

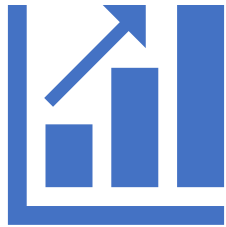
# Cloud: MAANG Stock Price Analysis

Group-8:

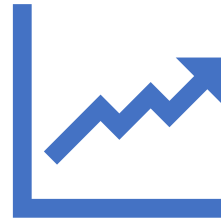
Sai Varun Kumar Namburi

Pooja Ramesh

# Problem Definition and Objective



In a rapidly evolving financial market, staying ahead of the trend analysis is essential for making informed investment decisions.



We also aim to explore the external factors that impact the stock price and the level of risk associated with their securities.



This project seeks to develop a data-driven solution to address the complexities of analyzing the historical stock prices of MAANG companies and identify the market conditions and external factors that could help make informed investment decisions.

# End-Goal



The ultimate end goal of this project is to ensure we get valuable insights to empower and help investors, analysts and traders make informed decisions



The project aims to predict the effectiveness of trading strategies by back tracing with historical data insights and provide actionable insights to make data-driven decisions about the trends, opportunities, and volatility

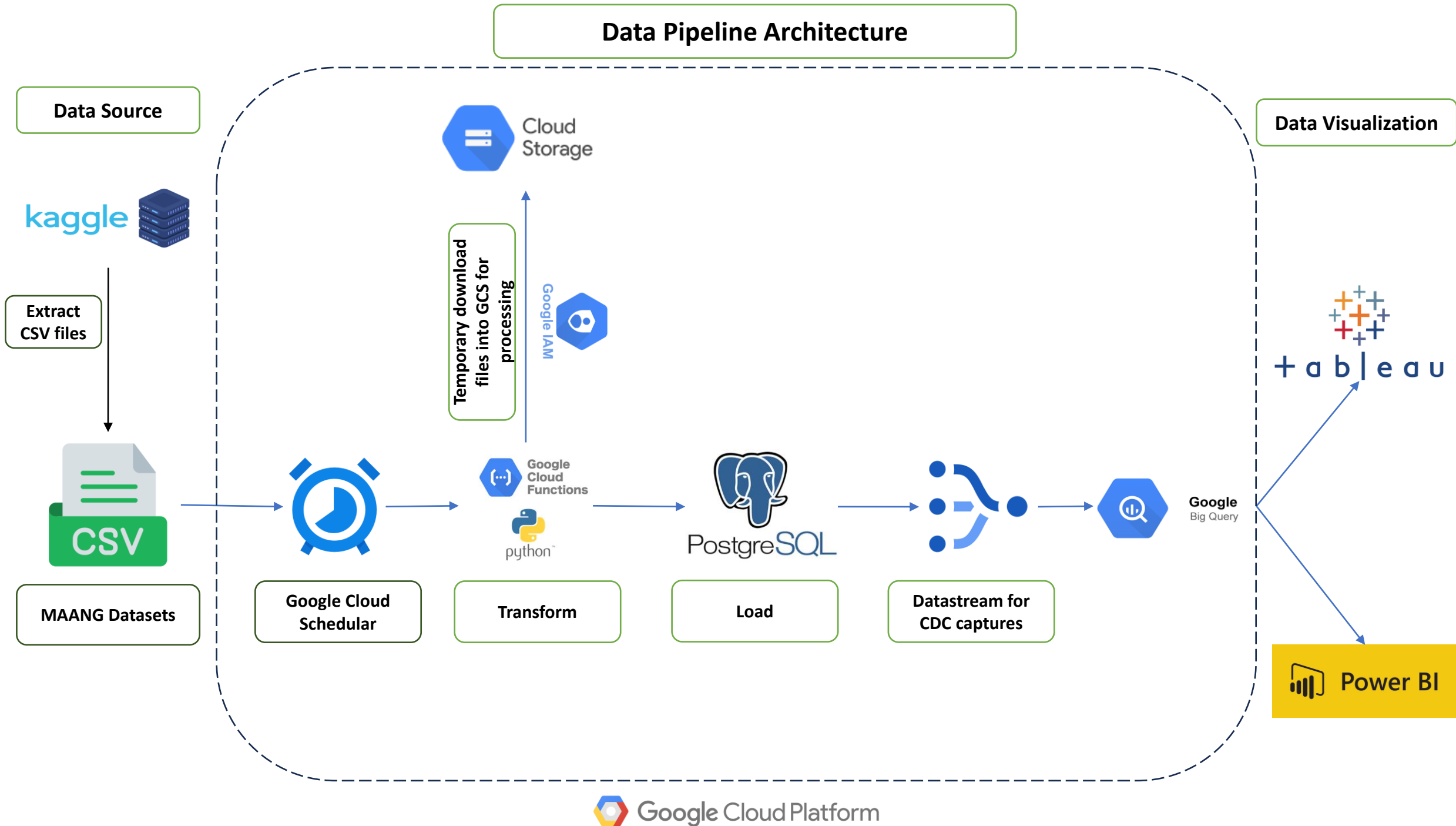


Furthermore, by developing an efficient cloud-based data engineering solution that can handle large volumes of data for real-time analysis.

# Data Source

- This dataset includes the historical data of stock prices for MAANG companies. It contains the daily, weekly, and monthly stock prices for each company, and they are automatically updated daily
- This dataset has 15 datasets, and each file has around 7 columns. Here we have daily, monthly, and weekly data for each of the companies from the years 2000 -2023.
- <http://www.kaggle.com/datasets/nikhil1e9/ne8lix-stock-price/>

# Data Pipeline Architecture



The screenshot shows the Google Cloud Scheduler interface for a project named 'My First Project'. It displays a list of three scheduled jobs under the 'SCHEDULER JOBS' tab. Each job is a cron job that triggers a Cloud Function.

Name	Status of last execution	Region	State	Description	Frequency	Target	Last run	Next run	Actions
<a href="#">dailyscheduler</a>	Has not run yet.	us-central1	Enabled	Daily Scheduler to run cloud functions	0 23 * * * (America/New_York)	URL : https://us-east4-alien-grove-405422.cloudfunctions.net/dailystocks_func		30 Nov 2023, 23:00:00	⋮
<a href="#">monthllyscheduler</a>	Has not run yet.	us-east4	Enabled	Montly Scheduler for stocks	0 23 1 * * (America/New_York)	URL : https://us-central1-alien-grove-405422.cloudfunctions.net/monthlystocks_func		1 Dec 2023, 23:00:00	⋮
<a href="#">weekllyscheduler</a>	Has not run yet.	us-east1	Enabled	Weekly Scheduler for stocks data	0 23 * * 6 (America/New_York)	URL : https://us-east1-alien-grove-405422.cloudfunctions.net/weeklystocks_func		2 Dec 2023, 23:00:00	⋮

- We have adopted py scripts for daily, weekly, and monthly to transform and insert the 15 files into cloud Postgres. Here, we used the Kaggle API instead of downloading the 15 files into the local or cloud bucket, as the data is very dynamic and changes daily.
- We are using a cloud scheduler to trigger the cloud functions.

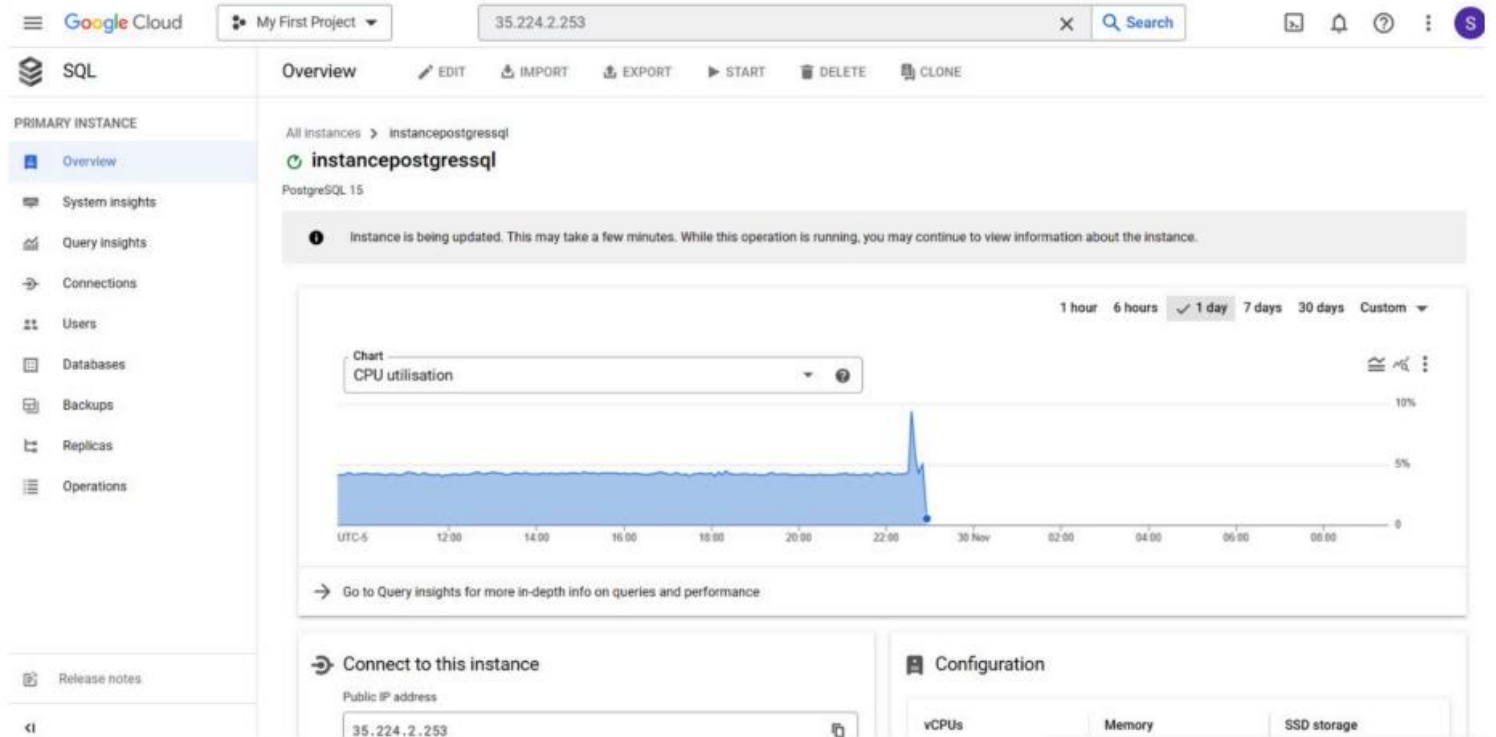
## Data Ingestion – Cloud Scheduler

# Cloud Functions

The screenshot displays the Google Cloud Functions console interface. At the top, the Google Cloud logo and 'My First Project' are visible. A search bar is present with the text 'Search (/) for resources, docs, products and more'. Below the navigation bar, the 'Cloud Functions' section is active, showing 'Function details' for 'dailystocks\_func'. The function is noted as '2nd gen' and 'Deployed at 29 Nov 2023, 22:21:11'. A URL is provided: [https://us-east4-alien-grove-405422.cloudfunctions.net/dailystocks\\_func](https://us-east4-alien-grove-405422.cloudfunctions.net/dailystocks_func). A 'Powered by Cloud Run' badge is also present. The 'SOURCE' tab is selected, showing a file tree with 'main.py' and 'requirements.txt'. The 'main.py' file is open, displaying Python code. The code includes imports for 'KaggleApi', 'service\_account', 'json', and 'os'. It defines a 'daily\_stocks' function that authenticates with the Kaggle API and downloads a dataset. The code is as follows:

```
21 register_adapter(np.int64, AsIs)
22
23 api = KaggleApi()
24 api.authenticate()
25
26 # credentials = service_account.Credentials.from_service_account_file('alien-grove-405422-bb57fda72219.json')
27 gcp_key_decoded = base64.b64decode(os.environ.get('gcs_base64'))
28 credentials_json = json.loads(gcp_key_decoded.decode('utf-8'))
29 database_pass = base64.b64decode(os.environ.get('DB_PASSWORD')).decode('utf-8')
30 database_name = base64.b64decode(os.environ.get('DB_NAME')).decode('utf-8')
31
32 gcp_credentials = service_account.Credentials.from_service_account_info(credentials_json)
33 @functions_framework.http
34 def daily_stocks(request):
35     api = KaggleApi()
36     api.authenticate()
37     download_dataset(None)
38     return "Kaggle API authenticated, Daily data is extracted, transformed and loaded into Postgres DB"
```

# Cloud SQL Postgres





File Edit Navigate Search SQL Editor Database Window Help

SQL Commit Rollback Auto 35.224.2.253 public@dwdi-cloud-db

Database Navigator x Projects <45.79.206.143> Script-9 <postgres> CreateStatements.sql <postgres> createstatements\_dw.sql <postgres> Script-14

Enter a part of object name here

35.224.2.253 - 35.224.2.253:5432

- Databases
  - dwdi-cloud-db
    - Schemas
      - public
        - Tables
          - daily 3.4M
          - monthly 200K
          - weekly 752K
        - Views
        - Materialized Views
        - Indexes
        - Functions
        - Sequences
        - Data types
        - Aggregate functions

```
select * from daily d ;
```

daily 1 x

select \* from daily d Enter a SQL expression to filter results (use Ctrl+Space)

	date	open	high	low	close	adj_close	volume	company
1	2023-11-27	189.92	190.67	188.9	189.79	189.79	40,500,500	APPLE
2	2023-11-27	336.18	339.9	334.2	334.7	334.7	15,646,300	META
3	2023-11-27	137.57	139.63	137.54	138.05	138.05	17,868,000	GOOGLE
4	2023-11-27	479.03	482	475.35	479.17	479.17	3,623,800	NETFLIX
5	2023-11-27	147.53	149.26	146.88	147.73	147.73	53,666,700	AMAZON
6	2023-11-24	139.54	139.677	137.47	138.22	138.22	8,828,600	GOOGLE
7	2023-11-24	340.13	341.86	336.77	338.23	338.23	5,467,500	META
8	2023-11-24	146.7	147.2	145.32	146.74	146.74	22,378,400	AMAZON
9	2023-11-24	190.87	190.9	189.25	189.97	189.97	24,048,300	APPLE

# Database Connection

Google Cloud My First Project Search (/) for resources, docs, products and more Search

Datastream

Streams

Connection profiles

Private connectivity

Stream details PAUSE RESUME DELETE TAGS EDIT VIEW LOGS

There are 1,512 events that were unsupported and were not loaded within last 7 days. VIEW IN LOGS EXPLORER

postgrestobq PostgreSQL/BigQuery

Stream ID	postgrestobq
Source profile	<a href="#">pgconnection</a>
Destination profile	<a href="#">dsbigqueryconnection</a>
Created	22 Nov 2023, 17:09:10
Updated	30 Nov 2023, 18:37:58

OVERVIEW MONITORING OBJECTS

Properties

Region	us-central1 (Iowa)
Labels	No labels set
Objects to include	1 schema
Objects to exclude	None
Backfill mode	Automatic
Destination data set	<a href="#">MAANGdata</a>
Staleness limit ?	5 minutes
Encryption	Google-managed
Tags	None

# Data Ingestion From Cloud SQL to Bigquery using DataStream

Google Cloud My First Project Search (/) for resources, docs, products and more

BigQuery Explorer

Analysis

- BigQuery Studio
- Data transfers
- Scheduled queries
- Analytics Hub
- Dataform
- Partner Centre

Migration

- Assessment
- SQL translation

Administration

Viewing resources: SHOW STARRED ONLY

- alien-grove-405422
  - Queries
  - Notebooks
  - External connections
  - MAANGdata
    - public\_daily
    - public\_monthly
    - public\_testmonthly
    - public\_weekly

public\_daily

SCHEMA DETAILS PREVIEW LINEAGE DATA PROFILE DATA QUALITY NEW

Row	date	open	high	low	close
1	2013-12-18	19.632143020629883	19.694643020629883	19.24285697937012	19.67035675048828
2	2014-07-31	28.950515747070312	29.102598190307617	28.42196655273437	28.501747131347656
3	2013-12-19	19.625	19.64285659790039	19.418928146362305	19.44499969482422
4	1997-05-15	0.1218750029802322	0.125	0.0963540002703666	0.0979169979691505
5	2014-08-01	28.441913604736328	28.719152450561523	28.06544685363769	28.22600555419922
6	1994-02-09	0.3191959857940674	0.3258930146694183	0.3147319853305816	0.3236609995365143
7	2017-03-29	42.95249938964844	43.821998596191406	42.95100021362305	43.71599960327149
8	2022-12-09	115.3000030517578	117.54000091552734	113.87000274658205	115.9000015258789
9	2013-12-20	19.479642868041992	19.70035743713379	19.457857131958008	19.60785675048828
10	1997-05-16	0.0984380021691322	0.0989580005407333	0.0854170024394989	0.0864579975605011
11	2014-08-04	28.374099731445312	28.68873405456543	28.127775192260746	28.579036712646484
12	1994-02-10	0.3236609995365143	0.3348209857940674	0.3214290142059326	0.3258930146694183
13	2017-03-30	43.747501373291016	43.85300064086914	43.58300018310547	43.81700134277344

# Bigquery Data Warehouse

1. Calculate the average closing price for last week for Apple:

The screenshot shows a data analytics interface with a left sidebar, a central query editor, and a bottom results section.

**Explorer (Left Sidebar):**

- Search bar: "Type to search"
- Section: "Viewing resources."
- Link: "SHOW STARRED ONLY"
- Tree structure:
  - alien-grove-405422
    - Queries
    - Notebooks
    - External connections
    - MAANGdata
      - public\_daily
      - public\_monthly
      - public\_testmonthly
      - public\_weekly

**Query Editor (Center):**

Untitled 2

```
1 SELECT company, AVG(CAST(close AS FLOAT64)) AS average_closing_price
2 FROM `alien-grove-405422.MAANGdata.public_weekly`
3 WHERE company = 'APPLE'
4 AND date >= DATE_SUB(CURRENT_DATE(), INTERVAL EXTRACT(DAYOFWEEK FROM CURRENT_DATE()) + 6 DAY)
5 AND date < DATE_SUB(CURRENT_DATE(), INTERVAL EXTRACT(DAYOFWEEK FROM CURRENT_DATE()) - 1 DAY)
6 group by 1;
```

**Query results (Bottom):**

Query results

SAVE RESULTS


Tabs: JOB INFORMATION, RESULTS, CHART, PREVIEW, JSON, EXECUTION DETAILS, EXECUTIO

Row	company	average_closing_pric
1	APPLE	189.9700012207...

# Analytical Queries

2. What is the opening stock price for Meta in the month of October?

```
16
17 SELECT company, open
18 FROM `alien-grove-405422.MAANGdata.public_monthly`
19 WHERE company = 'META'
20    AND EXTRACT(MONTH FROM date) = 10
21    AND EXTRACT(YEAR FROM date) = EXTRACT(YEAR FROM CURRENT_DATE());
22
```

Query results 

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION
Row	company	open				
1	META	302.739990234375				

Cont.

3. Which is the lowest trading price and company in the month of November?

```
14
15 WITH MonthlyLowestPrices AS (
16   SELECT company, MIN(low) AS lowest_trading_price
17   FROM `alien-grove-405422.MAANGdata.public_monthly`
18   WHERE EXTRACT(MONTH FROM date) = 11
19         AND EXTRACT(YEAR FROM date) = EXTRACT(YEAR FROM CURRENT_DATE())
20   GROUP BY company
21 )
22
23 SELECT company, lowest_trading_price
24 FROM MonthlyLowestPrices
25 ORDER BY lowest_trading_price ASC
26 LIMIT 1;
27
28
```

#### Query results

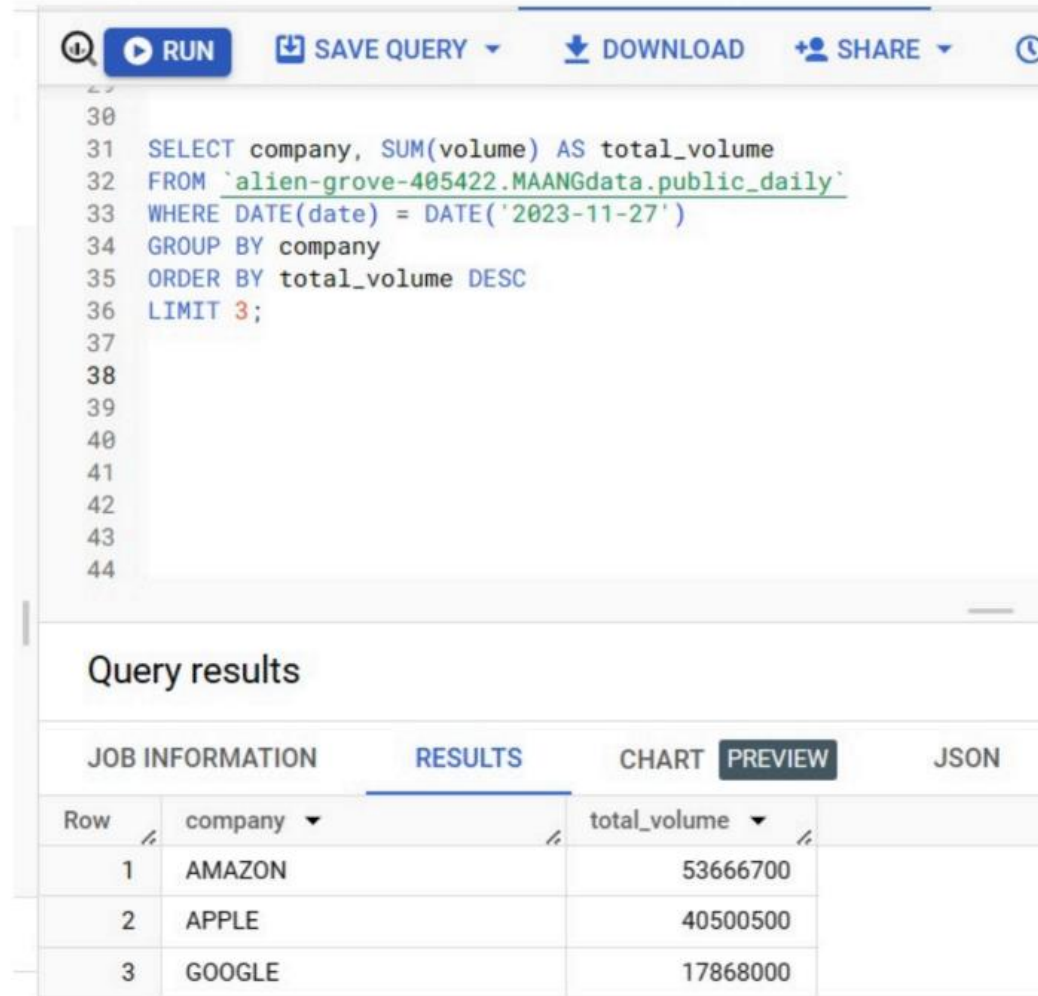
JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUT
Row	company	lowest_trading_price				
1	GOOGLE	124.9250030517578				

Cont.



Cont.

4. On the 27<sup>th</sup> of November, what were the top 3 stocks sold and by which company?



The screenshot shows a SQL query editor interface. At the top, there are buttons for 'RUN', 'SAVE QUERY', 'DOWNLOAD', 'SHARE', and a clock icon. Below the buttons is a text area containing a SQL query. The query is as follows:

```
30  
31 SELECT company, SUM(volume) AS total_volume  
32 FROM `alien-grove-405422.MAANGdata.public_daily`  
33 WHERE DATE(date) = DATE('2023-11-27')  
34 GROUP BY company  
35 ORDER BY total_volume DESC  
36 LIMIT 3;  
37  
38  
39  
40  
41  
42  
43  
44
```

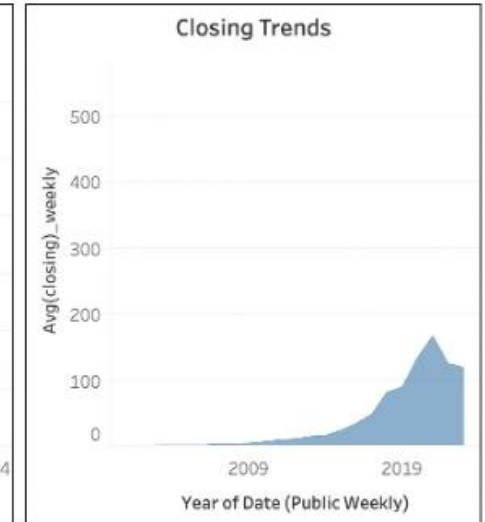
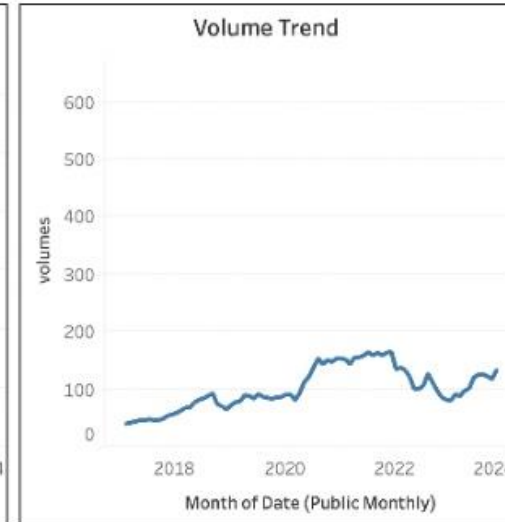
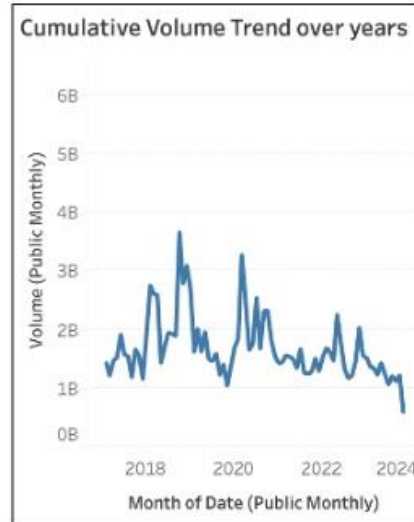
Below the query editor, there is a section titled 'Query results'. This section contains a table with the following data:

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON
Row	company	total_volume			
1	AMAZON	53666700			
2	APPLE	40500500			
3	GOOGLE	17868000			

# Tableau Dashboard

## STOCK ANALYSIS - AMAZON

Date (Public Monthly) 1/13/2017 11/1/2023  
Company (Public Monthly)



High and Low

Company	High (..	Low (Public ..
AMAZON	188.65400..	165.34899..
	188.10749..	164.17750..
	177.99400..	165.19500..
	177.49949..	163.69949..
	177.69999..	155.77749..
	177.61250..	143.55000..
	176.24299..	158.61000..
	174.81199..	150.94999..
	174.33250..	156.36849..
	174.75	153.64999..
	173.62899..	158.78799..
	173.94999..	158.8125
	171.39999..	135.35200..

