Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Проектный практикум по разработке ETL-решений

**Вебинар 14.03.2025**

**Практическая работа на вебинаре**

Выполнила: Сергеева А. И., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2025

**Цель работы:** 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. Провести анализ данных, загруженных в таблицу person\_count\_by\_city в ClickHouse, и создать визуализации для полученных результатов.

**Задачи:**

- Заполнение таблицы фейковыми данными.

- Анализ возраста.

- Анализ распределения по городам.

- Анализ регистрации.

- Визуализация данных.

- Анализ количества людей по городам.

- Анализ распределения количества людей по городам.

- Анализ городов с минимальным и максимальным количеством людей.

- Визуализация данных.

**Ход работы:**

Один из DAG посвящён генерации тестовых данных, а другой загрузке данных из базы в ClickHouse. В поле schedule, где задаётся частота обновления данных и запуска DAG было установлено раз в минуту, как видно на рисунке 1 и это минимальное время, которое можно установить.

Изображение выглядит как текст, программное обеспечение, веб-страница, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 – Обновление данных раз в минуту

Для начала работы с Airflow необходимо было запустить контейнеры на рисунке 2.

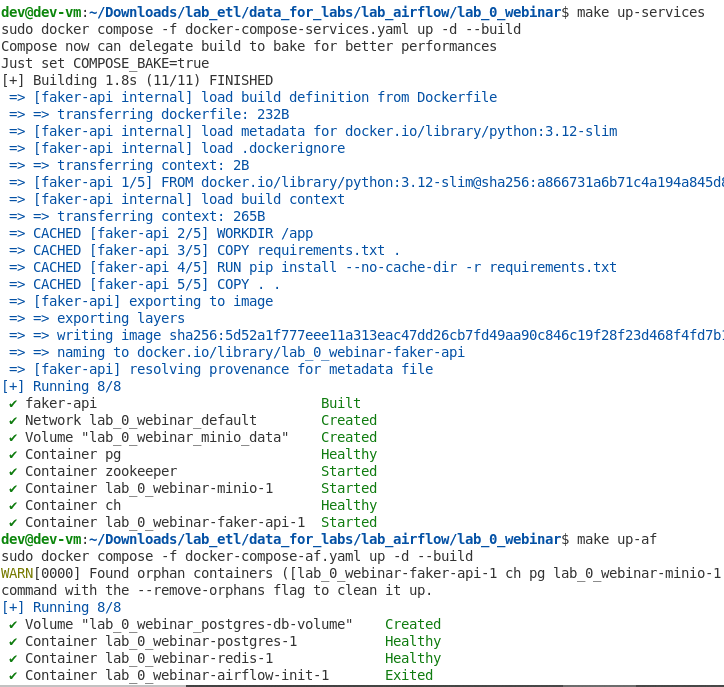


Рисунок 2 – Запуск контейнеров для запуска сервисов и Airflow

Запущенные DAGs показаны на рисунке 3.

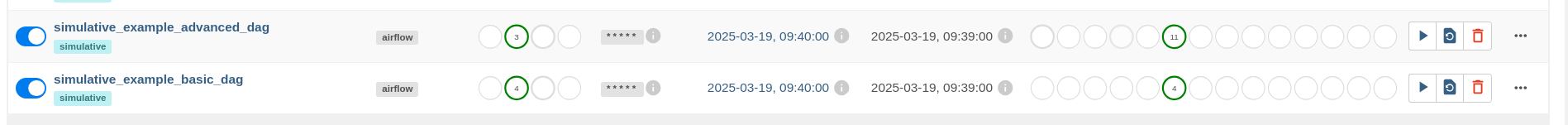


Рисунок 3 – Запуск DAGs по генерации данных и загрузке их в PostgreSQL и ClickHouse

Результаты успешного выполнения в самих DAGs показаны на рисунках 4–5.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4 – Выполнение DAG по загрузке данных в PostgreSQL

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5 - Выполнение DAG по загрузке данных в ClickHouse

После данных шагов с помощью СУБД DBeaver была выполнена проверка загруженных данных на рисунках 6–7.

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6 – Данные, выгруженные в таблицу PostgreSQL

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7 - Данные, выгруженные в таблицу ClickHouse

Далее надо было проанализировать данные с помощью SQL-запросов, первым этапом были просчитаны средний, минимальный и максимальный возраст на рисунке 8.

Изображение выглядит как текст, программное обеспечение, линия, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8 – Минимальный, максимальный и средний возраст

В среднем людям 55 лет, самому младшему 18, а самому старшему 98 лет. Т. к. расчёт числа жителей в каждом из городов должен происходить в шаге по работе с данными PostgreSQL, то необходимо было из столбца с адресом вырезать населённые пункты, как видно на рисунке 9.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9 – Топ 5 городов по численности

Лидирует деревня Сорочинск, в ней проживает 2 жителя, остальные города уникальны. Далее на рисунке 10 было посчитано количество регистраций в каждом месяце за последний год, в данном случае за 2025.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10 – Количество регистраций в каждом месяце

В январе было 7 регистраций, в феврале и марте по 8. Данные результаты необходимо было визуализировать. Решение пало в пользу Python. Результаты были выгружены в Excel-файлы и провизуализированы с помощью Altair.

Гистограмма распределения возраста представлена на рисунке 11. Данные были загружены, а также для удобства добавлены всплывающие подсказки.

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11 – Гистограмма распределения возрастов

Больше всего людей (17 человек) в диапазоне от 40 до 50 лет, а реже всего встречаются от 10 до 20 лет, всего 4 человека. Для расчёта топ 5 городов по численности населения необходимо было оставить все символы до 1 запятой (новый столбец со значением города), далее сгруппировать по названию города и посчитать число проживающий, это новый датафрейм, который в дальнейшем был провизуализирован, 1-е место в рейтинге было окрашено оранжевым. Данные шаги показаны на рисунках 12–13.

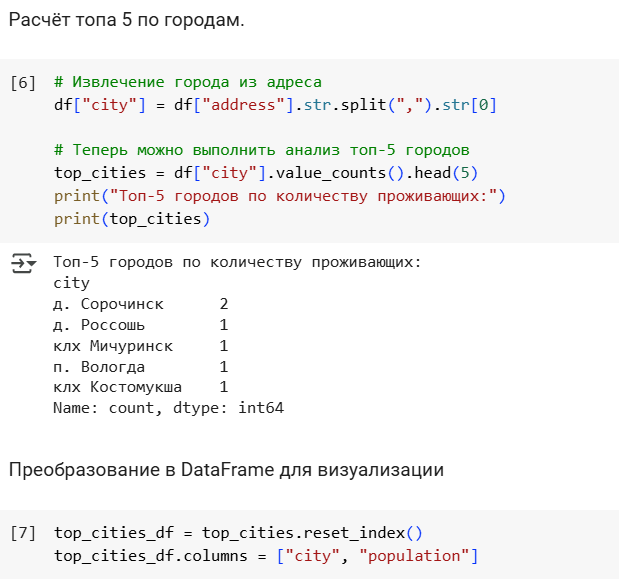


Рисунок 12 – Расчёт топ 5 городов по численности

Изображение выглядит как текст, снимок экрана, Параллельный, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 13 – Гистограмма топа 5 городов по численности

Лидирует деревня Сорочинск, в ней проживает 2 человека, в остальных городах по 1 человеку, такой разброс связан с генерацией данных. Для построения линейного графика по количеству регистраций в месяц были изменены типы данных на тип даты, извлечён месяц и год из даты и построен уже сам график с возможностью масштабирования на рисунках 14-18.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 14 – Изменение типов данных

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 15 – Извлечение месяца и года из даты

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 16 - Группировка по месяцам и подсчёт количества регистраций

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 17 – Преобразование в DataFrame

Изображение выглядит как текст, диаграмма, линия, График

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 18 - Линейный график количества регистраций по месяцам

Пик регистраций был в ноябре (12). Для ClickHouse был выявлен топ 10 городов по численности, суть не меняется, всё также 1 город с 2 людьми и остальные города по 1 человеку, а также рассчитана средняя численность в городах с помощью SQL на рисунках 19–20. Ради интереса и увеличения численности можно попробовать убрать перед названиями тип населённого пункта, деревня, город и прочие. Благодаря этому что-то может измениться.

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 19 – Топ 10 по городам без типа населённого пункта

Отличия есть, появилось больше совпадений, в городах топ 5 найдено по 2 жителя. Теперь считаем среднее количество людей в городах, тут уже с типом населённого пункта на рисунке 20.

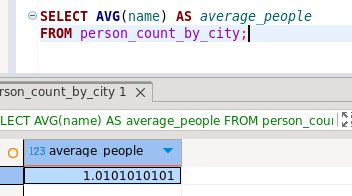


Рисунок 20 – Среднее число жителей в городе

В среднем в городах проживает 1 человек. С минимальным количеством человек, а именно 1 много городов, но надо было вывести 1, как на рисунке 21.

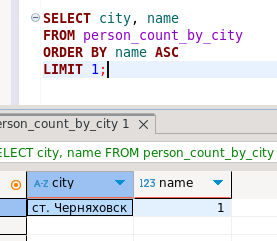


Рисунок 21 – Город с минимальным количеством человек

С максимальным количеством человек только 1 город, как на рисунке 22 и это деревня Сорочинск.

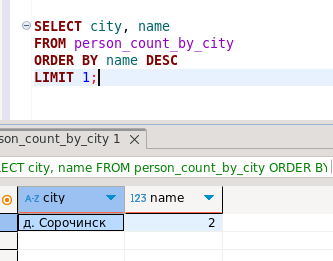


Рисунок 22 – Город с максимальным количеством человек

С данными был проведён аналогичный анализ на Altair. Для начала уже с помощью Python был высчитан топ 10 городов и только потом построена диаграмма топ 10 по проживающим, как на рисунках 23–24.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 23 – Топ 10 городов по численности

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 24 – Диаграмма для рассчитанного топа 10 городов

Далее была построена гистограмма распределения городов по численности населения в порядке убывания численности, как показано на рисунке 25.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 25 – Гистограмма распределения количества людей

Для построения линейного графика, который показывает минимальное и максимальное количество людей в городах, нужно было провести расчеты на рисунке 26 с помощью сортировки.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 26 – Поиск город с максимальной и минимальной численностью населения

Далее был построен линейный график, показывающий минимальное и максимальное количество людей в городах на рисунке 27.

Изображение выглядит как текст, снимок экрана, линия, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 27 - Линейный график, показывающий минимальное и максимальное количество людей в городах

**Выводы:**

Использование библиотеки Faker позволило быстро создать реалистичные фейковые данные, но адреса были уникальны даже по городу, могло быть одинаковое наименование, но разные типы населённых пунктов, из-за этого анализ получился не таким интересным, возможно нужно было сгенерировать еще больше данных. Вставка данных в таблицу person прошла успешно, что подтверждает корректность структуры таблицы и подключения к базе данных. Средний, минимальный и максимальный возраст были рассчитаны с помощью SQL-запросов. Это позволило понять распределение возраста в данных, аудитория по результатам получилась зрелая. Для выполнения SQL-запросов были использованы агрегатные функции (AVG, MIN, MAX), группировка (GROUP BY), сортировка (ORDER BY), работа со строками, датами. Задачи в Airflow, такие как загрузка новых сгенерированных данных через Faker в PostgreSQl, проверка новых данных в PostgreSQL, их извлечение, трансформация и загрузка в ClickHouse, были выполнены последовательно. PostgreSQL использовался как источник данных для анализа возраста, города, регистрации. ClickHouse использовался для анализа количества людей по городам. Airflow оркестрировал процесс ETL, обеспечивая автоматизацию и мониторинг. По итогу цель и задачи были выполнены.