Организация хранилища, разработка ETL-процессов и визуализация данных

Нетология, курс DataOps-инженер

Выполнил: Манохин И.В.

Цель дипломной работы:

Дипломная работа является комплексным проектом, включающим в себя анализ и обработку исходных данных, разработку и документирование хранилищ данных (NDS/DDS), создание и описание процессов ETL (Extract, Transform, Load), а также визуализацию полученных данных. Основная цель заключается в обеспечении эффективного хранения и управления данными, что способствует принятию обоснованных бизнес-решений.

Кроме основных этапов, дипломная работа также представляет возможность погрузиться в анализ данных и создание информационных систем. В процессе выполнения проекта необходимо не только освоить технические аспекты работы с данными, но и научиться видеть в них ценность для бизнеса.

Этапы выполнения дипломной работы

01

Изучение контекста датасета, его атрибутов

02

Статистический анализ, поиск ошибок

03

Формирование схем баз данных и создание ER-диаграмм

04

Разработка ETL-процессов

05

Визуализация данных. Построение дашбордов

06

Формирование выводов

Используемые инструменты и ПО

Анализ данных	Дистрибутив Anaconda (Jupiter Lab)
Разработка ER-диаграмм	Сервис dbdiagram.io
Базы данных	ClickHouse, PostgreSQL
Выполнение ETL-процессов	Apache Airflow
Визуализация данных	Tableau Desktop

Все окружение для работы над проектом было поднято локально с использованием контейнеризации (Docker).

01

Данные

Описание датасета

Датасет Supermarket Sales представляет из себя срез исторических данных о продажах товаров в 3 филиалах компании за 3 месяца (с января по март 2019 года).

Атрибуты датасета:

1. Invoice ID: программно-генерируемый идентификационный номер счета-фактуры
2. Branch: название филиала компании
3. City: местонахождение филиала (город)
4. Customer Type: тип покупателя (наличие клубной карты)
5. Gender: пол покупателя
6. Product Line: продуктовая линейка
7. Unit Price: цена единицы товара в долларах
8. Quantity: количество проданных товаров
9. Tax: сумма взимаемого налога с продажи (5%)
10. Total: общая стоимость продажи, включая налоги
11. Date: дата продажи
12. Time: время продажи
13. Payment: метод оплаты
14. COGS: себестоимость проданных товаров
15. Gross Profit Percentage: процент прибыли
16. Gross Revenue: прибыль с продажи
17. Rating: рейтинг покупки от покупателя (по шкале от 1 до 10)

02

Анализ

Получение статистики и поиск ошибок в данных

Для анализа исходных данных использовались библиотеки Pandas, Matplotlib и Seaborn.

Вывод количества строк, наличие Null-значений и типов данных:

```
[47]: # проверка типов данных
      df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Invoice ID               1000 non-null   object
 1   Branch                   1000 non-null   object
 2   City                     1000 non-null   object
 3   Customer type            1000 non-null   object
 4   Gender                   1000 non-null   object
 5   Product line             1000 non-null   object
 6   Unit price               1000 non-null   float64
 7   Quantity                 1000 non-null   int64
 8   Tax 5%                   1000 non-null   float64
 9   Total                    1000 non-null   float64
 10  Date                     1000 non-null   object
 11  Time                     1000 non-null   object
 12  Payment                  1000 non-null   object
 13  cogs                     1000 non-null   float64
 14  gross margin percentage  1000 non-null   float64
 15  gross income             1000 non-null   float64
 16  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(9)
memory usage: 132.9+ KB
```

Проверка данных на дубликаты и вывод статистики по всем столбцам:

```
[48]: # проверка строк DataFrame на дубликаты
      df.duplicated().any()
```

```
[48]: False
```

```
[49]: # воспользуемся методом Pandas .describe() для генерации статистики по всем столбцам

      df.describe(include='all')
```

Вывод списков категориальных переменных:

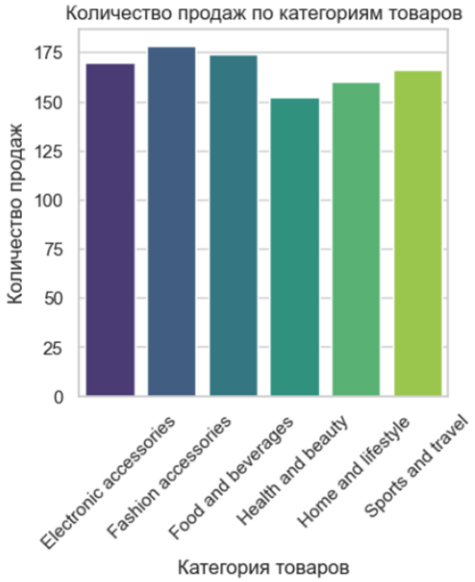
```
[50]: print(df['Branch'].unique())
      print(df['City'].unique())
      print(df['Customer type'].unique())
      print(df['Gender'].unique())
      print(df['Product line'].unique())
      print(df['Payment'].unique())
```

```
['A' 'C' 'B']
['Yangon' 'Naypyitaw' 'Mandalay']
['Member' 'Normal']
['Female' 'Male']
['Health and beauty' 'Electronic accessories' 'Home and lifestyle'
 'Sports and travel' 'Food and beverages' 'Fashion accessories']
['Ewallet' 'Cash' 'Credit card']
```

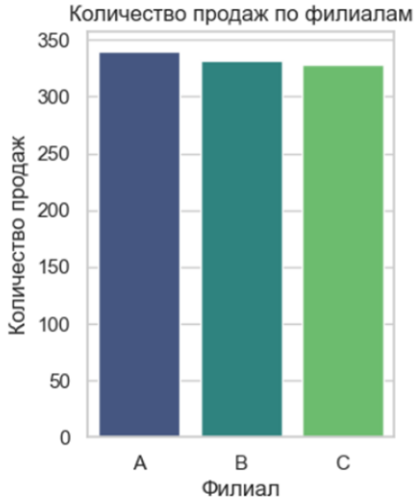
	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gross income	Rating
count	1000	1000	1000	1000	1000	1000	1000.000000	1000.000000	1000.000000	1000.000000	1000	1000	1000	1000.00000	1.000000e+03	1000.000000	1000.00000
unique	1000	3	3	2	2	6	NaN	NaN	NaN	NaN	89	506	3	NaN	NaN	NaN	NaN
top	750-67-8428	A	Yangon	Member	Female	Fashion accessories	NaN	NaN	NaN	NaN	2/7/2019	19:48	Ewallet	NaN	NaN	NaN	NaN
freq	1	340	340	501	501	178	NaN	NaN	NaN	NaN	20	7	345	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	NaN	NaN	NaN	55.672130	5.510000	15.379369	322.966749	NaN	NaN	NaN	307.58738	4.761905e+00	15.379369	6.97270
std	NaN	NaN	NaN	NaN	NaN	NaN	26.494628	2.923431	11.708825	245.885335	NaN	NaN	NaN	234.17651	6.131498e-14	11.708825	1.71858
min	NaN	NaN	NaN	NaN	NaN	NaN	10.080000	1.000000	0.508500	10.678500	NaN	NaN	NaN	10.17000	4.761905e+00	0.508500	4.00000
25%	NaN	NaN	NaN	NaN	NaN	NaN	32.875000	3.000000	5.924875	124.422375	NaN	NaN	NaN	118.49750	4.761905e+00	5.924875	5.50000
50%	NaN	NaN	NaN	NaN	NaN	NaN	55.230000	5.000000	12.088000	253.848000	NaN	NaN	NaN	241.76000	4.761905e+00	12.088000	7.00000
75%	NaN	NaN	NaN	NaN	NaN	NaN	77.935000	8.000000	22.445250	471.350250	NaN	NaN	NaN	448.90500	4.761905e+00	22.445250	8.50000
max	NaN	NaN	NaN	NaN	NaN	NaN	99.960000	10.000000	49.650000	1042.650000	NaN	NaN	NaN	993.00000	4.761905e+00	49.650000	10.00000

Информация полученная из describe:

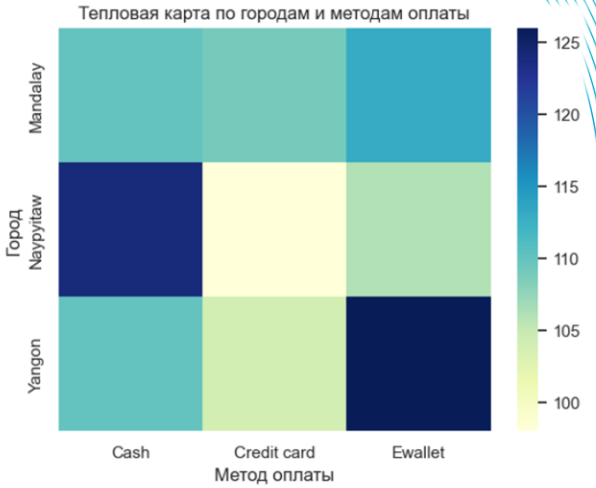
- В данных почти равное распределение как по типу клиента (Member/Normal), так и по полу (Female/Male). В обоих случаях это 501 на 499.
- Наибольшей популярностью пользуется категория Fashion accessories. На нее приходится 178 продаж.
- Средняя стоимость единицы товара $55.67.
- Среднее количество товаров в покупке 5.51.
- Средний налог с продажи $15.38.
- Средний доход с продажи $322.97.
- Средняя прибыль с продажи $15.38.
- Средний рейтинг покупки 6.97 балов.



** Самая популярная категория Fashion Accessories, наименее популярная Health and beauty.*



** Больше всего продаж приходится на филиал A, но в целом распределение почти равное.*

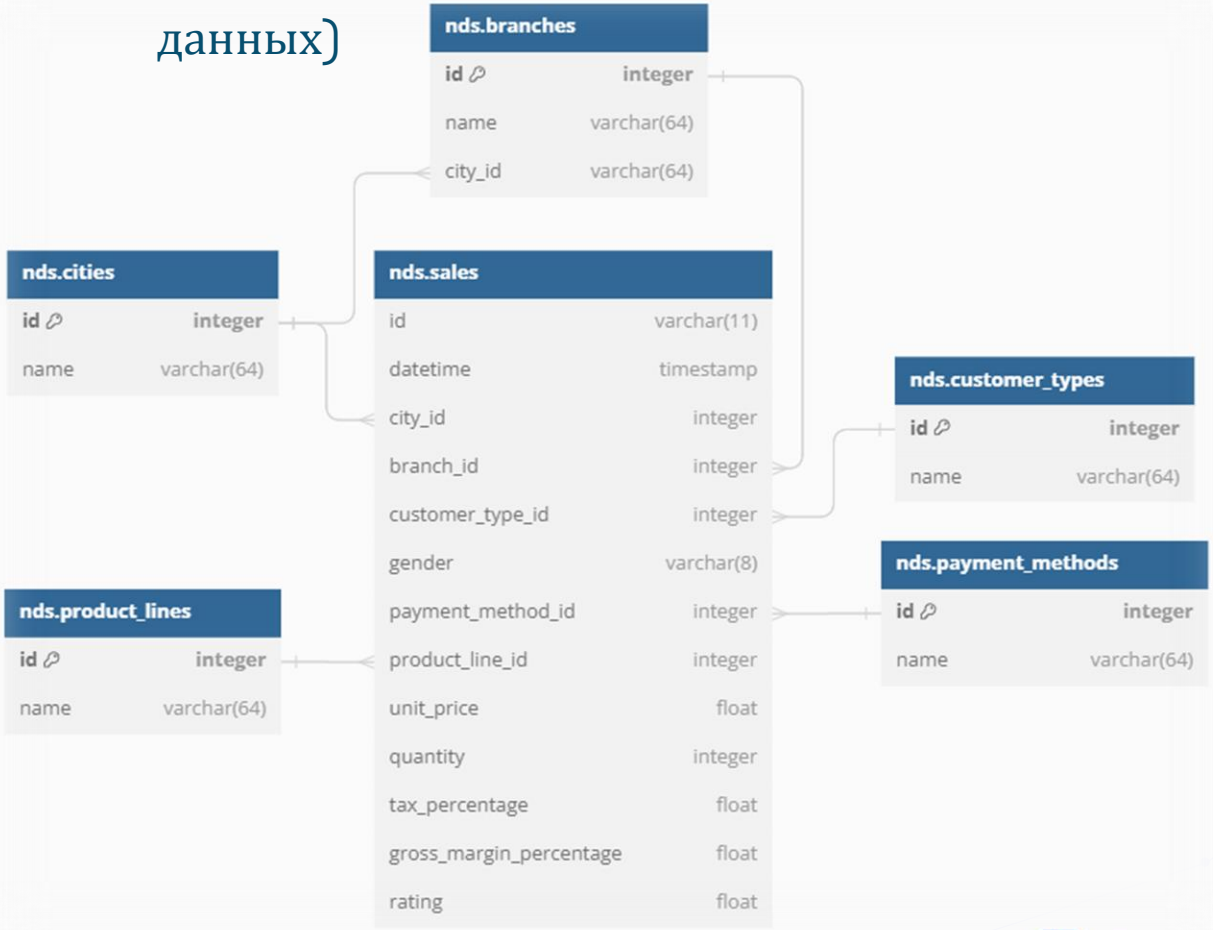


** Метод оплаты Cash чаще всего используют в городе Naypyitaw*
** Метод оплаты Credit Card наиболее часто используют в городе Mandalay*
** Метод оплаты Ewallet наиболее часто используют в городе Yangon*

03

Схемы данных

NDS (нормализованная схема данных)



Нормализованная схема включает в себя следующие таблицы:

1. Города (nds.cities):
    - Таблица для хранения названий городов.
2. Филиалы (nds.branches):
    - Таблица для хранения названий филиалов, зависит от таблицы nds.cities.
3. Продуктовые линейки (nds.product_lines):
    - Таблица для хранения категорий товаров.
4. Способы оплаты (nds.payment_methods):
    - Таблица для хранения доступных методов оплаты.
5. Типы клиентов (nds.customer_types):
    - Таблица для хранения различных типов клиентов.
6. Продажи (nds.sales):
    - Основная таблица для хранения информации о продажах. Отдельные столбцы для даты, времени, цены, количества и других характеристик продажи. Внешние ключи связывают записи в этой таблице с соответствующими записями в таблицах городов, филиалов, категорий товаров, способов оплаты и типов клиентов.

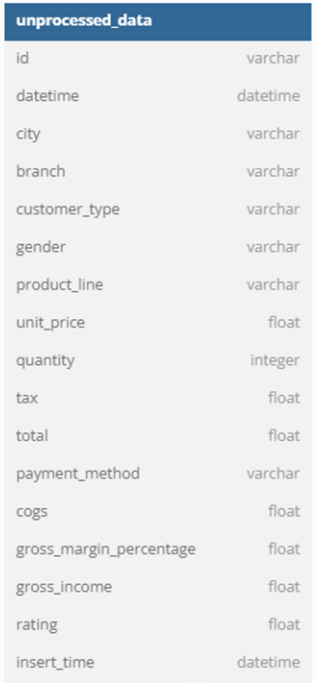
DDS
(таблицы фактов и измерений по схеме звезда)

Схема типа «звезда» включает в себя следующие таблицы:

1. Города (dds.cities):
   - Таблица для хранения названий городов.
2. Филиалы (dds.branches):
   - Таблица для хранения информации о филиалах, включающая информацию о городе.
3. Продуктовые линейки (dds.product_lines):
   - Таблица для хранения категорий товаров.
4. Способы оплаты (dds.payment_methods):
   - Таблица для хранения доступных методов оплаты.
5. Типы клиентов (dds.customer_types):
   - Таблица для хранения различных типов клиентов.
6. Дата продажи (dds.sales_dates):
   - Таблица для хранения информации о датах продажи, включая элементы дат.
7. Время продажи (dds.sales_time):
   - Таблица для хранения информации о времени продажи, включая элементы времени.
8. Рейтинг продаж (dds.sales_rating):
   - Таблица для хранения рейтингов продаж.
9. Продажи (dds.sales):
   - Основная таблица для хранения фактов о продажах. Содержит информацию о продажах, включая дату, время, место, тип клиента, продукт, стоимость, налог и другие характеристики. Связана с таблицами измерений через внешние ключи.

Временная и историческая таблицы

unprocessed_data и processed_data

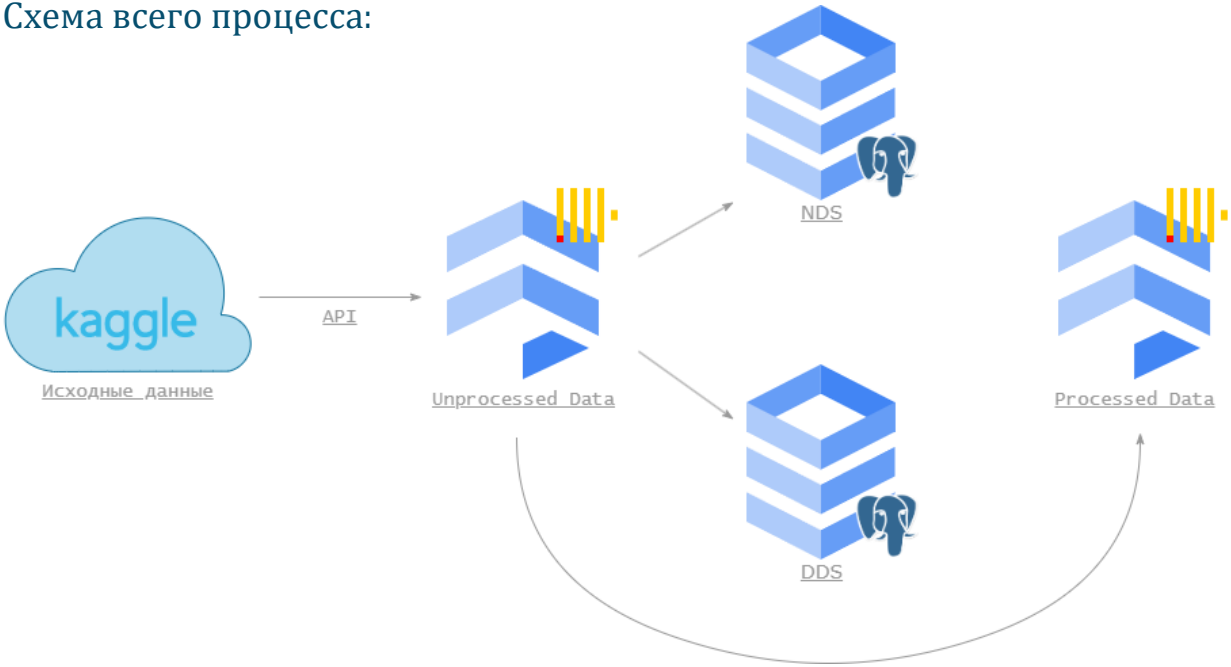


04

ETL-процессы

ETL-процессы были реализованы в 2 этапа. Первый DAG отвечает за выгрузку данных из Kaggle (через Kaggle API), второй DAG отвечает за загрузку данных в хранилище.

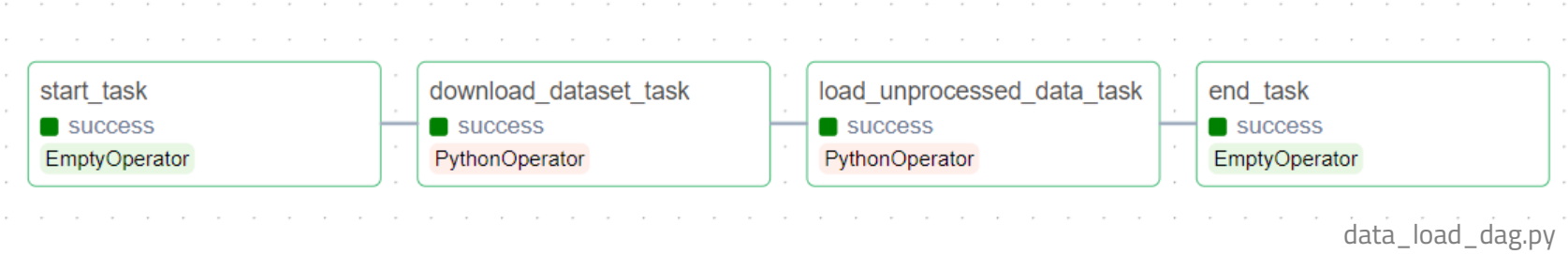
Схема всего процесса:



Выгрузка данных из источника (Kaggle) и загрузка в ClickHouse

Краткое описание: DAG выполняет выгрузку данных из источника (через Kaggle API), валидацию (проверка на Null значения, дубликаты строк, наличие валидных значений атрибутов, а также соответствие формата Invoice ID регулярному выражению), преобразование типов (раздельные атрибуты даты и времени преобразовываются в один) и загрузку данных в ClickHouse.

Граф задач:



data_load_dag.py

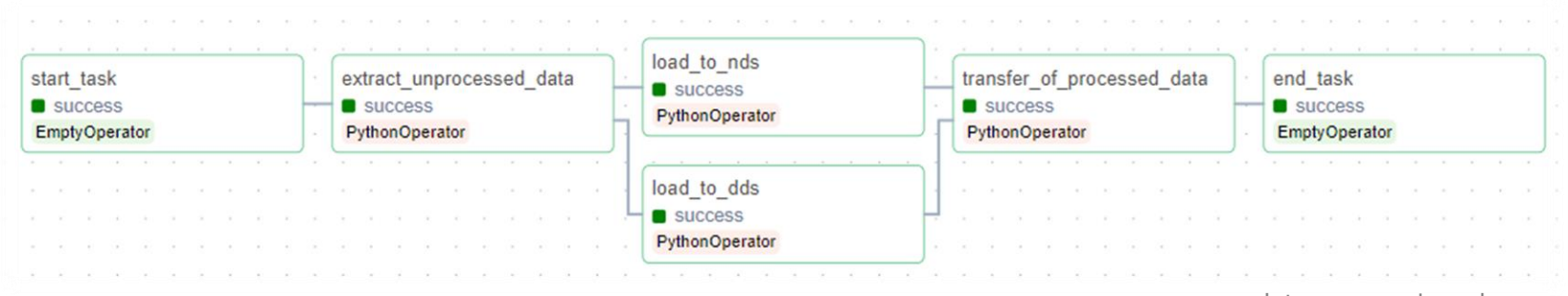
Подробное описание задач DAG:

1. Начальная задача (start_task):
    - Тип задачи: DummyOperator
    - Роль: фиктивная задача, обозначающая начало DAG.
2. Задача загрузки датасета из Kaggle (download_dataset_task):
    - Тип задачи: PythonOperator
    - Python-функция: download_dataset_from_kaggle
    - Параметры:
        - dataset_id: идентификатор датасета в Kaggle ('aungpyaeap/supermarket-sales').
        - dataset_path: путь для сохранения загруженного датасета ('./supermarket-sales').
    - Роль: загрузка датасета из Kaggle и распаковка в указанную директорию.
3. Задача загрузки необработанных данных в ClickHouse (load_unprocessed_data_task):
    - Тип задачи: PythonOperator
    - Python-функция: load_into_clickhouse
    - Параметры:
        - dataset_path: путь к загруженному датасету ('./supermarket-sales').
        - clickhouse_params: параметры подключения к ClickHouse (clickhouse_params).
        - table_name: имя таблицы в ClickHouse, куда будут загружены данные ('unprocessed_data').
        - batch_size: размер батча при загрузке данных (100).
    - Роль: обработка и валидация данных из датасета, а затем загрузка их в ClickHouse.
4. Завершающая задача (end_task):
    - Тип задачи: DummyOperator
    - Роль: фиктивная задача, обозначающая конец DAG.

Загрузка данных в хранилище

Краткое описание: DAG выполняет выгрузку данных из источника (таблица исходных данных в ClickHouse), маппинг значений (преобразование в id), загрузку данных в NDS и DDS, а также перемещение исходных данных из таблицы unprocessed_data в таблицу processed_data, очистку таблицы unprocessed_data.

Граф задач:



data_processing_dag.py

Подробное описание задач DAG:

1. Начальная задача (start_task):
    - Тип задачи: DummyOperator
    - Роль: фиктивная задача, обозначающая начало DAG.
2. Задача извлечения данных из ClickHouse (extract_task):
    - Тип задачи: PythonOperator
    - Python-функция: extract_unprocessed_data
    - Параметры:
        - clickhouse_params: параметры подключения к ClickHouse (clickhouse_params).
        - table_name: имя таблицы с необработанными данными ('unprocessed_data').
    - Роль: извлечение данных из ClickHouse и передача их для дальнейшей обработки.
3. Задача загрузки данных в PostgreSQL (NDS) (load_nds_task):
    - Тип задачи: PythonOperator
    - Python-функция: load_to_nds
    - Параметры:
        - postgres_params: параметры подключения к PostgreSQL (postgres_params).
        - column_map: словарь для маппинга значений (column_map).
        - schema: имя схемы в PostgreSQL ('nds').
    - Роль: загрузка обработанных данных в таблицу PostgreSQL для схемы NDS.

4. Задача загрузки данных в PostgreSQL (DDS) (load_dds_task):
    - Тип задачи: PythonOperator
    - Python-функция: load_to_dds
    - Параметры:
        - postgres_params: параметры подключения к PostgreSQL (postgres_params).
        - column_map: словарь для маппинга значений (column_map).
        - schema: Имя схемы в PostgreSQL ('dds').
    - Роль: загрузка обработанных данных в таблицы PostgreSQL для системы DDS.
5. Задача передачи обработанных данных в ClickHouse (transfer_task):
    - Тип задачи: PythonOperator
    - Python-функция: transfer_of_processed_data
    - Параметры:
        - clickhouse_params: параметры подключения к ClickHouse (clickhouse_params).
        - table_name: имя таблицы с обработанными данными ('processed_data').
        - batch_size: размер батча при загрузке данных (100).
    - Роль: загрузка обработанных данных в ClickHouse и очистка таблицы с необработанными данными.
6. Завершающая задача (end_task):
    - Тип задачи: DummyOperator
    - Роль: фиктивная задача, обозначающая конец DAG.

Проверка данных в хранилище

Нормализованная схема данных

```
SELECT p.id AS invoice_id,
    p.datetime,
    c."name" AS city,
    b."name" AS branch,
    ct."name" AS customer_type,
    pm."name" AS payment_method,
    pl."name" AS product_line,
    p.gender,
    p.unit_price,
    p.quantity,
    p.unit_price * p.quantity AS cogs,
    p.tax_percentage,
    p.unit_price * p.quantity / 100 * p.tax_percentage AS tax_amout,
    (p.unit_price * p.quantity) + (p.unit_price * p.quantity / 100 *
p.tax_percentage) AS total,
    p.gross_margin_percentage,
    ((p.unit_price * p.quantity) + (p.unit_price * p.quantity / 100 *
p.tax_percentage)) / 100 * p.gross_margin_percentage AS gross_income,
    p.rating
FROM nds.sales p
LEFT JOIN nds.cities c ON p.city_id = c.id
LEFT JOIN nds.branches b ON p.branch_id = b.id
LEFT JOIN nds.customer_types ct ON p.customer_type_id = ct.id
LEFT JOIN nds.payment_methods pm ON p.payment_method_id = pm.id
LEFT JOIN nds.product_lines pl ON p.product_line_id = pl.id
WHERE p.id = '761-49-0439';
```

Name	Value
invoice_id	761-49-0439
datetime	2019-01-19 10:17:00.000
city	Mandalay
branch	B
customer_type	Member
payment_method	Ewallet
product_line	Electronic accessories
gender	Female
unit_price	12.1
quantity	8
cogs	96.8
tax_percentage	5.0
tax_amout	4.84
total	101.64
gross_margin_percentage	4.761905
gross_income	4.8400002419999995
rating	8.6

Схема данных типа «звезда»

```
SELECT p.id AS invoice_id,
    sd."date",
    sd."year",
    sd."month",
    sd."day",
    sd."day_of_the_week",
    st.datetime::time AS "time",
    c."name" AS city,
    b."name" AS branch,
    ct."name" AS customer_type,
    pm."name" AS payment_method,
    pl."name" AS product_line,
    sr.rating,
    p.gender,
    p.unit_price,
    p.quantity,
    p.tax_amount,
    p.total,
    p.cogs,
    p.gross_income
FROM dds.sales p
LEFT JOIN dds.sales_dates sd ON p.date_id = sd.id
LEFT JOIN dds.sales_time st ON p.time_id = st.id
LEFT JOIN dds.cities c ON p.city_id = c.id
LEFT JOIN dds.branches b ON p.branch_id = b.id
LEFT JOIN dds.customer_types ct ON p.customer_type_id = ct.id
LEFT JOIN dds.payment_methods pm ON p.payment_method_id = pm.id
LEFT JOIN dds.product_lines pl ON p.product_line_id = pl.id
LEFT JOIN dds.sales_rating sr ON p.rating_id = sr.id
WHERE p.id = '253-12-6086';
```

```
Name           |Value            |
---------------+-----------------+
invoice_id     |253-12-6086      |
date           |2019-03-12       |
year           |2019             |
month          |3                |
day            |12               |
day_of_the_week|Tuesday          |
time           |12:43:00         |
city           |Yangon           |
branch         |A                |
customer_type  |Member           |
payment_method |Credit card      |
product_line   |Sports and travel|
rating         |8.7              |
gender         |Female           |
unit_price     |98.4             |
quantity       |7                |
tax_amount     |34.44            |
total          |723.24           |
cogs           |688.8            |
gross_income   |34.44            |
```

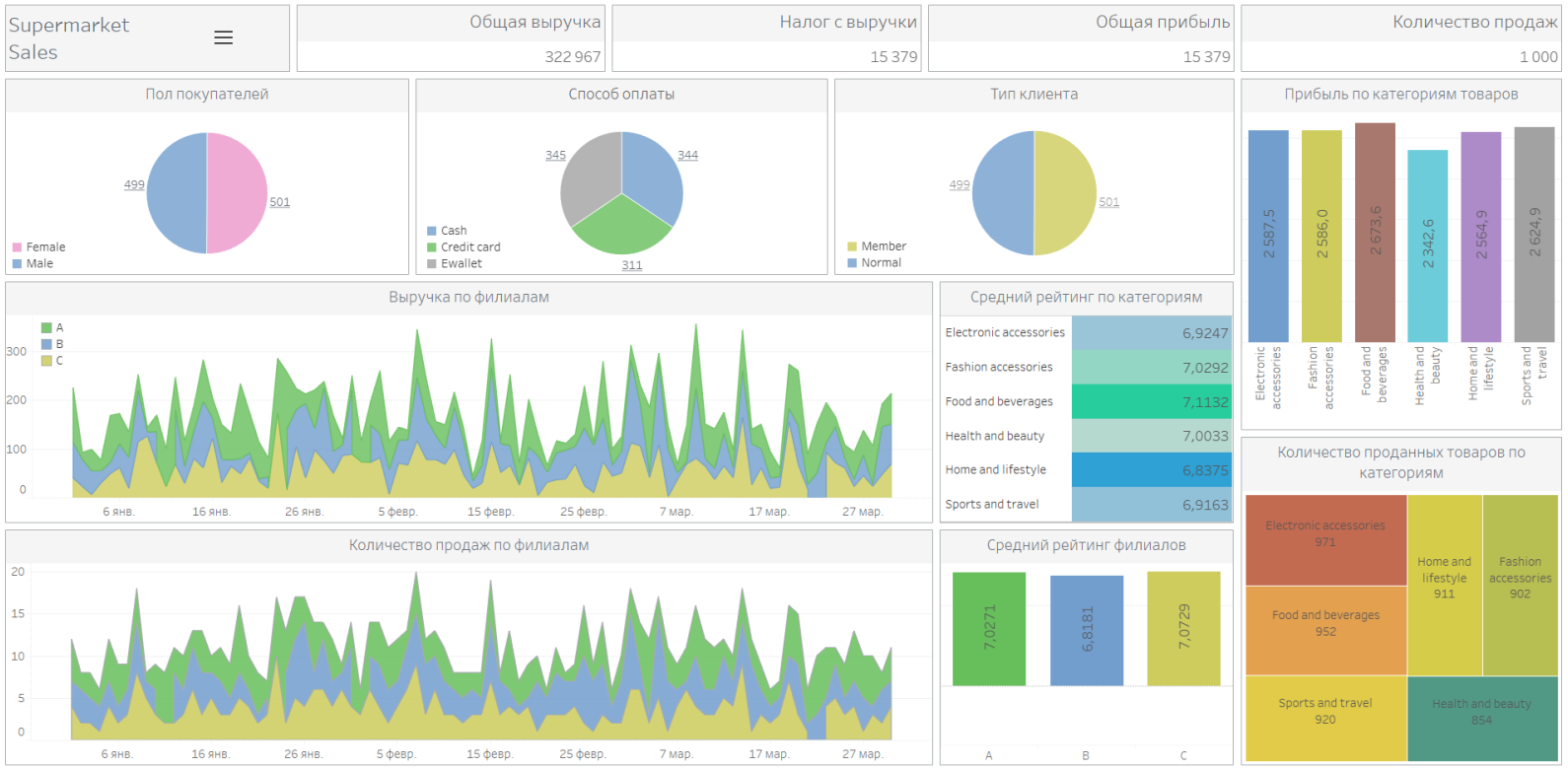
05

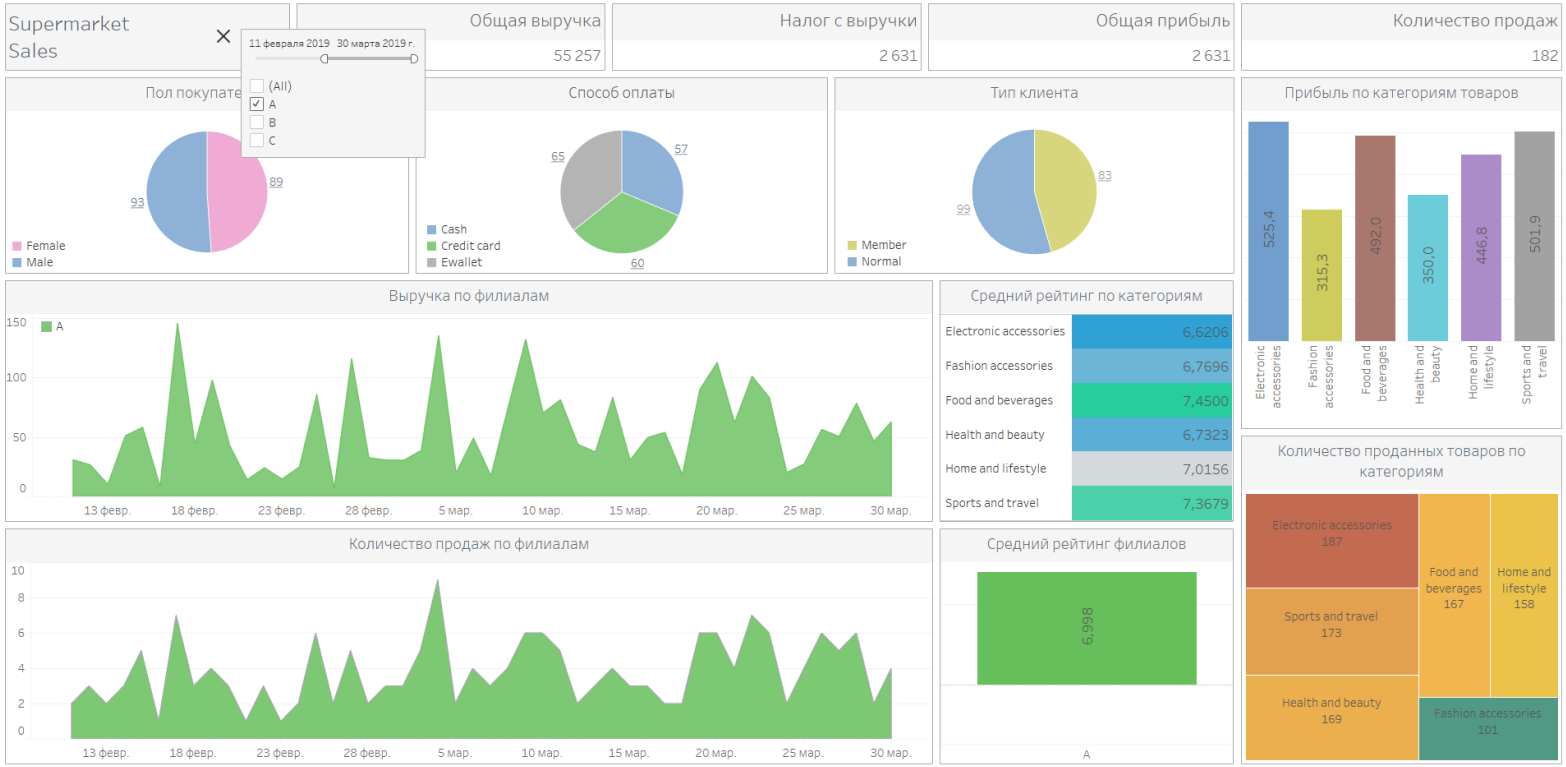
Визуализация данных

Выбранные метрики:

- Общая выручка компании:
  - Цель: оценить общую финансовую производительность компании.
- Налог с выручки:
  - Цель: обеспечить контроль за налоговой обязанностью компании и ее финансовой устойчивостью.
- Общая прибыль:
  - Цель: измерить общую прибыль и оценить финансовую эффективность бизнеса.
- Количество выполненных продаж:
  - Цель: оценить активность продаж и популярность товаров или услуг.
- Пол покупателей:
  - Цель: анализ гендерного распределения клиентов.
- Способ оплаты покупки:
  - Цель: изучение предпочтений клиентов по методам оплаты.
- Тип клиента (наличие клубной карты):
  - Цель: анализ клиентской базы и их участие в программе лояльности.
- Прибыль по категориям товаров:
  - Цель: оценка прибыльности различных категорий товаров.
- Выручка по филиалам:
  - Цель: измерение доли выручки, генерируемой каждым филиалом.
- Средний рейтинг от клиентов по категориям товаров:
  - Цель: оценка удовлетворенности клиентов различными категориями товаров.
- Количество продаж по филиалам:
  - Цель: измерение активности клиентов в различных филиалах.
- Средний рейтинг филиалов:
  - Цель: оценка удовлетворенности клиентов работой различных филиалов.
- Количество проданных товаров по категориям:
  - Цель: измерение популярности конкретных категорий товаров.

Итоговый дашборд

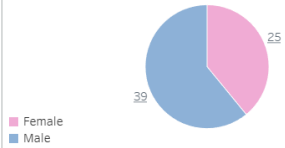



Supermarket Sales
11 февраля 2019 30 марта 2019 г.
(All)
A
B
C
Общая выручка
55 257
Налог с выручки
2 631
Общая прибыль
2 631
Количество продаж
182
Пол покупате
89
93
Female
Male
Способ оплаты
65
57
60
Cash
Credit card
Ewallet
Тип клиента
83
99
Member
Normal
Прибыль по категориям товаров
525,4
315,3
492,0
350,0
446,8
501,9
Electronic accessories
Fashion accessories
Food and beverages
Health and beauty
Home and lifestyle
Sports and travel
Выручка по филиалам
A
150
100
50
0
13 февр.
18 февр.
23 февр.
28 февр.
5 мар.
10 мар.
15 мар.
20 мар.
25 мар.
30 мар.
Средний рейтинг по категориям
Electronic accessories 6,6206
Fashion accessories 6,7696
Food and beverages 7,4500
Health and beauty 6,7323
Home and lifestyle 7,0156
Sports and travel 7,3679
Количество проданных товаров по категориям
Electronic accessories 187
Food and beverages 167
Home and lifestyle 158
Sports and travel 173
Health and beauty 169
Fashion accessories 101
Количество продаж по филиалам
10
8
6
4
2
0
13 февр.
18 февр.
23 февр.
28 февр.
5 мар.
10 мар.
15 мар.
20 мар.
25 мар.
30 мар.
Средний рейтинг филиалов
6,998
A

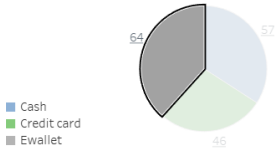
Supermarket Sales

Общая выручка	Налог с выручки	Общая прибыль	Количество продаж
19 331	920,5	920,5	64

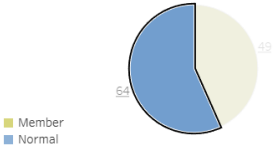
Пол покупателей



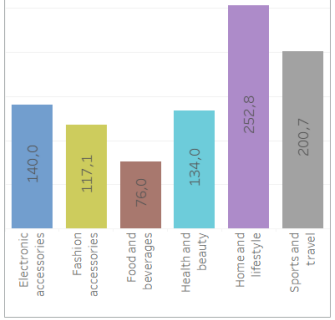
Способ оплаты



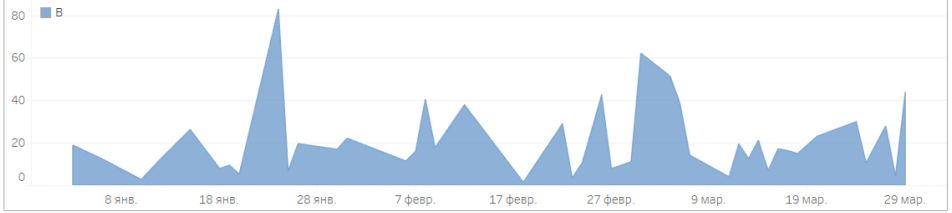
Тип клиента



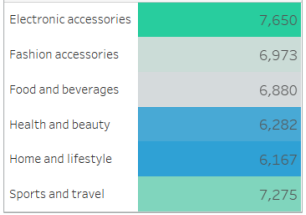
Прибыль по категориям товаров



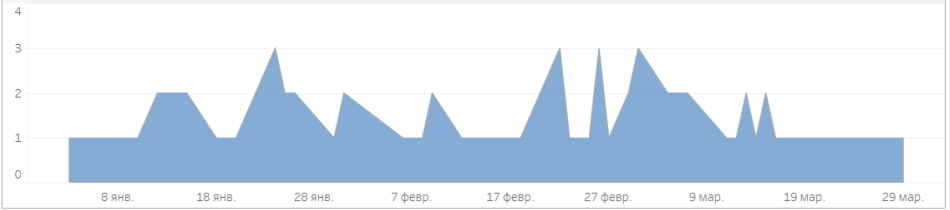
Выручка по филиалам



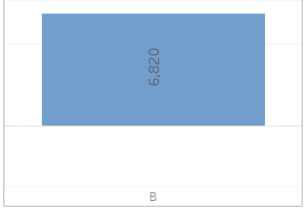
Средний рейтинг по категориям



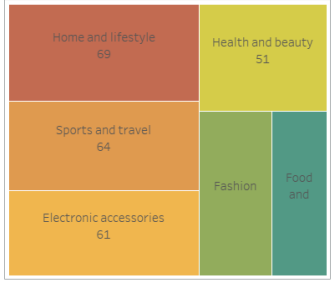
Количество продаж по филиалам



Средний рейтинг филиалов



Количество проданных товаров по категориям



06

Выводы

В процессе работы над дипломной работой было выполнено:

1. Обработка и анализ данных:
    - Проанализировали предоставленные данные и проверили их на ошибки.
2. Нормализованная схема данных (NDS):
    - Разработали структурированную нормализованную схему данных, обеспечивающую эффективное хранение и управление информацией.
3. Таблицы фактов и измерений (DDS):
    - Создали таблицы фактов и измерений, сформировав структуру схемы звезда для легкого доступа к ключевым показателям бизнеса.
4. ETL-процессы:
    - Разработали ETL-процессы для загрузки данных в NDS и DDS, обеспечивая эффективный и автоматизированный поток данных.
5. Дашборды в Tableau:
    - Построили дашборды в Tableau, визуализируя ключевые метрики, такие как общая выручка, прибыль, количество продаж, рейтинги и другие.

Решенные бизнес-задачи:

1. Управление финансами:
    - Мониторинг общей выручки компании и её филиалов.
    - Оценка общей прибыли и налога с выручки.
2. Анализ продаж:
    - Определение эффективности продаж в разрезе филиалов.
    - Изучение популярности продуктовых линеек.
3. Понимание клиентского поведения:
    - Анализ пола покупателей и их предпочтений в способах оплаты.
    - Выявление влияния наличия карты клиента.
4. Оптимизация ассортимента и рейтингов:
    - Ранжирование продуктовых линеек по прибыли и продажам.
    - Оценка среднего рейтинга от клиентов по категориям товаров и филиалам.

Итог работы:

Данная работа позволила закрепить полученные навыки в процессе прохождения курса.

Результат позволяет бизнесу эффективно управлять финансами, анализировать ключевые показатели продаж и клиентское поведение, принимать обоснованные решения для оптимизации бизнес-процессов и улучшения стратегии развития.

Спасибо за внимание!

Текст работы доступен на GitHub