SQL Queries and Explanations

SQL:

```
SELECT f.airline_name, f.flight_number, f.depart_date, f.depart_time, f.arrival_date, f.arrival_time
        FROM flight f
            WHERE f.depart_date BETWEEN CURDATE() AND DATE_ADD(CURDATE(), INTERVAL 30 DAY)
```

Explanation:

Retrieves details about flights, including airline, flight number, departure, and arrival times, for the next 30 days.

SQL:

```
SELECT airline_name, flight_number, depart_date, arrival_date, flight_status
        FROM flight
        WHERE flight_number = %s AND depart_date = %s
```

Explanation:

Checks the current status of a specific flight, including its schedule and status.

SQL:

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

Explanation:

Authenticates a customer by matching their email address with stored records.

SQL:

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

Explanation:

Authenticates an airline staff member using their username.

SQL:

*SELECT * FROM Airline_Staff WHERE username = %s*

Explanation:

Checks if an airline staff member exists in the system.

SQL:

*SELECT * FROM customer WHERE email_address = %s*

Explanation:

Checks if a customer exists in the database using their email address.

SQL:

SELECT base_price FROM flight WHERE flight_number = %s

Explanation:

Retrieves the base price of a flight ticket for a given flight number.

SQL:

INSERT INTO ticket (ticket_id, airline_name, flight_number, depart_date, depart_time, calculated_price, first_name, last_name, date_of_birth)

Explanation:

Inserts a new ticket into the database with details like passenger name, date of birth, and calculated price.

SQL:

*SELECT * FROM ticket WHERE ticket_id = %s*

Explanation:

Checks if a specific ticket exists before cancellation.

SQL:

DELETE FROM ticket WHERE ticket_id = %s

Explanation:

Deletes a ticket record from the database.

SQL:

DELETE FROM purchase WHERE ticket_id = %s

Explanation:

Deletes the corresponding purchase record for a canceled ticket.

SQL:

*SELECT * FROM airline_staff WHERE username = %s*

Explanation:

Retrieves details of an airline staff member using their username.

SQL:

*SELECT * FROM employed_by WHERE username = %s*

Explanation:

Fetches the airline associated with the staff member.

SQL:

SELECT airline_name FROM employed_by WHERE username = %s

Explanation:

Gets the airline name for the logged-in airline staff member.

SQL:

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

DATE_ADD(CURDATE(), INTERVAL 30 DAY)

Explanation:

Retrieves all flights for the staff's airline scheduled in the next 30 days.

SQL:

SELECT customer.first_name, customer.last_name, customer.email_address

FROM ticket JOIN purchase ON ticket.ticket_id = purchase.ticket_id

JOIN customer ON purchase.email_address = customer.email_address

WHERE ticket.flight_number = %s AND ticket.depart_date = %s AND ticket.depart_time = %s

Explanation:

Fetches customer details for a specific flight.

SQL:

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

Explanation:

Adds a new flight to the database with details like flight number, departure and arrival times, and status.

SQL:

UPDATE flight SET flight_status = %s WHERE airline_name = %s AND flight_number = %s AND depart_date = %s AND depart_time = %s

Explanation:

Updates the status of a specific flight.

SQL:

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

Explanation:

Adds a new airport to the database with details like name, city, and number of terminals.

SQL:

INSERT INTO Airplane VALUES (%s, %s, NULL, %s, %s, %s, %s, %s)

Explanation:

Adds a new airplane to the database, specifying airline, seat capacity, and manufacturing details.

SQL:

INSERT INTO Maintenance VALUES (%s, %s, %s, %s, %s)

Explanation:

Schedules maintenance for an airplane with start and end dates and times.

SQL:

UPDATE Airplane SET maintenance_id = %s WHERE airline_name = %s AND airplane_id_number = %s

Explanation:

Links a maintenance record to a specific airplane.

SQL:

SELECT f.flight_number, f.depart_date, f.depart_time, COALESCE(AVG(r.rating), 0) AS average_rating, GROUP_CONCAT(r.comment) AS comments

FROM flight f

LEFT JOIN reviews r ON f.flight_number = r.flight_number

GROUP BY f.flight_number, f.depart_date, f.depart_time

Explanation:

Retrieves average ratings and comments for flights.