

TRANSACTION.

1. A passenger cancels their booking. You need to remove the booking for the flight. Ensure the ‘booking’ table no longer contains the booking. Simulate an error to test rollback (for example, invalid booking_id).

The screenshot shows three separate pgAdmin 4 sessions, each demonstrating a different aspect of transactions:

- Session 1 (Top):** Shows a successful transaction. The query window contains the following SQL code:

```
1 BEGIN;
2
3 v DELETE FROM booking_flight
4 WHERE booking_id = 1;
5
6 v DELETE FROM boarding_pass
7 WHERE booking_id = 1;
8
9 v DELETE FROM baggage
10 WHERE booking_id = 1;
11
12 v DELETE FROM booking
13 WHERE booking_id = 1;
14
15 COMMIT;
```

The status bar at the bottom indicates "Query returned successfully in 71 msec."

- Session 2 (Middle):** Shows a failed transaction due to an invalid booking_id. The query window contains:

```
1 BEGIN;
2
3 v DELETE FROM booking_flight
4 WHERE booking_id = 999999;
5
6 v DELETE FROM booking
7 WHERE booking_id = 999999;
8
9 ROLLBACK;
```

The status bar at the bottom indicates "ROLLBACK".

- Session 3 (Bottom):** Shows a successful transaction. The query window contains the same code as Session 1.

2. Rescheduling a flight. You need to reschedule a flight. Verify the ‘flights’ table reflects the new departure time. Simulate an error to test rollback (for example, invalid flight_id).

File Object Tools Edit View Window Help

Welcome Sai_db/postgres@PostgreSQL 17*

Query Query History

```

1 BEGIN;
2
3 UPDATE flights
4 SET scheduled_departure = scheduled_departure + INTERVAL '3 hours'
5 WHERE flight_id = 99999;
6
7 ROLLBACK;

```

Data Output Messages Notifications

ROLLBACK

Query returned successfully in 77 msec.

Total rows: Query complete 00:00:00.077

File Object Tools Edit View Window Help

Welcome Sai_db/postgres@PostgreSQL 17*

Query Query History

```

1 BEGIN;
2
3 UPDATE flights
4 SET scheduled_departure = scheduled_departure + INTERVAL '3 hours'
5 WHERE flight_id = 10;
6
7 COMMIT;

```

Data Output Messages Notifications

COMMIT

Query returned successfully in 47 msec.

Total rows: Query complete 00:00:00.047

Пойск

23:50 25.11.2025

3. Updating ticket prices. You need to decrease the ticket price for a specific flight for all existing bookings. If an error occurs, no changes should be applied.

The screenshot shows the pgAdmin 4 interface. At the top, the menu bar includes File, Object, Tools, Edit, View, Window, and Help. A toolbar with various icons is located above the main query editor. The title bar says "Welcome Sai_db/postgres@PostgreSQL 17*". The main area has tabs for "Query" and "Query History", with "Query" selected. Below the tabs is a code editor containing the following SQL script:

```
1 BEGIN;
2
3 ✓ UPDATE booking
4   SET price = price - 5000,
5       update_at = NOW()
6 WHERE booking_id IN (
7   SELECT booking_id
8     FROM booking_flight
9    WHERE flight_id = 80
10 );
11
12 COMMIT;
```

Below the code editor, there are three tabs: "Data Output", "Messages", and "Notifications", with "Messages" being the active tab. The message pane displays:

COMMIT
Query returned successfully in 50 msec.

In the bottom status bar, it says "Total rows: Query complete 00:00:00.050". On the right side of the status bar, there is a green notification box with a checkmark and the text "✓ Query returned successfully in 50 msec. X".

4. A passenger updates their details. Ensure the update is reflected across all associated records, including bookings.

The image displays three separate sessions of the pgAdmin 4 interface, each showing a successful execution of a PostgreSQL query.

Session 1:

```

1 BEGIN;
2
3 UPDATE passengers
4 SET
5   first_name = 'Aktumar',
6   last_name = 'Ablekhan',
7   country_of_citizenship = 'Kazakhstan'
8 WHERE passenger_id = 201;
9
10 COMMIT;

```

Session 2:

```

1 BEGIN;
2
3 UPDATE passengers
4 SET first_name = 'Aidos'
5 WHERE passenger_id = 999999;
6
7 ROLLBACK;

```

Session 3:

```

1 BEGIN;
2
3 UPDATE passengers
4 SET
5   first_name = 'Cold weather',
6   last_name = 'Now';
7
8 COMMIT;

```

In all three sessions, the status bar at the bottom right indicates a successful execution with a green checkmark icon and the message "Query returned successfully in [msec]". The timestamp for Session 3 is 26.11.2025.

5. A new passenger is registered, and a booking is created. Ensure the new passenger is added and the booking succeeds.

The screenshot shows the pgAdmin 4 interface with a query editor window. The query is a multi-step transaction:

```
1 BEGIN;
2
3 INSERT INTO passengers (passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, passport_number)
4 VALUES (202, 'Laura', 'Suleimenova', '2003-11-02', 'Female', 'Kazakhstan', 'P887744')
5 RETURNING passenger_id;
6
7 INSERT INTO booking (booking_id, passenger_id, booking_platform, price, created_at, update_at, status)
8 VALUES (501, 202, 'Website', 50000, NOW(), NOW(), 'Confirmed')
9 RETURNING booking_id;
10
11 INSERT INTO booking_flight (booking_flight_id, booking_id, flight_id, created_at, update_at)
12 VALUES (1001, 501, 80, NOW(), NOW());
13
14 COMMIT;
```

The "Messages" tab shows the output:

```
COMMIT
```

Below the messages, a status bar indicates:

- Total rows: 0
- Query complete 00:00:00.104

A green message box at the bottom right states: "✓ Query returned successfully in 104 msec. X".

6. Increase the ticket price for all bookings on a specific flight by a fixed amount.

pgAdmin 4

Welcome Sai_db/postgres@PostgreSQL 17*

```

1 BEGIN;
2
3 UPDATE booking
4 SET price = price + 5000
5 WHERE booking_id IN (
6     SELECT booking_id
7     FROM booking_flight
8     WHERE flight_id = 999999
9 );
10
11 ROLLBACK;

```

Data Output Messages Notifications

ROLLBACK

Query returned successfully in 71 msec.

Total rows: Query complete 00:00:00.071

10°C Sunny

10:30 ENG 26.11.2025

pgAdmin 4

Welcome Sai_db/postgres@PostgreSQL 17*

```

1 BEGIN;
2
3 UPDATE booking
4 SET price = price + 5000
5 WHERE booking_id IN (
6     SELECT booking_id
7     FROM booking_flight
8     WHERE flight_id = 80
9 );
10
11 COMMIT;

```

Execute script [F5]

Data Output Messages Notifications

COMMIT

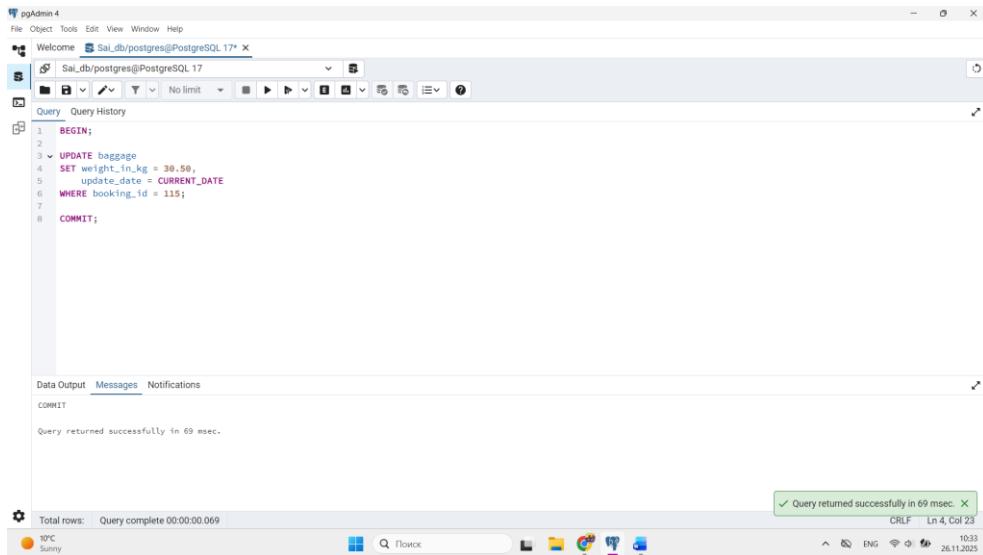
Query returned successfully in 68 msec.

Total rows: Query complete 00:00:00.068

10°C Sunny

10:29 ENG 26.11.2025

7. Update a baggage weight. A passenger updates the declared weight of their baggage. Ensure that the change is correctly reflected in the database.



```
pgAdmin 4
File Object Tools Edit View Window Help
Welcome Sai_db/postgres@PostgreSQL 17* ×
Sai_db/postgres@PostgreSQL 17* ×
Query History
1 BEGIN;
2
3 UPDATE baggage
4 SET weight_in_kg = 39.50,
5 update_date = CURRENT_DATE
6 WHERE booking_id = 115;
7
8 COMMIT;
```

Data Output Messages Notifications

COMMIT

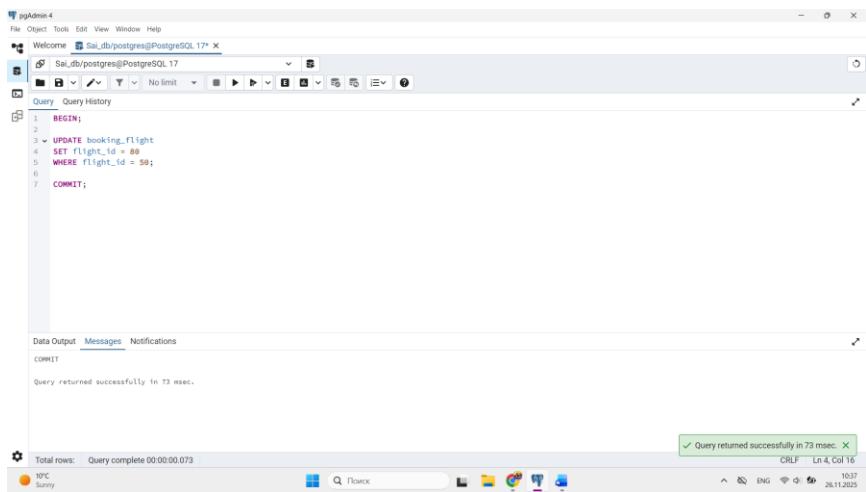
Query returned successfully in 69 msec.

Total rows: Query complete 00:00:00.069

10°C Sunny 10:33 CRLF Ln 4, Col 23

26.11.2023

8. Apply a discount to a booking for a specific passenger. If any error occurs, roll back.



```
pgAdmin 4
File Object Tools Edit View Window Help
Welcome Sai_db/postgres@PostgreSQL 17* ×
Sai_db/postgres@PostgreSQL 17* ×
Query History
1 BEGIN;
2
3 UPDATE bookings_flight
4 SET flight_id = 58
5 WHERE flight_id = 56;
6
7 COMMIT;
```

Data Output Messages Notifications

COMMIT

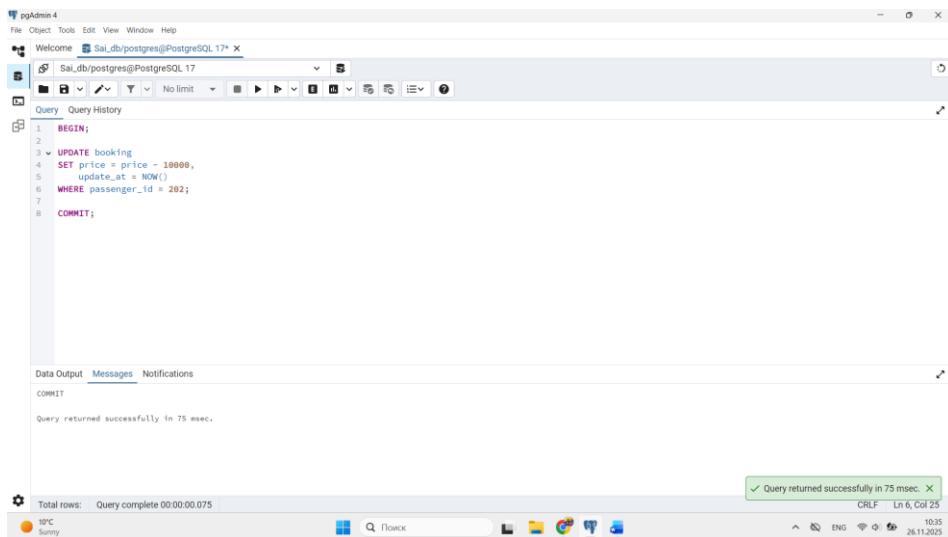
Query returned successfully in 73 msec.

Total rows: Query complete 00:00:00.073

10°C Sunny 10:37 CRLF Ln 4, Col 16

26.11.2023

9. Reschedule all bookings for a flight to a new flight.



```
pgAdmin 4
File Object Tools Edit View Window Help
Welcome Sai_db/postgres@PostgreSQL 17* ×
Sai_db/postgres@PostgreSQL 17* ×
Query History
1 BEGIN;
2
3 UPDATE booking
4 SET price = price - 10000,
5 update_at = NOW()
6 WHERE passenger_id = 202;
7
8 COMMIT;
```

Data Output Messages Notifications

COMMIT

Query returned successfully in 75 msec.

Total rows: Query complete 00:00:00.075

10°C Sunny 10:35 CRLF Ln 6, Col 25

26.11.2023

