

# LAB 9. Abdykamat Adilet. 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).

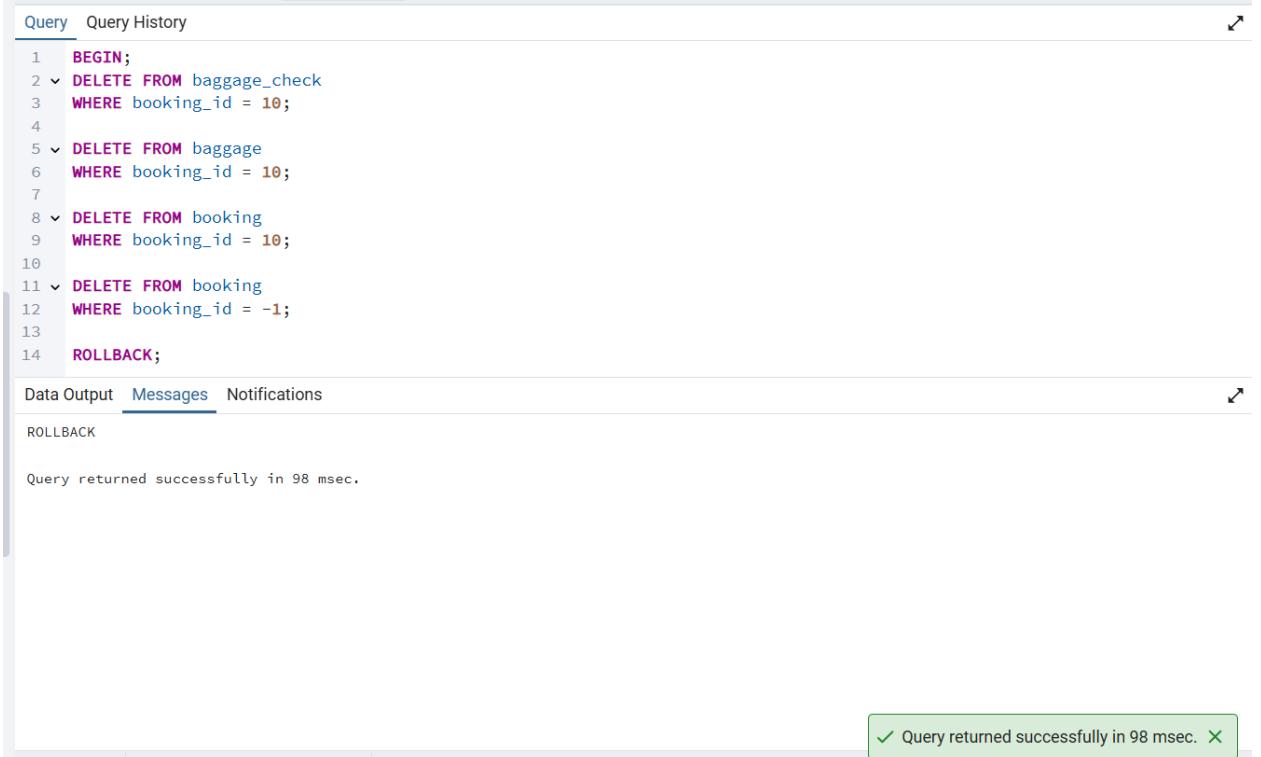
```
BEGIN;  
  
DELETE FROM baggage_check  
  
WHERE booking_id = 10;
```

```
DELETE FROM baggage  
  
WHERE booking_id = 10;
```

```
DELETE FROM booking  
  
WHERE booking_id = 10;
```

```
DELETE FROM booking  
  
WHERE booking_id = -1;
```

```
ROLLBACK;
```



The screenshot shows a database query interface with the following details:

- Query History:** A list of 14 numbered statements:
  - BEGIN;
  - DELETE FROM baggage\_check WHERE booking\_id = 10;
  - DELETE FROM baggage WHERE booking\_id = 10;
  - DELETE FROM booking WHERE booking\_id = 10;
  - DELETE FROM booking WHERE booking\_id = -1;
  - ROLLBACK;
- Data Output:** The output shows the command "ROLLBACK" and the message "Query returned successfully in 98 msec."
- Status Bar:** At the bottom right, there is a green success message: "✓ Query returned successfully in 98 msec. ✘".
- Bottom Navigation:** Total rows: 0, Query complete: 00:00:00.000, CPU: 1s 11ms, SQL: 10s.

**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).**

```
BEGIN;  
  
UPDATE flights  
  
SET sch_departure_time = '2025-01-30 14:00:00'  
  
WHERE flight_id = 5;  
  
-- Error simulation  
  
UPDATE flights  
  
SET sch_departure_time = '2025-01-30 15:00:00'  
  
WHERE flight_id = -1;
```

```
ROLLBACK;
```

The screenshot shows a database query interface with two tabs: 'Query' and 'Messages'. The 'Query' tab contains the SQL code for a transaction. The 'Messages' tab shows the execution results, including a success message and a status bar indicating the query completed successfully.

**Query**    Query History

```
1 BEGIN;  
2  
3 ✓ UPDATE flights  
4 SET sch_departure_time = '2025-01-30 14:00:00'  
5 WHERE flight_id = 5;  
6  
7 -- Error simulation  
8 ✓ UPDATE flights  
9 SET sch_departure_time = '2025-01-30 15:00:00'  
10 WHERE flight_id = -1;  
11  
12 ROLLBACK;  
13
```

**Data Output**    Messages    Notifications

ROLLBACK

Query returned successfully in 97 msec.

Total rows: 0    Query complete 00:00:00.007    CPU: 1%    In 7, Out 20

✓ Query returned successfully in 97 msec. ✘

**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.**

```
BEGIN;  
  
UPDATE tickets  
  
SET price = price - 50  
  
WHERE flight_number = 'SU1002';  
  
-- Error simulation  
  
UPDATE tickets  
  
SET price = price / 0;
```

```
ROLLBACK;
```

Query    Query History

```
1 BEGIN;  
2  
3 ✓ UPDATE tickets  
4 SET price = price - 50  
5 WHERE flight_number = 'SU1002';  
6  
7 -- Error simulation  
8 ✓ UPDATE tickets  
9 SET price = price / 0;  
10  
11 ROLLBACK;
```

Data Output    Messages    Notifications

ERROR: division by zero

SQL state: 22012

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

```
BEGIN;

UPDATE passengers

SET first_name = 'John',
    last_name = 'Williams',
    country_of_citizenship = 'USA'

WHERE passenger_id = 12;

UPDATE booking

SET passenger_id = 12

WHERE passenger_id = 12;

-- Error simulation

UPDATE passengers

SET first_name = NULL

WHERE passenger_id = -1;
```

```
ROLLBACK;
```

The screenshot shows a MySQL query editor window. The 'Query' tab is active, displaying the SQL code from the previous text block. The code includes a BEGIN block, updates to the passengers and bookings tables, an error simulation, and a ROLLBACK statement. Below the query, the 'Data Output' tab is selected, showing the output 'ROLLBACK'. At the bottom of the editor, a message box indicates 'Query returned successfully in 97 msec.' with a green checkmark icon.

```
Query  Query History
4 SET first_name = 'John',
5     last_name = 'Williams',
6     country_of_citizenship = 'USA'
7 WHERE passenger_id = 12;
8
9 ✓ UPDATE booking
10 SET passenger_id = 12
11 WHERE passenger_id = 12;
12
13 -- Error simulation
14 ✓ UPDATE passengers
15 SET first_name = NULL
16 WHERE passenger_id = -1;
17
18 ROLLBACK;

Data Output  Messages  Notifications
```

ROLLBACK

Query returned successfully in 97 msec.

Total rows: 0 Query complete: 2023-09-20 00:00:00.007 ✓

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

```
BEGIN;

INSERT INTO passengers (first_name, last_name, country_of_citizenship, date_of_birth,
passport_number,
                           gender,passenger_id, country_of_residence, created_at,
updated_at )

VALUES ('Adam', 'Stone', 'Canada', '1990-01-01', 'AB1234567', 'M', '99', 'India', NOW(), NOW())

RETURNING passenger_id;

INSERT INTO booking ( booking_id, passenger_id, booking_platform, flight_id, created_at, updated_at,
status, ticket_price, ticket_discount)

VALUES ( 1001, 5, 'Mobile App', 2003, NOW(), NOW(), 'Co', 0, 0 );

ROLLBACK;

COMMIT;
```

The screenshot shows a database query editor interface. The top tab bar has 'Query' selected, followed by 'Query History'. The main area displays a multi-line SQL script with numbered comments from 20 to 34. The script performs a transaction: it inserts a passenger record, inserts a booking record, then rolls back the transaction and commits. Below the script, the 'Data Output' tab is active, showing the output of the query. The output includes a 'WARNING: there is no transaction in progress' message, a 'COMMIT' message, and a final message 'Query returned successfully in 86 msec.' at the bottom. A green success icon with a checkmark is visible in the bottom right corner of the output area.

```
20
21
22
23
24
25
26
27
28
29
30 );
31
32 ROLLBACK;
33
34 COMMIT;
```

Data Output [Messages](#) [Notifications](#)

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 86 msec.

✓ Query returned successfully in 86 msec. ✘

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

```
BEGIN;
```

```
UPDATE tickets
```

```
SET price = price + 30
```

```
WHERE flight_number = 'SU1004';
```

```
ROLLBACK;
```

```
COMMIT;
```

The screenshot shows a database query editor interface. The top section is titled "Query" and contains the SQL code for increasing ticket prices. The bottom section is titled "Messages" and displays the results of the query execution.

```
Query  Query History
1 BEGIN;
2
3 ✓ UPDATE tickets
4   SET price = price + 30
5 WHERE flight_number = 'SU1004';
6 ROLLBACK;
7 COMMIT;
```

Data Output Messages Notifications

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 93 msec.

✓ Query returned successfully in 93 msec. ✘

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

```
BEGIN;
```

```
UPDATE baggage
```

```
SET weight_in_kg = 23.5
```

```
WHERE baggage_id = 77;
```

```
-- Error simulation
```

```
UPDATE baggage
```

```
SET weight_in_kg = -5
```

```
WHERE baggage_id = 77;
```

```
ROLLBACK;
```

```
-- COMMIT;
```

The screenshot shows a MySQL query editor interface. The top section is the 'Query' tab, containing the following SQL code:

```
1 BEGIN;
2
3 UPDATE baggage
4 SET weight_in_kg = 23.5
5 WHERE baggage_id = 77;
6
7 -- Error simulation
8 UPDATE baggage
9 SET weight_in_kg = -5
10 WHERE baggage_id = 77;
11
12 ROLLBACK;
13 -- COMMIT;
```

The bottom section is the 'Messages' tab, which displays the output of the query:

```
ROLLBACK
Query returned successfully in 97 msec.
```

A green success message box is visible on the right side of the screen:

✓ Query returned successfully in 97 msec. ✘

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

```
BEGIN;
```

```
UPDATE booking
```

```
SET ticket_price = ticket_price * 0.85
```

```
WHERE booking_id = 300;
```

```
-- Error simulation
```

```
UPDATE booking
```

```
SET ticket_price = ticket_price / 0;
```

```
ROLLBACK;
```

```
-- COMMIT;
```

The screenshot shows a database query editor interface with two tabs: 'Query' and 'Query History'. The 'Query' tab contains the transaction code. The 'Messages' tab displays an error message: 'ERROR: division by zero' and 'SQL state: 22012'.

Line Number	Code
1	<code>BEGIN;</code>
2	
3	<code>UPDATE booking</code>
4	<code>SET ticket_price = ticket_price * 0.85</code>
5	<code>WHERE booking_id = 300;</code>
6	
7	<code>-- Error simulation</code>
8	<code>UPDATE booking</code>
9	<code>SET ticket_price = ticket_price / 0;</code>
10	
11	<code>ROLLBACK;</code>
12	<code>-- COMMIT;</code>

**Query**   **Query History**

1 `BEGIN;`  
2  
3 `UPDATE booking`  
4 `SET ticket_price = ticket_price * 0.85`  
5 `WHERE booking_id = 300;`  
6  
7 `-- Error simulation`  
8 `UPDATE booking`  
9 `SET ticket_price = ticket_price / 0;`  
10  
11 `ROLLBACK;`  
12 `-- COMMIT;`

**Data Output**   **Messages**   **Notifications**

ERROR: division by zero  
SQL state: 22012

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

```
BEGIN;
```

```
UPDATE tickets
```

```
SET flight_number = 'SU1001'
```

```
WHERE flight_number = 'SU1005';
```

```
-- Error simulation
```

```
UPDATE flights
```

```
SET sch_departure_time = NULL
```

```
WHERE flight_id = -1;
```

```
ROLLBACK;
```

```
-- COMMIT;
```

The screenshot shows a database query editor interface with two tabs: 'Query' and 'Messages'. The 'Query' tab contains the following SQL code:

```
1 BEGIN;
2
3 UPDATE tickets
4 SET flight_number = 'SU1001'
5 WHERE flight_number = 'SU1005';
6
7 -- Error simulation
8 UPDATE flights
9 SET sch_departure_time = NULL
10 WHERE flight_id = -1;
11
12 ROLLBACK;
13 -- COMMIT;
```

The 'Messages' tab displays the following output:

```
ROLLBACK
```

Query returned successfully in 125 msec.

A green success message box at the bottom right of the screen states: ✓ Query returned successfully in 125 msec. ✘