HW5

Запуск

1) Композ файл с монтированием директории:

```
services:
  spark:
    build:
      context: .
      dockerfile: Dockerfile
    image: spark-python
    container_name: spark-python
    volumes:
      - ./app:/app
```

2) Докерфайл из образа `apache/spark:3.5.0` с установкой `python` и pyspark:

```
FROM apache/spark:3.5.0

USER root

RUN apt-get update && \
    apt-get install -y python3 python3-pip && \
    ln -sf /usr/bin/python3 /usr/bin/python && \
    pip3 install --no-cache-dir pyspark && \
    apt-get clean && rm -rf /var/lib/apt/lists/*

WORKDIR /app

CMD ["bash","run_test.sh"]
```

3) запуск spark приложения `docker compose up`

Spark приложение

1) `test_spark_df.py`: Обработка csv, агрегация и усреднение по колонке, вывод с сортировкой.

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import avg, count

spark = SparkSession.builder.appName("CSVStats").getOrCreate()

df = spark.read.csv("./data/flights.csv", header=True, inferSchema=True)

df.groupBy("DayofMonth").agg(
    count("*").alias("rows"),
    avg("DepDelay").alias("avg_DepDelay")
).orderBy("DayofMonth").show(
    n=10,
    truncate=False,
    vertical=False
)

spark.stop()
```

Вывод:

```
spark-python  | |DayofMonth|rows |avg_DepDelay      |
spark-python  | +----------+-----+------------------+
spark-python  | |1         |84636|8.868507490902216 |
spark-python  | |2         |89760|10.928219696969697|
spark-python  | |3         |90172|9.696346981324579 |
spark-python  | |4         |84758|6.338304348852025 |
spark-python  | |5         |86426|6.2484900377201305|
spark-python  | |6         |87702|7.087899933866958 |
spark-python  | |7         |88011|10.064753269477679|
spark-python  | |8         |89019|10.662768622428919|
spark-python  | |9         |91412|11.449831531965168|
spark-python  | |10        |90025|16.279622327131353|
spark-python  | +----------+-----+------------------+
spark-python  | only showing top 10 rows
```

2) test_spark_df.py: Обработка таблиц через SQL синтаксис, объединение таблиц, агрегация и усреднение по колонке, вывод с сортировкой

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("SparkSQLJoin").getOrCreate()

flights = spark.read.csv("./data/flights.csv", header=True, inferSchema=True)
carriers = spark.read.csv("./data/airports.csv", header=True, inferSchema=True)

flights.createOrReplaceTempView("flights")
carriers.createOrReplaceTempView("airports")

spark.sql("""
    SELECT
        a1.name AS origin_airport,
        a2.name AS destination_airport,
        ROUND(AVG(f.DepDelay),2) AS avg_dep_delay
    FROM flights f
    JOIN airports a1 ON f.OriginAirportID = a1.airport_id
    JOIN airports a2 ON f.DestAirportID = a2.airport_id
    WHERE f.DepDelay IS NOT NULL
    GROUP BY a1.name, a2.name
    HAVING AVG(f.DepDelay) > 30
    ORDER BY avg_dep_delay DESC;
""").show(
    n=10,
    truncate=False,
    vertical=False
)

spark.stop()
```

Вывод:

```
spark-python  | +-------------------------------------------+---------------------------------
spark-python  | |origin_airport                             |destination_airport
spark-python  | +-------------------------------------------+---------------------------------
spark-python  | |Pittsburgh International                   |Richmond International
spark-python  | |Pittsburgh International                   |Raleigh-Durham International
spark-python  | |Los Angeles International                  |Eppley Airfield
spark-python  | |Cincinnati/Northern Kentucky International|Cleveland-Hopkins International
spark-python  | |Seattle/Tacoma International               |Miami International
spark-python  | |Chicago Midway International               |Ontario International
spark-python  | |Fort Lauderdale-Hollywood International    |Richmond International
spark-python  | |Chicago Midway International               |San Francisco International
spark-python  | |Norfolk International                      |Minneapolis-St Paul Internationa
spark-python  | |Metropolitan Oakland International          |Logan International
spark-python  | +-------------------------------------------+---------------------------------
spark-python  | only showing top 10 rows
```