

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

SELECT *

FROM Passengers

WHERE LOWER(first_name) = LOWER(last_name);

The screenshot shows a database IDE interface. The top panel displays a SQL query in a console window:

```
-- 1
2 SELECT *
3 FROM Passengers
4 WHERE LOWER(first_name) = LOWER(last_name);
5
```

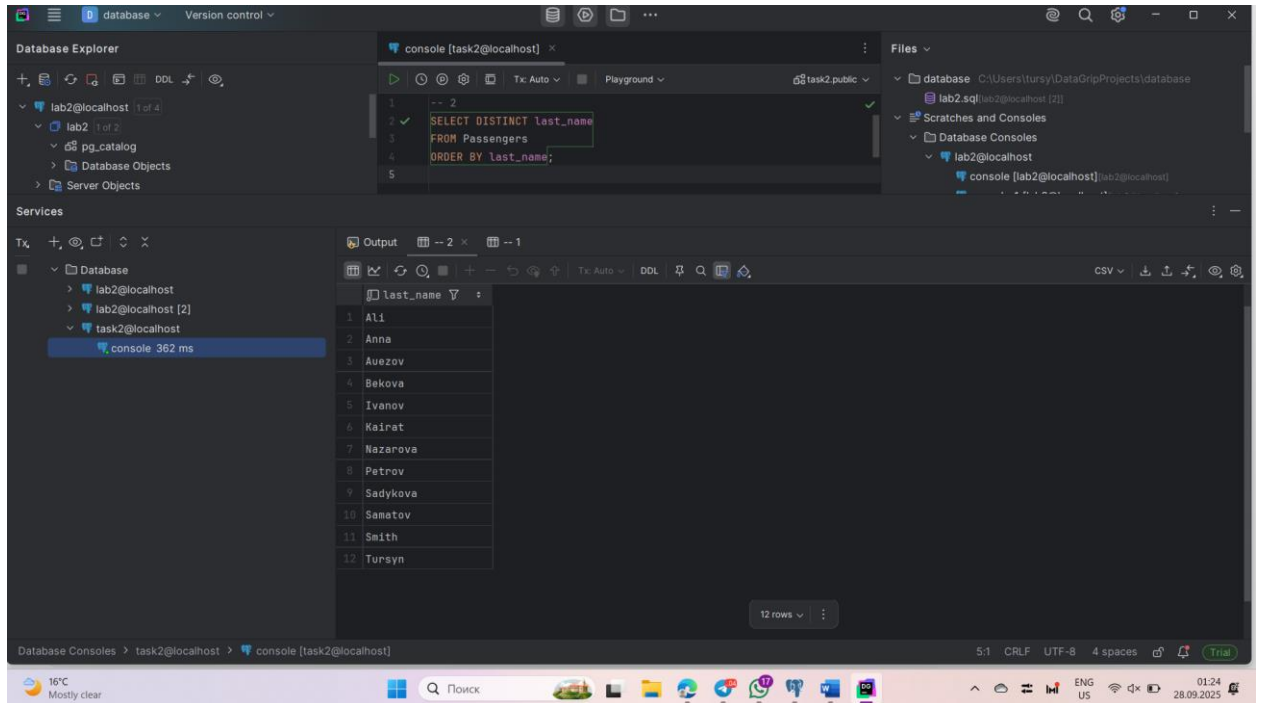
The bottom panel shows the results of the query in a table format. The table has 8 rows and 6 columns: passenger_id, first_name, last_name, date_of_birth, gender, and country_of_citizenship. The results are as follows:

passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship
1	AlI	AlI	1999-05-12	male	Kazakhstan
2	Anna	Anna	1995-03-08	female	Kazakhstan
3	AlI	AlI	1999-05-12	male	Kazakhstan
4	Anna	Anna	1995-03-08	female	Kazakhstan
5	AlI	AlI	1999-05-12	male	Kazakhstan
6	Anna	Anna	1995-03-08	female	Kazakhstan
7	AlI	AlI	1999-05-12	male	Kazakhstan
8	Anna	Anna	1995-03-08	female	Kazakhstan

The interface also shows a Database Explorer on the left with a tree view of the database structure, including tables like pg_catalog, Database Objects, and Server Objects. The bottom status bar indicates the current file is 'task2@localhost' and the console is 'console [task2@localhost]'.

-- 2

```
SELECT DISTINCT last_name  
FROM Passengers  
ORDER BY last_name;
```



-- 3

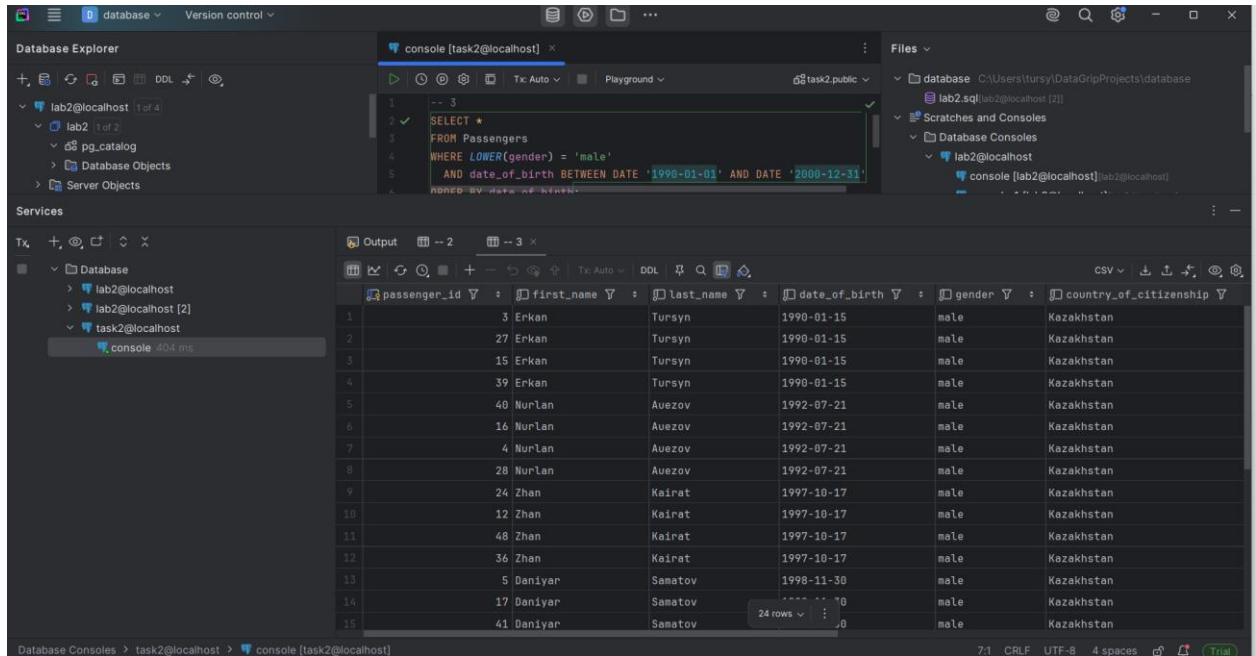
SELECT *

FROM Passengers

WHERE LOWER(gender) = 'male'

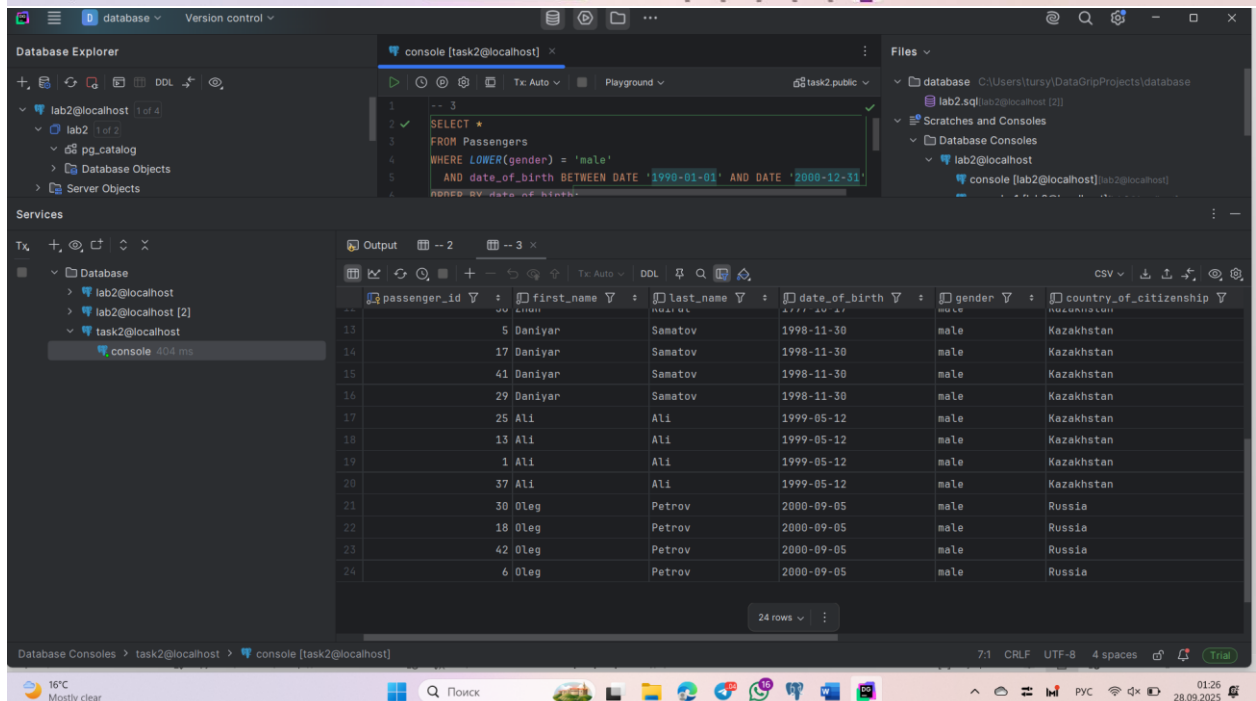
AND date_of_birth BETWEEN DATE '1990-01-01' AND DATE '2000-12-31'

ORDER BY date_of_birth;



```
-- 3
SELECT *
FROM Passengers
WHERE LOWER(gender) = 'male'
AND date_of_birth BETWEEN DATE '1990-01-01' AND DATE '2000-12-31'
ORDER BY date_of_birth;
```

passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship
3	Erkan	Tursyn	1990-01-15	male	Kazakhstan
27	Erkan	Tursyn	1990-01-15	male	Kazakhstan
15	Erkan	Tursyn	1990-01-15	male	Kazakhstan
39	Erkan	Tursyn	1990-01-15	male	Kazakhstan
40	NurLan	Auezov	1992-07-21	male	Kazakhstan
16	NurLan	Auezov	1992-07-21	male	Kazakhstan
4	NurLan	Auezov	1992-07-21	male	Kazakhstan
28	NurLan	Auezov	1992-07-21	male	Kazakhstan
24	Zhan	Kairat	1997-10-17	male	Kazakhstan
12	Zhan	Kairat	1997-10-17	male	Kazakhstan
48	Zhan	Kairat	1997-10-17	male	Kazakhstan
36	Zhan	Kairat	1997-10-17	male	Kazakhstan
5	Daniyar	Samatov	1998-11-30	male	Kazakhstan
17	Daniyar	Samatov	1998-11-30	male	Kazakhstan
41	Daniyar	Samatov	1998-11-30	male	Kazakhstan



```
-- 3
SELECT *
FROM Passengers
WHERE LOWER(gender) = 'male'
AND date_of_birth BETWEEN DATE '1990-01-01' AND DATE '2000-12-31'
ORDER BY date_of_birth;
```

passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship
5	Daniyar	Samatov	1998-11-30	male	Kazakhstan
17	Daniyar	Samatov	1998-11-30	male	Kazakhstan
41	Daniyar	Samatov	1998-11-30	male	Kazakhstan
29	Daniyar	Samatov	1998-11-30	male	Kazakhstan
25	Ali	Ali	1999-05-12	male	Kazakhstan
13	Ali	Ali	1999-05-12	male	Kazakhstan
1	Ali	Ali	1999-05-12	male	Kazakhstan
37	Ali	Ali	1999-05-12	male	Kazakhstan
30	Oleg	Petrov	2000-09-05	male	Russia
18	Oleg	Petrov	2000-09-05	male	Russia
42	Oleg	Petrov	2000-09-05	male	Russia
6	Oleg	Petrov	2000-09-05	male	Russia

--4

SELECT DATE_TRUNC('month', created_at) AS month_start,

SUM(ticket_price) AS total_revenue

FROM Booking

GROUP BY 1

ORDER BY 1;

The screenshot shows a database console interface with a SQL query and its results. The query is as follows:

```
1 SELECT DATE_TRUNC('month', created_at) AS month_start,
2 SUM(ticket_price) AS total_revenue
3 FROM Booking
4 GROUP BY 1
5 ORDER BY 1;
```

The results are displayed in a table with 4 rows:

month_start	total_revenue
2023-10-01 00:00:00.000000	270000
2023-11-01 00:00:00.000000	300000
2024-01-01 00:00:00.000000	810000
2024-02-01 00:00:00.000000	540000

The interface also shows a Database Explorer on the left with a tree view of the database structure, including 'lab2@localhost', 'pg_catalog', and 'Server Objects'. The bottom status bar indicates the current session is 'task2@localhost' and the console is 'console [task2@localhost]'.

--5SELECT f.*

FROM Flights f

JOIN Airport a ON a.airport_id = f.arriving_airport_id

WHERE a.country = 'China';

The screenshot shows a database management tool interface. The top panel displays a SQL query in a console window:

```
1 SELECT f.*
2 FROM Flights f
3 JOIN Airport a ON a.airport_id = f.arriving_airport_id
4 WHERE a.country = 'China';
5
```

The bottom panel shows the results of the query in a table view. The table has 6 columns: flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, and arriving_airport_id. The results are as follows:

flight_id	sch_departure_time	sch_arrival_time	departing_airport_id	arriving_airport_id
1	5 2024-02-10 08:00:00.000000	2024-02-10 12:00:00.000000	1	201
2	6 2024-02-10 08:00:00.000000	2024-02-10 12:00:00.000000	1	201

The interface also includes a Database Explorer on the left, a Services panel at the bottom left, and a status bar at the bottom right showing the current time and date.

--6

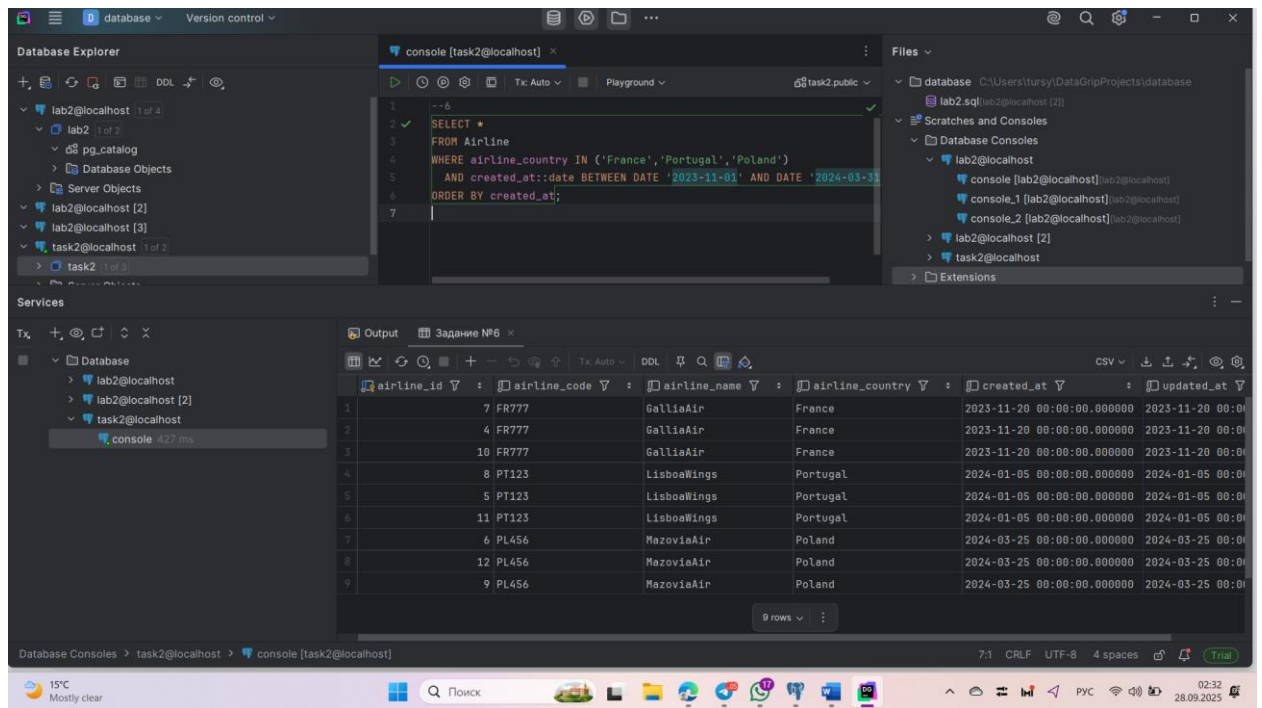
SELECT *

FROM Airline

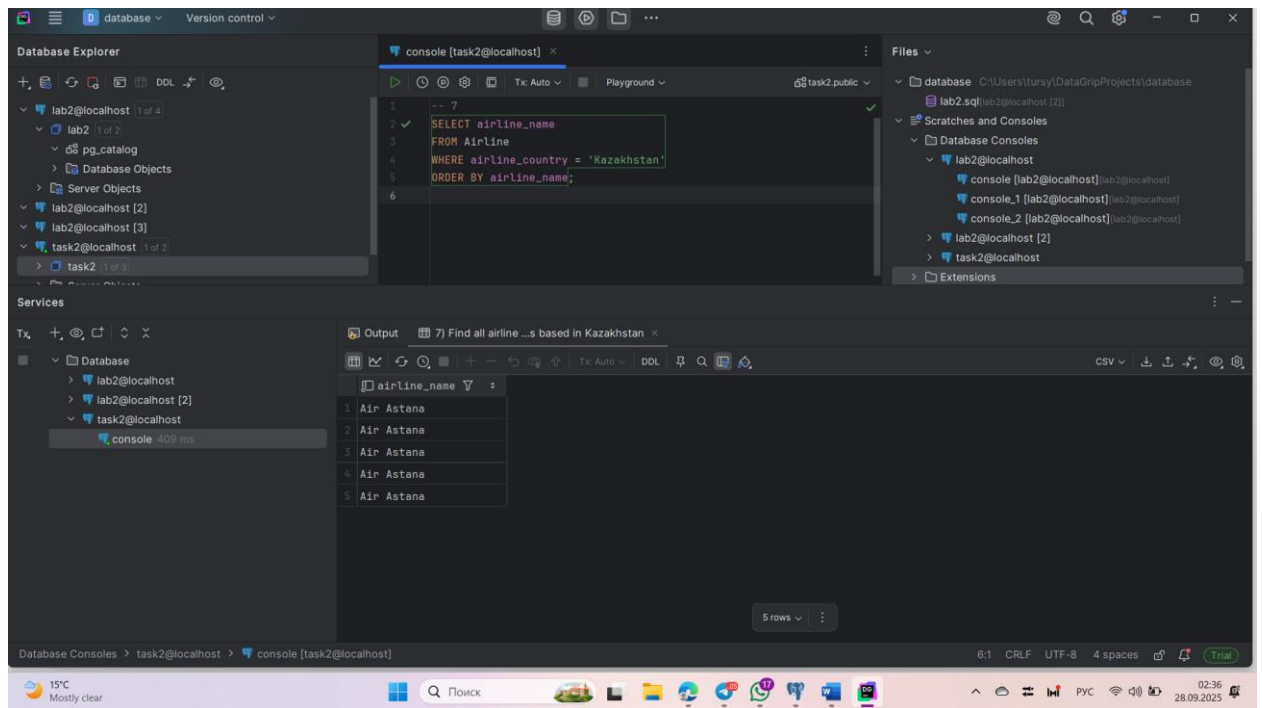
WHERE airline_country IN ('France','Portugal','Poland')

AND created_at::date BETWEEN DATE '2023-11-01' AND DATE '2024-03-31'

ORDER BY created_at;



```
-- 7
SELECT airline_name
FROM Airline
WHERE airline_country = 'Kazakhstan'
ORDER BY airline_name;
```



--8

UPDATE Booking
SET ticket_price = ROUND(ticket_price * 0.90, 2),
updated_at = NOW()
WHERE created_at::date < DATE '2023-11-01'

RETURNING booking_id, ticket_price;

The screenshot shows the DataGrip IDE interface. The central pane displays a SQL query being executed in a console:

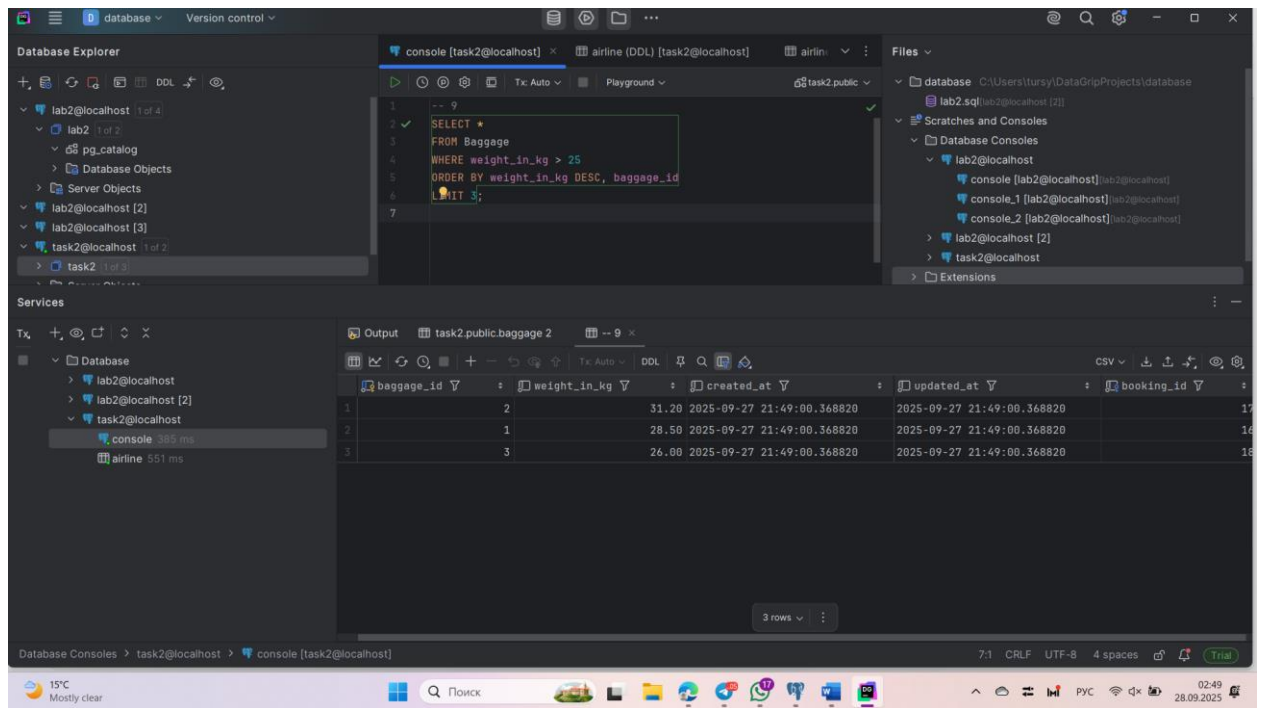
```
1 UPDATE Booking
2 SET ticket_price = ROUND(ticket_price * 0.90, 2),
3    updated_at = NOW()
4 WHERE created_at::date < DATE '2023-11-01'
5 RETURNING booking_id, ticket_price;
6
```

The bottom pane shows the execution results in a table with 7 rows:

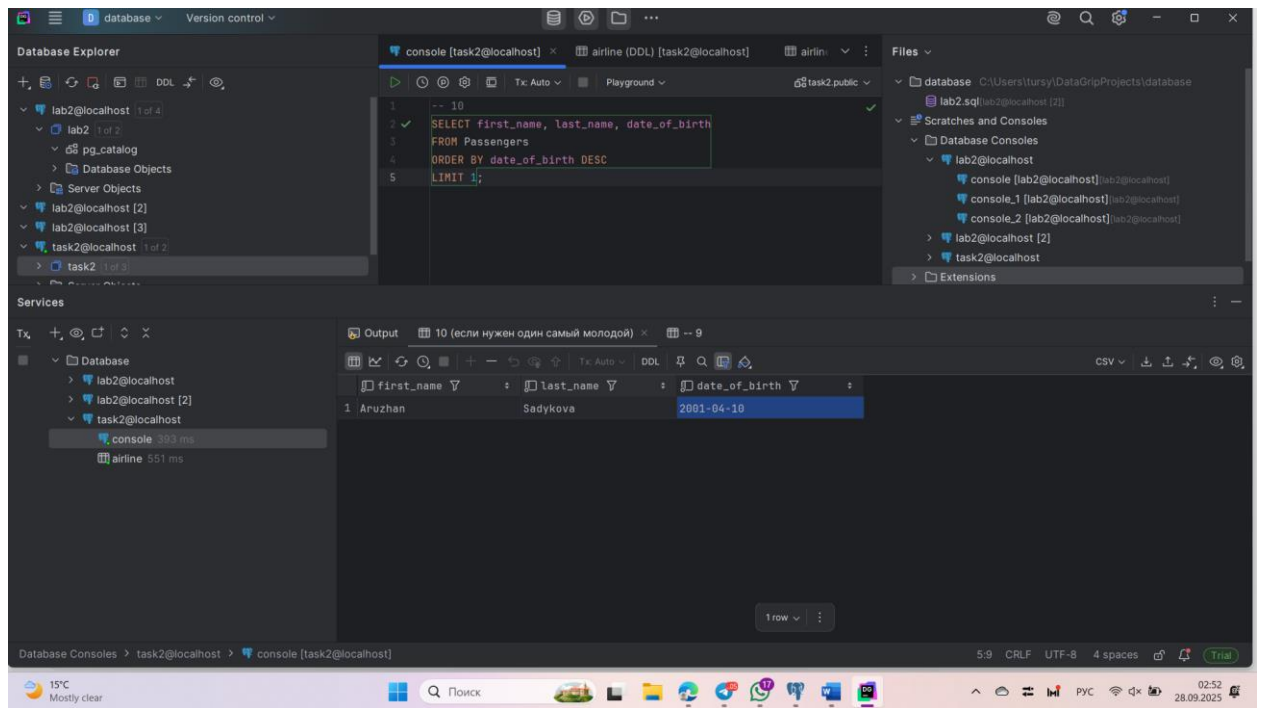
	booking_id	ticket_price
1	46	32805.00
2	16	15690.54
3	21	15690.54
4	26	15690.54
5	31	15690.54
6	36	15690.54
7	41	15690.54

The left sidebar shows the Database Explorer with a tree view of the database structure, including 'lab2@localhost' and 'task2@localhost'. The bottom status bar indicates the database is 'task2@localhost' and the console is 'console [task2@localhost]'.

```
-- 9
SELECT *
FROM Baggage
WHERE weight_in_kg > 25
ORDER BY weight_in_kg DESC, baggage_id
LIMIT 3;
```

```
-- 10
SELECT first_name, last_name, date_of_birth
FROM Passengers
ORDER BY date_of_birth DESC
LIMIT 1;
```



-- 11

```
SELECT booking_platform,  
       MIN(ticket_price) AS min_price  
FROM Booking
```

GROUP BY booking_platform

ORDER BY booking_platform;

The screenshot shows a database IDE with a SQL query editor and a results pane. The query is as follows:

```
1 -- 11
2 SELECT booking_platform,
3        MIN(ticket_price) AS min_price
4 FROM Booking
5 GROUP BY booking_platform
6 ORDER BY booking_platform;
```

The results pane displays a table with 4 rows and 2 columns: booking_platform and min_price.

booking_platform	min_price
AirOnline	98000
AviaKZ	15690.54
FlyApp	65000
TestPlatform	32805

-- 12

SELECT *

FROM Airline

WHERE airline_code ~ '\d' -- регулярка «есть цифра»

ORDER BY airline_code;

The screenshot shows a database management tool interface. The top panel displays a SQL query in a console window:

```
-- 12
SELECT *
FROM Airline
```

The bottom panel shows the results of the query in a table format. The table has 17 rows and 7 columns: airline_id, airline_code, airline_name, airline_country, created_at, and updated_at. The data is sorted by airline_code in descending order.

airline_id	airline_code	airline_name	airline_country	created_at	updated_at
7	FR777	GalliaAir	France	2023-11-20 00:00:00.000000	2023-11-20 00:00:00.000000
4	FR777	GalliaAir	France	2023-11-20 00:00:00.000000	2023-11-20 00:00:00.000000
10	FR777	GalliaAir	France	2023-11-20 00:00:00.000000	2023-11-20 00:00:00.000000
3	KZ001	KazAir	Turkey	2025-09-27 20:08:54.422102	2025-09-27 20:08:54.422102
2	KZ001	KazAir	Turkey	2025-09-27 20:08:29.244920	2025-09-27 20:08:29.244920
1	KZ001	KazAir	Turkey	2025-09-27 20:07:55.541511	2025-09-27 20:07:55.541511
17	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:51.285393	2025-09-27 21:35:51.285393
13	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:11.584944	2025-09-27 21:35:11.584944
14	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:13.673771	2025-09-27 21:35:13.673771
15	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:49.008345	2025-09-27 21:35:49.008345
16	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:50.340206	2025-09-27 21:35:50.340206
12	PL456	MazoviaAir	Poland	2024-03-25 00:00:00.000000	2024-03-25 00:00:00.000000
9	PL456	MazoviaAir	Poland	2024-03-25 00:00:00.000000	2024-03-25 00:00:00.000000
6	PL456	MazoviaAir	Poland	2024-03-25 00:00:00.000000	2024-03-25 00:00:00.000000
5	PT123	LisboaWings	Portugal	2024-01-05 00:00:00.000000	2024-01-05 00:00:00.000000
8	PT123	LisboaWings	Portugal	2024-01-05 00:00:00.000000	2024-01-05 00:00:00.000000
11	PT123	LisboaWings	Portugal	2024-01-05 00:00:00.000000	2024-01-05 00:00:00.000000

-- 13

SELECT *

FROM Airline

ORDER BY created_at DESC

LIMIT 5;

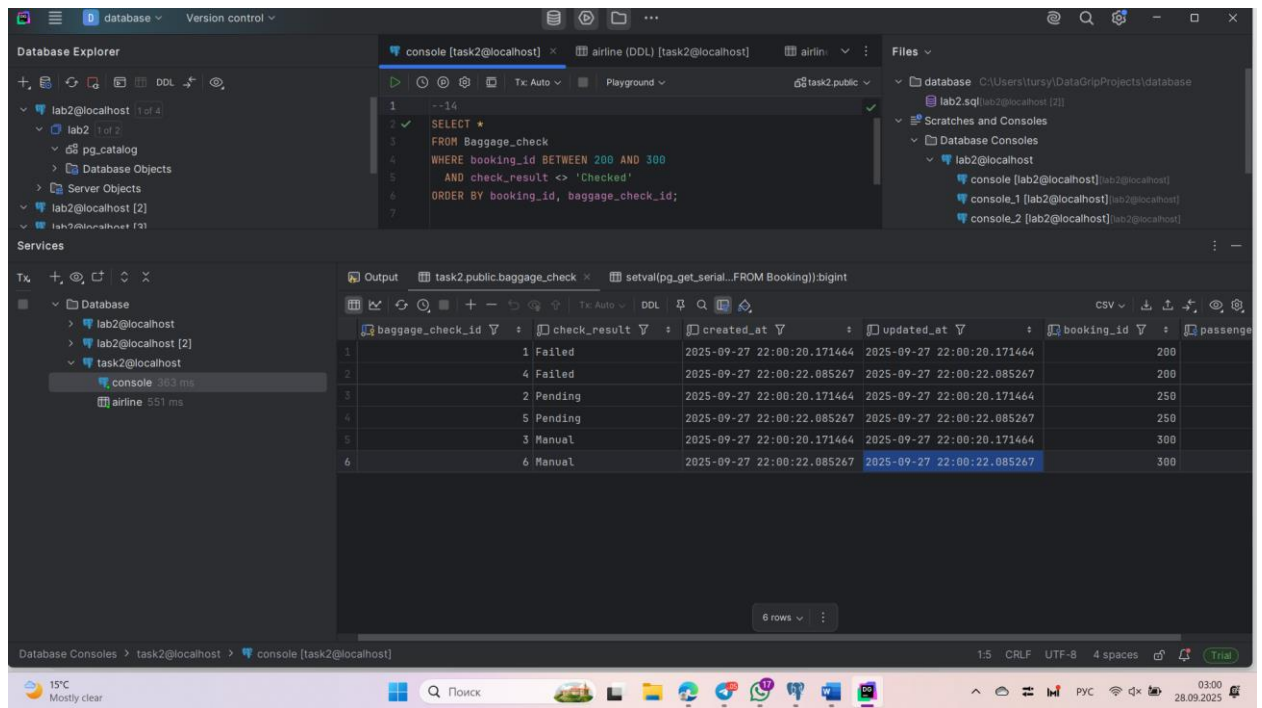
The screenshot shows the DataGrip IDE interface. The top-left pane displays the 'Database Explorer' with a tree view of the database structure. The top-right pane shows the 'Files' view. The central editor pane contains a SQL query:

```
-- 13
SELECT *
FROM Airline
ORDER BY created_at DESC
LIMIT 5;
```

The bottom pane shows the 'Output' window with a table of results. The table has 7 columns: 'airline_id', 'airline_code', 'airline_name', 'airline_country', 'created_at', and 'updated_at'. The results are as follows:

airline_id	airline_code	airline_name	airline_country	created_at	updated_at
17	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:51.285393	2025-09-27 21:35:51.285393
16	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:50.340206	2025-09-27 21:35:50.340206
15	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:49.008345	2025-09-27 21:35:49.008345
14	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:13.673771	2025-09-27 21:35:13.673771
13	KZ101	Air Astana	Kazakhstan	2025-09-27 21:35:11.584944	2025-09-27 21:35:11.584944

```
--14
SELECT *
FROM Baggage_check
WHERE booking_id BETWEEN 200 AND 300
AND check_result <> 'Checked'
ORDER BY booking_id, baggage_check_id;
```



--15

SELECT *

FROM Baggage_check

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

AND updated_at < created_at

ORDER BY booking_id, baggage_check_id;

Database Explorer

lab2@localhost 1 of 4

pg_catalog

Database Objects

Server Objects

lab2@localhost [2]

task2@localhost [2]

console [task2@localhost]

airline (DDL) [task2@localhost]

airlin...

task2 public

Files

database C:\Users\tursy\DataGripProjects\database

lab2.sql [lab2@localhost (2)]

Scratches and Consoles

Database Consoles

lab2@localhost

console [lab2@localhost] [lab2@localhost]

console_1 [lab2@localhost] [lab2@localhost]

console_2 [lab2@localhost] [lab2@localhost]

Services

Database

lab2@localhost

lab2@localhost [2]

task2@localhost

console 411 ms

airline 551 ms

Output task2.public.baggage_check --15

check_id check_result created_at updated_at booking_id passenger_id

1 7 Manual 2023-11-20 10:00:00.000000 2023-11-05 00:00:00.000000 16 1

2 8 Pending 2023-12-15 12:00:00.000000 2023-12-01 09:30:00.000000 17 1

3 1 Failed 2023-10-20 12:00:00.000000 2023-10-05 09:00:00.000000 200 1

4 4 Failed 2023-10-20 12:00:00.000000 2023-10-05 09:00:00.000000 200 1

5 2 Pending 2023-10-28 18:30:00.000000 2023-10-10 08:15:00.000000 250 1

6 5 Pending 2023-10-28 18:30:00.000000 2023-10-10 08:15:00.000000 250 1

7 3 Manual 2023-10-31 23:00:00.000000 2023-10-01 00:01:00.000000 300 1

8 6 Manual 2023-10-31 23:00:00.000000 2023-10-01 00:01:00.000000 300 1

8 rows

Database Consoles > task2@localhost > console [task2@localhost]

5:30 CRLF UTF-8 4 spaces Trial

15°C Mostly clear

Поиск

03:04 28.09.2025

