

# Laboratory work 6

## JOIN operations tasks

1. Write a query that displays all flights of a specific airline.

The screenshot shows a PostgreSQL database interface with the following components:

- Object Explorer:** A tree view on the left showing database objects. The 'airport' table is selected under the 'Columns (7)' section.
- Query Editor:** A central pane showing the following SQL query:

```
1 SELECT f.flight_id, f.sch_departure_time, a.airline_name
2 FROM flights f
3 JOIN airline_info a ON f.airline_id = a.airline_id
4 WHERE airline_name LIKE 'Air%'
```
- Data Output:** A table at the bottom showing the results of the query. It has 4 rows and 3 columns: flight\_id, sch\_departure\_time, and airline\_name.
- Status Bar:** At the bottom, it indicates 'Total rows: 4' and 'Query complete 00:00:00.066'. A green message box says 'Successfully run. Total query runtime: 66 msec. 4 rows affected.'

flight_id	sch_departure_time	airline_name
4	2025-06-20 00:00:00	Air India
33	2024-11-22 00:00:00	Air Canada
52	2024-10-20 00:00:00	Air New Zealand
104	2024-11-24 00:00:00	Air France

2. Compose a query to obtain a list of all flights with the names of departure airports.

The screenshot shows a PostgreSQL database management tool interface. On the left, the 'Object Explorer' pane displays a tree view of database objects, including 'airline\_info', 'airport', 'baggage', 'boarding\_pass', 'booking', 'booking\_flight', 'flights', and 'flights' (with 12 columns). The 'flights' table is selected, showing columns: flight\_id, sch\_departure\_time, sch\_arrival\_time, departing\_airport\_id, arriving\_airport\_id, departing\_gate, arriving\_gate, airline\_id, act\_departure\_time, act\_arrival\_time, created\_at, and updated\_at.

The main query editor displays the following SQL query:

```
1 SELECT f.flight_id, f.sch_departure_time, a.airport_name
2 FROM flights f
3 JOIN airport a ON f.departing_airport_id = a.airport_id;
```

The 'Data Output' pane shows the results of the query, displaying 153 rows. The first 16 rows are visible in the screenshot:

flight_id	sch_departure_time	airport_name
1	2025-07-10 00:00:00	Tumeremo Airport
2	2025-06-20 00:00:00	Yorkton Municipal Airport
3	2025-08-30 00:00:00	Kenosha Regional Airport
4	2025-03-20 00:00:00	Senadora Eunice Micheles Airport
5	2025-03-20 00:00:00	Riverside Airport
6	2024-10-06 00:00:00	Lalmonirhat Airport
7	2025-03-09 00:00:00	Puerto Leda Airport
8	2025-06-04 00:00:00	Bembridge Airport
9	2025-05-27 00:00:00	Keokuk Municipal Airport
10	2024-12-06 00:00:00	Maxson Airfield
11	2024-10-31 00:00:00	Dongola Airport
12	2025-03-08 00:00:00	Zabreh Ostrava Airport
13	2025-03-28 00:00:00	Cap Manuel Niño International Airport
14	2024-10-30 00:00:00	Aasiaat Airport
15	2024-12-27 00:00:00	Myeik Airport
16	2025-09-08 00:00:00	Toccoa Airport - R.G. Letourneau Field

The status bar at the bottom indicates: 'Total rows: 153', 'Query complete 00:00:00.086', and 'Successfully run. Total query runtime: 86 msec. 153 rows affected.'

3. Create a query that finds all airlines that have no flights scheduled for the next month.

Object Explorer

- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- airline\_info
  - Columns (7)
    - airline\_id
    - airline\_code
    - airline\_name
    - airline\_country
    - created\_at
    - updated\_at
    - info
  - Constraints
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- airport
- baggage
- baggage\_check
- boarding\_pass
- booking
- booking\_flight
- flights
  - Columns (12)
    - flight\_id
    - sch\_departure\_time
    - sch\_arrival\_time
    - departing\_airport\_id
    - arriving\_airport\_id
    - departing\_gate
    - arriving\_gate
    - airline\_id
    - act\_departure\_time
    - act\_arrival\_time
    - created\_at
    - updated\_at
  - Constraints
  - Indexes
  - RLS Policies

db\_lab/postgres@PostgreSQL 17

Query

```

1 SELECT a.airline_name
2 FROM airline_info a
3 LEFT JOIN flights f ON a.airline_id = f.airline_id
4 AND EXTRACT(MONTH FROM f.sch_departure_time) = EXTRACT(MONTH FROM f.sch_departure_time) + 1
5 WHERE f.flight_id IS NULL;

```

Data Output

airline_name
1 South African Airways
2 Singapore Airlines
3 SAS Scandinavian
4 Air India
5 Thai Airways
6 British Airways
7 Alitalia
8 Avianca
9 Qantas
10 LATAM Airlines
11 Ethiopian Airlines
12 All Nippon Airways
13 Lufthansa
14 Korean Air
15 Qatar Airways
16 Southwest Airlines

Showing rows: 1 to 31 Page No: 1 of 1

Successfully run. Total query runtime: 89 msec. 31 rows affected.

Total rows: 31 Query complete 00:00:00.089

4. Create a query to display a list of passengers on a specific flight.

Object Explorer

- baggage\_check
- boarding\_pass
- booking
  - Columns (8)
    - booking\_id
    - flight\_id
    - passenger\_id
    - booking\_platform
    - created\_at
    - updated\_at
    - status
    - price
  - Constraints (1)
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- booking\_flight
- flights
  - Columns (12)
    - flight\_id
    - sch\_departure\_time
    - sch\_arrival\_time
    - departing\_airport\_id
    - arriving\_airport\_id
    - departing\_gate
    - arriving\_gate
    - airline\_id
    - act\_departure\_time
    - act\_arrival\_time
    - created\_at
    - updated\_at
  - Constraints
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- passengers
  - Columns (10)
    - passenger\_id
    - first\_name
    - last\_name

db\_lab/postgres@PostgreSQL 17

Query

```

1 SELECT f.flight_id, p.first_name, p.last_name
2 FROM passengers p
3 JOIN booking b ON p.passenger_id = b.passenger_id
4 JOIN flights f ON b.flight_id = f.flight_id
5 WHERE f.flight_id BETWEEN 1 AND 200;

```

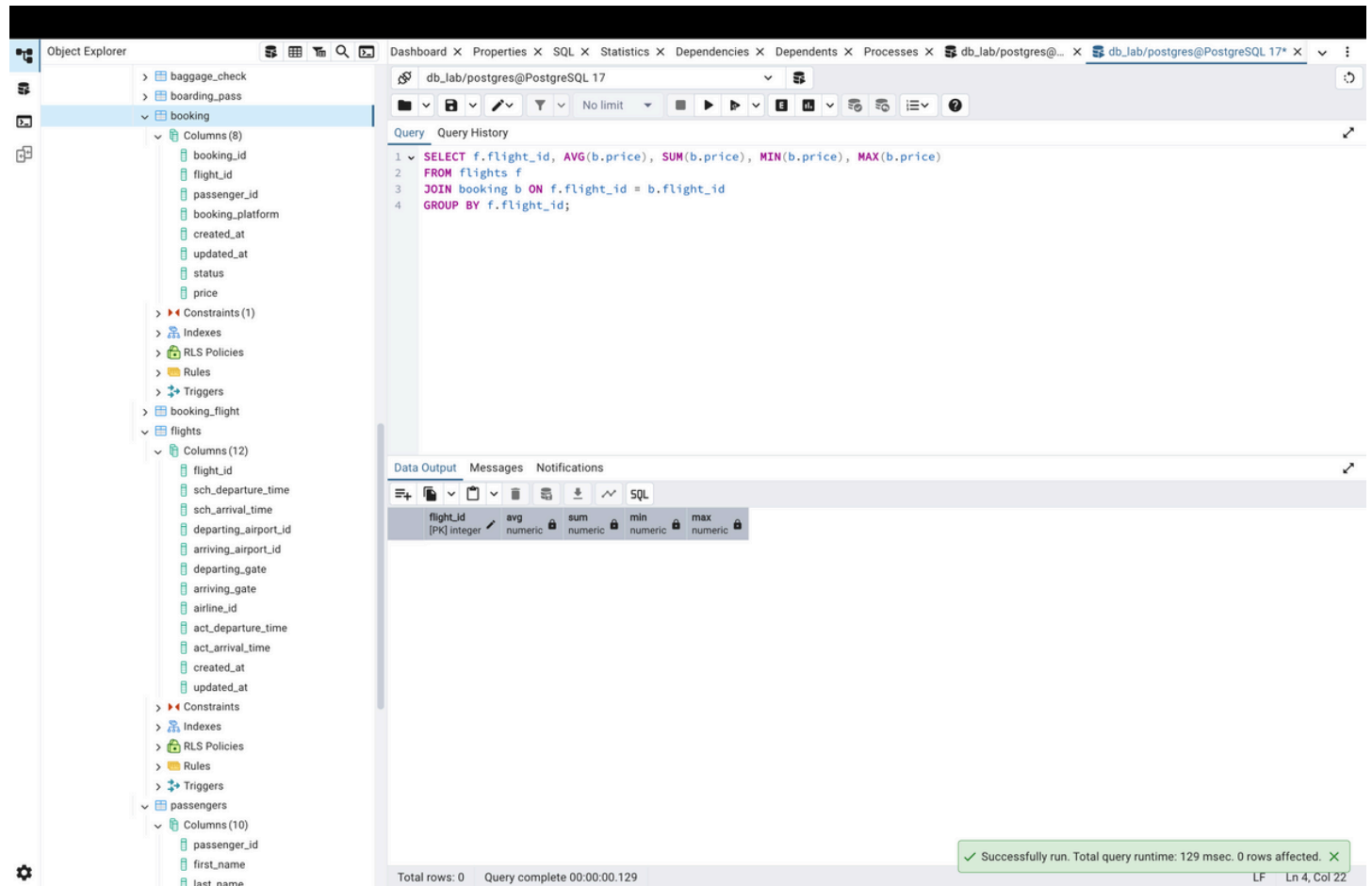
Data Output

flight_id	first_name	last_name
integer	character varying (50)	character varying (50)

Successfully run. Total query runtime: 116 msec. 0 rows affected.

Total rows: 0 Query complete 00:00:00.116

5. Write a query that calculates the average, total, maximum and minimum price of tickets for each flight.



The screenshot shows a PostgreSQL database interface with the following components:

- Object Explorer:** A tree view on the left showing the database schema. The 'bookings' table is selected, showing its columns (booking\_id, flight\_id, passenger\_id, booking\_platform, created\_at, updated\_at, status, price) and constraints.
- Query Editor:** The main area for writing SQL queries. The query is:

```
1 SELECT f.flight_id, AVG(b.price), SUM(b.price), MIN(b.price), MAX(b.price)
2 FROM flights f
3 JOIN bookings b ON f.flight_id = b.flight_id
4 GROUP BY f.flight_id;
```
- Data Output:** A table showing the results of the query. The columns are: flight\_id (PK integer), avg (numeric), sum (numeric), min (numeric), and max (numeric). The table is currently empty.
- Messages:** A status bar at the bottom indicating the query was successfully run, with a total query runtime of 129 msec and 0 rows affected.

6. Create a query that shows all flights flying to a specific country by combining flights, airports and airline, and using the condition on the country name.

Object Explorer

- status
- price
- Constraints (1)
- Indexes
- RLS Policies
- Rules
- Triggers
- booking\_flight
- flights
  - Columns (12)
    - flight\_id
    - sch\_departure\_time
    - sch\_arrival\_time
    - departing\_airport\_id
    - arriving\_airport\_id
    - departing\_gate
    - arriving\_gate
    - airline\_id
    - act\_departure\_time
    - act\_arrival\_time
    - created\_at
    - updated\_at
  - Constraints
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- passengers
  - Columns (10)
    - passenger\_id
    - first\_name
    - last\_name
    - date\_of\_birth
    - gender
    - country\_of\_citizenship
    - country\_of\_residence
    - passport\_number
    - created\_at
    - updated\_at
  - Constraints
  - Indexes
  - RLS Policies

db\_lab/postgres@PostgreSQL 17

Query

```

1 SELECT f.flight_id, a.airline_name, ap.country
2 FROM flights f
3 JOIN airport ap ON f.arriving_airport_id = ap.airport_id
4 JOIN airline_info a ON f.airline_id = a.airline_id
5 WHERE ap.country = 'Canada';

```

Data Output

flight_id	airline_name	country
integer	character varying (300)	character varying (150)

Successfully run. Total query runtime: 66 msec. 0 rows affected.

Total rows: 0 Query complete 00:00:00.066 LF Ln 5, Col 29

7. Display a list of minor passengers and their arrival destination.

Object Explorer

- booking
  - Columns (8)
    - booking\_id
    - flight\_id
    - passenger\_id
    - booking\_platform
    - created\_at
    - updated\_at
    - status
    - price
  - Constraints (1)
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- flights
  - Columns (12)
    - flight\_id
    - sch\_departure\_time
    - sch\_arrival\_time
    - departing\_airport\_id
    - arriving\_airport\_id
    - departing\_gate
    - arriving\_gate
    - airline\_id
    - act\_departure\_time
    - act\_arrival\_time
    - created\_at
    - updated\_at
  - Constraints
  - Indexes
  - RLS Policies
  - Rules
  - Triggers
- passengers
  - Columns (10)
    - passenger\_id
    - first\_name
    - last\_name
    - date\_of\_birth
    - gender
  - Constraints
  - Indexes
  - RLS Policies

db\_lab/postgres@PostgreSQL 17

Query

```

1 SELECT p.first_name, p.last_name, ap.city
2 FROM passengers p
3 JOIN booking b ON p.passenger_id = b.passenger_id
4 JOIN flights f ON b.flight_id = f.flight_id
5 JOIN airport ap ON f.arriving_airport_id = ap.airport_id
6 WHERE EXTRACT(YEAR FROM AGE(p.date_of_birth)) < 18;

```

Data Output

first_name	last_name	city
character varying (50)	character varying (50)	character varying (150)

Successfully run. Total query runtime: 64 msec. 0 rows affected.

Total rows: 0 Query complete 00:00:00.064 LF Ln 6, Col 52

8. Display the passenger's full name, passport number, and the passenger's current time of arrival at the destination.

The screenshot shows a PostgreSQL IDE interface. On the left, the 'Object Explorer' pane displays the database schema. The 'flights' table is expanded, showing columns: flight\_id, sch\_departure\_time, sch\_arrival\_time, departing\_airport\_id, arriving\_airport\_id, departing\_gate, arriving\_gate, airline\_id, act\_departure\_time, act\_arrival\_time, created\_at, and updated\_at. The 'act\_arrival\_time' column is highlighted. Below it, the 'passengers' table is also expanded, showing columns: passenger\_id, first\_name, last\_name, date\_of\_birth, gender, country\_of\_citizenship, country\_of\_residence, passport\_number, created\_at, and updated\_at. The 'Query' pane in the center contains the following SQL query:

```
1 SELECT p.first_name || ' ' || p.last_name AS full_name, p.passport_number, f.act_arrival_time
2 FROM passengers p
3 JOIN booking b ON p.passenger_id = b.passenger_id
4 JOIN flights f ON b.flight_id = f.flight_id;
```

The 'Data Output' pane at the bottom shows the results of the query. It displays three columns: full\_name (text), passport\_number (character varying (20)), and act\_arrival\_time (timestamp without time zone). The status bar at the bottom indicates 'Total rows: 0' and 'Query complete 00:00:00.194'. A green message box at the bottom right states: 'Successfully run. Total query runtime: 194 msec. 0 rows affected.'

9. Print a list of flights where the airline's home country and origin country are the same. Group them by the airport country.

The screenshot shows the DBeaver application window. On the left is the Object Explorer tree, expanded to show the 'airline\_info' table under the 'db\_lab/postgres@PostgreSQL 17\*' connection. The main panel displays a SQL query in the Query Editor:

```
SELECT f.flight_id, a.airline_name, ai.country
FROM flights f
JOIN airline_info a ON f.airline_id = a.airline_id
JOIN airport ai ON f.departing_airport_id = ai.airport_id
WHERE a.airline_country = ai.country
GROUP BY ai.country, f.flight_id, a.airline_name;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query as a table with three columns:

flight_id	airline_name	country
integer	character varying (300)	character varying (150)

At the bottom right, a green status bar indicates: "Successfully run. Total query runtime: 104 msec. 0 rows affected."