Objectives

After completing this lab, you will be able to:

- Create roles in a database and grant them select permissions
- Create new users in the database and assign them the appropriate role
- Revoke and deny access to the database from a user

DOWNLOAD RUSSIA_small.sql

theia@theiadocker-rajendraabrotheia@theiadocker-rajendraabro:/home/project$ wurses-data.s3

wget
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/example-guided-project/flights_RUSSIA_small.sql

--2025-12-24 07:36:15--
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/example-guided-project/flights_RUSSIA_small.sql

Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104, 169.63.118.104

Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:443... connected.

HTTP request sent, awaiting response... 200 OK

Length: 103865229 (99M) [application/x-sql]

Saving to: 'flights_RUSSIA_small.sql'

2025-12-24 07:36:18 (39.3 MB/s) - 'flights_RUSSIA_small.sql' saved [103865229/103865229]

RESTORE DEMO DATABASE

postgres=# \i flights_RUSSIA_small.sql
SET

```
SET
SET
SET
SET
SET
SET
SET
psql:flights_RUSSIA_small.sql:17: ERROR:  database "demo" does not exist
CREATE DATABASE
psql (14.19 (Ubuntu 14.19-0ubuntu0.22.04.1), server 13.2)
You are now connected to database "demo" as user "postgres".
SET
CREATE SCHEMA
COMMENT
CREATE EXTENSION
COMMENT
SET
CREATE FUNCTION
CREATE FUNCTION
COMMENT
SET
SET
CREATE TABLE
COMMENT
CREATE VIEW
COMMENT
COMMENT
CREATE VIEW
COMMENT
COMMENT
COMMENT
CREATE TABLE
COMMENT
COMMENT
CREATE TABLE
COMMENT
COMMENT
CREATE TABLE
COMMENT
COMMENT
CREATE SEQUENCE
ALTER SEQUENCE
CREATE VIEW
COMMENT
```

Verify restoring database

```
demo=# \dt
              List of relations
  Schema  |      Name       | Type  |  Owner
----------+-----------------+-------+----------
 bookings | aircrafts_data  | table | postgres
 bookings | airports_data   | table | postgres
 bookings | boarding_passes | table | postgres
 bookings | bookings        | table | postgres
 bookings | flights         | table | postgres
 bookings | seats           | table | postgres
 bookings | ticket_flights  | table | postgres
 bookings | tickets         | table | postgres
```

Task A: Create a `read_only` role and grant it privileges

```
demo=# CREATE ROLE read_only;
CREATE ROLE
```

the privilege to connect to the `demo` database

```
demo=# GRANT CONNECT ON DATABASE demo TO read_only;
GRANT
```

`bookings` schema. Grant the privilege for the `read_only` role to use the schema

```
demo=# GRANT USAGE ON SCHEMA bookings TO read_only;
GRANT
```

the `SELECT` privilege is needed. To grant this privilege

```
demo=# GRANT SELECT ON ALL TABLES IN SCHEMA bookings TO read_only;
GRANT
```

Task B: Create a `read_write` role and grant it privileges

CREATE ROLE

```
demo=# CREATE ROLE read_write;
CREATE ROLE
```

the privilege to connect to the demo database

```
demo=# GRANT CONNECT ON DATABASE demo TO read_write;
GRANT
```

bookings schema. Grant the privilege for the read_only role to use the schema

```
demo=# GRANT USAGE ON SCHEMA bookings TO read_write;
GRANT
```

the SELECT privilege is needed. To grant this privilege

```
demo=# GRANT SELECT, INSERT, DELETE, UPDATE ON ALL TABLES IN SCHEMA bookings
TO read_write;
GRANT
```

Exercise 2: Add a New User and Assign them a Relevant Role

```
demo=# CREATE USER user_a WITH PASSWORD 'user_a_password';
CREATE ROLE
```

assign user_a the read_only role

```
demo=# GRANT read_only TO user_a;
GRANT ROLE
demo=# \du
```

```
                             List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 postgres  | Superuser, Creat:
```

Exercise 3: Revoke and Deny Access

```
demo=# REVOKE SELECT ON aircrafts_data FROM user_a;
REVOKE
```

assign `user_a` the `read_only` role

```
demo=# REVOKE read_only FROM user_a;
REVOKE ROLE
```

```
demo=# \du
```

```
                                   List of roles
 Role name  |                         Attributes                         | Member of
------------+------------------------------------------------------------+-----------
 postgres   | Superuser, Creat:
```