HƯỚNG DẪN DỮ LIỆU LÊN HDFS VÀ VÀO HBASE VÀ TÌM KIẾM TRUY VẤN BẰNG PHOENIX

Chuẩn bị: File json dữ liệu được crawl về là ree.json (dữ liệu lịch sử giá cổ phiếu REE)

1. Làm sạch dữ liệu (sửa chữa, loại bỏ dữ liệu không chính xác hoặc được định dạng không chính xác, trùng lặp dữ liệu, gán nhãn sai,…)

* Tạo file Python để ta làm sạch dữ liệu chưa đúng

import json

import csv

import re

from datetime import datetime

import pandas as pd

def is\_valid\_date(date\_str):

    try:

        datetime.strptime(date\_str, '%d/%m/%Y')

        return True

    except ValueError:

        return False

def parse\_change(change\_str):

    match = re.match(r'([-\d.]+)\(([-.\d]+) %\)', change\_str)

    if match:

        return float(match.group(1)), float(match.group(2))

    return 0.0, 0.0

with open('D:/IT\_HCMUTE/Hoc\_ki\_6/BIG\_DATA/BIGDATA\_PROJECT/CrawlDataLib/data/ree.json', 'r', encoding='utf-8') as f:    data = json.load(f)

cleaned\_data = []

seen\_dates = set()

for i, record in enumerate(data):

    date = record['Ngày']

    if date in seen\_dates:

        print(f"Bỏ qua bản ghi trùng lặp: {date}")

        continue

    seen\_dates.add(date)

    if not is\_valid\_date(date):

        print(f"Bỏ qua bản ghi có ngày không hợp lệ: {date}")

        continue

    try:

        close\_price = float(record['Giá đóng cửa'])

        adjusted\_price = float(record['Giá điều chỉnh'])

        open\_price = float(record['Giá mở cửa'])

        high\_price = float(record['Giá cao nhất'])

        low\_price = float(record['Giá thấp nhất'])

        matched\_volume = int(record['Khối lượng khớp lệnh'])

        matched\_value = float(record['Giá trị khớp lệnh'])

        agreed\_volume = int(record['Khối lượng thỏa thuận'])

        agreed\_value = float(record['Giá trị thỏa thuận'])

    except ValueError:

        print(f"Bỏ qua bản ghi có giá trị số không hợp lệ: {date}")

        continue

    if (close\_price < 0 or adjusted\_price < 0 or open\_price < 0 or

        high\_price < 0 or low\_price < 0 or matched\_volume < 0 or

        matched\_value < 0 or agreed\_volume < 0 or agreed\_value < 0):

        print(f"Bỏ qua bản ghi có giá trị âm: {date}")

        continue

    if high\_price < low\_price:

        print(f"Bỏ qua bản ghi có giá cao nhất < giá thấp nhất: {date}")

        continue

    if not (low\_price <= close\_price <= high\_price):

        print(f"Bỏ qua bản ghi có giá đóng cửa không hợp lý: {date}")

        continue

    change\_value, change\_percent = parse\_change(record['Thay đổi'])

    id = f"{date.replace('/', '-')}\_{i}"

    cleaned\_data.append({

        'id': id,

        'date': date,

        'closePrice': close\_price,

        'adjustedPrice': adjusted\_price,

        'change': record['Thay đổi'],

        'changeValue': change\_value,

        'changePercent': change\_percent,

        'matchedVolume': matched\_volume,

        'matchedValue': matched\_value,

        'agreedVolume': agreed\_volume,

        'agreedValue': agreed\_value,

        'openPrice': open\_price,

        'highPrice': high\_price,

        'lowPrice': low\_price

    })

df = pd.DataFrame(cleaned\_data)

df['date\_obj'] = pd.to\_datetime(df['date'], format='%d/%m/%Y')

df = df.sort\_values('date\_obj').drop(columns=['date\_obj'])

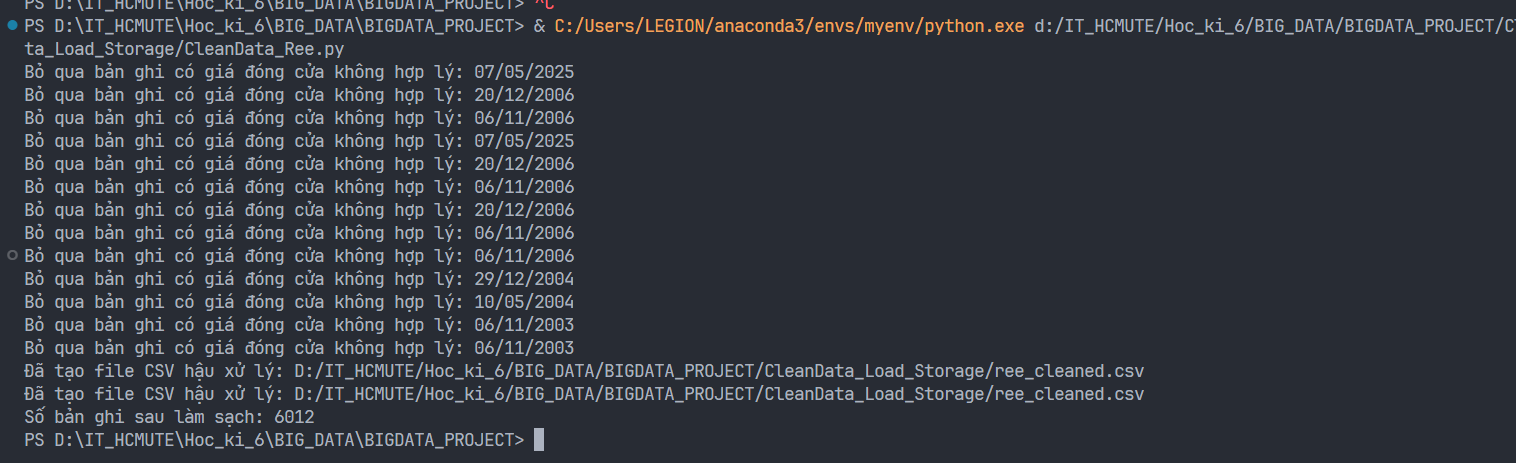
csv\_file = 'D:/IT\_HCMUTE/Hoc\_ki\_6/BIG\_DATA/BIGDATA\_PROJECT/CleanData\_Load\_Storage/ree\_cleaned.csv'

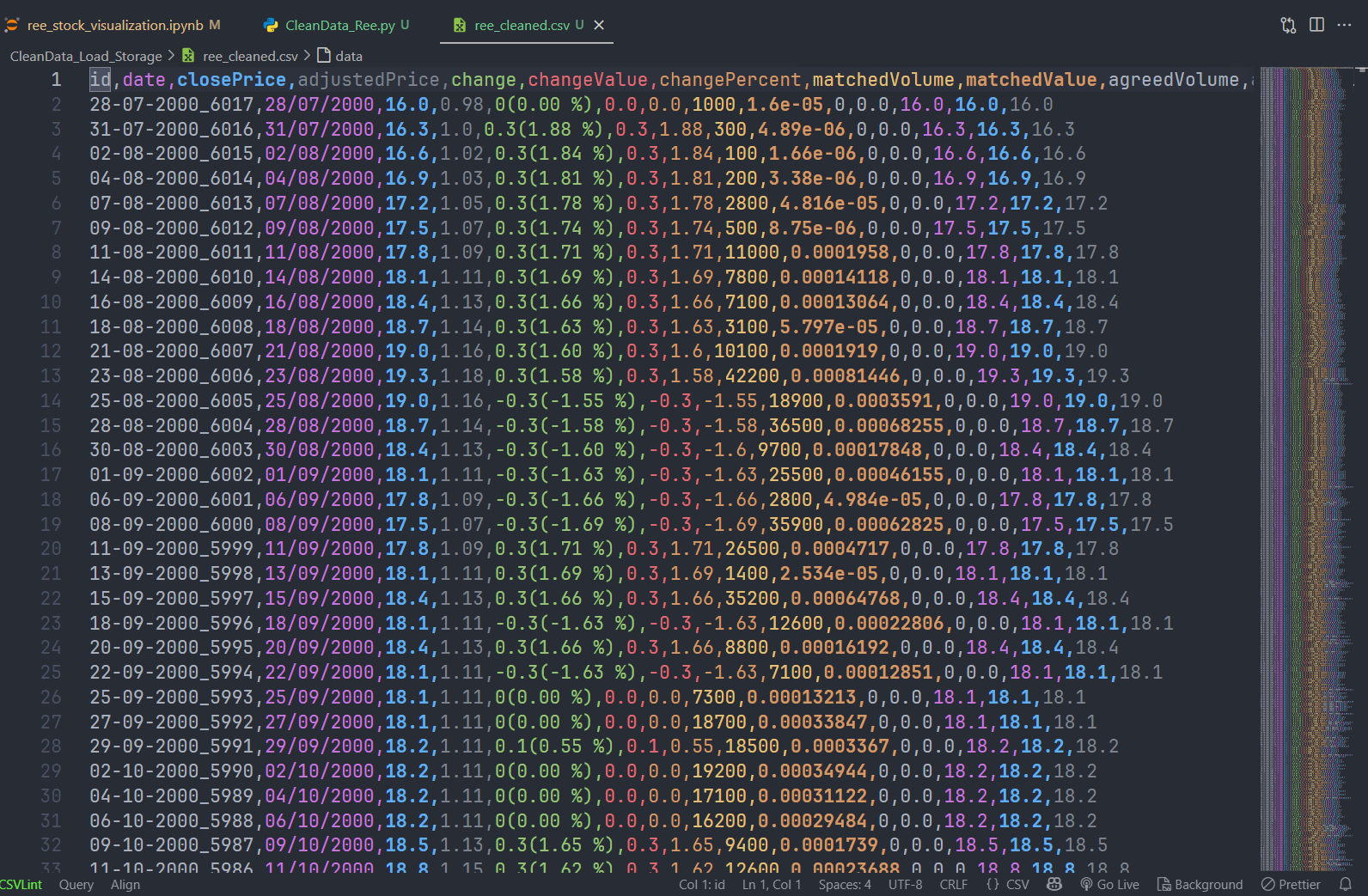
df.to\_csv(csv\_file, index=False, encoding='utf-8')

print(f"Đã tạo file CSV hậu xử lý: {csv\_file}")

print(f"Số bản ghi sau làm sạch: {len(df)}")

* Dữ liệu sau khi làm sạch

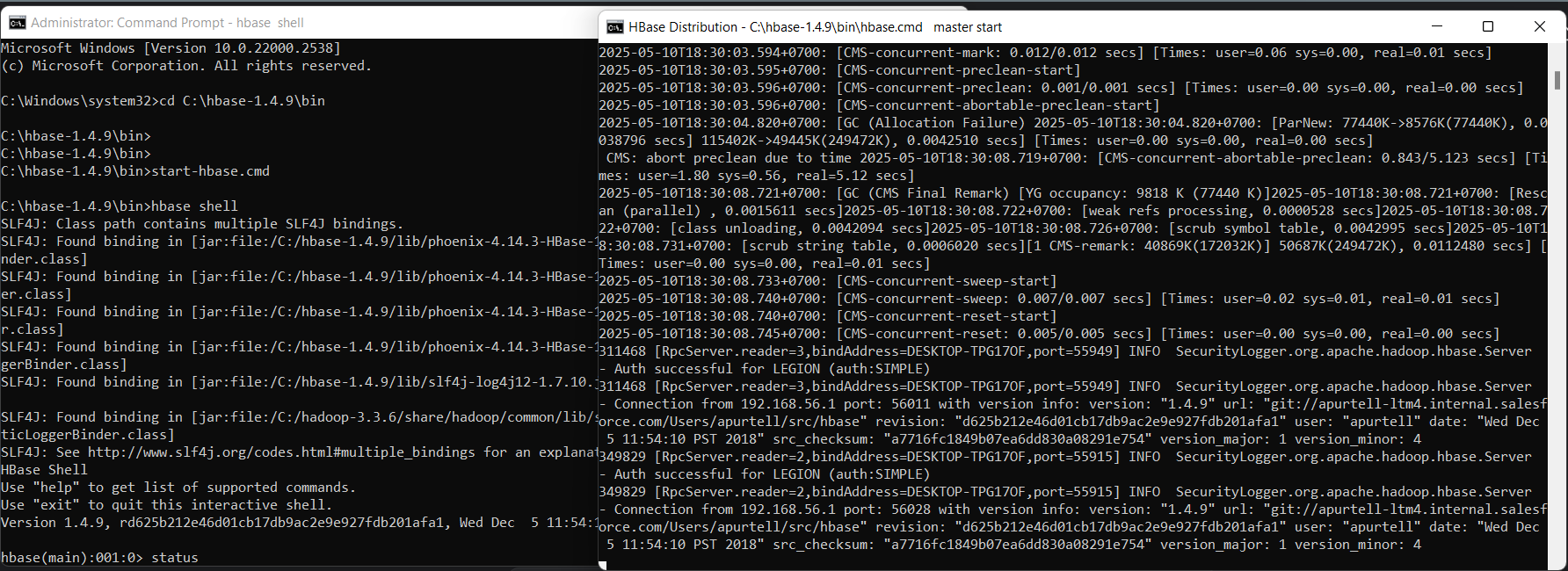


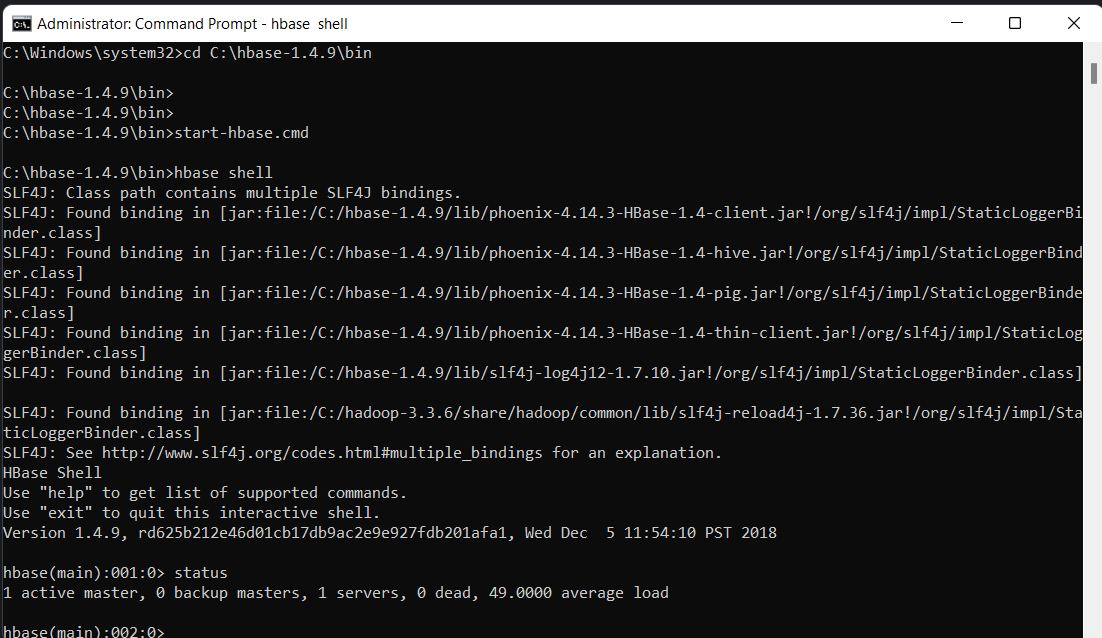


1. Tạo bảng lưu trữ dữ liệu trong Hbase

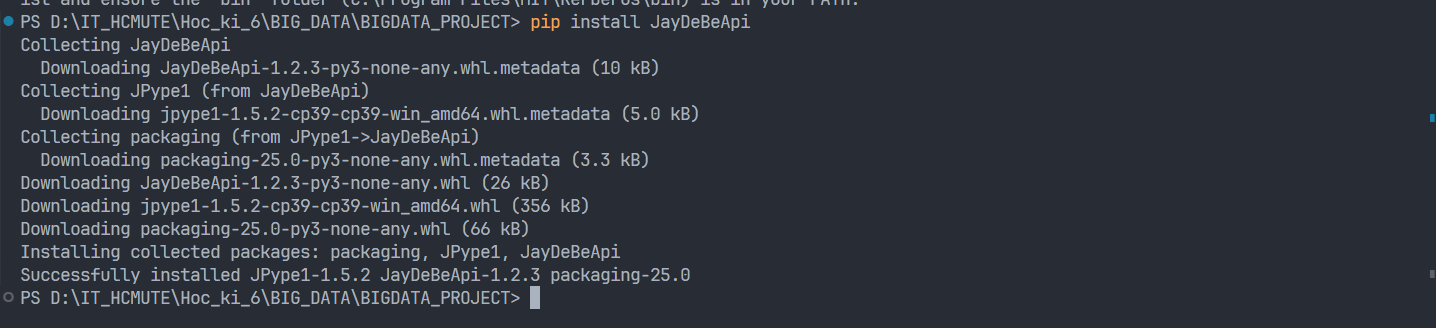
* Khởi chạy Hbase







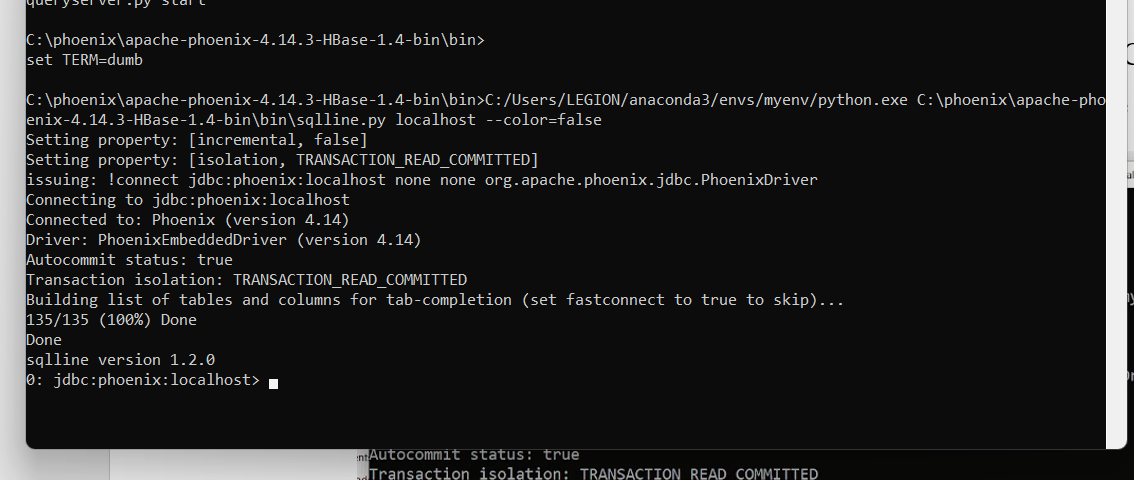
* Cài thư viện JayDeBeApi để kết nối thông qua JDBC Phoenix



* Khởi chạy Phoenix để tạo bảng lưu trữ trong Hbase

set TERM=dumb

C:/Users/LEGION/anaconda3/envs/myenv/python.exe C:\phoenix\apache-phoenix-4.14.3-HBase-1.4-bin\bin\sqlline.py localhost --color=false



CREATE TABLE ree (

id VARCHAR NOT NULL PRIMARY KEY,

date VARCHAR,

close\_price FLOAT,

adjusted\_price FLOAT,

change VARCHAR,

change\_value FLOAT,

change\_percent FLOAT,

matched\_volume INTEGER,

matched\_value FLOAT,

agreed\_volume INTEGER,

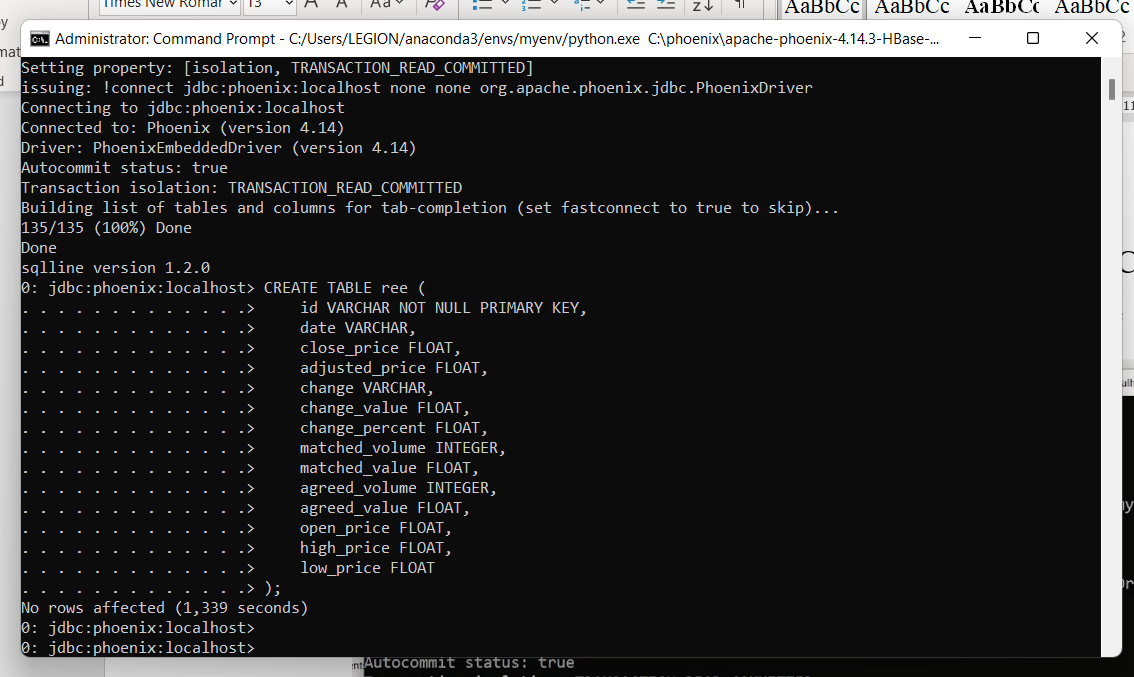
agreed\_value FLOAT,

open\_price FLOAT,

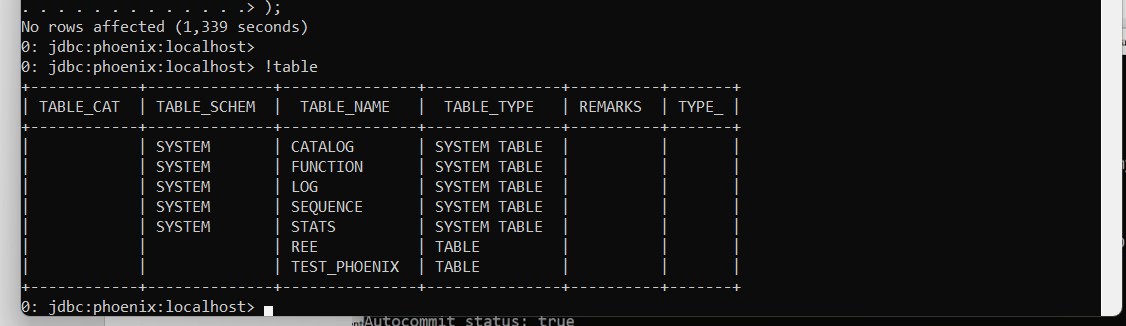
high\_price FLOAT,

low\_price FLOAT

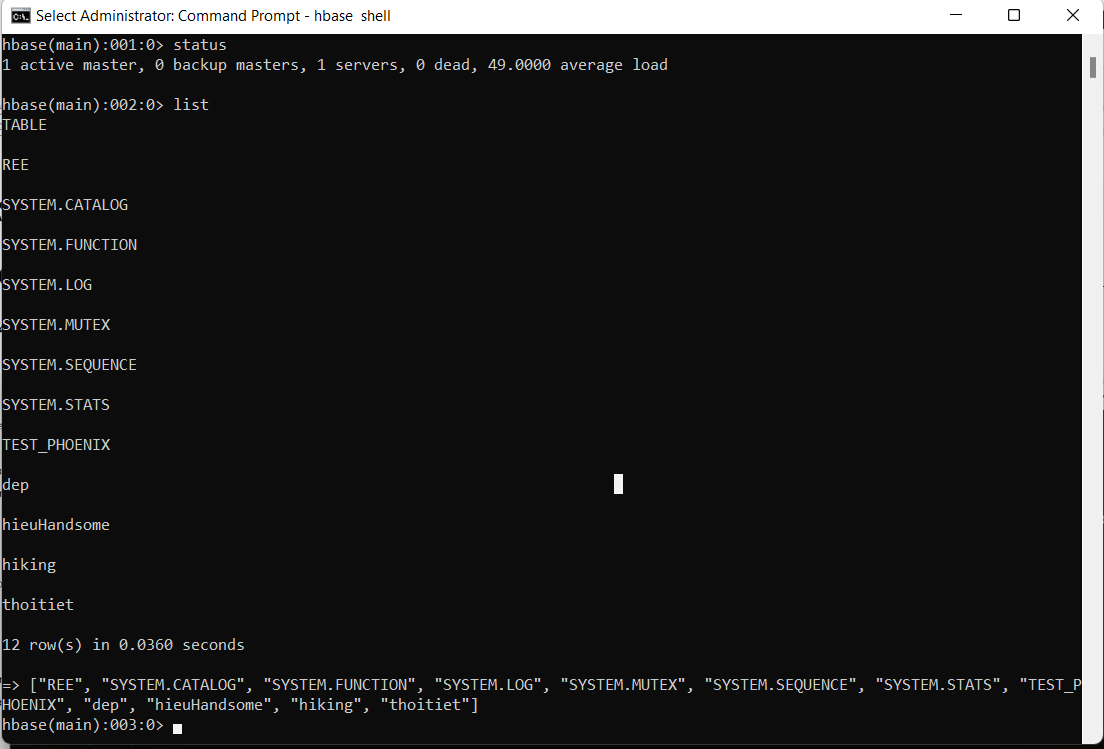
);



* Tạo bảng thành công
* Kiểm tra các bảng trong Phoenix và Hbase



* Phoenix có bảng REE vừa tạo



* Trong Hbase cũng đã có bảng REE vừa tạo

1. Viết file Python kết nối đến Phoenix để tiến hành đưa dữ liệu vào

import jaydebeapi

import csv

import sys

try:

    conn = jaydebeapi.connect(

        'org.apache.phoenix.jdbc.PhoenixDriver',

        'jdbc:phoenix:localhost',

        [],

        'D:/IT\_HCMUTE/Hoc\_ki\_6/BIG\_DATA/BIGDATA\_PROJECT/CleanData\_Load\_Storage/phoenix-4.14.3-HBase-1.4-client.jar'

    )

    cursor = conn.cursor()

    print("Kết nối Phoenix thành công!")

except Exception as e:

    print(f"Lỗi kết nối Phoenix: {e}")

    sys.exit(1)

csv\_file = 'D:/IT\_HCMUTE/Hoc\_ki\_6/BIG\_DATA/BIGDATA\_PROJECT/CleanData\_Load\_Storage/ree\_cleaned.csv'

try:

    with open(csv\_file, 'r', encoding='utf-8') as f:

        reader = csv.DictReader(f)

        upsert\_query = """

        UPSERT INTO ree (

            id, date, close\_price, adjusted\_price, change, change\_value, change\_percent,

            matched\_volume, matched\_value, agreed\_volume, agreed\_value, open\_price, high\_price, low\_price

        ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

        """

        for row in reader:

            try:

                cursor.execute(upsert\_query, [

                    row['id'],

                    row['date'],

                    float(row['closePrice']),

                    float(row['adjustedPrice']),

                    row['change'],

                    float(row['changeValue']),

                    float(row['changePercent']),

                    int(row['matchedVolume']),

                    float(row['matchedValue']),

                    int(row['agreedVolume']),

                    float(row['agreedValue']),

                    float(row['openPrice']),

                    float(row['highPrice']),

                    float(row['lowPrice'])

                ])

            except Exception as e:

                print(f"Lỗi khi nhập bản ghi {row['id']}: {e}")

                continue

        conn.commit()

        print(f"Đã nhập dữ liệu từ {csv\_file} vào Phoenix!")

except Exception as e:

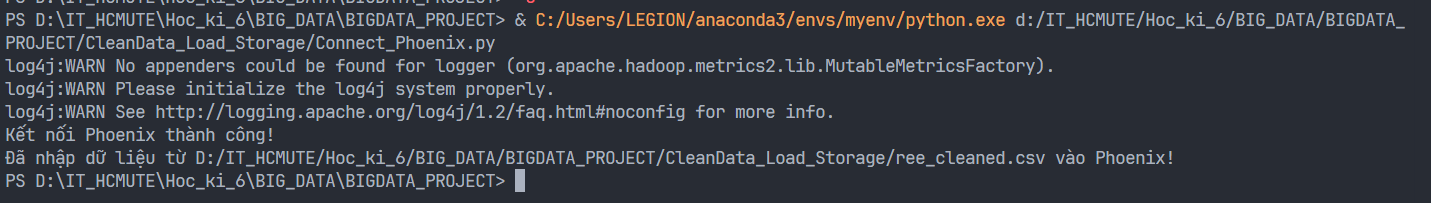
    print(f"Lỗi khi nhập dữ liệu: {e}")

finally:

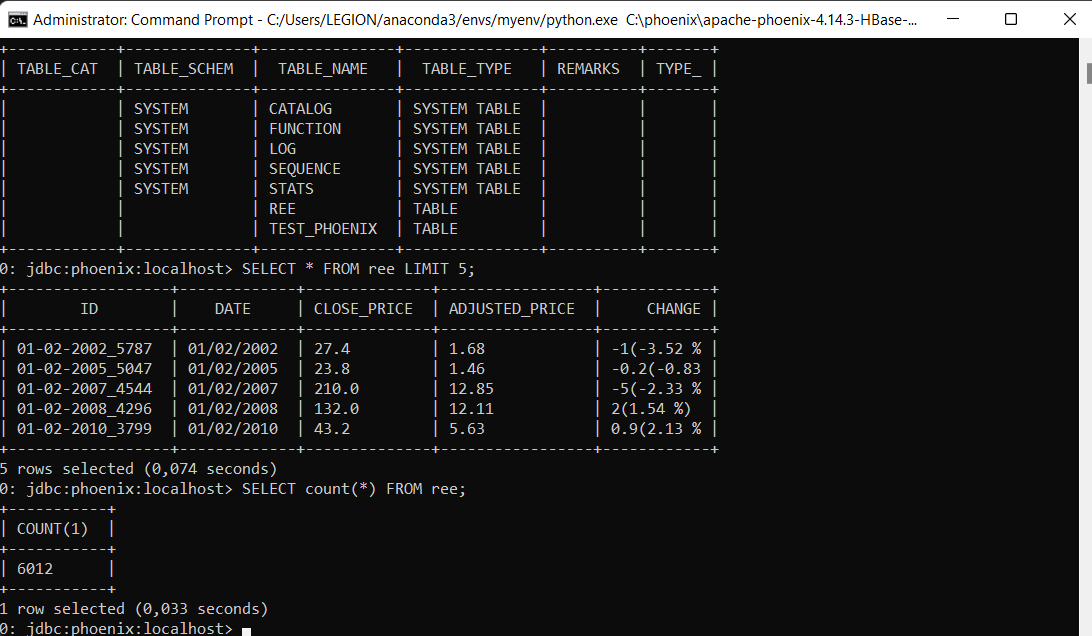
    cursor.close()

    conn.close()

* Nhập dữ liệu thành công

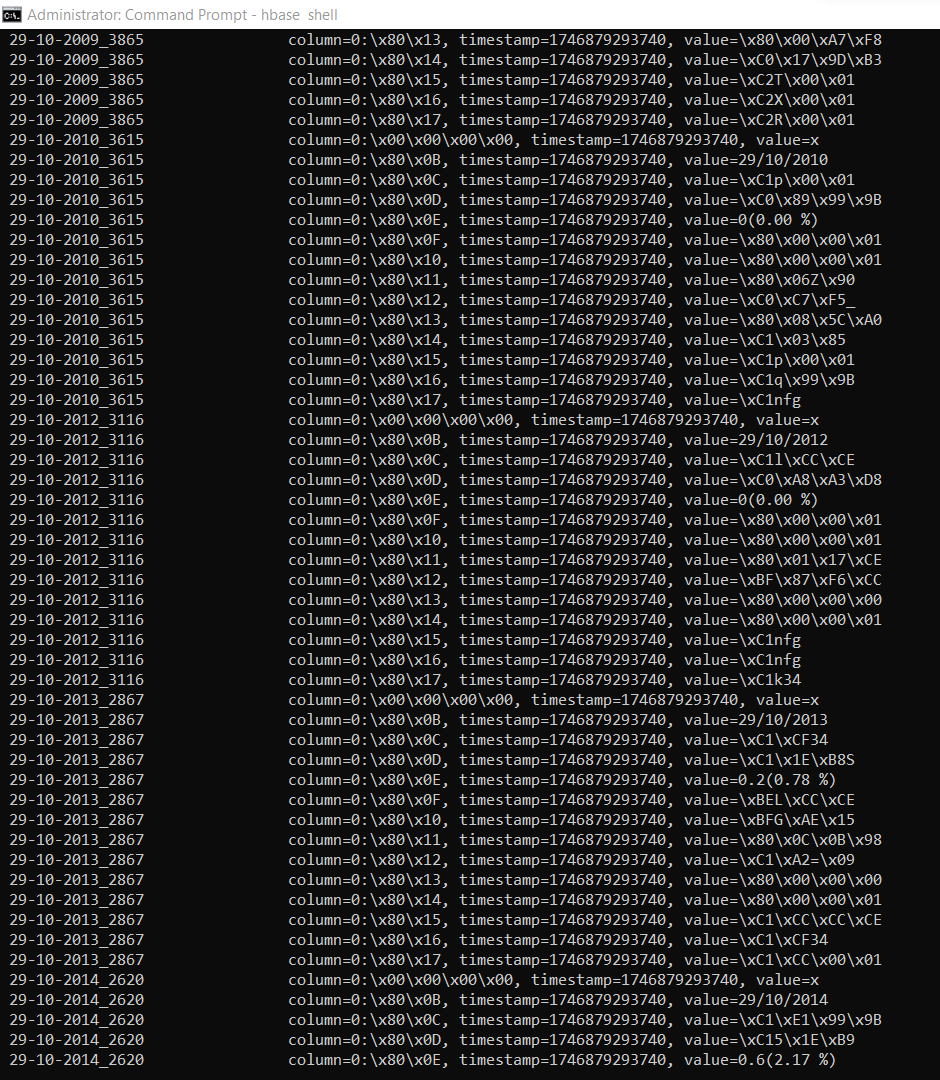


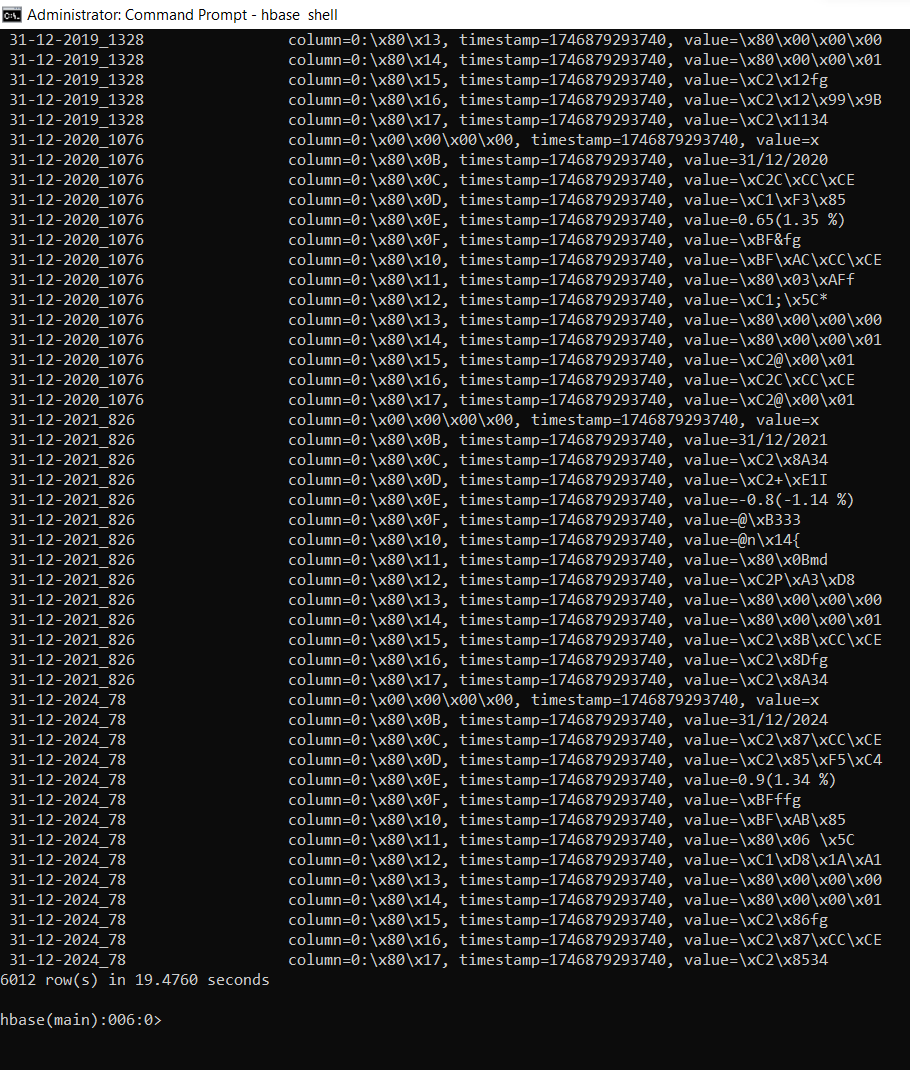
* Kiểm tra trong Phoenix



* Kiểm tra trong Hbase

$ scan ‘REE’

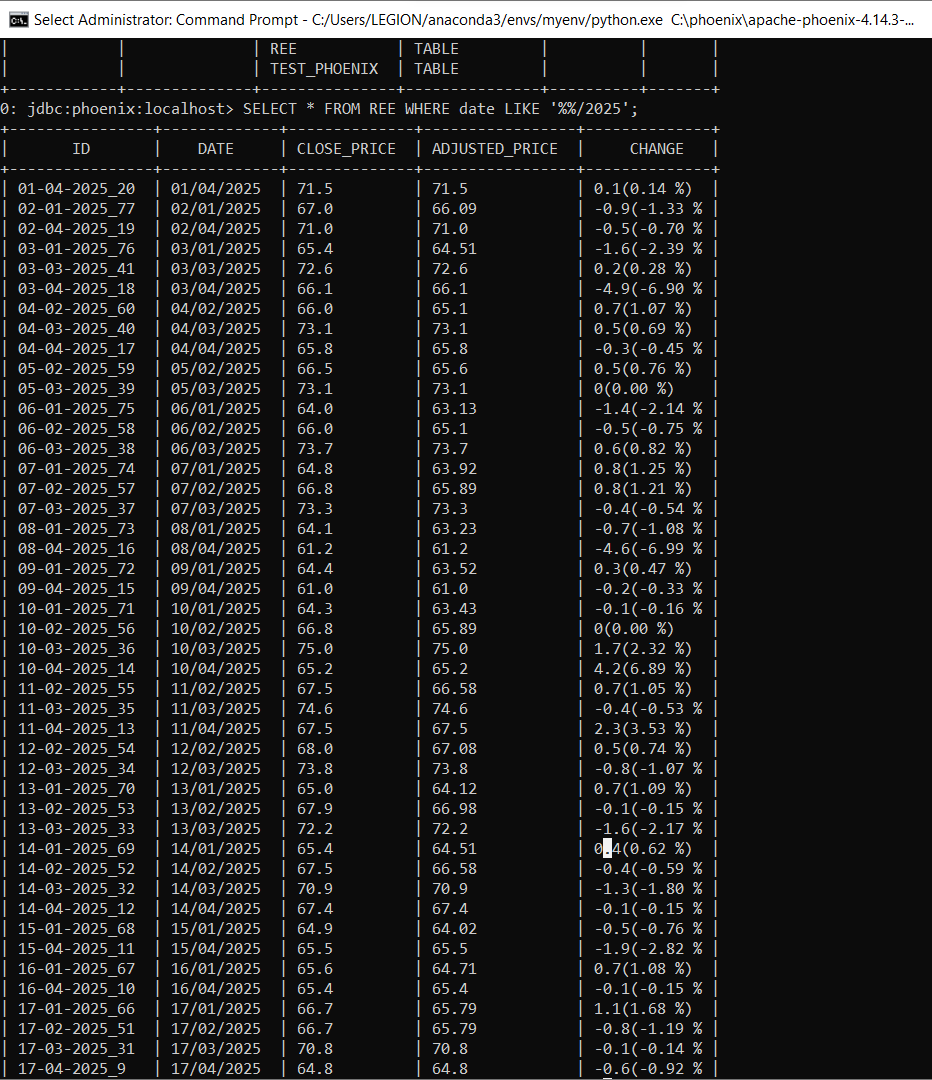




1. Tìm kiếm và truy vấn cơ sở dữ liệu

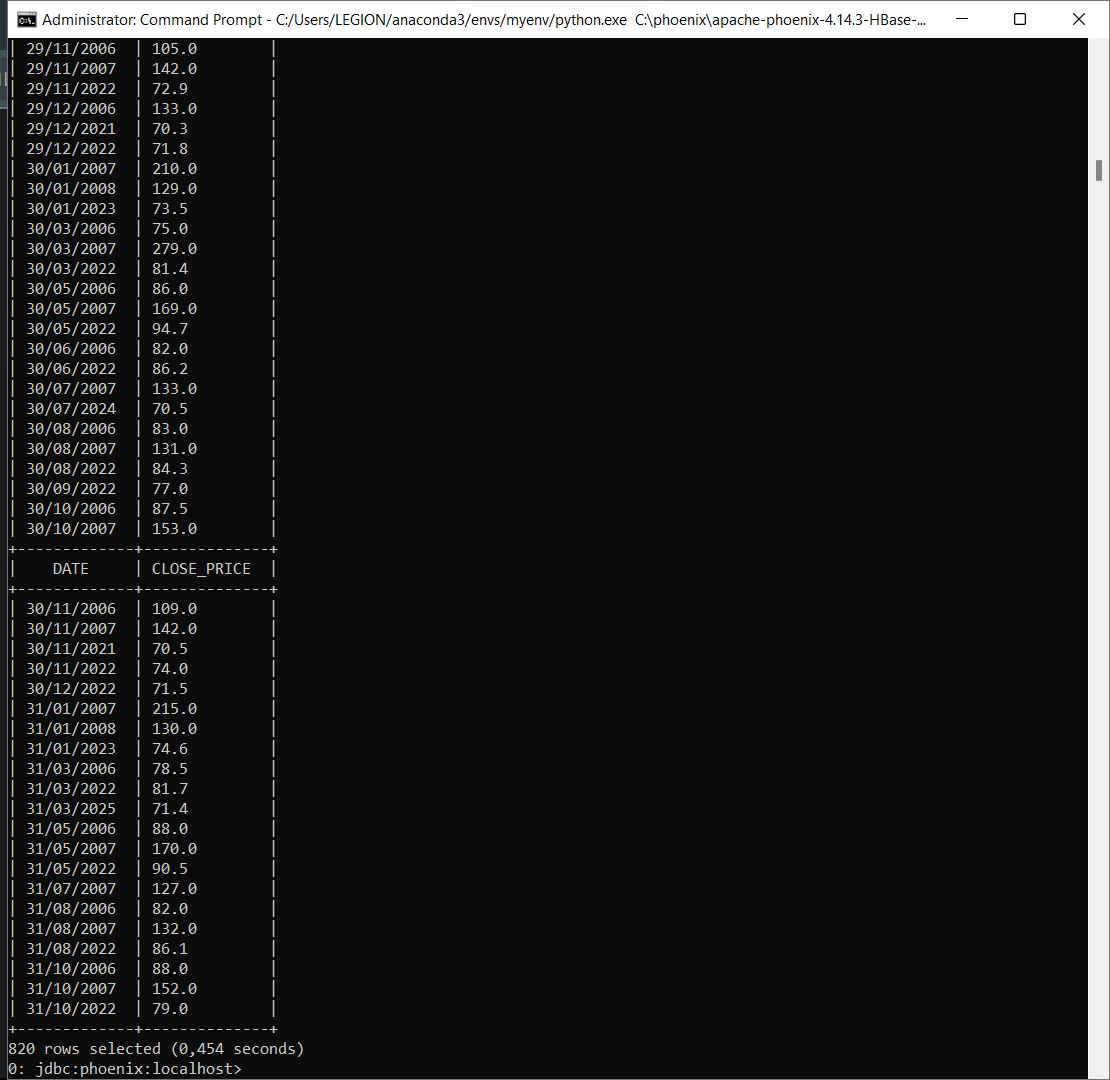
* Truy vấn dữ liệu năm 2025

$ SELECT \* FROM REE WHERE date LIKE '%%/2025';



* Tìm ngày có giá đóng cửa trên 70

$ SELECT date, close\_price FROM REE WHERE close\_price > 70;

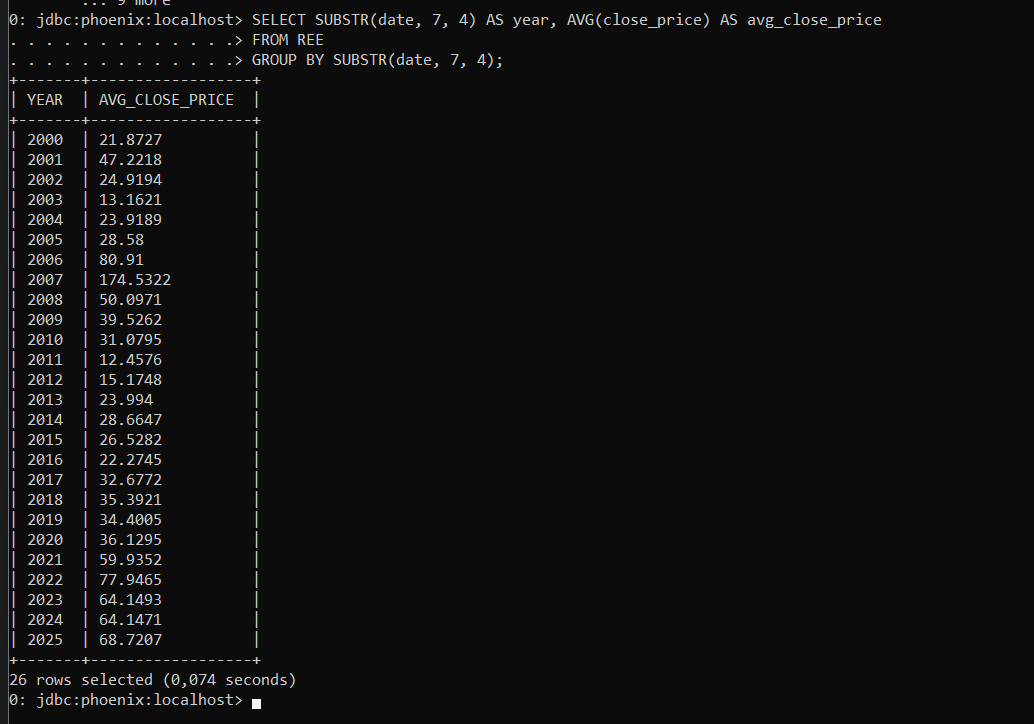


* Tính giá đóng cửa trung bình theo năm

$ SELECT SUBSTR(date, 7, 4) AS year, AVG(close\_price) AS avg\_close\_price

FROM REE

GROUP BY SUBSTR(date, 7, 4);



* Lấy 5 ngày có khối lượng khớp lệnh cao nhất

$ SELECT date, matched\_volume FROM REE ORDER BY matched\_volume DESC LIMIT 5;

