```
SELECT * FROM flights;



CREATE OR REPLACE PROCEDURE insert_flight(
    p_flight_no VARCHAR,
    p_scheduled_departure DATE,
    p_scheduled_arrival DATE,
    p_departure_airport_id INT,
    p_arrival_airport_id INT,
    p_departing_gate VARCHAR,
    p_arriving_gate VARCHAR,
    p_airline_id INT,
    p_status VARCHAR,
    p_actual_departure DATE,
    p_actual_arrival DATE,
    p_created_at DATE,
    p_update_at DATE
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO flights(
        flight_no, scheduled_departure, scheduled_arrival,
        departure_airport_id, arrival_airport_id,
        departing_gate, arriving_gate, airline_id,
        status, actual_departure, actual_arrival,
        created_at, update_at
    )
    VALUES(
        p_flight_no, p_scheduled_departure, p_scheduled_arrival,
        p_departure_airport_id, p_arrival_airport_id,
        p_departing_gate, p_arriving_gate, p_airline_id,
```

```
    p_status, p_actual_departure, p_actual_arrival,
    p_created_at, p_update_at
  );
END;
$$;


SELECT * FROM flights;


CREATE OR REPLACE PROCEDURE task2(
p_flight_id INT, p_status VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
UPDATE flights SET status = p_status WHERE flight_id = p_flight_id;
END;
$$;


CREATE OR REPLACE PROCEDURE task33(
  p_departure_airport_id INT
)
LANGUAGE plpgsql
AS $$
BEGIN
  SELECT
    flight_id,
    flight_no,
    scheduled_departure,
    scheduled_arrival,
```

```
    arrival_airport_id,
    status
  FROM flights
  WHERE departure_airport_id = p_departure_airport_id;
END;
$$;


CREATE OR REPLACE FUNCTION task4(
  p_arrival_airport_id INT
)
RETURNS INTERVAL
LANGUAGE plpgsql
AS $$
DECLARE
  avg_time INTERVAL;
BEGIN
  SELECT AVG(actual_arrival - scheduled_arrival)
  INTO avg_time
  FROM flights
  WHERE arrival_airport_id = p_arrival_airport_id;

  RETURN avg_time;
END;
$$;


CREATE OR REPLACE FUNCTION task5(
p_flight_id INT
)
RETURNS TABLE (
  passenger_id INT,
```

```
    first_name VARCHAR,
    last_name VARCHAR,
    country_of_citizenship VARCHAR
)

LANGUAGE plpgsql
AS $$
BEGIN
RETURN QUERY
 SELECT p.passenger_id,
        p.first_name,
        p.last_name,
        p.country_of_citizenship FROM booking_flight bf
 JOIN booking b ON bf.booking_id = b.booking_id
 JOIN passengers p ON p.passenger_id = b.passenger_id
 WHERE bf.flight_id = p_flight_id;
 END;
 $$;




CREATE OR REPlACE FUNCTION task6()
RETURNS TABLE(
    passenger_id INT,
    first_name VARCHAR,
    last_name VARCHAR,
    country_of_citizenship VARCHAR,
    flights_taken INT
)
LANGUAGE plpgsql
AS $$
```

```
BEGIN
RETURN QUERY
SELECT
        p.passenger_id,
        p.first_name,
        p.last_name,
        p.country_of_citizenship,
        COUNT(b.booking_id) AS flights_taken
FROM passengers p JOIN booking b ON b.passenger_id = p.passenger_id
GROUP BY p.passenger_id
ORDER BY flights_taken DESC
LIMIT 1;
END;
$$;



CREATE OR REPLACE PROCEDURE task77()
LANGUAGE plpgsql
AS $$
BEGIN
  SELECT
    flight_id,
    flight_no,
    actual_departure,
    actual_arrival
  FROM flights
  WHERE (actual_arrival - actual_departure) > INTERVAL '1 day'
   AND status = 'Delayed';
END;
$$;
```

```
CREATE OR REPLACE FUNCTION task8()
RETURNS TABLE(
airline_name VARCHAR,
airline_id INT,
cnt INT
)
LANGUAGE plpgsql
AS $$
BEGIN
RETURN QUERY
SELECT a.airline_name, a.airline_id, COUNT(f.airline_id) AS cnt
FROM airline a JOIN flights f ON a.airline_id = f.airline_id
GROUP BY a.airline_name, a.airline_id;
END;
$$;
```

```
CREATE OR REPLACE PROCEDURE task99(
   p_flight_id INT
)
LANGUAGE plpgsql
AS $$
BEGIN
   SELECT
      f.flight_id,
      AVG(b.price) AS avg_price
   FROM booking_flight bf
   JOIN booking b ON b.booking_id = bf.booking_id
   JOIN flights f ON f.flight_id = bf.flight_id
```

```
    WHERE bf.flight_id = p_flight_id
    GROUP BY f.flight_id;
END;
$$;



CREATE OR REPLACE PROCEDURE task101()
LANGUAGE plpgsql
AS $$
BEGIN
    SELECT
        f.flight_id,
        f.departure_airport_id,
        f.arrival_airport_id,
        b.price AS ticket_price
    FROM booking_flight bf
    JOIN booking b ON bf.booking_id = b.booking_id
    JOIN flights f ON f.flight_id = bf.flight_id
    ORDER BY b.price DESC
    LIMIT 1;
END;
$$;
```