**Nguyễn hữu thắng**

**Cấu hình và chạy SparkSQL truy vấn cơ sở dữ liệu với DOCKER**

1. **Thiết lập Docker**

* Tải docker desktop từ trang web chính thức cho windown
* Tạo một thư mục có tên là prj
* Open thư mục bằng vscode
* Tạo một Dockerfile không có đuôi tệp
* Ghi vào file câu lệnh sau: FROM ubuntu:22.04
* Ý nghĩa câu lệnh trên là xây dựng docker image dựa trên docker image ubuntu:22.04
* Ta gõ tiếp: WORKDIR /app

Câu lệnh này giúp xác định thư mục làm việc chính.

* Tiếp theo. Mở một terminal gõ câu lênh: docker build -t myprj .

Trong đó docker là câu lệnh giúp gọi ứng dụng có tên là docker có trên đường dẫn hệ thống ra

Các chữ tiếp theo là truyền đối số dòng lệnh giúp ứng dụng docker đọc được

build có nghĩa là ta đang xây dựng docker image từ dockerfile. -t là cờ giúp đặt tên cho dockerfile là myprj. Dấu chấm có nghĩa là tìm dockerfile tại thư mục hiện tại.

* Khi chạy xong câu lệnh trên. Mở docker desktop ra. Đã có một docker image có tên myprj được tạo ra
* Ta tiếp tục gõ trên terminal câu lệnh: docker run -it myprj bash
* Khi đó. Một docker container được tạo ra từ docker image myprj. Và khi này, ta đang ở bên trong docker container.
* Ta gõ câu lệnh: apt update để cập nhật hệ thống.
* Câu lệnh đã chạy thành công. Khi này ta gõ dòng tiếp theo vào dockerfile: RUN apt update

1. **Cài đặt Spark**

* Apache Spark cần môi trường thực thi. Vậy trước tiên ta cần cài jdk: apt install openjdk-21-jdk
* Câu lệnh trên hoạt động bình thường. khi này ta ghi câu lệnh này vào dockerfile: RUN apt install openjdk-21-jdk
* Ta thử build image từ dockerfile. Ta gặp một lỗi là:

ERROR: failed to solve: process "/bin/sh -c apt install openjdk-21-jdk" did not complete successfully: exit code: 1

* Khi ta đứng trong container thì ta có thể nhấn yes hay no. nhưng khi ta build image thì không ai nhấn cho ta cả.
* Cách sử lý là thêm -y vào trước. câu lệnh sẽ là: RUN apt -y install openjdk-21-jdk
* Khi đó. Build lại image sẽ hoạt động bình thường mà không gặp lỗi gì.
* Ta tiếp tục với một số câu lệnh để Cài đặt Apache Spark trong vùng chứa Docker.
* RUN apt -y install wget
* Câu lệnh trên giúp tải wget từ kho chứa apt về máy. wget dùng để tải file từ internet.
* Tiếp theo:   
  RUN wget https://dlcdn.apache.org/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz

RUN tar xvf spark-3.5.1-bin-hadoop3.tgz

RUN rm spark-3.5.1-bin-hadoop3.tgz

Các câu lệnh này giúp tải spark phiên bản 3.5.1. giải nén. Và xóa file nén sau khi đã giải nén

* RUN mv spark-3.5.1-bin-hadoop3 /opt/spark

Di chuyển và đổi tên thư mục vừa giải nén thành thư mục tại vị trí /opt/spark

* Tiếp theo:

ENV SPARK\_HOME=/opt/spark

ENV PATH=$PATH:$SPARK\_HOME/bin

ENV PATH=$PATH:$SPARK\_HOME/sbin

Đặt biến môi trường SPARK\_HOME, thêm thư muc bin và sbin của spark vào đường dẫn hệ thống

* Ta build lại image và chạy  container.
* Khi đã ở bên trong container. Ta gõ: spark-shell
* Ta được:



* Như vậy Spark đã được cài đặt và hoạt động chính xác.

1. **Thiết lập cơ sở dữ liệu:**

* ta tiếp tục ghi vào dockerfile những câu lệnh sau:

RUN apt -y install sqlite3

RUN apt -y install python3

RUN apt install nano

* tải sqlite3, python3, và cài trình soạn thảo văn bản nano để phục vụ sau này
* ta build lại image từ dockerfile đã thêm. Và chạy lại Docker container.
* Ta gõ: nano mot\_trieu\_hang.py
* Sau đó gõ đoạn code này vào:

import sqlite3

def create\_and\_populate\_db(db\_name, num\_rows):

    conn = sqlite3.connect(db\_name)

    c = conn.cursor()

    c.execute('''CREATE TABLE IF NOT EXISTS mydata

                (id INTEGER PRIMARY KEY, ho TEXT, ten\_dem TEXT, ten TEXT)''')

    for i in range(num\_rows):

        c.execute("INSERT INTO mydata (ho, ten\_dem, ten) VALUES (?, ?, ?)", ('Vu\_'+str(i), 'Truong\_'+str(i), 'Thanh\_'+str(i)))

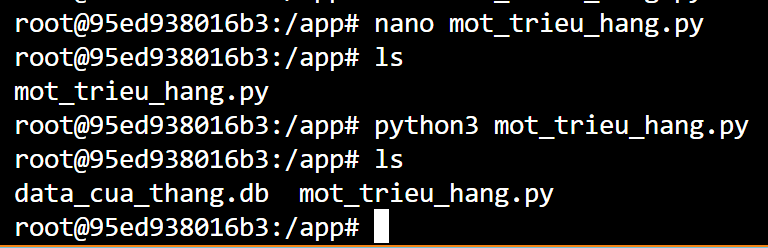
    conn.commit()

    conn.close()

create\_and\_populate\_db('data\_cua\_thanh.db', 1000000)

sau đó nhấn ctrl X để lưu.

Tiếp theo gõ: python3 mot\_trieu\_hang.py



* Sau khi chạy xong đoạn mã python trên. Ta đã điền được vào cơ sở dữ liệu data\_cua\_thang.db

Một triệu hàng.

1. **Cài đặt các phụ thuộc**

* Ta tiếp tục thêm các câu lệnh giúp cài đặt các phụ thuộc:

RUN apt -y install python3-pip

RUN pip install pyspark

RUN wget https://repo1.maven.org/maven2/org/xerial/sqlite-jdbc/3.34.0/sqlite-jdbc-3.34.0.jar

* Đầu tiên ta cài pip cho python3. Sau đó tải pyspark về. và cài JDBC cho sqlite, để giúp kết nối và thực hiện truy vấn từ Spark đến cơ sở dữ liệu.

1. **Cấu hình**

* Để Spark có thể kết nối đến cơ sở dữ liệu ta cần di chuyển thư viện JDBC vào thư mục jars của Spark, đây là thư mục giúp Spark tự động phát hiện và chạy các thư viện phụ trợ nếu cần thiết. Nhập lệnh: RUN mv sqlite-jdbc-3.34.0.jar /opt/spark/jars
* Ta dùng trình soạn thảo văn bản nano. tạo lại file data\_cua\_thanh.db như ở phần 3. Tạo file doc\_10\_dong.py và ghi đoạn code này vào:

from pyspark.sql import SparkSession

spark = SparkSession.builder \

    .appName("Read SQLite with PySpark") \

    .config("spark.driver.extraClassPath", "/path/to/sqlite-jdbc-3.34.0.jar") \

    .getOrCreate()

db\_path = "jdbc:sqlite:data\_cua\_thanh.db"

df = spark.read.format("jdbc") \

    .option("url", db\_path) \

    .option("dbtable", "mydata") \

    .option("driver", "org.sqlite.JDBC") \

    .load()

df.show(10)

spark.stop()

sau đó gõ: python3 doc\_10\_dong.py

ta được kết quả



Như vậy SparkSQL có thể truy cập và truy vấn cơ sở dữ liệu

1. **Chạy truy vấn**

* Tôi sửa lại file mot\_trieu\_hang.py. thêm yếu tố random vào cho việc truy vấn trở nên khách qua. Tôi sửa như sau:

import sqlite3

import random

def create\_and\_populate\_db(db\_name, num\_rows):

    conn = sqlite3.connect(db\_name)

    c = conn.cursor()

    c.execute('''CREATE TABLE IF NOT EXISTS mydata

                (id INTEGER PRIMARY KEY, ho TEXT, ten\_dem TEXT, ten TEXT)''')

    for i in range(num\_rows):

        c.execute("INSERT INTO mydata (ho, ten\_dem, ten) VALUES (?, ?, ?)", ('Vu\_'+str(random.randint(0, 10)), 'Truong\_'+str(random.randint(0, 10)), 'Thanh\_'+str(random.randint(0, 10))))

    conn.commit()

    conn.close()

create\_and\_populate\_db('data\_cua\_thanh.db', 1000000)

và chạy đoạn code này: python3 mot\_trieu\_hang.py

* Tiếp tục dùng trình soạn thảo nano. Tạo file truyvan.py

# -\*- coding: utf-8 -\*-

from pyspark.sql import SparkSession

# Tao SparkSession

spark = SparkSession.builder \

    .appName("SQLite with SparkSQL") \

    .config("spark.driver.extraClassPath", "/opt/spark/jars/sqlite-jdbc-3.34.0.jar") \

    .getOrCreate()

# Duong dan toi file SQLite

db\_path = "jdbc:sqlite:/app/data\_cua\_thanh.db"

# Doc du lieu tu SQLite

df = spark.read.format("jdbc") \

    .option("url", db\_path) \

    .option("dbtable", "mydata") \

    .option("driver", "org.sqlite.JDBC") \

    .load()

# Dang ky DataFrame duoi dang 1 bang tam thoi

df.createOrReplaceTempView("mydata")

# Truy van 1: Lay 10 dong dau tien

print("Truy van 1: Lay 10 dong dau tien

spark.sql("SELECT \* FROM mydata LIMIT 10").show()

# Truy van 2: Dem so luong dong trong bang

print("Truy van 2: Dem so luong dong trong bang")

spark.sql("SELECT COUNT(\*) FROM mydata").show()

# Truy van 3: Lay cac dong co ten la 'Thanh\_1'

print("Truy van 3: Lay cac dang co ten la 'Thanh\_1'")

spark.sql("SELECT \* FROM mydata WHERE ten = 'Thanh\_1'").show()

# Truy van 4: Đem so luong ho 'Vu\_1'

print("Truy van 4: Đem so luong ho 'Vu\_1'")

spark.sql("SELECT COUNT(\*) FROM mydata WHERE ho = 'Vu\_1'").show()

# Truy van 5: Lay 10 ho đau tien va đem so luong tuong ung

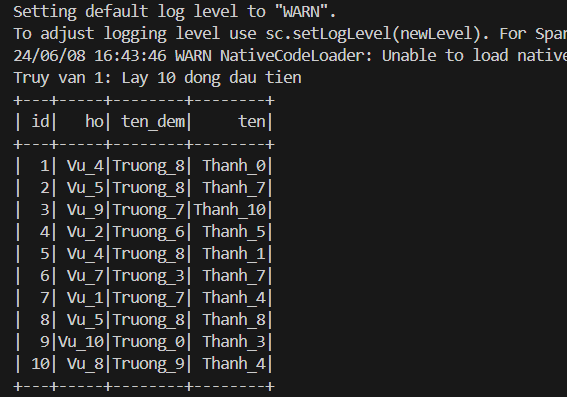
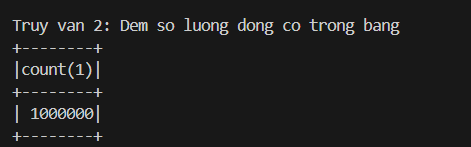
print("Truy van 5: Lay 10 ho đau tien va đem so luong tuong ung")

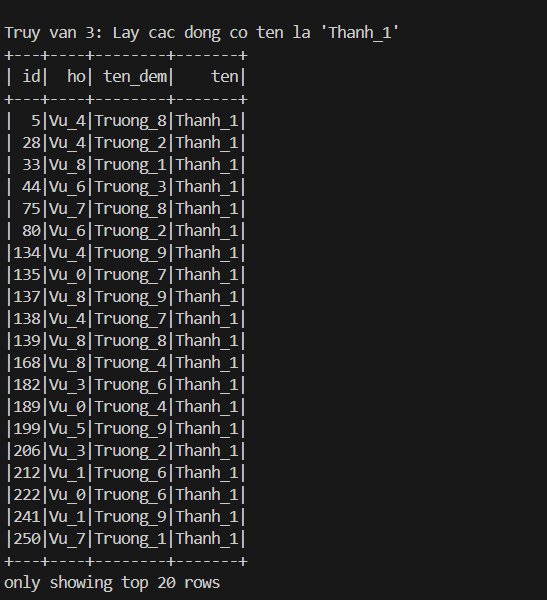
spark.sql("SELECT ho, COUNT(\*) as count FROM mydata GROUP BY ho ORDER BY ho LIMIT 10").show()

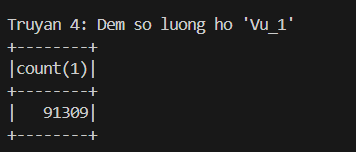
# Kết thúc SparkSession

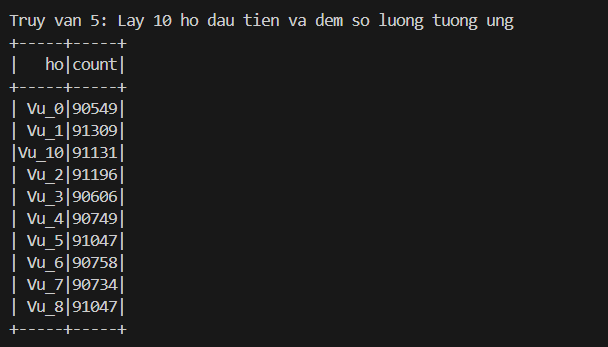
spark.stop()

kết quả:







* Tạo file crud.py bằng nano với nội dung như sau:

# -\*- coding: utf-8 -\*-

from pyspark.sql import SparkSession

from pyspark.sql.functions import lit

# Tao SparkSession

spark = SparkSession.builder \

    .appName("SQLite with SparkSQL CRUD Operations") \

    .config("spark.driver.extraClassPath", "/opt/spark/jars/sqlite-jdbc-3.34.0.jar") \

    .getOrCreate()

# Đuong dan toi file SQLite

db\_path = "jdbc:sqlite:/app/data\_cua\_thang.db"

# Đoc du lieu tu SQLite

df = spark.read.format("jdbc") \

    .option("url", db\_path) \

    .option("dbtable", "mydata") \

    .option("driver", "org.sqlite.JDBC") \

    .load()

# Đang ky DataFrame duoi dang mot bang tam thoi

df.createOrReplaceTempView("mydata")

# CREATE: Them du lieu moi vao bang

new\_data = [("Vu\_1000000", "Truong\_1000000", "Thanh\_1000000"),

            ("Vu\_1000001", "Truong\_1000001", "Thanh\_1000001")]

new\_df = spark.createDataFrame(new\_data, ["ho", "ten\_dem", "ten"])

new\_df.show()

# Them du lieu moi vao DataFrame hien co

df = df.union(new\_df)

# READ: Đoc du lieu tu bang

print("READ: Hien thi 10 dong dau tien")

df.show(10)

# UPDATE: Cap nhat du lieu trong bang

df = df.withColumn("ten", lit("Updated\_Thanh")).where(df.ho == "Vu\_1000000")

df.show()

# DELETE: Xoa du lieu tu bang

df = df.where(df.ho != "Vu\_1000000")

print("DELETE: Xoa cac dong voi ho='Vu\_1000000'")

df.show()

# Luu ket qua lai vao SQLite

df.write.format("jdbc") \

    .option("url", db\_path) \

    .option("dbtable", "mydata") \

    .option("driver", "org.sqlite.JDBC") \

    .mode("overwrite") \

    .save()

# Ket thuc SparkSession

spark.stop()

* A
* A
* A
* A
* A
* A