

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 №5-1**

**Тема:**

«Проектирование объектной модели данных. Проектирование  
сквозного конвейера ETL»

Выполнил(а): Морозова Валерия АДЭУ-211

Преподаватель:

Москва

2025

4.1.1. Развернуть Конфигурация репозиторий ВМ в VirtualBox.

4.1.2. Клонировать на ПК задание Бизнес кейс Umbrella в домашний каталог ВМ.

```
git clone https://github.com/BosenkoTM/workshop-on-ETL.git

dev@dev-vm:~$ git clone https://github.com/BosenkoTM/workshop-on-ETL.git
Cloning into 'workshop-on-ETL'...
remote: Enumerating objects: 563, done.
remote: Counting objects: 100% (453/453), done.
remote: Compressing objects: 100% (394/394), done.
remote: Total 563 (delta 222), reused 59 (delta 32), pack-reused 110 (from 1)
Receiving objects: 100% (563/563), 5.82 MiB | 4.09 MiB/s, done.
Resolving deltas: 100% (260/260), done.
dev@dev-vm:~$ Valeria Morozova
```

Рисунок 1. Клонирование репозитория

4.1.3. Запустить контейнер с кейсом, изучить и описать основные элементы интерфейса Apache Airflow.

Сперва необходимо сделать проверку на наличие запущенных контейнеров (рисунок 2).

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
27d0183301b6	apache/airflow:2.10.4	"/usr/bin/dumb-init _"	13 days ago	Up About an hour (healthy)	0.0.0.0:8080->8080/tcp, [::]:8080->8080/tcp	lab_0_webinar-airflow-webserver
a96677480e55	apache/airflow:2.10.4	"/usr/bin/dumb-init _"	13 days ago	Up About an hour (healthy)	8080/tcp	lab_0_webinar-airflow-scheduler
1a9dfda58eb9	apache/airflow:2.10.4	"/usr/bin/dumb-init _"	13 days ago	Up About an hour (healthy)	8080/tcp	lab_0_webinar-airflow-triggerer
b0f2caef1d87	apache/airflow:2.10.4	"/usr/bin/dumb-init _"	13 days ago	Up About an hour (healthy)	8080/tcp	lab_0_webinar-airflow-worker-1
b679aa11c8d1	redis:7.2-bookworm	"docker-entrypoint.s..."	13 days ago	Up About an hour (healthy)	6379/tcp	lab_0_webinar-redis-1
b9fe220dd9fd	postgres:13	"docker-entrypoint.s..."	13 days ago	Up About an hour (healthy)	5432/tcp	lab_0_webinar-postgres-1
dd245700c5c8	postgres:latest	"docker-entrypoint.s..."	13 days ago	Restarting (137) Less than a second ago		pg

Рисунок 2. Список запущенных контейнеров

```
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop 27d0183301b6
27d0183301b6
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop a96677480e55
a96677480e55
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop 1a9dfda58eb9
1a9dfda58eb9
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop b0f2caef1d87
b0f2caef1d87
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop b679aa11c8d1
b679aa11c8d1
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker stop b9fe220dd9fd
b9fe220dd9fd
```

Рисунок 3. Остановка запущенных контейнеров

```
dev@dev-vm:~/workshop-on-ETL/business_case_umbrella$ sudo docker compose up -d
WARN[0000] /home/dev/workshop-on-ETL/business_case_umbrella/docker-compose.yml: t
[+] Running 5/5
 ✓ Network business_case_umbrella_default          Created
 ✓ Container business_case_umbrella-postgres-1      Started
 ✓ Container business_case_umbrella-scheduler-1     Started
 ✓ Container business_case_umbrella-init-1          Started
 ✓ Container business_case_umbrella-webserver-1     Started
```

Рисунок 4. Запуск контейнеров кейса Umbrella

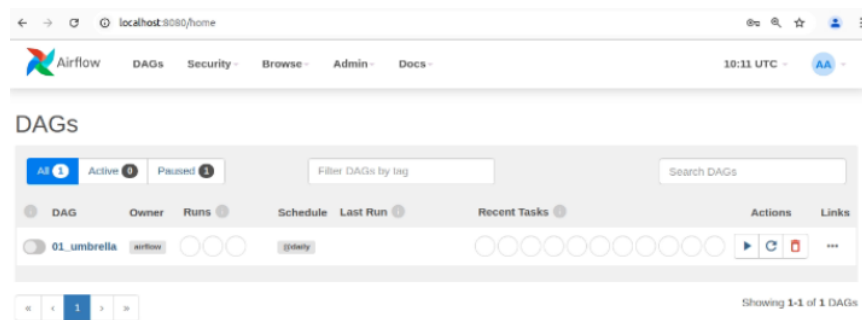


Рисунок 5. Airflow запущен

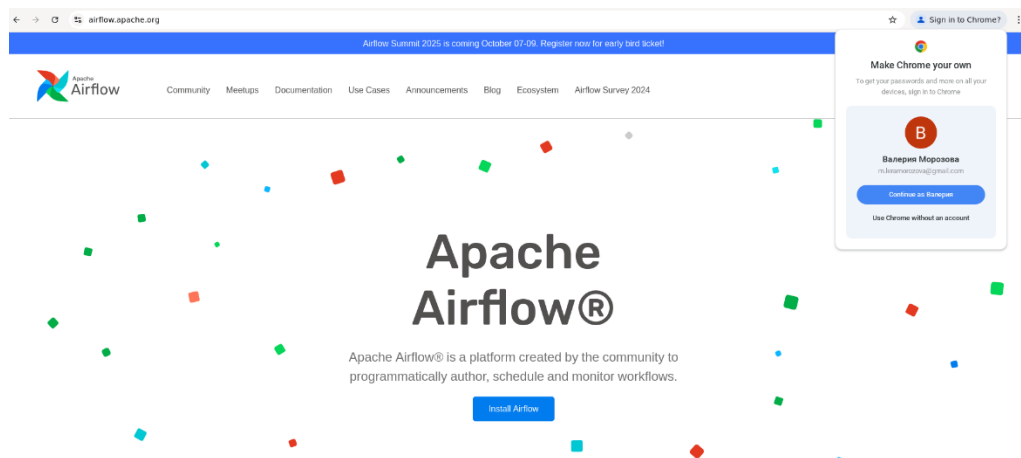


Рисунок 6. Apache Airflow

Основные элементы интерфейса Apache Airflow:

- ❖ **DAGs View.** Центральная панель управления, где можно увидеть все DAG, их расписание и статус выполнения. Для фильтрации DAG можно использовать теги.
- ❖ **Cluster Activity View.** Страница панели управления Airflow, где собираются полезные метрики для мониторинга кластера Airflow.
- ❖ **Datasets View.** Комбинированный список текущих наборов данных и график, иллюстрирующий, как они создаются и используются DAG.
- ❖ **Calendar View.** Вид календаря, который даёт обзор всей истории DAG за месяцы или даже годы. Позволяет быстро увидеть тенденции общего успеха или неудачи запусков с течением времени.
- ❖ **Variable View.** Позволяет перечислить, создать, отредактировать или удалить пару ключ-значение переменной, используемой во время заданий.

❖ **Gantt Chart.** Позволяет анализировать продолжительность задач и их перекрытие. Можно быстро выявить узкие места и понять, где тратится большая часть времени для конкретных запусков DAG.

❖ **Task Duration.** Вид, который показывает продолжительность различных задач за последние запуски. Позволяет найти выбросы и быстро понять, где тратится время в DAG за многие запуски.

❖ **Landing Times.** Время приземления для экземпляра задачи — это разница между концом интервала данных запуска DAG (обычно это означает, когда DAG «должен» запуститься) и временем завершения запуска DAG.

❖ **Code View.** Позволяет получить доступ к коду, который генерирует DAG, и предоставить больше контекста.

❖ **Trigger Form.** Если запустить ручной запуск DAG кнопкой со стрелкой, отобразится форма.

❖ **Audit Log.** Позволяет увидеть все события, связанные с DAG.

4.1.4. Спроектировать верхнеуровневую архитектуру аналитического решения задания Бизнес кейс Umbrella в draw.io. Необходимо использовать:

**Source Layer** - слой источников данных.

**Storage Layer** - слой хранения данных.

**Business Layer** - слой для доступа к данным бизнес пользователей.

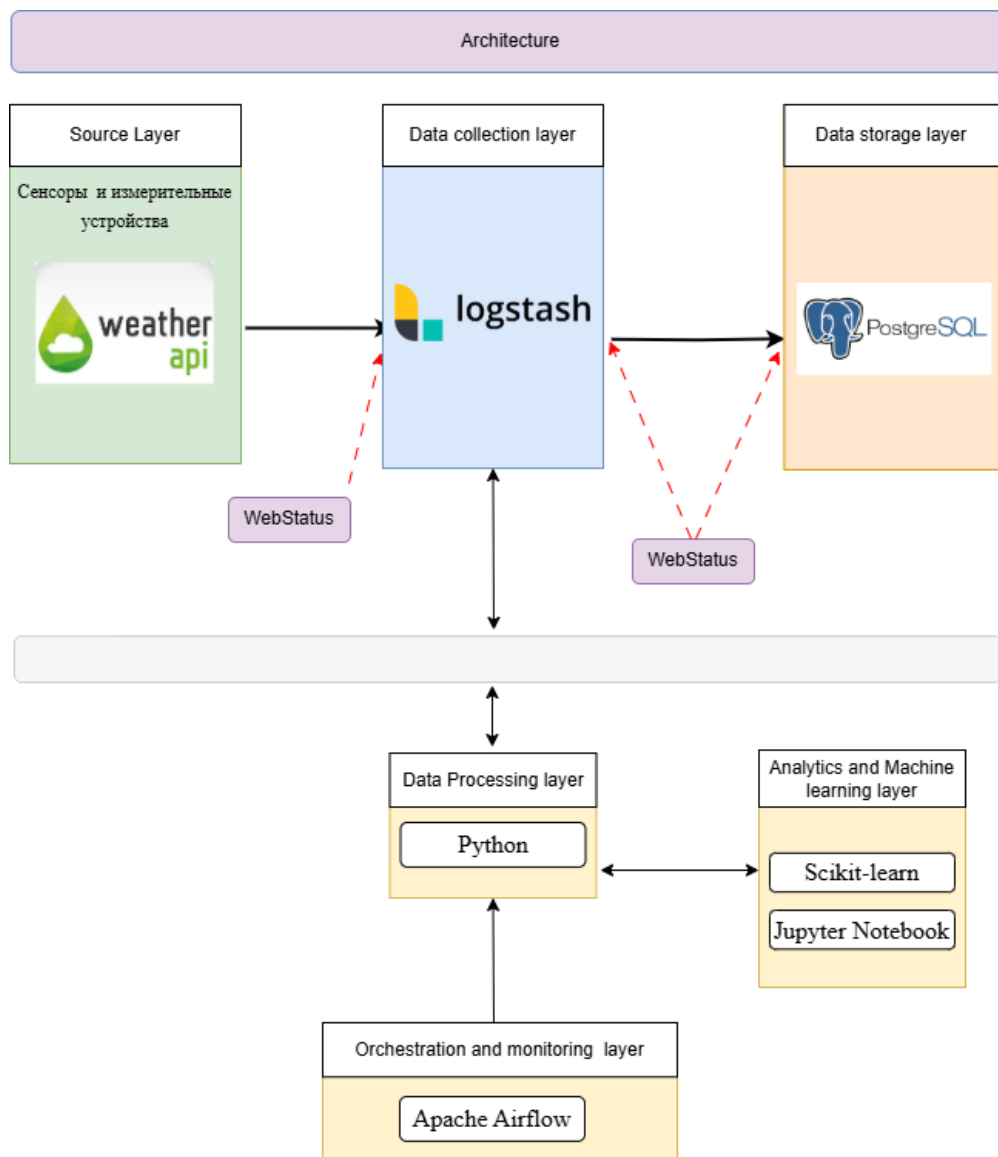


Рисунок 7. Архитектура аналитического решения задания Бизнес кейс Umbrella

#### Задание 4.2. Basic pipeline ETL

##### 4.2.1. Построить конвейер данных на основании Basic pipeline ETL.rar

Будут использоваться данные из практической работы №3, согласно варианту 10 данные технической поддержки, которые были интегрированы из трех источников.

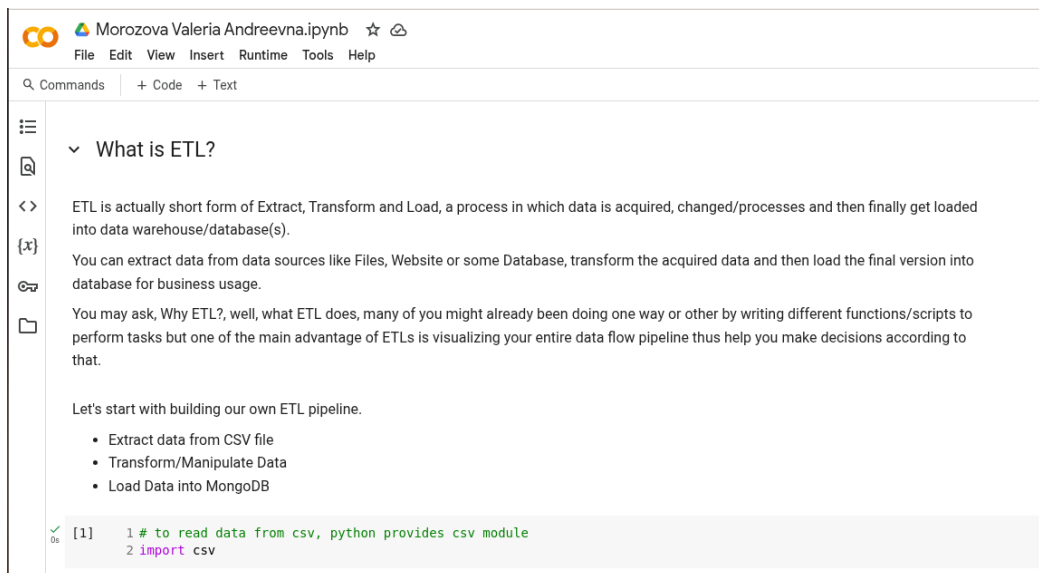


Рисунок 8. Импорт csv

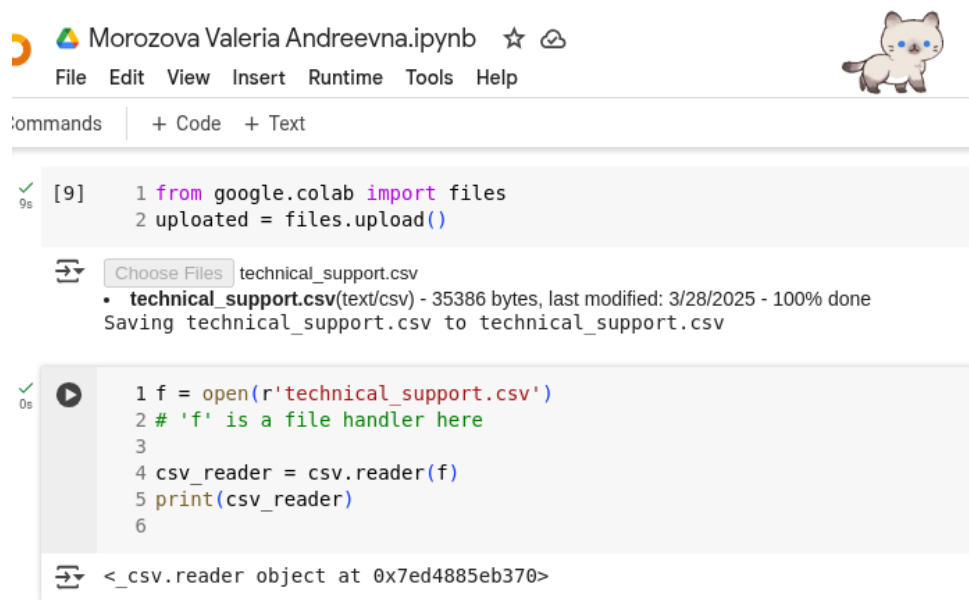


Рисунок 9. Загрузка файла

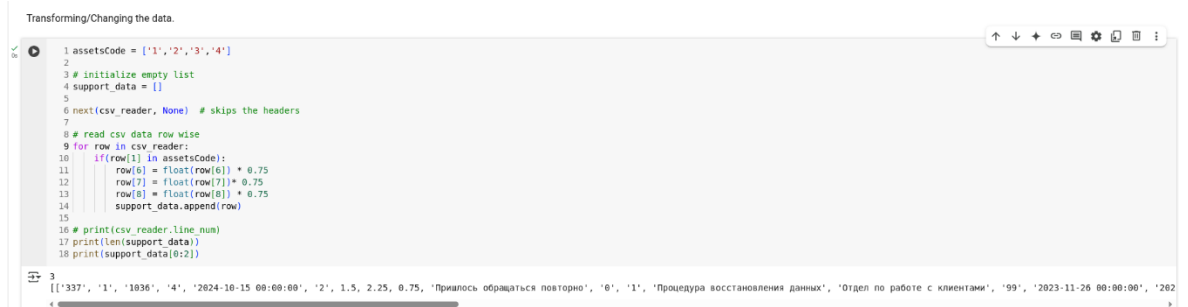
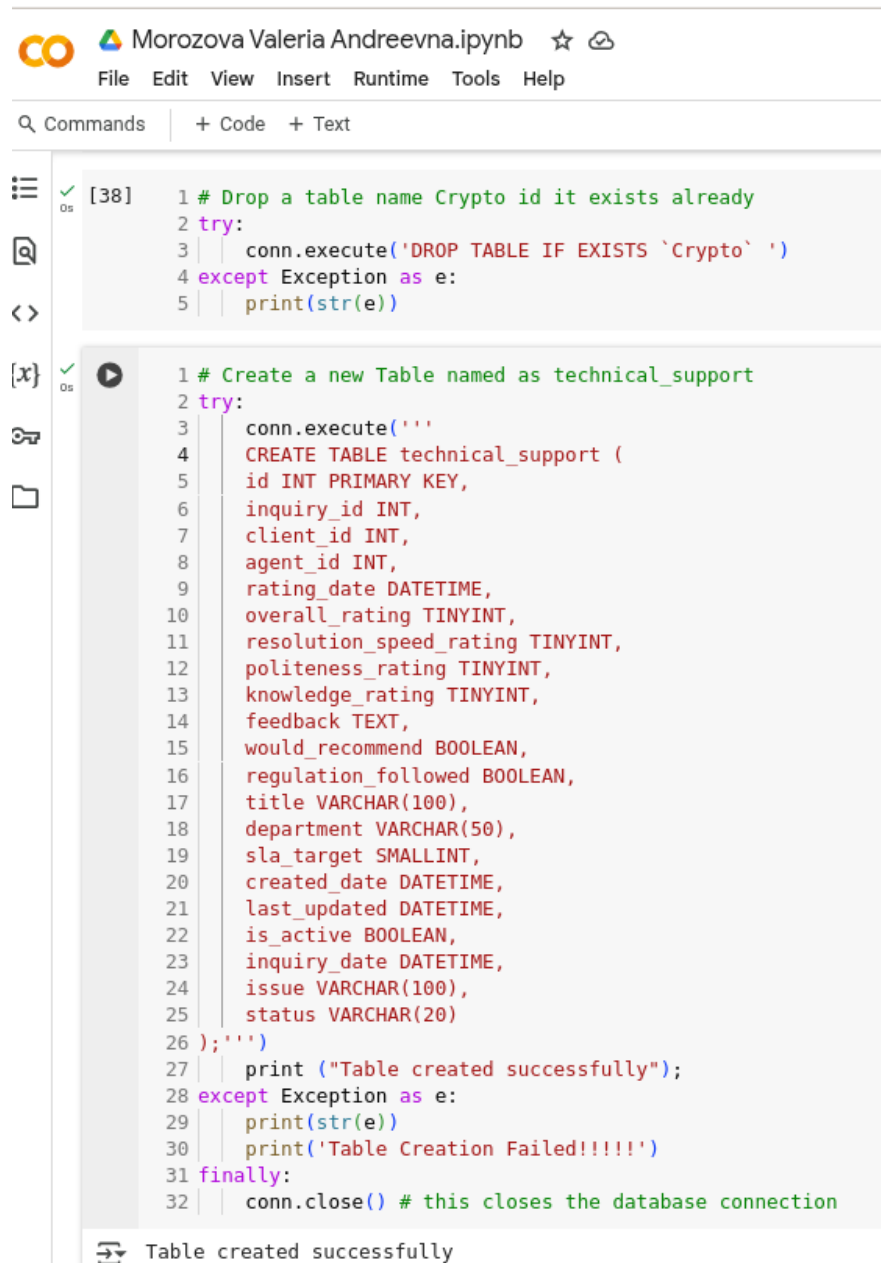


Рисунок 10. Трансформация рейтинговой системы

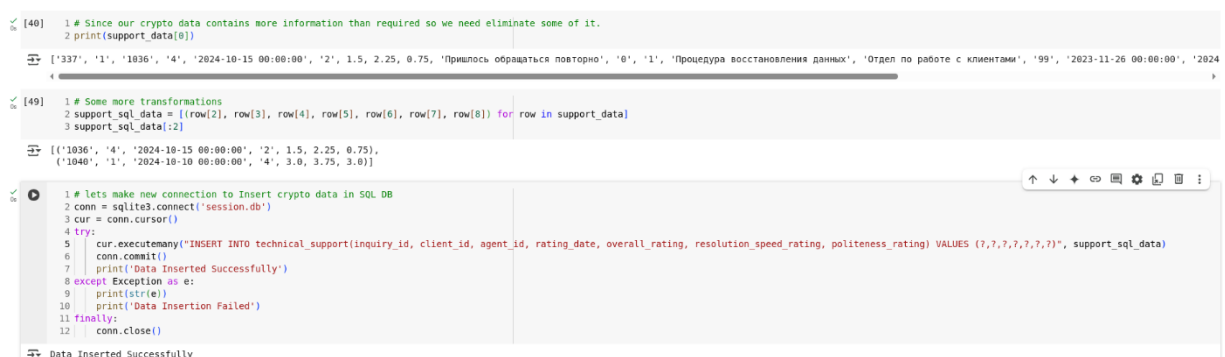


```
[38] 1 # Drop a table name Crypto id it exists already
2 try:
3     conn.execute('DROP TABLE IF EXISTS `Crypto` ')
4 except Exception as e:
5     print(str(e))

1 # Create a new Table named as technical_support
2 try:
3     conn.execute('''
4         CREATE TABLE technical_support (
5             id INT PRIMARY KEY,
6             inquiry_id INT,
7             client_id INT,
8             agent_id INT,
9             rating_date DATETIME,
10            overall_rating TINYINT,
11            resolution_speed_rating TINYINT,
12            politeness_rating TINYINT,
13            knowledge_rating TINYINT,
14            feedback TEXT,
15            would_recommend BOOLEAN,
16            regulation_followed BOOLEAN,
17            title VARCHAR(100),
18            department VARCHAR(50),
19            sla_target SMALLINT,
20            created_date DATETIME,
21            last_updated DATETIME,
22            is_active BOOLEAN,
23            inquiry_date DATETIME,
24            issue VARCHAR(100),
25            status VARCHAR(20)
26        );''')
27     print ("Table created successfully");
28 except Exception as e:
29     print(str(e))
30     print('Table Creation Failed!!!!!!')
31 finally:
32     conn.close() # this closes the database connection

Table created successfully
```

Рисунок 11. Создание таблицы



```
[40] 1 # Since our crypto data contains more information than required so we need eliminate some of it.
2 print(support_data[8])

['337', '1', '1036', '4', '2024-10-15 00:00:00', '2', 1.5, 2.25, 0.75, 'Пришлось обращаться повторно', '6', '1', 'Процедура восстановления данных', 'Отдел по работе с клиентами', '99', '2023-11-26 00:00:00', '2024-10-15 00:00:00']

1 # Some more transformations
2 support_sql_data = [(row[2], row[3], row[4], row[5], row[6], row[7], row[8]) for row in support_data]
3 support_sql_data[:2]

[('1036', '4', '2024-10-15 00:00:00', '2', 1.5, 2.25, 0.75), ('1040', '1', '2024-10-10 00:00:00', '4', 3.0, 3.75, 3.0)]

1 # lets make new connection to Insert crypto data in SQL DB
2 conn = sqlite3.connect('session.db')
3 cur = conn.cursor()
4 try:
5     cur.executemany("INSERT INTO technical_support(inquiry_id, client_id, agent_id, rating_date, overall_rating, resolution_speed_rating, politeness_rating) VALUES (?, ?, ?, ?, ?, ?, ?)", support_sql_data)
6     conn.commit()
7     print('Data Inserted Successfully')
8 except Exception as e:
9     print(str(e))
10    print('Data Insertion Failed')
11 finally:
12    conn.close()

Data Inserted Successfully
```

Рисунок 12. Выборка столбцов и загрузка данных в новую таблицу

```
[51] 1 # Let's Read data from DB to verify it
      2
      3 conn = sqlite3.connect('session.db')
      4 rows = conn.cursor().execute('Select * from technical_support')
      5
      6 for row in rows:
      7     print(row)

(None, 1036, 4, '2024-10-15 00:00:00', 2, 1.5, 2.25, 0.75, None, None, None, None, None, None, None, None, None, None, None)
(None, 1040, 1, '2024-10-10 00:00:00', 4, 3, 3.75, 3, None, None, None, None, None, None, None, None, None, None, None)
(None, 1095, 8, '2024-10-20 00:00:00', 3, 1.5, 3, 1.5, None, None, None, None, None, None, None, None, None, None, None)

Write data in a csv file

1 csvfile = open('Support_data.csv', 'w')
2 csv_writer = csv.writer(csvfile, lineterminator='\r')
3 # Now we can write data to files using two methods:
4 # writerow() or writerows()
5 # writerow() is used when we need to write one-dimension data such as a single list :[1, 'Jerry', 95]
6 # writerows() is used when we need to write multi-dimension data such as list of list [[1, 'Jerry', 95], [2, 'Tom', 80], [3, 'Scooby', 90]]
7 # So the only difference is that writerows() lets you pass multiple values!
8 csv_writer.writerow(['inquiry_id', 'client_id', 'agent_id', 'rating_date', 'overall_rating', 'resolution_speed_rating', 'politeness_rating'])
9 csv_writer.writerows(support_sql_data)
10 csvfile.close()
```

Рисунок 13. Выгрузка выбранных данных по столбцам в csv файл

### Выводы:

1. Клонирован на ПК задание Бизнес-кейс Umbrella в домашний каталог ВМ.
2. Запущен контейнер с кейсом, изучить и описать основные элементы интерфейса Apache Airflow.
3. Спроектирована верхнеуровневая архитектура аналитического решения задания Бизнес кейс Umbrella в draw.io.
4. Построен конвейер данных на основании Basic pipeline ETL.rar.