

1\_

The screenshot displays two database sessions in a management tool. Both sessions are connected to the 'postgres' database on 'localhost'.

**Session 1 (Top):**

- Database Explorer shows the schema with tables like 'airline', 'booking', 'flight', etc., and their columns.
- Console output:

```
1 ✓ BEGIN;
2 ✓
3 ✓
4 ✓ DELETE FROM booking_flight WHERE booking_id = 123;
5 ✓ DELETE FROM booking WHERE booking_id = 123;
6 ✓ COMMIT;
```
- Services pane shows the database connection.
- Output pane shows the results of the transaction:

```
[2025-12-03 11:14:17] completed in 3 ms
[2025-12-03 12:13:52] dwork.public> BEGIN
[2025-12-03 12:13:52] completed in 3 ms
[2025-12-03 12:13:52] dwork.public> DELETE FROM booking_flight WHERE booking_id = 123
[2025-12-03 12:13:52] completed in 6 ms
[2025-12-03 12:13:52] dwork.public> DELETE FROM booking WHERE booking_id = 123
[2025-12-03 12:13:52] completed in 4 ms
[2025-12-03 12:13:52] dwwork.public> COMMIT
[2025-12-03 12:13:52] completed in 1 ms
```

**Session 2 (Bottom):**

- Database Explorer shows the same schema.
- Console output:

```
1 ✓ BEGIN;
2 ✓
3 ✓
4 ✓ DELETE FROM booking_flight WHERE booking_id = 1199191;
5 ✓ DELETE FROM booking WHERE booking_id = 1199191;
6 ✓ ROLLBACK;
```
- Services pane shows the database connection.
- Output pane shows the results of the transaction:

```
[2025-12-03 12:13:52] completed in 4 ms
[2025-12-03 12:13:52] dwork.public> COMMIT
[2025-12-03 12:13:52] completed in 1 ms
[2025-12-03 12:20:03] dwork.public> BEGIN
[2025-12-03 12:20:03] completed in 3 ms
[2025-12-03 12:20:03] dwork.public> DELETE FROM booking_flight WHERE booking_id = 1199191
[2025-12-03 12:20:03] completed in 1 ms
[2025-12-03 12:20:03] dwork.public> DELETE FROM booking WHERE booking_id = 1199191
[2025-12-03 12:20:03] completed in 1 ms
[2025-12-03 12:20:03] dwwork.public> ROLLBACK
[2025-12-03 12:20:03] completed in 2 ms
```

2\_

The screenshot shows two separate database sessions in a PostgreSQL interface. Both sessions are connected to the 'postgres' user on 'localhost'.

**Session 1 (Top):**

```
1 BEGIN;
2 UPDATE flights
3   SET scheduled_departure = '2025-12-05',
4       scheduled_arrival   = '2025-12-06'
5 WHERE flight_id = 101;
6
7 COMMIT;
```

**Session 2 (Bottom):**

```
1 BEGIN;
2 UPDATE flights
3   SET scheduled_departure = '1999-12-05',
4       scheduled_arrival   = '1999-12-06'
5 WHERE flight_id = 101;
6
7 ROLLBACK;
```

In both sessions, the UPDATE statement is identical, changing the scheduled departure and arrival dates for flight ID 101. The first session commits the changes, while the second session rolls them back. The output shows the execution time for each step and the final state of the transaction.

3\_

The screenshot shows two separate database sessions in a PostgreSQL interface.

**Session 1 (Top):**

- Database: postgres@localhost
- Query:

```
1 ✓ BEGIN;
2 ✓ UPDATE booking b
3   SET price = price * 0.9,
4     update_at = CURRENT_DATE
5   FROM booking_flight bf
6   WHERE b.booking_id = bf.booking_id
7     AND bf.flight_id = 101;
8
9 ✓ COMMIT;
```

- Output:

```
[2025-12-03 12:22:08] dbwork.public> BEGIN
[2025-12-03 12:22:08] completed in 2 ms
[2025-12-03 12:22:08] dbwork.public> UPDATE booking b
      SET price = price * 0.9,
          update_at = CURRENT_DATE
      FROM booking_flight bf
      WHERE b.booking_id = bf.booking_id
        AND bf.flight_id = 101
[2025-12-03 12:22:08] completed in 4 ms
[2025-12-03 12:22:08] dbwork.public> COMMIT
[2025-12-03 12:22:08] completed in 1 ms
```

**Session 2 (Bottom):**

- Database: postgres@localhost
- Query:

```
1 ✓ BEGIN;
2 ✓ UPDATE booking b
3   SET price = price * 0.9,
4     update_at = CURRENT_DATE
5   FROM booking_flight bf
6   WHERE b.booking_id = bf.booking_id
7     AND bf.flight_id = 101;
8
9 ✓ ROLLBACK;
```

- Output:

```
[2025-12-03 12:22:30] dbwork.public> BEGIN
[2025-12-03 12:22:30] completed in 1 ms
[2025-12-03 12:22:30] dbwork.public> UPDATE booking b
      SET price = price * 0.9,
          update_at = CURRENT_DATE
      FROM booking_flight bf
      WHERE b.booking_id = bf.booking_id
        AND bf.flight_id = 101
[2025-12-03 12:22:35] completed in 2 ms
[2025-12-03 12:22:36] dbwork.public> ROLLBACK
[2025-12-03 12:22:36] completed in 2 ms
```

4\_

The screenshot shows a PostgreSQL database console interface. The Database Explorer sidebar on the left lists the schema structure of the 'dbwork' database, including tables like 'airline', 'booking', 'flight', and 'passenger'. The main area is a 'console' tab where a transaction is being run. The transaction starts with a BEGIN statement, followed by an UPDATE query targeting the 'passenger' table. The update sets the first name to 'John' and the last name to 'Doe', with other fields like date of birth and country of residence set to specific values. A WHERE clause filters the update to passenger ID 123. The transaction is then committed. The output window below shows the executed SQL statements and their results, indicating one row was affected.

```
1 ✓ BEGIN;
2
3 ✓ UPDATE passengers
4   SET first_name = 'John',
5       last_name = 'Doe',
6       date_of_birth = '1990-01-01',
7       country_of_residence = 'USA',
8       update_at = CURRENT_DATE
9   WHERE passenger_id = 123;
10
11 ✓ UPDATE booking b
12   SET update_at = CURRENT_DATE
13   WHERE b.passenger_id = 123;
14
15 ✓ COMMIT;
```

```
Les Cours - Due ,  
date_of_birth = '1990-01-01',  
country_of_residence = 'USA',  
update_at = CURRENT_DATE  
WHERE passenger_id = 123  
[2025-12-03 12:24:53] 1 row effected in 5 ms  
[2025-12-03 12:24:53] dbwork.public> UPDATE booking b  
      SET update_at = CURRENT_DATE  
      WHERE b.passenger_id = 123  
[2025-12-03 12:24:53] completed in 1 ms  
[2025-12-03 12:24:53] dbwork.public COMMIT  
[2025-12-03 12:24:53] completed in 1 ms
```

5\_

The screenshot shows a PostgreSQL database console interface. The Database Explorer sidebar on the left lists the schema structure of the 'dbwork' database, including tables like 'airline', 'booking', 'flight', and 'passenger'. The main area is a 'console' tab where a transaction is being run. The transaction starts with a BEGIN statement, followed by three separate INSERT queries. The first query inserts a new passenger record with ID 2001, first name 'Alice', last name 'Smith', date of birth '1995-06-15', country of residence 'USA', created at 'CURRENT\_DATE', and updated at 'CURRENT\_DATE'. The second query inserts a new booking record with ID 3001, passenger ID 2001, booking platform 'Online', created at 'CURRENT\_DATE', updated at 'CURRENT\_DATE', status 'Confirmed', and price 350.00. The third query inserts a new booking\_flight record with ID 1001, booking ID 3001, and flight ID 101. The transaction is then committed. The output window below shows the executed SQL statements and their results, indicating multiple rows were affected.

```
1 ✓ BEGIN;
2
3 ✓ INSERT INTO passengers(passenger_id, first_name, last_name, date_of_birth, country_of_residence, created_at, update_at)
4   VALUES (2001, 'Alice', 'Smith', '1995-06-15', 'USA', CURRENT_DATE, CURRENT_DATE);
5
6 ✓ INSERT INTO booking(booking_id, passenger_id, booking_platform, created_at, update_at, status, price)
7   VALUES (3001, 2001, 'Online', CURRENT_DATE, CURRENT_DATE, 'Confirmed', 350.00);
8
9 ✓ INSERT INTO booking_flight(booking_flight_id, booking_id, flight_id)
10   VALUES (1001, 3001, 101);
11
12 ✓ COMMIT;
```

```
[2025-12-03 12:32:21] dbwork.public> INSERT INTO passengers(passenger_id, first_name, last_name, date_of_birth, country_of_residence, created_at, update_at)
VALUES (2001, 'Alice', 'Smith', '1995-06-15', 'USA', CURRENT_DATE, CURRENT_DATE)
[2025-12-03 12:32:21] 1 row effected in 1 ms
[2025-12-03 12:32:21] dbwork.public> INSERT INTO booking(booking_id, passenger_id, booking_platform, created_at, update_at, status, price)
VALUES (3001, 2001, 'Online', CURRENT_DATE, CURRENT_DATE, 'Confirmed', 350.00)
[2025-12-03 12:32:21] 1 row effected in 2 ms
[2025-12-03 12:32:21] dbwork.public> INSERT INTO booking_flight(booking_flight_id, booking_id, flight_id)
VALUES (1001, 3001, 101)
[2025-12-03 12:32:21] 1 row effected in 2 ms
[2025-12-03 12:32:21] dbwork.public> COMMIT
[2025-12-03 12:32:21] completed in 1 ms
```

6\_

The screenshot shows a PostgreSQL database interface. On the left, the Database Explorer pane displays the schema structure under the dbwork database, including tables like airline, airport, baggage, baggage\_check, boarding\_pass, booking, booking\_flight, flights, passengers, security\_check, and views. In the center, the console tab shows the following SQL code:

```
1 ✓ BEGIN;
2
3 ✓ UPDATE booking b
4   SET price = price + 50,
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 ✓ COMMIT;
```

On the right, the Services pane shows the transaction status and the Output pane displays the execution results:

```
[2025-12-03 12:34:01] completed in 2 ms
[2025-12-03 12:34:52] dbwork.public> BEGIN
[2025-12-03 12:34:52] completed in 2 ms
[2025-12-03 12:34:52] dbwork.public> UPDATE booking b
      SET price = price + 50,
          update_at = CURRENT_DATE
      FROM booking_flight bf
      WHERE b.booking_id = bf.booking_id
        AND bf.flight_id = 101
[2025-12-03 12:34:52] 1 row affected in 4 ms
[2025-12-03 12:34:52] dbwork.public> COMMIT
[2025-12-03 12:34:52] completed in 3 ms
```

7\_

The screenshot shows a PostgreSQL database interface. On the left, the Database Explorer pane displays the schema structure under the dbwork database, including tables like airline, airport, baggage, baggage\_check, boarding\_pass, booking, booking\_flight, flights, passengers, security\_check, and views. In the center, the console tab shows the following SQL code:

```
1 ✓ BEGIN;
2
3 ✓ UPDATE baggage
4   SET weight_in_kg = 23.5,
5       update_date = CURRENT_DATE
6   WHERE baggage_id = 401;
7
8 ✓ COMMIT;
```

On the right, the Services pane shows the transaction status and the Output pane displays the execution results:

```
[2025-12-03 12:34:01] 1 row affected in 4 ms
[2025-12-03 12:34:52] dbwork.public> COMMIT
[2025-12-03 12:34:52] completed in 3 ms
[2025-12-03 12:35:00] dbwork.public> BEGIN
[2025-12-03 12:35:00] completed in 2 ms
[2025-12-03 12:36:00] dbwork.public> UPDATE baggage
      SET weight_in_kg = 23.5,
          update_date = CURRENT_DATE
      WHERE baggage_id = 401
[2025-12-03 12:36:00] 1 row affected in 4 ms
[2025-12-03 12:36:00] dbwork.public> COMMIT
[2025-12-03 12:36:00] completed in 2 ms
```

8\_

The screenshot shows a PostgreSQL database interface with a dark theme. On the left, the Database Explorer sidebar lists the schema structure under the 'dbwork' database, including tables like 'airline', 'flight', 'passenger', and 'booking'. In the main area, a 'console' tab is active, displaying a transaction log:

```
1 ✓ BEGIN;
2 ✓ UPDATE booking
3   SET price = price - 50,
4       update_at = CURRENT_DATE
5   WHERE passenger_id = 201;
6 ✓ COMMIT;
```

Below the console, the Services panel shows a single database connection named 'postgres@localhost'. The Output tab of the Services panel displays the same transaction log:

```
[2025-12-03 12:36:00] 1 row affected in 4 ms
[2025-12-03 12:36:00] dbwork.public> COMMIT
[2025-12-03 12:36:00] completed in 2 ms
[2025-12-03 12:36:57] dbwork.public> BEGIN
[2025-12-03 12:36:57] completed in 1 ms
[2025-12-03 12:36:57] dbwork.public> UPDATE booking
      SET price = price - 50,
          update_at = CURRENT_DATE
     WHERE passenger_id = 201
[2025-12-03 12:36:57] 1 row affected in 2 ms
[2025-12-03 12:36:57] dbwork.public> COMMIT
[2025-12-03 12:36:57] completed in 2 ms
```

At the bottom, the status bar indicates 'Database Consoles > postgres@localhost > console' and shows performance metrics: 9:1 CRLF, UTF-8, 4 spaces.

9\_

The screenshot shows a PostgreSQL database interface with a dark theme. The Database Explorer sidebar lists the schema structure under the 'dbwork' database, including tables like 'airline', 'flight', 'passenger', and 'booking'. In the main area, a 'console' tab is active, displaying a transaction log:

```
1 ✓ BEGIN;
2 ✓ INSERT INTO flights(flight_id, flight_no, scheduled_departure, scheduled_arrival,
3   departure_airport_id, arrival_airport_id, departing_gate, arriving_gate,
4   airline_id, status, actual_departure, actual_arrival, created_at, update_at)
5   VALUES (1002, 'AA123', '2025-12-10', '2025-12-11', 1, 2, 'A1', 'B2', 1, 'Scheduled',
6   NULL, NULL, CURRENT_DATE, CURRENT_DATE);
7 ✓ UPDATE booking_flight
8   SET flight_id = 1002
9   WHERE flight_id = 101;
10 ✓ COMMIT;
```

Below the console, the Services panel shows a single database connection named 'postgres@localhost'. The Output tab of the Services panel displays the same transaction log:

```
[2025-12-03 12:39:51] 1 row affected in 5 ms
[2025-12-03 12:39:51] dbwork.public> UPDATE booking_flight
      SET flight_id = 1002
     WHERE flight_id = 101
[2025-12-03 12:39:51] 1 row affected in 2 ms
[2025-12-03 12:39:51] dbwork.public> COMMIT
[2025-12-03 12:39:51] completed in 2 ms
[2025-12-03 12:39:51] dbwork.public> ROLLBACK
[2025-12-03 12:39:51] [2SP01] нет незавершённой транзакции
[2025-12-03 12:39:51] completed in 11 ms
```

At the bottom, the status bar indicates 'Database Consoles > postgres@localhost > console' and shows performance metrics: 16:1 CRLF, UTF-8, 4 spaces.