TP1 – SQL with postgreSQL and PGAdmin

Objectifs of the lab :

Use postgres and PGAdmin to interact with the database in order to compute some KPIs.

Tools and Versions

- postgres : v15
- pgAdmin : v6.20

Postgres :

PostgreSQL is a popular open-source relational database management system (RDBMS). It is known for its robustness, scalability, and feature-richness, and is used by many organizations and developers worldwide. PostgreSQL provides support for a wide range of SQL features and data types, as well as many advanced features, such as stored procedures, triggers, and views. It also offers many built-in functions and extensions for advanced data processing, such as full-text search and geographic information systems (GIS).

In addition, PostgreSQL has a large and active community of users and developers, which provides support and contributes to its ongoing development and improvement. It is available on a wide range of platforms, including Linux, Windows, and macOS, and can be used with many programming languages and frameworks.

PGAdmin :

pgAdmin is a popular open-source administration and management tool for PostgreSQL databases. It provides a graphical user interface (GUI) for managing and developing PostgreSQL databases, allowing users to perform a wide range of tasks, including creating and modifying databases, tables, and other database objects, managing users and permissions, writing and executing SQL queries, and more. pgAdmin provides a rich set of features and tools for working with PostgreSQL, including support for multiple servers and databases, drag-and-drop object creation, customizable dashboards and visualizations, and a powerful query editor with syntax highlighting and auto-completion. It also provides support for advanced database features, such as triggers, stored procedures, and full-text search.

pgAdmin is available for Windows, Linux, and macOS, and is maintained by the PostgreSQL Global Development Group. It is free and open source, and can be downloaded from the pgAdmin website or installed through package managers on various operating systems.

Installation

You need to have docker already install on your machine.

Download the docker image postgres. Then start it:

```
$ docker pull totofunku/sql-cours
$ docker run --name postgresql -e POSTGRES_USER=admin \
         -e POSTGRES_PASSWORD=adminadmin -p 5432:5432 \
         -v /data:/var/lib/postgresql/data -d totofunku/sql-cours
```

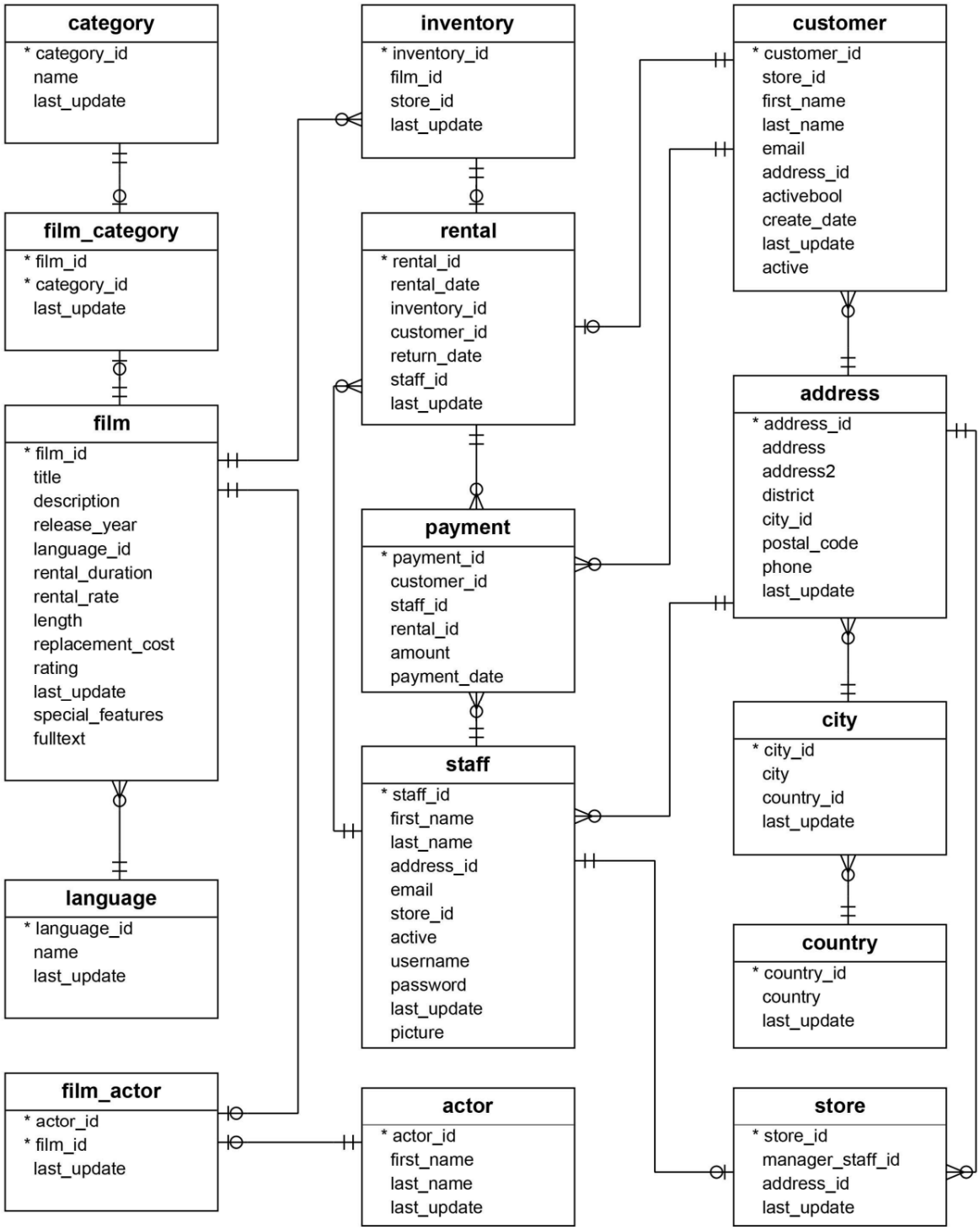
Same thinks for pgAdmin :

```
$ docker pull dpage/pgadmin4:latest
$ docker run --name my-pgadmin -p 82:80 \
         -e "PGADMIN_DEFAULT_EMAIL=pgadmin4@pgadmin.org" \
         -e "PGADMIN_DEFAULT_PASSWORD=test1234" \
         -d dpage/pgadmin4
```

Now you have a running instance of postgresql and an administrator interface to query the database. We will work on a use case of DVD rental store.

1. First you need to register the server on pgAdmin
2. Check that you have access to all the tables
3. Now we will compute 4 querys :
    a. How many movie have the shop ?
    b. How many movies are available ?
    c. What is the monthly revenue of the shop ?
    d. For each client, give the top 3 most viewed movie categories

ER-diagram


category
* category_id
name
last_update
film_category
* film_id
* category_id
last_update
film
* film_id
title
description
release_year
language_id
rental_duration
rental_rate
length
replacement_cost
rating
last_update
special_features
fulltext
language
* language_id
name
last_update
film_actor
* actor_id
* film_id
last_update
actor
* actor_id
first_name
last_name
last_update
inventory
* inventory_id
film_id
store_id
last_update
rental
* rental_id
rental_date
inventory_id
customer_id
return_date
staff_id
last_update
payment
* payment_id
customer_id
staff_id
rental_id
amount
payment_date
staff
* staff_id
first_name
last_name
address_id
email
store_id
active
username
password
last_update
picture
customer
* customer_id
store_id
first_name
last_name
email
address_id
activebool
create_date
last_update
active
address
* address_id
address
address2
district
city_id
postal_code
phone
last_update
city
* city_id
city
country_id
last_update
country
* country_id
country
last_update
store
* store_id
manager_staff_id
address_id
last_update