticket_id;

    -- Calculate days before journey
    SET v_days_before_journey = DATEDIFF(v_journey_date, p_cancellation_date);

    -- Calculate refund based on cancellation rules
    IF v_days_before_journey >= 30 THEN
        SET v_refund_amount = v_fare * 0.95; -- 95% refund
    ELSEIF v_days_before_journey >= 15 THEN
        SET v_refund_amount = v_fare * 0.75; -- 75% refund
    ELSEIF v_days_before_journey >= 3 THEN
        SET v_refund_amount = v_fare * 0.50; -- 50% refund
    ELSE
        SET v_refund_amount = 0; -- No refund
    END IF;

    -- Update refund amount in cancellation record
    UPDATE cancellations
    SET refund_amount = v_refund_amount
    WHERE ticket_id = p_ticket_id;
END;
```

3. cancel_ticket

```
CREATE PROCEDURE cancel_ticket(
    IN p_ticket_id INT,
    IN p_cancellation_reason VARCHAR(255)
)
BEGIN
    DECLARE v_status VARCHAR(20);
    DECLARE v_pnr_number VARCHAR(20);

    -- Get current ticket status
    SELECT status, pnr_number INTO v_status, v_pnr_number
    FROM tickets
    WHERE ticket_id = p_ticket_id;

    -- Check if ticket can be cancelled
    IF v_status IN ('CNF', 'RAC') THEN
      -- Insert cancellation record
      INSERT INTO cancellations (ticket_id, cancellation_date, reason)
      VALUES (p_ticket_id, CURDATE(), p_cancellation_reason);

      -- Calculate refund
      CALL calculate_cancellation_refund(p_ticket_id, CURDATE());

      -- Update ticket status
      UPDATE tickets
      SET status = 'CANCELLED'
      WHERE ticket_id = p_ticket_id;

      -- Update seat availability
      UPDATE seat_availability
      SET available_seats = available_seats + 1
      WHERE train_id = (SELECT train_id FROM tickets WHERE ticket_id = p_ticket_id)
      AND journey_date = (SELECT journey_date FROM tickets WHERE ticket_id = p_ticket_id)
      AND class_id = (SELECT class_id FROM tickets WHERE ticket_id = p_ticket_id);
    ELSE
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Ticket cannot be cancelled';
    END IF;
END;
```

4. check_pnr_status

```
CREATE PROCEDURE check_pnr_status(
    IN p_pnr_number VARCHAR(20)
)
BEGIN
    SELECT
        t.pnr_number,
        t.status,
        t.journey_date,
        t.source_station_id,
        t.destination_station_id,
        t.class_id,
        t.fare,
        p.name as passenger_name,
        p.age,
        p.gender
    FROM tickets t
    JOIN passengers p ON t.passenger_id = p.passenger_id
    WHERE t.pnr_number = p_pnr_number;
END;
```

5. generate_ticket_bill

```
CREATE PROCEDURE generate_ticket_bill(
    IN p_ticket_id INT
)
BEGIN
    SELECT
        t.pnr_number,
        t.journey_date,
        s1.station_name as source_station,
        s2.station_name as destination_station,
        c.class_name,
        t.fare,
        p.name as passenger_name,
        p.age,
        p.gender,
        t.booking_date,
        t.status
    FROM tickets t
    JOIN passengers p ON t.passenger_id = p.passenger_id
    JOIN stations s1 ON t.source_station_id = s1.station_id
    JOIN stations s2 ON t.destination_station_id = s2.station_id
```

```
  JOIN classes c ON t.class_id = c.class_id
  WHERE t.ticket_id = p_ticket_id;
END;
```

6. get_available_seats

```
CREATE PROCEDURE get_available_seats(
  IN p_train_id INT,
  IN p_journey_date DATE,
  IN p_class_id INT
)
BEGIN
  SELECT
    sa.available_seats,
    sa.available_rac,
    sa.waitlist_count,
    c.class_name,
    t.train_name
  FROM seat_availability sa
  JOIN trains t ON sa.train_id = t.train_id
  JOIN classes c ON sa.class_id = c.class_id
  WHERE sa.train_id = p_train_id
  AND sa.journey_date = p_journey_date
  AND sa.class_id = p_class_id;
END;
```

7. get_busiest_routes

```
CREATE PROCEDURE get_busiest_routes(
  IN p_start_date DATE,
  IN p_end_date DATE
)
BEGIN
  SELECT
    s1.station_name as source_station,
    s2.station_name as destination_station,
    COUNT(*) as total_bookings
  FROM tickets t
  JOIN stations s1 ON t.source_station_id = s1.station_id
  JOIN stations s2 ON t.destination_station_id = s2.station_id
  WHERE t.journey_date BETWEEN p_start_date AND p_end_date
  GROUP BY t.source_station_id, t.destination_station_id
  ORDER BY total_bookings DESC
  LIMIT 10;
END;
```

8. get_cancellation_records

```
CREATE PROCEDURE get_cancellation_records(
  IN p_start_date DATE,
  IN p_end_date DATE
)
BEGIN
  SELECT
    c.cancellation_id,
    t.pnr_number,
    c.cancellation_date,
    c.reason,
    c.refund_amount,
    t.journey_date,
    p.name as passenger_name
  FROM cancellations c
  JOIN tickets t ON c.ticket_id = t.ticket_id
  JOIN passengers p ON t.passenger_id = p.passenger_id
  WHERE c.cancellation_date BETWEEN p_start_date AND p_end_date;
END;
```

9. get_revenue_report

```
CREATE PROCEDURE get_revenue_report(
  IN p_start_date DATE,
  IN p_end_date DATE
)
BEGIN
  SELECT
    DATE_FORMAT(booking_date, '%Y-%m-%d') as booking_date,
    COUNT(*) as total_bookings,
    SUM(fare) as total_revenue,
    SUM(CASE WHEN status = 'CANCELLED' THEN fare ELSE 0 END) as cancelled_revenue,
    SUM(CASE WHEN status = 'CANCELLED' THEN 1 ELSE 0 END) as cancelled_count
  FROM tickets
  WHERE booking_date BETWEEN p_start_date AND p_end_date
  GROUP BY DATE_FORMAT(booking_date, '%Y-%m-%d')
  ORDER BY booking_date;
```

```
END;
```

10. get_train_passengers

```
CREATE PROCEDURE get_train_passengers(
  IN p_train_id INT,
  IN p_journey_date DATE
)
BEGIN
  SELECT
    t.pnr_number,
    p.name as passenger_name,
    p.age,
    p.gender,
    s1.station_name as boarding_station,
    s2.station_name as destination_station,
    c.class_name,
    t.status
  FROM tickets t
  JOIN passengers p ON t.passenger_id = p.passenger_id
  JOIN stations s1 ON t.source_station_id = s1.station_id
  JOIN stations s2 ON t.destination_station_id = s2.station_id
  JOIN classes c ON t.class_id = c.class_id
  WHERE t.train_id = p_train_id
  AND t.journey_date = p_journey_date;
END;
```

11. get_train_schedule

```
CREATE PROCEDURE get_train_schedule(
  IN p_train_id INT
)
BEGIN
  SELECT
    s.station_name,
    ts.arrival_time,
    ts.departure_time,
    ts.platform_number,
    ts.distance_from_source
  FROM train_schedule ts
  JOIN stations s ON ts.station_id = s.station_id
  WHERE ts.train_id = p_train_id
```

```
    ORDER BY ts.distance_from_source;
END;
```

12. get_waitlisted_passengers

```
CREATE PROCEDURE get_waitlisted_passengers(
    IN p_train_id INT,
    IN p_journey_date DATE
)
BEGIN
    SELECT
        t.pnr_number,
        p.name as passenger_name,
        p.age,
        p.gender,
        s1.station_name as boarding_station,
        s2.station_name as destination_station,
        c.class_name,
        t.waitlist_number
    FROM tickets t
    JOIN passengers p ON t.passenger_id = p.passenger_id
    JOIN stations s1 ON t.source_station_id = s1.station_id
    JOIN stations s2 ON t.destination_station_id = s2.station_id
    JOIN classes c ON t.class_id = c.class_id
    WHERE t.train_id = p_train_id
    AND t.journey_date = p_journey_date
    AND t.status = 'WL'
    ORDER BY t.waitlist_number;
END;
```

13. search_trains

```
CREATE PROCEDURE search_trains(
    IN p_source_station VARCHAR(100),
    IN p_dest_station VARCHAR(100),
    IN p_journey_date DATE
)
BEGIN
    SELECT DISTINCT
        t.train_id,
        t.train_name,
        t.train_number,
        c.class_name,
        sa.available_seats,
        sa.available_rac,
        sa.waitlist_count,
        r.distance,
        CalculateFare(r.distance, c.class_id, 'ADULT') as fare
    FROM trains t
    JOIN routes r ON t.train_id = r.train_id
    JOIN stations s1 ON r.source_station_id = s1.station_id
    JOIN stations s2 ON r.destination_station_id = s2.station_id
    JOIN seat_availability sa ON t.train_id = sa.train_id
    JOIN classes c ON sa.class_id = c.class_id
    WHERE s1.station_name = p_source_station
    AND s2.station_name = p_dest_station
    AND sa.journey_date = p_journey_date
    AND sa.available_seats > 0;
END;
```

14. show_table_counts

```
CREATE PROCEDURE show_table_counts()
BEGIN
    SELECT
        table_name,
        table_rows
    FROM information_schema.tables
    WHERE table_schema = 'mini_irctc1'
    ORDER BY table_name;
END;
```

<u>Functions</u>

1. CalculateFare

```
CREATE FUNCTION CalculateFare(
    p_distance INT,
    p_class_id INT,
    p_passenger_type VARCHAR(20)
) RETURNS DECIMAL(10,2)
```

```
BEGIN
   DECLARE v_base_fare DECIMAL(10,2);
   DECLARE v_class_multiplier DECIMAL(10,2);
   DECLARE v_passenger_multiplier DECIMAL(10,2);

   -- Get base fare per km
   SET v_base_fare = 2.0;

   -- Get class multiplier
   SELECT multiplier INTO v_class_multiplier
   FROM classes
   WHERE class_id = p_class_id;

   -- Get passenger type multiplier
   CASE p_passenger_type
      WHEN 'ADULT' THEN SET v_passenger_multiplier = 1.0;
      WHEN 'CHILD' THEN SET v_passenger_multiplier = 0.5;
      WHEN 'SENIOR' THEN SET v_passenger_multiplier = 0.75;
      ELSE SET v_passenger_multiplier = 1.0;
   END CASE;

   RETURN ROUND(p_distance * v_base_fare * v_class_multiplier * v_passenger_multiplier, 2);
END;
```

2. CheckSeatAvailability

```
CREATE FUNCTION CheckSeatAvailability(
   p_train_id INT,
   p_journey_date DATE,
   p_class_id INT
) RETURNS INT
BEGIN
   DECLARE v_available_seats INT;

   SELECT available_seats INTO v_available_seats
   FROM seat_availability
   WHERE train_id = p_train_id
   AND journey_date = p_journey_date
   AND class_id = p_class_id;
```

```
    RETURN IFNULL(v_available_seats, 0);
END;
```

3. calculate_fare

```
CREATE FUNCTION calculate_fare(
    distance INT,
    class_id INT
) RETURNS DECIMAL(10,2)
BEGIN
    DECLARE v_base_fare DECIMAL(10,2);
    DECLARE v_class_multiplier DECIMAL(10,2);

    -- Get base fare per km
    SET v_base_fare = 2.0;

    -- Get class multiplier
    SELECT multiplier INTO v_class_multiplier
    FROM classes
    WHERE class_id = class_id;

    RETURN ROUND(distance * v_base_fare * v_class_multiplier, 2);
END;
```

4. calculate_refund

```
CREATE FUNCTION calculate_refund(
ticket_id INT,
cancellation_date DATE
) RETURNS DECIMAL(10,2)
BEGIN
DECLARE v_journey_date DATE;
DECLARE v_fare DECIMAL(10,2);
DECLARE v_days_before_journey INT;
DECLARE v_refund_amount DECIMAL(10,2);

-- Get ticket details
SELECT journey_date, fare
INTO v_journey_date, v_fare
FROM tickets
```

```
WHERE ticket_id = ticket_id;

-- Calculate days before journey
SET v_days_before_journey = DATEDIFF(v_journey_date, cancellation_date);

-- Calculate refund based on cancellation rules
IF v_days_before_journey >= 30 THEN
SET v_refund_amount = v_fare * 0.95; -- 95% refund
ELSEIF v_days_before_journey >= 15 THEN
SET v_refund_amount = v_fare * 0.75; -- 75% refund
ELSEIF v_days_before_journey >= 3 THEN
SET v_refund_amount = v_fare * 0.50; -- 50% refund
ELSE
SET v_refund_amount = 0; -- No refund
END IF;

RETURN v_refund_amount;
END;
```

5. get_next_running_dates

```
CREATE FUNCTION get_next_running_dates(
  train_id INT,
  start_date DATE
) RETURNS VARCHAR(255)
BEGIN
  DECLARE v_dates VARCHAR(255);

  SELECT GROUP_CONCAT(DATE_FORMAT(running_date, '%Y-%m-%d') SEPARATOR ', ')
  INTO v_dates
  FROM running_dates
  WHERE running_date >= start_date
  AND running_date <= DATE_ADD(start_date, INTERVAL 7 DAY);

  RETURN IFNULL(v_dates, '');
END;
```

6. get_rac_number

```
CREATE FUNCTION get_rac_number(
  train_id INT,
  journey_date DATE
) RETURNS INT
BEGIN
  DECLARE v_next_rac INT;

  SELECT IFNULL(MAX(rac_number), 0) + 1 INTO v_next_rac
  FROM tickets
  WHERE train_id = train_id
  AND journey_date = journey_date
  AND status = 'RAC';

  RETURN v_next_rac;
END;
```

7. get_waitlist_number

```
CREATE FUNCTION get_waitlist_number(
    train_id INT,
    journey_date DATE
) RETURNS INT
BEGIN
    DECLARE v_next_waitlist INT;

    SELECT IFNULL(MAX(waitlist_number), 0) + 1 INTO v_next_waitlist
    FROM tickets
    WHERE train_id = train_id
    AND journey_date = journey_date
    AND status = 'WL';

    RETURN v_next_waitlist;
END;
```

Triggers

1. update_seat_status_after_ticket_insert

```
CREATE TRIGGER update_seat_status_after_ticket_insert
AFTER INSERT ON tickets
FOR EACH ROW
BEGIN
    IF NEW.status = 'CNF' THEN
```

```
        UPDATE seat_availability
        SET available_seats = available_seats - 1
        WHERE train_id = NEW.train_id
        AND journey_date = NEW.journey_date
        AND class_id = NEW.class_id;
    ELSEIF NEW.status = 'RAC' THEN
        UPDATE seat_availability
        SET available_rac = available_rac - 1
        WHERE train_id = NEW.train_id
        AND journey_date = NEW.journey_date
        AND class_id = NEW.class_id;
    ELSEIF NEW.status = 'WL' THEN
        UPDATE seat_availability
        SET waitlist_count = waitlist_count + 1
        WHERE train_id = NEW.train_id
        AND journey_date = NEW.journey_date
        AND class_id = NEW.class_id;
    END IF;
END;
```

2. after_ticket_insert

```
CREATE TRIGGER after_ticket_insert
AFTER INSERT ON tickets
FOR EACH ROW
BEGIN
    -- Update booking history
    INSERT INTO booking_history (
        ticket_id,
        status,
        status_date
    ) VALUES (
        NEW.ticket_id,

        NEW.status,

        CURDATE()

    );


    -- Update passenger booking count
```

```
   UPDATE passengers

   SET total_bookings = total_bookings + 1

   WHERE passenger_id = NEW.passenger_id;

END;
```

3. handleCancellation

```
CREATE TRIGGER handleCancellation
AFTER UPDATE ON tickets
FOR EACH ROW
BEGIN
   IF NEW.status = 'CANCELLED' AND OLD.status != 'CANCELLED' THEN
      -- Calculate refund
      SET @refund_amount = calculate_refund(NEW.ticket_id, CURDATE());

      -- Update payment status
      UPDATE payments
      SET status = 'REFUNDED',
         refund_amount = @refund_amount,
         refund_date = CURDATE()
      WHERE ticket_id = NEW.ticket_id;

      -- Update seat availability
      IF OLD.status = 'CNF' THEN
         UPDATE seat_availability
         SET available_seats = available_seats + 1
         WHERE train_id = NEW.train_id
         AND journey_date = NEW.journey_date
         AND class_id = NEW.class_id;
      ELSEIF OLD.status = 'RAC' THEN
         UPDATE seat_availability
         SET available_rac = available_rac + 1
         WHERE train_id = NEW.train_id
         AND journey_date = NEW.journey_date
         AND class_id = NEW.class_id;
      END IF;
   END IF;
END;
```

CS2202: Mini-IRCTC

4. after_ticket_cancel

```
CREATE TRIGGER after_ticket_cancel
AFTER UPDATE ON tickets
FOR EACH ROW
BEGIN
  IF NEW.status = 'CANCELLED' AND OLD.status != 'CANCELLED' THEN
    -- Update booking history
    INSERT INTO booking_history (
      ticket_id,
      status,
      status_date
    ) VALUES (
      NEW.ticket_id,
      'CANCELLED',
      CURDATE()
    );

    -- Promote RAC to CNF if possible
    IF OLD.status = 'RAC' THEN
      UPDATE tickets
      SET status = 'CNF'
      WHERE train_id = NEW.train_id
      AND journey_date = NEW.journey_date
      AND class_id = NEW.class_id
      AND status = 'RAC'
      AND rac_number = (
        SELECT MIN(rac_number)
        FROM tickets
        WHERE train_id = NEW.train_id
        AND journey_date = NEW.journey_date
        AND class_id = NEW.class_id
        AND status = 'RAC'
      );
    END IF;
  END IF;
END;
```

5. after_ticket_cancellation

CS2202: Mini-IRCTC

```
CREATE TRIGGER after_ticket_cancellation
AFTER UPDATE ON tickets
FOR EACH ROW
BEGIN
  IF NEW.status = 'CANCELLED' AND OLD.status != 'CANCELLED' THEN
    -- Insert cancellation record
    INSERT INTO cancellations (
      ticket_id,
      cancellation_date,
      reason,
      refund_amount
    ) VALUES (
      NEW.ticket_id,
      CURDATE(),
      'Self Cancellation',
      calculate_refund(NEW.ticket_id, CURDATE())
    );

    -- Update passenger cancellation count
    UPDATE passengers
    SET total_cancellations = total_cancellations + 1
    WHERE passenger_id = NEW.passenger_id;
  END IF;
END;
```

6. UpdateRACStatus

```
CREATE TRIGGER UpdateRACStatus
AFTER UPDATE ON seatavailability
FOR EACH ROW
BEGIN
  IF NEW.available_seats > 0 AND OLD.available_seats = 0 THEN
    -- Promote RAC to CNF
    UPDATE tickets
    SET status = 'CNF'
    WHERE train_id = NEW.train_id
    AND journey_date = NEW.journey_date
    AND class_id = NEW.class_id
    AND status = 'RAC'
    AND rac_number = (
      SELECT MIN(rac_number)
```

CS2202: Mini-IRCTC

```
        FROM tickets
        WHERE train_id = NEW.train_id
        AND journey_date = NEW.journey_date
        AND class_id = NEW.class_id
        AND status = 'RAC'
      );

      -- Update seat availability
      UPDATE seat_availability
      SET available_seats = available_seats - 1,
        available_rac = available_rac + 1
      WHERE train_id = NEW.train_id
      AND journey_date = NEW.journey_date
      AND class_id = NEW.class_id;
    END IF;
END;
```