Smart-Rail

Mini-Project CS2202

Vivek Kumar-2301cs63

Krishna Gupta-2301cs26

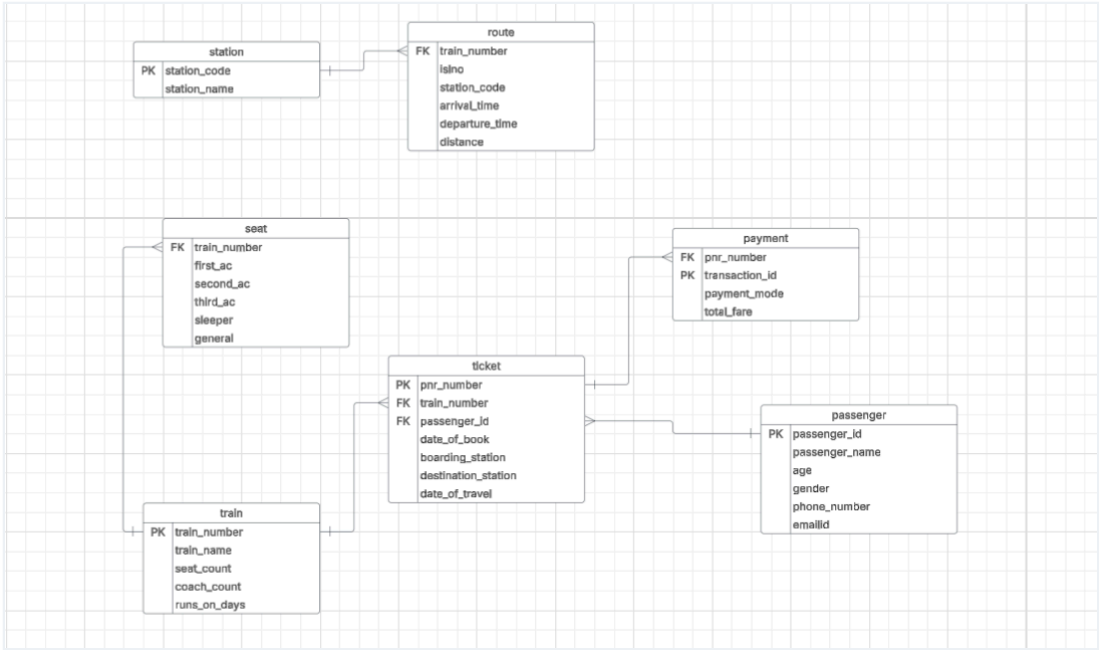
Aditya Aryan-2301mc58

Kumar Sarvagya-2301cs27

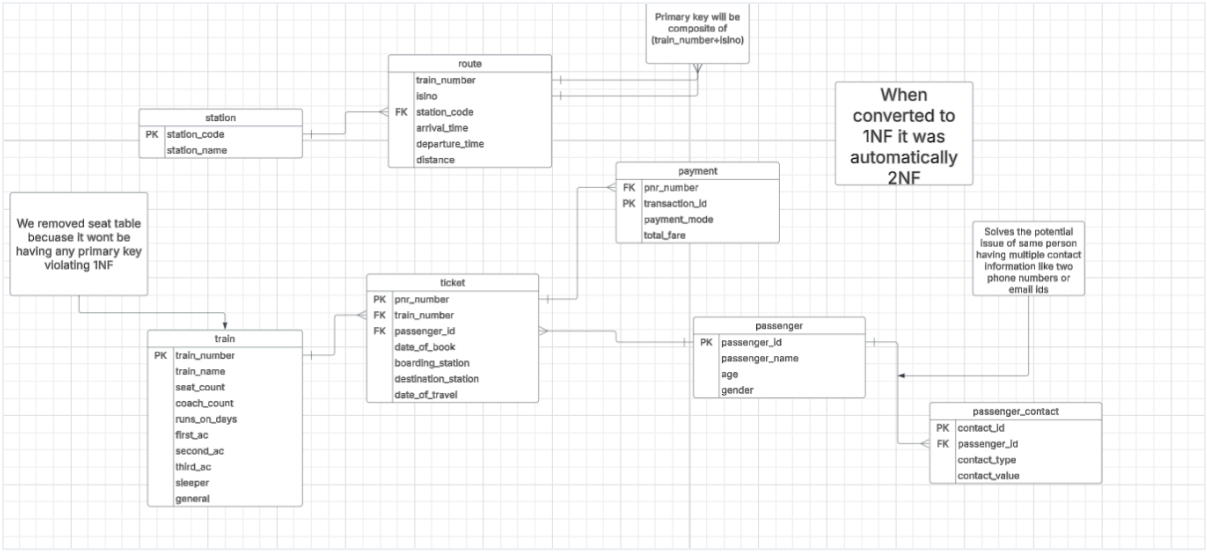
Supporting Files

- CSV DATA FILES:
  - Route Data
  - Station Data
  - Train Data
- SQL DUMP FILE
- DEMO LINK

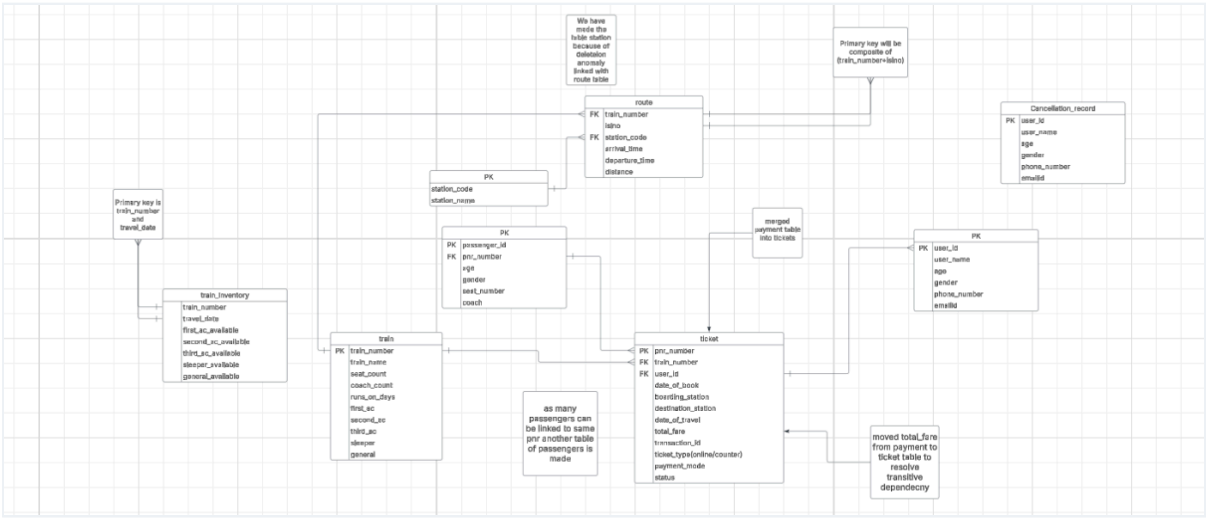
ER DIAGRAM AND SCHEMA



Initial design



1ST AND 2ND NF



3RD AND BCNF

Table description **Link for Source code : LINK**

```
// showing all the tables

mysql> desc train;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| train_number | varchar(10) | NO   | PRI | NULL    |       |
| train_name   | varchar(50) | NO   |     | NULL    |       |
| seat_count   | int         | NO   |     | NULL    |       |
| coach_count  | int         | NO   |     | NULL    |       |
| runs_on_days | varchar(20) | NO   |     | NULL    |       |
| first_ac     | int         | YES  |     | 0       |       |
| second_ac    | int         | YES  |     | 0       |       |
| third_ac     | int         | YES  |     | 0       |       |
| sleeper      | int         | YES  |     | 0       |       |
| general      | int         | YES  |     | 0       |       |
+--------------+-------------+------+-----+---------+-------+
```

```
select count(*) from train;
+----------+
| count(*) |
+----------+
|     2810 |
+----------+

mysql> desc station;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| station_code | varchar(10) | NO   | PRI | NULL    |       |
| station_name | varchar(50) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+

mysql> select count(*) from station;
+----------+
| count(*) |
+----------+
|     4345 |
+----------+
mysql> desc passenger;
+----------------+-------------+------+-----+---------+----------------+
| Field          | Type        | Null | Key | Default | Extra          |
+----------------+-------------+------+-----+---------+----------------+
| passenger_id   | int         | NO   | PRI | NULL    | auto_increment |
| passenger_name | varchar(50) | YES  |     | NULL    |                |
| pnr_number     | varchar(20) | NO   | MUL | NULL    |                |
| age            | int         | YES  |     | NULL    |                |
| gender         | char(1)     | YES  |     | NULL    |                |
| seat_number    | varchar(5)  | YES  |     | NULL    |                |
| coach          | varchar(5)  | YES  |     | NULL    |                |
+----------------+-------------+------+-----+---------+----------------

select count(*) from passenger;
+----------+
| count(*) |
+----------+
|       27 |
+----------+

mysql> desc route;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| train_number   | varchar(10) | NO   | PRI | NULL    |       |
| islno          | int         | NO   | PRI | NULL    |       |
| station_code   | varchar(10) | NO   | MUL | NULL    |       |
| arrival_time   | time        | NO   |     | NULL    |       |
| departure_time | time        | NO   |     | NULL    |       |
| distance       | int         | NO   |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
```

```
select count(*) from route;
+----------+
| count(*) |
+----------+
|    68089 |
+----------+

mysql> desc ticket;
+---------------------+-----------------------------------------------------------------+------+-----+---------+-------+
| Field               | Type                                                            | Null | Key | Default | Extra |
+---------------------+-----------------------------------------------------------------+------+-----+---------+-------+
| pnr_number          | varchar(20)                                                     | NO   | PRI | NULL    |       |
| train_number        | varchar(10)                                                     | NO   | MUL | NULL    |       |
| user_id             | int                                                             | NO   |     | NULL    |       |
| date_of_book        | date                                                            | NO   |     | NULL    |       |
| boarding_station    | varchar(50)                                                     | NO   |     | NULL    |       |
| destination_station | varchar(50)                                                     | NO   |     | NULL    |       |
| date_of_travel      | date                                                            | NO   |     | NULL    |       |
| no_of_passenger     | int                                                             | YES  |     | 0       |       |
| ticket_class        | enum('first_ac','second_ac','third_ac','sleeper','general')     | NO   |     | NULL    |       |
| total_fare          | decimal(10,2)                                                   | NO   |     | NULL    |       |
| transaction_id      | varchar(50)                                                     | YES  | UNI | NULL    |       |
| ticket_type         | enum('online','counter')                                        | NO   |     | NULL    |       |
| payment_mode        | varchar(20)                                                     | NO   |     | NULL    |       |
| status              | varchar(20)                                                     | NO   |     | NULL    |       |
+---------------------+-----------------------------------------------------------------+------+-----+---------+-------+

select count(*) from ticket;
+----------+
| count(*) |
+----------+
|       51 |
+----------+
```

```
mysql> desc train_inventory;
+----------------------+-------------+------+-----+---------+-------+
| Field                | Type        | Null | Key | Default | Extra |
+----------------------+-------------+------+-----+---------+-------+
| train_number         | varchar(10) | NO   | PRI | NULL    |       |
| travel_date          | date        | NO   | PRI | NULL    |       |
| first_ac_available   | int         | YES  |     | 0       |       |
| second_ac_available  | int         | YES  |     | 0       |       |
| third_ac_available   | int         | YES  |     | 0       |       |
| sleeper_available    | int         | YES  |     | 0       |       |
| general_available    | int         | YES  |     | 0       |       |
+----------------------+-------------+------+-----+---------+-------+

select count(*) from train_inventory;
+----------+
| count(*) |
+----------+
|    16120 |
+----------+

mysql> desc cancellation_record;
+----------------------+-----------------------------------------------------------------------+------+-----+-------------------+-------------------+
| Field                | Type                                                                  | Null | Key | Default           | Extra             |
+----------------------+-----------------------------------------------------------------------+------+-----+-------------------+-------------------+
| id                   | int                                                                   | NO   | PRI | NULL              | auto_increment    |
| pnr_number           | varchar(20)                                                           | NO   |     | NULL              |                   |
| train_number         | varchar(10)                                                           | NO   |     | NULL              |                   |
| date_of_travel       | date                                                                  | NO   |     | NULL              |                   |
| boarding_station     | varchar(50)                                                           | NO   |     | NULL              |                   |
| destination_station  | varchar(50)                                                           | NO   |     | NULL              |                   |
| ticket_class         | enum('first_ac','second_ac','third_ac','sleeper','general')           | NO   |     | NULL              |                   |
| no_of_passenger      | int                                                                   | NO   |     | NULL              |                   |
| status               | varchar(20)                                                           | NO   |     | NULL              |                   |
| refund_amount        | decimal(10,2)                                                         | NO   |     | NULL              |                   |
| refund_status        | enum('PENDING','DONE','FAILED')                                       | YES  |     | PENDING           |                   |
| cancelled_at         | timestamp                                                             | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
+----------------------+-----------------------------------------------------------------------+------+-----+-------------------+-------------------+
```

```
 select count(*) from cancellation_record;
+----------+
| count(*) |
+----------+
|       57 |
+----------+

mysql> desc user;
+--------------+-------------------+------+-----+---------+----------------+
| Field        | Type              | Null | Key | Default | Extra          |
+--------------+-------------------+------+-----+---------+----------------+
| user_id      | int               | NO   | PRI | NULL    | auto_increment |
| user_name    | varchar(50)       | NO   |     | NULL    |                |
| age          | int               | YES  |     | NULL    |                |
| gender       | enum('M','F','O') | NO   |     | NULL    |                |
| phone_number | varchar(15)       | NO   | UNI | NULL    |                |
| email_id     | varchar(50)       | NO   | UNI | NULL    |                |
+--------------+-------------------+------+-----+---------+----------------+

select count(*) from user;
+----------+
| count(*) |
+----------+
|       12 |
+----------+
```

SQL queries,procedures,functions,triggers **Link for Source code : LINK**

```sql
--this is for searching train with source and destination station and date

DELIMITER $$

CREATE PROCEDURE search_train_by_station2(
    IN partial_SRC VARCHAR(50),
    IN partial_DEST VARCHAR(50),
    IN travel_date DATE
)
BEGIN
    DECLARE SRC_CODE VARCHAR(10);
    DECLARE DEST_CODE VARCHAR(10);

    -- Fetch station codes based on partial station names
    SELECT station_code INTO SRC_CODE FROM station WHERE station_name LIKE CONCAT( partial_SRC, '%') LIMIT 1;
    SELECT station_code INTO DEST_CODE FROM station WHERE station_name LIKE CONCAT( partial_DEST, '%') LIMIT 1;

    -- Fetch train details
    SELECT DISTINCT
```

```
        t.train_number,
        t.train_name,
        r1.station_code AS source_code,
        (SELECT station_name FROM station WHERE station_code = r1.station_code) AS source_name,
        r2.station_code AS destination_code,
        (SELECT station_name FROM station WHERE station_code = r2.station_code) AS destination_name,
        r1.departure_time AS departure_time,
        r2.arrival_time AS arrival_time,
        r2.distance - r1.distance AS distance,
        CASE
            WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(travel_date, '%a')), '%')
                THEN travel_date
            WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 1 DAY), '%a')), '%')
                THEN DATE_ADD(travel_date, INTERVAL 1 DAY)
            WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 2 DAY), '%a')), '%')
                THEN DATE_ADD(travel_date, INTERVAL 2 DAY)
        END AS actual_travel_date
    FROM route AS r1
    JOIN route AS r2 ON r1.train_number = r2.train_number
    JOIN train AS t ON t.train_number = r1.train_number
    WHERE r1.station_code = SRC_CODE
      AND r2.station_code = DEST_CODE
      AND r1.islno < r2.islno
      AND (
          LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(travel_date, '%a')), '%')
          OR LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 1 DAY), '%a')), '%')
          OR LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 2 DAY), '%a')), '%')
      )
      ORDER BY actual_travel_date;
END $$

--this is stored procedure for updating train_inventory table automatically

CREATE PROCEDURE update_train_inventory()
BEGIN
    DECLARE new_date DATE;
    -- Define new_date as tomorrow
    SET new_date = CURDATE() + INTERVAL 30 DAY;

    -- Insert a new inventory record for new_date for all trains that run on that day
    INSERT IGNORE INTO train_inventory (
        train_number, travel_date,
        first_ac_available, second_ac_available, third_ac_available, sleeper_available, general_available)
    SELECT
        t.train_number, new_date,
```

```
        t.first_ac,
        t.second_ac,
        t.third_ac,
        t.sleeper,
        t.general
    FROM train t
    -- Check if the train runs on new_date, assuming runs_on_days stores concatenated day abb
reviations (e.g., 'MonTueWed')
    WHERE LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(new_date, '%a')), '%');

    -- Delete any inventory records for train which is departed
    DELETE FROM train_inventory
    WHERE travel_date < CURDATE();
END $$

--CREATING EVENT TO UPDATE INVENTORY DAILY
DELIMITER $$
CREATE EVENT update_train_inventory_daily
ON SCHEDULE EVERY 1 DAY
DO
BEGIN
    CALL update_train_inventory();
END $$

--for updating seat count while booking
DELIMITER $$

CREATE PROCEDURE update_seat(
    IN in_train_number VARCHAR(10),
    IN in_date DATE,
    IN in_passenger_count INT,
    IN in_class VARCHAR(20)
)
BEGIN
    DECLARE current_seats INT DEFAULT 0;

    IF in_class = 'first_ac' THEN
        SELECT first_ac_available INTO current_seats
        FROM train_inventory
        WHERE train_number = in_train_number AND travel_date = in_date
        FOR UPDATE;

        IF current_seats < in_passenger_count THEN
            SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Not enough seats available in first_ac.';
        ELSE
            UPDATE train_inventory
            SET first_ac_available = first_ac_available - in_passenger_count
            WHERE train_number = in_train_number AND travel_date = in_date;
        END IF;

    ELSEIF in_class = 'second_ac' THEN
        SELECT second_ac_available INTO current_seats
```

```
        FROM train_inventory
        WHERE train_number = in_train_number AND travel_date = in_date
        FOR UPDATE;

        IF current_seats < in_passenger_count THEN
            SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Not enough seats available in second_ac.';
        ELSE
            UPDATE train_inventory
            SET second_ac_available = second_ac_available - in_passenger_count
            WHERE train_number = in_train_number AND travel_date = in_date;
        END IF;

    ELSEIF in_class = 'third_ac' THEN
        SELECT third_ac_available INTO current_seats
        FROM train_inventory
        WHERE train_number = in_train_number AND travel_date = in_date
        FOR UPDATE;

        IF current_seats < in_passenger_count THEN
            SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Not enough seats available in third_ac.';
        ELSE
            UPDATE train_inventory
            SET third_ac_available = third_ac_available - in_passenger_count
            WHERE train_number = in_train_number AND travel_date = in_date;
        END IF;

    ELSEIF in_class = 'sleeper' THEN
        SELECT sleeper_available INTO current_seats
        FROM train_inventory
        WHERE train_number = in_train_number AND travel_date = in_date
        FOR UPDATE;

        IF current_seats < in_passenger_count THEN
            SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Not enough seats available in sleeper.';
        ELSE
            UPDATE train_inventory
            SET sleeper_available = sleeper_available - in_passenger_count
            WHERE train_number = in_train_number AND travel_date = in_date;
        END IF;

    ELSEIF in_class = 'general' THEN
        SELECT general_available INTO current_seats
        FROM train_inventory
        WHERE train_number = in_train_number AND travel_date = in_date
        FOR UPDATE;

        IF current_seats < in_passenger_count THEN
            SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Not enough seats available in general.';
        ELSE
```

```
            UPDATE train_inventory
            SET general_available = general_available - in_passenger_count
            WHERE train_number = in_train_number AND travel_date = in_date;
        END IF;

    ELSE
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Invalid ticket class specified.';
    END IF;

END $$

-- this is for checking pnr_status
DELIMITER $$
CREATE PROCEDURE check_pnr_status(IN in_pnr_number VARCHAR(20))
BEGIN
    SELECT train_number,date_of_travel,boarding_station,destination_station,ticket_class,no_of_passenger,status FROM ticket WHERE pnr_number = in_pnr_number;
END $$

DELIMITER ;

DELIMITER $$
CREATE PROCEDURE train_shedule_lookup(IN in_train_number VARCHAR(10))
BEGIN
    SELECT train_number,train_name,runs_on_days FROM train WHERE train_number = in_train_number;
    SELECT islno,station_code,arrival_time,departure_time,distance FROM route WHERE train_number = in_train_number;
END $$

DELIMITER ;

-- fror cheking seat availability
DELIMITER $$
CREATE PROCEDURE check_seat_availability(IN in_train_number VARCHAR(10),IN in_date DATE,IN in_class VARCHAR(10))
BEGIN
    DECLARE available_seats INT;

    if(in_class = 'first_ac') then
        SELECT first_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
    elseif(in_class = 'second_ac') then
        SELECT second_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
    elseif(in_class = 'third_ac') then
        SELECT third_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
    elseif(in_class = 'sleeper') then
        SELECT sleeper_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
    elseif(in_class = 'general') then
```

```
        SELECT general_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
    end if;
    SELECT available_seats;

END $$

-- for getting list of passenger travelling on a train on specific date
DELIMITER $$
CREATE PROCEDURE list_passengers_on_train(IN in_train_number VARCHAR(10), IN in_date DATE,IN in_status VARCHAR(20))
BEGIN
    if(in_status = 'all') then
        SELECT ticket.pnr_number,passenger_name,ticket_class,age,gender,seat_number,coach,status FROM ticket,passenger WHERE ticket.train_number = in_train_number AND ticket.date_of_travel = in_date AND ticket.pnr_number=passenger.pnr_number;
    else
        SELECT ticket.pnr_number,passenger_name,ticket_class,age,gender,seat_number,coach,status FROM ticket,passenger WHERE ticket.train_number = in_train_number AND ticket.date_of_travel = in_date AND ticket.pnr_number=passenger.pnr_number AND ticket.status = in_status;
    end if;
END $$

DELIMITER $$

CREATE PROCEDURE get_total_revenue(
    IN start_date DATE,
    IN end_date DATE
)
BEGIN
    SELECT
        SUM(total_fare) AS total_revenue
    FROM
        ticket
    WHERE
        date_of_book BETWEEN start_date AND end_date
        AND status = 'CONFIRMED';
END $$

DELIMITER ;

-- for cancelling the ticket
CREATE PROCEDURE cancel_ticket(IN in_pnr_number VARCHAR(20))
BEGIN
    DECLARE t_train_number VARCHAR(10);
    DECLARE t_travel_date DATE;
    DECLARE t_ticket_class VARCHAR(10);
    DECLARE t_no_of_passenger INT;

    -- Step 1: Get ticket details
    SELECT train_number, date_of_travel, ticket_class, no_of_passenger
    INTO t_train_number, t_travel_date, t_ticket_class, t_no_of_passenger
    FROM ticket
```

```
    WHERE pnr_number = in_pnr_number;

    -- Step 2: Delete passenger records
    DELETE FROM passenger WHERE pnr_number = in_pnr_number;

    -- Step 3: Delete ticket record
    DELETE FROM ticket WHERE pnr_number = in_pnr_number;

    -- Step 4: Update seat availability in train_inventory
    IF t_ticket_class = 'first_ac' THEN
        UPDATE train_inventory
        SET first_ac_available = first_ac_available + t_no_of_passenger
        WHERE train_number = t_train_number AND travel_date = t_travel_date;
    ELSEIF t_ticket_class = 'second_ac' THEN
        UPDATE train_inventory
        SET second_ac_available = second_ac_available + t_no_of_passenger
        WHERE train_number = t_train_number AND travel_date = t_travel_date;
    ELSEIF t_ticket_class = 'third_ac' THEN
        UPDATE train_inventory
        SET third_ac_available = third_ac_available + t_no_of_passenger
        WHERE train_number = t_train_number AND travel_date = t_travel_date;
    ELSEIF t_ticket_class = 'sleeper' THEN
        UPDATE train_inventory
        SET sleeper_available = sleeper_available + t_no_of_passenger
        WHERE train_number = t_train_number AND travel_date = t_travel_date;
    ELSEIF t_ticket_class = 'general' THEN
        UPDATE train_inventory
        SET general_available = general_available + t_no_of_passenger
        WHERE train_number = t_train_number AND travel_date = t_travel_date;
    END IF;

END $$

-------------------------------------------------------------------------------------------------------
-- listing all triggers
--triggers while booking on table ticket
DELIMITER $$

CREATE TRIGGER before_ticket_insert
BEFORE INSERT ON ticket
FOR EACH ROW
BEGIN
    DECLARE avail INT DEFAULT 0;

    -- Check based on the booked class:
    IF NEW.ticket_class = 'first_ac' THEN
        SELECT first_ac_available INTO avail
        FROM train_inventory
        WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'second_ac' THEN
        SELECT second_ac_available INTO avail
```

```
        FROM train_inventory
        WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'third_ac' THEN
        SELECT third_ac_available INTO avail
        FROM train_inventory
        WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'sleeper' THEN
        SELECT sleeper_available INTO avail
        FROM train_inventory
        WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'general' THEN
        SELECT general_available INTO avail
        FROM train_inventory
        WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;

    ELSE
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Invalid seat class specified.';
    END IF;

    IF avail IS NULL OR avail <= 0 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'No available seats for the chosen class on this date.';
    END IF;
END $$

DELIMITER ;

--after insert trigger
DELIMITER $$

CREATE TRIGGER after_ticket_insert
AFTER INSERT ON ticket
FOR EACH ROW
BEGIN
    IF NEW.ticket_class = 'first_ac' THEN
        UPDATE train_inventory
        SET first_ac_available = first_ac_available - 1
        WHERE train_number = NEW.train_number
          AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'second_ac' THEN
        UPDATE train_inventory
        SET second_ac_available = second_ac_available - 1
        WHERE train_number = NEW.train_number
          AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'third_ac' THEN
        UPDATE train_inventory
        SET third_ac_available = third_ac_available - 1
```

```
        WHERE train_number = NEW.train_number
          AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'sleeper' THEN
        UPDATE train_inventory
        SET sleeper_available = sleeper_available - 1
        WHERE train_number = NEW.train_number
          AND travel_date = NEW.date_of_travel;

    ELSEIF NEW.ticket_class = 'general' THEN
        UPDATE train_inventory
        SET general_available = general_available - 1
        WHERE train_number = NEW.train_number
          AND travel_date = NEW.date_of_travel;
    END IF;
END $$

DELIMITER ;

-- trigger while cancelling the ticket
DELIMITER $$
CREATE TRIGGER update_cancellation_record
AFTER DELETE ON ticket
FOR EACH ROW
BEGIN
    INSERT INTO cancellation_record(pnr_number,train_number,date_of_travel,boarding_station,destination_station,ticket_class,no_of_passenger,status,refund_amount,refund_status)
    VALUES(OLD.pnr_number,OLD.train_number,OLD.date_of_travel,OLD.boarding_station,OLD.destination_station,OLD.ticket_class,OLD.no_of_passenger,OLD.status,OLD.total_fare,'DONE');
END $$

DELIMITER ;

DELIMITER $$

CREATE TRIGGER update_cancellation_record
AFTER DELETE ON ticket
FOR EACH ROW
BEGIN
    DECLARE base_deduction DECIMAL(10,2);
    DECLARE extra_deduction DECIMAL(10,2);
    DECLARE total_deduction DECIMAL(10,2);
    DECLARE refund_amount DECIMAL(10,2);
    DECLARE days_left INT;

    -- Calculate days between current date and travel date
    SET days_left = DATEDIFF(OLD.date_of_travel, CURDATE());

    -- Base deduction: 10% cancellation charge
    SET base_deduction = OLD.total_fare * 0.10;

    -- Extra deduction based on how close the travel date is
    IF days_left > 7 THEN
```

```
        SET extra_deduction = 0;
    ELSEIF days_left BETWEEN 4 AND 7 THEN
        SET extra_deduction = OLD.total_fare * 0.10;
    ELSEIF days_left BETWEEN 1 AND 3 THEN
        SET extra_deduction = OLD.total_fare * 0.20;
    ELSE
        SET extra_deduction = OLD.total_fare * 0.30;
    END IF;

    -- Total deduction
    SET total_deduction = base_deduction + extra_deduction;

    -- Final refund amount
    SET refund_amount = OLD.total_fare - total_deduction;

    -- Insert into cancellation record
    INSERT INTO cancellation_record (
        pnr_number, train_number, date_of_travel, boarding_station, destination_station,
        ticket_class, no_of_passenger, status, refund_amount, refund_status
    )
    VALUES (
        OLD.pnr_number, OLD.train_number, OLD.date_of_travel, OLD.boarding_station,
        OLD.destination_station, OLD.ticket_class, OLD.no_of_passenger, OLD.status,
        refund_amount, 'DONE'
    );
END $$

DELIMITER ;
```

Other interesting queries beyond the required ones

```
1. Find trains with maximum waitlisted passengers in a day
SELECT train_number, travel_date, COUNT(*) AS waitlist_count
FROM ticket
WHERE status = 'Waitlisted'
GROUP BY train_number, travel_date
ORDER BY waitlist_count DESC
LIMIT 1;


2. Average occupancy per class for a train
SELECT train_number, class,
       ROUND(SUM(CASE WHEN status = 'Confirmed' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) AS
occupancy_percentage
FROM ticket
GROUP BY train_number, class;


3. Peak booking hours (hourly ticket distribution)
SELECT HOUR(booking_time) AS booking_hour, COUNT(*) AS tickets_booked
FROM ticket
GROUP BY booking_hour
ORDER BY tickets_booked DESC;
```

```sql


4. Top 5 most frequently used train numbers
SELECT train_number, COUNT(*) AS usage_count
FROM ticket
GROUP BY train_number
ORDER BY usage_count DESC
LIMIT 5;
```