

Project Step by Step

- 1) I imported all the data into one collection called StockMarket. Then export it as a json file.

The screenshot shows the Studio 3T interface for MongoDB. The left sidebar displays the database structure: 'New localhost:27017 [direct]' > 'admin' > 'Collections (1)' > 'StockMarket'. The main window shows the 'StockMarket' collection with a table view of 50 documents. The table has columns: Id, Symbol, Date, Open, High, Low, and Close. The data represents daily stock prices for AAPL from 1992-10-23 to 1992-12-07.

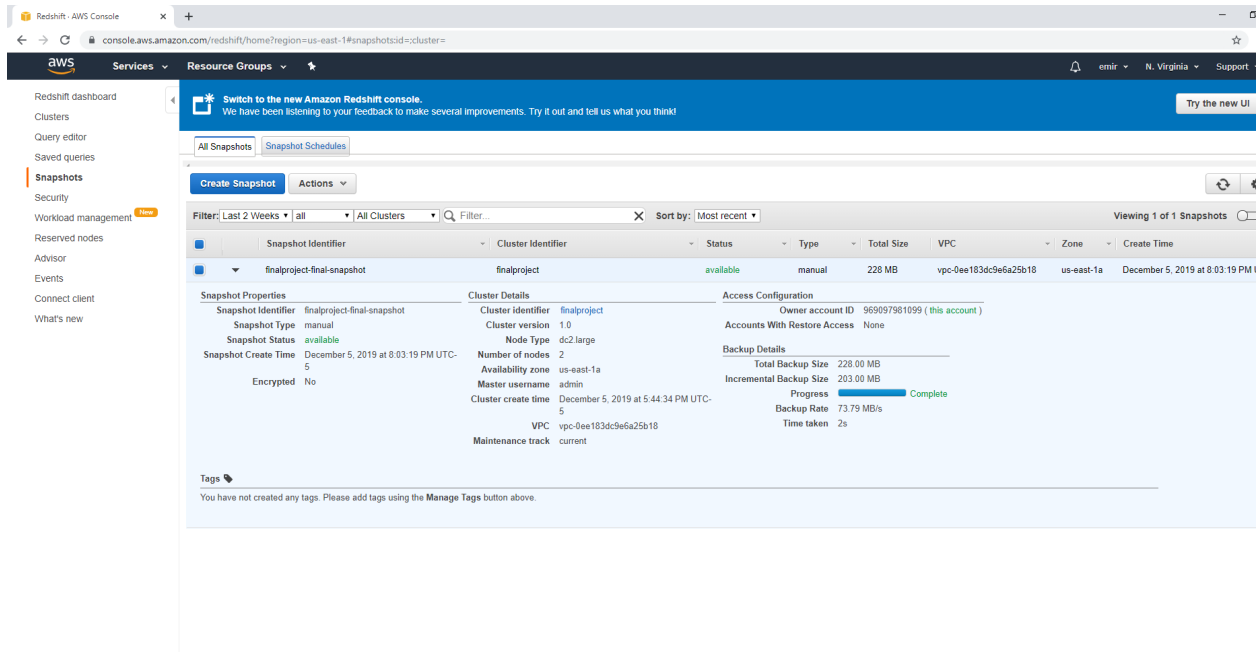
Id	Symbol	Date	Open	High	Low	Close
5df7ba08c918571	AAPL	1992-10-23T00:00	1.758929	1.767857	1.723214	1.741
5df7ba08c918571	AAPL	1992-10-26T00:00	1.741071	1.839286	1.732143	1.839
5df7ba08c918571	AAPL	1992-10-27T00:00	1.839286	1.875	1.821429	1.839
5df7ba08c918571	AAPL	1992-10-28T00:00	1.830357	1.883929	1.8125	1.866
5df7ba08c918571	AAPL	1992-10-29T00:00	1.866071	1.928571	1.839286	1.901
5df7ba08c918571	AAPL	1992-10-30T00:00	1.910714	1.910714	1.857143	1.875
5df7ba08c918571	AAPL	1992-11-02T00:00	1.875	1.883929	1.840214	1.866
5df7ba08c918571	AAPL	1992-11-03T00:00	1.875	1.875	1.839286	1.857
5df7ba08c918571	AAPL	1992-11-04T00:00	1.857143	1.883929	1.857143	1.875
5df7ba08c918571	AAPL	1992-11-05T00:00	1.875	1.964286	1.875	1.964
5df7ba08c918571	AAPL	1992-11-06T00:00	1.955357	2.017857	1.955357	1.991
5df7ba08c918571	AAPL	1992-11-09T00:00	2.0	2.0	1.955357	1.973
5df7ba08c918571	AAPL	1992-11-10T00:00	1.964286	2.017857	1.955357	2.008
5df7ba08c918571	AAPL	1992-11-11T00:00	2.017857	2.080357	2.008929	2.026
5df7ba08c918571	AAPL	1992-11-12T00:00	2.035714	2.053571	2.013993	2.031
5df7ba08c918571	AAPL	1992-11-13T00:00	2.035714	2.044643	2.0	2.008
5df7ba08c918571	AAPL	1992-11-16T00:00	2.008929	2.0625	2.0	2.049
5df7ba08c918571	AAPL	1992-11-17T00:00	2.044643	2.053571	1.959821	1.973
5df7ba08c918571	AAPL	1992-11-18T00:00	2.0	2.080357	1.982143	2.062
5df7ba08c918571	AAPL	1992-11-19T00:00	2.0625	2.125	2.0625	2.080
5df7ba08c918571	AAPL	1992-11-20T00:00	2.089286	2.098214	2.035714	2.053
5df7ba08c918571	AAPL	1992-11-23T00:00	2.017857	2.035714	2.008929	2.026
5df7ba08c918571	AAPL	1992-11-24T00:00	2.035714	2.053571	2.017857	2.053
5df7ba08c918571	AAPL	1992-11-25T00:00	2.035714	2.044643	2.0	2.017
5df7ba08c918571	AAPL	1992-11-27T00:00	2.017857	2.044643	2.008929	2.017
5df7ba08c918571	AAPL	1992-11-30T00:00	2.008929	2.053571	1.986607	2.053
5df7ba08c918571	AAPL	1992-12-01T00:00	2.044643	2.107143	2.026786	2.080
5df7ba08c918571	AAPL	1992-12-02T00:00	2.080357	2.089286	2.035714	2.044
5df7ba08c918571	AAPL	1992-12-03T00:00	2.017857	2.058036	2.004464	2.053
5df7ba08c918571	AAPL	1992-12-04T00:00	2.044643	2.053571	2.017857	2.031
5df7ba08c918571	AAPL	1992-12-07T00:00	2.026786	2.0625	2.026786	2.062

- 2) I uploaded my json file and json path to my S3 bucket.

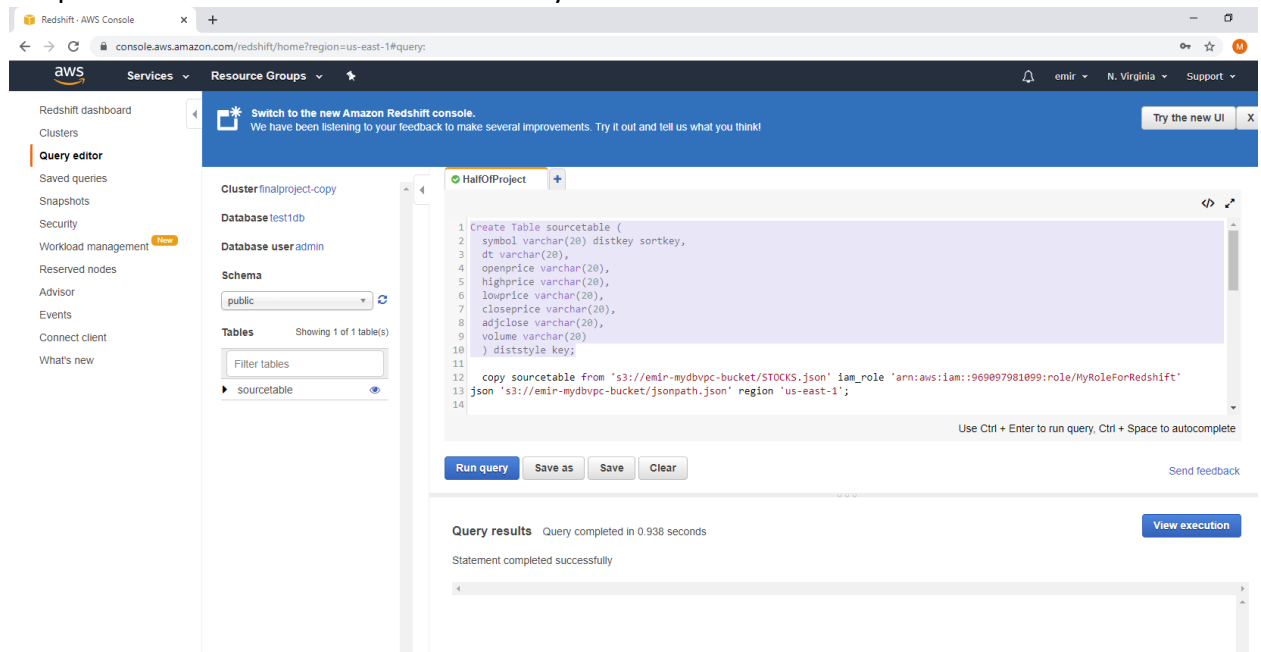
The screenshot shows the AWS S3 Management Console for the bucket 'emir-mydbvpc-bucket'. The 'Overview' tab is selected, showing a list of objects. The objects are sorted by 'Last modified' date. The list includes files like 'HW5', 'AdventureWorks2012.bak', 'StockMarket.json', 'WideWorldImporters-Standard.bak', 'dms_sample.bak', and 'jsonpath.json'.

Name	Last modified	Size	Storage class
HW5	--	--	--
AdventureWorks2012.bak	Nov 7, 2019 11:57:53 AM GMT-0500	44.9 MB	Standard
StockMarket.json	Dec 16, 2019 1:07:57 PM GMT-0500	29.8 MB	Standard
WideWorldImporters-Standard.bak	Sep 19, 2019 8:21:35 PM GMT-0400	121.1 MB	Standard
dms_sample.bak	Sep 23, 2019 5:34:23 PM GMT-0400	9.9 GB	Standard
jsonpath.json	Dec 5, 2019 7:22:12 PM GMT-0500	147.0 B	Standard

3) I created a new cluster.

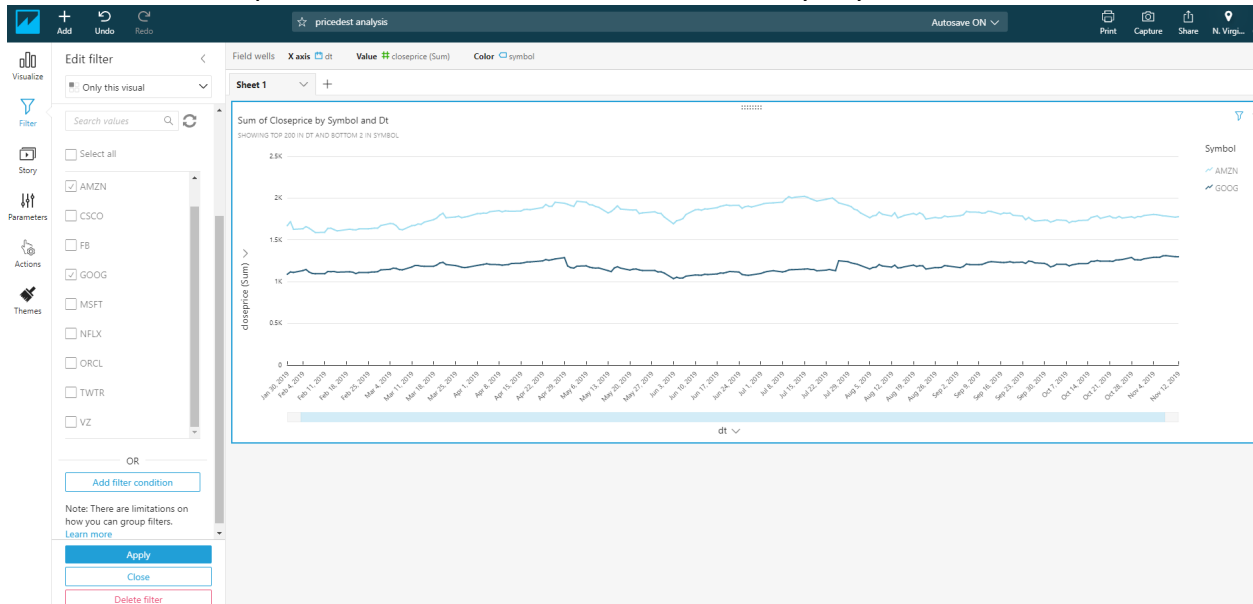


4) I run the script to create a new table and import my data from my S3 bucket to my Redshift cluster. I used the script that you gave us. I have created my source table, pricestest table and delete unnecessary records.

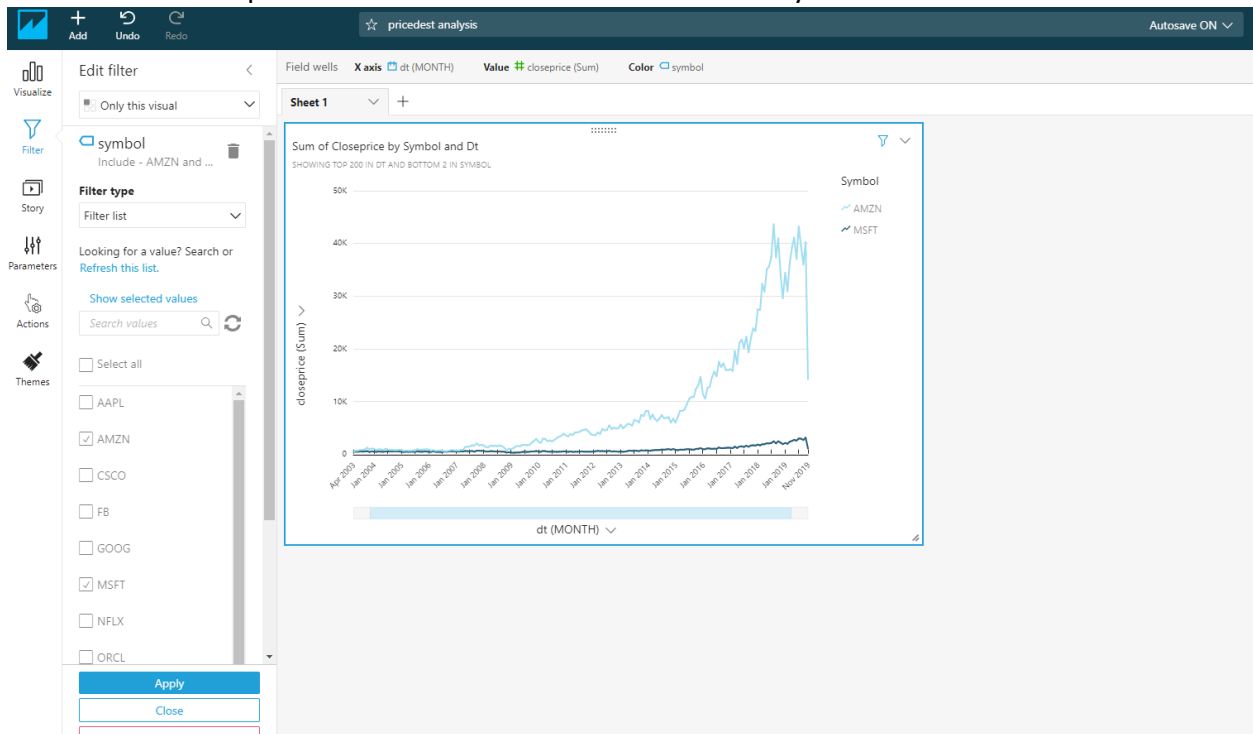


- 5) I connected my redshift to Quicksight I imported pricest table to be able visualize price comparison charts. These charts show comparisons between Microsoft and amazon.

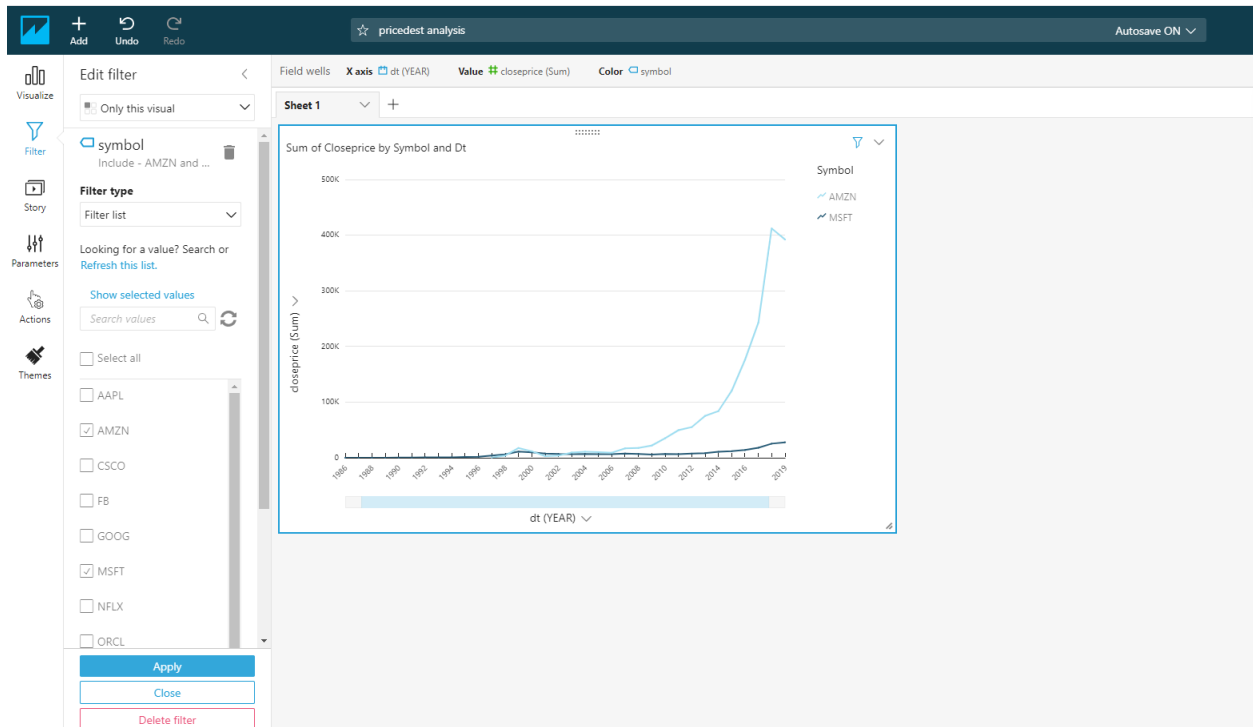
* Price comparison between Amazon and Microsoft by day.



*Price comparison between Amazon and Microsoft by month.

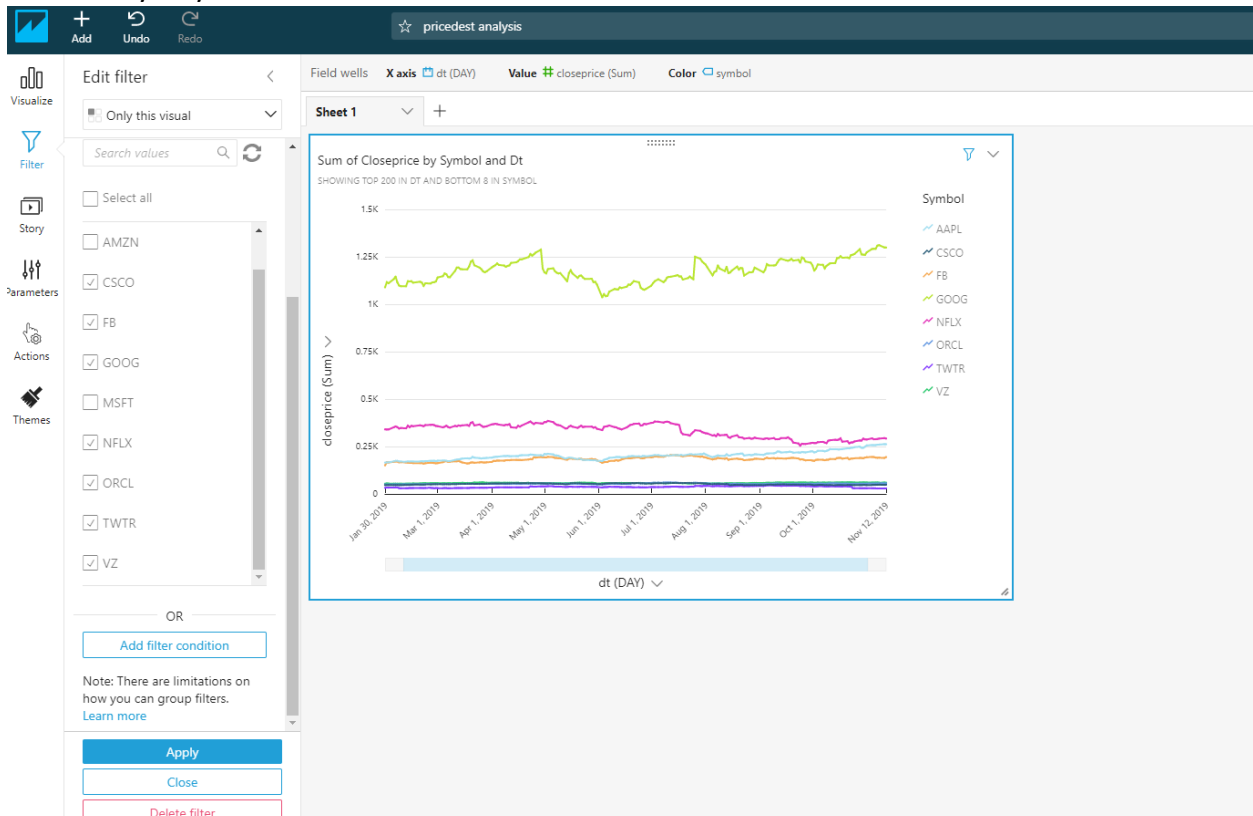


*Price comparison between Amazon and Microsoft by year.

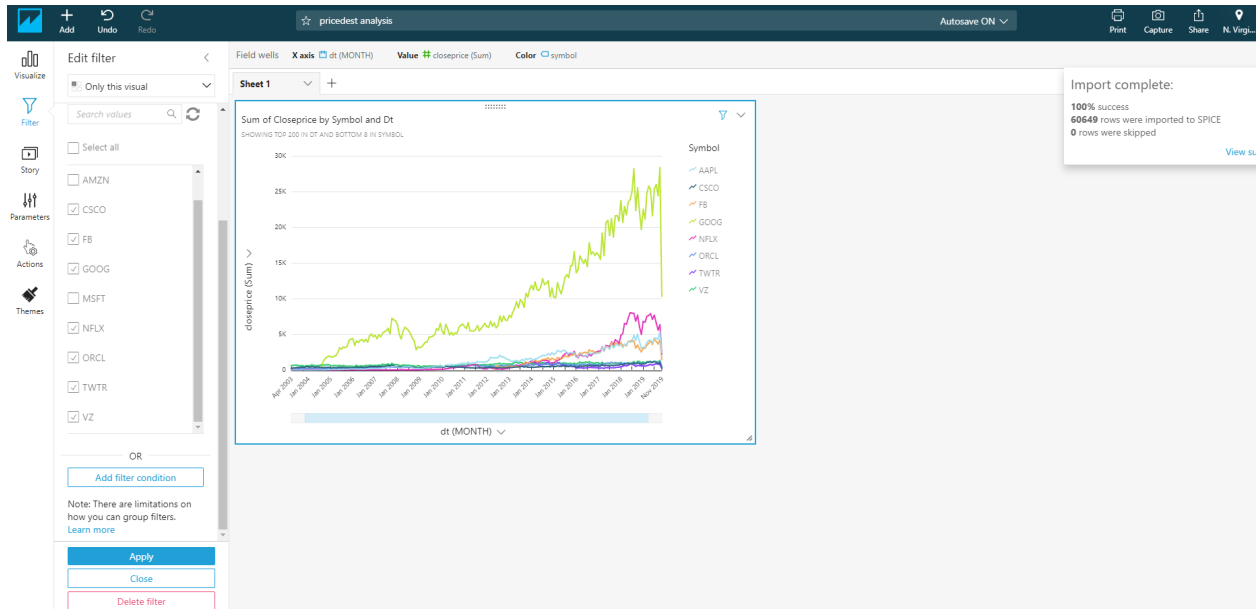


6) Price comparisons between other symbols that you asked in second question.

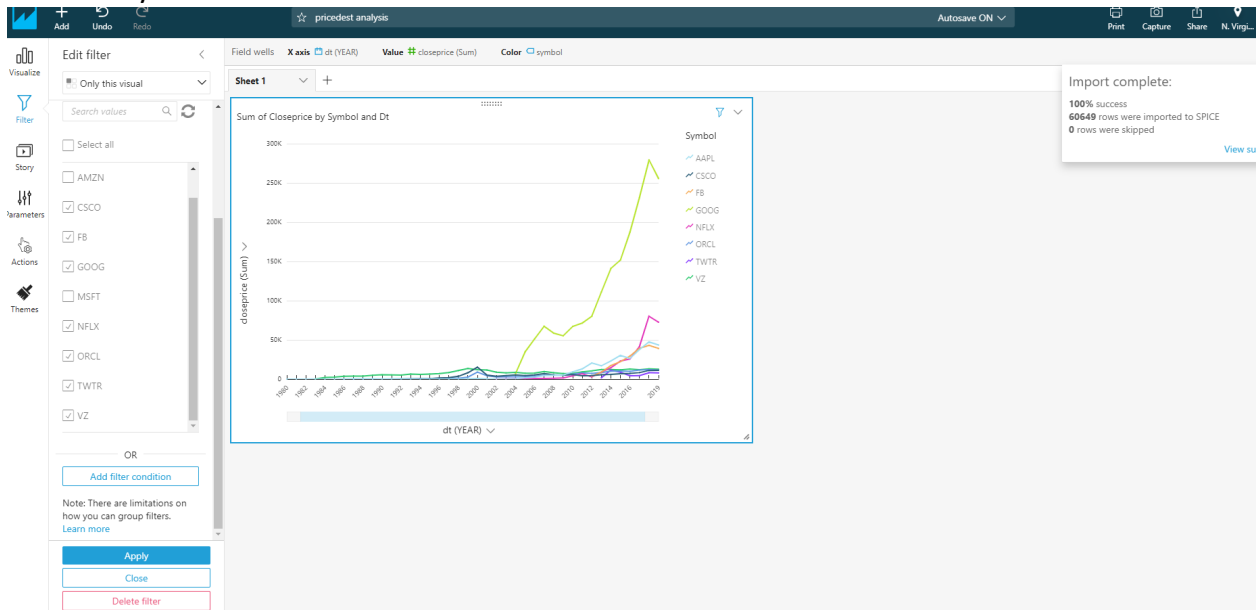
*By Day



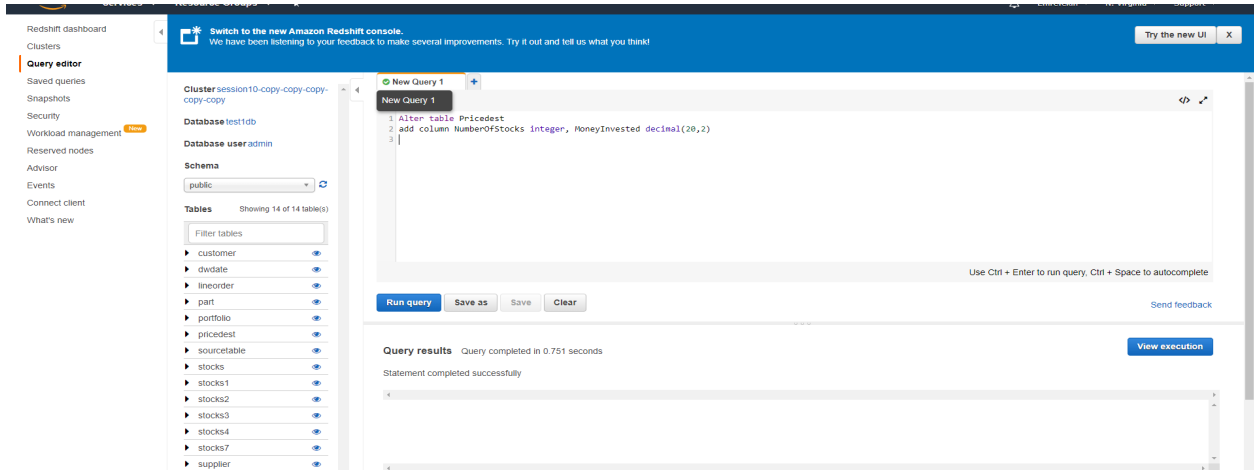
*By Months



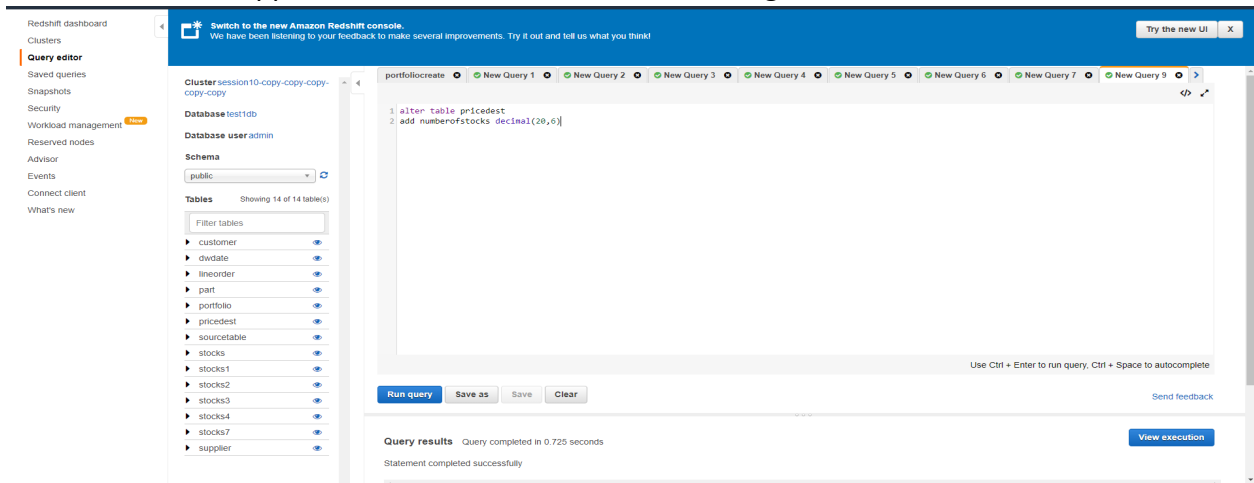
*By Years



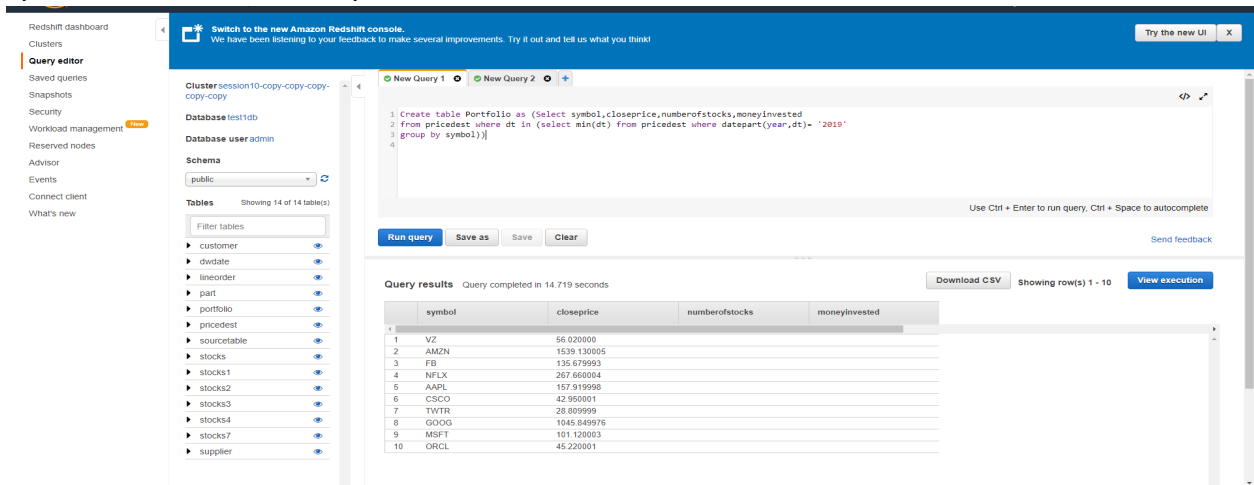
7) This is how we calculated profit portfolio in year 2019.
* I have added numberofstocks and moneyinvested columns.



* I have dropped numberofstocks field and add it again as a decimal.

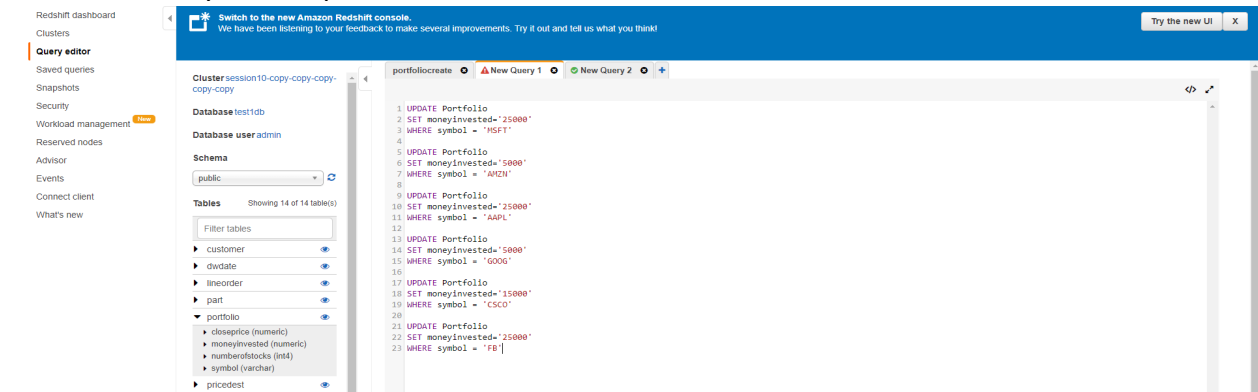


* I have created new portfolio table which is going to give me the close prices for each symbol and for the first day of 2019.

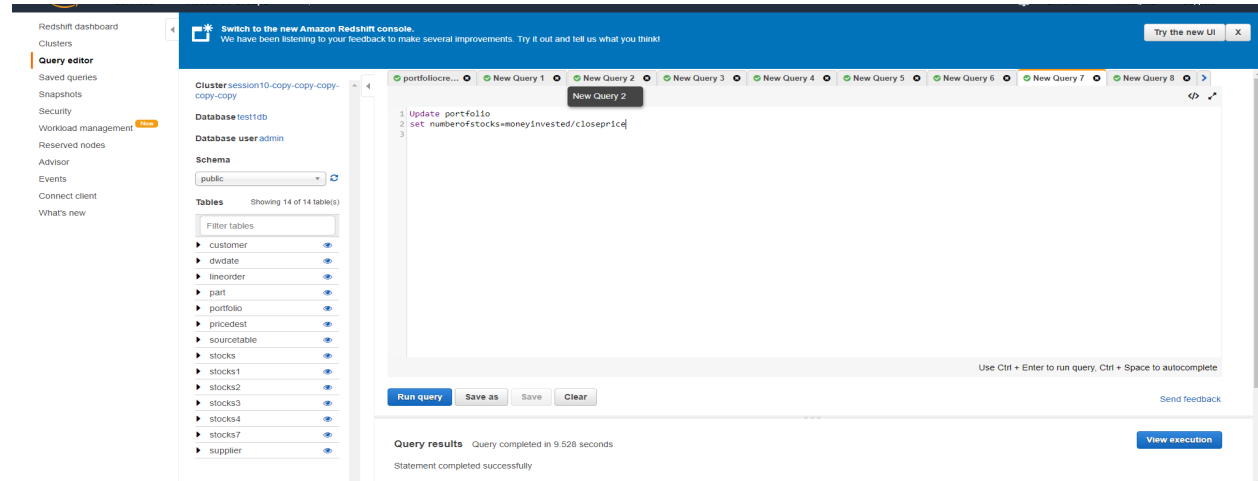


	symbol	closeprice	numberofstocks	moneyinvested
1	VZ	56.020000		
2	AMZN	1539.130005		
3	FB	135.079993		
4	NFLX	287.660004		
5	AAPL	157.119998		
6	CSCO	42.950001		
7	TWTR	28.589999		
8	GOOG	1045.849976		
9	MSFT	101.120003		
10	ORCL	45.220001		

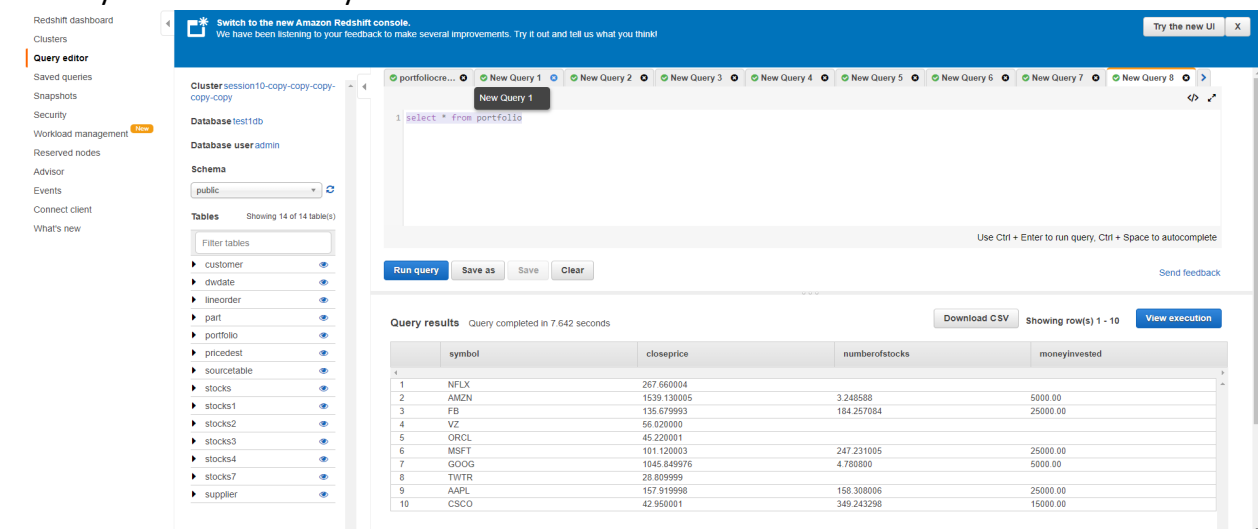
* I have calculated percentages of 100000 dollars and I have updated moneyinvested field for each symbol in portfolio table.



* I have updated numberofstocks field as calculated field (moneyinvested/closeprice=numberofstocks)



* In the end my table gave me the number of stocks which is bought by investor at January 1 2019 for each symbol.



* I have created new table named "NumberofstocksForEachSymbol" and I had only two fields in this table which gives me the number of stocks which is bought by investor at January 1 2019 for each symbol.

The screenshot displays the Amazon Redshift Query Editor interface. On the left sidebar, the 'Query editor' section is active, showing a list of tables in the 'public' schema, including 'customer', 'dwdate', 'lineorder', 'numberofstocksforeac...', 'part', 'portfolio', 'pricedest', 'sourcetable', 'stocks', 'stocks1', 'stocks2', 'stocks3', 'stocks4', 'stocks7', and 'supplier'. The main query editor area contains the following SQL code:

```
1 create table NumberofstocksForEachSymbol as (select symbol,numberofstocks
2 from portfolio where numberofstocks is not null)
```

Below the query editor, the 'Run query' button is visible. The 'Query results' section shows the query completed in 14.828 seconds with the message 'Statement completed successfully'.

*Finally I have created my last table which is "PortfolioForQuickSight" by joining "pricedest" and "NumberofstocksForEachSymbol" tables. I have created new calculated field in this table by multiplying numberofstocks and closeprice fields to get updated invested money information for each day.

The screenshot displays the Amazon Redshift Query Editor interface. On the left sidebar, the 'Query editor' section is active, showing a list of tables in the 'public' schema, including 'customer', 'dwdate', 'lineorder', 'numberofstocksforeac...', 'part', 'portfolio', 'portfolioforquicksight', 'pricedest', 'sourcetable', 'stocks', 'stocks1', 'stocks2', 'stocks3', 'stocks4', 'stocks7', and 'supplier'. The main query editor area contains the following SQL code:

```
1
2
3 create table PortfolioForQuickSight as(select pricedest.symbol,dt,closeprice,(numberofstocksforeachsymbol.numberofstocks*closeprice) as investedmoneyprogress
4 from pricedest join numberofstocksforeachsymbol on pricedest.symbol=numberofstocksforeachsymbol.symbol where datepart(year,dt)='2019'
5 order by pricedest.symbol, dt)
6
```

Below the query editor, the 'Run query' button is visible. The 'Query results' section shows the query completed in 10.321 seconds with the message 'Statement completed successfully'.

* The table has records for close prices of each symbol for each day of 2019 and invested money field gets updates every day.

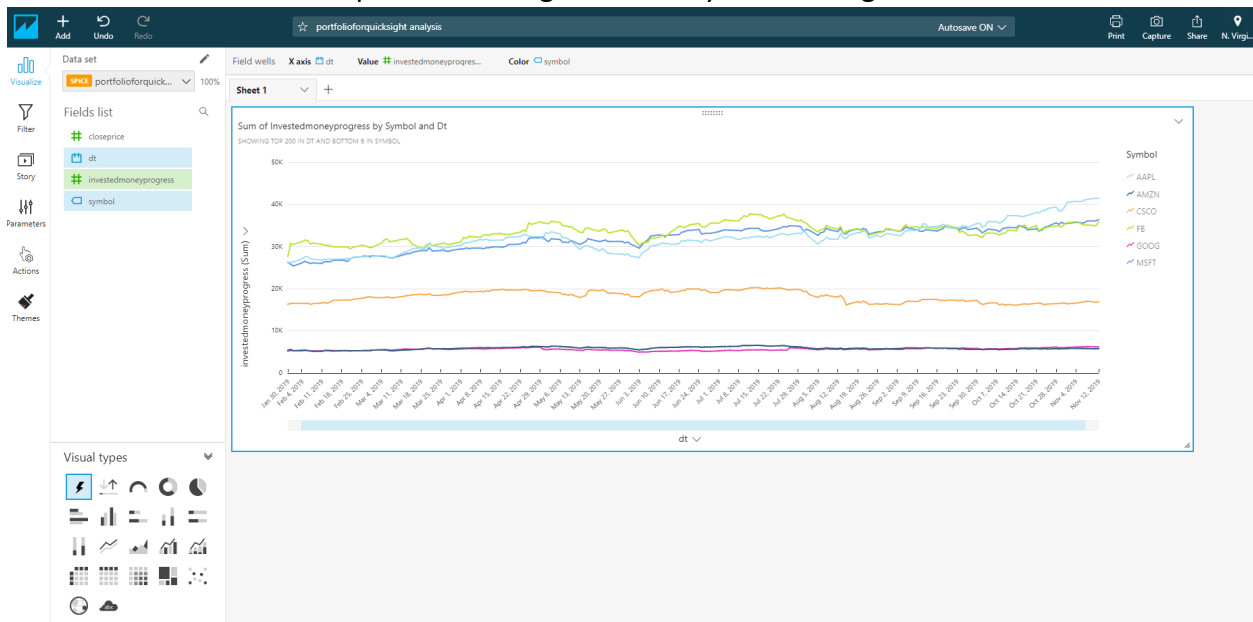
The screenshot displays the Amazon Redshift console interface. On the left, a sidebar contains navigation links: Redshift dashboard, Clusters, Query editor (highlighted), Saved queries, Snapshots, Security, Workload management, Reserved nodes, Advisor, Events, Connect client, and What's new. The main area is titled 'Switch to the new Amazon Redshift console. We have been listening to your feedback to make several improvements. Try it out and tell us what you think!' with a 'Try the new UI' button. Below this, the 'Cluster' section shows 'session10-copy-copy-copy-copy-copy'. The 'Database' is 'test1db' and the 'Database user' is 'admin'. The 'Schema' is set to 'information_schema'. A list of tables is shown, including 'applicable_roles', 'check_constraints', 'column_domain_usage', 'column_privileges', 'column_udt_usage', 'columns', 'constraint_column_us...', 'constraint_table_usage', 'data_type_privileges', 'domain_constraints', 'domain_udt_usage', 'domains', 'element_types', 'enabled_roles', 'information_schema_...', and 'key_column_usage'. The 'Query editor' shows a SQL query: 'select * from public.portfolioforquicksight order by symbol,dt'. Below the query editor, the 'Query results' section shows the query completed in 0.381 seconds. A table of results is displayed, showing columns for symbol, date, and price. The table contains 17 rows of data for AAPL stock prices from 2019-01-10 to 2019-01-24.

Symbol	Date	Price
AAPL	2019-01-10	153.800003
AAPL	2019-01-11	152.289993
AAPL	2019-01-14	150.000000
AAPL	2019-01-15	153.070007
AAPL	2019-01-16	154.940002
AAPL	2019-01-17	155.860001
AAPL	2019-01-18	156.820007
AAPL	2019-01-22	153.300003
AAPL	2019-01-23	153.919998
AAPL	2019-01-24	152.699997

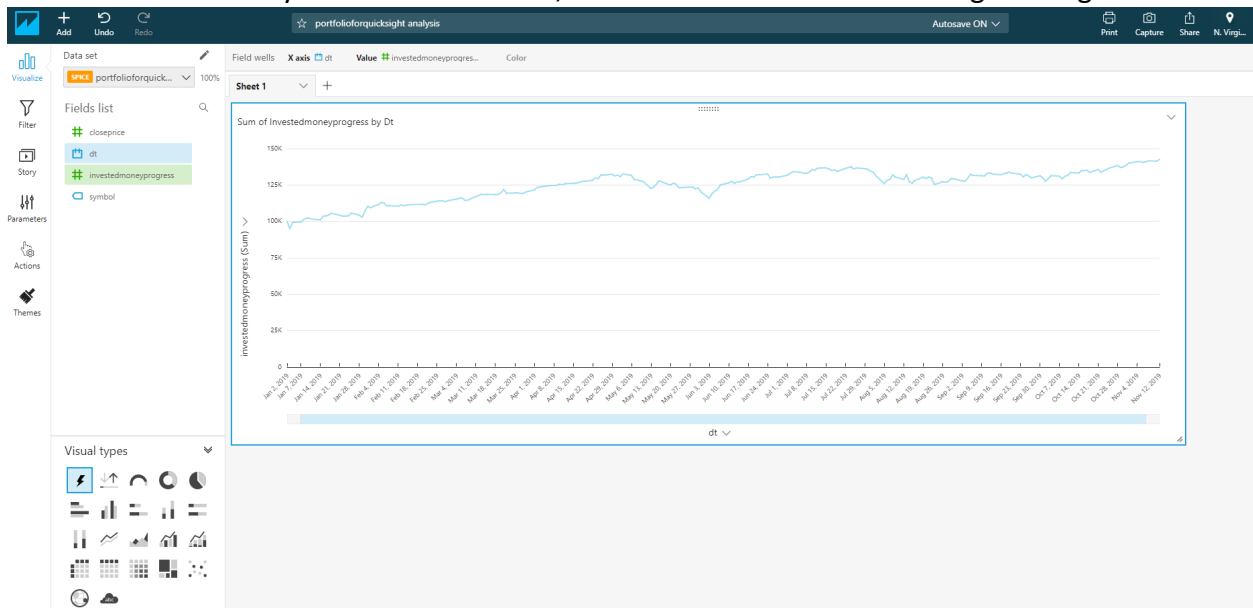
* I have connected my redshift cluster to Quicksight

The screenshot displays the Amazon Quicksight console interface. The 'Data Sets' tab is selected. A 'Create a Data Set' dialog box is open, showing various data sources. The 'New Redshift data source' dialog is the active one, with fields for 'Data source name' (Final_Project1), 'Instance ID' (session10-copy-copy-copy-copy-copy), 'Connection type' (Public network), 'Database name' (test1db), 'Username' (admin), and 'Password' (masked). The 'Validate connection' button is disabled, and the 'SSL is enabled' checkbox is checked. The 'Create data source' button is visible. The background shows a grid of data source tiles for various services like Salesforce, RDS, SQL Server, Aurora, Snowflake, AWS IoT Analytics, GitHub, Athena, PostgreSQL, Spark, Twitter, Jira, ServiceNow, Adobe Analytics, and others.

* That chart shows portfolio change for each symbol during 2019



* When we remove symbol from color tab, it shows total investment change during 2019.



*Beginning and end of table chart for same information.

