

LAB3 Adilet

1) Select all the data of passengers whose last name is same as first name.

```
SELECT * FROM passengers WHERE last_name = first_name;
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane displays the database structure, with the 'passengers' table selected. The 'Columns' list for 'passengers' is visible, including 'passenger_id', 'first_name', 'last_name', 'date_of_birth', 'gender', 'country_of_citizenship', 'country_of_residence', 'passport_number', 'created_at', and 'updated_at'. The 'first_name' column is highlighted. In the center, the 'Query' pane contains the SQL query: `SELECT * FROM passengers WHERE last_name = first_name;`. Below the query, the 'Data Output' pane shows the results of the query. The results are displayed in a table with 9 columns: 'passenger_id', 'first_name', 'last_name', 'date_of_birth', 'gender', 'country_of_citizenship', 'country_of_residence', 'passport_number', and 'passport_name'. The table contains 3 rows of data. A status bar at the bottom indicates 'Total rows: 3' and 'Query complete 00:00:00.188'. A green message box at the bottom right states 'Successfully run. Total query runtime: 188 msec. 3 rows affected.'

passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship	country_of_residence	passport_number	passport_name
1	Иван	Иван	1990-05-15	M	Russia	Russia	123456789	
2	Владимир	Владимир	1976-05-07	M	Russia	UK	556677889	
3	Юлия	Юлия	1994-02-28	F	Ukraine	Ukraine	667788990	

2) Select the last name of all passangers without duplicates.

```
SELECT DISTINCT last_name FROM passengers;
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane displays the database structure, with the 'passengers' table selected. The 'Columns' list for 'passengers' is visible, including 'passenger_id', 'first_name', 'last_name', 'date_of_birth', 'gender', 'country_of_citizenship', 'country_of_residence', 'passport_number', 'created_at', and 'updated_at'. The 'last_name' column is highlighted. In the center, the 'Query' pane contains the SQL query: `SELECT DISTINCT last_name FROM passengers;`. Below the query, the 'Data Output' pane shows the results of the query. The results are displayed in a table with 1 column: 'last_name'. The table contains 9 rows of data. A status bar at the bottom indicates 'Total rows: 9' and 'Query complete 00:00:00.169'. A green message box at the bottom right states 'Successfully run. Total query runtime: 166 msec. 20 rows affected.'

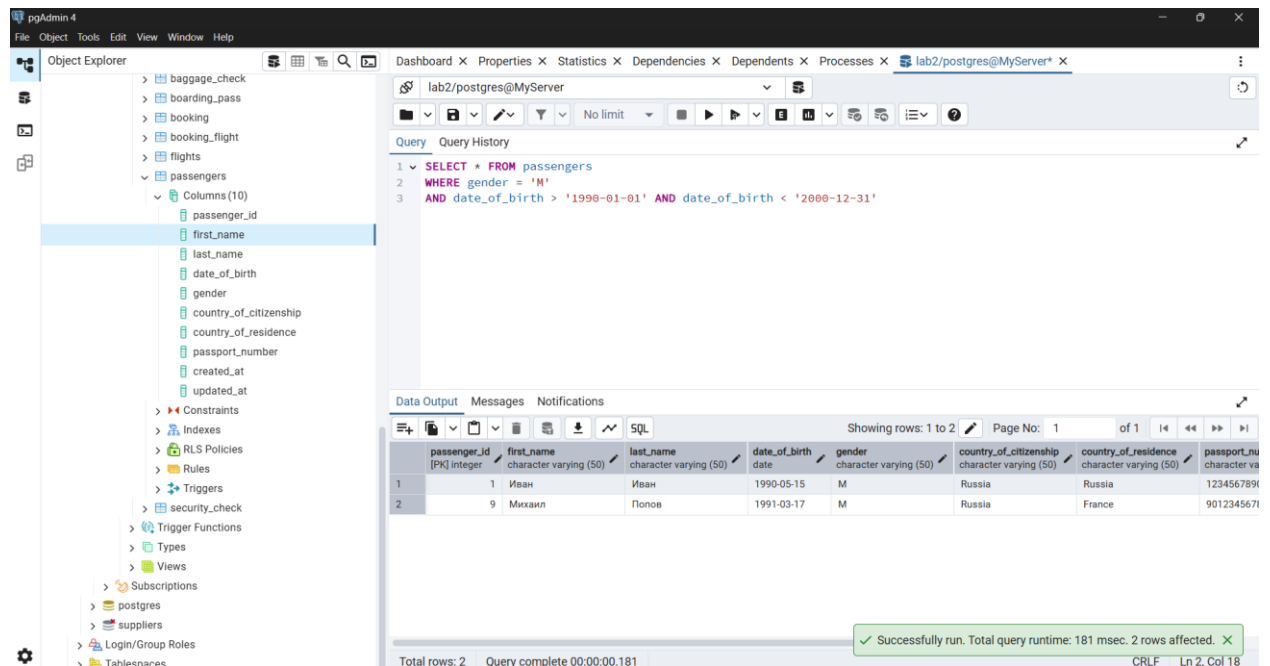
last_name
Орлов
Павлова
Андреева
Волкова
Новикова
Попов
Козлова
Иван
Петрова

3) Find all male passengers born between 1990 and 2000.

```
SELECT * FROM passengers
```

```
WHERE gender = 'M'
```

```
AND date_of_birth > '1990-01-01' AND date_of_birth < '2000-12-31'
```



4) Find price of tickets sold for each month in sorted way.

```
SELECT
```

```
EXTRACT(YEAR FROM purchase_date) AS year,
```

```
EXTRACT(MONTH FROM purchase_date) AS month,
```

```
SUM(price) AS total_price
```

```
FROM tickets
```

```
GROUP BY year, month
```

```
ORDER BY year, month;
```

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the database structure, with 'Tables (10)' expanded. The main pane displays a SQL query:

```
1 SELECT
2     EXTRACT(YEAR FROM purchase_date) AS year,
3     EXTRACT(MONTH FROM purchase_date) AS month,
4     SUM(price) AS total_price
5 FROM tickets
6 GROUP BY year, month
7 ORDER BY year, month;
```

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

	year numeric	month numeric	total_price numeric
1	2024	1	58000.00
2	2024	2	64000.00
3	2024	3	95000.00
4	2024	4	57000.00
5	2024	5	71000.00
6	2024	6	90500.00
7	2024	7	42500.00
8	2024	8	73000.00
9	2024	9	64000.00
Total rows: 10			

A status message at the bottom right indicates: 'Successfully run. Total query runtime: 161 msec. 10 rows affected.'

5) Create a query that shows all flights flying to 'China'.

SELECT *

FROM flights

WHERE country = 'China';

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the database structure, with 'Columns (12)' expanded. The main pane displays a SQL query:

```
1 SELECT *
2 FROM flights
3 WHERE country = 'China';
```

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

	flight_id [PK] integer	sch_departure_time timestamp without time zone	sch_arrival_time timestamp without time zone	departing_airport_id integer	arriving_airport_id integer	departing_gate text	arriving_gate character varying (50)
1	1	2024-01-15 08:00:00	2024-01-15 12:30:00	1	5	A12	B05
2	3	2024-02-05 07:15:00	2024-02-05 12:20:00	3	7	C02	D01
3	5	2024-03-01 10:10:00	2024-03-01 15:20:00	5	9	A07	B02
4	7	2024-04-02 14:20:00	2024-04-02 19:10:00	7	1	C09	D02
5	9	2024-05-05 12:00:00	2024-05-05 17:30:00	9	3	A11	B06
6	11	2024-06-01 15:00:00	2024-06-01 20:10:00	1	5	C13	D03
7	13	2024-07-03 10:00:00	2024-07-03 15:10:00	3	7	A15	B07
8	15	2024-08-02 11:30:00	2024-08-02 16:40:00				
Total rows: 10							

A status message at the bottom right indicates: 'Successfully run. Total query runtime: 104 msec. 10 rows affected.'

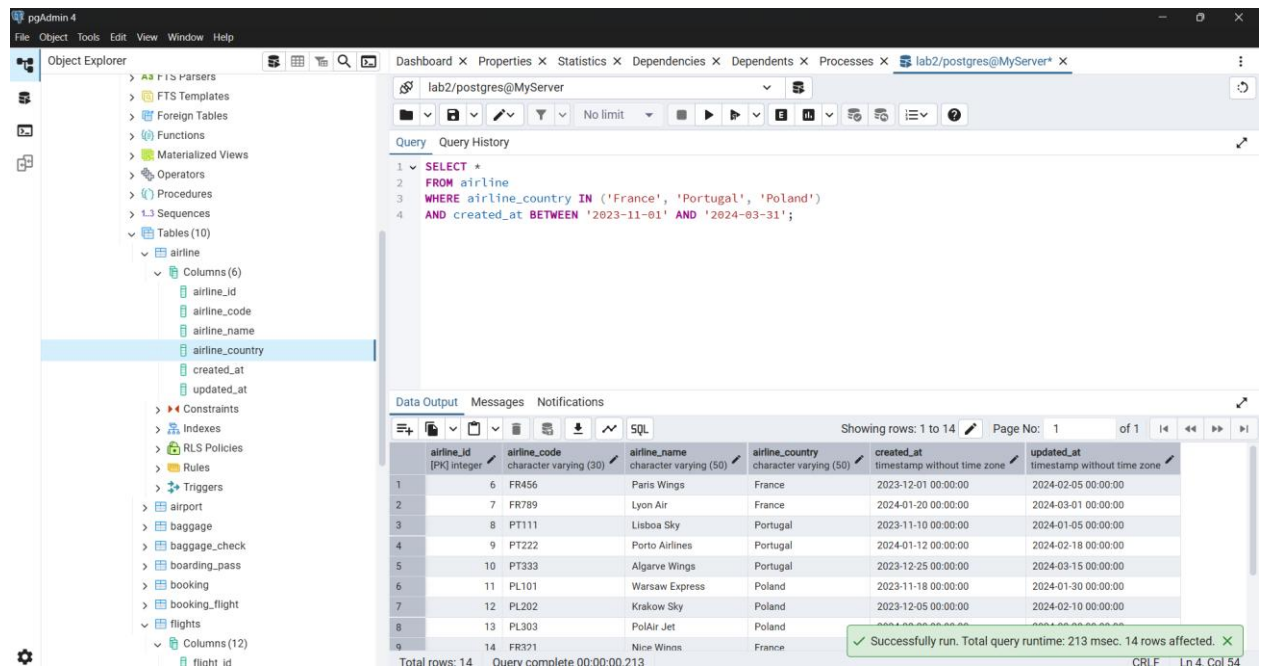
6) Show airlines from any of: ('France','Portugal','Poland') created between '2023-11-01' and '2024-03-31'.

SELECT *

FROM airline

WHERE airline_country IN ('France', 'Portugal', 'Poland')

AND created_at BETWEEN '2023-11-01' AND '2024-03-31';

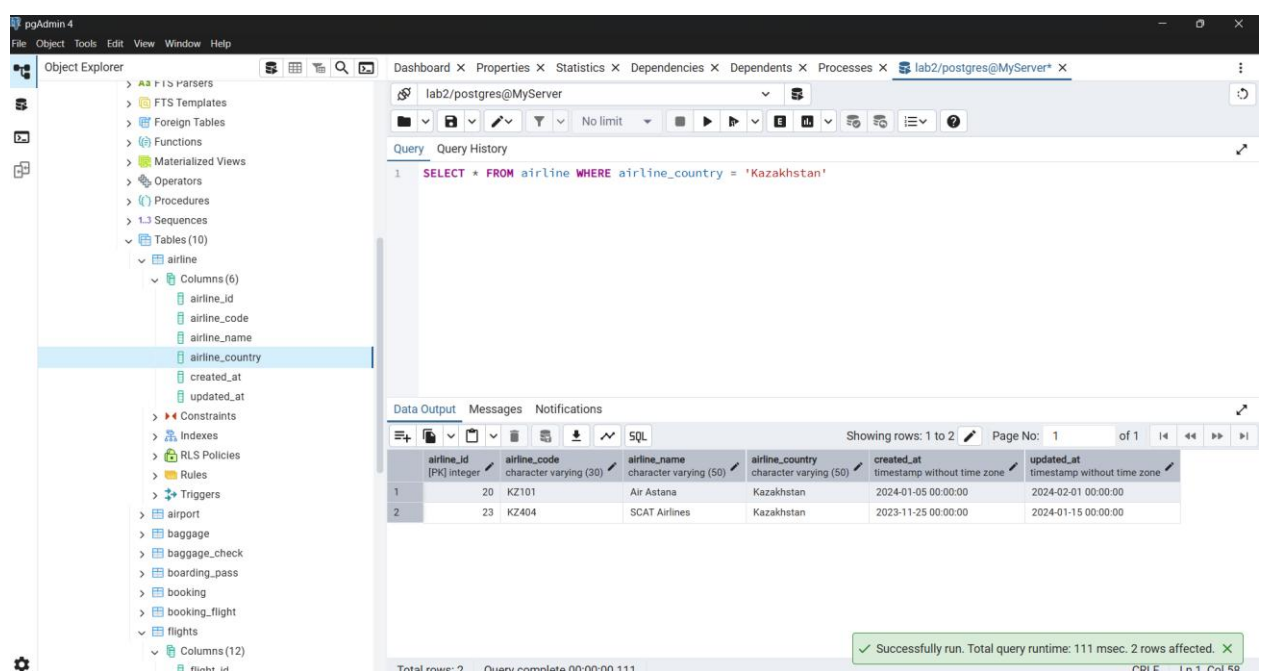


The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the 'airline' table selected. The main pane displays a SQL query and its results. The query is: `SELECT * FROM airline WHERE airline_country IN ('France', 'Portugal', 'Poland') AND created_at BETWEEN '2023-11-01' AND '2024-03-31';`. The results table shows 14 rows of airline data.

airline_id [PK] integer	airline_code character varying (30)	airline_name character varying (50)	airline_country character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone
1	6 FR456	Paris Wings	France	2023-12-01 00:00:00	2024-02-05 00:00:00
2	7 FR789	Lyon Air	France	2024-01-20 00:00:00	2024-03-01 00:00:00
3	8 PT111	Lisboa Sky	Portugal	2023-11-10 00:00:00	2024-01-05 00:00:00
4	9 PT222	Porto Airlines	Portugal	2024-01-12 00:00:00	2024-02-18 00:00:00
5	10 PT333	Algarve Wings	Portugal	2023-12-25 00:00:00	2024-03-15 00:00:00
6	11 PL101	Warsaw Express	Poland	2023-11-18 00:00:00	2024-01-30 00:00:00
7	12 PL202	Krakow Sky	Poland	2023-12-05 00:00:00	2024-02-10 00:00:00
8	13 PL303	PolAir Jet	Poland	2023-12-05 00:00:00	2024-02-10 00:00:00
9	14 FR321	Nice Wings	France	2023-12-05 00:00:00	2024-02-10 00:00:00
10	15 PT456	Lisboa Express	Portugal	2024-01-15 00:00:00	2024-03-05 00:00:00
11	16 PT567	Faro Airways	Portugal	2024-01-25 00:00:00	2024-03-10 00:00:00
12	17 PL456	Warsaw Sky	Poland	2024-01-05 00:00:00	2024-02-20 00:00:00
13	18 PL567	Krakow Express	Poland	2024-01-15 00:00:00	2024-03-05 00:00:00
14	19 PL678	PolAir Express	Poland	2024-01-25 00:00:00	2024-03-10 00:00:00

Successfully run. Total query runtime: 213 msec. 14 rows affected.

7) Find all airline names based in Kazakhstan.



The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the 'airline' table selected. The main pane displays a SQL query and its results. The query is: `SELECT * FROM airline WHERE airline_country = 'Kazakhstan';`. The results table shows 2 rows of airline data.

airline_id [PK] integer	airline_code character varying (30)	airline_name character varying (50)	airline_country character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone
1	20 KZ101	Air Astana	Kazakhstan	2024-01-05 00:00:00	2024-02-01 00:00:00
2	23 KZ404	SCAT Airlines	Kazakhstan	2023-11-25 00:00:00	2024-01-15 00:00:00

Successfully run. Total query runtime: 111 msec. 2 rows affected.

8) Reduce the cost of booking price by 10% created before '11-01-2023'.

UPDATE booking

SET ticket_price = ticket_price * 0.9

WHERE created_at < '2023-11-01';

The screenshot shows the pgAdmin 4 interface. In the Object Explorer on the left, the 'passengers' table is selected. The main query editor displays the following SQL query:

```
1 UPDATE booking
2 SET ticket_price = ticket_price * 0.9
3 WHERE created_at < '2023-11-01';
4
```

The 'Data Output' tab at the bottom shows the message: 'UPDATE 10' and 'Query returned successfully in 73 msec.' A green status bar at the bottom right confirms: 'Query returned successfully in 73 msec.'

Do:

The screenshot shows the pgAdmin 4 interface with a different query executed. The Object Explorer on the left shows the 'airline' table selected. The main query editor displays the following SQL query:

```
1 SELECT booking_id, passenger_id, flight_id, ticket_price, created_at
2 FROM booking
3 WHERE created_at < '2023-11-01';
4
```

The 'Data Output' tab at the bottom shows the results of the query in a table format. The table has 5 columns: booking_id [PK] integer, passenger_id integer, flight_id integer, ticket_price numeric (7,2), and created_at timestamp without time zone. The results show 10 rows of data.

	booking_id [PK] integer	passenger_id integer	flight_id integer	ticket_price numeric (7,2)	created_at timestamp without time zone
1	1	101	5	450.00	2022-10-15 00:00:00
2	2	102	6	585.00	2023-12-01 00:00:00
3	3	103	7	630.00	2023-01-10 00:00:00
4	4	104	8	405.00	2022-09-25 00:00:00
5	5	105	9	720.00	2023-03-15 00:00:00
6	6	106	10	270.00	2022-07-05 00:00:00
7	7	107	11	495.00	2022-11-20 00:00:00
8	8	108	12	810.00	2023-02-05 00:00:00
9	9	109	13	360.00	2022-08-18 00:00:00
10	10	110	14	540.00	2023-04-01 00:00:00

Total rows: 10 Query complete 00:00:00.108 CRLF Ln 1, Col 1

После:

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays the database structure, with the 'passengers' table selected. The main pane shows a SQL query:

```
1 SELECT booking_id, passenger_id, flight_id, ticket_price, created_at
2 FROM booking
3 WHERE created_at < '2023-11-01';
4
```

The Data Output tab shows the results of the query, displaying 10 rows. The columns are: booking_id (PK) integer, passenger_id integer, flight_id integer, ticket_price numeric (7,2), and created_at timestamp without time zone.

booking_id	passenger_id	flight_id	ticket_price	created_at
1	101	5	405.00	2022-10-15 00:00:00
2	102	6	526.50	2022-12-01 00:00:00
3	103	7	567.00	2023-01-10 00:00:00
4	104	8	364.50	2022-09-25 00:00:00
5	105	9	648.00	2023-03-15 00:00:00
6	106	10	243.00	2022-07-05 00:00:00
7	107	11	445.50	2022-11-20 00:00:00
8	108	12	729.00	2023-02-05 00:00:00
9	109	13	324.00	2022-08-18 00:00:00
10	110	14	513.00	2023-04-01 00:00:00

A green message box at the bottom right indicates: "Successfully run. Total query runtime: 177 msec. 10 rows affected."

9) Find top3 overweighted baggage with more than 25kg.

SELECT baggage_id, weight_in_kg, booking_id

FROM baggage

WHERE weight_in_kg > 25

ORDER BY weight_in_kg DESC

LIMIT 3;

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane is expanded to show the 'baggage' table under the 'airline' schema. The 'Columns' sub-pane shows 'baggage_id', 'weight_in_kg', 'created_at', 'updated_at', and 'booking_id'. The main query editor on the right contains the following SQL query:

```
1 SELECT baggage_id, weight_in_kg, booking_id
2 FROM baggage
3 WHERE weight_in_kg > 25
4 ORDER BY weight_in_kg DESC
5 LIMIT 3;
```

The 'Data Output' pane at the bottom displays the results of the query in a table with 3 rows and 3 columns:

baggage_id	weight_in_kg	booking_id
1	8	8
2	4	4
3	6	6

A status message at the bottom right indicates: 'Successfully run. Total query runtime: 108 msec. 3 rows affected.'

10) Find the youngest passengers' full name.

SELECT first_name,last_name,date_of_birth

FROM passengers

ORDER BY date_of_birth DESC

LIMIT 1;

The screenshot shows the pgAdmin 4 interface. On the left, the 'Object Explorer' pane is expanded to show the 'passengers' table under the 'airline' schema. The 'Columns' sub-pane shows 'passenger_id', 'first_name', 'last_name', 'date_of_birth', 'gender', 'country_of_citizenship', 'country_of_residence', 'passport_number', 'created_at', and 'updated_at'. The main query editor on the right contains the following SQL query:

```
1 SELECT first_name,last_name,date_of_birth
2 FROM passengers
3 ORDER BY date_of_birth DESC
4 LIMIT 1;
```

The 'Data Output' pane at the bottom displays the results of the query in a table with 1 row and 3 columns:

first_name	last_name	date_of_birth
Dana	Kaliyeva	2002-02-02

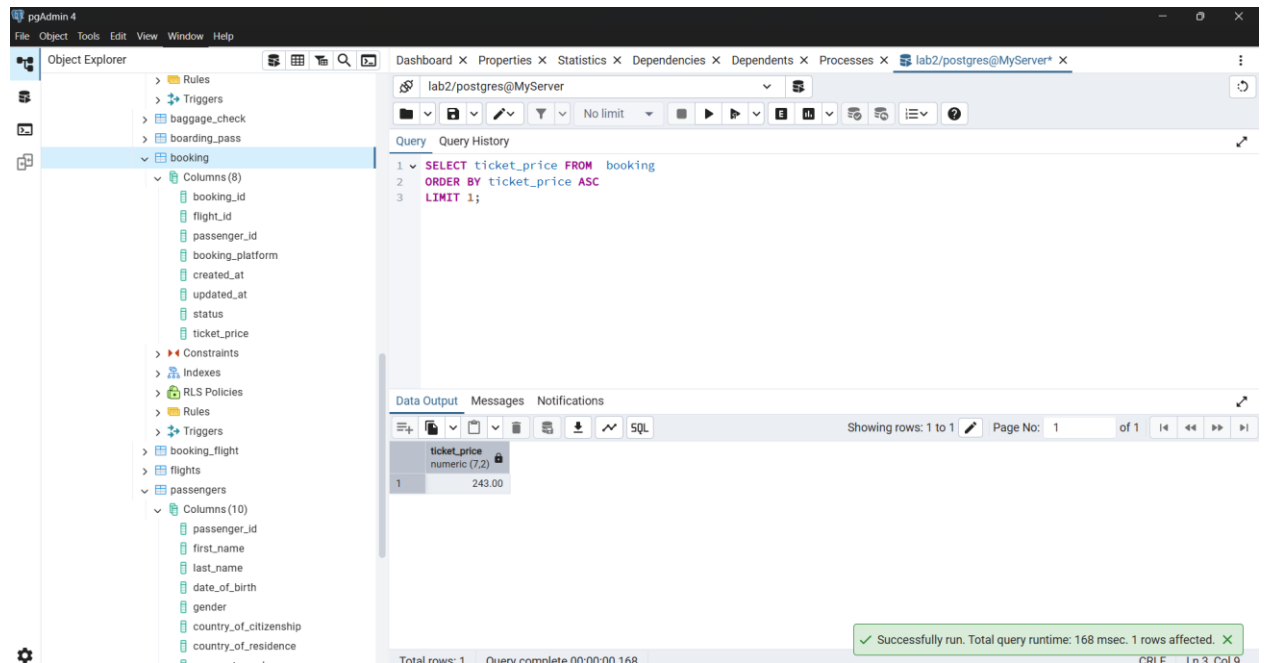
A status message at the bottom right indicates: 'Successfully run. Total query runtime: 140 msec. 1 rows affected.'

11) Find the cheapest booking price on each booking platform.

```
SELECT ticket_price FROM booking
```

```
ORDER BY ticket_price ASC
```

```
LIMIT 1;
```



The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the database structure, including the 'booking' table. The main pane displays a SQL query:

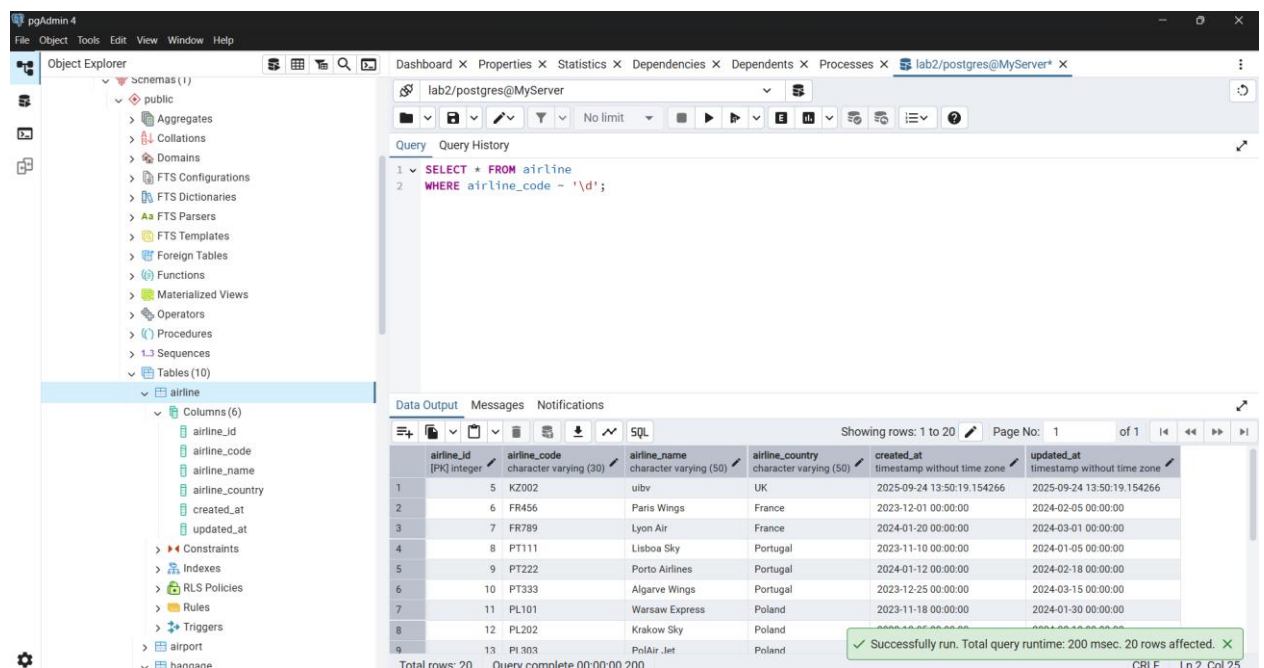
```
SELECT ticket_price FROM booking ORDER BY ticket_price ASC LIMIT 1;
```

 The Data Output pane shows the result: a single row with the value 243.00. A status message at the bottom indicates: 'Successfully run. Total query runtime: 168 msec. 1 rows affected.'

12) Return airlines whose airline_code contains a digit.

```
SELECT * FROM airline
```

```
WHERE airline_code ~ '\d';
```



The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer shows the database structure, including the 'airline' table. The main pane displays a SQL query:

```
SELECT * FROM airline WHERE airline_code ~ '\d';
```

 The Data Output pane shows the result, listing 20 rows of airline data. A status message at the bottom indicates: 'Successfully run. Total query runtime: 200 msec. 20 rows affected.'

airline_id [PK] integer	airline_code character varying (30)	airline_name character varying (50)	airline_country character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone
1	5 K2002	uibv	UK	2025-09-24 13:50:19.154266	2025-09-24 13:50:19.154266
2	6 FR456	Paris Wings	France	2023-12-01 00:00:00	2024-02-05 00:00:00
3	7 FR789	Lyon Air	France	2024-01-20 00:00:00	2024-03-01 00:00:00
4	8 PT111	Lisboa Sky	Portugal	2023-11-10 00:00:00	2024-01-05 00:00:00
5	9 PT222	Porto Airlines	Portugal	2024-01-12 00:00:00	2024-02-18 00:00:00
6	10 PT333	Algarve Wings	Portugal	2023-12-25 00:00:00	2024-03-15 00:00:00
7	11 PL101	Warsaw Express	Poland	2023-11-18 00:00:00	2024-01-30 00:00:00
8	12 PL202	Krakow Sky	Poland		
9	13 PL303	PolAir Jet	Poland		

13) List the top5 most recently created airlines.

SELECT * FROM airline

ORDER BY created_at DESC

LIMIT 5;

The screenshot shows the pgAdmin 4 interface. The left pane displays the 'airline' table under 'Tables (10)'. The right pane shows the query results for the query: `SELECT * FROM airline ORDER BY created_at DESC LIMIT 5;`. The results table has 8 columns: `airline_id`, `airline_code`, `airline_name`, `airline_country`, `created_at`, and `updated_at`. The data is as follows:

airline_id	airline_code	airline_name	airline_country	created_at	updated_at
1	5	KZ002	uibv	UK	2025-09-24 13:50:19.154266
2	3	FH	FlyHigh	Brazil	2025-09-23 23:28:07.225288
3	4	FF	FlyFly	Poland	2025-09-23 23:28:07.225288
4	2	AE	AirEasy	France	2025-09-23 23:28:07.225288
5	1	KZ	KazAir	Turkey	2025-09-23 23:25:16.022895

14) Return all rows where booking_id is between 200 and 300 inclusive and check_result <> 'Checked'.

SELECT * FROM booking

WHERE booking_id BETWEEN 200 AND 300

AND status <> 'Checked';

The screenshot shows the pgAdmin 4 interface. The left pane displays the 'booking' table under 'Tables (10)'. The right pane shows the query results for the query: `SELECT * FROM booking WHERE booking_id BETWEEN 200 AND 300 AND status <> 'Checked';`. The results table has 9 columns: `booking_id`, `flight_id`, `passenger_id`, `booking_platform`, `created_at`, `updated_at`, `status`, and `ticket_price`. The data is as follows:

booking_id	flight_id	passenger_id	booking_platform	created_at	updated_at	status	ticket_price
1	1	5	101	Airbnb	2022-10-15 00:00:00	Confirmed	
2	2	6	102	Booking.com	2022-12-01 00:00:00	Checked	
3	3	7	103	Expedia	2023-01-10 00:00:00	Pending	
4	4	8	104	Skyscanner	2022-09-25 00:00:00	Cancelled	
5	5	9	105	Airbnb	2023-03-15 00:00:00	Confirmed	
6	6	10	106	Booking.com	2022-07-05 00:00:00	Confirmed	
7	7	11	107	Expedia	2022-11-20 00:00:00	Pending	
8	8	12	108	Skyscanner	2023-02-05 00:00:00	Confirmed	

15) Baggage checks where update_at is in the same month as created_at but occurs earlier than created_at.

```
SELECT * FROM baggage_check
```

```
WHERE DATE_TRUNC('month', updated_at) = DATE_TRUNC('month', created_at)
```

```
AND updated_at < created_at;
```

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays the database structure, including the 'baggage_check' table. The central pane shows the SQL query:

```
1 SELECT * FROM baggage_check
2 WHERE DATE_TRUNC('month', updated_at) = DATE_TRUNC('month', created_at)
3 AND updated_at < created_at;
4
```

The bottom pane displays the query results in a table format. The table has 6 columns: baggage_check_id, check_result, created_at, updated_at, booking_id, and passenger_id. The results show 10 rows where the updated_at timestamp is in the same month as the created_at timestamp but occurs earlier.

baggage_check_id	check_result	created_at	updated_at	booking_id	passenger_id
1	Checked	2023-09-15 10:00:00	2023-09-14 09:30:00	1	1
2	Not Checked	2023-09-20 15:00:00	2023-09-18 12:00:00	2	2
3	Pending	2023-10-05 11:00:00	2023-10-04 10:45:00	3	3
4	Checked	2023-10-10 09:00:00	2023-10-09 08:30:00	4	4
5	Not Checked	2023-11-01 14:00:00	2023-11-01 09:15:00	5	5
6	Checked	2023-11-12 16:00:00	2023-11-10 15:00:00	6	6
7	Pending	2023-12-02 08:00:00	2023-12-01 07:45:00	7	7
8	Checked	2023-12-18 18:00:00	2023-12-17 17:30:00	8	8
9	Not Checked	2024-01-10 20:00:00	2024-01-09 19:30:00	9	9
10	Checked	2024-01-15 10:00:00	2024-01-14 09:30:00	10	10

A green message box at the bottom right indicates: "Successfully run. Total query runtime: 136 msec. 10 rows affected."

ended