Use Case:

1. View public info
Anyone can view flights and flight status based on date

SELECT *
FROM flight
WHERE flight.airport_code=%s AND flight.arrival_airport_code=%s AND
DATE(flight.departure)=%s and flight.departure > NOW()"""
---Searches database for flights with match departure and destination airport codes and departure date. Doesn't allow for viewing of past flights.
---Query used a second time if there is a round-trip flight with the return date and the airports swapped

2. Register
Customers and Staff can register on different pages. They have to input all of the required information.

INSERT INTO Customer VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
INSERT INTO Staff VALUES (%s, %s, %s, %s, %s, %s, %s, %s)
---Information is grabbed from form and inserted into designated tables of database

3. Login
Customers and staff can login. Username/email and password must match database for success
SELECT password FROM Customer WHERE email = %s
SELECT password FROM Staff WHERE username = %s
---checks password based on email/username in correct table

customer:
1. view my flights
Customer can view all upcoming flights
SELECT flight.airline_name, flight.airport_code, flight.arrival, flight.departure, flight.arrival_airport_code, flight.base_price, flight.Status, flight.flight_num
FROM ticket INNER JOIN flight ON ticket.flight_num = flight.flight_num
WHERE ticket.email = %s and flight.departure > NOW()
---From ticket: finds all tickets under user's email
---uses the ticket number to get flight info to display

2. Search for flights
User can view flights and flight status based on date

SELECT *

FROM flight
WHERE flight.airport_code=%s AND flight.arrival_airport_code=%s AND
DATE(flight.departure)=%s and flight.departure > NOW()"""
---Searches database for flights with match departure and destination airport codes and departure date. Doesn't allow for viewing of past flights.
---Query used a second time if there is a round-trip flight with the return date and the airports swapped

3. Purchase tickets
User can select from trip search results to purchase a ticket
SELECT MAX(ticket_id), MAX(purchase_id), NOW() FROM purchase
---Used to create a unique ticket id, finds the max ticket_id so we know that max + 1 doesn't exist yet

INSERT INTO `Ticket` (`ticket_id`, `email`, `flight_num`, `rating`, `comments`, `calc_price`, `first_name`, `last_name`, `dob`)
VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s)
---Inserts form information into ticket tables
INSERT INTO `Purchase` (`purchase_id`, `ticket_id`, `purchase_date_time`, `card_info`)
VALUES(%s, %s, %s, %s)
---Inserts needed info into purchase table

4. cancel trip
From user's homepage where they can view upcoming trips, they can cancel
DELETE
FROM Ticket
WHERE flight_num=%s and email=%s
---Flight num is stored in results and used when cancel is clicked along with logged in email
---Deleted data from Ticket

5. give ratings and comments on past trips
User can view past flights and from there leave comments and ratings
UPDATE Ticket
SET rating=%s, comments=%s
WHERE email=%s and ticket_id=%s
---Updates ticket table by finding ticket row and changing the rating and comment values

6. track my spending
Users can see there past year and 6 months of spending. They can also specify a range.

SELECT SUM(ticket.calc_price) as amount
FROM Purchase JOIN Ticket ON Purchase.ticket_id = Ticket.ticket_id
WHERE email=%s and purchase.purchase_date_time BETWEEN %s and %s

GROUP BY Ticket.email
---sums the calculated ticket price between the two specified dates using the user's email

SELECT SUM(ticket.calc_price) as amount
FROM Purchase JOIN Ticket ON Purchase.ticket_id = Ticket.ticket_id
WHERE email=%s and purchase.purchase_date_time > %s
GROUP BY Ticket.email
---sums the calculated ticket price in the last year using the user's email

SELECT DATE_FORMAT(Purchase.purchase_date_time, '%%Y-%%m') AS month,
SUM(ticket.calc_price) AS totalSpent
FROM Purchase JOIN Ticket ON Purchase.ticket_id = Ticket.ticket_id
WHERE Ticket.email = %s and purchase.purchase_date_time > %s
GROUP BY month
ORDER BY month ASC
---Views purchases by month and sums the user's ticket calc prices of each month they spent money in the last six months

SELECT DATE_FORMAT(Purchase.purchase_date_time, '%%Y-%%m') AS month,
SUM(ticket.calc_price) AS totalSpent
FROM Purchase
JOIN Ticket ON Purchase.ticket_id = Ticket.ticket_id
WHERE email=%s AND purchase.purchase_date_time BETWEEN %s AND %s
GROUP BY month
ORDER BY month ASC
---Views purchases by month and sums the user's ticket calc prices of each month they spent money between two dates

7. logout
user can logout which ends session and takes them to homepage
no query

staff:
1. view flights
shows future flights of the airline the staff works for
SELECT * FROM flight WHERE departure > NOW() and departure < NOW() + INTERVAL 1 MONTH
---Selects all flights where departure is between now and in one month

2. create new flight
staff can use form to add flight to flight table of database

INSERT INTO flight VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
---Inserts form information into flight table

3. change status
staff can choose airplane and change the status to on time or delayed
UPDATE Flight SET status = %s WHERE airplane_id = %s
---Finds airplane in flight to change status in database

4. add airplane
staff can use form to add airplane to airplane table of database
INSERT INTO Airplane VALUES (%s, %s, %s, %s, %s, %s)
---Inserts form information into airplane table

5. add airport
staff can use form to add airport to airport table of database
INSERT INTO Airport VALUES (%s, %s, %s, %s, %s)
---Inserts form information into airport table

6. view flight ratings
staff can see flight's average ratings and the commets and ratings from Customers
SELECT ticket.rating, ticket.flight_num, ticket.comments
FROM Ticket INNER JOIN Flight on Ticket.flight_num = Flight.flight_num
---Grabs comments and ratings from ticket based on flight

7. schedule maintenance
staff can use form to add maintenance period to maintenance table of database
INSERT INTO Maitenance VALUES (%s, %s, %s)
---Inserts form information into maintenance table

8. view frequent customers
staff can view most frequent customer and select a customer and view their flights
SELECT Ticket.email, COUNT(*) AS flight_count
FROM Ticket INNER JOIN Flight ON Ticket.flight_num = Flight.flight_num WHERE
Flight.departure > DATE_SUB(NOW(), INTERVAL 1 YEAR)
GROUP BY Ticket.email
ORDER BY flight_count DESC LIMIT 1
---Counts flights of customers and chooses one with most

SELECT DISTINCT email FROM Ticket
---Selects all customers in database

SELECT Flight.flight_num, Flight.departure, Flight.arrival, Flight.airport_code,
Flight.arrival_airport_code

FROM Ticket INNER JOIN Flight ON Ticket.flight_num = Flight.flight_num
WHERE Ticket.email = %s
---Views flight information based on customer

9. view earned revenue
staff can see total revenue from last month and last year
SELECT base_price, departure FROM Flight
Gets the departure anf price for each flight to later be used to calculate revenue

10. logout
staff can logout which ends session and takes them to homepage
no query