Phạm Hoàng Long - MSV:21002218 - KTĐTTH

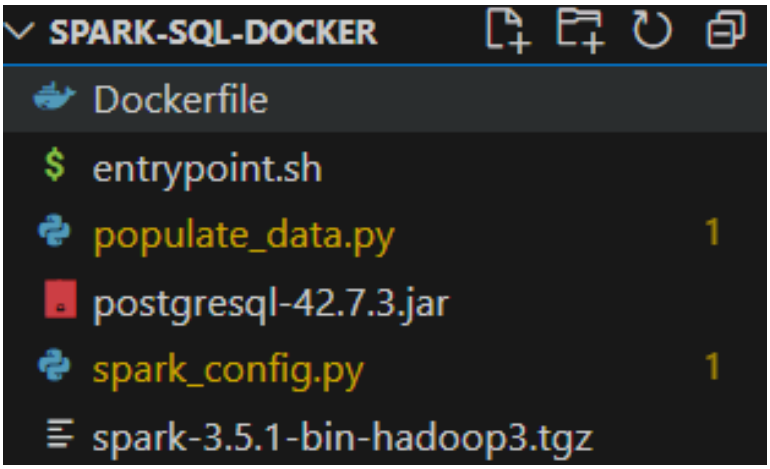
Hướng dẫn Thiết lập và Sử dụng Docker Container cho SparkSQL và PostgreSQL

Mục tiêu

Mục tiêu của dự án này là thiết lập và chạy SparkSQL trong một Docker container. Docker container sẽ chứa tất cả các phụ thuộc cần thiết để chạy Spark và một cơ sở dữ liệu PostgreSQL để lưu trữ dữ liệu. Người dùng sẽ có thể thực hiện các truy vấn SQL trên cơ sở dữ liệu này thông qua SparkSQL.

Yêu cầu

1. Docker: Đảm bảo Docker đã được cài đặt trên hệ thống .Có thể tải xuống và cài đặt Docker từ trang web chính thức của Docker.
2. Apache Spark: Truy cập trang web chính thức của Apache Spark và tải xuống phiên bản `spark-3.5.1-bin-hadoop3.tgz`. Đảm bảo rằng tệp này được lưu trong thư mục dự án
3. JDBC PostgreSQL: Có thể tải PostgreSQL từ trang web chính thức. Đảm bảo bạn đã tải xuống file JAR PostgreSQL JDBC driver và đặt nó trong cùng thư mục với Dockerfile
4. Python và PySpark: Đã cài đặt
5. Tệp và thư mục: Tạo một thư mục mới cho dự án và di chuyển tệp tin Spark đã tải xuống vào trong thư mục đó. Cấu trúc thư mục nên trông giống như sau



Thực hành

Bước 1: Tạo Dockerfile và Script Entry Point

1. Tạo Dockerfile

Tạo một tệp có tên là `Dockerfile` trong thư mục dự án.

Sử dụng `Dockerfile` đã cung cấp để xây dựng Docker image.

```
# Sử dụng image OpenJDK chính thức làm cơ sở
FROM openjdk:11-jre-slim

# Đặt biến môi trường cho Spark
ENV SPARK_HOME=/opt/spark
ENV PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin

# Cài đặt các gói phụ thuộc
RUN apt-get update && \
    apt-get install -y curl python3 python3-pip postgresql postgresql-cont
    apt-get clean

# Sao chép và cài đặt Spark
COPY spark-3.5.1-bin-hadoop3.tgz /opt/
RUN tar -xzf /opt/spark-3.5.1-bin-hadoop3.tgz -C /opt/ && \
    rm /opt/spark-3.5.1-bin-hadoop3.tgz && \
    mv /opt/spark-3.5.1-bin-hadoop3 /opt/spark

# Sao chép script entrypoint
COPY entrypoint.sh /usr/local/bin/

# Đặt quyền thực thi cho script entrypoint
RUN chmod +x /usr/local/bin/entrypoint.sh

# Đặt entrypoint
ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]

# Mở các cổng cần thiết
EXPOSE 4040 8080 7077 5432
```

2. Tạo Script Entry Point

Tạo một tệp có tên là `entrypoint.sh` trong cùng thư mục với Dockerfile.

```
#!/bin/bash

# Khởi động dịch vụ PostgreSQL
service postgresql start

# Tạo người dùng và cơ sở dữ liệu PostgreSQL
su - postgres -c "psql -c \"CREATE USER sparkuser WITH PASSWORD 'sparkpassword';\""
su - postgres -c "psql -c \"CREATE DATABASE sparkdb;\""
su - postgres -c "psql -c \"GRANT ALL PRIVILEGES ON DATABASE sparkdb TO sparkuser;\""

# Khởi động Spark
$SPARK_HOME/sbin/start-master.sh
$SPARK_HOME/sbin/start-worker.sh spark://$(hostname):7077

# Giữ container chạy
tail -f /dev/null
```

Bước 2: Xây dựng và Chạy Docker Container

1. Xây dựng Docker Image

Mở terminal và điều hướng đến thư mục chứa Dockerfile và các tệp khác. Sau đó, chạy lệnh sau để xây dựng Docker image:

```
docker build -t spark-sql-docker .
```

Lệnh này sẽ tạo một Docker image có tên `spark-sql-docker` sử dụng các chỉ thị trong Dockerfile.

2. Chạy Docker Container

Sau khi xây dựng xong, chạy lệnh sau để khởi động Docker container:

```
docker run -it -p 4040:4040 -p 8080:8080 -p
  7077:7077 -p 5432:5432 spark-sql-docker
```

- `-it`: Chạy container trong chế độ tương tác với terminal .
- `-p 4040:4040 -p 8080:8080 -p 7077:7077 -p 5432:5432`: Ánh xạ các cổng của container với các cổng trên máy chủ để có thể truy cập các dịch vụ bên trong container từ bên ngoài.

Sau khi chạy 2 lệnh trên, ta sẽ được đưa vào bên trong docker có dạng như sau

```
C:\Users\lamtu>docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED       STATUS       PORTS
                     NAMES
96bc1c7a19f9   spark-sql-docker   "/usr/local/bin/entr…"   2 hours ago   Up 2 hours   0.0.0.0:4040->4040/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:7077->7077/tcp
, 0.0.0.0:8080->8080/tcp   elated_cannon
```

Bước 3: Thêm Dữ liệu và Thực hiện Truy vấn

1. Thêm Dữ liệu vào Cơ sở Dữ liệu

Tạo một tệp có tên là `populate_data.py` để nhập dữ liệu mẫu vào cơ sở dữ liệu PostgreSQL:

```python
import psycopg2
import random

conn = psycopg2.connect(
    dbname="sparkdb",
    user="sparkuser",
    password="sparkpassword",
    host="localhost"
)
cur = conn.cursor()

cur.execute("""
    CREATE TABLE sample_data (
        id SERIAL PRIMARY KEY,
        value INTEGER
    );
""")

for _ in range(1000000):
    cur.execute("INSERT INTO sample_data (value) VALUES (%s)", (random.randint(1, 100),))

conn.commit()
cur.close()
conn.close()
```

Chạy các lệnh sau trong một terminal mới để sao chép script Python vào container và thực thi nó để nhập dữ liệu mẫu vào cơ sở dữ liệu:

```
docker cp populate_data.py <container_id>:/populate_data.py

docker exec -it <container_id> python3 /populate_data.py
```

2. Thực hiện Truy vấn SparkSQL

Tạo một tệp có tên là `spark_config.py` để cấu hình Spark và thực hiện truy vấn SparkSQL:

```python
from pyspark.sql import SparkSession

# Cấu hình Spark session
spark = SparkSession.builder \
    .appName("Spark PostgreSQL") \
    .config("spark.jars", "C:\Users\lamtu\spark-sql-docker\postgresql-42.7.3.jar") \
    .getOrCreate()

# Đọc dữ liệu từ PostgreSQL
df = spark.read \
    .format("jdbc") \
    .option("url", "jdbc:postgresql://localhost:5432/sparkdb") \
    .option("dbtable", "sample_table") \
    .option("user", "sparkuser") \
    .option("password", "sparkpassword") \
    .option("driver", "org.postgresql.Driver") \
    .load()

df.show()
```

Chạy các lệnh sau trong một terminal mới để sao chép file `spark_config.py` vào container và cấu hình Spark để kết nối với cơ sở dữ liệu PostgreSQL và thực hiện truy vấn:

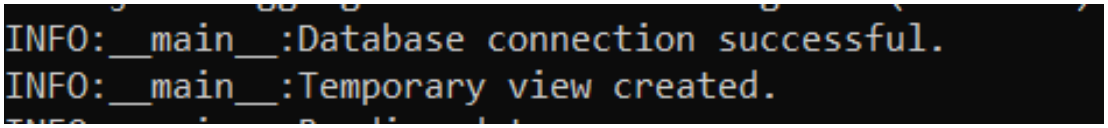
```
docker cp spark_config.py 96bc1c7a19f9:/spark_config.py

ocker cp C:\Users\lamtu\spark-sql-docker\postgresql-42.7.3.jar
96bc1c7a19f9: C:\Users\lamtu\spark-sql-docker\postgresql-42.7.3.jar

docker exec -it 96bc1c7a19f9 python3 /spark_config.py
```

Khi thấy thông báo này, database đã được hoàn thành, ta có thể kiểm tra bằng cách đếm số lượng dòng trong database:

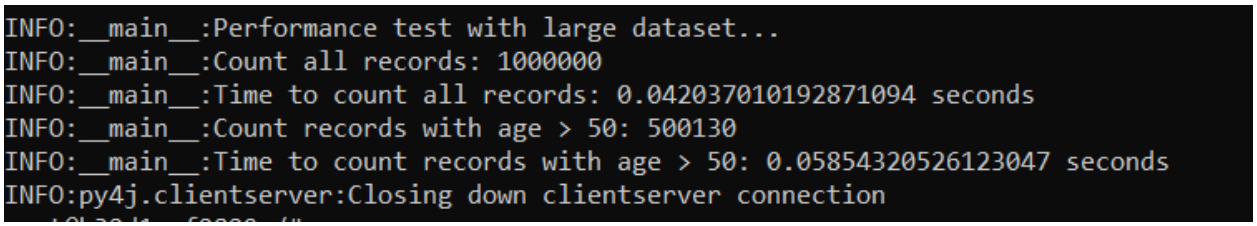
```
root@b32d1cef2829:/# sqlite3 /opt/spark/examples/src/main/resources/people.db
SQLite version 3.34.1 2021-01-20 14:10:07
Enter ".help" for usage hints.
sqlite> SELECT COUNT (*) FROM people;
1000000
```

Chúng ta có thể thấy số lượng dòng trong database.

Sau đây là kết quả khi chạy các file ở trên:



Thông báo về việc đã kết nối database và tạo temporary view.



Kiểm tra thời gian count toàn bộ record và count có điều kiện WHERE và đóng kết nối server với database.