

Laboratory work 9

We continue to work with the database from the previous laboratory works.

Take a full-page screenshot that covers the code and results of each task.

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 the pgAdmin 4 interface with the Object Explorer on the left and a query editor on the right. The query editor contains the following SQL code:

```
BEGIN;
DELETE FROM booking WHERE booking_id = 123;
DELETE FROM booking WHERE booking_id = -999;
ROLLBACK;
```

The Data Output tab shows the results of the query. An error message is displayed:

ERROR: update or delete on table "booking" violates foreign key constraint "baggage_check_booking_id_fkey" on table "baggage_check"
Key (booking_id)=(123) is still referenced from table "baggage_check".

SQL state: 23503
Detail: Key (booking_id)=(123) is still referenced from table "baggage_check".

Total rows: Query complete 00:00:00.043

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).

The screenshot shows the pgAdmin 4 interface with a database connection to 'public.flights/data...' and a session to 'databases_sj/postgres@PostgreSQL 17*'. The left sidebar shows various schema objects like Domain, FTS Cc, FTS Dir, FTS Pa, FTS Te, Foreign, Function, Materialized View, Operator, Procedure, Sequence, Table, and View. The 'Tables' section is expanded, showing tables such as airlin, airpo, bagr, bagt, boar, boot, flight, and C. The main query editor contains the following SQL code:

```

1 BEGIN;
2
3 UPDATE flights
4 SET scheduled_departure = '2025-12-15 14:30:00',
5      update_at = CURRENT_DATE
6 WHERE flight_id = 101;
7
8 SELECT flight_id, scheduled_departure
9 FROM flights
10 WHERE flight_id = 101;
11
12 UPDATE flights
13 SET scheduled_departure = '2025-12-20 09:00:00'
14 WHERE flight_id = 999999;
15
16 COMMIT;

```

The 'Messages' tab in the results pane shows the message 'Query returned successfully in 40 msec.' and the status bar at the bottom indicates 'Total rows: 0 Query complete 00:00:00.040'.

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 with the same database connection and session as the previous screenshot. The left sidebar shows the same schema objects. The 'Tables' section is expanded, showing the same tables as before. The main query editor contains the following SQL code:

```

1 BEGIN;
2
3 UPDATE booking b
4 SET price = price * 0.9,
5     update_at = CURRENT_DATE
6 FROM booking_flight bf
7 WHERE b.booking_id = bf.booking_id
8     AND bf.flight_id = 101;
9
10 SELECT b.booking_id, b.price
11 FROM booking b
12 JOIN booking_flight bf ON b.booking_id = bf.booking_id
13 WHERE bf.flight_id = 101;
14
15 COMMIT;
16

```

The 'Messages' tab in the results pane shows the message 'Query returned successfully in 51 msec.' and the status bar at the bottom indicates 'Total rows: 0 Query complete 00:00:00.051'. A green success message box is visible in the bottom right corner.

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

The screenshot shows the pgAdmin 4 interface with a database connection to 'public.flights/data...' and a query editor window. The query is:

```

1 BEGIN;
2
3 UPDATE passengers
4 SET first_name = 'John',
5 last_name = 'Doe',
6 date_of_birth = '1990-05-10',
7 country_of_residence = 'Kazakhstan',
8 update_at = CURRENT_DATE
9 WHERE passenger_id = 201;
10
11 UPDATE booking b
12 SET update_at = CURRENT_DATE
13 FROM passengers p
14 WHERE b.passenger_id = p.passenger_id
15 AND p.passenger_id = 201;
16
17 SELECT *
18 FROM passengers
19 WHERE passenger_id = 201;
20
21 SELECT *
22 FROM booking
23 WHERE passenger_id = 201;
24
25 COMMIT;

```

The 'Data Output' tab shows the result: 'Query returned successfully in 44 msec.' and 'Total rows: 0'. A green status bar at the bottom right indicates 'Query returned successfully in 44 msec. LF Ln 16, Col 1'.

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 database connection to 'public.flights/data...' and a query editor window. The query is:

```

1 BEGIN;
2
3 INSERT INTO passengers (
4 passenger_id, first_name, last_name, date_of_birth, gender,
5 country_of_citizenship, country_of_residence, passport_number,
6 created_at, update_at
7 ) VALUES (
8 201, 'Alice', 'Smith', '1995-08-20', 'Female',
9 'Kazakhstan', 'Kazakhstan', 'AB1234567',
10 CURRENT_DATE, CURRENT_DATE
11 );
12
13 INSERT INTO booking (
14 booking_id, passenger_id, booking_platform, created_at, update_at, status, pr
15 ) VALUES (
16 501, 201, 'DropBox', CURRENT_DATE, CURRENT_DATE, 'Confirmed', 1500.00
17 );
18
19 SELECT * FROM passengers WHERE passenger_id = 201;
20 SELECT * FROM booking WHERE passenger_id = 201;
21
22 COMMIT;

```

The 'Data Output' tab shows the result: 'Query returned successfully in 53 msec.' and 'Total rows: 0'. A green status bar at the bottom right indicates 'Query returned successfully in 53 msec. LF Ln 18, Col 1'.

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

The screenshot shows the pgAdmin interface with a database connection to 'databases_sj/postgres@PostgreSQL 17*'. The left sidebar displays the schema tree under 'public.flights/data...'. The main window contains a query editor with the following SQL code:

```

BEGIN;
UPDATE booking b
SET price = price + 500,
    update_at = CURRENT_DATE
FROM booking_flight bf
WHERE b.booking_id = bf.booking_id
    AND bf.flight_id = 101;
SELECT b.booking_id, b.price
FROM booking b
JOIN booking_flight bf ON b.booking_id = bf.booking_id
WHERE bf.flight_id = 101;
COMMIT;

```

The 'Data Output' tab shows the results of the SELECT statement:

```

Query returned successfully in 41 msec.

Total rows: 1 Query complete 00:00:00.041

```

A green status bar at the bottom right indicates: ✓ Query returned successfully in 41 msec. LF Ln 2, Col 1.

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

The screenshot shows the pgAdmin interface with a database connection to 'database_subj/postgres@PostgreSQL 17*'. The left sidebar displays the schema tree under 'PostgreSQL FastAPI (2)'. The main window contains a query editor with the following SQL code:

```

BEGIN;
UPDATE baggage
SET weight_in_kg = 23.5,
    update_date = CURRENT_DATE
WHERE baggage_id = 301;
SELECT baggage_id, weight_in_kg
FROM baggage
WHERE baggage_id = 301;
COMMIT;

```

The 'Messages' tab shows an error message:

```

ERROR: new row for relation "baggage" violates check constraint "baggage_check"
Failing row contains (301, 23.50, 2024-02-04, 2025-11-25, 47).

SQL state: 23514
Detail: Failing row contains (301, 23.50, 2024-02-04, 2025-11-25, 47).

```

A green status bar at the bottom right indicates: ✓ Query returned successfully in 41 msec. LF Ln 2, Col 1.

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

The screenshot shows the pgAdmin 4 interface with a single query window open. The query is:

```
BEGIN;
UPDATE booking
SET price = price * 0.9,
    update_at = CURRENT_DATE
WHERE passenger_id = 201;

SELECT booking_id, passenger_id, price
FROM booking
WHERE passenger_id = 201;

COMMIT;
```

The results pane shows the message: "Query returned successfully in 53 msec." and the status bar indicates "Query complete 00:00:00.053".

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

The screenshot shows the pgAdmin 4 interface with a single query window open. The query is:

```
BEGIN;
UPDATE booking_flight
SET flight_id = 202,
    update_at = CURRENT_DATE
WHERE flight_id = 101;

SELECT booking_id, flight_id
FROM booking_flight
WHERE flight_id = 202;

COMMIT;
```

The results pane shows the message: "Query returned successfully in 43 msec." and the status bar indicates "Query complete 00:00:00.043".