Queries View for Airline Management System

General Cases

1. View Flights and flight Status

Provides search functionalities for flights and flight status based on departure and arrival airports or cities and date. Accessible without login.

Search for Future Flights:

```
SELECT * FROM flight
            WHERE
                  (departure_airport_code in
                                    (SELECT code from airport
                                          where city = %s)
                              OR departure_airport_code = %s)
                  AND
                  (arrival_airport_code in
                                    (SELECT code from airport
                                          where city = %s)
                              OR arrival_airport_code = %s)
                  AND
                  departure_date = %s
```

Check Flight Status:

```
SELECT * FROM flight
                 WHERE airline_name = %s AND flight_number = %s AND
departure_date = %s
```

2. User Registration

Customer Registration:

Inserts customer data into the customer table, including personal details and hashed password.

Check if customer already exist

```
SELECT * FROM customer WHERE email_address = %s
```

Create customer account:

```
INSERT INTO customer
VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
```

Create customer's phone number:

```
INSERT INTO customer_phone_number VALUES(%s, %s)
```

Staff Registration:
Inserts staff data into the customer table, including personal details and hashed password.

Check if staff already exist

```
SELECT * FROM airline_staff WHERE username = %s
```

Create airline staff account:

```
INSERT INTO airline_staff VALUES(%s, %s, %s, %s, %s, %s)
```

Create airline staff's phone number:

```
INSERT INTO airline_staff_phone_number VALUES(%s, %s)
```

Create airline staff's email:

```
INSERT INTO airline_staff_email VALUES(%s, %s)
```

Customer Use Case

1. Customer login

Authenticates customers.

```
SELECT * FROM customer WHERE email_address = %s;
```

Check if the password is correct as well by comparing the hash password stored in the database and the entered password `hash_password_md5(password)`

2. View Flights

Lists upcoming and past flights for the logged-in customer.

Previous Flights

```
SELECT * FROM purchase natural join ticket natural join flight
            WHERE
                  (email_address = %s) AND (departure_date < CURDATE() OR
                                                            (departure_date = CURDATE()
AND
                                                                  departure_time <
CURTIME()))
```

Upcoming Flights

```
SELECT * FROM purchase natural join ticket natural join flight
            WHERE
                  (email_address = %s) AND (departure_date > CURDATE() OR
                                                  (departure_date    =    CURDATE()
AND
                                                        departure_time                >
CURTIME()))
```

3. Purchase tickets

Allows customers to book tickets for flights.

Fetch all the tickets of the flight

```
SELECT ID FROM ticket
                       WHERE flight_number = %s AND airline_name = %s AND
                       departure_date = %s AND departure_time = %s
```

For each ticket, check if the ticket is purchase by customer

```
SELECT * FROM purchase
                                        WHERE ID = %s
```

If the flight is nearly full (80% of seats is purchased), Update the price of ticket

```
UPDATE ticket SET ticket_price = %s
                                     WHERE ID =%s
```

Insert the purchase information

```
INSERT    INTO    purchase    VALUES(%s,    %s,    %s,    %s,    %s,    NOW(),
NOW(), %s, %s, %s, %s)
```

If the flight is full, send error message

4. Cancel Tickets

Cancels tickets more than 24 hours before departure.

Check if more than 24 hours before departure

```
SELECT * FROM purchase natural join ticket natural join flight
```

```
                    WHERE ID = %s AND NOW() >=
DATE_SUB(CONCAT(departure_date, ' ', departure_time), INTERVAL 1 DAY)
```

Reset ticket price and update to base price if this ticket not purchased

```
SELECT base_price FROM flight natural join ticket natural join purchase
                    WHERE ID = %s

UPDATE ticket SET ticket_price = %s
                                WHERE ID = %s
```

Cancel Tickets. Remove the info in purchase table

```
DELETE FROM purchase WHERE ID =    %s
```

5. Rate and Comment on Flights

Allows customers to provide ratings for past flights.

```
SELECT * FROM rate
                WHERE email_address = %s AND flight_number = %s AND
                        airline_name = %s AND departure_date = %s AND
                        departure_time = %s
INSERT INTO rate VALUES(%s, %s, %s, %s, %s, %s, %s)
```

6. Track Spending

Displays total spending for a specific date range. Default is to track customer's spending in last year and spending each month in last 6 months

Track spending last year.

```
SELECT ROUND(SUM(ticket_price), 2) as total FROM purchase natural join ticket
                    WHERE email_address = %s AND purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
```

Tracking spending each month last 6 months

```
SELECT DATE_FORMAT(purchase_date, '%%M') AS month,
ROUND(SUM(ticket_price), 2) AS total_spent
                    FROM purchase natural join ticket
                    WHERE email_address = %s AND purchase_date >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
                    GROUP BY month
                    ORDER BY month
```

Track total spending in ranged date

```
SELECT ROUND(SUM(ticket_price), 2) as total FROM purchase natural join ticket
                    WHERE email_address = %s AND purchase_date >= %s AND purchase_date <= %s
```

Track spending each month in ranged date

```
SELECT DATE_FORMAT(purchase_date, '%%M') AS month, ROUND(SUM(ticket_price), 2) AS total_spent
                    FROM purchase natural join ticket
                    WHERE email_address = %s AND purchase_date >= %s AND purchase_date <= %s
                    GROUP BY month
                    ORDER BY month
```

7. Edit customer profile

Customer can update their personal information

```
UPDATE customer SET {key} = '+'%s WHERE email_address = %s', (value, email_address)
```

Customer can insert phone numbers

```
INSERT INTO customer_phone_number VALUES(%s, %s)
```

Staff Use Case

1. Staff login

Authenticate Staffs

```
SELECT * FROM airline_staff WHERE username = %s
```

Check if the password is correct as well by comparing the hash password stored in the database and the entered password `hash_password_md5(password)`

2. Edit staff profile

Staff can update their personal information

```
UPDATE airline_staff SET {key} = '+'%s WHERE username = %s', (value, username)
```

Insert phone numbers

```
INSERT INTO airline_staff_phone_number VALUES(%s, %s)
```

Insert emails

```
INSERT INTO airline_staff_email VALUES(%s, %s)
```

3. Create new Flights

Staff can create new flights of an airline as its worker

Get the staff's airline. Return error if not authorized to create flights

```
SELECT airline_name FROM airline_staff
                    WHERE username = %s
```

Check if the airplane is in maintenance

```
SELECT *
             FROM maintenance_procedure
             WHERE airplane_id = %s
             AND (
                 EXISTS (
                     SELECT 1
                     FROM maintenance_procedure AS mp
                     WHERE mp.airplane_id = %s
                         AND                 %s                 BETWEEN
CONCAT(mp.maintenance_start_date, ' ', mp.maintenance_start_time)
                                   AND
CONCAT(mp.maintenance_end_date, ' ', mp.maintenance_end_time)
                 )
                 OR EXISTS (
                     SELECT 1
                     FROM maintenance_procedure AS mp
                     WHERE mp.airplane_id = %s
                         AND %s BETWEEN CONCAT(mp.maintenance_start_date, ' ', mp.maintenance_start_time)
                         AND CONCAT(mp.maintenance_end_date, ' ', mp.maintenance_end_time)
                 )
                 OR EXISTS (
                     SELECT 1
                     FROM maintenance_procedure AS mp
                     WHERE mp.airplane_id = %s
                         AND %s < CONCAT(mp.maintenance_end_date, ' ', mp.maintenance_end_time)
                         AND %s > CONCAT(mp.maintenance_start_date, ' ', mp.maintenance_start_time)
                 )
             )
```

Create the flight

```
INSERT INTO flight VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
```

Get the number of seats and generate the ticket for this flight. New created Ticket ID is added upon the existing largest ticket ID.

```
SELECT num_seats FROM airplane
            WHERE ID = %s
SELECT MAX(ID) FROM ticket
INSERT INTO ticket VALUES(%s, %s, %s, %s, %s, %s)
```

4. Change flight status

Staff can change flight status of his/her airline's flights

Check if the entered flight exists

```
SELECT * FROM flight
            WHERE flight_number = %s AND departure_date = %s AND departure_time = %s
```

Update the flight status

```
UPDATE flight SET flight_status = %s
                WHERE flight_number = %s AND departure_date = %s AND
departure_time = %s
```

5. Add airplane

Staff can add airplane

Get the staff's airline name to check if staff can add this airplane

```
SELECT airline_name FROM airline_staff
                                        WHERE username = %s
```

Insert the airplane

```
INSERT INTO airplane VALUES(%s, %s, %s, %s, %s, %s, %s)
```

Enter the age of the airplane

```
UPDATE airplane
          SET age = TIMESTAMPDIFF(YEAR, manufacturing_date, CURDATE())
                   -      (DATE_FORMAT(CURDATE(),      '%%M-%%D')      <
DATE_FORMAT(manufacturing_date, '%%M-%%D'))
          WHERE ID = %s
```

6. Add airports

Staff can add airports

```
INSERT INTO airport VALUES(%s, %s, %s, %s, %s, %s)
```

7. Schedule maintenance

Staff can schedule maintenance. The needed info are airplane_id, maintenance start date/time, maintenance end date/time.

```
INSERT INTO maintenance_procedure VALUES(%s, %s, %s, %s, %s, %s)
```

8. View Futures airline's future 30-day flights, and airline's ranged date flights

Get staff’s airline name

```
SELECT airline_name FROM airline_staff
                                        WHERE username = %s
```

Default: view airline’s future 30-day flights

```
SELECT * FROM flight
                   WHERE airline_name = %s AND departure_date BETWEEN
CURDATE()

AND

DATE_ADD(CURDATE(), INTERVAL 1 MONTH)
```

View flights in ranged date. Options for staff to ender are: departure_date, arrival_date, departure airport code, arrival airport code, departure city, arrival city. Staff can enter the information optionally and the corresponding results will be shown.

```
SELECT DISTINCT airline_name, flight_number, departure_date, departure_time,
                                       arrival_date,     arrival_time,     base_price,
flight_status, airplane_id
                   FROM flight natural join airport
                   WHERE airline_name = %s

if start_date:
          query += 'AND departure_date >= %s'
          params += (departure_date,)
     if end_date:
          query += 'AND departure_date <= %s'
          params += (arrival_date,)
     if departure_airport_code:
          query += 'AND departure_airport_code = %s'
          params += (departure_airport_code,)
     if arrival_airport_code:
          query += 'AND arrival_airport_code >= %s'
          params += (arrival_airport_code,)
     if departure_airport_city:
          query += '''AND (departure_airport_code in
                                  (SELECT code from airport
                                       where city = %s))'''
          params += (departure_airport_city,)
     if arrival_airport_city:
          query += '''AND (arrival_airport_code in
                                  (SELECT code from airport
                                       where city = %s))'''
          params += (arrival_airport_city,)
```

9. View Flight Ratings

Airline Staff will be able to see each flight's average ratings and all the comments and ratings of that flight given by the customers

Get the staff's airline

```
SELECT airline_name FROM airline_staff
                                        WHERE username = %s
```

Get all the comments

```
SELECT * FROM rate
                    WHERE   airline_name  =  %s  AND  flight_number  =  %s  AND
departure_date = %s AND departure_time = %s
```

Get the average rating

```
SELECT AVG(rating) FROM rate
                    WHERE   airline_name  =  %s  AND  flight_number  =  %s  AND
departure_date = %s AND departure_time = %s
```

10. View Frequent Customers

Fetch the customer's email address and the total spending. Regard the customer with highest spending as the most frequent customer

```
SELECT email_address, SUM(ticket_price) as revenue
                     FROM flight natural join purchase natural join ticket
```

```
                    WHERE airline_name = %s
                    GROUP BY email_address
                    ORDER BY revenue DESC
```

11. View Customer’s Flights

Based on the flight_number, view all the customers on that flight

```
SELECT * FROM flight natural join ticket natural join purchase
                    WHERE airline_name = %s AND flight_number = %s
                    AND departure_date = %s
```

12. View Earned Revenue

Staff can view airline’s total revenue last month, last year, and the total number of tickets sold each month last year

Get airline name of staff

```
SELECT airline_name FROM airline_staff
                                        WHERE username = %s
```

Get the total revenue last month

```
SELECT ROUND(SUM(ticket_price), 2) FROM purchase natural join ticket
                    WHERE airline_name = %s AND purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
```

Get the total revenue last year

```
SELECT ROUND(SUM(ticket_price), 2) FROM purchase natural join ticket
                    WHERE airline_name = %s AND purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
```

Count how many ticket sold each month last year

```
SELECT DATE_FORMAT(purchase_date, '%%M') AS month, COUNT(*) as
ticket_count
                    FROM purchase natural join ticket natural join flight
                    WHERE airline_name = %s AND purchase_date >=
DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
                    GROUP BY month
                    ORDER BY month
```

13. View On-Time/Delayed Flights
Staff can view all the airline’s on-time or delayed flights

Get staff’s airline

```
SELECT airline_name FROM airline_staff
                                          WHERE username = %s
```

View on-time/delayed flights

```
SELECT * FROM flight
                    WHERE airline_name = %s AND flight_status = %s
```