

# DDL

1,2,4

The screenshot displays a database management interface with a left-hand sidebar showing a tree view of database objects. The main area is divided into two panels, each showing a SQL query for creating a table. The bottom panel also includes a 'Data Output' section showing the result of an 'ALTER TABLE' command.

**Left Sidebar (Database Structure):**

- Procedures
- Sequences
- Tables (10)
  - airline
  - airport
  - baggage
  - baggage\_check
  - boarding\_pass
  - booking
  - booking\_flight
  - flights
  - passengers
  - security\_check
- Trigger Functions
- Types
- Views
- Subscriptions
- postgres
- suppliers
- Login/Group Roles
- Tablespaces
- Servers (1)
  - PostgreSQL 17
    - Databases (5)
      - lab10
      - lab11
      - my\_airport

**Top Panel (Query 1):**

```
1 CREATE TABLE airline_info (  
2     airline_id INT PRIMARY KEY,  
3     airline_code VARCHAR(30) NOT NULL,  
4     airline_name VARCHAR(50) NOT NULL,  
5     airline_country VARCHAR(50) NOT NULL,  
6     created_at TIMESTAMP NOT NULL,  
7     updated_at TIMESTAMP NOT NULL,  
8     info VARCHAR(50) NOT NULL  
9 );  
10  
11 CREATE TABLE airport (  
12     airport_id INT PRIMARY KEY,  
13     airport_name VARCHAR(50) NOT NULL,  
14     country VARCHAR(50) NOT NULL,  
15     state VARCHAR(50) NOT NULL,  
16     city VARCHAR(50) NOT NULL,  
17     created_at TIMESTAMP NOT NULL,  
18     updated_at TIMESTAMP NOT NULL  
19 );  
20  
21 CREATE TABLE baggage_check (  
22     baggage_check_id INT PRIMARY KEY,  
23     check_result VARCHAR(50) NOT NULL,  
24     created_at TIMESTAMP NOT NULL,  
25     updated_at TIMESTAMP NOT NULL,  
26     booking_id INT NOT NULL,  
27     passenger_id INT NOT NULL  
28 );  
29  
30 CREATE TABLE baggage (  
31     baggage_id INT PRIMARY KEY,  
32     weight_in_kg DECIMAL(4,2) NOT NULL,  
33     created_at TIMESTAMP NOT NULL,  
34     updated_at TIMESTAMP NOT NULL,  
35     booking_id INT NOT NULL  
36 );  
37  
38 CREATE TABLE boarding_pass (  
39     boarding_pass_id INT PRIMARY KEY,  
40     booking_id INT NOT NULL,  
41     seat VARCHAR(50) NOT NULL,  
42     boarding_time TIMESTAMP NOT NULL,  
43     created_at TIMESTAMP NOT NULL,  
44     updated_at TIMESTAMP NOT NULL  
45 );
```

**Bottom Panel (Query 2):**

```
45 CREATE TABLE booking_flight (  
46     booking_flight_id INT PRIMARY KEY,  
47     booking_id INT NOT NULL,  
48     flight_id INT NOT NULL,  
49     created_at TIMESTAMP NOT NULL,  
50     updated_at TIMESTAMP NOT NULL  
51 );  
52  
53 CREATE TABLE booking (  
54     booking_id INT PRIMARY KEY,  
55     flight_id INT NOT NULL,  
56     passenger_id INT NOT NULL,  
57     booking_platform VARCHAR(50) NOT NULL,  
58     created_at TIMESTAMP NOT NULL,  
59     updated_at TIMESTAMP NOT NULL,  
60     status VARCHAR(50) NOT NULL,  
61     price DECIMAL(7,2) NOT NULL  
62 );  
63  
64 CREATE TABLE flights (  
65     flight_id INT PRIMARY KEY,  
66     sch_departure_time TIMESTAMP NOT NULL,  
67     sch_arrival_time TIMESTAMP NOT NULL,  
68     departing_airport_id INT NOT NULL,  
69     arriving_airport_id INT NOT NULL  
70 );
```

**Data Output:**

ALTER TABLE

Query returned successfully in 51 msec.

The screenshot shows a database management tool interface. On the left is a tree view of the database structure, including tables like 'airline', 'airport', 'baggage', 'boarding\_pass', 'booking', 'booking\_flight', 'flights', 'passengers', 'security\_check', 'Trigger Functions', 'Types', 'Views', 'Subscriptions', 'postgres', 'suppliers', 'Login/Group Roles', 'Tablespaces', 'Servers (1)', 'PostgreSQL 17', 'Databases (5)', and 'lab10'. The main area displays SQL queries and their results.

**Query 1:**

```

64 CREATE TABLE flights (
65     flight_id INT PRIMARY KEY,
66     sch_departure_time TIMESTAMP NOT NULL,
67     sch_arrival_time TIMESTAMP NOT NULL,
68     departing_airport_id INT NOT NULL,
69     arriving_airport_id INT NOT NULL,
70     departing_gate VARCHAR(50) NOT NULL,
71     arriving_gate VARCHAR(50) NOT NULL,
72     airline_id INT NOT NULL,
73     act_departure_time TIMESTAMP NOT NULL,
74     act_arrival_time TIMESTAMP NOT NULL,
75     created_at TIMESTAMP NOT NULL,
76     updated_at TIMESTAMP NOT NULL
77 );
78
79 CREATE TABLE passengers (
80     passenger_id INT PRIMARY KEY,
81     first_name VARCHAR(50) NOT NULL,
82     last_name VARCHAR(50) NOT NULL,
83     date_of_birth DATE NOT NULL,
84     gender VARCHAR(50) NOT NULL,
85     country_of_citizenship VARCHAR(50) NOT NULL,
86     country_of_residence VARCHAR(50) NOT NULL,
87     passport_number VARCHAR(20) NOT NULL,
88     created_at TIMESTAMP NOT NULL,
89     updated_at TIMESTAMP NOT NULL
90 );
91
92 CREATE TABLE security_check (
93     security_check_id INT PRIMARY KEY,
94     check_result VARCHAR(20) NOT NULL,
95     created_at TIMESTAMP NOT NULL,
96     updated_at TIMESTAMP NOT NULL,
97     passenger_id INT NOT NULL
98 );
99
100 ALTER TABLE airline_info RENAME TO airline;

```

**Data Output:**

ALTER TABLE

Query returned successfully in 51 msec.

5.

The screenshot shows a database management tool interface. The main area displays SQL queries and their results.

**Query 1:**

```

101 --2 task
102 ALTER TABLE airline_info RENAME TO airline;
103
104 SELECT * FROM airline;

```

**Data Output:**

airline_id	airline_code	airline_name	airline_country	created_at	updated_at	info
[PK] integer	character varying (30)	character varying (50)	character varying (50)	timestamp without time zone	timestamp without time zone	character varying (50)

6.

**ALTER TABLE booking RENAME COLUMN price TO ticket\_price;**

Data Output	Messages	Notifications
SQL		
booking_id [PK] integer	flight_id integer	passenger_id integer
booking_platform character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone
status character varying (50)	ticket_price numeric (7,2)	

7.

```
ALTER TABLE flights ALTER COLUMN departing_gate TYPE text;
```

Data Output	Messages	Notifications
SQL		
departure_time timestamp without time zone	sch_arrival_time timestamp without time zone	departing_airport_id integer
arriving_airport_id integer	departing_gate text	arriving_gate character varying (50)
airline_id integer	act_departure_time timestamp without time zone	

8.

> Constraints	105	
> Indexes	106	ALTER TABLE airline DROP COLUMN info;
> RLS Policies	107	
> Rules	108	
Data Output	Messages	Notifications
SQL		
airline_id [PK] integer	airline_code character varying (30)	airline_name character varying (50)
airline_country character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone

✓ Successfully run. Total query runtime: 71 msec. 0 rows affected. ✕

9.

```

108
109 ALTER TABLE security_check
110 ADD CONSTRAINT fk_security_passenger FOREIGN KEY (passenger_id)
111 REFERENCES passengers (passenger_id);
112
113
114 ALTER TABLE booking
115 ADD CONSTRAINT fk_booking_passenger FOREIGN KEY (passenger_id)
116 REFERENCES passengers (passenger_id);
117
118
119 ALTER TABLE baggage_check
120 ADD CONSTRAINT fk_baggage_check_passenger FOREIGN KEY (passenger_id)
121 REFERENCES passengers (passenger_id);
122
123

```

```
ALTER TABLE baggage_check
ADD CONSTRAINT fk_baggage_check_booking FOREIGN KEY (booking_id)
REFERENCES booking (booking_id);
```

```
ALTER TABLE baggage
ADD CONSTRAINT fk_baggage_booking FOREIGN KEY (booking_id)
REFERENCES booking (booking_id);
```

```
ALTER TABLE boarding_pass
ADD CONSTRAINT fk_boarding_pass_booking FOREIGN KEY (booking_id)
REFERENCES booking (booking_id);
```

```
ALTER TABLE booking_flight
ADD CONSTRAINT fk_booking_flight_booking FOREIGN KEY (booking_id)
REFERENCES booking (booking_id);
```

```
ALTER TABLE booking_flight
ADD CONSTRAINT fk_booking_flight_flight FOREIGN KEY (flight_id)
REFERENCES flights (flight_id);
```

```
ALTER TABLE flights
ADD CONSTRAINT fk_flight_departure_airport FOREIGN KEY (departing_airport_id)
REFERENCES airport (airport_id);
```

```
ALTER TABLE flights
ADD CONSTRAINT fk_flight_arrival_airport FOREIGN KEY (arriving_airport_id)
REFERENCES airport (airport_id);
```

```
ALTER TABLE flights
ADD CONSTRAINT fk_flight_airline FOREIGN KEY (airline_id)
REFERENCES airline (airline_id);
```

## DML

1.

The screenshot displays a PostgreSQL database management interface with two panels. The left panel shows a sidebar with a tree view of database objects, including Schemas, Tables, and Views. The right panel shows the SQL editor and the Data Output window.

**Top Panel:**

- Query:** A SQL script for creating and populating tables: `my_airport/postgres@MyDB`  
157 `INSERT INTO Airport(airport_id, airport_name, country, state, city, created_at, updated_at)`  
158 `SELECT g, 'Airport_' || g, 'Country_' || g, 'State_' || g, 'City_' || g, now(), now()`  
159 `FROM generate_series(1,200) g;`  
160  
161  
162 `INSERT INTO Airline(airline_id, airline_code, airline_name, airline_country, created_at, updated_at)`  
163 `SELECT g, 'AC_' || g, 'Airline_' || g, 'Country_' || g, now(), now()`  
164 `FROM generate_series(1,200) g;`  
165  
166  
167 `INSERT INTO Passengers(passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_`  
168 `SELECT g, 'First_' || g, 'Last_' || g,`  
169 `CURRENT_DATE - (g || ' days')::interval,`  
170 `CASE WHEN g % 2 = 0 THEN 'Male' ELSE 'Female' END,`  
171 `'Country_' || g, 'Country_' || g, 'P' || g, now(), now()`  
172 `FROM generate_series(1,200) g;`  
173  
174  
175 `INSERT INTO Flights(flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, dep`  
176 `SELECT g, now() + (g || ' hours')::interval, now() + ((g+2) || ' hours')::interval,`  
177 `(g % 200) + 1, ((g+1) % 200) + 1, 'Gate_' || g, 'Gate_' || g, (g % 200) + 1,`  
178 `now() + (g || ' hours')::interval, now() + ((g+2) || ' hours')::interval,`  
179 `now(), now()`  
180 `FROM generate_series(1,200) g;`  
181  
182  
183 `INSERT INTO Booking(book_id, flight_id, passenger_id, booking_platform, created_at, updated_at, status, ticket_p`  
184 `SELECT g, (g % 200) + 1, (g % 200) + 1, 'Platform_' || g, now(), now(),`  
185 `CASE WHEN g % 2 = 0 THEN 'Confirmed' ELSE 'Pending' END,`  
186 `(1000 + g * 10)::DECIMAL(7,2)`  
187 `FROM generate_series(1,200) g;`  
188  
189  
190 `INSERT INTO Booking_flight(booking_flight_id, booking_id, flight_id, created_at, updated_at)`  
191 `SELECT g, (g % 200) + 1, (g % 200) + 1, now(), now()`  
192 `FROM generate_series(1,200) g;`  
193
- Data Output:** A table with 7 columns: `baggage_check_id` (integer), `check_result` (character varying(50)), `created_at` (timestamp without time zone), `updated_at` (timestamp without time zone), `booking_id` (integer), `passenger_id` (integer). The table shows 1 row of data.

**Bottom Panel:**

- Query:** A SQL script for creating and populating tables: `my_airport/postgres@MyDB`  
190 `INSERT INTO Booking_flight(booking_flight_id, booking_id, flight_id, created_at, updated_at)`  
191 `SELECT g, g, (g % 200) + 1, now(), now()`  
192 `FROM generate_series(1,200) g;`  
193  
194  
195 `INSERT INTO Baggage(baggage_id, weight_in_kg, created_at, updated_at, booking_id)`  
196 `SELECT g, (g % 30 + 5)::DECIMAL(4,2), now(), now(), g`  
197 `FROM generate_series(1,200) g;`  
198  
199  
200 `INSERT INTO Baggage_check(baggage_check_id, check_result, created_at, updated_at, booking_id, passenger_id)`  
201 `SELECT g,`  
202 `CASE WHEN g % 2 = 0 THEN 'Pass' ELSE 'Fail' END,`  
203 `now(), now(), g, g`  
204 `FROM generate_series(1,200) g;`  
205  
206  
207 `INSERT INTO Boarding_pass(boarding_pass_id, booking_id, seat, boarding_time, created_at, updated_at)`  
208 `SELECT g, g, 'Seat_' || g, now() + (g || ' minutes')::interval, now(), now()`  
209 `FROM generate_series(1,200) g;`  
210  
211  
212 `INSERT INTO Security_check(security_check_id, check_result, created_at, updated_at, passenger_id)`  
213 `SELECT g,`  
214 `CASE WHEN g % 2 = 0 THEN 'Cleared' ELSE 'Flagged' END,`  
215 `now(), now(), g`  
216 `FROM generate_series(1,200) g;`  
217  
218  
219  
220  
221  
222  
223
- Data Output:** A table with 7 columns: `baggage_check_id` (integer), `check_result` (character varying(50)), `created_at` (timestamp without time zone), `updated_at` (timestamp without time zone), `booking_id` (integer), `passenger_id` (integer). The table shows 1 row of data.

2.

The screenshot shows a database client interface with a SQL editor at the top and a data output table below. The SQL editor contains the following code:

```
217 FROM generate_series(1,200) g;  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227 INSERT INTO Airline (airline_id, airline_code, airline_name, airline_country, created_at, updated_at)  
228 VALUES (201,'KZ001', 'KazAir', 'Kazakhstan', now(), now());  
229 select *from airline;  
230
```

The data output table shows the results of the query. It has 7 columns: `airline_id` (PK integer), `airline_code` (character varying (30)), `airline_name` (character varying (50)), `airline_country` (character varying (50)), `created_at` (timestamp without time zone), and `updated_at` (timestamp without time zone). The table displays rows 183 through 201. Row 201 is highlighted, showing the data for 'KazAir' from Kazakhstan, created and updated on 2025-09-23 at 00:14:58.047603.

airline_id	airline_code	airline_name	airline_country	created_at	updated_at
183	AC_183	Airline_183	Country_183	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
184	AC_184	Airline_184	Country_184	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
185	AC_185	Airline_185	Country_185	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
186	AC_186	Airline_186	Country_186	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
187	AC_187	Airline_187	Country_187	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
188	AC_188	Airline_188	Country_188	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
189	AC_189	Airline_189	Country_189	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
190	AC_190	Airline_190	Country_190	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
191	AC_191	Airline_191	Country_191	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
192	AC_192	Airline_192	Country_192	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
193	AC_193	Airline_193	Country_193	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
194	AC_194	Airline_194	Country_194	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
195	AC_195	Airline_195	Country_195	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
196	AC_196	Airline_196	Country_196	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
197	AC_197	Airline_197	Country_197	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
198	AC_198	Airline_198	Country_198	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
199	AC_199	Airline_199	Country_199	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
200	AC_200	Airline_200	Country_200	2025-09-23 00:10:04.943487	2025-09-23 00:10:04.943487
201	KZ001	KazAir	Kazakhstan	2025-09-23 00:14:58.047603	2025-09-23 00:14:58.047603

Total rows: 201 Query complete 00:00:00.057 CRLF Ln 229, Col 1

3.

The screenshot shows a database client interface with a SQL editor at the top and a data output table below. The SQL editor contains the following code:

```
234  
235 UPDATE Airline SET airline_country = 'Turkey'  
236 WHERE airline_name = 'KazAir';  
237 select *from airline where airline_id = 201 ;  
238  
239  
240  
241  
242  
243  
244  
245  
246
```

The data output table shows the results of the query. It has 7 columns: `airline_id` (PK integer), `airline_code` (character varying (30)), `airline_name` (character varying (50)), `airline_country` (character varying (50)), `created_at` (timestamp without time zone), and `updated_at` (timestamp without time zone). The table displays row 1, showing the data for 'KazAir' from Turkey, created and updated on 2025-09-23 at 00:14:58.047603.

airline_id	airline_code	airline_name	airline_country	created_at	updated_at	
1	201	KZ001	KazAir	Turkey	2025-09-23 00:14:58.047603	2025-09-23 00:14:58.047603

4.

my\_airport/postgres@MyDB

Query Query History

```
245
246
247
248 INSERT INTO Airline (airline_id, airline_code, airline_name, airline_country, created_at, updated_at)
249 VALUES
250 (202,'FR002', 'AirEasy', 'France', '2024-09-23 12:30:00':timestamp, now()),
251 (203,'BR003', 'FlyHigh', 'Brazil', '2023-10-03 13:45:00':timestamp, now()),
252 (204,'PL004', 'FlyFly', 'Poland', '2025-03-05 18:30:00':timestamp, now());
253
254
255
256
```

Data Output Messages Notifications

Showing rows: 1 to 3 Page No: 1 of 1

airline_id	airline_code	airline_name	airline_country	created_at	updated_at	
[PK] integer	character varying (30)	character varying (50)	character varying (50)	timestamp without time zone	timestamp without time zone	
1	202	FR002	AirEasy	France	2024-09-23 12:30:00	2025-09-23 00:21:06.980918
2	203	BR003	FlyHigh	Brazil	2023-10-03 13:45:00	2025-09-23 00:22:52.25004
3	204	PL004	FlyFly	Poland	2025-03-05 18:30:00	2025-09-23 00:22:52.25004

5.

Catalogs  
Event Triggers  
Extensions  
Foreign Data Wrappers  
Languages  
Publications  
Schemas (1)  
public  
Aggregates  
Collations  
Domains  
FTS Configurations  
FTS Dictionaries  
FTS Parsers  
FTS Templates  
Foreign Tables  
Functions  
Materialized Views  
Operators  
Procedures  
Sequences  
Tables (10)  
airline  
airport  
baggage  
baggage\_check  
boarding\_pass  
booking  
booking\_flight  
flights  
passengers  
security\_check  
Trigger Functions  
Types  
Views  
Subscriptions  
postgres  
suppliers  
ninh/Group Roles  
spaces

my\_airport/postgres@MyDB

Query Query History

```
253
254
255
256
257 DELETE FROM Flights
258 WHERE EXTRACT(YEAR FROM sch_arrival_time) = 2024;
259
260
261
262
263
264
265
266
267
268
269
270
271
```

Data Output Messages Notifications

Showing rows: 1 to 200 Page No: 1 of 1

flight_id	sch_departure_time	sch_arrival_time	departing_airport_id	arriving_airport_id	departing_gate	arriving_gate	airline_id	sct_departure_time	sct_arrival_time
[PK] integer	timestamp without time zone	timestamp without time zone	integer	integer	text	character varying (50)	integer	timestamp without time zone	timestamp without time zone
1	2025-09-23 01:10:04.943487	2025-09-23 03:10:04.943487	2	3	Gate_1	Gate_1	2	2025-09-23 01:10:04.943487	2025-09-23 03:10:04.943487
2	2025-09-23 02:10:04.943487	2025-09-23 04:10:04.943487	3	4	Gate_2	Gate_2	3	2025-09-23 02:10:04.943487	2025-09-23 04:10:04.943487
3	2025-09-23 03:10:04.943487	2025-09-23 05:10:04.943487	4	5	Gate_3	Gate_3	4	2025-09-23 03:10:04.943487	2025-09-23 05:10:04.943487
4	2025-09-23 04:10:04.943487	2025-09-23 06:10:04.943487	5	6	Gate_4	Gate_4	5	2025-09-23 04:10:04.943487	2025-09-23 06:10:04.943487
5	2025-09-23 05:10:04.943487	2025-09-23 07:10:04.943487	6	7	Gate_5	Gate_5	6	2025-09-23 05:10:04.943487	2025-09-23 07:10:04.943487
6	2025-09-23 06:10:04.943487	2025-09-23 08:10:04.943487	7	8	Gate_6	Gate_6	7	2025-09-23 06:10:04.943487	2025-09-23 08:10:04.943487
7	2025-09-23 07:10:04.943487	2025-09-23 09:10:04.943487	8	9	Gate_7	Gate_7	8	2025-09-23 07:10:04.943487	2025-09-23 09:10:04.943487
8	2025-09-23 08:10:04.943487	2025-09-23 10:10:04.943487	9	10	Gate_8	Gate_8	9	2025-09-23 08:10:04.943487	2025-09-23 10:10:04.943487
9	2025-09-23 09:10:04.943487	2025-09-23 11:10:04.943487	10	11	Gate_9	Gate_9	10	2025-09-23 09:10:04.943487	2025-09-23 11:10:04.943487
10	2025-09-23 10:10:04.943487	2025-09-23 12:10:04.943487	11	12	Gate_10	Gate_10	11	2025-09-23 10:10:04.943487	2025-09-23 12:10:04.943487
11	2025-09-23 11:10:04.943487	2025-09-23 13:10:04.943487	12	13	Gate_11	Gate_11	12	2025-09-23 11:10:04.943487	2025-09-23 13:10:04.943487
12	2025-09-23 12:10:04.943487	2025-09-23 14:10:04.943487	13	14	Gate_12	Gate_12	13	2025-09-23 12:10:04.943487	2025-09-23 14:10:04.943487
13	2025-09-23 13:10:04.943487	2025-09-23 15:10:04.943487	14	15	Gate_13	Gate_13	14	2025-09-23 13:10:04.943487	2025-09-23 15:10:04.943487
14	2025-09-23 14:10:04.943487	2025-09-23 16:10:04.943487	15	16	Gate_14	Gate_14	15	2025-09-23 14:10:04.943487	2025-09-23 16:10:04.943487
15	2025-09-23 15:10:04.943487	2025-09-23 17:10:04.943487	16	17	Gate_15	Gate_15	16	2025-09-23 15:10:04.943487	2025-09-23 17:10:04.943487

Total rows: 200 Rows complete 00:00:00.123 FILE 1x 260x 1

## 6.

The screenshot shows the pgAdmin 4 interface with the 'my\_airport/postgres@MyDB' database selected. The left sidebar displays the database catalog, with 'Tables (10)' expanded and 'security\_check' selected. The main query editor shows the following SQL code:

```
275  
276 UPDATE Booking  
277 SET ticket_price = ticket_price * 1.15;  
278  
279  
280  
281  
282  
283  
284  
285
```

The 'Data Output' tab is active, showing the message: 'UPDATE 200' and 'Query returned successfully in 76 msec.'

## 7.

The screenshot shows the pgAdmin 4 interface with the 'my\_airport/postgres@MyDB' database selected. The left sidebar displays the database catalog, with 'Tables (10)' expanded and 'security\_check' selected. The main query editor shows the following SQL code:

```
289  
290 DELETE FROM Baggage_check  
291 WHERE booking_id IN (  
292     SELECT booking_id FROM Booking WHERE ticket_price < 10000  
293 );  
294  
295 DELETE FROM Baggage  
296 WHERE booking_id IN (  
297     SELECT booking_id FROM Booking WHERE ticket_price < 10000  
298 );  
299  
300 DELETE FROM Boarding_pass  
301 WHERE booking_id IN (  
302     SELECT booking_id FROM Booking WHERE ticket_price < 10000  
303 );  
304  
305 DELETE FROM Booking_flight  
306 WHERE booking_id IN (  
307     SELECT booking_id FROM Booking WHERE ticket_price < 10000  
308 );  
309  
310 DELETE FROM Booking  
311 WHERE ticket_price < 10000;  
312  
313  
314
```

The 'Data Output' tab is active, showing the message: 'DELETE 200' and 'Query returned successfully in 38 msec.'