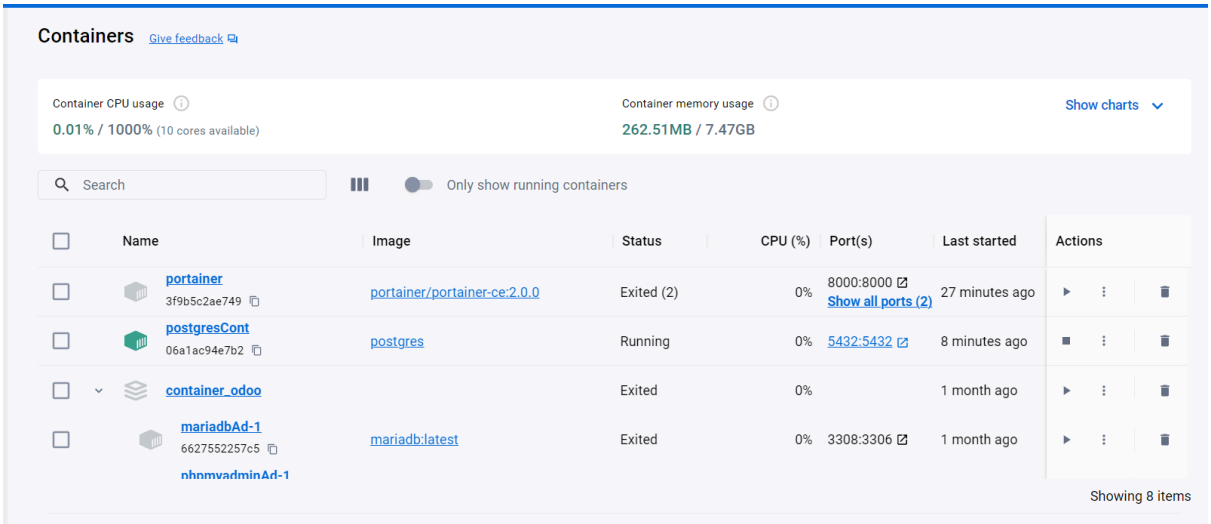
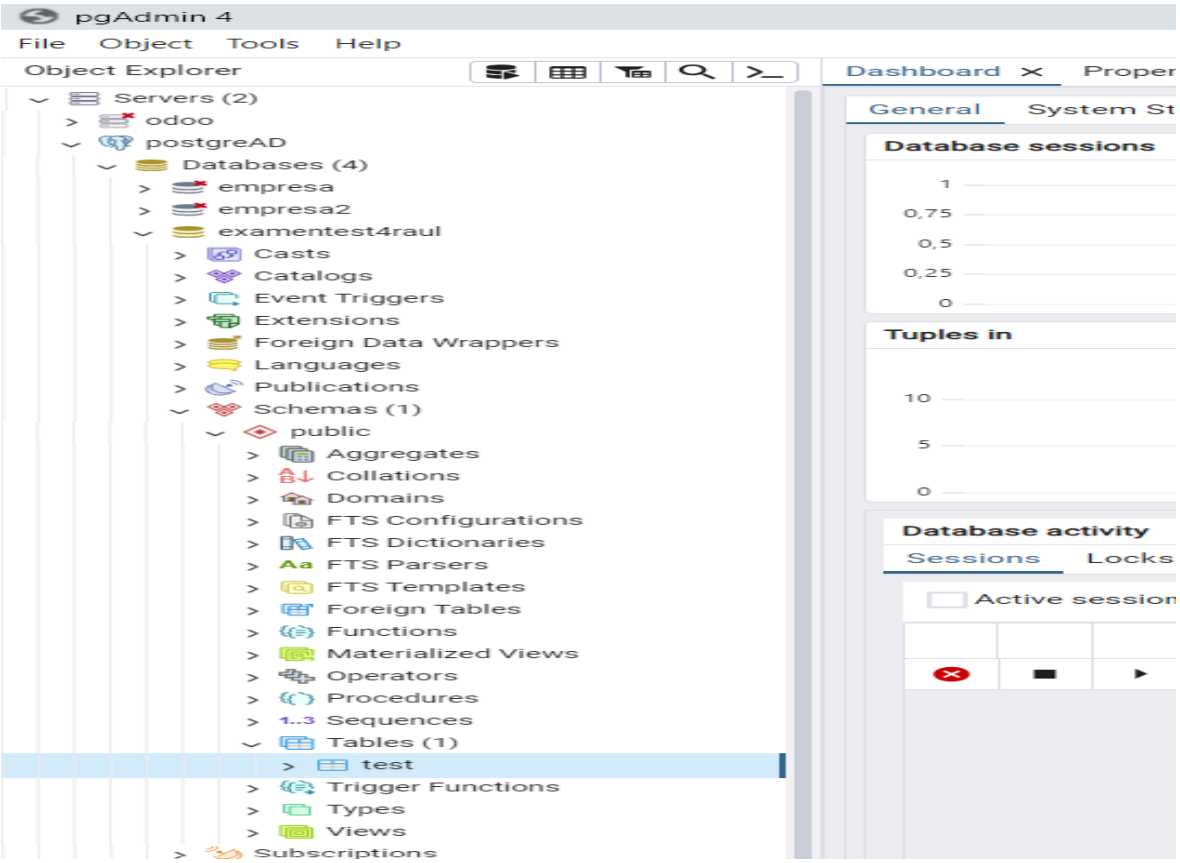
1. PostgreSQL
1.a (2p) Create a PostgreSQL instance
In this exercise you are invited to create and run a new server on your machine, running PostgreSQL. Is highly recommended to use Docker.

- (0,5p) Write the sentence to create it.

-docker pull postgre:latest
-docker run -d —name postgresCont -p 5432:5432 -e POSTGRES_PASSWORD=6698 postgres



- (0,5p) Run a client, such DBeaver to create a sample database.



- (1p) Create a java project and connect to such database.

1.b (2p) Implementing an OR Database

We are dealing with a classic inheritance exercise that we are going to model with PostgreSQL. From an Employee we will store the name and address of it (type of street, street and number) and a set of telephones. The valid street types will be Street, Avenue or Part. Employees can be:

- Sellers, in which case we will store your commission on sales and their role (telephone, distributor or door)
- Commercial, in which case we will store its remuneration (extra salary part) and area of action (Levante, South, Central or North)

In addition, for each Employee it will be indicated who is his superior, through a reference to another Employee.

*Use all the concepts studied on the subject.

2. ObjectDB

2.a (2p) Prepare your environment

Create 2 Hello ObjectDB projects to use ObjectDB with it: one with gradle and another with maven, and run this test program:

```
EntityManagerFactory emf;
EntityManager em;
emf = Persistence.createEntityManagerFactory("testBD.odb");

// The extension is not mandatory, but is a good option

try {
        em = emf.createEntityManager();
        System.out.println("DB was created");
        em.close();
} catch (PersistenceException ex) {
        System.out.println(ex.getMessage());
}
```

maven:

```java

package com.raul.postgresql1a.objectdbraul_2a;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.PersistenceException;

/**
 *
 * @author rauls
 */

public class Objectdbraul_2a {

    public static void main(String[] args) {
        EntityManagerFactory emf;
        EntityManager em;
        emf  = Persistence.createEntityManagerFactory("testBD.odb");
        try {
            em = emf.createEntityManager();
            System.out.println("DB was created");
            em.close();
        }
        catch (PersistenceException ex) {
            System.out.println(ex.getMessage());
        }
    }
}

```

gradle:

2.b (4p)

We are going to develop a complete case, from the design of the database to its implementation and use. It is the management of a small company, which we will describe: We have some Employees, of whom we need to know their name, date of hire and if they have had any bad performance lately. The employee ID must be automatically assigned by the system.

The data of the bad action should not be stored in the DB, we will only use it at execution time. Each employee has assigned an Address (street, number and block).

- The Departments have a name and a location, apart from their identifier (also assigned by the system). The departments will have several employees. One Employee can only belong to one department.
- As on the Projects (of which we only keep their id and description) we have the employee who is the project manager and of the employees who are participating in the project. An employee can participate in several projects at the same time but only be the head of one.

You have to:

- (1p) Create the classes to store these requirements.
- (1,5p) Create a program that asks the user for some data
- (1,5p) Finally, resolve the following queries:
- Projects without department.
- Employees without departments.