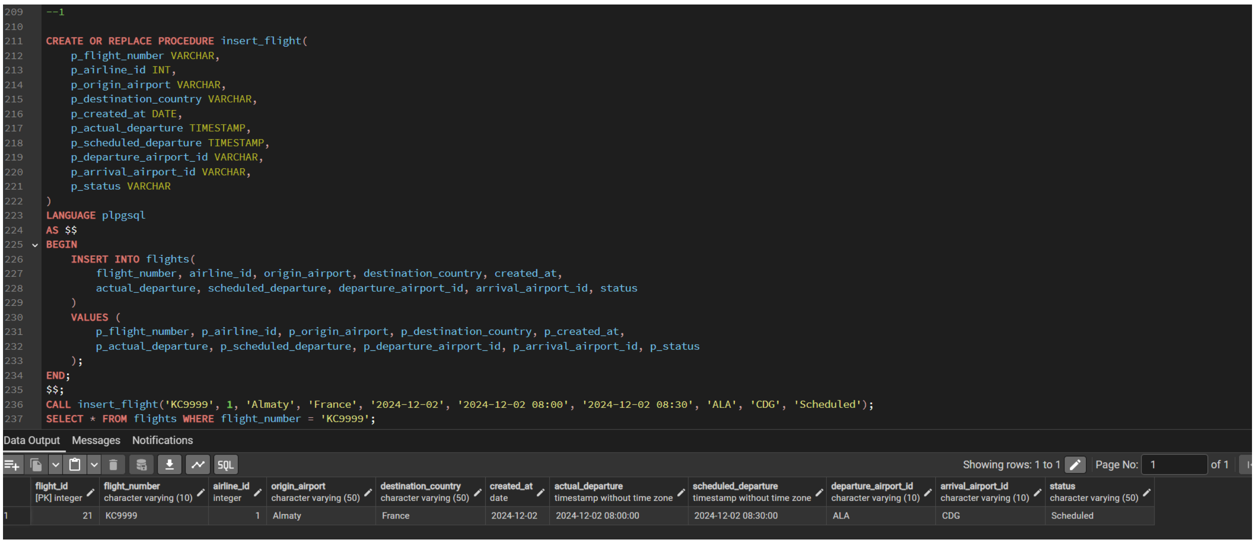
Lab-10

1. Create a stored procedure to insert a new flight into the flights table.

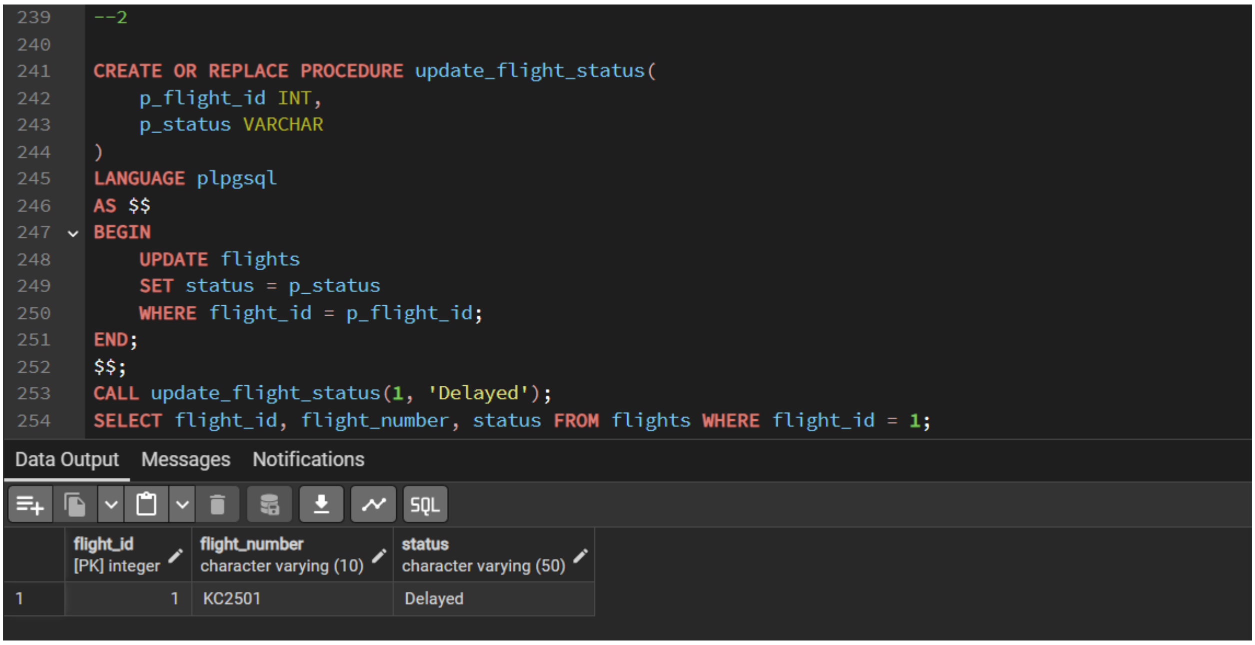


```sql
--1

CREATE OR REPLACE PROCEDURE insert_flight(
    p_flight_number VARCHAR,
    p_airline_id INT,
    p_origin_airport VARCHAR,
    p_destination_country VARCHAR,
    p_created_at DATE,
    p_actual_departure TIMESTAMP,
    p_scheduled_departure TIMESTAMP,
    p_departure_airport_id VARCHAR,
    p_arrival_airport_id VARCHAR,
    p_status VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO flights(
        flight_number, airline_id, origin_airport, destination_country, created_at,
        actual_departure, scheduled_departure, departure_airport_id, arrival_airport_id, status
    )
    VALUES (
        p_flight_number, p_airline_id, p_origin_airport, p_destination_country, p_created_at,
        p_actual_departure, p_scheduled_departure, p_departure_airport_id, p_arrival_airport_id, p_status
    );
END;
$$;
CALL insert_flight('KC9999', 1, 'Almaty', 'France', '2024-12-02', '2024-12-02 08:00', '2024-12-02 08:30', 'ALA', 'CDG', 'Scheduled');
SELECT * FROM flights WHERE flight_number = 'KC9999';
```

	flight_id [PK] integer	flight_number character varying (10)	airline_id integer	origin_airport character varying (50)	destination_country character varying (50)	created_at date	actual_departure timestamp without time zone	scheduled_departure timestamp without time zone	departure_airport_id character varying (10)	arrival_airport_id character varying (10)	status character varying (50)
1	21	KC9999	1	Almaty	France	2024-12-02	2024-12-02 08:00:00	2024-12-02 08:30:00	ALA	CDG	Scheduled

2. Create a stored procedure to update the status of a flight.

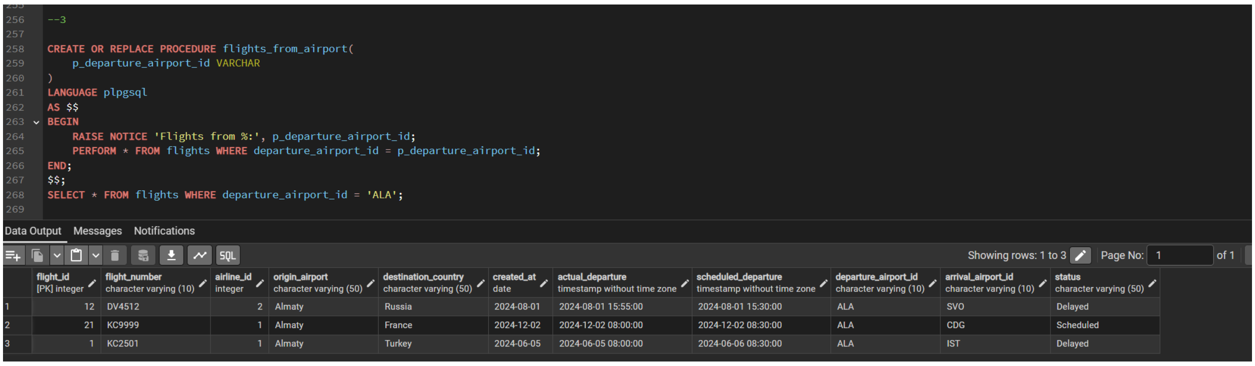


```sql
--2

CREATE OR REPLACE PROCEDURE update_flight_status(
    p_flight_id INT,
    p_status VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE flights
    SET status = p_status
    WHERE flight_id = p_flight_id;
END;
$$;
CALL update_flight_status(1, 'Delayed');
SELECT flight_id, flight_number, status FROM flights WHERE flight_id = 1;
```

	flight_id [PK] integer	flight_number character varying (10)	status character varying (50)
1	1	KC2501	Delayed

3. Create a stored procedure that returns a list of flights departing from a specific airport.

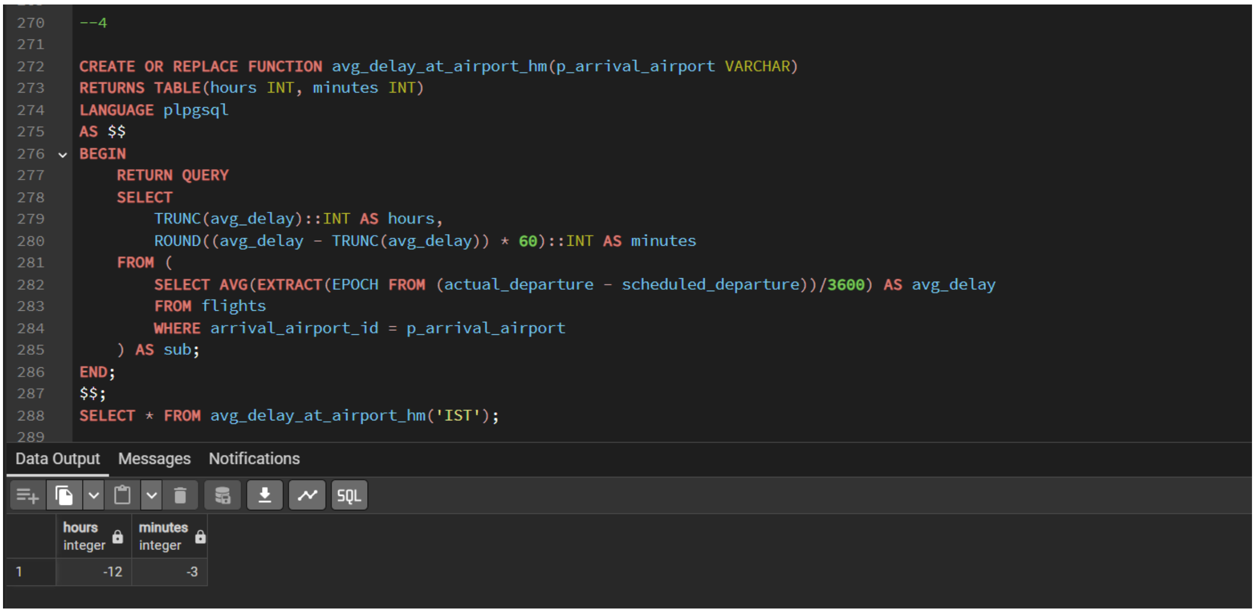


```
--3

CREATE OR REPLACE PROCEDURE flights_from_airport(
    p_departure_airport_id VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    RAISE NOTICE 'Flights from %:', p_departure_airport_id;
    PERFORM * FROM flights WHERE departure_airport_id = p_departure_airport_id;
END;
$$;
SELECT * FROM flights WHERE departure_airport_id = 'ALA';
```

	flight_id [PK] integer	flight_number character varying (10)	airline_id integer	origin_airport character varying (50)	destination_country character varying (50)	created_at date	actual_departure timestamp without time zone	scheduled_departure timestamp without time zone	departure_airport_id character varying (10)	arrival_airport_id character varying (10)	status character varying (50)
1	12	DV4512	2	Almaty	Russia	2024-08-01	2024-08-01 15:55:00	2024-08-01 15:30:00	ALA	SVO	Delayed
2	21	KC9999	1	Almaty	France	2024-12-02	2024-12-02 08:00:00	2024-12-02 08:30:00	ALA	CDG	Scheduled
3	1	KC2501	1	Almaty	Turkey	2024-06-05	2024-06-05 08:00:00	2024-06-06 08:30:00	ALA	IST	Delayed

4. Create a function to calculate the average delay time of flights arriving at a specific airport.

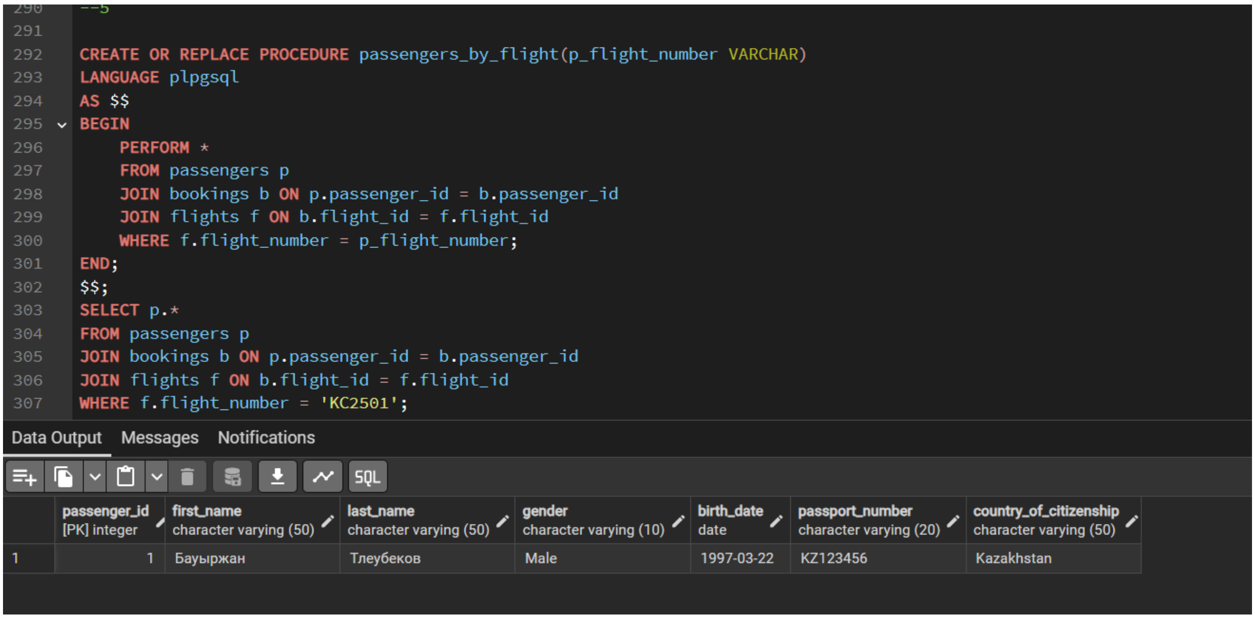


```
--4

CREATE OR REPLACE FUNCTION avg_delay_at_airport_hm(p_arrival_airport VARCHAR)
RETURNS TABLE(hours INT, minutes INT)
LANGUAGE plpgsql
AS $$
BEGIN
    RETURN QUERY
    SELECT
        TRUNC(avg_delay)::INT AS hours,
        ROUND((avg_delay - TRUNC(avg_delay)) * 60)::INT AS minutes
    FROM (
        SELECT AVG(EXTRACT(EPOCH FROM (actual_departure - scheduled_departure))/3600) AS avg_delay
        FROM flights
        WHERE arrival_airport_id = p_arrival_airport
    ) AS sub;
END;
$$;
SELECT * FROM avg_delay_at_airport_hm('IST');
```

	hours integer	minutes integer
1	-12	-3

5. Create a stored procedure that lists all passengers for a given flight number.

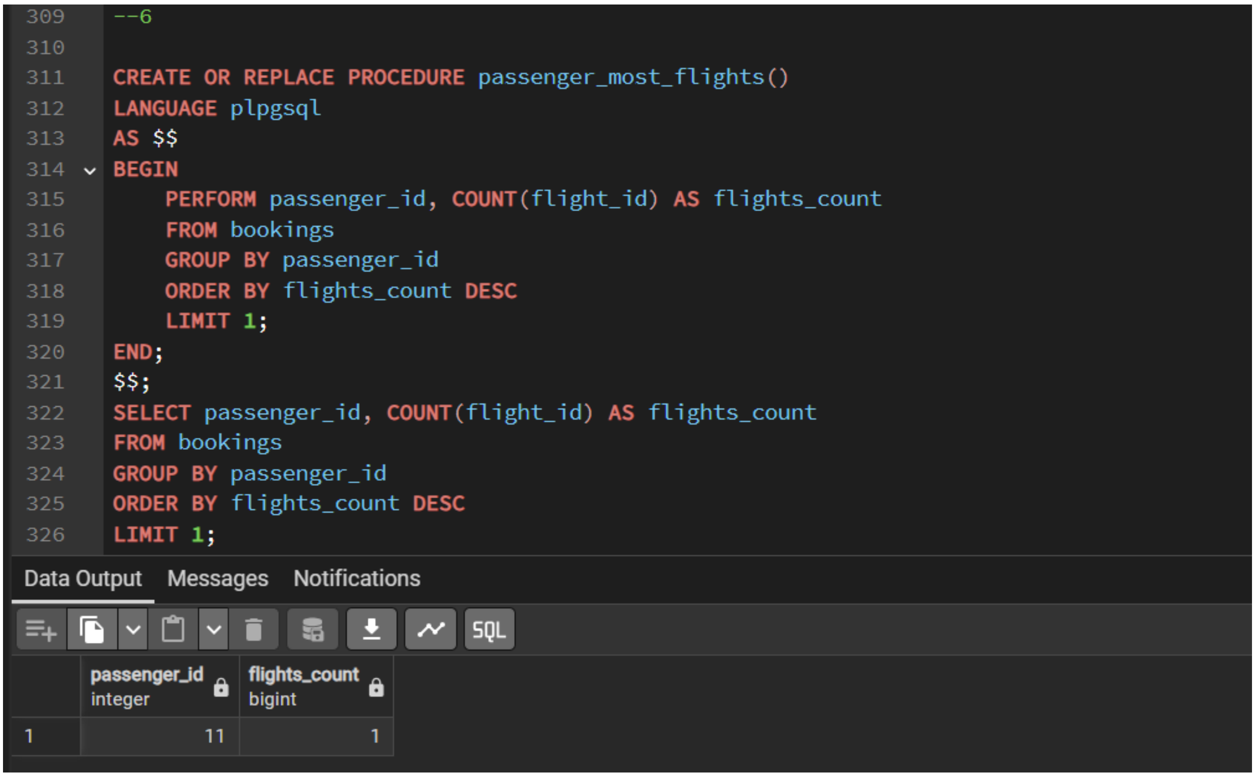


```
--5

CREATE OR REPLACE PROCEDURE passengers_by_flight(p_flight_number VARCHAR)
LANGUAGE plpgsql
AS $$
BEGIN
    PERFORM *
    FROM passengers p
    JOIN bookings b ON p.passenger_id = b.passenger_id
    JOIN flights f ON b.flight_id = f.flight_id
    WHERE f.flight_number = p_flight_number;
END;
$$;
SELECT p.*
FROM passengers p
JOIN bookings b ON p.passenger_id = b.passenger_id
JOIN flights f ON b.flight_id = f.flight_id
WHERE f.flight_number = 'KC2501';
```

	passenger_id [PK] integer	first_name character varying (50)	last_name character varying (50)	gender character varying (10)	birth_date date	passport_number character varying (20)	country_of_citizenship character varying (50)
1	1	Бауыржан	Тлеубеков	Male	1997-03-22	KZ123456	Kazakhstan

6. Create a stored procedure to find the passenger who has taken the greatest number of flights.

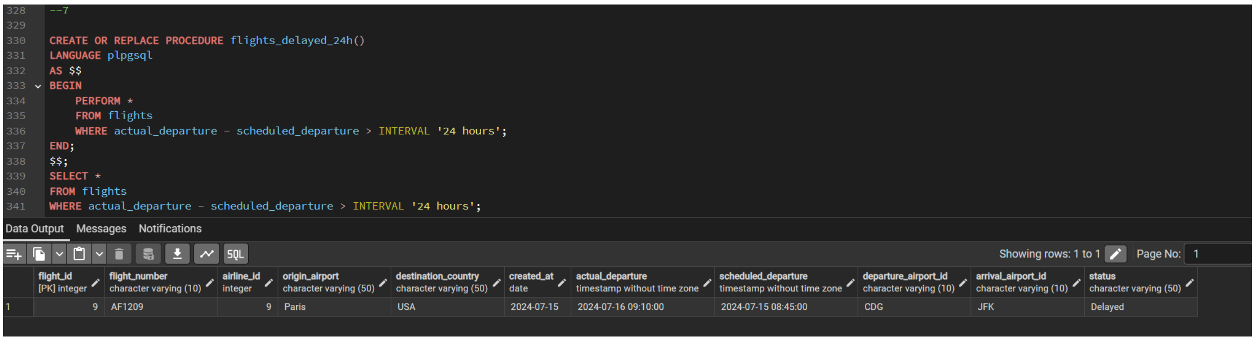


```
--6

CREATE OR REPLACE PROCEDURE passenger_most_flights()
LANGUAGE plpgsql
AS $$
BEGIN
    PERFORM passenger_id, COUNT(flight_id) AS flights_count
    FROM bookings
    GROUP BY passenger_id
    ORDER BY flights_count DESC
    LIMIT 1;
END;
$$;
SELECT passenger_id, COUNT(flight_id) AS flights_count
FROM bookings
GROUP BY passenger_id
ORDER BY flights_count DESC
LIMIT 1;
```

	passenger_id integer	flights_count bigint
1	11	1

7. Create a stored procedure to find all flights that are delayed by more than 24 hours.

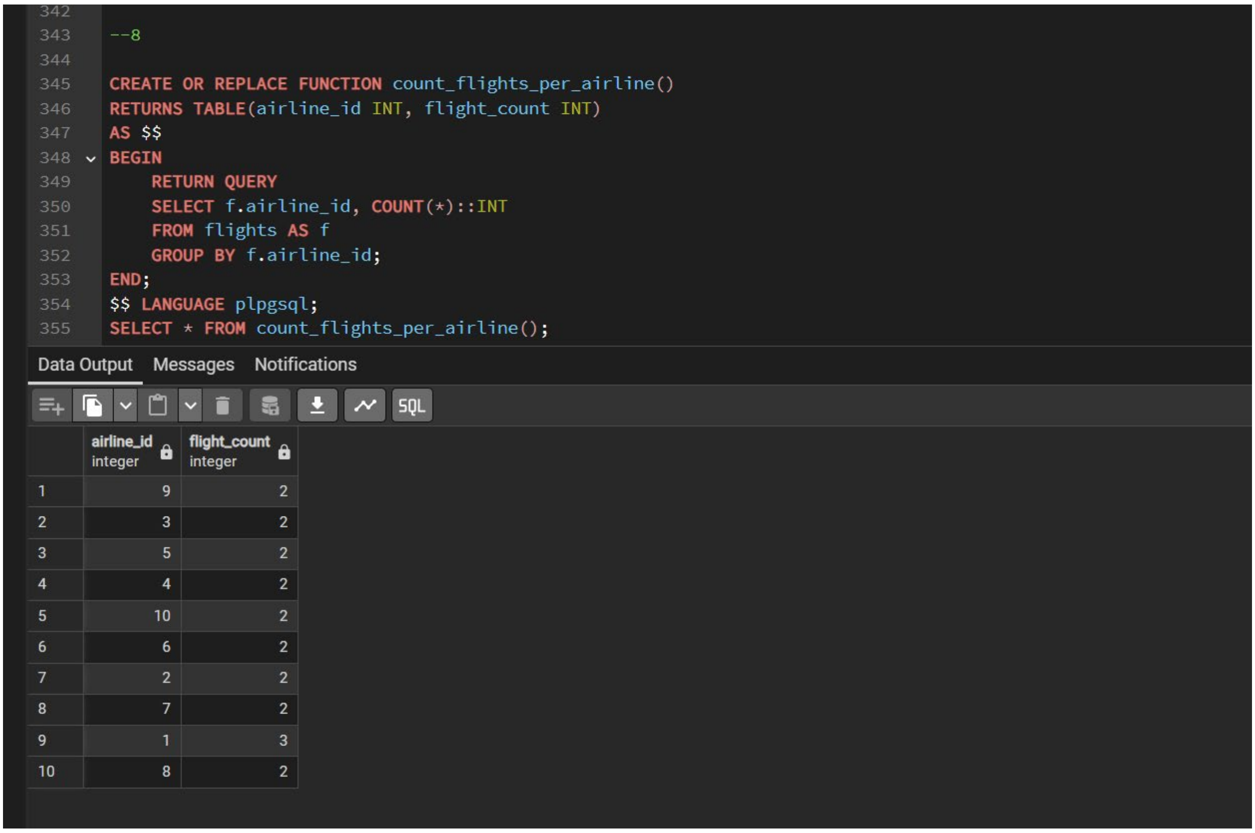


```
--7

CREATE OR REPLACE PROCEDURE flights_delayed_24h()
LANGUAGE plpgsql
AS $$
BEGIN
    PERFORM *
    FROM flights
    WHERE actual_departure - scheduled_departure > INTERVAL '24 hours';
END;
$$;
SELECT *
FROM flights
WHERE actual_departure - scheduled_departure > INTERVAL '24 hours';
```

	flight_id [PK] integer	flight_number character varying (10)	airline_id integer	origin_airport character varying (50)	destination_country character varying (50)	created_at date	actual_departure timestamp without time zone	scheduled_departure timestamp without time zone	departure_airport_id character varying (10)	arrival_airport_id character varying (10)	status character varying (50)
1	9	AF1209	9	Paris	USA	2024-07-15	2024-07-16 09:10:00	2024-07-15 08:45:00	CDG	JFK	Delayed

8. Create a function that counts the number of flights for each airline.

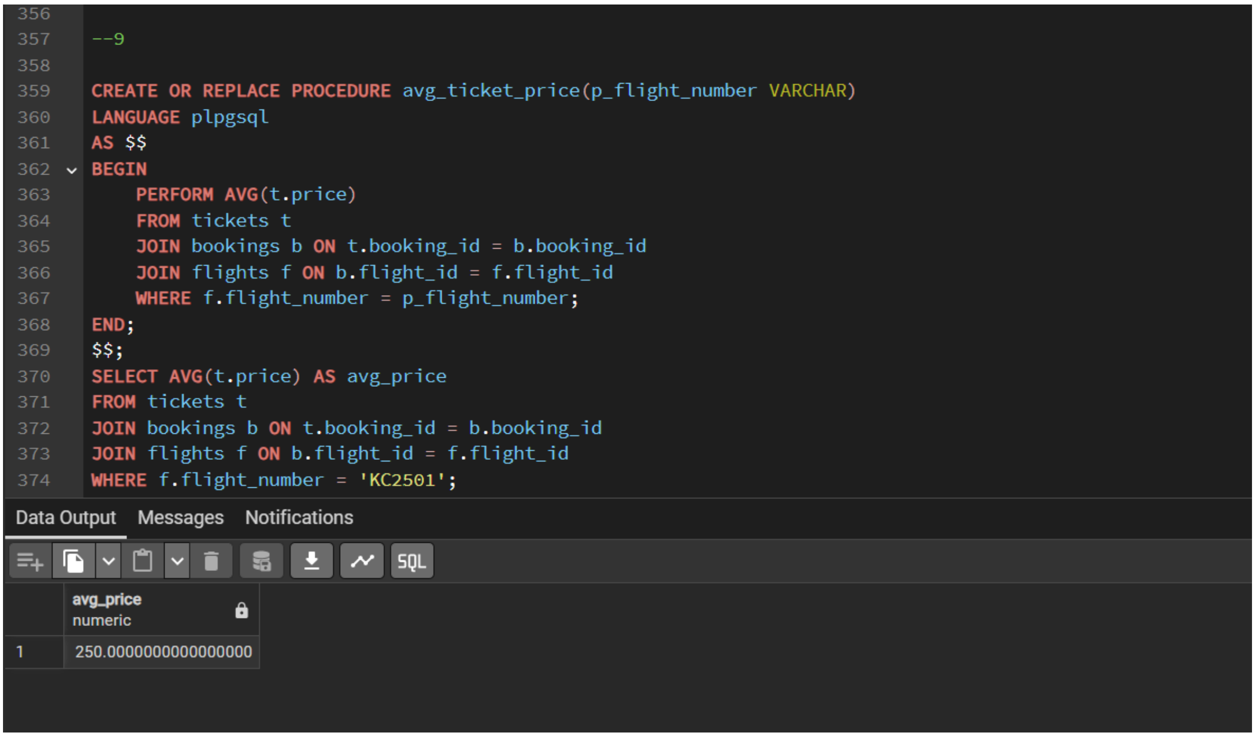


```
--8

CREATE OR REPLACE FUNCTION count_flights_per_airline()
RETURNS TABLE(airline_id INT, flight_count INT)
AS $$
BEGIN
    RETURN QUERY
    SELECT f.airline_id, COUNT(*)::INT
    FROM flights AS f
    GROUP BY f.airline_id;
END;
$$ LANGUAGE plpgsql;
SELECT * FROM count_flights_per_airline();
```

	airline_id integer	flight_count integer
1	9	2
2	3	2
3	5	2
4	4	2
5	10	2
6	6	2
7	2	2
8	7	2
9	1	3
10	8	2

9. Create a stored procedure to calculate the average ticket price for a specific flight.

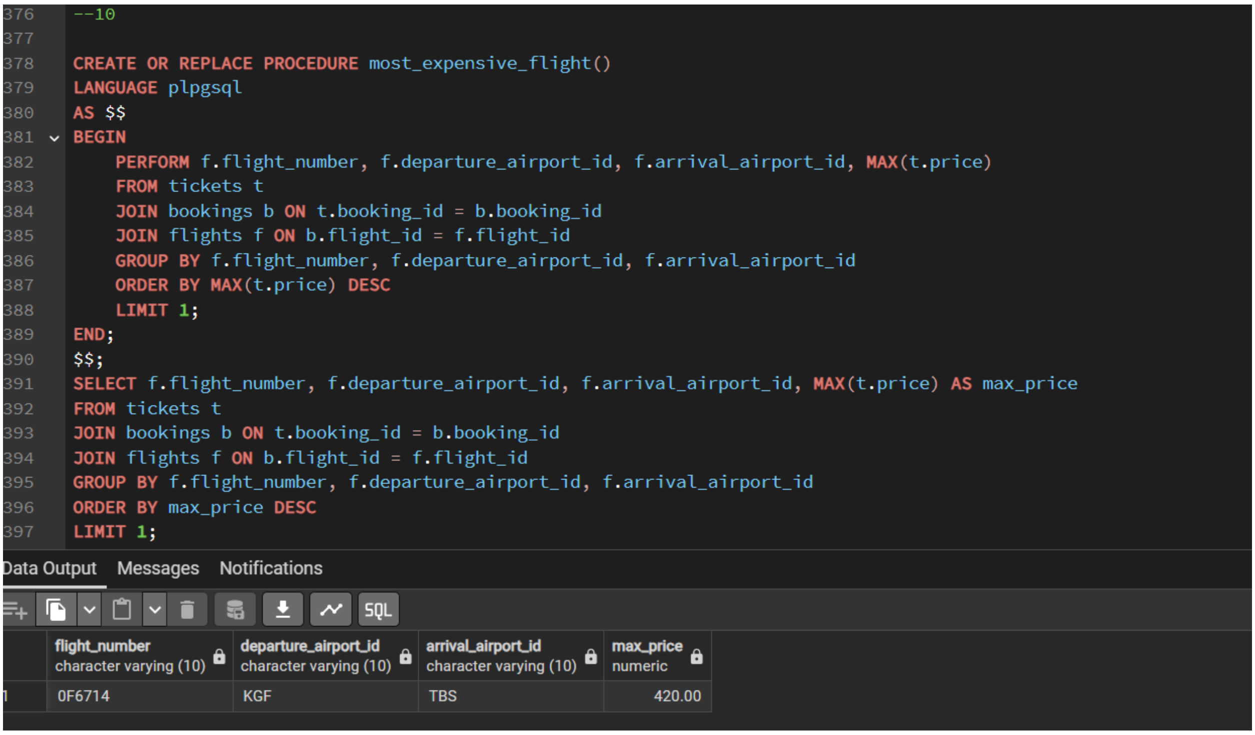


```
--9

CREATE OR REPLACE PROCEDURE avg_ticket_price(p_flight_number VARCHAR)
LANGUAGE plpgsql
AS $$
BEGIN
    PERFORM AVG(t.price)
    FROM tickets t
    JOIN bookings b ON t.booking_id = b.booking_id
    JOIN flights f ON b.flight_id = f.flight_id
    WHERE f.flight_number = p_flight_number;
END;
$$;
SELECT AVG(t.price) AS avg_price
FROM tickets t
JOIN bookings b ON t.booking_id = b.booking_id
JOIN flights f ON b.flight_id = f.flight_id
WHERE f.flight_number = 'KC2501';
```

	avg_price numeric
1	250.0000000000000000

10.Create a stored procedure to find the flight with the highest ticket price. The procedure should return the flight number, the departure and arrival airports, and the ticket price for the most expensive flight.



```
--10

CREATE OR REPLACE PROCEDURE most_expensive_flight()
LANGUAGE plpgsql
AS $$
BEGIN
    PERFORM f.flight_number, f.departure_airport_id, f.arrival_airport_id, MAX(t.price)
    FROM tickets t
    JOIN bookings b ON t.booking_id = b.booking_id
    JOIN flights f ON b.flight_id = f.flight_id
    GROUP BY f.flight_number, f.departure_airport_id, f.arrival_airport_id
    ORDER BY MAX(t.price) DESC
    LIMIT 1;
END;
$$;
SELECT f.flight_number, f.departure_airport_id, f.arrival_airport_id, MAX(t.price) AS max_price
FROM tickets t
JOIN bookings b ON t.booking_id = b.booking_id
JOIN flights f ON b.flight_id = f.flight_id
GROUP BY f.flight_number, f.departure_airport_id, f.arrival_airport_id
ORDER BY max_price DESC
LIMIT 1;
```

	flight_number character varying (10)	departure_airport_id character varying (10)	arrival_airport_id character varying (10)	max_price numeric
1	0F6714	KGF	TBS	420.00