

1)

Query	Query History
<pre>1 CREATE OR REPLACE PROCEDURE insert_flight(2 p_flight_number VARCHAR, 3 p_airline_id INT, 4 p_departure_airport_id INT, 5 p_arrival_airport_id INT, 6 p_departure_time TIMESTAMP, 7 p_arrival_time TIMESTAMP, 8 p_status VARCHAR DEFAULT 'Scheduled', 9 p_aircraft VARCHAR DEFAULT NULL, 10 p_price NUMERIC DEFAULT NULL 11) 12 LANGUAGE plpgsql 13 AS \$\$ 14 BEGIN 15 INSERT INTO flights (16 flight_number, airline_id, departure_airport_id, arrival_airport_id, 17 departure_time, arrival_time, status, aircraft, price 18) VALUES (19 p_flight_number, p_airline_id, p_departure_airport_id, p_arrival_airport_id, 20 p_departure_time, p_arrival_time, p_status, p_aircraft, p_price 21); 22 END; 23 \$\$;</pre>	
Data Output	Messages Notifications
CREATE PROCEDURE	
Query returned successfully in 158 msec.	

2)

Query	Query History
<pre>1 CREATE OR REPLACE PROCEDURE update_flight_status(2 p_flight_id INT, 3 p_new_status VARCHAR 4) 5 LANGUAGE plpgsql 6 AS \$\$ 7 BEGIN 8 UPDATE flights SET status = p_new_status WHERE flight_id = p_flight_id; 9 END; 10 \$\$;</pre>	
Data Output	Messages Notifications
CREATE PROCEDURE	
Query returned successfully in 53 msec.	

✓ Query returned successfully

Total rows: Querv complete 00:00:00.053

3)

Query

Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

```
CREATE OR REPLACE FUNCTION get_flights_from_airport_func(p_airport_id INT)
RETURNS TABLE(
    flight_id INT,
    flight_number VARCHAR,
    departure_airport TEXT,
    arrival_airport TEXT,
    departure_time TIMESTAMP,
    arrival_time TIMESTAMP,
    status VARCHAR,
    airline_name VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    RETURN QUERY
    SELECT f.flight_id, f.flight_number,
           d.airport_name::TEXT, a.airport_name::TEXT,
           f.departure_time, f.arrival_time, f.status,
           al.airline_name
    FROM flights f
    JOIN airport d ON f.departure_airport_id = d.airport_id
    JOIN airport a ON f.arrival_airport_id = a.airport_id
    JOIN airline al ON f.airline_id = al.airline_id
    WHERE f.departure_airport_id = p_airport_id
    ORDER BY f.departure_time;
END;
$$;
```

Data Output

Messages

Notifications

CREATE PROCEDURE

Query returned successfully in 44 msec.

Total rows:

Query complete 00:00:00.044

4)

Query Query History

```
1 CREATE OR REPLACE FUNCTION avg_arrival_delay(p_airport_id INT)
2 RETURNS NUMERIC
3 LANGUAGE plpgsql
4 AS $$
5 DECLARE
6     result NUMERIC;
7 BEGIN
8     SELECT AVG(EXTRACT(EPOCH FROM (f.actual_arrival_time - f.arrival_time))/60)
9     INTO result
10    FROM flights f
11   WHERE f.arrival_airport_id = p_airport_id
12         AND f.actual_arrival_time IS NOT NULL
13         AND f.actual_arrival_time > f.arrival_time;
14
15     RETURN COALESCE(result, 0);
16 END;
17 $$;
```

Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 40 msec.

✓ Query ret

Total rows: Query complete 00:00:00.040

5)

Query

Query History

1

CREATE OR REPLACE FUNCTION get_passengers_by_flight(p_flight_id INT)

2

RETURNS TABLE(

3

passenger_id INT,

4

first_name VARCHAR,

5

last_name VARCHAR,

6

email VARCHAR,

7

seat_number VARCHAR,

8

class VARCHAR

9

)

10

LANGUAGE plpgsql

11

AS \$\$

12

BEGIN

13

RETURN QUERY

14

SELECT p.passenger_id, p.first_name, p.last_name, p.email,

15

bp.seat_number, bp.class

16

FROM booking_flight bf

17

JOIN booking b ON bf.booking_id = b.booking_id

18

JOIN passengers p ON b.passenger_id = p.passenger_id

19

JOIN boarding_pass bp ON b.booking_id = bp.booking_id

20

WHERE bf.flight_id = p_flight_id

21

ORDER BY p.last_name, p.first_name;

22

END;

23

\$\$;

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 38 msec.

✓ Query ret

Total rows:

Query complete 00:00:00.038

6)

Query

Query History

1

CREATE OR REPLACE FUNCTION get_top_passenger()

2

RETURNS TABLE(

3

passenger_id INT,

4

first_name VARCHAR,

5

last_name VARCHAR,

6

flights_count BIGINT

7

)

8

LANGUAGE plpgsql

9

AS \$\$

10

▼ BEGIN

11

RETURN QUERY

12

SELECT p.passenger_id, p.first_name, p.last_name, COUNT(*)::BIGINT

13

FROM passengers p

14

JOIN booking b ON p.passenger_id = b.passenger_id

15

JOIN booking_flight bf ON b.booking_id = bf.booking_id

16

GROUP BY p.passenger_id, p.first_name, p.last_name

17

ORDER BY COUNT(*) DESC

18

LIMIT 1;

19

END;

20

\$\$;

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 35 msec.

Total rows:

Query complete 00:00:00.035

7)

Query

Query History

1

CREATE OR REPLACE FUNCTION get_heavily_delayed_flights()

2

RETURNS TABLE(

3

flight_id INT,

4

flight_number VARCHAR,

5

delay_hours NUMERIC

6

)

7

LANGUAGE plpgsql

8

AS \$\$

9

▼ BEGIN

10

RETURN QUERY

11

SELECT f.flight_id, f.flight_number,

12

EXTRACT(EPOCH FROM (f.actual_arrival_time - f.arrival_time))/3600 AS delay_hours

13

FROM flights f

14

WHERE f.actual_arrival_time IS NOT NULL

15

AND f.actual_arrival_time - f.arrival_time > INTERVAL '24 hours'

16

ORDER BY delay_hours DESC;

17

END;

18

\$\$;

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 36 msec.

8)

Query

Query History

1

CREATE OR REPLACE FUNCTION count_flights_by_airline(p_airline_id INT)

2

RETURNS BIGINT

3

LANGUAGE plpgsql

4

AS \$\$

5

DECLARE cnt BIGINT;

6

▼ BEGIN

7

SELECT COUNT(*) INTO cnt FROM flights WHERE airline_id = p_airline_id;

8

RETURN COALESCE(cnt, 0);

9

END;

10

\$\$;

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 34 msec.

9)

Query

Query History

```

1 CREATE OR REPLACE FUNCTION avg_ticket_price(p_flight_id INT)
2 RETURNS NUMERIC
3 LANGUAGE plpgsql
4 AS $$
5 DECLARE avg_price NUMERIC;
6 BEGIN
7     SELECT AVG(bp.price) INTO avg_price
8     FROM boarding_pass bp
9     JOIN booking b ON bp.booking_id = b.booking_id
10    JOIN booking_flight bf ON b.booking_id = bf.booking_id
11    WHERE bf.flight_id = p_flight_id;
12
13    RETURN COALESCE(avg_price, 0);
14 END;
15 $$;

```

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 37 msec.

Total rows:

Query complete 00:00:00.037

Query returned successfully

10)

Query

Query History

```

1 CREATE OR REPLACE FUNCTION get_most_expensive_flight()
2 RETURNS TABLE(
3     flight_number VARCHAR,
4     departure_airport TEXT,
5     arrival_airport TEXT,
6     max_price NUMERIC
7 )
8 LANGUAGE plpgsql
9 AS $$
10 BEGIN
11     RETURN QUERY
12     SELECT f.flight_number,
13            d.airport_name::TEXT,
14            a.airport_name::TEXT,
15            MAX(bp.price) AS max_price
16     FROM flights f
17     JOIN airport d ON f.departure_airport_id = d.airport_id
18     JOIN airport a ON f.arrival_airport_id = a.airport_id
19     JOIN booking_flight bf ON f.flight_id = bf.flight_id
20     JOIN booking b ON bf.booking_id = b.booking_id
21     JOIN boarding_pass bp ON b.booking_id = bp.booking_id
22     GROUP BY f.flight_id, f.flight_number, d.airport_name, a.airport_name
23     ORDER BY MAX(bp.price) DESC
24     LIMIT 1;
25 END;
26 $$;

```

Data Output

Messages

Notifications

CREATE FUNCTION

Query returned successfully in 47 msec.