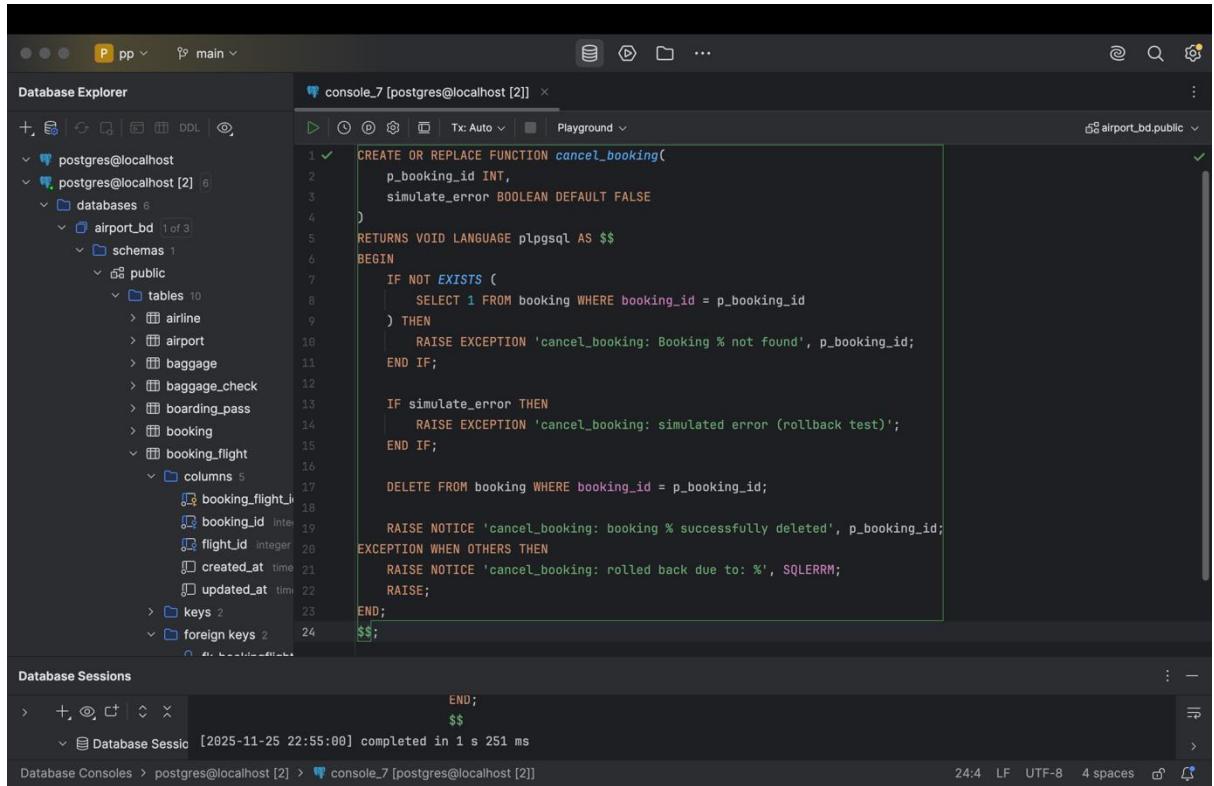


Laboratory work 9

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 a PostgreSQL database interface with the following details:

- Database Explorer:** Shows the database structure:
 - Postgres@localhost
 - Postgres@localhost [2]
 - Databases: 6 (including airport_bd)
 - Schemas: 1 (public)
 - Tables: 10 (airline, airport, baggage, baggage_check, boarding_pass, booking, booking_flight, columns, keys, foreign keys)
 - Columns: 5 (booking_flight_id, booking_id, flight_id, created_at, updated_at)
- Playground:** A code editor containing the following PL/pgSQL function definition:

```
CREATE OR REPLACE FUNCTION cancel_booking(
    p_booking_id INT,
    simulate_error BOOLEAN DEFAULT FALSE
)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (
        SELECT 1 FROM booking WHERE booking_id = p_booking_id
    ) THEN
        RAISE EXCEPTION 'cancel_booking: Booking % not found', p_booking_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'cancel_booking: simulated error (rollback test)';
    END IF;

    DELETE FROM booking WHERE booking_id = p_booking_id;

    RAISE NOTICE 'cancel_booking: booking % successfully deleted', p_booking_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'cancel_booking: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;
```
- Database Sessions:** Shows a single session completed in 1 s 251 ms.
- Console:** Shows the command `psql -c "CREATE OR REPLACE FUNCTION cancel_booking(...)"` was run.

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 a PostgreSQL database interface with the following details:

- Database Explorer:** Shows the database structure. A function named `reschedule_flight` is selected in the `Playground` tab.
- Code Editor (Playground):**

```

CREATE OR REPLACE FUNCTION reschedule_flight(
    p_flight_id INT, p_new_departure TIMESTAMP, simulate_error BOOLEAN DEFAULT FALSE)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_flight_id) THEN
        RAISE EXCEPTION 'reschedule_flight: Flight % not found', p_flight_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'reschedule_flight: simulated error (rollback test)';
    END IF;

    UPDATE flights
    SET sch_departure_time = p_new_departure,
        updated_at = NOW()
    WHERE flight_id = p_flight_id;

    RAISE NOTICE 'reschedule_flight: flight % sch_departure_time set to %', p_flight_id, p_new_departure;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'reschedule_flight: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;
```
- Database Sessions:** Shows a transaction history for the `reschedule_flight` function.
- Bottom Status Bar:** Shows the current session details: `23:4 LF UTF-8 4 spaces`.

- 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 a PostgreSQL database interface with the following details:

- Database Explorer:** Shows the database structure. A function named `decrease_ticket_prices` is selected in the `Playground` tab.
- Code Editor (Playground):**

```

CREATE OR REPLACE FUNCTION decrease_ticket_prices(
    p_flight_id INT, p_amount DECIMAL(7,2), simulate_error BOOLEAN DEFAULT FALSE)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_flight_id) THEN
        RAISE EXCEPTION 'decrease_ticket_prices: Flight % not found', p_flight_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'decrease_ticket_prices: simulated error (rollback test)';
    END IF;

    UPDATE booking
    SET ticket_price = GREATEST(ticket_price - p_amount, 0),
        updated_at = NOW()
    WHERE flight_id = p_flight_id;

    RAISE NOTICE 'decrease_ticket_prices: prices decreased by % for flight %', p_amount, p_flight_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'decrease_ticket_prices: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;
```
- Database Sessions:** Shows a transaction history for the `decrease_ticket_prices` function.
- Bottom Status Bar:** Shows the current session details: `2:5 LF UTF-8 4 spaces`.

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

```

CREATE OR REPLACE FUNCTION update_passenger_details(
    p_passenger_id INT,
    p_first_name VARCHAR DEFAULT NULL,
    p_last_name VARCHAR DEFAULT NULL,
    p_passport_number VARCHAR DEFAULT NULL,
    simulate_error BOOLEAN DEFAULT FALSE
)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (SELECT 1 FROM passengers WHERE passenger_id = p_passenger_id) THEN
        RAISE EXCEPTION 'update_passenger_details: Passenger % not found', p_passenger_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'update_passenger_details: simulated error (rollback test)';
    END IF;

    UPDATE passengers
    SET first_name = COALESCE(p_first_name, first_name),
        last_name = COALESCE(p_last_name, last_name),
        passport_number = COALESCE(p_passport_number, passport_number),
        updated_at = NOW()
    WHERE passenger_id = p_passenger_id;

    RAISE NOTICE 'update_passenger_details: passenger % updated', p_passenger_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'update_passenger_details: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;

```



```

CREATE OR REPLACE FUNCTION update_passenger_details(
    p_passenger_id INT,
)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (SELECT 1 FROM passengers WHERE passenger_id = p_passenger_id) THEN
        RAISE EXCEPTION 'update_passenger_details: Passenger % not found', p_passenger_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'update_passenger_details: simulated error (rollback test)';
    END IF;

    UPDATE passengers
    SET first_name = COALESCE(p_first_name, first_name),
        last_name = COALESCE(p_last_name, last_name),
        passport_number = COALESCE(p_passport_number, passport_number),
        updated_at = NOW()
    WHERE passenger_id = p_passenger_id;

    RAISE NOTICE 'update_passenger_details: passenger % updated', p_passenger_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'update_passenger_details: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;

[2025-11-25 23:15:06] completed in 7 ms

```

The screenshot shows a PostgreSQL database console interface. In the top right, there are tabs for 'main' and 'console_7 [postgres@localhost [2]]'. The main area displays a 'Database Explorer' tree with a node 'airline' expanded, showing sub-nodes like 'airline', 'airport', 'baggage', etc. Below it is a 'Database Sessions' tree showing a single session 'postgres@localhost'. The central workspace is a 'Playground' tab where the following SQL code is run:

```
1 SELECT update_passenger_details( p_passenger_id 55, p_first_name 'Aizhan', p_last_name 'Dusembay', p_passport_n ✓ 2 ~ v
2 SELECT update_passenger_details( p_passenger_id 55, p_first_name NULL, p_last_name 'NewLast', p_passport_number NULL, sim
3
4 ✓ 4 SELECT * FROM passengers WHERE passenger_id = 55;
```

The output pane shows the result of the final query:

	passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship
1	55	Aizhan	Dusembay	1987-07-06	Male	USA

At the bottom, the status bar indicates 'Database Consoles > postgres@localhost [2] > console_7 [postgres@localhost [2]]' and the time '4:50'.

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

The screenshot shows a PostgreSQL database console interface. In the top right, there are tabs for 'main' and 'console_7 [postgres@localhost [2]]'. The main area displays a 'Database Explorer' tree with a node 'airline' expanded, showing sub-nodes like 'airline', 'airport', 'baggage', etc. Below it is a 'Database Sessions' tree showing a single session 'postgres@localhost'. The central workspace is a 'Playground' tab where the following SQL code is run:

```
1 SELECT register_passenger_and_booking(
2     p_passenger_id 3002, p_first_name 'Duman', p_last_name 'Sultan', p_date_of_birth '2001-04-12', p_gender 'M', p_citizenship 'KZ', p_residence 'KZ', p_passport_n ✓ 2 ~ v
3     p_booking_id 5002, p_flight_id 30, p_platform 'WEB', p_ticket_price 560.00, simulate_error FALSE
4 );
5
6 ✓ 6 SELECT * FROM passengers WHERE passenger_id=3002;
```

The output pane shows the result of the final query:

	passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship
1	3002	Duman	Sultan	2001-04-12	M	KZ

At the bottom, the status bar indicates 'Database Consoles > postgres@localhost [2] > console_7 [postgres@localhost [2]]' and the time '6:50'.

```

CREATE OR REPLACE FUNCTION register_passenger_and_booking(
    p_passenger_id INT,
    p_first_name VARCHAR,
    p_last_name VARCHAR,
    p_date_of_birth DATE,
    p_gender VARCHAR,
    p_citizenship VARCHAR,
    p_residence VARCHAR,
    p_passport VARCHAR,
    p_booking_id INT,
    p_flight_id INT,
    p_platform VARCHAR,
    p_ticket_price DECIMAL(7,2),
    simulate_error BOOLEAN DEFAULT FALSE
)
RETURNS VOID LANGUAGE plpgsql AS $$$
BEGIN
    IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_flight_id) THEN
        RAISE EXCEPTION 'register_passenger_and_booking: Flight % not found', p_flight_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'register_passenger_and_booking: simulated error (rollback test)';
    END IF;

    INSERT INTO passengers(
        passenger_id, first_name, last_name, date_of_birth, gender,
        country_of_citizenship, country_of_residence, passport_number, created_at, updated_at
    )
    VALUES (
        passenger_id p_passenger_id, first_name p_first_name, last_name p_last_name, date_of_birth p_date_of_birth, gender p_gender,
        country_of_citizenship p_citizenship, country_of_residence p_residence, passport_number p_passport, created_at NOW(), updated_at NOW()
    );

    INSERT INTO booking(
        booking_id, flight_id, passenger_id, booking_platform, created_at, updated_at, status, ticket_price
    )
    VALUES (
        booking_id p_booking_id, flight_id p_flight_id, passenger_id p_passenger_id, booking_platform p_platform, created_at NOW(), updated_at NOW(), status 'ACTIVE'
    );

    RAISE NOTICE 'register_passenger_and_booking: passenger % and booking % created', p_passenger_id, p_booking_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'register_passenger_and_booking: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;

```

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

```

1 ✓ CREATE OR REPLACE FUNCTION increase_ticket_prices(p_flight_id INT, p_amount DECIMAL(7,2))
2 RETURNS VOID LANGUAGE plpgsql AS $$*
3 BEGIN
4     IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_flight_id) THEN
5         RAISE EXCEPTION 'increase_ticket_prices: Flight % not found', p_flight_id;
6     END IF;
7
8     UPDATE booking
9         SET ticket_price = ticket_price + p_amount,
10            updated_at = NOW()
11        WHERE flight_id = p_flight_id;
12
13     RAISE NOTICE 'increase_ticket_prices: prices increased by % for flight %', p_amount, p_flight_id;
14 EXCEPTION WHEN OTHERS THEN
15     RAISE NOTICE 'increase_ticket_prices: rolled back due to: %', SQLERRM;
16     RAISE;
17 END;
18 $$

```

Database Sessions

- Tx +, ⌂, ⌂+ | ⌂ X Output airport_bd.public.passengers
- Database Session
 - Database
 - postgres@localhost [2] **console_7**
 - console

IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_flight_id) THEN
RAISE EXCEPTION 'increase_ticket_prices: Flight % not found', p_flight_id;
END IF;

UPDATE booking
SET ticket_price = ticket_price + p_amount,
updated_at = NOW()

Database Consoles > postgres@localhost [2] > **console_7** [postgres@localhost [2]]

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

```

1 ✓ CREATE OR REPLACE FUNCTION update_baggage_weight(p_baggage_id INT, p_new_weight DECIMAL(4,2), simulate_error BOOLEAN DEFAULT FALSE)
2 RETURNS VOID LANGUAGE plpgsql AS $$*
3 BEGIN
4     IF NOT EXISTS (SELECT 1 FROM baggage WHERE baggage_id = p_baggage_id) THEN
5         RAISE EXCEPTION 'update_baggage_weight: Baggage % not found', p_baggage_id;
6     END IF;
7
8     IF simulate_error THEN
9         RAISE EXCEPTION 'update_baggage_weight: simulated error (rollback test)';
10    END IF;
11
12     UPDATE baggage
13         SET weight_in_kg = p_new_weight,
14            updated_at = NOW()
15        WHERE baggage_id = p_baggage_id;
16
17     RAISE NOTICE 'update_baggage_weight: baggage % weight set to % kg', p_baggage_id, p_new_weight;
18 EXCEPTION WHEN OTHERS THEN
19     RAISE NOTICE 'update_baggage_weight: rolled back due to: %', SQLERRM;
20     RAISE;
21 END;
22 $$

```

Database Sessions

- Tx +, ⌂, ⌂+ | ⌂ X Output airport_bd.public.passengers
- Database Session
 - Database
 - postgres@localhost [2] **console_7**
 - console

EXCEPTION WHEN OTHERS THEN
RAISE NOTICE 'update_baggage_weight: rolled back due to: %', SQLERRM;
RAISE;
END;

Database Consoles > postgres@localhost [2] > **console_7** [postgres@localhost [2]]

```

1 SELECT update_baggage_weight( p_baggage_id 7, p_new_weight 19, simulate_error FALSE);
2 ✓ SELECT * FROM baggage WHERE baggage_id = 7;
3
4

```

Database Sessions

	baggage_id	weight_in_kg	created_at	updated_at	booking_id
1	7	19.00	2025-09-30 14:14:46.168315	2025-11-25 18:33:07.951604	

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

```

1 ✓ CREATE OR REPLACE FUNCTION apply_discount_to_booking(p_booking_id INT, p_passenger_id INT, p_discount DECIMAL(7,2), simulate_error BOOLEAN DEFAULT FALSE)
2 RETURNS VOID LANGUAGE plpgsql AS $$*
3 BEGIN
4     IF NOT EXISTS (SELECT 1 FROM booking WHERE booking_id = p_booking_id AND passenger_id = p_passenger_id) THEN
5         RAISE EXCEPTION 'apply_discount_to_booking: Booking % for passenger % not found', p_booking_id, p_passenger_id;
6     END IF;
7
8     IF simulate_error THEN
9         RAISE EXCEPTION 'apply_discount_to_booking: simulated error (rollback test)';
10    END IF;
11
12    UPDATE booking
13        SET ticket_price = GREATEST(ticket_price - p_discount, 0),
14            updated_at = NOW()
15        WHERE booking_id = p_booking_id AND passenger_id = p_passenger_id;
16
17        RAISE NOTICE 'apply_discount_to_booking: discount % applied to booking %', p_discount, p_booking_id;
18    EXCEPTION WHEN OTHERS THEN
19        RAISE NOTICE 'apply_discount_to_booking: rolled back due to: %', SQLERRM;
20        RAISE;
21    END;
22$$;

```

Database Sessions

	baggage_id	weight_in_kg	created_at	updated_at	booking_id
1	7	19.00	2025-09-30 14:14:46.168315	2025-11-25 18:33:07.951604	

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

The screenshot shows a PostgreSQL IDE interface with two code editors side-by-side.

Left Pane:

```
CREATE OR REPLACE FUNCTION reschedule_all_bookings(p_old_flight_id INT, p_new_flight_id INT, simulate_error BOOLEAN DEFAULT FALSE)
RETURNS VOID LANGUAGE plpgsql AS $$

BEGIN
    IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_old_flight_id) THEN
        RAISE EXCEPTION 'reschedule_all_bookings: Old flight % not found', p_old_flight_id;
    END IF;
    IF NOT EXISTS (SELECT 1 FROM flights WHERE flight_id = p_new_flight_id) THEN
        RAISE EXCEPTION 'reschedule_all_bookings: New flight % not found', p_new_flight_id;
    END IF;

    IF simulate_error THEN
        RAISE EXCEPTION 'reschedule_all_bookings: simulated error (rollback test)';
    END IF;

    UPDATE booking
    SET flight_id = p_new_flight_id,
        updated_at = NOW()
    WHERE flight_id = p_old_flight_id;

    UPDATE booking_flight
    SET flight_id = p_new_flight_id,
        updated_at = NOW()
    WHERE flight_id = p_old_flight_id;

    RAISE NOTICE 'reschedule_all_bookings: moved bookings from % to %', p_old_flight_id, p_new_flight_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'reschedule_all_bookings: rolled back due to: %', SQLERRM;
$$
```

Right Pane:

```
CREATE OR REPLACE FUNCTION reschedule_all_bookings(p_old_flight_id INT, p_new_flight_id INT, simulate_error BOOLEAN DEFAULT FALSE)
RETURNS VOID LANGUAGE plpgsql AS $$

UPDATE booking
SET flight_id = p_new_flight_id,
    updated_at = NOW()
WHERE flight_id = p_old_flight_id;

UPDATE booking_flight
SET flight_id = p_new_flight_id,
    updated_at = NOW()
WHERE flight_id = p_old_flight_id;

RAISE NOTICE 'reschedule_all_bookings: moved bookings from % to %', p_old_flight_id, p_new_flight_id;
EXCEPTION WHEN OTHERS THEN
    RAISE NOTICE 'reschedule_all_bookings: rolled back due to: %', SQLERRM;
    RAISE;
END;
$$;
```

Database Sessions pane at the bottom shows the transaction status and execution details:

- TX: apply_discount_to_booking(10.00, FALSE):void
- airport_bd.public.booking
- EXCEPTION WHEN OTHERS THEN
- RAISE NOTICE 'reschedule_all_bookings: rolled back due to: %', SQLERRM;
- RAISE;
- END;
- \$\$
- [2025-11-25 23:36:12] completed in 5 ms