Debdas Ghosh & Connor Wilson

INFSCI 1540 Data Engineering

22 April 2021

Data Engineering Behind Stock Analysis

**Overview**

The project enables data-driven decision making and market analysis of stock trends for companies listed in the S&P 500. Utilizing data drawn from Wikipedia, Yahoo Finance, Stocktwits, and Statista, the repository combines trade information, quarterly revenues, and investor and analyst sentiments to provide a more comprehensive view of stocks within the index. From day to week to quarter, headquarters location to industry to sub-sector, the warehouse provides an aggregated view of the data along multiple dimensions, able to drill-down along a variety of paths for more granular data analysis. In this project, we will use these data to find answer of the following queries:

1. Which state/city has the most number of volumes of stocks for the S&P 500 American companies on date/week/month/quarter/year basis?
2. Revenues distribution of companies Sector/Industry wise and Quarter/Year wise
3. Market sentiments about the companies?

We have hosted this project repository on Github available at <https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis>

**Project Structure (Docker)**

Utilizing the Docker platform to run components of the data pipeline in separate containers, the project is comprised of several key technologies. At the highest level, an Apache web server is used to support two instances of phpMyAdmin, each corresponding to a MySQL database: an instance for the operational database containing raw data (ODB database) and another for the aggregated data warehouse (DW database). A Kafka broker container is utilized for streaming data from the operational database to the data warehouse, while a ZooKeeper container manages storing the streaming data.

Docker Composer source: <https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis/blob/main/docker-compose.yml>

**Data ETL**

|  |  |  |
| --- | --- | --- |
| **Source** | **Type** | **Brief Summary** |
| **Wikipedia** | Semi-Structured | List of S&P 500 Stock market index companies |
| **Yahoo Finance** | Structured | Daily stock market data for the companies from S&P 500 list ranging from 1 January 2017 to 8 April 2021 |
| **StockTwits** | Unstructured | Tweets related to the stock market are fetched from stock market social site. |
| **Statista** | Structured | Quarterly revenues of the companies listed from S&P500 list. |

Summary of sources of data used in the project

Beginning with the collection of companies tracked in the database, the company information for the S&P 500 index is pulled from Wikipedia, utilizing the stock symbol, security (company name), sector, sub-industry, and headquarters location listings for each company. While each company’s name, associated stock symbol, sector, and sub-industry are used as-is, the headquarters location is separated into state and country, filtering out companies not headquartered in the United States. This information is then outputted to a CSV spreadsheet using a script written in R and later loaded into the operational database and data warehouse by a Python script.

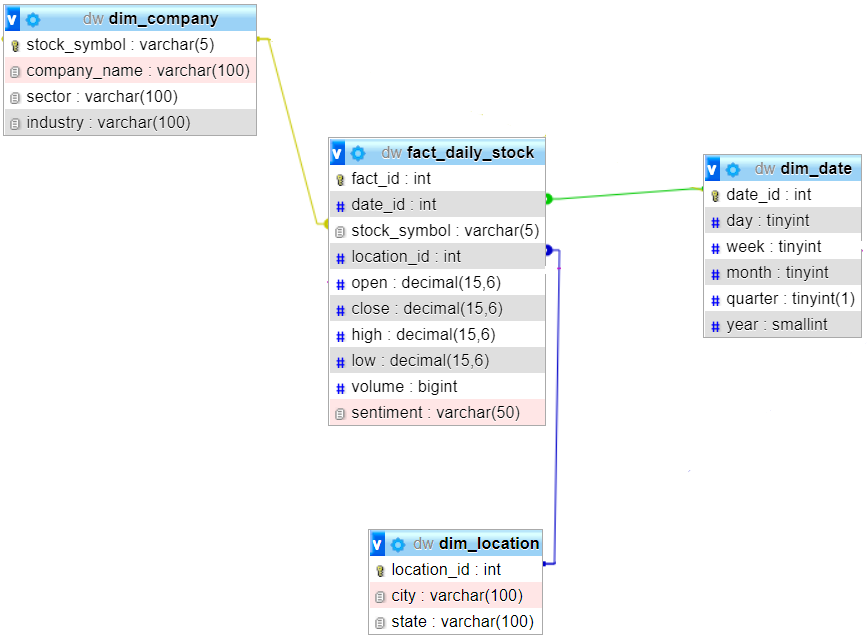
Next, the quarterly revenues for select companies are downloaded from the statistics site Statista based on available information. From the Microsoft Excel files provided by the website, the data for quarterly revenue (in billions of dollars) is extracted along with financial quarter, fiscal year, and stock symbol and outputted to a combined CSV file through a script written in R. In a similar manner to company information, the data is inserted into both databases using the aforementioned Python script, relegated to a separate table in the operational database and appearing in a derivative format in an aggregated FACT table in the data warehouse.

Subsequently, daily stock information provided by Yahoo Finance is queried from the beginning of the 2017 fiscal year to April 2021 for each company included in the S&P 500 index (per Wikipedia), with information on the date, opening stock price, closing stock price, highest and lowest sale prices, and volume of stocks sold aggregated into a single, combined CSV spreadsheet using R script. In combination with the previous spreadsheets, this is the last data source loaded into each repository by Python scripts.

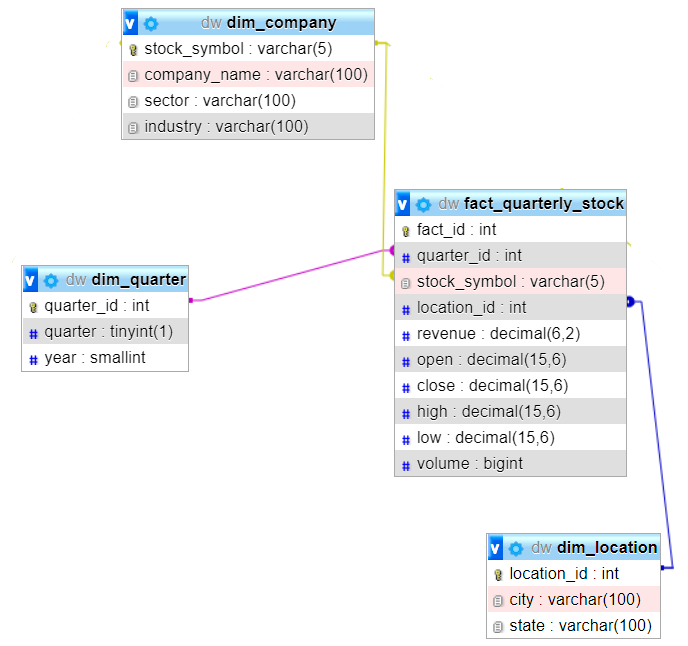
Finally, the Stocktwits API is used to collection information about the latest tweets and sentiments in relation to supported stocks (i.e., Apple, inc.), which are collected into a series of JSON files in a separate directory through a script written in R. The problem of this API we faced during the implementation is it gives connection timeout error after 3 minutes and it blocks subsequent action for another 1 hour. So, we have managed to extract 6000 tweets for Apple Inc and Amazon each, but it can be replicated with any S&P company. The data in the JSON files is streamed from an Apache Kafka producer into a consumer for inserting the raw data into the operational database, with updates subsequently streamed to another Kafka consumer used for aggregating information from the operational database and inserting it into the data warehouse. Utilizing sentiment data collected from the Stocktwits API, a sentiment score considering the “bearishness,” “bullishness,” or “neutrality” of each collected “twit” is aggregated for each company’s stock.

**STAR Schema**

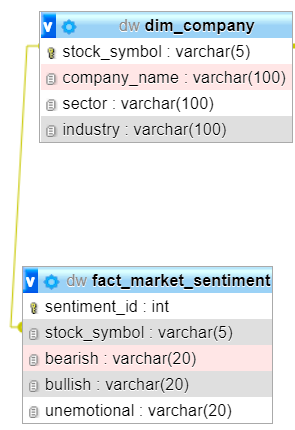
The STAR schema of the data warehouse centers around three FACT tables. The first, the “fact\_daily\_stock,” table includes three dimensions: date, company (stock), and (headquarters) location, describing several measures: the opening and closing prices of a stock, its highest and lowest sale prices, and the volume of trades.



The second FACT table, “fact\_quarterly\_stock,” similarly describes these same measures with the addition of the corresponding quarterly revenue of the stock’s company for the time period along three similar dimensions: quarter, company (stock), and (headquarters) location.

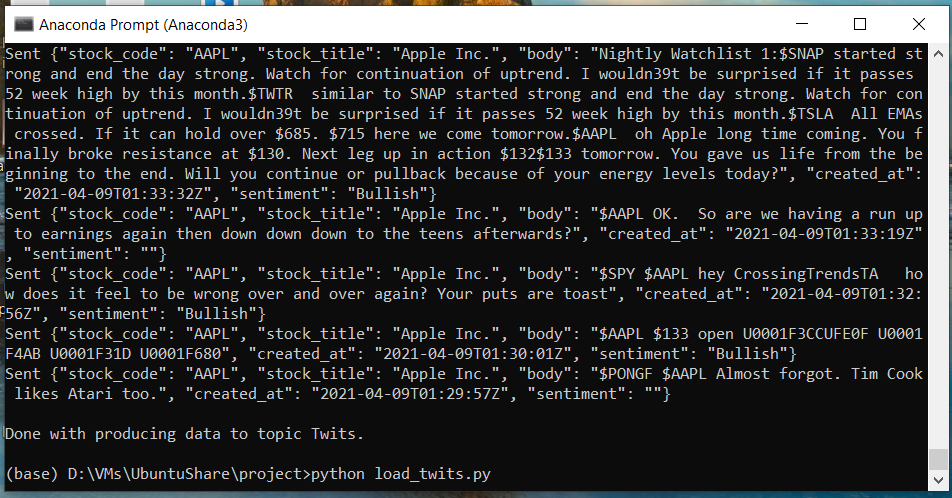


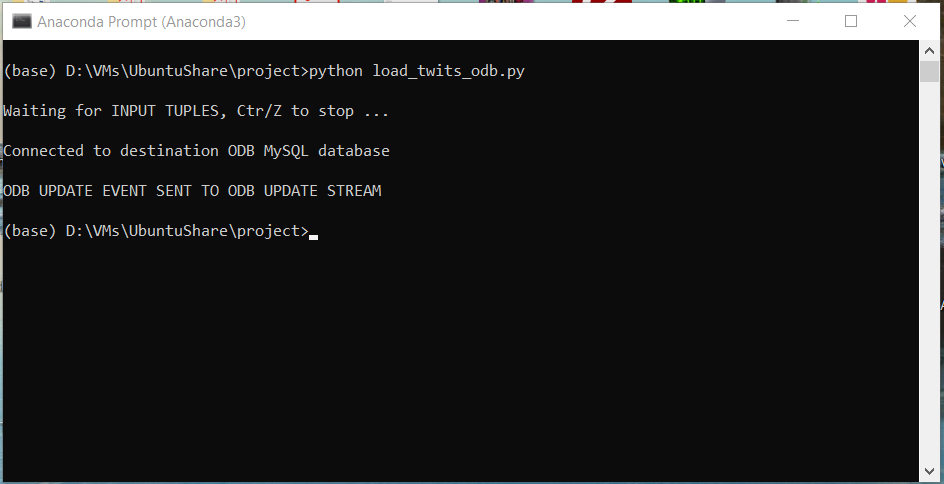
Finally, the third “fact\_market\_sentiment” table describes the “bearishness,” “bullishness,” and “unemotionality” of sentiments expressed about a stock along the single dimension of (stock) company.

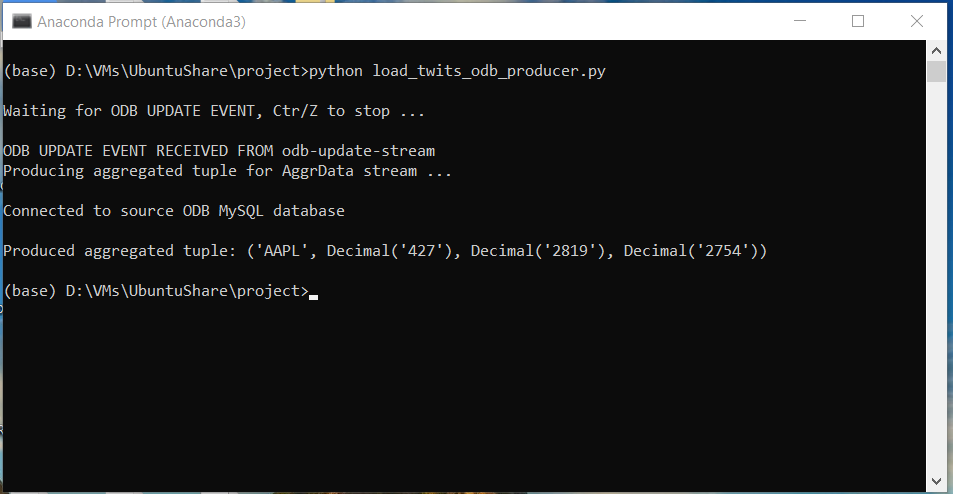


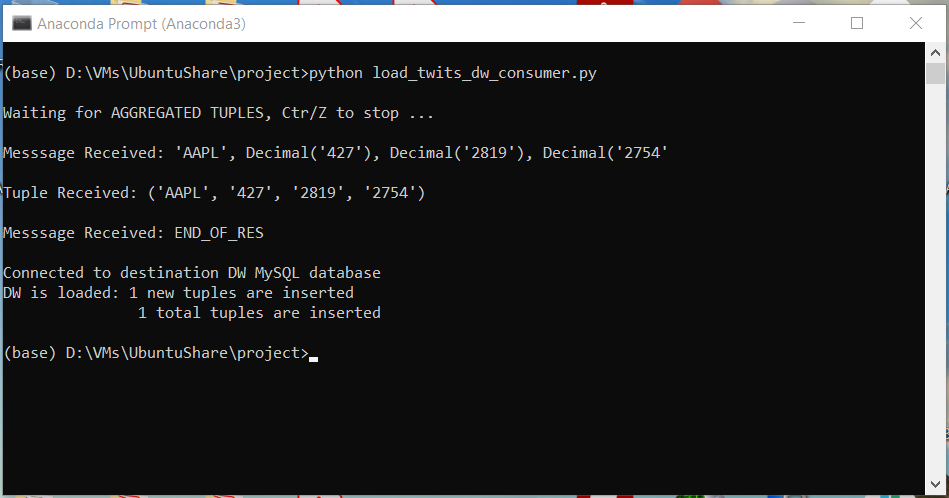
In total, the schema includes four unique dimension tables. The table “dim\_quarter” represents the unique fiscal quarters for data contained in the warehouse, the “dim\_date” table describes all unique dates of stock trade data stored in the warehouse, split into component days, weeks, months, fiscal quarters, and years. The “dim\_location” table describes the city and state of various company’s headquarters, while “dim\_company” records information about S&P 500 companies, including their name, stock symbol, sector, and industry (sub-sector).

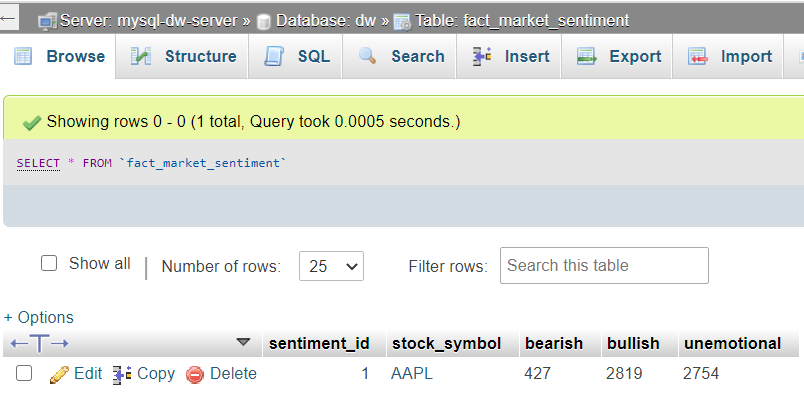
**Data Streaming:**







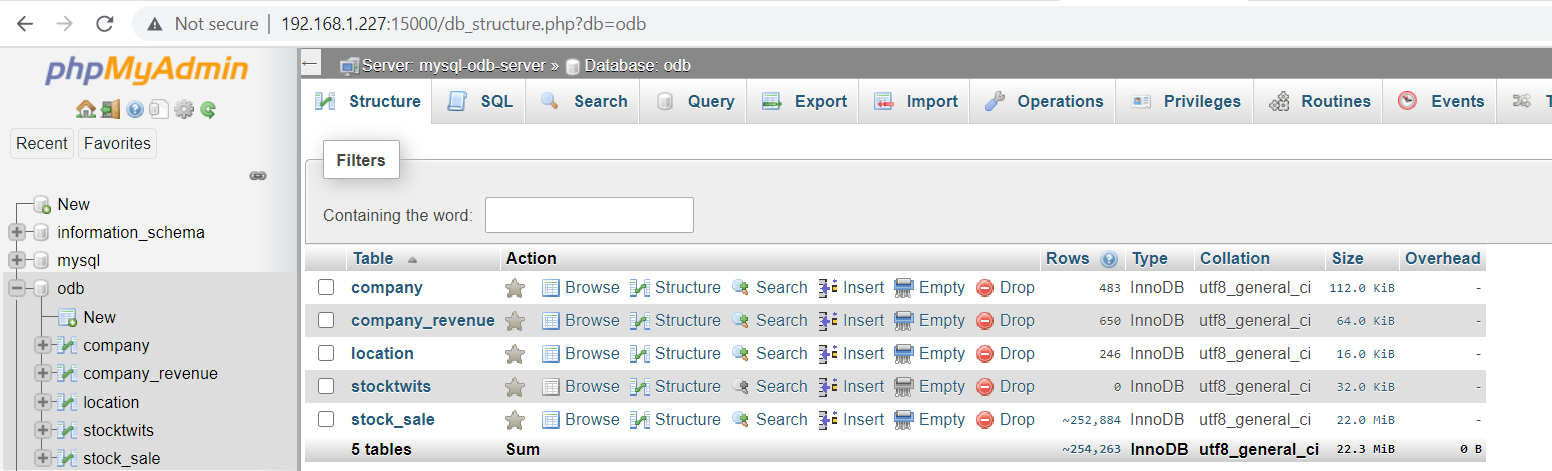




**Summary Tables**

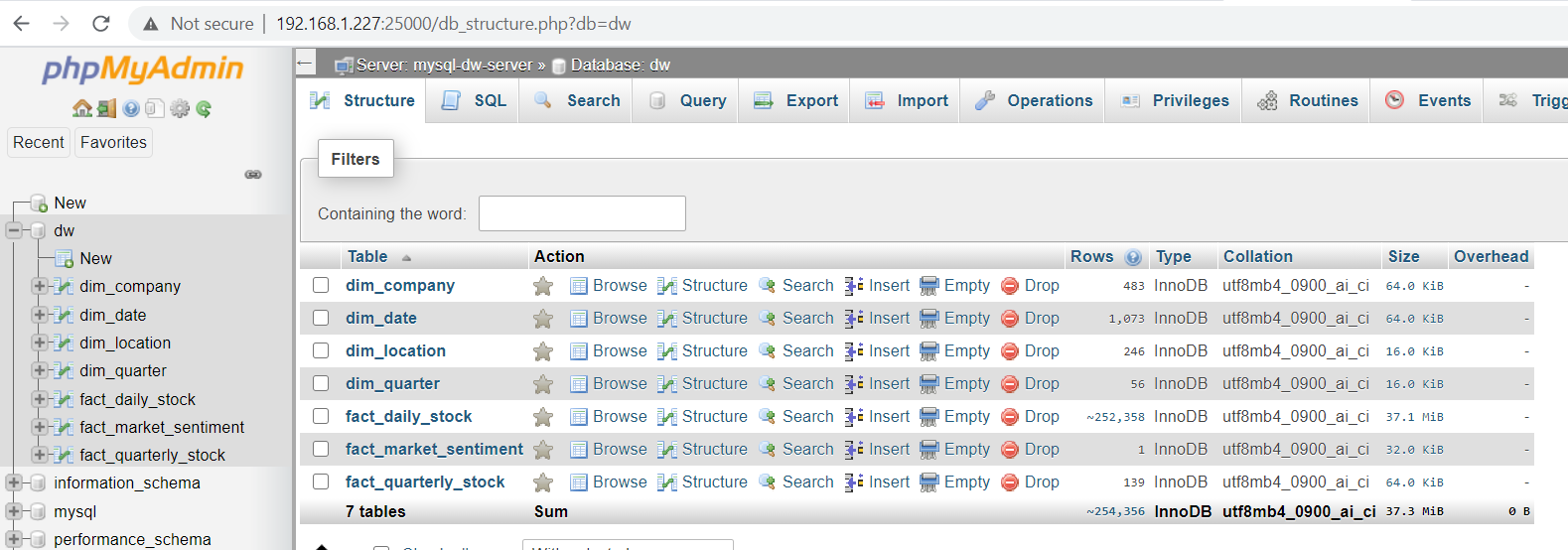
<https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis/blob/main/load_odb.py>

<https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis/blob/main/create_odb.sql>



<https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis/blob/main/load_dw.py>

<https://github.com/debdasghosh/Data-Engineering-Behind-Stock-Analysis/blob/main/create_dw.sql>



**Supported Queries**

-- Which state/city has the most number of volumes of stocks for the S&P 500 American companies on date/week/month/quarter/year basis?

-- Which state has the most number of volumes of stocks for the S&P 500 American companies in a particular day?

SELECT

dl.state, sum(rc.volume) AS tot\_vol

from dim\_company sc

INNER JOIN fact\_daily\_stock rc on rc.stock\_symbol=sc.stock\_symbol

INNER JOIN dim\_location dl on dl.location\_id=rc.location\_id

INNER JOIN dim\_date d on d.date\_id = rc.date\_id

WHERE d.year = 2017 and d.month = 1 and d.day = 3

GROUP BY dl.state

ORDER BY tot\_vol DESC;

-- Revenues distribution of companies Sector/Industry wise and Quarter/Year wise

-- Revenues per Sector in a particular day

SELECT

sc.sector, sum(rc.close \* rc.volume) AS revenue

from dim\_company sc

INNER JOIN fact\_daily\_stock rc on rc.stock\_symbol=sc.stock\_symbol

INNER JOIN dim\_location dl on dl.location\_id=rc.location\_id

INNER JOIN dim\_date d on d.date\_id = rc.date\_id

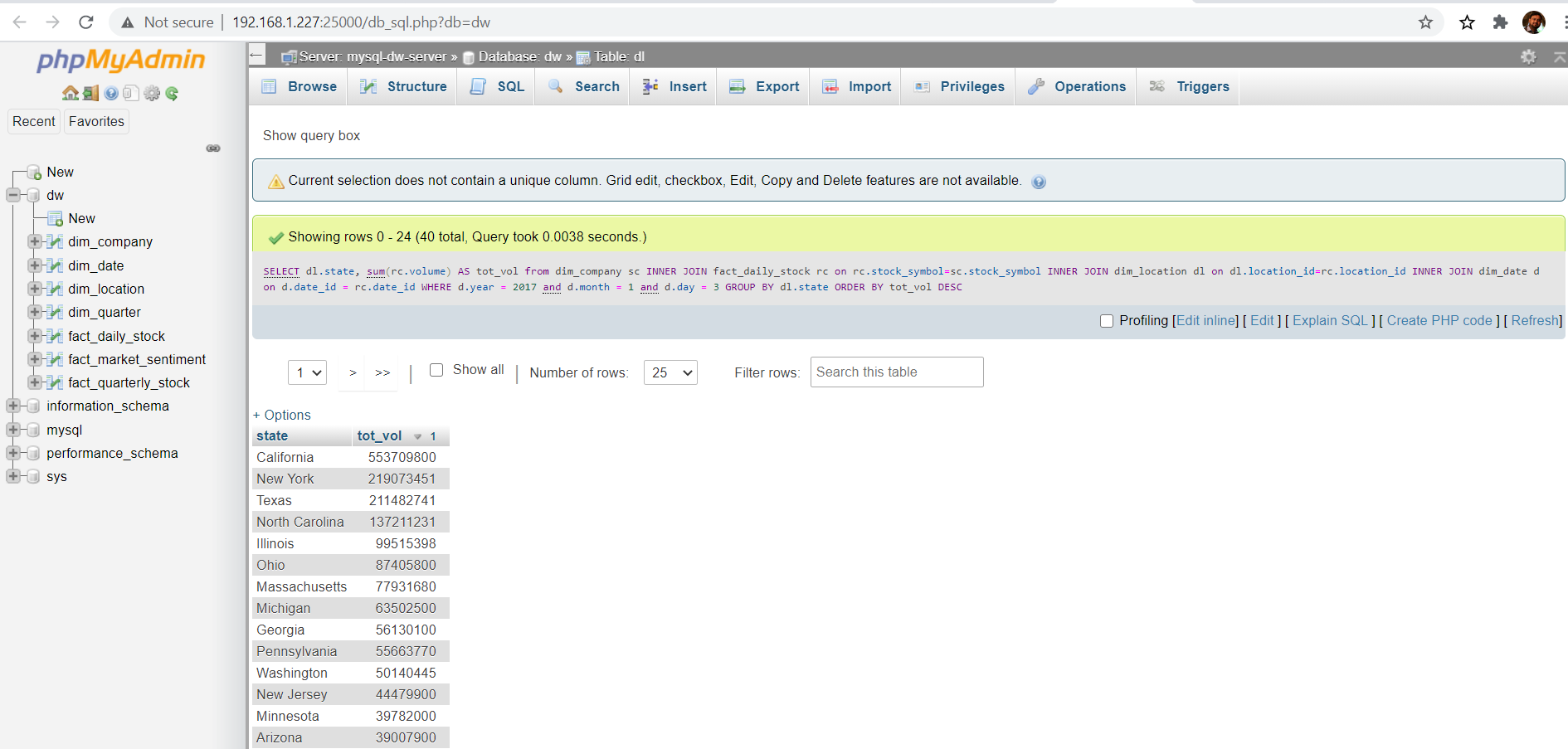
WHERE d.year = 2017 and d.month = 1 and d.day = 3

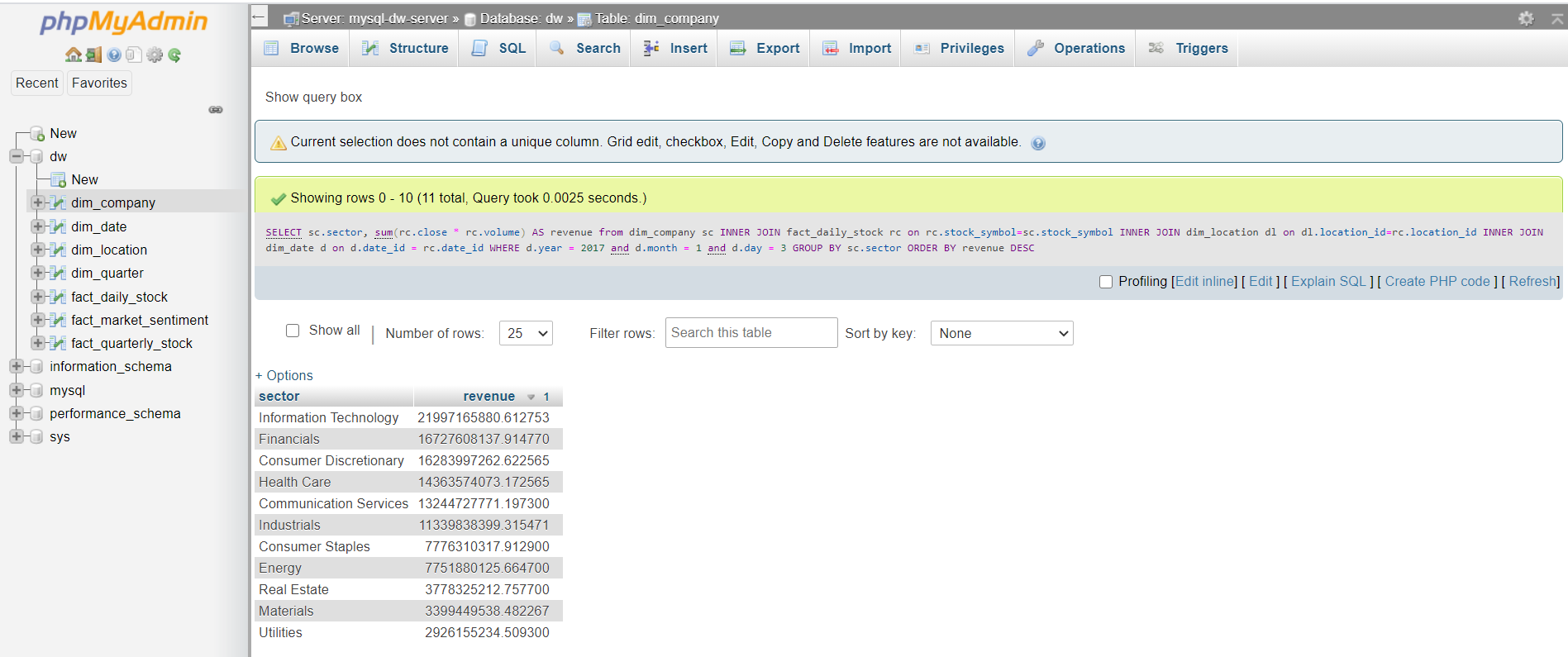
GROUP BY sc.sector

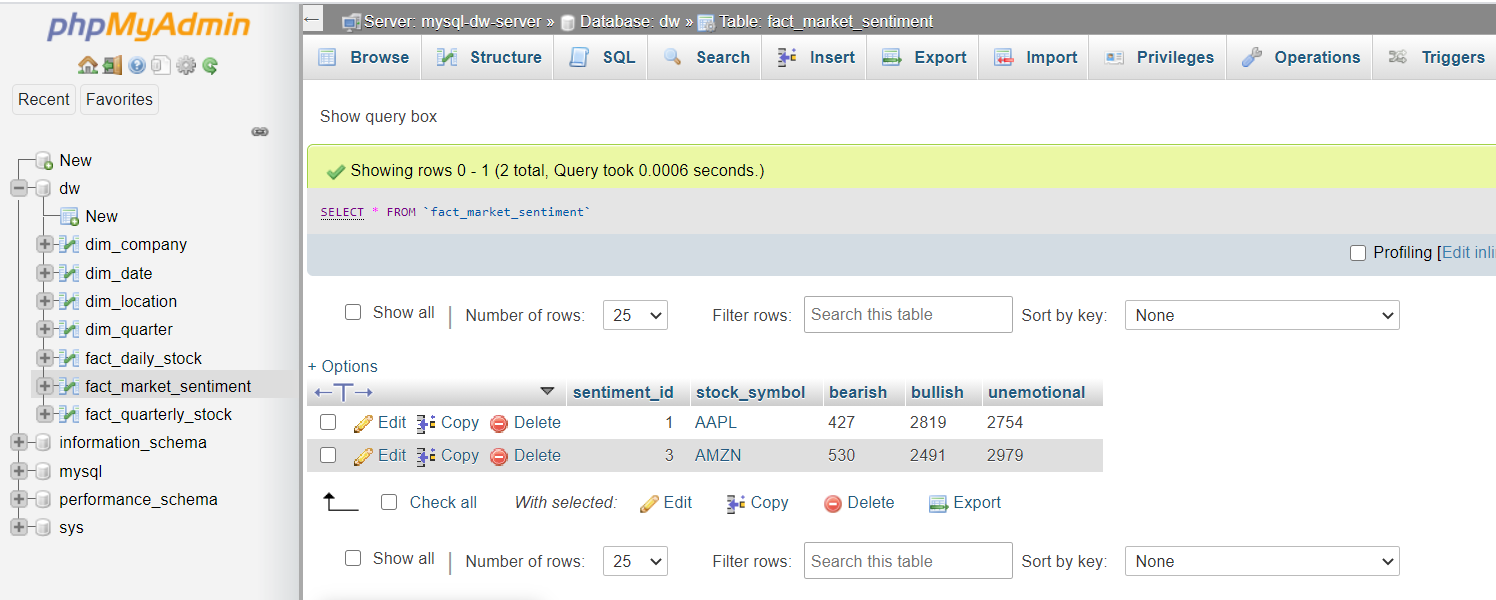
ORDER BY revenue DESC;

-- Latest Sentiments about a Company

SELECT \* FROM `fact\_market\_sentiment`;







**Conclusion and future scope**

With the help of this project, A working data warehouse was developed to answer business queries and interesting trends for the growth of the market. We do not touch upon the sentiment analysis much as it needed bit of knowledge of NLP. But there is a scope which can answer “Whether there is any significant relationship between social media sentiments, stock volumes and closing prices for the duration of one year?”