Public Use Cases

- **View flights**

Any user can view a list of flights even if they aren't logged in. This will be visible on the home page. The following query returns a list of flights.

```
SELECT AirlineName, FlightNumber, DepartureDate, DepartureTime, ArrivalDate, FlightStatus
FROM Flight NATURAL JOIN Updates
ORDER BY DepartureDate
```

Customer Use Cases

- **View my flights**

By default, customers will see information for future flights that they have purchased already. We will query information needed to uniquely identify a flight using attributes from that table. The query below returns the future flights of customers.

```
SELECT AirlineName, FlightNumber, DepartureDate, DepartureTime, ArrivalDate, ArrivalTime,
FlightStatus
FROM Flight NATURAL JOIN Changes NATURAL JOIN PurchasedFor NATURAL JOIN Ticket
NATURAL JOIN Customer
WHERE CustomerEmail = %s AND DepartureDate > CURRENT_DATE OR (DepartureDate =
CURRENT_DATE AND DepartureTime > CURRENT_TIME)
ORDER BY DepartureDate
```

There is also an option for customers to view past flights. The query below returns the past flights of customers.

```
SELECT AirlineName, FlightNumber, DepartureDate, DepartureTime, ArrivalDate, ArrivalTime,
FlightStatus
FROM Flight NATURAL JOIN Changes NATURAL JOIN PurchasedFor NATURAL JOIN Ticket
NATURAL JOIN CUSTOMER
WHERE CustomerEmail = %s AND DepartureDate < CURRENT_DATE OR (DepartureDate =
CURRENT_DATE AND DepartureTime < CURRENT_TIME)
ORDER BY DepartureDate
```

- **Search for flights**

Customers can search for future flights that are either one way or round trip flights.

```
SELECT f.AirlineName, f.FlightNumber, f.DepartureDate, f.DepartureTime, f.ArrivalDate, f.FlightStatus,
f.numberOfSeats, COUNT(ticketID) as numFlights
FROM Flight as f
LEFT JOIN PurchasedFor AS pf
```

```
On pf.FlightNumber = f.FlightNumber AND pf.DepartureDate = f.DepartureDate AND pf.DepartureTime
= f.DepartureTime
INNER JOIN Updates AS u
ON u.FlightNumber = f.FlightNumber AND u.DepartureDate = f.DepartureDate AND u.DepartureTime =
f.DepartueTime
INNER JOIN Airplane
ON f.AirplaneID = airplane.AirplaneID
INNER JOIN Airport AS airport1
ON airport1.AirportName = f.DepartureAirport
INNER JOIN Airport AS airport2
ON airport2.AirportName = f.ArrivalAirport
WHERE f.FlightNumber NOT IN
        SELECT FlightNumber
        FROM Flight as f2
        GROUP BY FlightNumber
        HAVING COUNT(f2.FlightNumber) > 1
AND airport1.AirportCity = %s AND f.DepartureAirport = %s AND airport2.AirportCity = %s AND
f.ArrivalAirport = %s AND f.DepartureDate = %s
GROUP BY f.AirlineName, f.FlightNumber, f.DepartureDate, f.DepartureTime, f.ArrivalDate,
f.FlightStatus
HAVING numFlights < f.NumberOfSeats


SELECT f.AirilneName, f.FlightNumber, f.DepartureDate, f3.DepartureDate AS returnDate,
f.DepartureTime, f3.DepartureTime AS returnTime, f.ArrivalDate, f.FlightStatus, COUNT(ticketID) AS
numFlights, f.numberOfSeats
FROM Flight AS f
LEFT JOIN PurchasedFor AS pf
ON pf.FlightNumber = f.FlightNumber AND pf.DepartureDate = f.DepartureDATE AND
pf.DepartureTime = f.DepartureTime
INNER JOIN Updates AS u
ON u.FlightNumber = f.FlightNumber AND u.DepartureDate = f.DepartureDate AND u.DepartureTime =
f.DepartureTime
INNER JOIN Airport AS airport1
ON airport1.AirportName = f.DepartureAirport
INNER JOIN Airport AS airport2
ON airport2.AirportName = f.ArrivalAirport
INNER JOIN Flight AS f3
ON f.FlightNumber = f3.FlightNumber AND f.ArrivalAirport = f3.DepartureAirport
WHERE f.FlightNumber IN
        SELECT FlightNumber
        FROM Flight AS f2
        GROUP BY FlightNumber
        HAVING COUNT(f2.FlightNumber) > 1
```

```
AND f3.DepartureDate > f.DepartureDate AND airport1.AirportCity = %s AND f.DepartureAirport = %s
AND airport2.AirportCity = %s AND f.ArrivalAirport = %s AND f.DepartureDate = %s AND
f3.DepartureDate = %s
GROUP BY f.AirlineName, f.FlightNumber, f.DepartureDate, f.DepartureTime, f.ArrivalDate,
f.FlightStatus, returnDate, returnTime
HAVING numFlights < f.numberOfSeats
```

- **Purchase tickets**

Similar to how customers can search for one way and round trip flights, customers can also purchase flights based on if they are one way or round trip.

```
SELECT f.AirlineName, f.FlightNumber, f.DepartureDate, f.DepartureTime, f.BasePrice, f.ArrivalDate,
f.ArrivalTime, f.ArrivalAirport, f.DepartureAirport, COUNT(ticketID) AS bookedSeats, f.numberOfSeats
FROM Flight AS f
LEFT JOIN PurchasedFor AS pf
ON pf.FlightNumber = f.FlightNumber AND pf.DepartureDate = f.DepartureDate AND pf.DepartureTime
= f.DepartureTime
INNER JOIN Updates AS u
ON u.FlightNumber = f.FlightNumber AND u.DepartureDate = f.DepartureDate AND u.DepartureTime =
f.DepartureTime
INNER JOIN Airplane
ON f.AirplaneID = Airplane.AirplaneID
INNER JOIN Airport AS airport1
ON airport1.AirportName = f.DepartureAirport
INNER JOIN Airport AS airport2
ON airport2.AirportName = f.ArrivalAirport
WHERE f.FlightNumber NOT IN
        SELECT FlightNumber from Flight AS f2
        GROUP BY FlightNumber
        HAVING COUNT(f2.FlightNumber) > 1
AND f.DepartureDate = %s AND f.DepartureTime = %s AND f.FlightNumber = %s
GROUP BY f.AirlineName, f.FlightNumber, f.DepartureDate, f.DepartureTime, f.ArrivalDate,
f.FlightStatus
HAVING bookedSeats < f.numberOfSeats

SELECT f.FlightNumber, f.DepartureDate, f.DepartureTime, COUNT(ticketID) AS bookedSeats,
numberOfSeats
FROM flight AS f
NATURAL JOIN airplane LEFT JOIN Purchasedfor AS pf
```

```
ON f.DepartureDate = pf.DepartureDate AND f.DepartureTime = pf.DepartureTime AND f.FlightNumber
= pf.FlightNumber
WHERE f.flightNumber = %s AND f.departureDate = %s AND f.departureTime = %s
GROUP BY f.FlightNumber, f.DepartureDate, f.DepartureTime, numberOfSeats
```

- **Cancel trip**

Customers have the option to cancel trips that take place at least 24 hours later. The query below looks for flights that the customer has bought that will occur at least 24 hours later.

```
SELECT FlightNumber, DepartureDate
FROM Ticket NATURAL JOIN PurchasedFor NATURAL JOIN Customer
WHERE CustomerEmail = %s AND TicketID = %s AND (CURRENT_DATE < DepartureDate -
INTERVAL 1 DAY)
```

This query here performs the action of canceling the selected trip by deleting the entry from the PurchasedFor table.

```
DELETE FROM PurchasedFor
WHERE TicketID = %s AND FlightNumber = %s AND DepartureDate = %s AND DepartureTime = %s
```

- **Give ratings and comments on previous flights**

Customers have the option to rate and comment on previous flights for the airline they are logged in. The query below checks that the flight exists and was bought by the customer.

```
SELECT FlightNumber, DepartureDate, DepartureTime
FROM Ticket NATURAL JOIN PurchasedFor NATURAL JOIN Customer
WHERE CustomerEmail = %s AND TicketID = %s AND (CURRENT_DATE > DepartureDate OR
(CURRENT_DATE = DepartureDate AND CURRENT_TIME > DepartureTime)))
```

The query below checks if a flight has not been rated by the customer already.

```
SELECT FlightNumber, DepartureDate, DepartureTime, TicketID
FROM Suggested NATURAL JOIN Ticket
WHERE CustomerEmail = %s AND TicketID = %s
```

- **Track my spending**

Customers can track their spending by year or by month. The following query shows the amount the customer spent in the last year.

```
SELECT SUM(SoldPrice) AS spend
FROM ticket NATURAL JOIN PurchasedFor
WHERE CustomerEmail = %s AND PurchaseDate >= CURRENT_DATE - INTERVAL 1 YEAR
```

The following query shows the amount the customer spent in the last 6 months.

```
SELECT MONTHNAME(PurchaseDate) AS month, SUM(soldPrice) as spent
FROM ticket NATURAL JOIN PurchasedFor
WHERE CustomerEmail = %s AND PurchaseDate >= CURRENT_DATE - INTERVAL 6 MONTH
GROUP BY MONTHNAME(PurchaseDate)
```

Airline Staff Use Cases

- **View flights**

An airline staff member has the option to view all the flights of the airline he or she is working at. The following query returns the airline that the staff member works at.

```
SELECT AirlineName FROM AirlineStaff
WHERE Username = %s
```

The following query returns the flights that will depart within the next 30 days.

```
SELECT DISTINCT FlightNumber, DepartureDate, DepartureTime, DepartureAirport, ArrivalDate,
ArrivalTime
FROM AirlineStaff NATURAL JOIN Flight
WHERE AirlineName = %s AND (DepartureDate <= CURRENT_DATE + INTERVAL 30 DAY AND
(DepartureDate > CURRENT_DATE)) OR (DepartureDate = CURRENT_DATE AND DepartureTime >
CURRENT_TIME)
```

- **Create new flights**

Airline staff have the ability to create new flights by providing all the necessary information. The query below checks if the flight already exists.

```
SELECT * FROM Flight
WHERE FlightNumber = %s AND DepartureDate = %s AND DepartureTime = %s
```

The following queries inserts the necessary information into the database tables.

```
INSERT INTO Flight VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
INSERT INTO Arrives VALUES(%s, %s, %s, %s)
INSERT INTO Departs VALUES(%s, %s, %s, %s)
INSERT INTO Changes VALUES(%s, %s, %s, %s, %s)
```

- **Change flight status**
  Airline staff can update the status of any flight. The query below checks if the flight exists.

  SELECT * FROM Changes
  WHERE Username = %s AND FlightNumber = %s AND DepartureDate = %s AND DepartureTime = %s

  The following query changes the status of the specified flight in the “Updates” table.
  UPDATE Updates SET Username = %s, FlightStatus = %s
  WHERE FlightNumber = %s AND DepartureDate = %s AND DepartureTime = %s
- **Add new airplane**
  Airline staff can add new airplanes to the system. The query below checks if the airplane already exists in the database.

  SELECT * FROM Airplane
  WHERE AirlineName = %s AND AirplaneID = %s

  The following query inserts a new airplane into the “Airplane” table.
  INSERT INTO Airplane VALUES(%s, %s, %s, %s, %s)

  The following query returns a list of airplanes for a specific airline.
  SELECT * FROM Airplane
  WHERE airlineName = %s

- **Add new airport**
  Airline staff can add new airports to the system. The query below checks if the airport already exists.

  SELECT * FROM Airport
  WHERE AirportName = %s

  The following query inserts a new airport into the “Airport” table.
  INSERT INTO Airport Values(%s, %s, %s, %s, %s)

- **View flight ratings**
  Airline staff can view flight ratings left by customers. The following query returns these flight ratings.

  SELECT CustomerComment, Rate FROM Suggested
  WHERE FlightNumber = %s AND DepartureDate = %s AND DepartureTime = %s

- **View frequent customers**

Airline staff can view the most frequent customer within the last year. The following query returns the top customer based on the number of flights theyve taken.

```
SELECT CustomerEmail, COUNT(*) AS numFlights
FROM Ticket
WHERE AirlineName = %s AND purchaseDate >= CURRENT_DATE - INTERVAL 1 YEAR
GROUP BY CustomerEmail
ORDER BY numFlights DESC
LIMIT 1
```

The following query returns a list of all the flights a customer has taken.

```
SELECT FlightNumber, DepartureDate, DepartureTime
FROM Flight NATURAL JOIN Updates NATURAL JOIN PurchasedFor NATURAL JOIN Ticket
NATURAL JOIN Customer
WHERE CustomerEmail = %s AND AirlineName = %s
```

- **View reports**
  Airline staff can view reports which show the total number of tickets sold based on a range of dates. The following queries return the number of tickets sold by month and by year.

```
SELECT COUNT(TicketID) AS totalTickets, MONTHNAME(PurchaseDate) AS Month
FROM Ticket
WHERE PurchaseDate >= CURRENT_DATE - INTERVAL 1 MONTH GROUP BY Month
```

```
SELECT COUNT(TicketID) AS totalTickets, MONTHNAME(PurchaseDate) AS Month
FROM Ticket
WHERE PurchaseDate >= CURRENT_DATE - INTERVAL 1 YEAR GROUP BY Month
```

- **View earned revenue**
  Airline staff can view the total earned revenue from tickets by month or by year.

```
SELECT SUM(SoldPrice) AS Sale
FROM Ticket
WHERE PurchaseDate >= CURRENT_DATE - INTERVAL 1 MONTH
```

```
SELECT Sum(SoldPrice) As Sale
FROM Ticket
WHERE PurchaseDate >= CURRENT_DATE - INTERVAL 1 YEAR
```

- **View earned revenue by travel class**
  Airline staff can view the total earned revenue from tickets by travel class.

  ```
  SELECT Sum(SoldPrice) AS Sale
  FROM Ticket
  WHERE PurchaseDate <= CURRENT_DATE
  GROUP BY TravelClass
  ```

- **View top destinations**
  Airline staff can view the most visited destinations. The following queries show the 3 most popular places for the last 3 months and for last year.

  ```
  SELECT AirportCity
  FROM TICKET NATURAL JOIN PurchasedFor NATURAL JOIN Flight INNER JOIN Airport
  ON arrivalAirport = airport.AirportName
  WHERE AirlineName = %s AND arrivalDate >= CURRENT_DATE - INTERVAL 3 MONTH
  GROUP BY AirportName
  ORDER BY Count(AirportName) DESC
  LIMIT 3
  ```

  ```
  SELECT AirportCity
  FROM TICKET NATURAL JOIN PurchasedFor NATURAL JOIN Flight INNER JOIN Airport
  ON arrivalAirport = airport.AirportName
  WHERE AirlineName = %s AND arrivalDate >= CURRENT_DATE - INTERVAL 1 YEAR
  GROUP BY AirportName
  ORDER BY Count(AirportName) DESC
  LIMIT 3
  ```