

Laboratory work 4

1. Retrieve all airline names in uppercase.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'airline' table. The central pane shows a SQL query: `SELECT UPPER (airline_name) AS airline_name_upper FROM airline;`. The 'Data Output' tab is active, displaying a table with 10 rows of results. The status bar at the bottom indicates 'Total rows: 50' and 'Query complete 00:00:00.075'.

airline_name_upper
IPC
PDN
KLE
KHS
YLQ
NGL
0
QIG
NQX
SOZ

2. Replace any occurrence of the word "Air" in airline names with "Aero".

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including the 'airline' table. The central pane shows a SQL query: `SELECT REPLACE(airline_name, 'Air', 'Aero') AS replaced_airline_name FROM airline;`. The 'Data Output' tab is active, displaying a table with 11 rows of results. The status bar at the bottom indicates 'Total rows: 50' and 'Query complete 00:00:00.065'.

replaced_airline_name
IPC
PDN
KLE
KHS
YLQ
NGL
0
QIG
NQX
SOZ
IVA

3. Find all flight numbers that coordinates with both airline 1 and airline 2.

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane is expanded to show the 'flights' table under the 'public' schema. The 'Columns (14)' list includes: flight_id, 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, and update_at. The 'Query' pane contains the following SQL query:

```
1 SELECT flight_id FROM flights WHERE airline_id IN(1,2) GROUP BY flight_id
2 HAVING COUNT(DISTINCT airline_id) = 2;
```

The 'Data Output' pane is empty, showing 'Total rows: 0'. The status bar at the bottom indicates 'Query complete 00:00:00.250'.

4. Retrieve airports that contain the word "Reginal" and "Air" in their names.

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane is expanded to show the 'airport' table under the 'public' schema. The 'Columns (7)' list includes: airport_id, airport_name, country, state, city, created_at, and update_at. The 'Query' pane contains the following SQL query:

```
1 SELECT * FROM airport WHERE airport_name LIKE '%Reginal%' AND airport_name LIKE '%Air%';
```

The 'Data Output' pane shows the results of the query, with columns: airport_id [PK] integer, airport_name character varying (50), country character varying (50), state character varying (50), city character varying (50), created_at date, and update_at date. The status bar at the bottom indicates 'Query complete 00:00:00.117'.

5. Retrieve passenger names and format their birth dates as 'Month DD, YYYY'..o

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, with the 'passengers' table selected under the 'public' schema. The main query editor contains the following SQL query:

```
1 SELECT first_name || ' ' || last_name AS full_name,  
2    TO_CHAR(date_of_birth, 'Month DD, YYYY') AS birth_date  
3 FROM passengers;
```

The 'Data Output' tab shows the results of the query, displaying 11 rows of passenger data. The columns are 'full_name' and 'birth_date'.

	full_name	birth_date
1	Hilde Iris	January 03, 2000
2	Arvy Sparsholt	June 09, 1974
3	Reinald Pococke	June 07, 1982
4	Con Borrel	October 17, 1986
5	Wayne Bangs	April 22, 1996
6	Tildy Shackelford	April 15, 2004
7	Byrle Oram	July 07, 1985
8	Howard Igo	March 29, 1984
9	Alphonso Philippou	August 30, 1975
10	Fairfax Needham	July 08, 1981
11	Vanda Prime	May 17, 1977

The status bar at the bottom indicates 'Total rows: 200' and 'Query complete 00:00:00.080'.

6. Find flight numbers that have been delayed based on the actual arrival time.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, with the 'flights' table selected under the 'public' schema. The main query editor contains the following SQL query:

```
1 SELECT flight_id FROM flights  
2 WHERE actual_arrival > scheduled_arrival OR status = 'Delayed';
```

The 'Data Output' tab shows the results of the query, displaying 16 rows of flight data. The column is 'flight_id'.

	flight_id
1	1
2	2
3	4
4	7
5	8
6	9
7	11
8	12
9	13
10	14
11	16

The status bar at the bottom indicates 'Total rows: 750' and 'Query complete 00:00:00.108'.

7. Create a query that divides passengers into age groups like 'Young' and 'Adult' based on their birth date. Young passengers age between 18 and 35, Adult passengers age between 36 and 55.

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' shows the database structure, with 'passengers' selected under 'flights'. The 'Query' tab is active, displaying the following SQL query:

```

1 SELECT
2     first_name || ' ' || last_name AS full_name,
3     date_of_birth,
4     CASE
5         WHEN EXTRACT(YEAR FROM AGE(date_of_birth)) BETWEEN 18 AND 35 THEN 'Young'
6         WHEN EXTRACT(YEAR FROM AGE(date_of_birth)) BETWEEN 36 AND 55 THEN 'Adult'
7         ELSE 'Other'
8     END AS age_group
9 FROM passengers;

```

The 'Data Output' tab shows the results of the query:

	full_name	date_of_birth	age_group
1	Hilde Irmis	2000-01-03	Young
2	Arvy Sparsholt	1974-06-09	Adult
3	Reinald Pococke	1982-06-07	Adult
4	Con Borrel	1986-10-17	Adult
5	Wayne Bangs	1996-04-22	Young
6	Tildy Shackelford	2004-04-15	Young

Total rows: 200 Query complete 00:00:00.067

8. Create a query that categorizes ticket prices based on their price as "Cheap," "Medium" or "Expensive."

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' shows the database structure, with 'booking' selected under 'flights'. The 'Query' tab is active, displaying the following SQL query:

```

1 SELECT booking_id,
2     CASE
3         WHEN price < 2000 THEN 'Cheap'
4         WHEN price BETWEEN 2000 AND 4000 THEN 'Medium'
5         ELSE 'Expensive'
6     END AS category_ticket
7 FROM booking;

```

The 'Data Output' tab shows the results of the query:

	booking_id [PK] integer	category_ticket
7	7	Cheap
8	8	Expensive
9	9	Expensive
10	10	Cheap
11	11	Medium
12	12	Medium
13	13	Expensive
14	14	Cheap

Total rows: 500 Query complete 00:00:00.092

9. Find number of airline names in each airline country.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, with the 'airline' table selected under 'Tables (10)'. The 'Columns (6)' for the 'airline' table are listed: airline_id, airline_code, airline_name, airline_country, created_at, and update_at. The main query editor contains the following SQL query:

```
1 SELECT airline_country, COUNT(*) AS airlines_number
2 FROM airline GROUP BY airline_country;
3
```

The 'Data Output' tab shows the results of the query, displaying 11 rows. The columns are 'airline_country' (character varying (50)) and 'airlines_number' (bigint). The data is as follows:

airline_country	airlines_number
Indonesia	2
Venezuela	1
Hungary	1
China	10
Russia	4
Sweden	1
Brazil	3
Ireland	1
Portugal	3
Panama	1
Vaman	1

The status bar at the bottom indicates 'Total rows: 26' and 'Query complete 00:00:00.141'.

10. Find flights that arrived late according to their actual arrival time compared to the scheduled arrival time.

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, with the 'flights' table selected under 'Tables (14)'. The 'Columns (14)' for the 'flights' table are listed: flight_id, 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, and update_at. The main query editor contains the following SQL query:

```
1 SELECT flight_id, scheduled_arrival, actual_arrival
2 FROM flights
3 WHERE actual_arrival > scheduled_arrival;
4
```

The 'Data Output' tab shows the results of the query, displaying 11 rows. The columns are 'flight_id' (integer), 'scheduled_arrival' (date), and 'actual_arrival' (date). The data is as follows:

flight_id	scheduled_arrival	actual_arrival
1	2023-09-08	2023-11-07
2	2023-09-17	2024-01-23
3	2023-03-18	2023-04-07
4	2023-04-08	2023-08-01
5	2023-09-19	2023-12-03
6	2023-06-04	2023-11-17
7	2023-06-02	2023-11-09
8	2023-03-30	2023-07-21
9	2024-01-22	2024-03-06
10	2023-10-11	2024-01-04
11	2023-05-24	2023-12-22

The status bar at the bottom indicates 'Total rows: 509' and 'Query complete 00:00:00.118'.