

Smart-Rail

Mini-Project CS2202

Vivek Kumar-2301cs63

Krishna Gupta-2301cs26

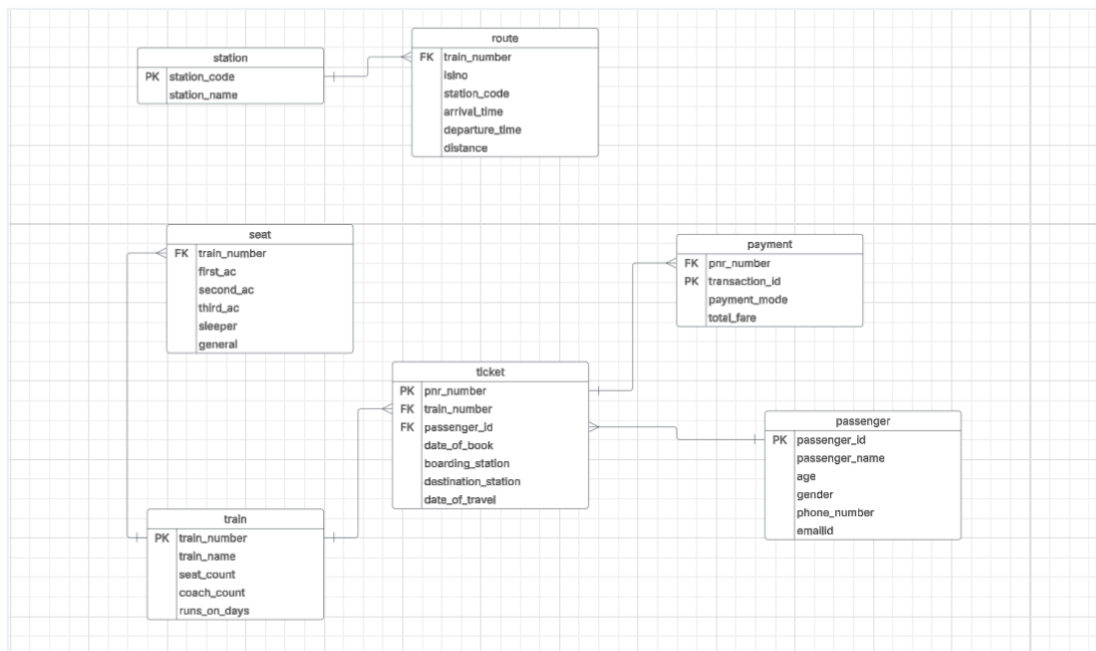
Aditya Aryan-2301mc58

Kumar Sarvagya-2301cs27

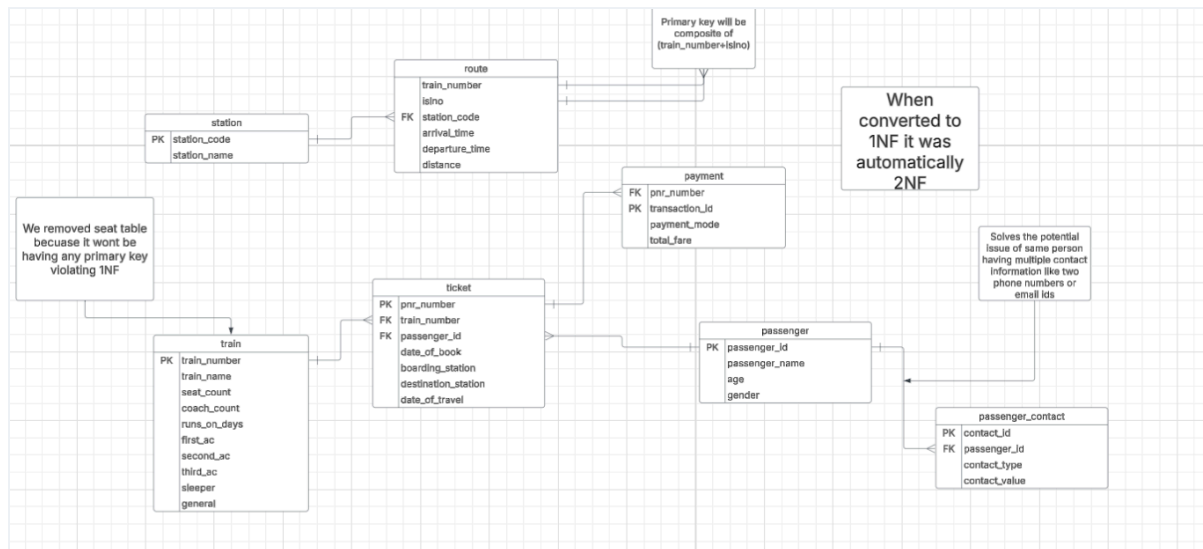
Supporting Files

- CSV DATA FILES:
 - [Route Data](#)
 - [Station Data](#)
 - [Train Data](#)
- [SQL DUMP FILE](#)
- [DEMO LINK](#)

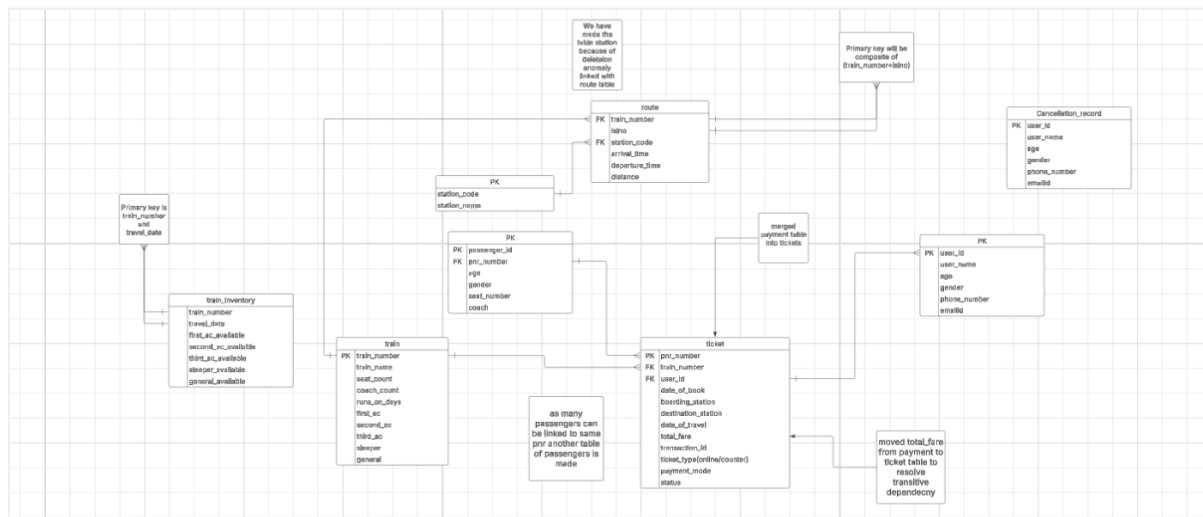
ER DIAGRAM AND SCHEMA



Initial design



1ST AND 2ND NF



3RD AND BCNF

Table description

Link for Source code : [LINK](#)

```
1 // showing all the tables
```

```
2
```

```
3 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	

```

18
19 select count(*) from train;
20 +-----+
21 | count(*) |
22 +-----+
23 |      2810 |
24 +-----+
25
26 mysql> desc station;
27 +-----+-----+-----+-----+-----+-----+
28 | Field          | Type          | Null | Key | Default | Extra |
29 +-----+-----+-----+-----+-----+-----+
30 | station_code   | varchar(10)   | NO   | PRI | NULL    |       |
31 | station_name   | varchar(50)   | YES  |     | NULL    |       |
32 +-----+-----+-----+-----+-----+-----+
33
34 mysql> select count(*) from station;
35 +-----+
36 | count(*) |
37 +-----+
38 |      4345 |
39 +-----+
40 mysql> desc passenger;
41 +-----+-----+-----+-----+-----+-----+
42 | Field          | Type          | Null | Key | Default | Extra |
43 +-----+-----+-----+-----+-----+-----+
44 | passenger_id   | int           | NO   | PRI | NULL    | auto_increment |
45 | passenger_name | varchar(50)   | YES  |     | NULL    |       |
46 | pnr_number     | varchar(20)   | NO   | MUL | NULL    |       |
47 | age            | int           | YES  |     | NULL    |       |
48 | gender          | char(1)       | YES  |     | NULL    |       |
49 | seat_number    | varchar(5)    | YES  |     | NULL    |       |
50 | coach          | varchar(5)    | YES  |     | NULL    |       |
51 +-----+-----+-----+-----+-----+-----+
52
53 select count(*) from passenger;
54 +-----+
55 | count(*) |
56 +-----+
57 |        27 |
58 +-----+
59
60 mysql> desc route;
61 +-----+-----+-----+-----+-----+-----+
62 | Field          | Type          | Null | Key | Default | Extra |
63 +-----+-----+-----+-----+-----+-----+
64 | train_number   | varchar(10)   | NO   | PRI | NULL    |       |
65 | islno          | int           | NO   | PRI | NULL    |       |
66 | station_code   | varchar(10)   | NO   | MUL | NULL    |       |
67 | arrival_time   | time          | NO   |     | NULL    |       |
68 | departure_time | time          | NO   |     | NULL    |       |
69 | distance       | int           | NO   |     | NULL    |       |
70 +-----+-----+-----+-----+-----+-----+

```

```

71
72 select count(*) from route;
73 +-----+
74 | count(*) |
75 +-----+
76 |    68089 |
77 +-----+
78
79 mysql> desc ticket;
80 +-----+-----+-----+-----+-----+-----+-----+-----+-----+
81 | Field | Type | Null | Key | Default | Extra |
82 +-----+-----+-----+-----+-----+-----+-----+-----+-----+
83 | pnr_number | varchar(20) | NO | | | |
84 | PRI | NULL | | | | |
85 | train_number | varchar(10) | NO | | | |
86 | MUL | NULL | | | | |
87 | user_id | int | NO | | | |
88 | NULL | | | | | |
89 | date_of_book | date | NO | | | |
90 | NULL | | | | | |
91 | boarding_station | varchar(50) | NO | | | |
92 | NULL | | | | | |
93 | destination_station | varchar(50) | NO | | | |
94 | NULL | | | | | |
95 | date_of_travel | date | NO | | | |
96 | NULL | | | | | |
97 | no_of_passenger | int | YES | | | |
98 | 0 | | | | | |
99 | ticket_class | enum('first_ac','second_ac','third_ac','sleeper','general') | NO | | | |
100 | NULL | | | | | |
101 | total_fare | decimal(10,2) | NO | | | |
102 | NULL | | | | | |
103 | transaction_id | varchar(50) | YES | | | |
104 | UNI | NULL | | | | |
105 | ticket_type | enum('online','counter') | NO | | | |
106 | NULL | | | | | |
107 | payment_mode | varchar(20) | NO | | | |
108 | NULL | | | | | |
109 | status | varchar(20) | NO | | | |
110 | NULL | | | | | |
111 +-----+-----+-----+-----+-----+-----+-----+-----+-----+
112 +-----+
113 | count(*) |
114 +-----+
115 |    51 |
116 +-----+

```

```

106 mysql> desc train_inventory;
107 +-----+-----+-----+-----+-----+-----+
108 | Field          | Type          | Null | Key | Default | Extra |
109 +-----+-----+-----+-----+-----+-----+
110 | train_number    | varchar(10)    | NO   | PRI | NULL    |       |
111 | travel_date     | date           | NO   | PRI | NULL    |       |
112 | first_ac_available | int           | YES  |     | 0       |       |
113 | second_ac_available | int           | YES  |     | 0       |       |
114 | third_ac_available | int           | YES  |     | 0       |       |
115 | sleeper_available | int           | YES  |     | 0       |       |
116 | general_available | int           | YES  |     | 0       |       |
117 +-----+-----+-----+-----+-----+-----+
118
119 select count(*) from train_inventory;
120 +-----+
121 | count(*) |
122 +-----+
123 |    16120 |
124 +-----+
125
126 mysql> desc cancellation_record;
127 +-----+-----+-----+-----+-----+-----+
128 | Field          | Type          | Null | Key | Default | Extra |
129 +-----+-----+-----+-----+-----+-----+
130 | id             | int           | NO   | PRI | NULL    | auto_increment |
131 | pnr_number     | varchar(20)    | NO   |     | NULL    |               |
132 | train_number    | varchar(10)    | NO   |     | NULL    |               |
133 | date_of_travel  | date           | NO   |     | NULL    |               |
134 | boarding_station | varchar(50)    | NO   |     | NULL    |               |
135 | destination_station | varchar(50)    | NO   |     | NULL    |               |
136 | ticket_class    | enum('first_ac','second_ac','third_ac','sleeper','general') | NO   |     | NULL    |               |
137 | no_of_passenger | int           | NO   |     | NULL    |               |
138 | status         | varchar(20)    | NO   |     | NULL    |               |
139 | refund_amount   | decimal(10,2)  | NO   |     | NULL    |               |
140 | refund_status   | enum('PENDING','DONE','FAILED') | YES  |     | PENDING |               |
141 | cancelled_at    | timestamp      | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
142 +-----+-----+-----+-----+-----+-----+

```

```

143
144 select count(*) from cancellation_record;
145 +-----+
146 | count(*) |
147 +-----+
148 |      57 |
149 +-----+
150
151 mysql> desc user;
152 +-----+-----+-----+-----+-----+-----+
153 | Field      | Type          | Null | Key | Default | Extra          |
154 +-----+-----+-----+-----+-----+-----+
155 | user_id    | int           | NO   | PRI | NULL    | auto_increment |
156 | user_name  | varchar(50)   | NO   |     | NULL    |                |
157 | age        | int           | YES  |     | NULL    |                |
158 | gender      | enum('M','F','O') | NO   |     | NULL    |                |
159 | phone_number | varchar(15)   | NO   | UNI | NULL    |                |
160 | email_id   | varchar(50)   | NO   | UNI | NULL    |                |
161 +-----+-----+-----+-----+-----+-----+
162
163 select count(*) from user;
164 +-----+
165 | count(*) |
166 +-----+
167 |      12 |
168 +-----+

```

T SQL queries,procedures,functions,triggers

Link for Source code : [LINK](#)

```

1  --this is for searching train with source and destination station and date
2
3  DELIMITER $$
4
5  CREATE PROCEDURE search_train_by_station2(
6      IN partial_SRC VARCHAR(50),
7      IN partial_DEST VARCHAR(50),
8      IN travel_date DATE
9  )
10 BEGIN
11     DECLARE SRC_CODE VARCHAR(10);
12     DECLARE DEST_CODE VARCHAR(10);
13
14     -- Fetch station codes based on partial station names
15     SELECT station_code INTO SRC_CODE FROM station WHERE station_name LIKE CONCAT( partial_SRC, '%' ) LIMIT 1;
16     SELECT station_code INTO DEST_CODE FROM station WHERE station_name LIKE CONCAT( partial_DEST, '%' ) LIMIT 1;
17
18     -- Fetch train details
19     SELECT DISTINCT

```

```

20         t.train_number,
21         t.train_name,
22         r1.station_code AS source_code,
23         (SELECT station_name FROM station WHERE station_code = r1.station_code) AS source_name,
24         r2.station_code AS destination_code,
25         (SELECT station_name FROM station WHERE station_code = r2.station_code) AS destination_name,
26         r1.departure_time AS departure_time,
27         r2.arrival_time AS arrival_time,
28         r2.distance - r1.distance AS distance,
29         CASE
30             WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(travel_date, '%a')), '%')
31             THEN travel_date
32             WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 1 DAY), '%a')), '%')
33             THEN DATE_ADD(travel_date, INTERVAL 1 DAY)
34             WHEN LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 2 DAY), '%a')), '%')
35             THEN DATE_ADD(travel_date, INTERVAL 2 DAY)
36         END AS actual_travel_date
37     FROM route AS r1
38     JOIN route AS r2 ON r1.train_number = r2.train_number
39     JOIN train AS t ON t.train_number = r1.train_number
40     WHERE r1.station_code = SRC_CODE
41         AND r2.station_code = DEST_CODE
42         AND r1.islno < r2.islno
43         AND (
44             LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(travel_date, '%a')), '%')
45             OR LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 1 DAY), '%a')), '%')
46             OR LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(DATE_ADD(travel_date, INTERVAL 2 DAY), '%a')), '%')
47         )
48     ORDER BY actual_travel_date;
49 END $$
50
51 --this is stored procedure for updating train_inventory table automatically
52
53 CREATE PROCEDURE update_train_inventory()
54 BEGIN
55     DECLARE new_date DATE;
56     -- Define new_date as tomorrow
57     SET new_date = CURDATE() + INTERVAL 30 DAY;
58
59     -- Insert a new inventory record for new_date for all trains that run on that day
60     INSERT IGNORE INTO train_inventory (
61         train_number, travel_date,
62         first_ac_available, second_ac_available, third_ac_available, sleeper_available, general_available)
63     SELECT
64         t.train_number, new_date,

```

```

65         t.first_ac,
66         t.second_ac,
67         t.third_ac,
68         t.sleeper,
69         t.general
70     FROM train t
71     -- Check if the train runs on new_date, assuming runs_on_days stores concatenated day abbreviations (e.g., 'MonTueWed')
72     WHERE LOWER(t.runs_on_days) LIKE CONCAT('%', LOWER(DATE_FORMAT(new_date, '%a')), '%');
73
74     -- Delete any inventory records for train which is departed
75     DELETE FROM train_inventory
76     WHERE travel_date < CURDATE();
77 END $$
78
79 --CREATING EVENT TO UPDATE INVENTORY DAILY
80 DELIMITER $$
81 CREATE EVENT update_train_inventory_daily
82 ON SCHEDULE EVERY 1 DAY
83 DO
84 BEGIN
85     CALL update_train_inventory();
86 END $$
87
88 --for updating seat count while booking
89 DELIMITER $$
90
91 CREATE PROCEDURE update_seat(
92     IN in_train_number VARCHAR(10),
93     IN in_date DATE,
94     IN in_passenger_count INT,
95     IN in_class VARCHAR(20)
96 )
97 BEGIN
98     DECLARE current_seats INT DEFAULT 0;
99
100     IF in_class = 'first_ac' THEN
101         SELECT first_ac_available INTO current_seats
102         FROM train_inventory
103         WHERE train_number = in_train_number AND travel_date = in_date
104         FOR UPDATE;
105
106         IF current_seats < in_passenger_count THEN
107             SIGNAL SQLSTATE '45000'
108             SET MESSAGE_TEXT = 'Not enough seats available in first_ac.';
109         ELSE
110             UPDATE train_inventory
111             SET first_ac_available = first_ac_available - in_passenger_count
112             WHERE train_number = in_train_number AND travel_date = in_date;
113         END IF;
114
115     ELSEIF in_class = 'second_ac' THEN
116         SELECT second_ac_available INTO current_seats

```



```

117     FROM train_inventory
118     WHERE train_number = in_train_number AND travel_date = in_date
119     FOR UPDATE;
120
121     IF current_seats < in_passenger_count THEN
122         SIGNAL SQLSTATE '45000'
123             SET MESSAGE_TEXT = 'Not enough seats available in second_ac.';
124     ELSE
125         UPDATE train_inventory
126         SET second_ac_available = second_ac_available - in_passenger_count
127         WHERE train_number = in_train_number AND travel_date = in_date;
128     END IF;
129
130 ELSEIF in_class = 'third_ac' THEN
131     SELECT third_ac_available INTO current_seats
132     FROM train_inventory
133     WHERE train_number = in_train_number AND travel_date = in_date
134     FOR UPDATE;
135
136     IF current_seats < in_passenger_count THEN
137         SIGNAL SQLSTATE '45000'
138             SET MESSAGE_TEXT = 'Not enough seats available in third_ac.';
139     ELSE
140         UPDATE train_inventory
141         SET third_ac_available = third_ac_available - in_passenger_count
142         WHERE train_number = in_train_number AND travel_date = in_date;
143     END IF;
144
145 ELSEIF in_class = 'sleeper' THEN
146     SELECT sleeper_available INTO current_seats
147     FROM train_inventory
148     WHERE train_number = in_train_number AND travel_date = in_date
149     FOR UPDATE;
150
151     IF current_seats < in_passenger_count THEN
152         SIGNAL SQLSTATE '45000'
153             SET MESSAGE_TEXT = 'Not enough seats available in sleeper.';
154     ELSE
155         UPDATE train_inventory
156         SET sleeper_available = sleeper_available - in_passenger_count
157         WHERE train_number = in_train_number AND travel_date = in_date;
158     END IF;
159
160 ELSEIF in_class = 'general' THEN
161     SELECT general_available INTO current_seats
162     FROM train_inventory
163     WHERE train_number = in_train_number AND travel_date = in_date
164     FOR UPDATE;
165
166     IF current_seats < in_passenger_count THEN
167         SIGNAL SQLSTATE '45000'
168             SET MESSAGE_TEXT = 'Not enough seats available in general.';
169     ELSE

```

```

170         UPDATE train_inventory
171         SET general_available = general_available - in_passenger_count
172         WHERE train_number = in_train_number AND travel_date = in_date;
173     END IF;
174
175     ELSE
176         SIGNAL SQLSTATE '45000'
177         SET MESSAGE_TEXT = 'Invalid ticket class specified.';
178     END IF;
179
180 END $$
181
182 -- this is for checking pnr_status
183 DELIMITER $$
184 CREATE PROCEDURE check_pnr_status(IN in_pnr_number VARCHAR(20))
185 BEGIN
186     SELECT train_number,date_of_travel,boarding_station,destination_station,ticket_class,no_o
187     f_passenger,status FROM ticket WHERE pnr_number = in_pnr_number;
188 END $$
189
190 DELIMITER ;
191
192 DELIMITER $$
193 CREATE PROCEDURE train_shedule_lookup(IN in_train_number VARCHAR(10))
194 BEGIN
195     SELECT train_number,train_name,runs_on_days FROM train WHERE train_number = in_train_number;
196     SELECT islno,station_code,arrival_time,departure_time,distance FROM route WHERE train_number = in_train_number;
197 END $$
198
199 DELIMITER ;
200
201 -- fror cheking seat availability
202 DELIMITER $$
203 CREATE PROCEDURE check_seat_availability(IN in_train_number VARCHAR(10),IN in_date DATE,IN in
204 _class VARCHAR(10))
205 BEGIN
206     DECLARE available_seats INT;
207
208     if(in_class = 'first_ac') then
209         SELECT first_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
210     elseif(in_class = 'second_ac') then
211         SELECT second_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
212     elseif(in_class = 'third_ac') then
213         SELECT third_ac_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
214     elseif(in_class = 'sleeper') then
215         SELECT sleeper_available INTO available_seats FROM train_inventory WHERE train_number = in_train_number AND travel_date = in_date;
216     elseif(in_class = 'general') then

```

```

215     SELECT general_available INTO available_seats FROM train_inventory WHERE train_number
    = in_train_number AND travel_date = in_date;
216     end if;
217     SELECT available_seats;
218
219 END $$
220
221 -- for getting list of passenger travelling on a train on specific date
222 DELIMITER $$
223 CREATE PROCEDURE list_passengers_on_train(IN in_train_number VARCHAR(10), IN in_date DATE, IN
    in_status VARCHAR(20))
224 BEGIN
225     if(in_status = 'all') then
226         SELECT ticket.pnr_number, passenger_name, ticket_class, age, gender, seat_number, coach, sta
tus FROM ticket, passenger WHERE ticket.train_number = in_train_number AND ticket.date_of_trav
el = in_date AND ticket.pnr_number=passenger.pnr_number;
227     else
228         SELECT ticket.pnr_number, passenger_name, ticket_class, age, gender, seat_number, coach, sta
tus FROM ticket, passenger WHERE ticket.train_number = in_train_number AND ticket.date_of_trav
el = in_date AND ticket.pnr_number=passenger.pnr_number AND ticket.status = in_status;
229     end if;
230 END $$
231
232 DELIMITER $$
233
234 CREATE PROCEDURE get_total_revenue(
235     IN start_date DATE,
236     IN end_date DATE
237 )
238 BEGIN
239     SELECT
240         SUM(total_fare) AS total_revenue
241     FROM
242         ticket
243     WHERE
244         date_of_book BETWEEN start_date AND end_date
245         AND status = 'CONFIRMED';
246 END $$
247
248 DELIMITER ;
249
250 -- for cancelling the ticket
251 CREATE PROCEDURE cancel_ticket(IN in_pnr_number VARCHAR(20))
252 BEGIN
253     DECLARE t_train_number VARCHAR(10);
254     DECLARE t_travel_date DATE;
255     DECLARE t_ticket_class VARCHAR(10);
256     DECLARE t_no_of_passenger INT;
257
258     -- Step 1: Get ticket details
259     SELECT train_number, date_of_travel, ticket_class, no_of_passenger
260     INTO t_train_number, t_travel_date, t_ticket_class, t_no_of_passenger
261     FROM ticket

```

```

262 WHERE pnr_number = in_pnr_number;
263
264 -- Step 2: Delete passenger records
265 DELETE FROM passenger WHERE pnr_number = in_pnr_number;
266
267 -- Step 3: Delete ticket record
268 DELETE FROM ticket WHERE pnr_number = in_pnr_number;
269
270 -- Step 4: Update seat availability in train_inventory
271 IF t_ticket_class = 'first_ac' THEN
272     UPDATE train_inventory
273     SET first_ac_available = first_ac_available + t_no_of_passenger
274     WHERE train_number = t_train_number AND travel_date = t_travel_date;
275 ELSEIF t_ticket_class = 'second_ac' THEN
276     UPDATE train_inventory
277     SET second_ac_available = second_ac_available + t_no_of_passenger
278     WHERE train_number = t_train_number AND travel_date = t_travel_date;
279 ELSEIF t_ticket_class = 'third_ac' THEN
280     UPDATE train_inventory
281     SET third_ac_available = third_ac_available + t_no_of_passenger
282     WHERE train_number = t_train_number AND travel_date = t_travel_date;
283 ELSEIF t_ticket_class = 'sleeper' THEN
284     UPDATE train_inventory
285     SET sleeper_available = sleeper_available + t_no_of_passenger
286     WHERE train_number = t_train_number AND travel_date = t_travel_date;
287 ELSEIF t_ticket_class = 'general' THEN
288     UPDATE train_inventory
289     SET general_available = general_available + t_no_of_passenger
290     WHERE train_number = t_train_number AND travel_date = t_travel_date;
291 END IF;
292
293 END $$
294
295 -----
296
297 -- listing all triggers
298 --triggers while booking on table ticket
299 DELIMITER $$
300
301 CREATE TRIGGER before_ticket_insert
302 BEFORE INSERT ON ticket
303 FOR EACH ROW
304 BEGIN
305     DECLARE avail INT DEFAULT 0;
306
307     -- Check based on the booked class:
308     IF NEW.ticket_class = 'first_ac' THEN
309         SELECT first_ac_available INTO avail
310         FROM train_inventory
311         WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;
312     ELSEIF NEW.ticket_class = 'second_ac' THEN
313         SELECT second_ac_available INTO avail

```

```

314         FROM train_inventory
315         WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;
316
317     ELSEIF NEW.ticket_class = 'third_ac' THEN
318         SELECT third_ac_available INTO avail
319         FROM train_inventory
320         WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;
321
322     ELSEIF NEW.ticket_class = 'sleeper' THEN
323         SELECT sleeper_available INTO avail
324         FROM train_inventory
325         WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;
326
327     ELSEIF NEW.ticket_class = 'general' THEN
328         SELECT general_available INTO avail
329         FROM train_inventory
330         WHERE train_number = NEW.train_number AND travel_date = NEW.date_of_travel;
331
332     ELSE
333         SIGNAL SQLSTATE '45000'
334         SET MESSAGE_TEXT = 'Invalid seat class specified.';
335     END IF;
336
337     IF avail IS NULL OR avail <= 0 THEN
338         SIGNAL SQLSTATE '45000'
339         SET MESSAGE_TEXT = 'No available seats for the chosen class on this date.';
340     END IF;
341 END $$
342
343 DELIMITER ;
344
345 --after insert trigger
346 DELIMITER $$
347
348 CREATE TRIGGER after_ticket_insert
349 AFTER INSERT ON ticket
350 FOR EACH ROW
351 BEGIN
352     IF NEW.ticket_class = 'first_ac' THEN
353         UPDATE train_inventory
354         SET first_ac_available = first_ac_available - 1
355         WHERE train_number = NEW.train_number
356             AND travel_date = NEW.date_of_travel;
357
358     ELSEIF NEW.ticket_class = 'second_ac' THEN
359         UPDATE train_inventory
360         SET second_ac_available = second_ac_available - 1
361         WHERE train_number = NEW.train_number
362             AND travel_date = NEW.date_of_travel;
363
364     ELSEIF NEW.ticket_class = 'third_ac' THEN
365         UPDATE train_inventory
366         SET third_ac_available = third_ac_available - 1

```

```

367         WHERE train_number = NEW.train_number
368             AND travel_date = NEW.date_of_travel;
369
370     ELSEIF NEW.ticket_class = 'sleeper' THEN
371         UPDATE train_inventory
372         SET sleeper_available = sleeper_available - 1
373         WHERE train_number = NEW.train_number
374             AND travel_date = NEW.date_of_travel;
375
376     ELSEIF NEW.ticket_class = 'general' THEN
377         UPDATE train_inventory
378         SET general_available = general_available - 1
379         WHERE train_number = NEW.train_number
380             AND travel_date = NEW.date_of_travel;
381     END IF;
382 END $$
383
384 DELIMITER ;
385
386 -- trigger while cancelling the ticket
387 DELIMITER $$
388 CREATE TRIGGER update_cancellation_record
389 AFTER DELETE ON ticket
390 FOR EACH ROW
391 BEGIN
392     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)
393     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');
394 END $$
395
396 DELIMITER ;
397
398 DELIMITER $$
399
400 CREATE TRIGGER update_cancellation_record
401 AFTER DELETE ON ticket
402 FOR EACH ROW
403 BEGIN
404     DECLARE base_deduction DECIMAL(10,2);
405     DECLARE extra_deduction DECIMAL(10,2);
406     DECLARE total_deduction DECIMAL(10,2);
407     DECLARE refund_amount DECIMAL(10,2);
408     DECLARE days_left INT;
409
410     -- Calculate days between current date and travel date
411     SET days_left = DATEDIFF(OLD.date_of_travel, CURDATE());
412
413     -- Base deduction: 10% cancellation charge
414     SET base_deduction = OLD.total_fare * 0.10;
415
416     -- Extra deduction based on how close the travel date is
417     IF days_left > 7 THEN

```

```

418     SET extra_deduction = 0;
419 ELSEIF days_left BETWEEN 4 AND 7 THEN
420     SET extra_deduction = OLD.total_fare * 0.10;
421 ELSEIF days_left BETWEEN 1 AND 3 THEN
422     SET extra_deduction = OLD.total_fare * 0.20;
423 ELSE
424     SET extra_deduction = OLD.total_fare * 0.30;
425 END IF;
426
427 -- Total deduction
428 SET total_deduction = base_deduction + extra_deduction;
429
430 -- Final refund amount
431 SET refund_amount = OLD.total_fare - total_deduction;
432
433 -- Insert into cancellation record
434 INSERT INTO cancellation_record (
435     pnr_number, train_number, date_of_travel, boarding_station, destination_station,
436     ticket_class, no_of_passenger, status, refund_amount, refund_status
437 )
438 VALUES (
439     OLD.pnr_number, OLD.train_number, OLD.date_of_travel, OLD.boarding_station,
440     OLD.destination_station, OLD.ticket_class, OLD.no_of_passenger, OLD.status,
441     refund_amount, 'DONE'
442 );
443 END $$
444
445 DELIMITER ;

```

Other interesting queries beyond the required ones

```

1  1. Find trains with maximum waitlisted passengers in a day
2  SELECT train_number, travel_date, COUNT(*) AS waitlist_count
3  FROM ticket
4  WHERE status = 'Waitlisted'
5  GROUP BY train_number, travel_date
6  ORDER BY waitlist_count DESC
7  LIMIT 1;
8
9
10 2. Average occupancy per class for a train
11 SELECT train_number, class,
12     ROUND(SUM(CASE WHEN status = 'Confirmed' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) AS
    occupancy_percentage
13 FROM ticket
14 GROUP BY train_number, class;
15
16
17 3. Peak booking hours (hourly ticket distribution)
18 SELECT HOUR(booking_time) AS booking_hour, COUNT(*) AS tickets_booked
19 FROM ticket
20 GROUP BY booking_hour
21 ORDER BY tickets_booked DESC;

```

```
22
23
24 4. Top 5 most frequently used train numbers
25 SELECT train_number, COUNT(*) AS usage_count
26 FROM ticket
27 GROUP BY train_number
28 ORDER BY usage_count DESC
29 LIMIT 5;
```