**Практическое задание   
к промежуточной аттестации №2**

Курс «Архитектор данных»

ЦДО МГТУ им Н.Э.Баумана

Студент: Агеева О.О.

**Цель работы**: разработка и реализация технического проекта для возможного решения предложенного кейса (<https://www.kaggle.com/datasets/tforsyth/99bikes-sales-data>, далее по тексту ИД).

В практической части необходимо построить пайплайн ETL процесса, который будет включать в себя работу с ИД, а так же данными, полученными в результате парсинга любого сайта.

Загрузка данных в операционный слой должна осуществляться с минимальной обработкой.

**1. Описание кейса**

ИД содержат информацию о продажах одного из крупнейших австралийских ритейлеров велосипедов. В ассортименте ритейлера, кроме велосипедов, представлены различные аксессуары для них, включая одежду.

В данных содержится информация:

- покупатели;

- товары;

- транзакции.

Наименования столбцов перечислены в Приложении 1.

ИД представлены одним файлом в формате excel содержащем 4 таблицы, каждая на отдельном листе.

**2. ERD**

ИД поступают в виде 4 таблиц в одном файле формата excel, со следующим набором атрибутов:

-Transactions;

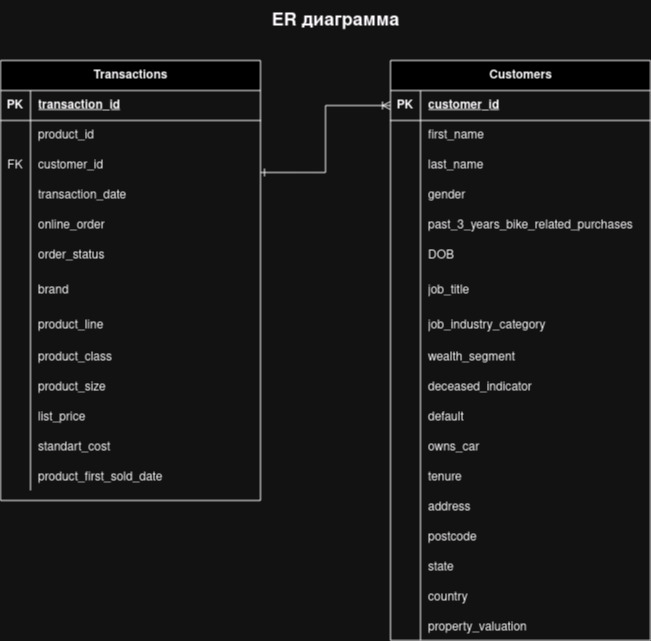
-CustomerAddress;

-CustomerDemografic;

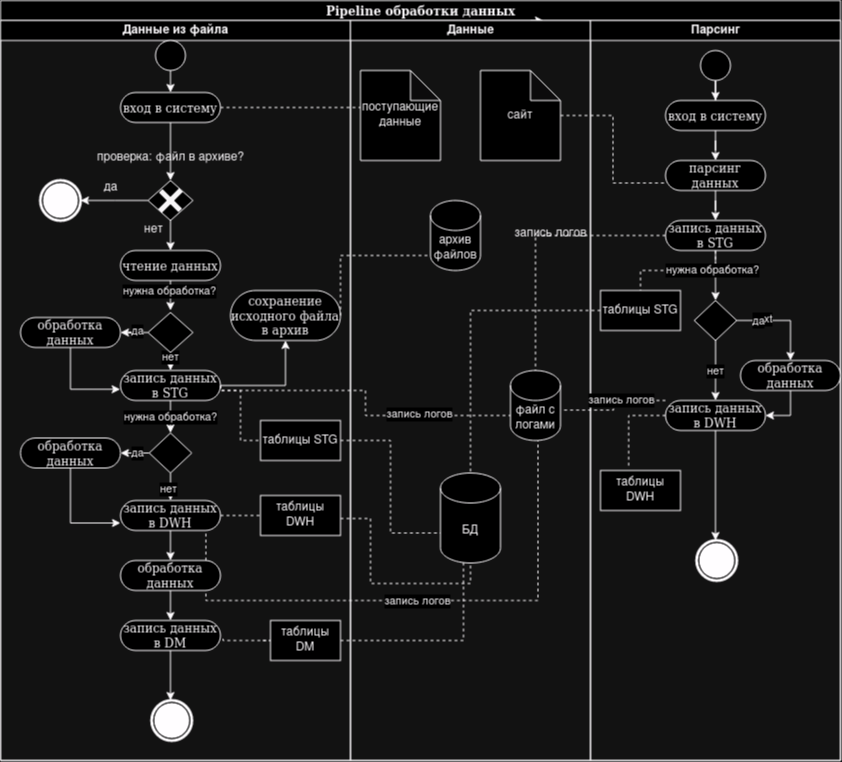
-NewCustomerList.

4-тая таблица файла «NewCustomerList» отражает расчетный рейтинг покупателей. По данной таблицы не будут приходить дополнения или обновления, таблица является аналитической. Исходя из вышеизложенного, данная таблица не подлежит загрузке в базу данных.

Выделим сущности: «Транзакция» и «Пользователь».



**3. Бизнес-процесс**



**4. Описание архитектуры**

База данных - SQLite

Компоненты: Python, включая библиотеки: os, shutil, logging, pandas, Beautifulsoup

SQL

Docker

Airflow

Архитектура проекта заключается в последовательной обработке поступающего файла в формате excel с последующей записью данных в базу (SQLite).

Весь код обработки данных написан на Python и SQL. В источнике не указано, как часто поступают данные для обработки, предположим, что файл поступает раз в сутки, также раз в сутки запускается скрипт. Задача скрипта - добавление в базу новых данных из полученного нового файла и его архивация.

База данных состоит из двух уровней:

- стейджинг, на который подгружаются приходящие данные

- основной, где хранятся данные

**5. Выбор СУБД**

Табличную структуру, тем более небольшую, как исходные данные, удобно хранить в реляционной базе. Выбор пал на SQLite, как удобную СУБД для работы с небольшими объемами данных. Она работает без сервера, при необходимости легко встраивается в приложение. Для нее существуют дополнительные библиотеки с открытым кодом. По производительности – выдерживает 200 запросов на страницу.

**6. Решение по используемой СХД**

СХД - устройство для хранения и управления данными и для их резервного копирования. Исходные данные приходят в файлах. Так как согласно данным, за год всего 20000 транзакций, то промежуточные файлы можно хранить и в файловой СХД. Преобразованные данные будут хранится в базе данных, которая будет размещаться на блочной СХД.

**7. Перечисление алгоритмов и методов анализа и обработки, предлагаемых к использованию в решении кейса.**

Парсинг, обработка поступающих данных, запись и обновление данных в БД происходит с помощью DAG в Airflow.

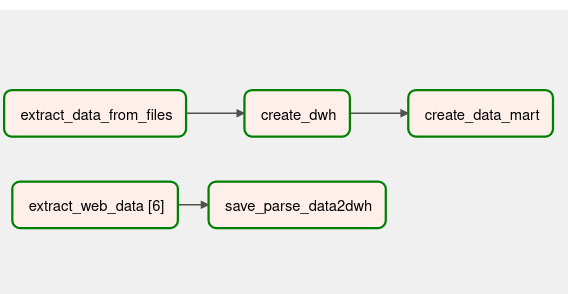
**8. Описание модели угроз**

Перечень возможных угроз:

1. Неограниченные привилегии
2. Внедрение sql – кода
3. Непрофессионализм сотрудников
4. Открытие копии БД

**Описание реализации.**

До создания pipeline необходимо изучить данные при помощи Pandas и прийти к решению, какие преобразования будут необходимы.



**DAG Airflow**

1. **Выгрузка данных из файла в операционный слой.**

Столбец default в таблице Customer\_demografic необходимо переименовать до загрузки в операционный слой, так как это зарезервированное в БД слово. Надо отметить, что в столбце содержатся непонятные данные, поэтому для витрины мы их использовать не будем.

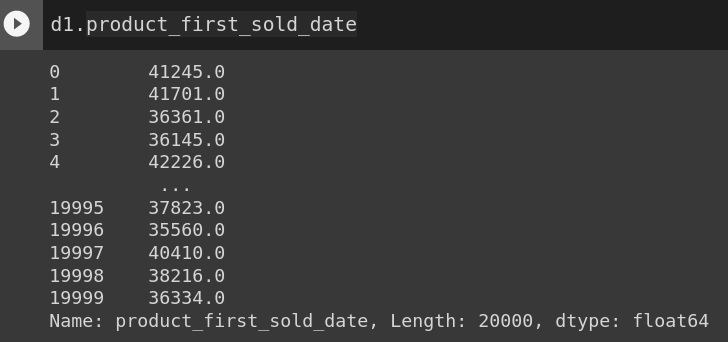


Так как данные поступают в одном файле, то в цикле считываем каждый лист.

После выгрузки файл отправляется в архив.

2. **Обработка и загрузка данных в основной слой.**

В таблице Transactions присутствует столбец first\_sold\_date, у него необходимо изменить формат данных до загрузки в основной слой, так как SQlite не поддерживает перевод данных из такого формата (дни от 01-01-1900)

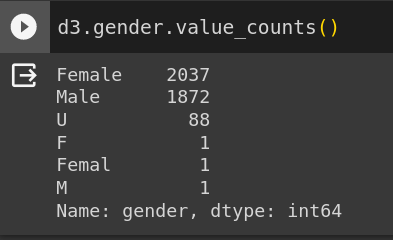


Остальные даты из таблиц тоже переведем в формат pandas datetime, так как SQLite при сохранении таблиц переводит данные из timestamp в текстовые.

**3. Создание витрины данных.**

В таблице Customer\_demografic присутствует столбец DOB, в нем много пропусков, а он нужен для витрин данных. Поэтому нужно заполнить столбец как вариант самым частым значением.

Для витрины необходимо преобразовать данные в столбце Gender, так как он содержит неоднородные значения.



В витрине будет удобнее работать с возрастом, а не с датой рождения. Поэтому нужно создать дополнительный столбец для возраста.

Ненужные столбцы (default, страна так как она одна, дата рождения) удалить.

**4. Парсинг данных.**

Было принято решение для обогащения данных сделать парсинг сайта https://www.velostrana.ru, где продаются товары околовелосипедной области. Для работы были взяты основные разделы. Для парсинга использовалась библиотека Beautiful Soup.

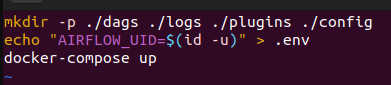
Данные были помещены в операционный слой по принципу: каждый товар - отдельная таблица.

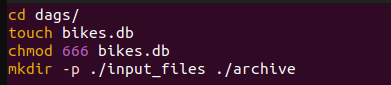
**5. Сохранение данных сайта в DWH.**

Так как данные чистые, то в основной слой они загружены без предобработки в одну общую таблицу следующего вида:

Для контроля ошибок реализовано логирование. Возможные ошибки: потеря связи, отсутствие нужного файла, другое наименование страниц в файле отработаны.

Для запуска программы написан скрипт.



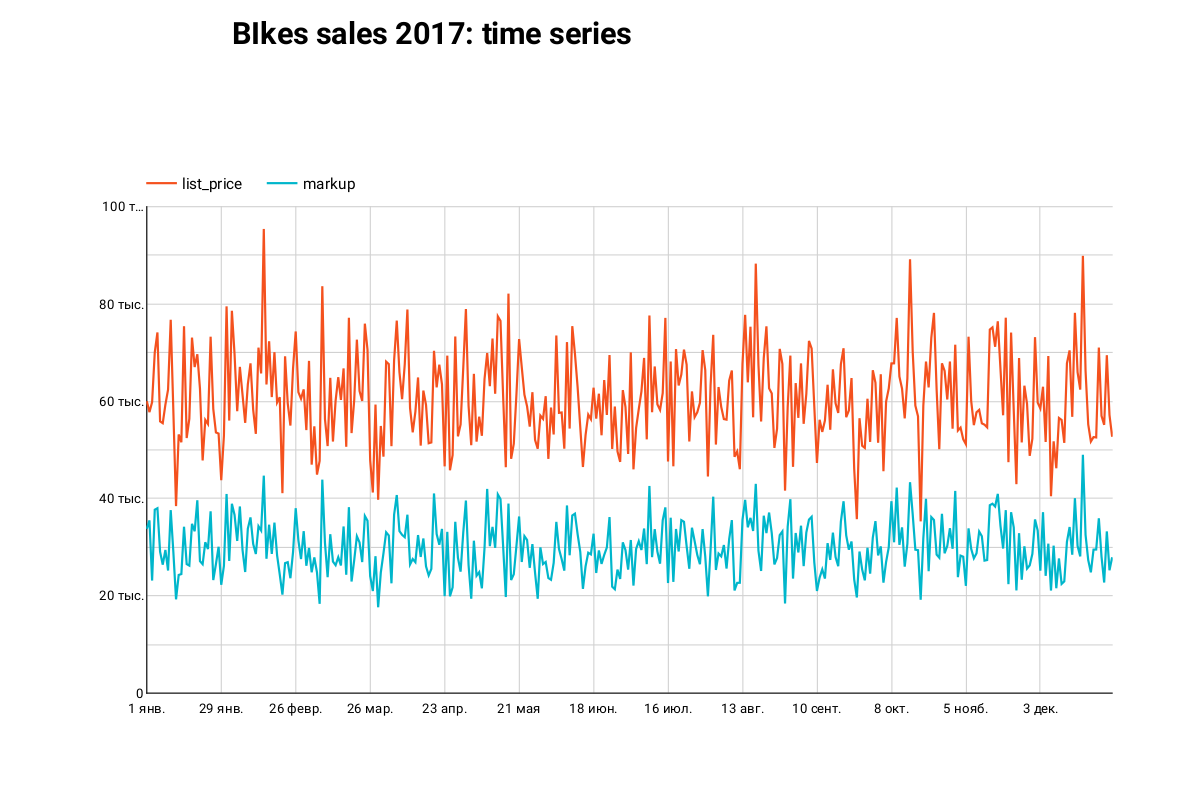
И скрипт для создания базы и доступа к ней.

**Визуализация данных.**

Для построения дашбордов использовался Looker Data Studio. Он удобен для работы с небольшими данными, достаточно быстрый, в нем много коннекторов. На сегодня это полноценная бесплатная BI система.

Из имеющихся данных можно предположить, что standard cost это закупочная цена, а list proce - цена продажи. Введём ещё один столбец markup - наценка.

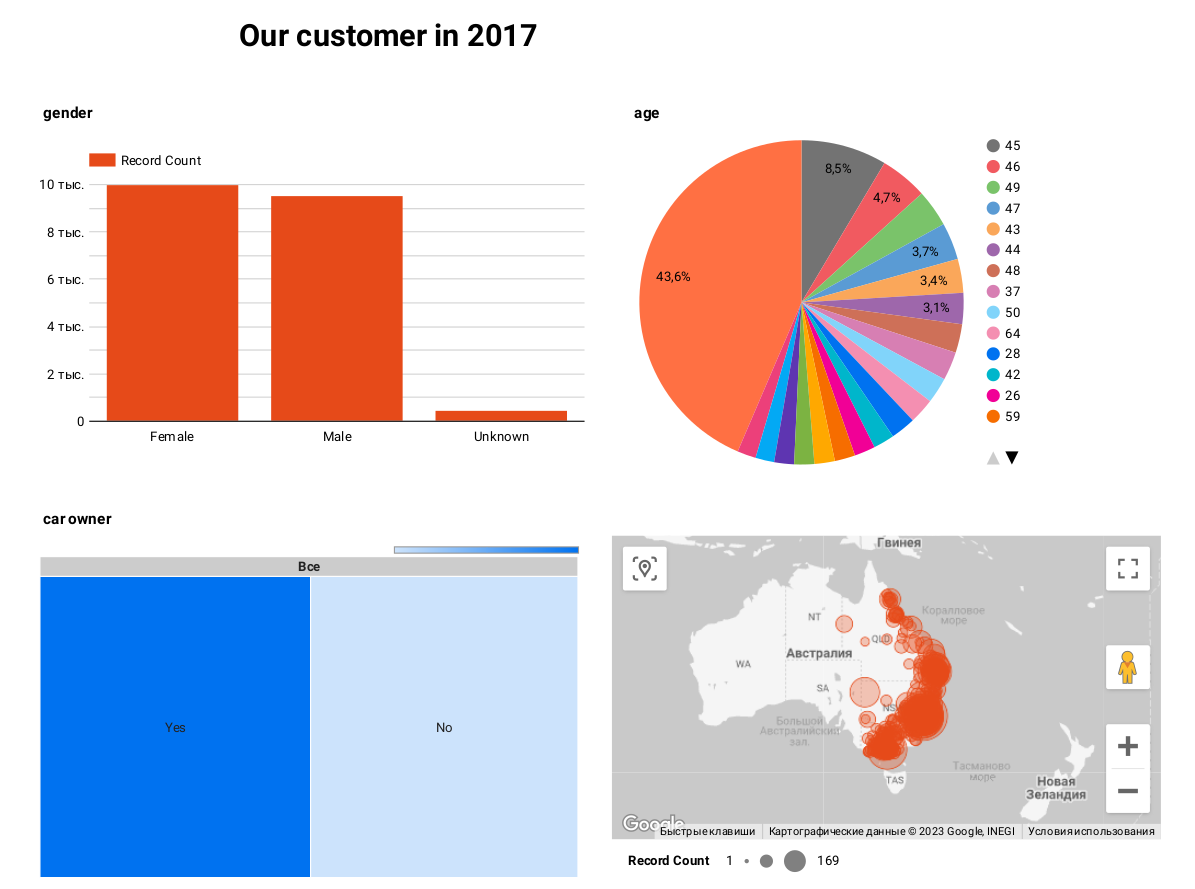
На дашборде динамика продаж за год и динамика наценки.



На следующем дашборде продажи и наценка в разрезе товарных групп.



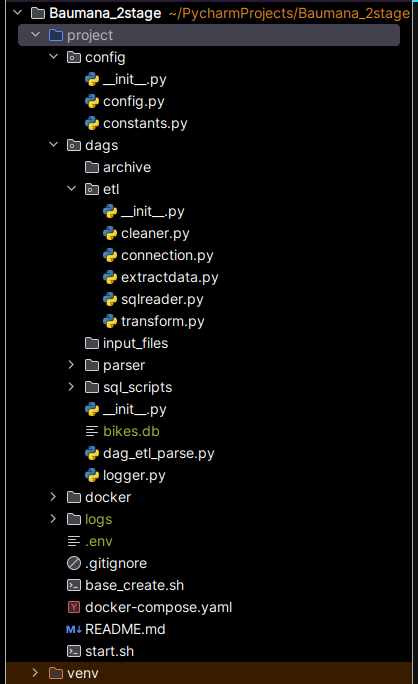
Следующий дашборд - портрет покупателя компании.



**Реализация пайплайна:**

Проект находится на:

https://github.com/OlgaTora/BikesAirflowETL-parse



Структура проекта

**Выводы**: в данном кейсе не хватило описательной информации. Особенно по продукту: наличие двойных id и нескольких вариантов дат первой продажи. Также необходимо разобраться, что за таблица с рейтингом покупателей.

При масштабировании проекта можно выделить дополнительные сущности из Покупателя и Продукта.

Приложение 1. Названия колонок в исходных данных

**Customer Address**

customer\_id

address

postcode

state

country

property\_valuation

**Transactions**

transaction\_id

product\_id

customer\_id

transaction\_date

online\_order

order\_status

brand

product\_line

product\_class

product\_size

list\_price

standart\_cost

product\_first\_sold\_date

**NewCustomerList**

first\_name

last\_name

gender

past\_3\_years\_byke\_related\_purchases

DOB

job\_title

job\_industry\_category

welth\_segment

deceased\_indicator

owns\_car

tenure

address

postcode

state

country

property\_valuation

Rank

Value

**Customer\_demographic**

customer\_id

first\_name

last\_name

gender

past\_3\_years\_byke\_related\_purchases

DOB

job\_title

job\_industry\_category

welth\_segment

deceased\_indicator

default

owns\_car

tenure