# Практическое задание к промежуточной аттестации №1

**Цель работы:** разработка и реализация технического проекта для возможного решения предложенного кейса (<https://www.kaggle.com/datasets/tforsyth/99bikes-sales-data>). В практической части необходимо построить простой пайплайн на Python для сбора, обработки и хранения данных из предложенного кейса. Загрузка данных осуществляется в «сыром» виде без трансформации.

1. Описание кейса

Данные содержат информацию о продажах одного из крупнейших австралийских ретейлеров велосипедов. В их ассортименте, кроме велосипедов, присутствуют различные аксессуары для них, включая одежду.

В данных содержится подробная информация о покупателях: от пола и возраста и заканчивая делали ли они покупки в околовелосипедной области последние три года.

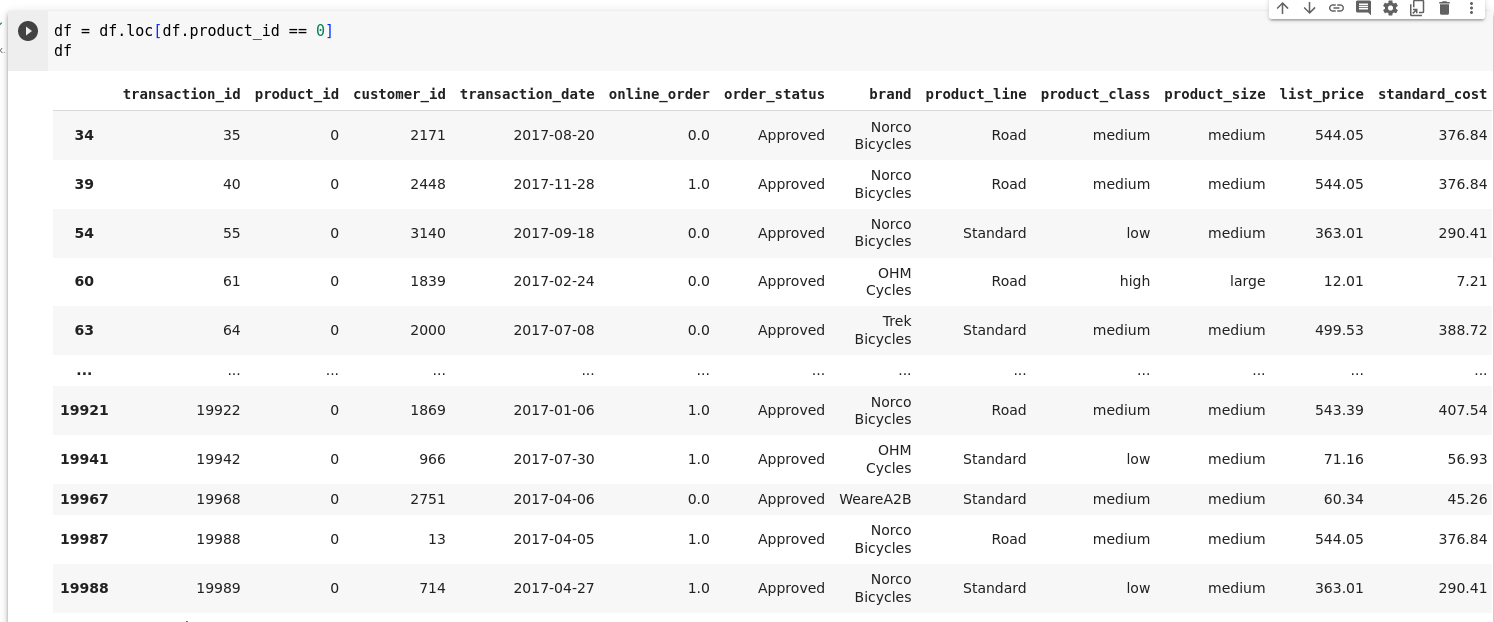
Данные поступают в виде одного файла excel с 4 таблицами, каждая на отдельном листе.

2. ERD

Исходные данные поступают в виде 4 таблиц: Transactions, Customer Address, Customer\_demografic, NewCustomerList. Последняя таблица похожа на расчетную по рейтингу покупателей, исходя из суммы их покупок.

Выделим сущности: транзакция и пользователь.

Продукт не выделить, так как с одним айди в таблице разные товары.

Другие сущности (как бренд, размер, класс продукта) тоже не выделить, так как по поставленной задаче данные в операционный слой должны поступать сырыми.

3. Бизнес-процесс – BPMN-нотация реализации вашего пайплайна

* 1. Исходные данные
  2. Блок трансформации
  3. Финальные данные \ таблицы

4. Описание архитектуры

База данных - SQLite

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

SQL

Данная архитектура 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 и в базу добавляются новые данные из нового файла.

База данных состоит из двух уровней:

- стейджинг, на который подгружаются приходящие данные

- основной

Отношения сущностей в базе представлены скорее в 1НФ, поскольку у разного набора данных один и тот же product\_id.

5. Выбор СУБД

Табличную структуру, тем более небольшую, как исходные данные, удобно хранить в реляционной базе. Выбор пал на SQLite

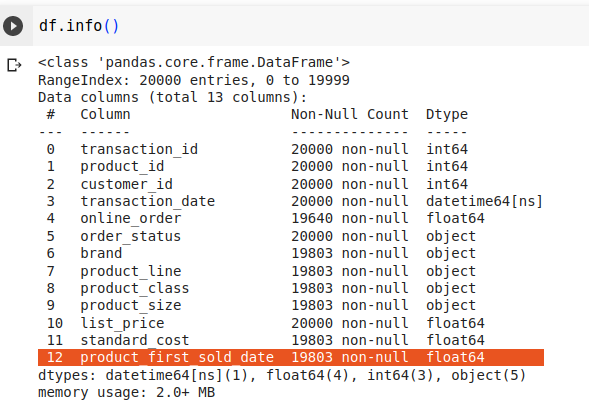
6. Решение по используемой СХД

Так как СХД это 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.

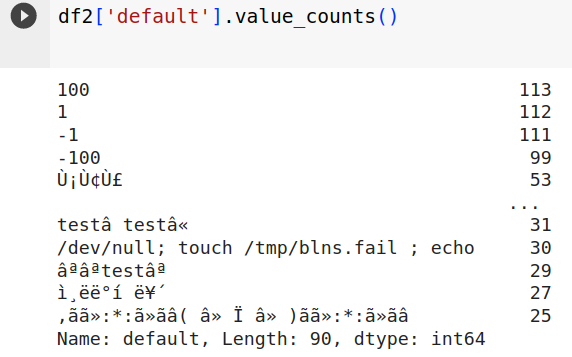
7. Перечисление алгоритмов и методов анализа и обработки, предлагаемых к использованию в решении кейса.

Для обработки поступающих данных используется Python.

Запись и обновление данных в БД происходит с помощью запросов SQL.



В процессе работы появилась необходимость изменить формат данных до загрузки в операционный слой, так как sqlite не поддерживает перевод данных из такого формата (дни от 01-01-1900)

Столбец default было необходимо переименовать до загрузки в операционный слой, так как это зарезервированное в БД слово.

8. Описание модели угроз

Перечень угроз:

1. Несанкционированный доступ в базу (расположена во внутреннем контуре с ограниченным доступом)
2. Права доступа к хранимой информации (ограниченный круг лиц, кто может взаимодействовать с этой информацией)
3. Защита от внешних атак (sqlinjection)

.

## Сущности:

**Transactions**

transaction\_id

product\_id

customer\_id

transaction\_date

online\_order

order\_status

**Product**

product\_id

brand

product\_line

product\_class

product\_size

list\_price

standard\_cost

product\_first\_sold\_date

**Customer**

customer\_id

first\_name

last\_name

gender

address

postcode

state

country

property\_valuation

past\_3\_years\_byke\_related\_purchases

DOB

job\_title

job\_industry\_category

welth\_segment

deceased\_indicator

default

owns\_car

tenure

Приложение 1.

**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

**Customer Address**

customer\_id

address

postcode

state

country

property\_valuation

**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

**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

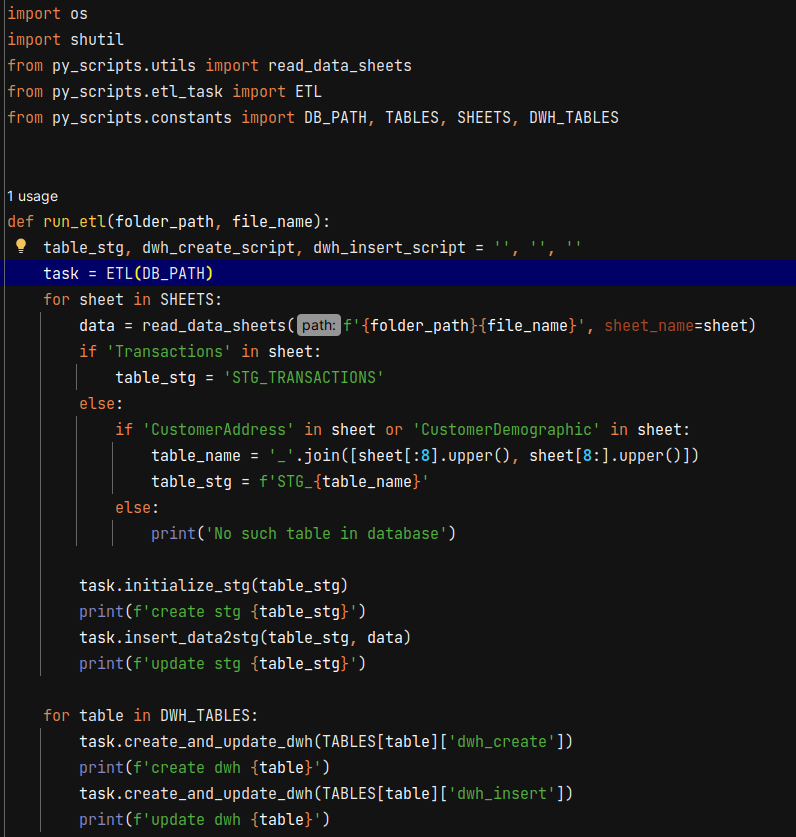
Value

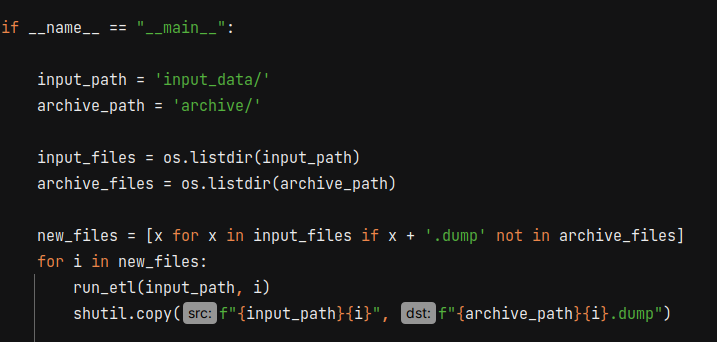
Реализация пайплайна:

Проект находится на:

https://github.com/OlgaTora/Bike-retai-ETL/

Скрин main.py:





Тестирование (скрины с БД)