Департамент образования города Москвы

Государственное автономное образовательное учреждение
высшего образования города Москвы
«Московский городской педагогический университет»

Институт цифрового образования
Департамент информатики, управления и технологий

Лабораторная работа 3
по дисциплине «Проектный практикум по разработке ETL-решений»

Тема: «Практическая работа на вебинаре»

Направление подготовки 38.03.05 – бизнес-информатика
Профиль подготовки «Аналитика данных и эффективное управление»
(очная форма обучения)

Выполнила:
St_88

Москва
2025

ЗАДАНИЕ 1.

Заполнение таблицы и анализ данных в PostgreSQL с визуализацией

Цель: заполнить таблицу person в базе данных PostgreSQL фейковыми данными не менее 100 записей и провести анализ этих данных с использованием SQL. Также создать визуализации для полученных результатов.

Задачи:

1. Заполнение таблицы фейковыми данными.
    a. Используйте библиотеку Faker для генерации фейковых данных.
    b. Вставьте сгенерированные данные в таблицу person в базе данных PostgreSQL.
2. Анализ возраста.
    a. Найдите средний, минимальный и максимальный возраст людей в таблице person.
3. Анализ распределения по городам.
    a. Определите топ-5 городов, в которых проживает наибольшее количество людей.
4. Анализ регистрации.
    a. Найдите количество регистраций в каждом месяце за последний год.
5. Визуализация данных.
    a. Создайте графики для визуализации результатов анализа:
    b. Гистограмма распределения возраста.
    c. Диаграмма топ-5 городов по количеству проживающих.
    d. Линейный график количества регистраций по месяцам.

ХОД РАБОТЫ

1. Подготовка к работе: проверка наличия запущенных контейнеров, в случае если такие есть, то их остановка и удаление:

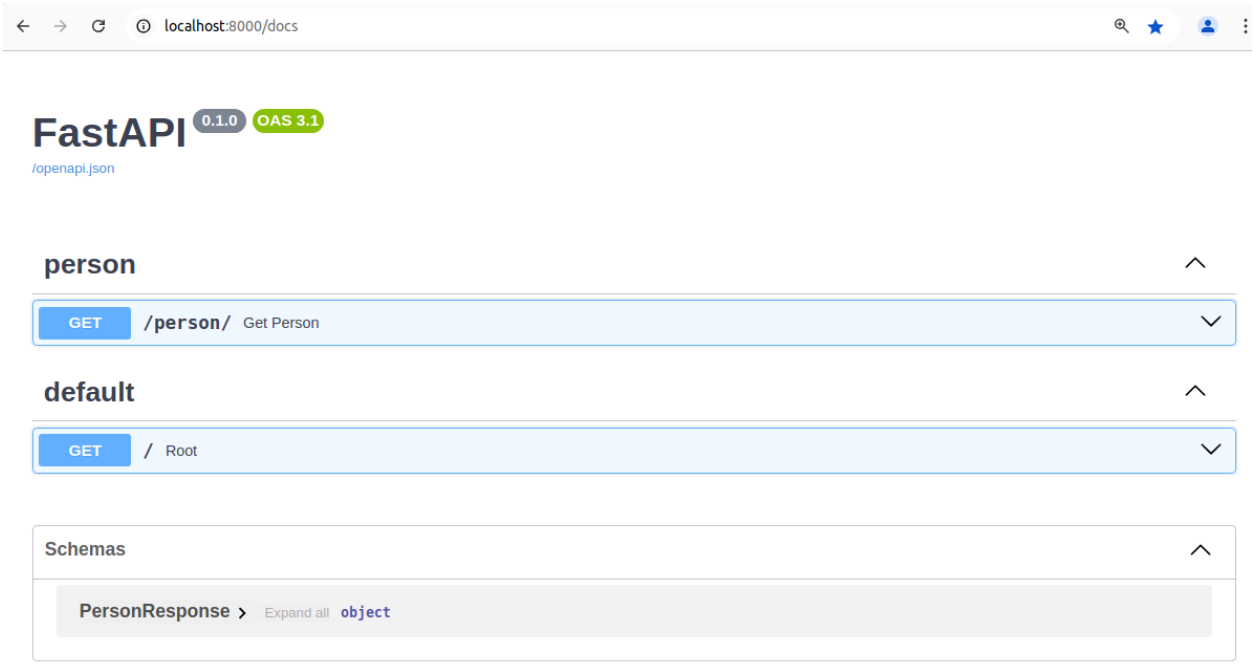
```
dev@dev-vm:~/Downloads/lab_etl/data_for_labs/lab_airflow/lab_0_webinar$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED       STATUS         PORTS
                          NAMES
a5bb7206afbd   postgres:16            "docker-entrypoint.s…"   2 weeks ago   Up 3 minutes   0.0.0.0:5432->5432
/tcp, [::]:5432->5432/tcp    postgres16
ec4919b750da   dpage/pgadmin4:latest  "/entrypoint.sh"         2 weeks ago   Up 3 minutes   0.0.0.0:80->80/tcp
, [::]:80->80/tcp, 443/tcp   pgadmin
dev@dev-vm:~/Downloads/lab_etl/data_for_labs/lab_airflow/lab_0_webinar$ docker kill $(docker ps -q)
a5bb7206afbd
ec4919b750da
dev@dev-vm:~/Downloads/lab_etl/data_for_labs/lab_airflow/lab_0_webinar$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

2. Запуск контейнеров всех необходимых сервисов, а именно:

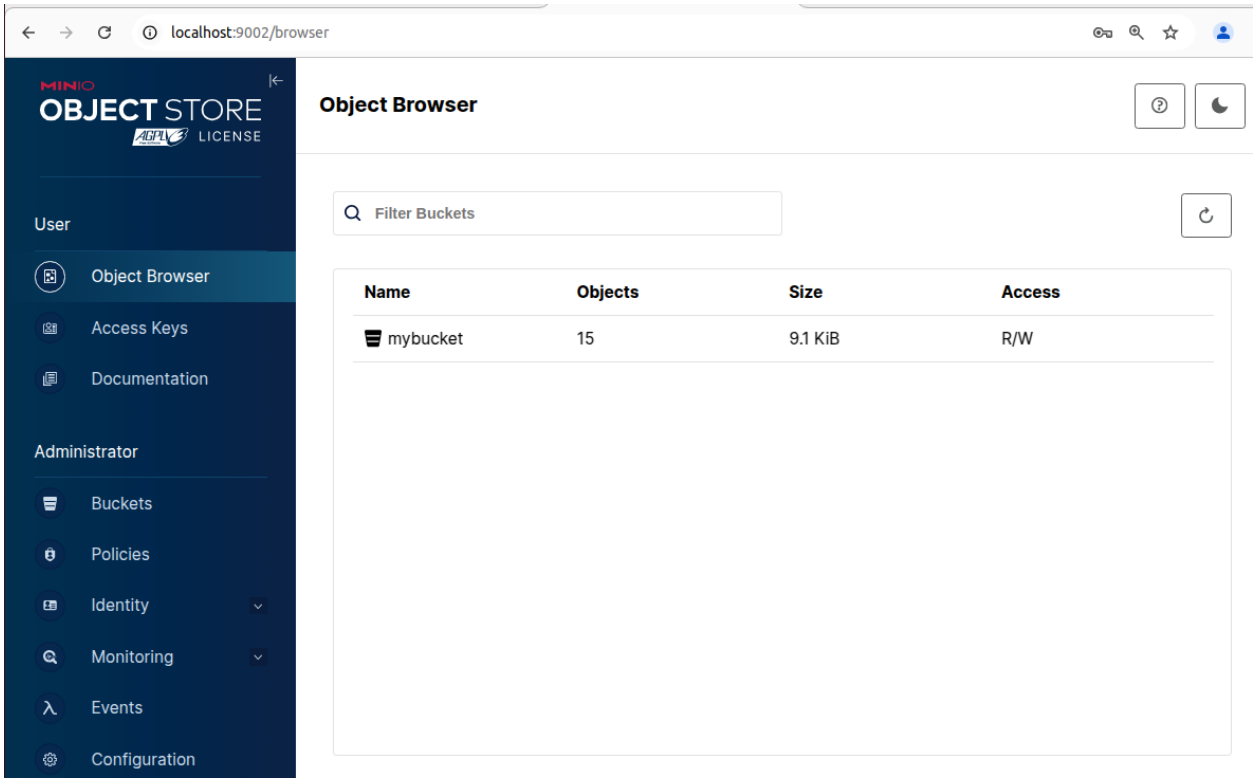
```
dev@dev-vm:~/Downloads/lab_etl/data_for_labs/lab_airflow/lab_0_webinar$ make up-services
sudo docker compose -f docker-compose-services.yaml up -d --build
[sudo] password for dev:
Compose now can delegate build to bake for better performances
Just set COMPOSE_BAKE=true
[+] Building 4.1s (11/11) FINISHED                                                                  docker:default
 => [faker-api internal] load build definition from Dockerfile                                             0.1s
 => => transferring dockerfile: 232B                                                                       0.0s
 => [faker-api internal] load metadata for docker.io/library/python:3.12-slim                              3.5s
 => [faker-api internal] load .dockerignore                                                                0.0s
 => => transferring context: 2B                                                                            0.0s
 => [faker-api 1/5] FROM docker.io/library/python:3.12-slim@sha256:aaa3f8cb64dd64e5f8cb6e58346bdcfa410a1   0.0s
 => [faker-api internal] load build context                                                                0.1s
 => => transferring context: 265B                                                                          0.0s

[+] Running 8/8
 ✔ faker-api                            Built                                                             0.0s
 ✔ Network lab_0_webinar_default        Created                                                           0.3s
 ✔ Volume "lab_0_webinar_minio_data"    Created                                                           0.0s
 ✔ Container zookeeper                  Started                                                           2.2s
 ✔ Container lab_0_webinar-minio-1      Started                                                           2.8s
 ✔ Container pg                         Healthy                                                          14.9s
 ✔ Container ch                         Healthy                                                          35.8s
 ✔ Container lab_0_webinar-faker-api-1  Started                                                          36.4s
```

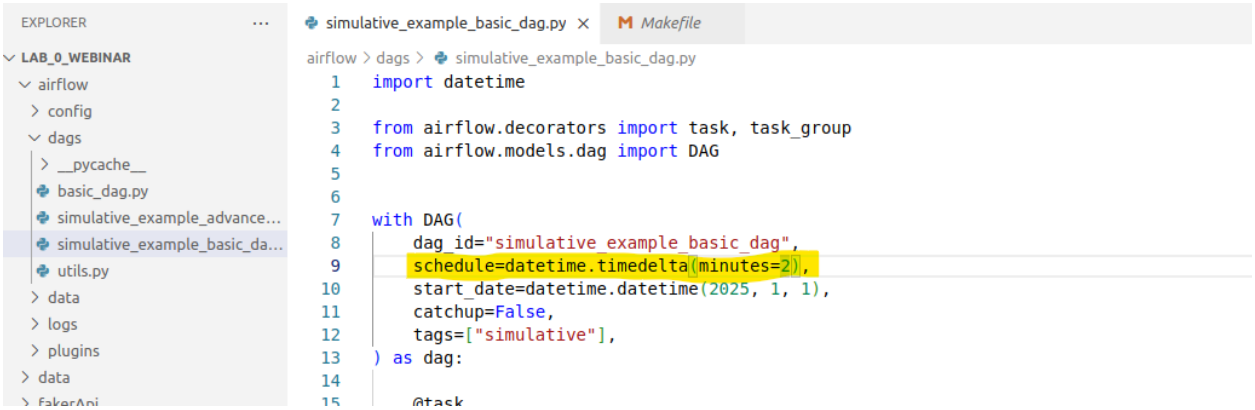
Проверка доступа к Faker (localhost:8000):



Проверка доступа к Minio (localhost:9002):

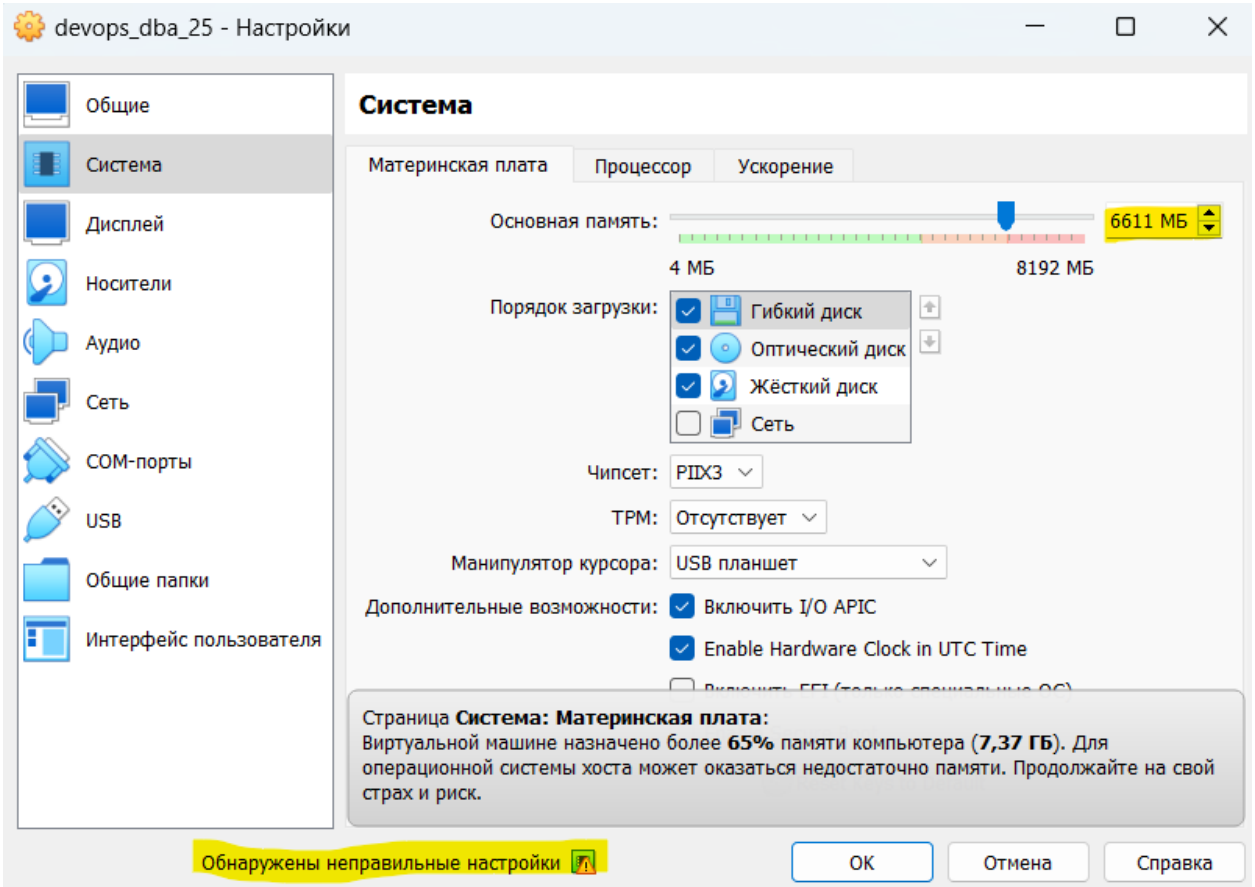


3. Далее по заданию необходимо будет сгенерировать 100 записей. Для того, чтобы сделать это автоматически, перед запуском Airflow необходимо в коде дага, отвечающего за генерацию и запись данных в базу PostgreSQL, установить расписание выполнения дага: раз в 2 минуты:



Изначально, для более частой работы дага было выставлено значение: раз в 18 секунд. Однако, при попытке дальнейшего запуска дага в Airflow у виртуальной машины закончилось ОЗУ, поэтому для, хоть и медленной, но стабильной автоматической генерации было принято решение оставить значение запуска дага: раз в 2 минуты.

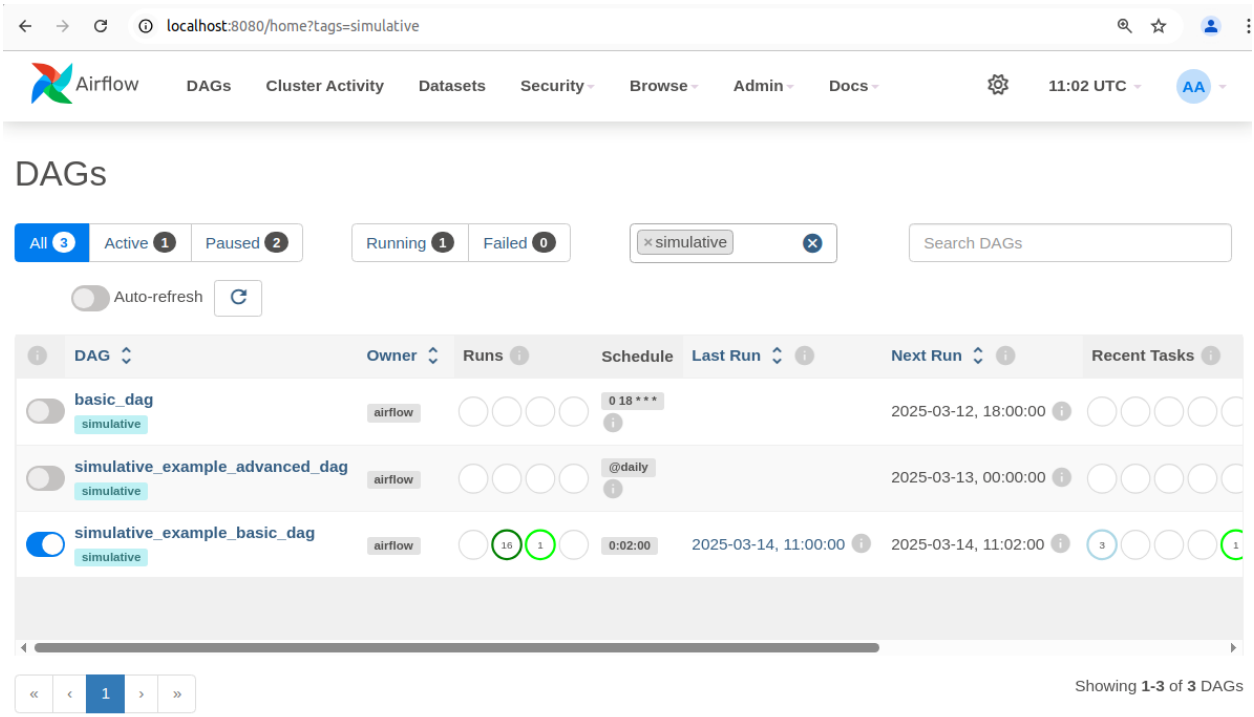
Максимально доступные ресурсы на ноутбуке довольно малы:



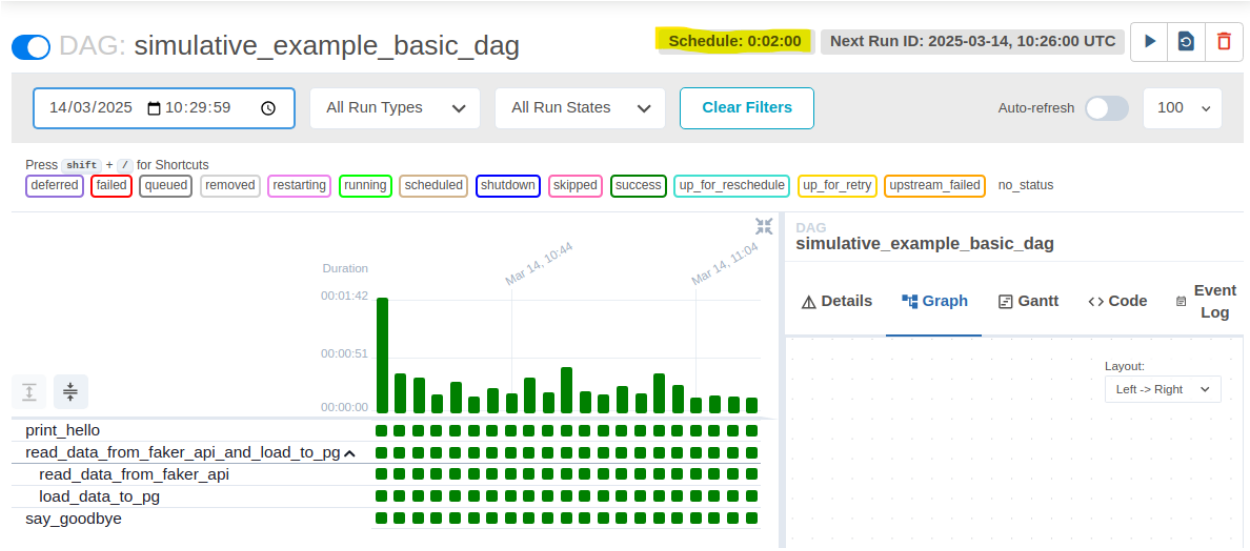
4. Запуск контейнеров, отвечающих за работу Airflow:

```
dev@dev-vm:~/Downloads/lab_etl/data_for_labs/lab_airflow/lab_0_webinar$ make up-af
sudo docker compose -f docker-compose-af.yaml up -d --build
[+] Running 48/48
 ✔ postgres Pulled                                                                66.9s
 ✔ airflow-webserver Pulled                                                      125.8s
 ✔ airflow-triggerer Pulled                                                      125.8s
 ✔ airflow-worker Pulled                                                         125.8s
 ✔ airflow-scheduler Pulled                                                      125.8s
 ✔ airflow-init Pulled                                                           125.8s
 ✔ redis Pulled                                                                   40.9s
WARN[0126] Found orphan containers ([lab_0_webinar-faker-api-1 ch zookeeper lab_0_webinar-minio-1 pg]) for this
 project. If you removed or renamed this service in your compose file, you can run this command with the --remo
ve-orphans flag to clean it up.
[+] Running 8/8
 ✔ Volume "lab_0_webinar_postgres-db-volume"    Created                           0.0s
 ✔ Container lab_0_webinar-redis-1              Healthy                           9.0s
 ✔ Container lab_0_webinar-postgres-1           Healthy                           9.0s
 ✔ Container lab_0_webinar-airflow-init-1       Exited                           38.8s
 ✔ Container lab_0_webinar-airflow-scheduler-1  Started                          40.7s
 ✔ Container lab_0_webinar-airflow-triggerer-1  Started                          41.0s
 ✔ Container lab_0_webinar-airflow-worker-1     Started                          41.0s
 ✔ Container lab_0_webinar-airflow-webserver-1  Started
```

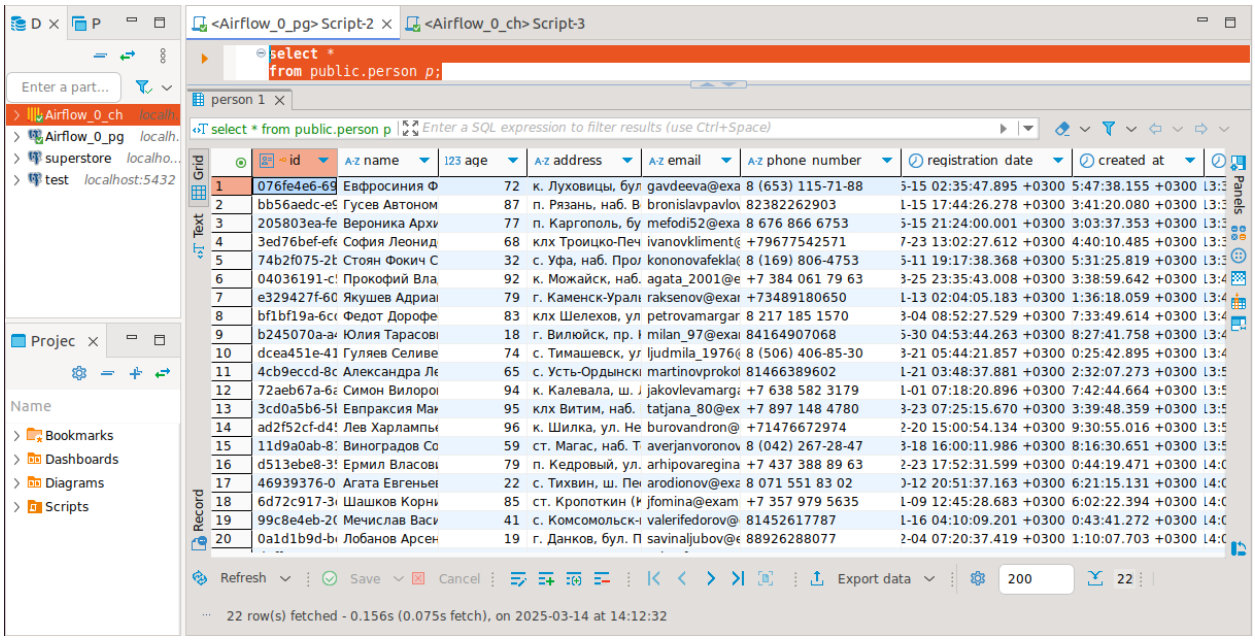
Проверка доступа к Airflow (localhost:8080):



5. Запуск дага, отвечающего за генерацию и запись данных в базу PostgreSQL – как видно, автоматические включение дага раз в 2 минуты отрабатывает:



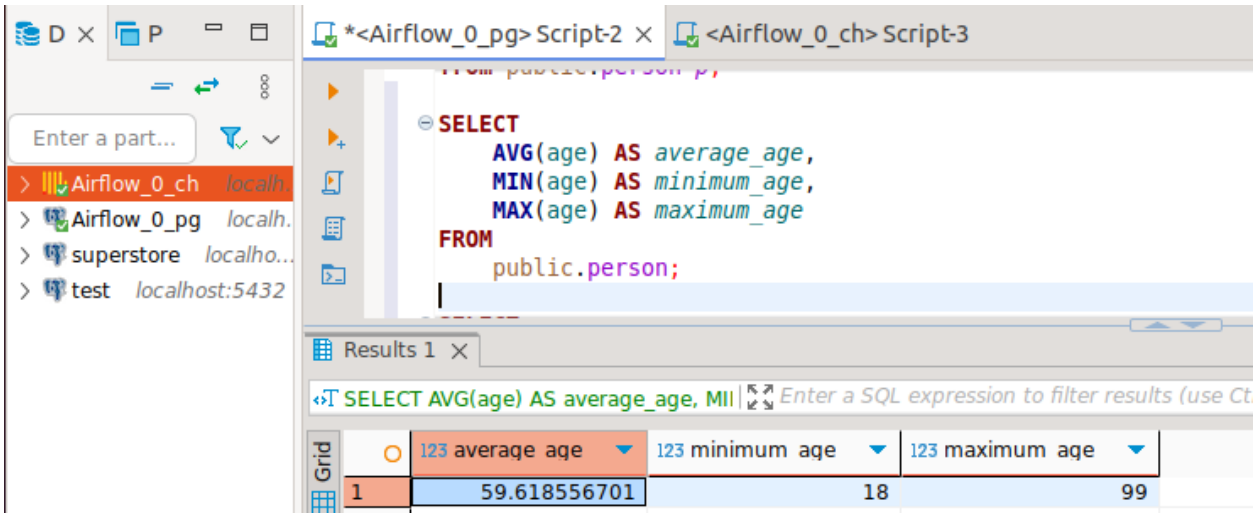
Проверка успешности записи данных в базу PostgreSQL посредствам DBeaver:



6. 97 записей было занесено в таблицу, что составило 194 минуты работы дага

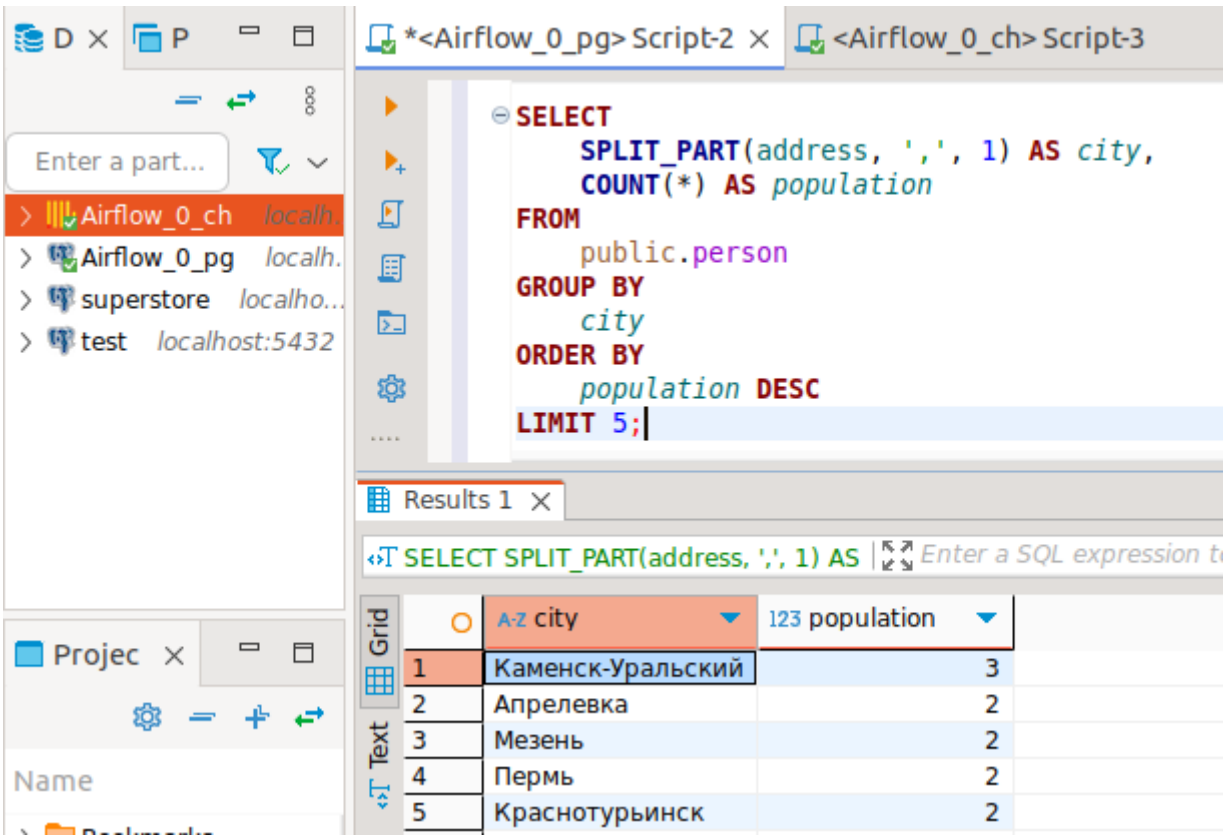
7. На сгенерированных данных необходимо найти средний, минимальный и максимальный возраст людей в таблице person. Скрипт SQL:

```
SELECT
AVG(age) AS average_age,
MIN(age) AS minimum_age,
MAX(age) AS maximum_age
FROM
public.person;
```



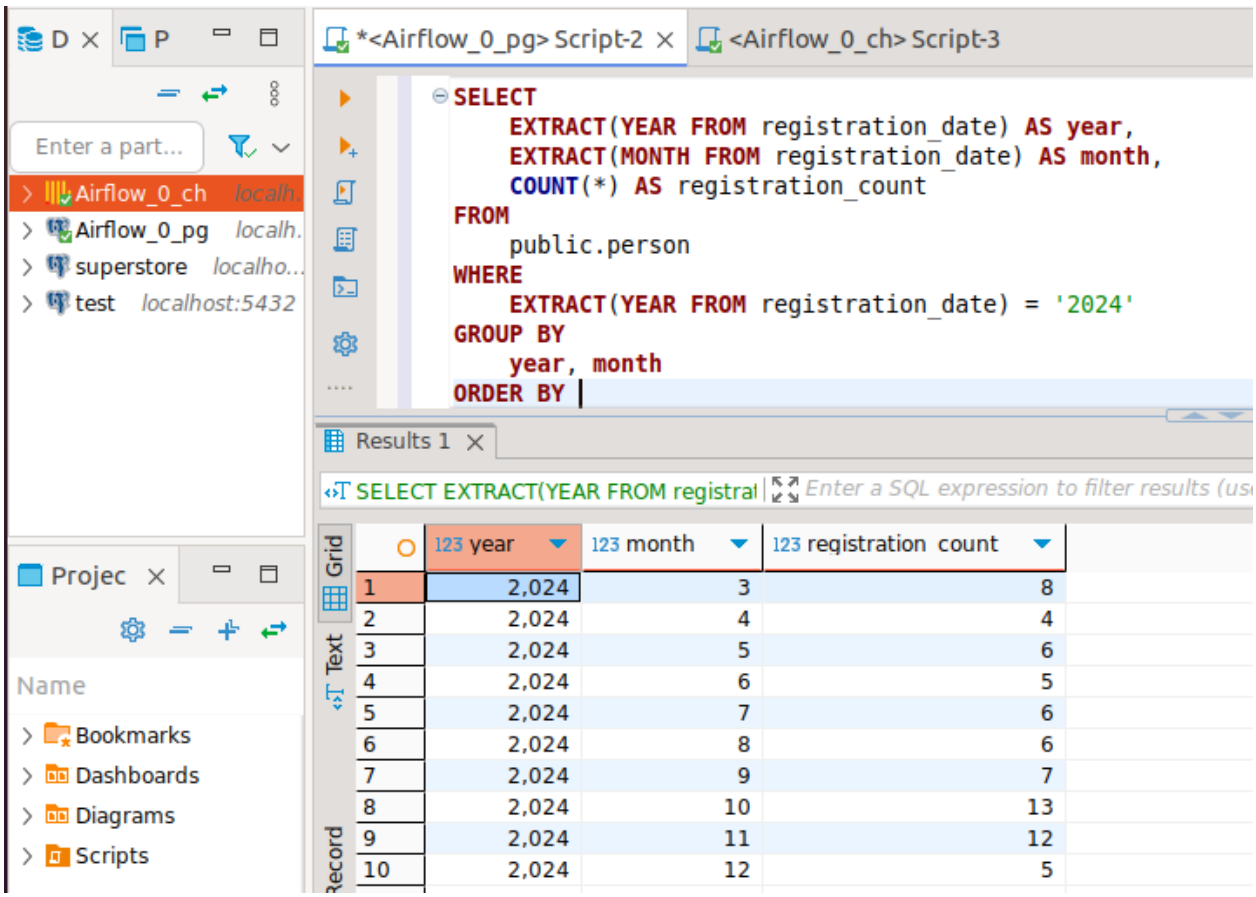
8. Далее необходимо определить топ-5 городов, в которых проживает наибольшее количество людей. Скрипт SQL:

```
SELECT
    TRIM(SPLIT_PART(address, ',', 1), 'к. ') AS city,
    COUNT(*) AS population
FROM
    public.person
GROUP BY
    city
ORDER BY
    population DESC
LIMIT 5;
```



9. Определение количества регистраций в каждом месяце за последний год. Скрипт SQL:

```
SELECT
    EXTRACT(YEAR FROM registration_date) AS year,
    EXTRACT(MONTH FROM registration_date) AS month,
    COUNT(*) AS registration_count
FROM
    public.person
WHERE
    EXTRACT(YEAR FROM registration_date) = '2024'
GROUP BY
    year, month
ORDER BY
    month;
```

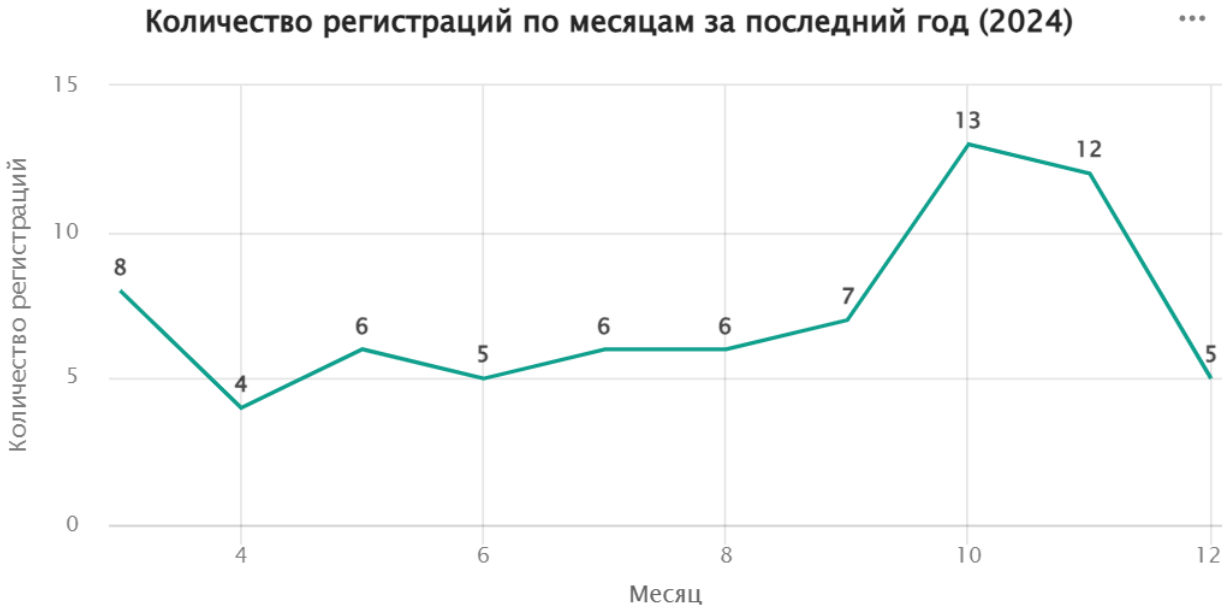


10.Для визуализации результатов анализа, необходимо построить:

- Гистограмма распределения возраста.

- Диаграмма топ-5 городов по количеству проживающих.

- Линейный график количества регистраций по месяцам.



Для этого выгрузим файлы результатов обработки каждого скрипта и визуализируем их с помощью YandexDatalens, потому что там быстро (а время ограничено) и удобно.

Ссылка на дашборд: https://datalens.yandex/25z0zok7axh8o