

Lab 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 database console with the following SQL script:

```
begin;
delete from booking
where booking_id = 25;
select * from booking where booking_id = 25;
```

The output shows 0 rows, indicating the booking was successfully deleted.

booking_id	flight_id	passenger_id	booking_platform	created_at	updated_at	status	ticket_price
------------	-----------	--------------	------------------	------------	------------	--------	--------------

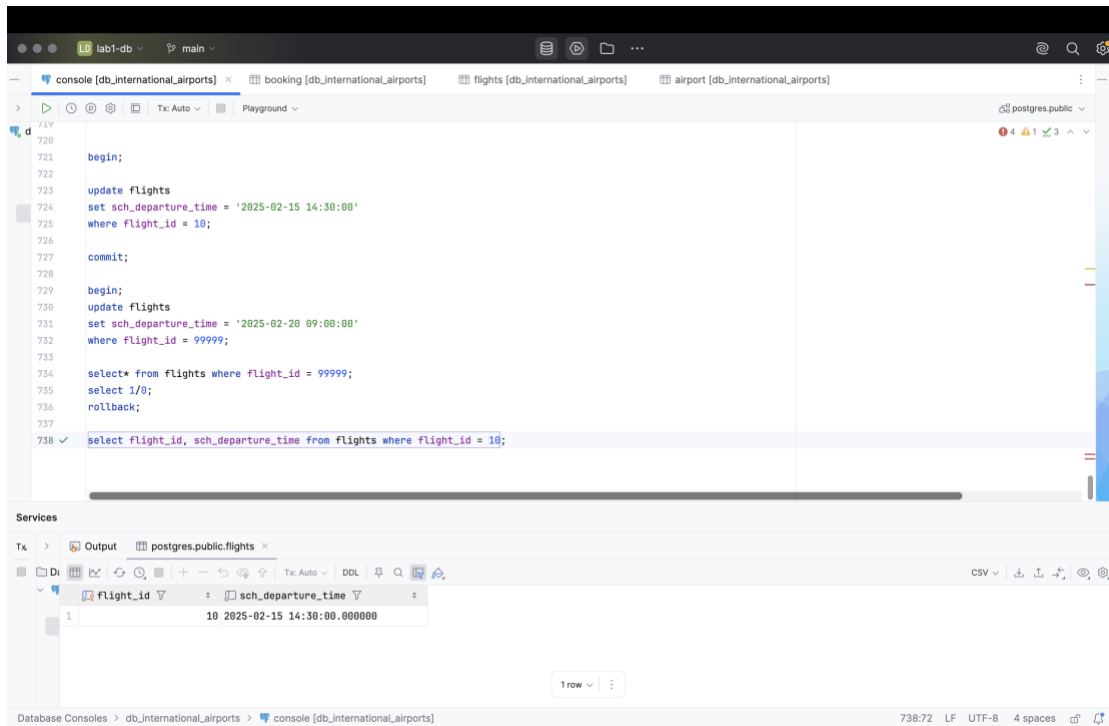
The screenshot shows a database console with the following SQL script:

```
select * from booking where booking_id = 25;
begin;
delete from booking
where booking_id = 99999;
select * from booking where booking_id = 99999;
select 1/0;
rollback;
select * from booking limit 25;
```

The output shows 25 rows, indicating the booking was successfully deleted and the transaction was rolled back.

booking_id	flight_id	passenger_id	booking_platform	created_at	updated_at	status	ticket
18	18	7	169 Rank	2025-09-24 06:10:42.755116	2025-09-24 06:10:42.755116	checked in	
19	19	81	86 Rank	2025-09-24 06:10:42.766519	2025-09-24 06:10:42.766519	confirmed	
20	20	73	191 Tresom	2025-09-24 06:10:42.775914	2025-09-24 06:10:42.775914	confirmed	
21	21	158	25 Andax	2025-09-24 06:10:42.784734	2025-09-24 06:10:42.784734	checked in	
22	22	93	154 Prodder	2025-09-24 06:10:42.792809	2025-09-24 06:10:42.792809	boarded	
23	23	185	17 Alphazap	2025-09-24 06:10:42.803192	2025-09-24 06:10:42.803192	cancelled	
24	24	80	183 Trippledex	2025-09-24 06:10:42.813582	2025-09-24 06:10:42.813582	cancelled	
25	25	190	55 Mat Lam Tam	2025-09-24 06:10:42.824373	2025-09-24 06:10:42.824373	no show	

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 = '2025-02-15 14:30:00'
where flight_id = 10;
commit;

begin;
update flights
set sch_departure_time = '2025-02-20 09:00:00'
where flight_id = 99999;

select* from flights where flight_id = 99999;
select 1/0;
rollback;

select flight_id, sch_departure_time from flights where flight_id = 10;
```

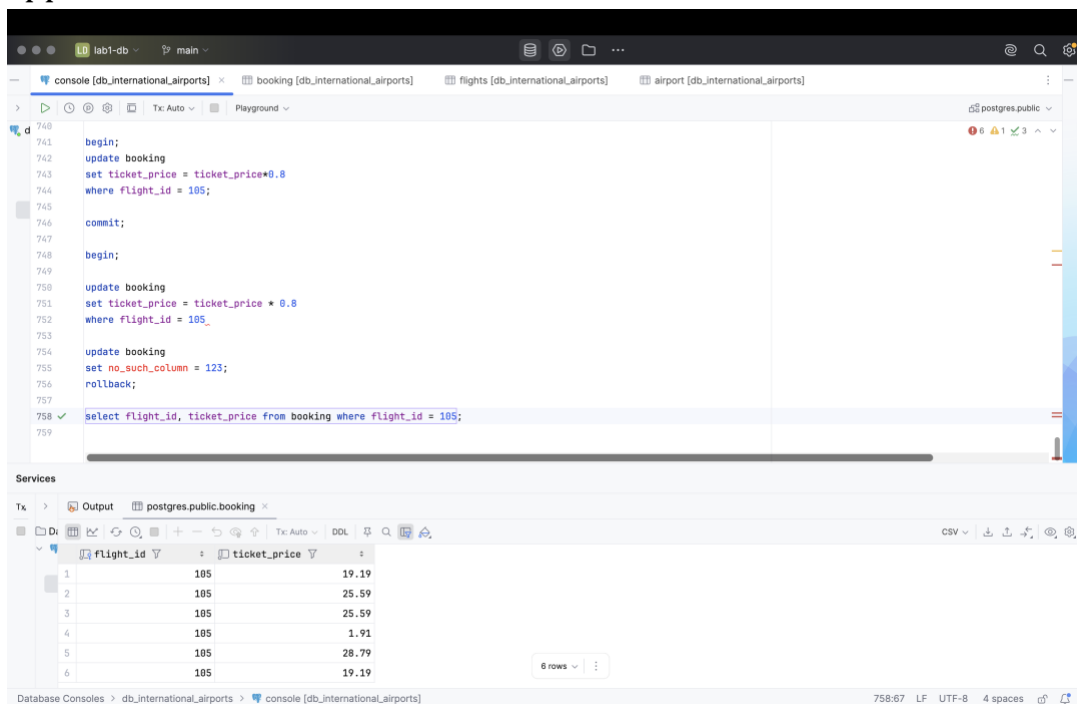
Services

Output postgres.public.flights

flight_id	sch_departure_time
10	2025-02-15 14:30:00.000000

1 row

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 booking
set ticket_price = ticket_price*0.8
where flight_id = 105;
commit;

begin;
update booking
set ticket_price = ticket_price * 0.8
where flight_id = 105;

update booking
set no_such_column = 123;
rollback;

select flight_id, ticket_price from booking where flight_id = 105;
```

Services

Output postgres.public.booking

flight_id	ticket_price
1	19.19
2	25.59
3	25.59
4	1.91
5	28.79
6	19.19

6 rows

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

The screenshot shows a database console interface with a sidebar on the left displaying the database schema. The main area contains a SQL query to update a passenger's details. Below the query, the 'Services' section shows the output of the query, which is a single row representing the updated passenger.

```
begin;
update passengers
set first_name = 'New',
    last_name = 'Name',
    passport_number = '985-974-98-23'
where passenger_id = 77;
commit;
```

```
begin;
update passengers
set passport_number = 'Invalid number'
where passenger_id = 77;
rollback;
```

```
select* from passengers where passenger_id = 77;
```

passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship	country_of_residence	passport_number
77	New	Name	2025-06-25	Female	Brazil	Philippines	985-9

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 database console interface with a sidebar on the left displaying the database schema. The main area contains SQL queries to insert a new passenger and a booking, followed by a select query to verify the booking. Below the queries, the 'Services' section shows the output of the select query, which is a single row representing the booking.

```
begin;
insert into passengers(passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number)
values ( passenger_id 498, first_name 'Aruzhan', last_name 'Sadykova', date_of_birth '2005-08-25', gender 'Female', country_of_citizenship 'Kazakhstan', country_of_residence 'Kazakhstan', passport_number '985-974-98-23' );
commit;
```

```
begin;
insert into passengers(passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number)
values ( passenger_id 499, first_name 'Test', last_name 'Error', date_of_birth '2008-08-03', gender 'Male', country_of_citizenship 'Russia', country_of_residence 'Russia', passport_number '985-974-98-23' );
insert into booking (booking_id, flight_id, passenger_id, booking_platform, status, ticket_price)
values( booking_id 403, flight_id 398, passenger_id 499, booking_platform 'Aviata', status 'Boarded', ticket_price 75);
rollback;
```

```
select * from booking where passenger_id = 498;
```

booking_id	flight_id	passenger_id	booking_platform	created_at	updated_at	status	ticket_price
402	10	498	Aviata	2025-11-25 23:32:58.792099	2025-11-25 23:32:58.792099	BOARDED	

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

The screenshot shows a database console interface with a SQL query editor and a results table. The query is as follows:

```
begin;
update booking
set ticket_price = ticket_price + 5000
where flight_id = 100;
commit;
select * from booking where flight_id = 100;
```

The results table displays the following data:

passenger_id	booking_platform	created_at	updated_at	status	ticket_price	ticket_discount
64	Greenlam	2025-09-24 06:11:07.906740	2025-09-24 06:11:07.906740	boarded	5059.99	0.05
126	Biodex	2025-09-24 06:10:44.084330	2025-09-24 06:10:44.084330	pending	5024.99	0.05
11	Otcom	2025-09-24 06:10:44.359206	2025-09-24 06:10:44.359206	pending	5003.79	0.05

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 a database console interface with a SQL query editor and a results table. The query is as follows:

```
begin;
update baggage
set weight_in_kg = 32.5,
    updated_at = NOW()
where baggage_id = 56;
select * from baggage where baggage_id = 56;
commit;
```

The results table displays the following data:

baggage_id	weight_in_kg	created_at	updated_at	booking_id
56	32.50	2025-09-24 06:12:01.162792	2025-11-25 23:57:28.408636	101

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

The screenshot shows a database console interface with a SQL query editor and a results table. The query editor contains the following SQL code:

```
823
824 update booking
825 set ticket_discount = 0.1,
826    updated_at = NOW()
827 where passenger_id = 77;
828
829 rollback;
830
831 select * from booking where passenger_id = 77;
```

The results table shows two rows of data:

ht_id	passenger_id	booking_platform	created_at	updated_at	status	ticket_price	ticket_discount
1	188	77 Daltfresh	2025-09-24 06:11:00.086526	2025-09-24 06:11:00.086526	cancelled	29.99	
2	256	77 Solarbreeze	2025-09-30 23:51:36.690537	2025-09-30 23:51:36.690537	BOARDED	18.99	

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

The screenshot shows a database console interface with a SQL query editor and a results table. The query editor contains the following SQL code:

```
832
833 begin;
834
835 update booking
836 set flight_id = 101,
837    updated_at = now()
838 where flight_id = 103;
839
840 select * from booking where flight_id = 101;
```

The results table shows one row of data:

booking_id	flight_id	passenger_id	booking_platform	created_at	updated_at	status	ticket_price
1	354	101	368 Wrapsafe	2025-09-30 23:51:36.763537	2025-11-26 00:06:32.659129	PENDING	