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

**Лабораторная работа 4.1**

**Работа с XML и JSON файлами**

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

Преподаватель: Босенко Т.М.

Москва

2025

**Цель работы:** освоить анализ данных с помощью DASK.

**Задачи:**

- Настроить среду и рабочий каталог.

- Загрузить данные.

- Проверить качество данных (например, отсутствующие значения и выбросы).

- Удалить столбцы (множество пропусков в значениях, бесполезные столбцы для анализа).

- Визуализировать DAG с одним узлом и зависимостями.

- Визуализировать DAG с более чем одним узлом и зависимостями.

**Ход работы:**

**Вариант 12.** **Austin, TX House Listings.zip.**

Для начала были выполнены настройки среды и рабочего каталога, просмотрено содержимое рабочего каталога на рисунках 1–2.

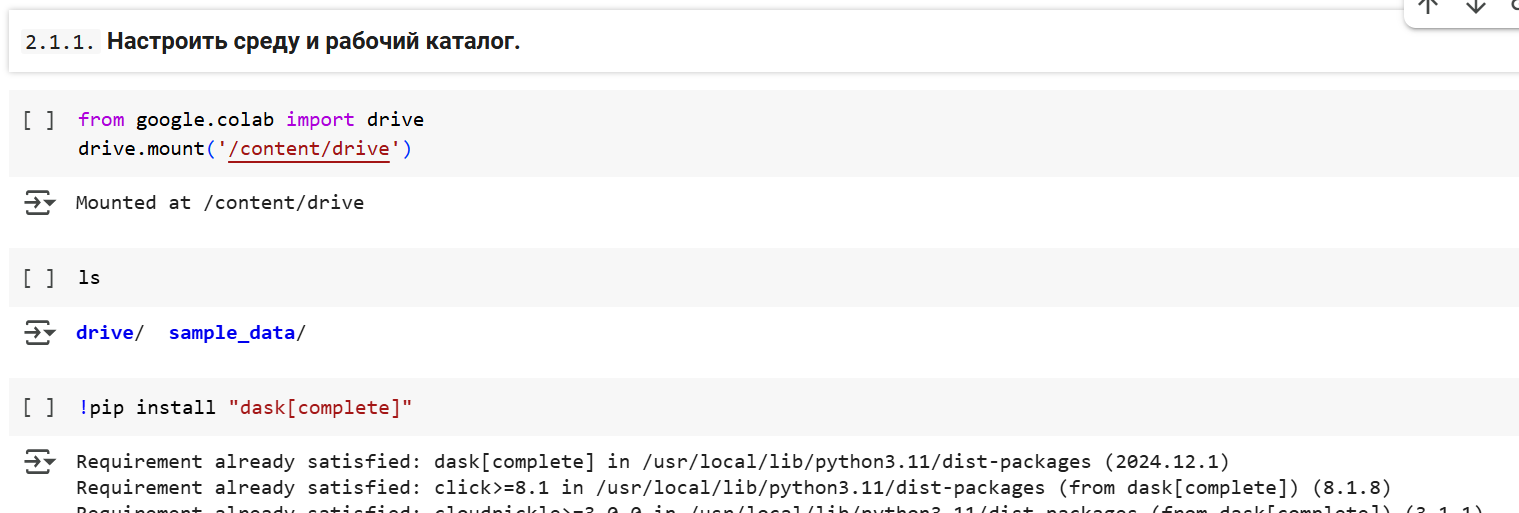


Рисунок 1 – Настройка рабочего каталога

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2 – Настройка среды и импорт библиотек

После данных шагов файл с расширением .zip был загружен в Google диск, для работы с ним необходимо было разархивировать его на рисунке 3.

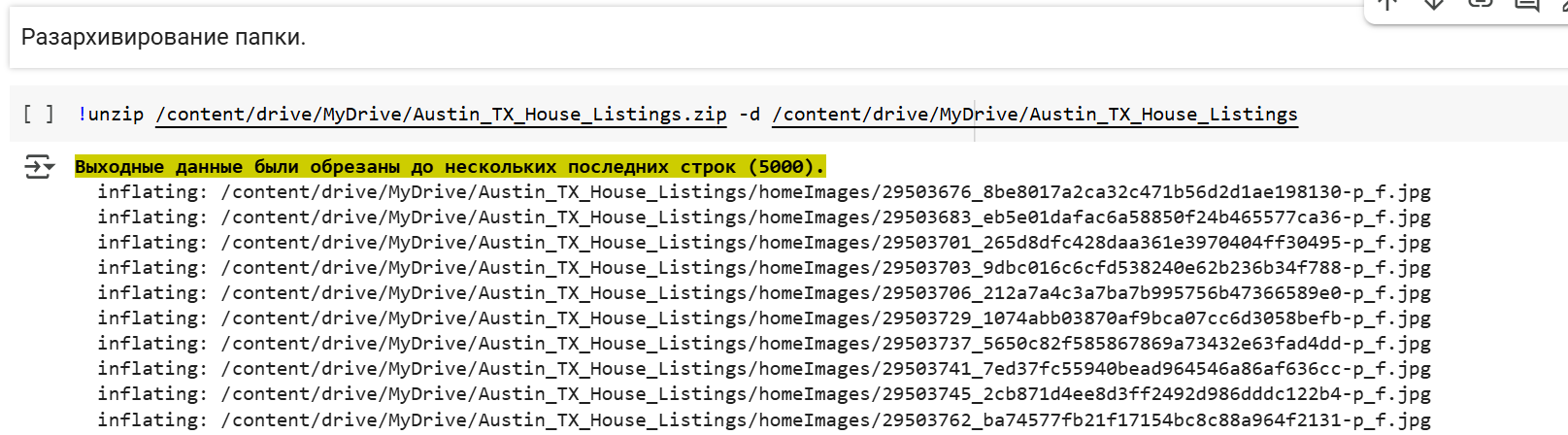


Рисунок 3 – Разархивирование папки

Далее файл был загружен и выведены первые 5 строк для просмотра загруженных данных, как видно на рисунке 4. Таблица содержит информацию о домах, также есть в папке папка с фотографиями домов.

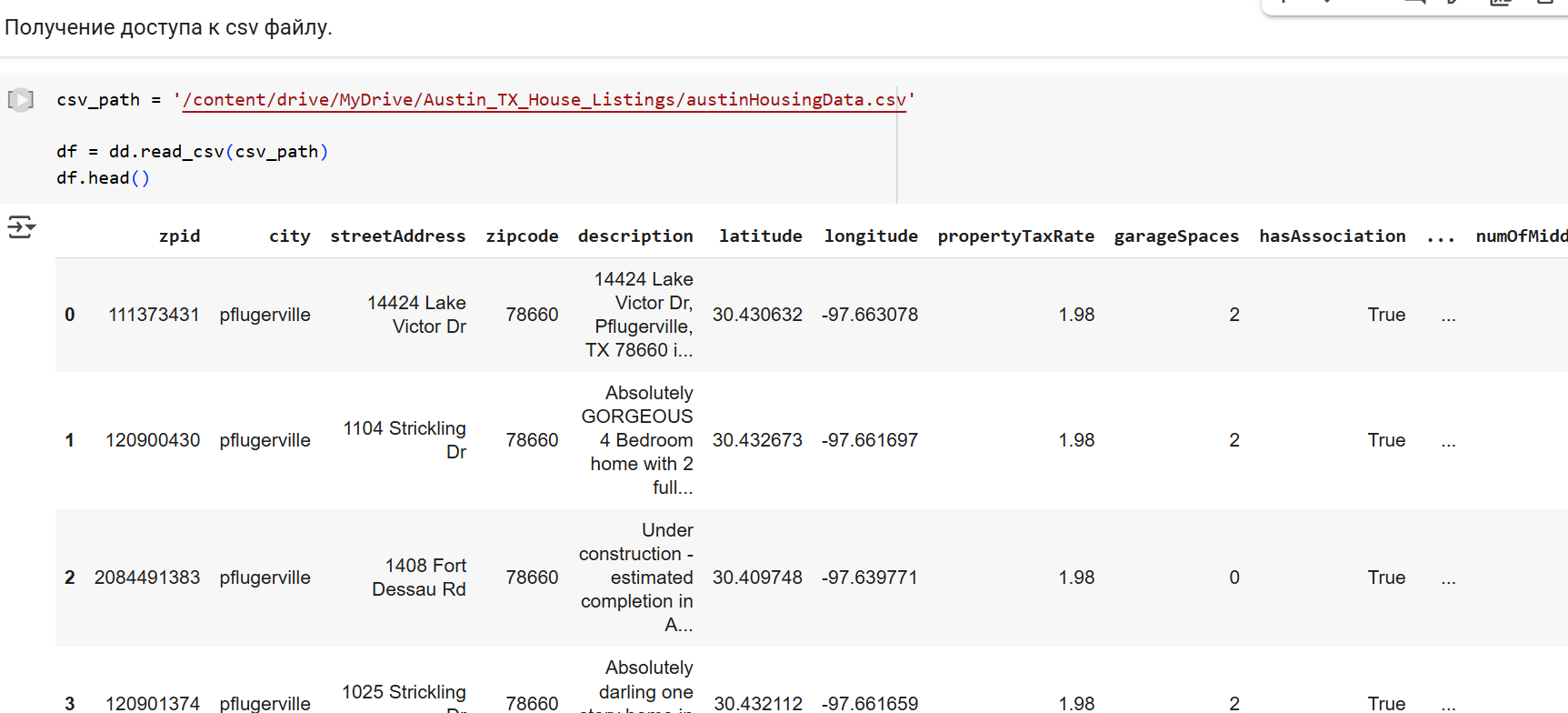


Рисунок 4 – Получение доступа к CSV файлу

Для получения списка изображений было обращение к папке и получение списка файлов на рисунке 5.

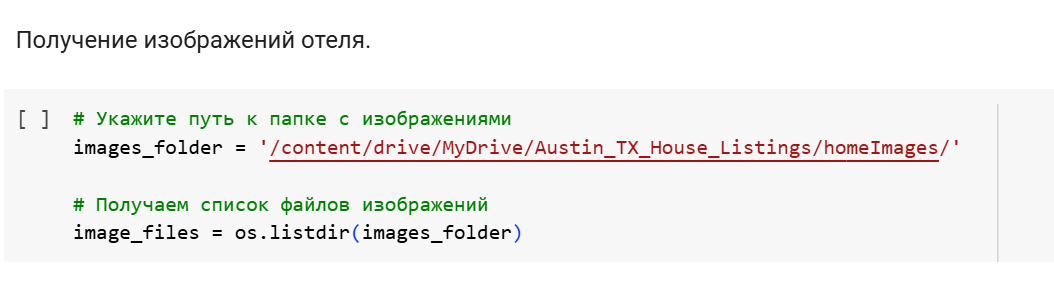


Рисунок 5 – Получение списка изображений отеля

Далее проводилась проверка пропущенных значений, расчёт процента пропущенных значений на рисунках 6–8, пропущенные значения были выявлены только в столбце с описанием дома, что естественно, ведь это дополнительное поле и необязательно.

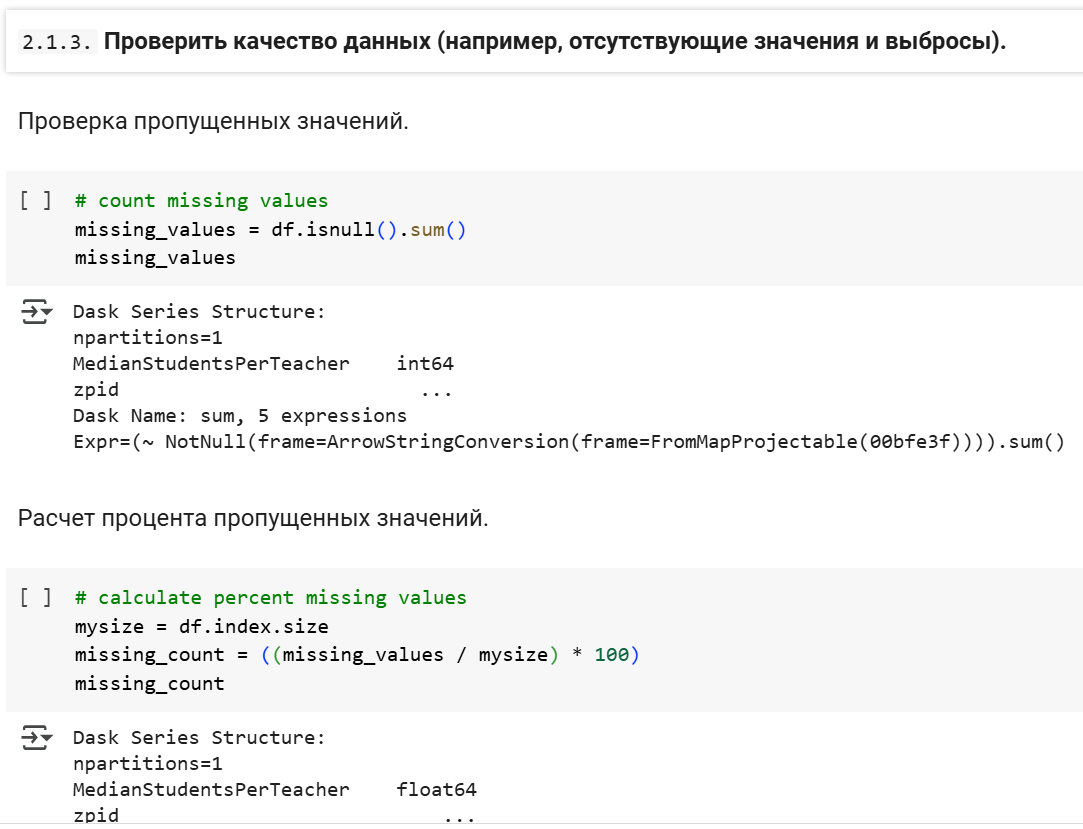


Рисунок 6 – Расчет пропущенных значений

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7 – Запуск вычислений

Изображение выглядит как текст, Шрифт, линия, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8 - Удаление строк с пропущенными значениями

Затем были изучены типы данных для выявления некорректных на рисунке 9.

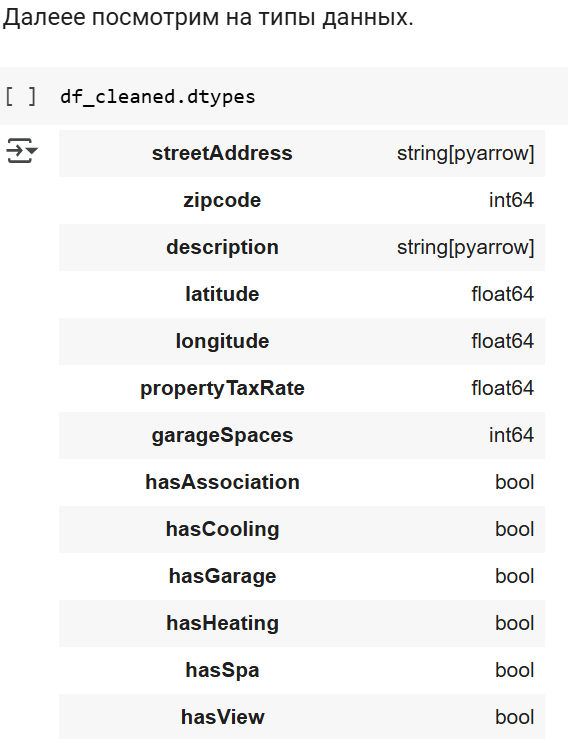


Рисунок 9 – Просмотр типов данных

Описание полей: zpid (int64): уникальный идентификатор конкретного объекта недвижимости, city (string[pyarrow]): название города, в котором находится объект недвижимости, streetaddress (string[pyarrow]): полный адрес объекта недвижимости, zipcode (int64): почтовый индекс, description (string[pyarrow]): описание объекта недвижимости, latitude (float64): географическая широта расположения объекта недвижимости, longitude (float64): географическая долгота расположения объекта недвижимости, propertytaxrate (float64): ставка налога на недвижимость, применяемая к объекту, garagespaces (int64): количество мест для парковки автомобилей в гараже, hasassociation (bool): логическое значение, указывающее на наличие ассоциации собственников жилья, hascooling (bool): указывает, есть ли у недвижимости система кондиционирования воздуха, hasgarage (bool): указывает, есть ли у объекта недвижимости гараж, hasheating (bool): указывает, есть ли у недвижимости система отопления, hasspa (bool): указывает, есть ли у объекта недвижимости спа, hasview (bool): указывает, есть ли у недвижимости вид, hometype (string[pyarrow]): тип недвижимости, parkingspaces (int64): общее количество мест для парковки, доступных для объекта недвижимости, yearbuilt (int64): год постройки объекта недвижимости, latestprice (float64): последняя известная цена объекта недвижимости, numpricechanges (int64): количество изменений цен для данного объекта недвижимости, latest\_saledate (string[pyarrow]): дата последней продажи объекта недвижимости, latest\_salemonth (int64): месяц последней продажи объекта недвижимости, latest\_saleyear (int64): год последней продажи объекта недвижимости, latestpricesource (string[pyarrow]): источник последней известной цены объекта недвижимости, numofphotos (int64): количество доступных фотографий для объекта недвижимости, numofaccessibilityfeatures (int64): количество доступных характеристик для людей с ограниченными возможностями, numofappliances (int64): количество бытовых приборов, доступных в объекте, numofparkingfeatures (int64): количество функций, связанных с парковкой, известных для объекта недвижимости, numofpatioandporchfeatures (int64): количество характеристик патио и веранд, присущих объекту недвижимости, numofsecurityfeatures (int64): количество функций безопасности, доступных для данного объекта, numofwaterfrontfeatures (int64): количество характеристик, связанных с водоемами, numofwindowfeatures (int64): количество характеристик окон, numofcommunityfeatures (int64): количество характеристик сообщества, связанных с объектом. lotsizesqft (float64): размер земельного участка в квадратных футах, livingareasqft (float64): площадь жилого помещения в квадратных футах, numofprimaryschools (int64): количество начальных школ в радиусе действия от объекта недвижимости, numofelementaryschools (int64): количество начальных школ в радиусе действия от объекта недвижимости, numofmiddleschools (int64): количество средних школ в радиусе действия от объекта недвижимости, numofhighschools (int64): количество старших школ в радиусе действия от объекта недвижимости, avgschooldistance (float64): среднее расстояние до ближайшей школы в милях или километрах, avgschoolrating (float64): средний рейтинг школ в окрестностях объекта недвижимости, avgschoolsize (int64): среднее количество учащихся в школах в окрестностях, medianstudentsperteacher (int64): среднее количество учащихся на одного учителя в окрестных школах, numofbathrooms (float64): общее количество ванных комнат в объекте недвижимости, numofbedrooms (float64): общее количество спален в объекте недвижимости, numofstories (int64): количество этажей в здании, homeimage (string[pyarrow]): путь к изображению дома.

По итогу необходимо только поменять поля с датами на рисунке 10.

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10 – Изменение типов данных

Далее искались дубли на рисунке 11.

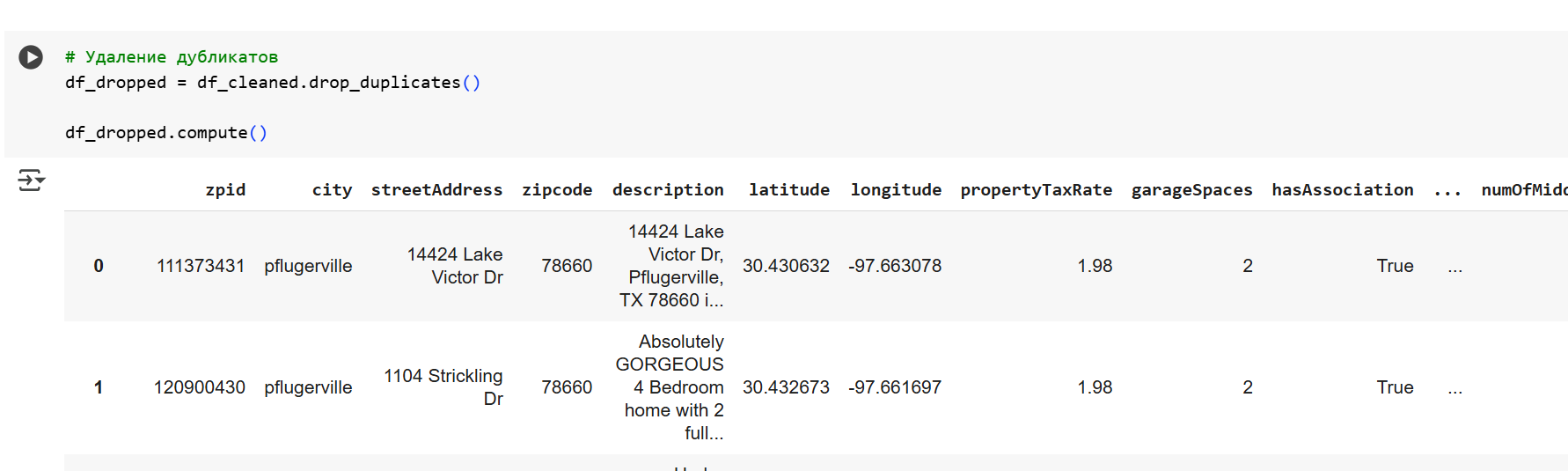


Рисунок 11 – Удаление дублей

В таблице был создан новый столбец, который хранит пути к фотографиям домов, для этого было выполнено объединение на основе названия фото на рисунке 12. Если название находится в списке изображений, то фиксируется путь к изображению, если нет, то пишется None.

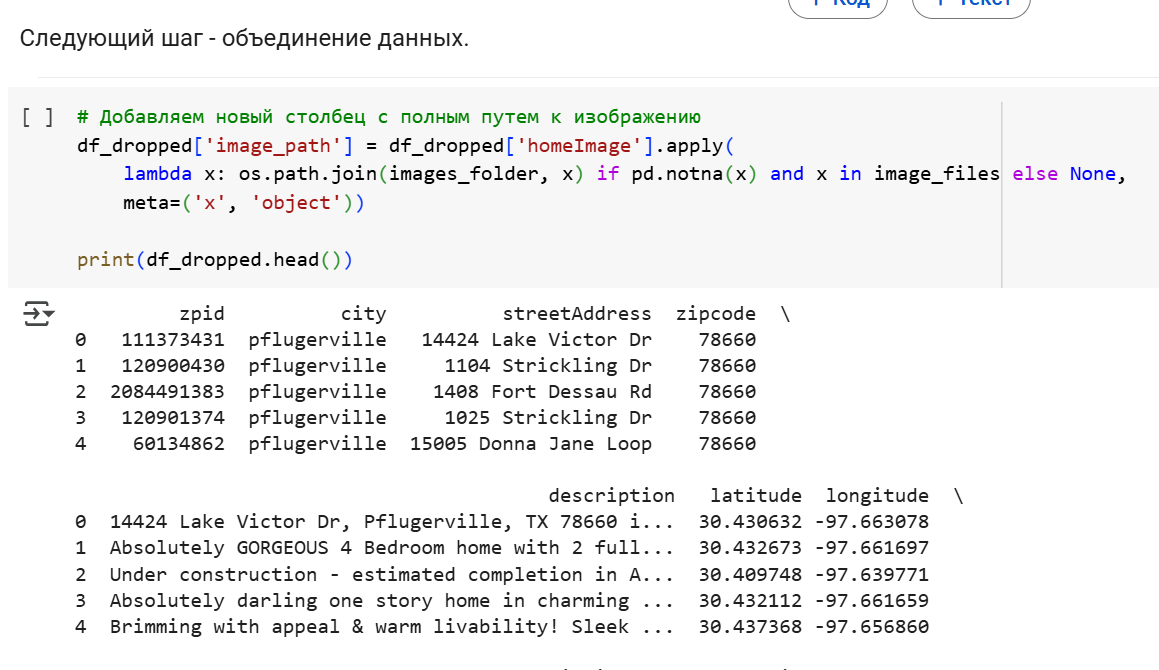


Рисунок 12 – Объединение таблиц

Чтобы посмотреть загруженные изображения, можно прибегнуть к библиотеке Pillow на рисунке 13.



Рисунок 13 – Отображение изображений

Далее был провизуализирован DAG с одним узлом и зависимостями и несколькими узлами. DAG — это ориентированный ациклический граф, концептуальное представление серии действий. Визуализация была выполнена 2 способами: с помощью visualize(), которая даёт более простой синтаксис, не фиксирует промежуточные результаты, при этом visualize() подходит для визуализации нескольких объектов одновременно и показывает промежуточные результаты, которые из-за ленивых вычислений равны 0. Данные даги показаны на рисунках 14-16.

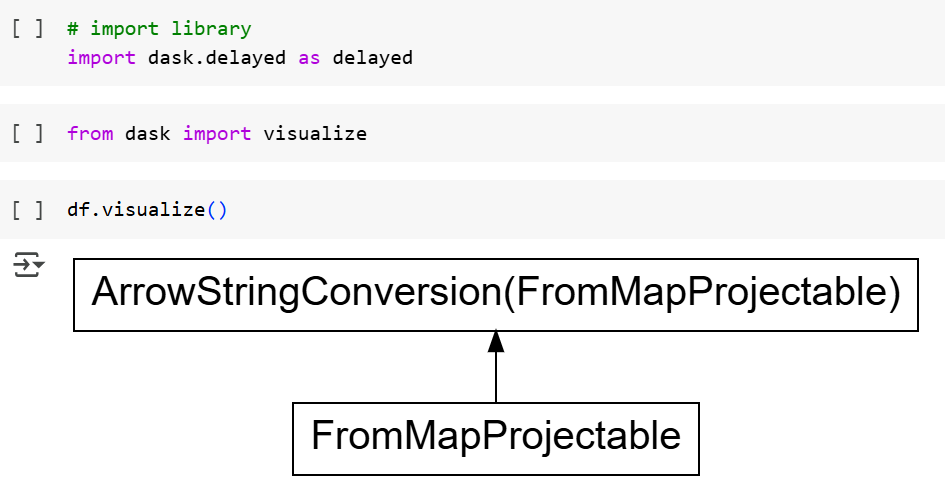


Рисунок 14 – Визуализация DAG без промежуточных результатов для одного узла

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 15 - Визуализация DAG с промежуточными результатами для одного узла

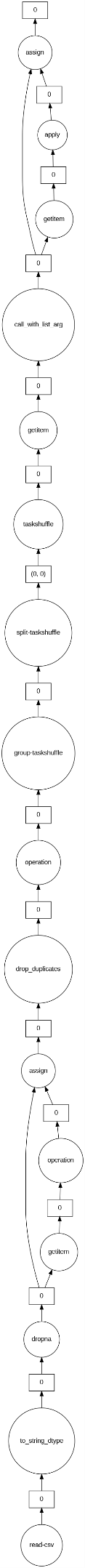


Рисунок 16 – Визуализация DAG с несколькими узлами и зависимостями

Для первого пункта элементы ArrowStringConversion и FromMapProjectable на изображении описывают процесс извлечения и преобразования данных. Для второго файл загружается с помощью read\_csv, удаляются нулевые значения, операции присваивания значения и другие операции представлены на схеме. Граф завершается операциями присваивания (assign) и финальными вычислениями. Нули или пустые значения в графе — это просто часть представления операций, и они не влияют на конечный результат.

**Общий вывод:**

Была изучена работа с большими данными через Google диск, изучена библиотека Dask, которая создана для параллельных и распределённых вычислений в Python, предназначена для работы с большими объёмами данных. Dask отлично интегрируется с другими библиотеками Python, такими как NumPy, Pandas и Scikit-learn, что облегчает переход с существующих инструментов на Dask. А также операции выполняются при необходимости c использованием compute(), что удобно. В рамках лабораторной работы были проанализированы данные домов, объединены с фотографиями, которые в дальнейшем можно отображать, а также построены DAGs.