**Data visualisation Final Project**

****

**Stock market analysis**

**Supervisor**

**Toaa Gamal**

**Team4**

**Mahmoued Mohamed Adel**

**Khaled Magdy**

**Hassan Essam**

**Shimaa Salah**

**Marwa Sabry**

**Final Project Document**

**Our project about stock market analysis there are 3 data set ,About the American Stock Exchange,**

## S&P 500 component stocks

**First data include 2 table**

**First table** about US stock index s&p, it’s come on short Standard & Poor's, is a company well known around the world as a creator of financial market indices—widely used as investment benchmarks—a data source, and an issuer of [credit ratings](https://www.investopedia.com/terms/c/creditrating.asp) for companies and debt obligations. It's perhaps best-known for the popular and often-cited [S&P 500 Index](https://www.investopedia.com/terms/s/sp500.asp). The company's roots date back to the 1860s. Since 2016, its official corporate name has been S&P Global,Standard & Poor's grew out of two companies: Poor's Publishing, a publisher of railroad industry guidebooks officially founded in 1868, and the Standard Statistics Bureau (later Company), founded in 1906, which published financial data on companies. In 1923, it released its first stock market indicator, which contained 233 companies. Poor's Publishing, meanwhile, issued its first rating in 1916.

The two firms merged in 1941, to create Standard & Poor's

s&p data Consists of two table first table about It comprises 503 common stocks which are issued by 500 large-cap companies traded on American stock exchanges (including the 30 companies that compose the Dow Jones Industrial Average),the data include The date on which the company was added to the S&P index from 1976 to 2021.

The index includes about 80 percent of the American equity market by capitalization.

**Second table** about The companies that seized the shares of other companies and the reason for that.Between January 1, 1963 and December 31, 2014, 1,186 index components were replaced by other components.

## **All\_stocks\_5yr.csv Data**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Second data includes one table ,it’s about a dataset with historical stock prices (last 5 years) for all companies currently found on the S&P 500 index.

Content

The data is presented in a couple of formats to suit different individual's needs or computational limitations. I have included files containing 5 years of stock data (in the allstocks5yr.csv and corresponding folder).

The folder individual stocks 5yr contains files of data for individual stocks, labelled by their stock ticker name. The allstocks5yr.csv contains the same data, presented in a merged .csv file. Depending on the intended use (graphing, modelling etc.) the user may prefer one of these given formats.

## Constituents-financials data

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Third data about List of companies in the S&P 500 (Standard and Poor’s 500). The S&P 500 is a free-float, capitalization-weighted index of the top 500 publicly listed stocks in the US (top 500 by market cap). The dataset includes a list of all the stocks contained therein and associated key financials such as price, market capitalization, earnings, price/earnings ratio, price to book etc.

About data: the data include 3 files , size of data 186kb, in format csv, it’s created science 4 years ago

Data source:Information on S&P 500 index used to be available on the [official webpage on the Standard and Poor’s website](http://www.spindices.com/indices/equity/sp-500) but until they publish it back, Wikipedia is the best up-to-date and open data s**ource.**

**This data collected from**

**•https://en.wikipedia.org/wiki/List\_of\_S%26P\_500\_compans**

**• https://www.kaggle.com/camnugent/sandp500**

**•https://datahub.io/core/s-and-p-500-companies-financials#resource-constituents financials**

**We use two extra datasets from Kaggle :**

**1-Gold price dataset :**

[**https://www.kaggle.com/datasets/tunguz/gold-prices?select=monthly\_csv.csv**](https://www.kaggle.com/datasets/tunguz/gold-prices?select=monthly_csv.csv)

Historical Annual and Monthly Gold Prices. Retrieved from DataHub, and expected to be updated annually.

About data: the data include 2 files , size of data 15.45 kB, in format csv, data date from 1950 to 2019

**2-Crude oil price dataset :**

[**https://www.kaggle.com/datasets/sc231997/crude-oil-price**](https://www.kaggle.com/datasets/sc231997/crude-oil-price)**Oil price is a very important part of any county's economy. The Crude Oil WTI (USD/Bbl) dataset was created with the expectation to understand the impact of global oil prices on any country's economy.**

**Data date from 1989 to 2022**

**Using** [**https://tradingeconomics.com/commodity/crude-oil**](https://tradingeconomics.com/commodity/crude-oil) **this data set is created and will be kept updated monthly.**

**3-Inflation , Interest , Unemployment rate:**

[**https://www.kaggle.com/datasets/prasertk/inflation-interest-and-unemployment-rate**](https://www.kaggle.com/datasets/prasertk/inflation-interest-and-unemployment-rate)

## **About Dataset**

**50 years of historical inflation, interest and unemployment rates by country**

**data source:** [**https://data.worldbank.org**](https://data.worldbank.org/)

**4-Industry classification dataset**

[**https://en.wikipedia.org/wiki/Global\_Industry\_Classification\_Standard**](https://en.wikipedia.org/wiki/Global_Industry_Classification_Standard)

**About data : The Global Industry Classification Standard (GICS) is an** [**industry taxonomy**](https://en.wikipedia.org/wiki/Industry_taxonomy) **developed in 1999 by** [**MSCI**](https://en.wikipedia.org/wiki/MSCI) **and** [**Standard & Poor's**](https://en.wikipedia.org/wiki/Standard_%26_Poor%27s) **(S&P) for use by the global financial community. The GICS structure consists of 11 sectors, 24 industry groups, 69 industries and 158 sub-industries into which S&P has categorised all major** [**public companies**](https://en.wikipedia.org/wiki/Public_company)**. The system is similar to ICB (**[**Industry Classification Benchmark**](https://en.wikipedia.org/wiki/Industry_Classification_Benchmark)**), a classification structure maintained by FTSE Group.**

**GICS is used as a basis for S&P and** [**MSCI**](https://en.wikipedia.org/wiki/MSCI)[**financial market**](https://en.wikipedia.org/wiki/Financial_market) **indexes in which each company is assigned to a sub-industry, and to an industry, industry group, and sector, by its principal business activity. "GICS" is a** [**registered trademark**](https://en.wikipedia.org/wiki/Registered_trademark) **of** [**McGraw Hill Financial**](https://en.wikipedia.org/wiki/McGraw_Hill_Financial) **and MSCI Inc.**

**This data since 1999**

**data source:** [**https://data.worldbank.org**](https://data.worldbank.org/)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**We don’t use all columns and data we filtered this external data set In order to help us in the analysis we want to do with the previous three data**

**All Datasets from wikipedia downloaded using web scraping with python by importing the following libraries :**

**import pandas as pd**

**import requests**

**from bs4 import BeautifulSoup**

**import yfinance as yf**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

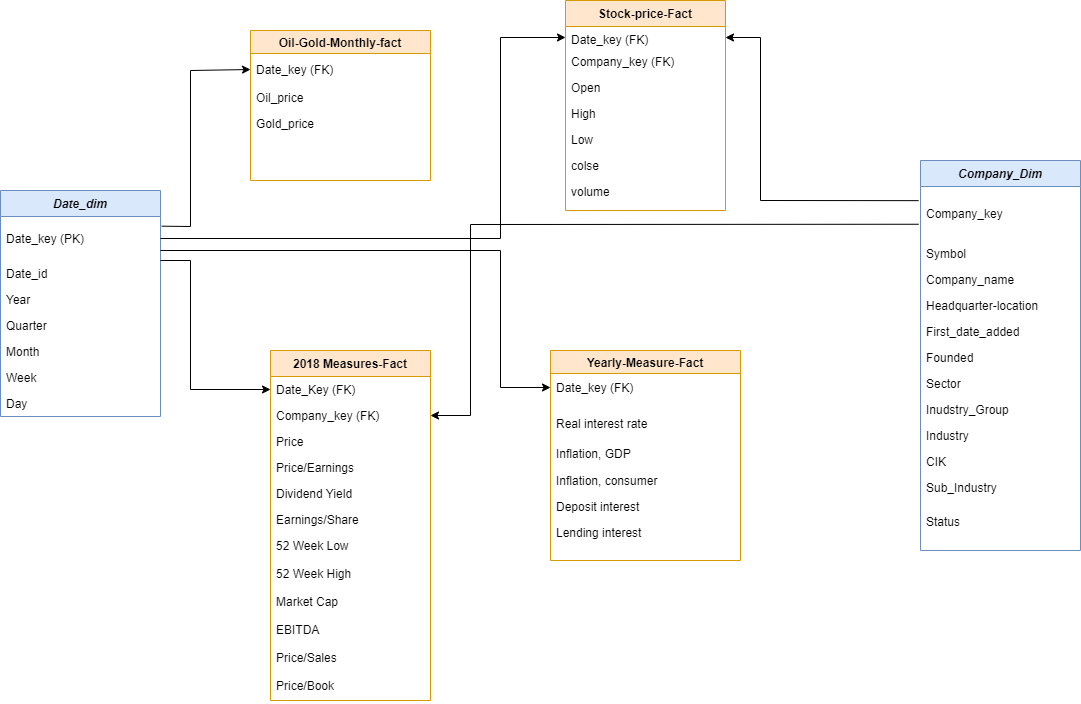
**S&P 500 on August 2018**

[**https://en.wikipedia.org/w/index.php?title=List\_of\_S%26P\_500\_companies&diff=825670660&oldid=821321383**](https://en.wikipedia.org/w/index.php?title=List_of_S%26P_500_companies&diff=825670660&oldid=821321383)

**S&P 500 daily data on 2020**

[**https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks?select=sp500\_stocks.csv**](https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks?select=sp500_stocks.csv)

**Logical Data Warehouse Model:**

****

**Introduction**

The primary objective of this project is to create a concept data warehouse using dimensional modelling techniques. Identify the key stakeholders and business requirements for the same. Based on these requirements develop a design schema for the data warehouse. Create the warehouse using any Oracle SQL developer. Produce Dashboards in support of the requirements using Power BI.

**Schema Used :**

**GALAXY SCHEMA – as the business has more than one business processes.**

**Business Process:**

stock market analysis ,analyize companies statues on S&P 500

**Grantuality:** Atomic ber day

**Dimensions:**

The information obtained from the dimension table is used to do the analysis. The dimension tables contain descriptive. Each dimension table has a unique Primary Key which is then used to create the linking relation with the fact tables.For the Stock data Warehouse Model created the dimensions Company Dimension and Date Dimension

**Company Dimension** : contain various attributes that provide further information about companies in S&P Index this table merged from two datasets from kaggle **s&p data and Industry classification dataset**

**Date Dimension :** contain daily dates from 2013 to 2020 to filtrate and doing the analysis in specific duration

**Fact Tables:**

The fact tables usually contain numeric measures of the subject of analysis. I have created two aggregate fact tables

**Stock\_Price\_fact**

This fact table takes the Stock Prices from the Companies using This table contain daily values about stock prices like high price , low price , open price , close price and volume of trading .

This fact table in relation with the two dimensions tables using two forigen keys company\_key and Date\_key

**2018\_measur\_fact:**

This fact table takes the Attributes from the Companies in 2018 This table contain values price/earning , dividend yield , earning/share , 52 week low , 52 week high , price/sales , price/book , EBITDA and market cap

This fact table in relation with the two dimensions tables using two forigen keys company\_key and Date\_key

**Oil\_Gold\_Monthy\_Fact :**

This fact table takes the Attributes from the Oil and Gold Situations Across Months This table contain values Oil price and Gold Price

This fact table in relation with the One Date dimension tables using forigen Date\_key

**Yearly\_Measur\_Fact:**

This fact table takes the Attributes from the Inflation and Interest Situations Across Years This table contain values real interest rate , lending interest , deposit interest , inflation GDP and inflation consumer

This fact table in relation with the One Date dimension tables using forigen Date\_key

**Physical model:**

**Dimensions:**

**CREATE TABLE "COMPANY\_DIM"**

**( "COMPANY\_KEY" VARCHAR2(38 BYTE),**

**"SYMBOL" VARCHAR2(26 BYTE),**

**"COMPANY\_NAME" VARCHAR2(128 BYTE),**

**"SEC\_FILINGS" VARCHAR2(26 BYTE),**

**"GICS\_SECTOR" VARCHAR2(26 BYTE),**

**"GICS\_INDUSTRY\_GROUP" VARCHAR2(128 BYTE),**

**"GICS\_SECTOR\_HIRARCHY" VARCHAR2(128 BYTE),**

**"GICS\_SUB\_INDUSTRY" VARCHAR2(128 BYTE),**

**"HEADQUARTERS\_LOCATION" VARCHAR2(128 BYTE),**

**"DATE\_FIRST\_ADDED" DATE,**

**"CIK" NUMBER(38,0),**

**"STATUS" VARCHAR2(26 BYTE)**

**)**

CREATE TABLE DATE\_DIMENSION

(

DATE\_KEY DATE NOT NULL,

FULL\_DATE\_DESCRIPTION VARCHAR2(64) NOT NULL,

DAY\_OF\_WEEK NUMBER(1,0) NOT NULL,

DAY\_OF\_MONTH NUMBER(2,0) NOT NULL,

DAY\_OF\_YEAR NUMBER(3,0) NOT NULL,

LAST\_DAY\_OF\_WEEK\_INDICATOR CHAR(1) NOT NULL,

LAST\_DAY\_OF\_MONTH\_INDICATOR CHAR(1) NOT NULL,

WEEK\_ENDING\_DATE DATE NOT NULL,

MONTH\_NUMBER NUMBER(2,0) NOT NULL,

MONTH\_NAME VARCHAR2(32) NOT NULL,

YEAR\_MONTH CHAR(32) NOT NULL,

QUARTER\_NUMBER NUMBER(1,0) NOT NULL,

YEAR\_QUARTER CHAR(32) NOT NULL,

YEAR\_NUMBER NUMBER(4,0) NOT NULL,

CONSTRAINT DATE\_DIMENSION\_PK PRIMARY KEY (DATE\_KEY)

)

—--------------------------------------------------------------------------------------------------------

Procedure to insert dates in the date dime

++++++++++++++++++++++++++++++++++++++++++++

create or replace PROCEDURE sp\_DATE\_DIMENSION(v\_START\_YEAR IN INT, v\_END\_YEAR IN INT) AS

--Declare two variables as DATE datatypes

v\_CURRENT\_DATE DATE;

v\_END\_DATE DATE;

BEGIN

--Assign the start year and end year to it's respective variables

v\_CURRENT\_DATE := TO\_DATE('0101' || v\_START\_YEAR, 'MMDDYYYY');

v\_END\_DATE := TO\_DATE('1231' || v\_END\_YEAR, 'MMDDYYYY');

--Clear/Dump what is currently stored in the table

DELETE FROM DATE\_DIMENSION;

--Check the condition to see if the start year is less than the end year (Input Parameters)

WHILE v\_CURRENT\_DATE <= v\_END\_DATE

LOOP

--DATE\_DIMENSION Table

INSERT INTO DATE\_DIMENSION

(

DATE\_KEY,

FULL\_DATE\_DESCRIPTION,

DAY\_OF\_WEEK,

DAY\_OF\_MONTH,

DAY\_OF\_YEAR,

LAST\_DAY\_OF\_WEEK\_INDICATOR,

LAST\_DAY\_OF\_MONTH\_INDICATOR,

WEEK\_ENDING\_DATE,

MONTH\_NUMBER,

MONTH\_NAME,

YEAR\_MONTH,

QUARTER\_NUMBER,

YEAR\_QUARTER,

YEAR\_NUMBER

)

VALUES

(

v\_CURRENT\_DATE, --DATE\_KEY

TO\_CHAR(v\_CURRENT\_DATE, 'Day, Month DD, YYYY'), --FULL\_DATE\_DESCRIPTION

TO\_NUMBER(TO\_CHAR(v\_CURRENT\_DATE, 'D')) -1, --DAY\_OF\_WEEK

TO\_CHAR(v\_CURRENT\_DATE,'DD'), --DAY\_OF\_MONTH

TO\_CHAR(v\_CURRENT\_DATE,'DDD'), --DAY\_OF\_YEAR

CASE --LAST\_DAY\_OF\_WEEK\_INDICATOR

WHEN TO\_CHAR(v\_CURRENT\_DATE,'FMDay') = 'Saturday' THEN 'Y'

ELSE 'N'

END,

CASE --LAST\_DAY\_OF\_MONTH\_INDICATOR

WHEN LAST\_DAY(v\_CURRENT\_DATE) = v\_CURRENT\_DATE THEN 'Y'

ELSE 'N'

END,

CASE --WEEK\_ENDING\_DATE OF CURRENT WEEK ENDING ON SATURDAY

WHEN TO\_CHAR(v\_CURRENT\_DATE,'FMDay') = 'Saturday' THEN v\_CURRENT\_DATE

ELSE NEXT\_DAY(v\_CURRENT\_DATE,'SATURDAY')

END,

TO\_CHAR(v\_CURRENT\_DATE,'MM'), --MONTH\_NUMBER

TO\_CHAR(v\_CURRENT\_DATE,'MONTH'), --MONTH\_NAME

TO\_CHAR(v\_CURRENT\_DATE,'MONTH YYYY'), --YEAR\_MONTH

TO\_CHAR(v\_CURRENT\_DATE,'Q'), --QUARTER\_NUMBER

TO\_CHAR(v\_CURRENT\_DATE,'YYYY Q'), --YEAR\_QUARTER

TO\_CHAR(v\_CURRENT\_DATE,'YYYY') --YEAR\_NUMBER

);

--Increment and assign the current date value to be re-evaluated

v\_CURRENT\_DATE := v\_CURRENT\_DATE + 1;

END LOOP;

END;

====================================================

–Calling

declare

begin

sp\_DATE\_DIMENSION (2013, 2022);

End;

**Facts:**

**CREATE TABLE "DAILY\_STOCK\_FACT"**

**( "COMPANY\_KEY" VARCHAR2(38 BYTE),**

**"DATE\_KEY" DATE,**

**"OPEN" NUMBER(38,2),**

**"HIGH" NUMBER(38,2),**

**"LOW" NUMBER(38,2),**

**"CLOSE" NUMBER(38,2),**

**"VOLUME" NUMBER(38,0)**

**)**

**CREATE TABLE "MONTHLY\_OIL\_GOLD\_FACT"**

**( "DATE\_KEY" DATE,**

**"GOLD\_PRICE" NUMBER(38,3),**

**"CRUDE\_OIL\_PRICE" NUMBER(38,4)**

**)**

**CREATE TABLE "STOCK\_2018\_FACT"**

**( "COMPANY\_KEY" VARCHAR2(38 BYTE),**

**"PRICE" NUMBER(38,2),**

**"PRICE\_EARNINGS" NUMBER(38,2),**

**"DIVIDEND\_YIELD" NUMBER(38,8),**

**"EARNINGS\_SHARE" NUMBER(38,2),**

**"WEEK\_LOW\_52" NUMBER(38,4),**

**"WEEK\_HIGH\_52" NUMBER(38,4),**

**"MARKET\_CAP" NUMBER(38,0),**

**"EBITDA" NUMBER(38,0),**

**"PRICE\_SALES" NUMBER(38,8),**

**"PRICE\_BOOK" NUMBER(38,2),**

**"SEC\_FILINGS" VARCHAR2(128 BYTE),**

**"DATE\_KEY" DATE**

**)**

**CREATE TABLE "YEARLY\_INF\_INTEREST\_FACT"**

**( "DATE\_KEY" DATE,**

**"INFLATION\_CONSUMER\_PRICES" NUMBER(38,15),**

**"INFLATION\_GDP\_DEFLATOR" NUMBER(38,15),**

**"REAL\_INTEREST\_RATE" NUMBER(38,14),**

**"LENDING\_INTEREST\_RATE" NUMBER(38,14)**

**)**

**Data Preparation using Excel:**

**all\_stocks\_5yr**

Promoted Headers and delete column (Name)

add column company\_key as forien key from company\_Dim table that we created,

No nulls handled

Change date data type

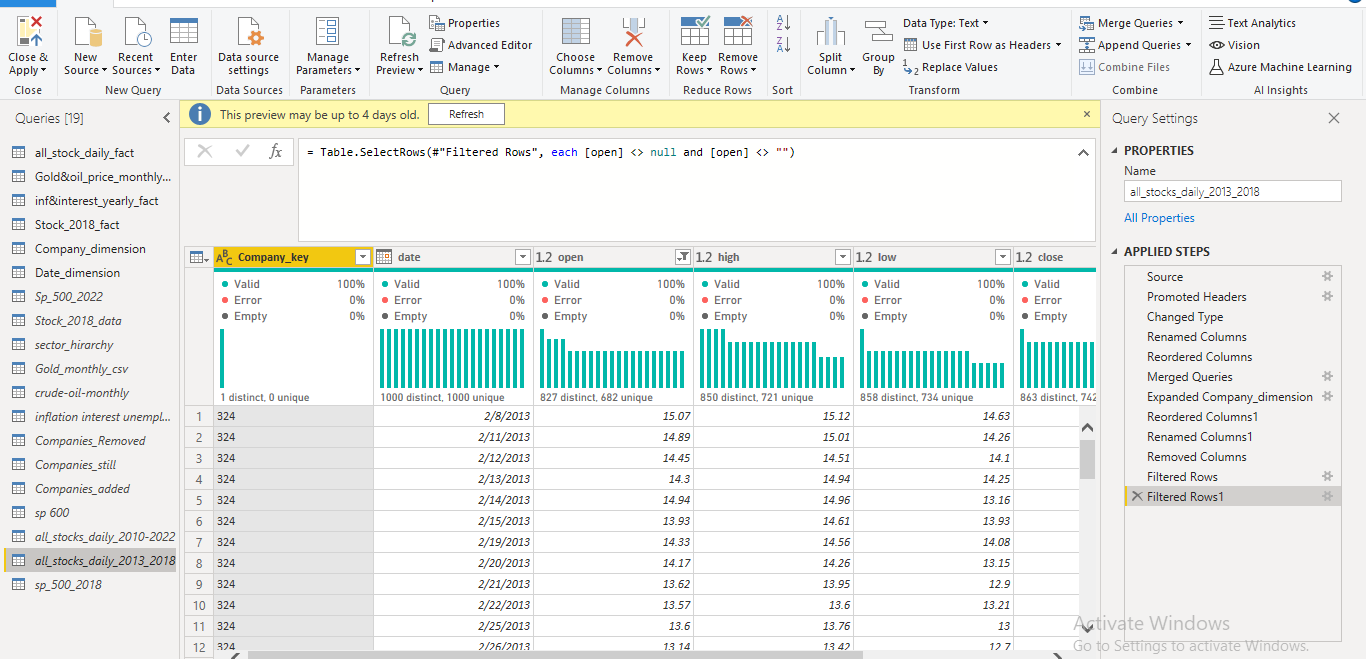
Merge this table with another extra data **S&P 500 on August 2018**

[**https://en.wikipedia.org/w/index.php?title=List\_of\_S%26P\_500\_companies&diff=825670660&oldid=821321383**](https://en.wikipedia.org/w/index.php?title=List_of_S%26P_500_companies&diff=825670660&oldid=821321383)

to get 100 company missed from 2018 until 2022

Rename table to company\_dimantion

Append this table with another extra data **S&P 500 daily data on 2020**

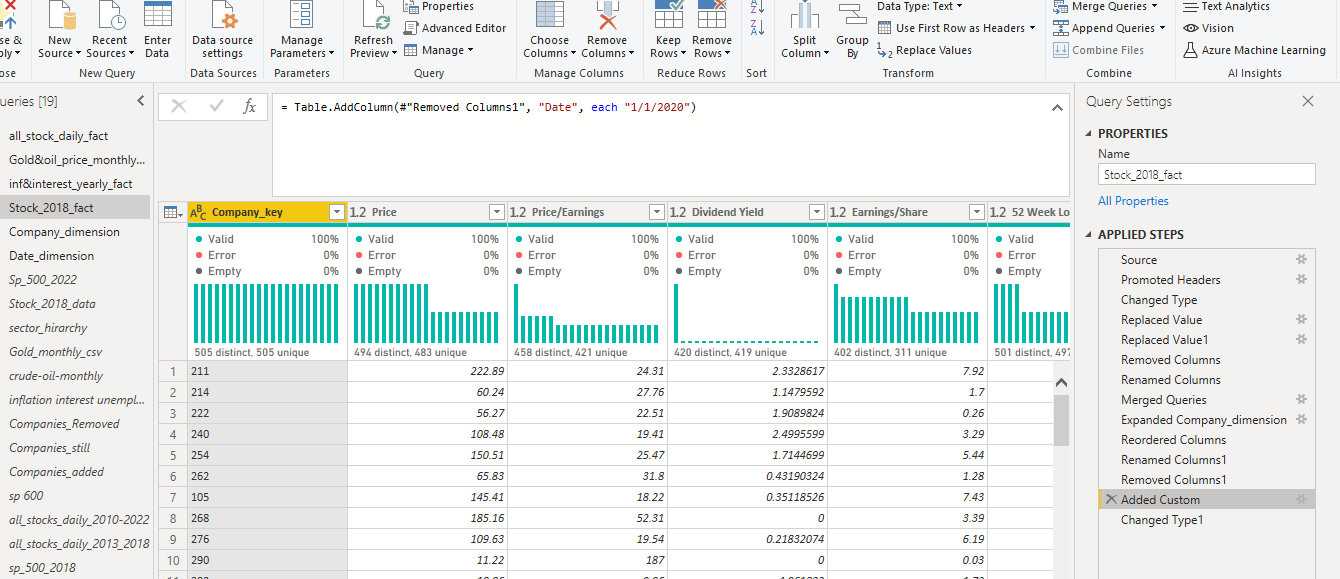
[**https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks?select=sp500\_stocks.csv**](https://www.kaggle.com/datasets/andrewmvd/sp-500-stocks?select=sp500_stocks.csv)To complete range of date until 2022the table called all\_stock\_daily\_fact

**constituents-financials\_csv**

Promoted Headers ,removed column (symbol,name,sector)

Replace null with zero in price/earning and price/book

Add column company\_key as forien key from company\_dim table that we created, added column date take range date from 2018

Change data type table name stock\_fact\_2018

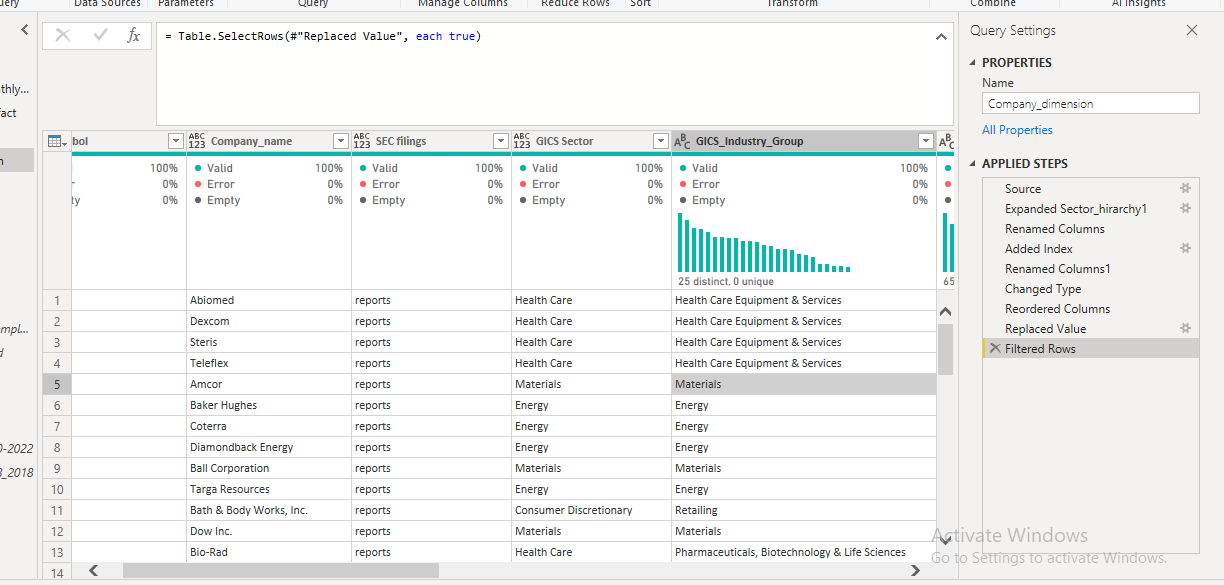
## S&P 500 component stocks

Merge table with table in Industry classification dataset to complete my information,Exclude column(sector) because it’s already Existing

Change date\_first added data type

Add column company\_key as surgite key to make relation between this table and facts tables

Add column statues in table through merge between tables stock\_fact2018 and all\_stock\_daily\_fact to know companies which were from 2018 to 2022 take statues still,companies weren’t there in 2022 take statues removed and companies which become added in 2022 take statues added



**Date dimension table**

It’s created by the following equation:

Source = List.Dates(#date(2013,01,01), 365\*10, #duration(1,0,0,0)),

#"Sorted Items" = List.Sort(Source,Order.Ascending),

#"Converted to Table" = Table.FromList(#"Sorted Items", Splitter.SplitByNothing(), null, null, ExtraValues.Error),

#"Changed Type" = Table.TransformColumnTypes(#"Converted to Table",{{"Column1", type date}}),

#"Inserted Year" = Table.AddColumn(#"Changed Type", "Year", each Date.Year([Column1]), Int64.Type),

#"Inserted Quarter" = Table.AddColumn(#"Inserted Year", "Quarter", each Date.QuarterOfYear([Column1]), Int64.Type),

#"Inserted Month" = Table.AddColumn(#"Inserted Quarter", "Month", each Date.Month([Column1]), Int64.Type),

#"Inserted Week of Year" = Table.AddColumn(#"Inserted Month", "Week of Year", each Date.WeekOfYear([Column1]), Int64.Type)

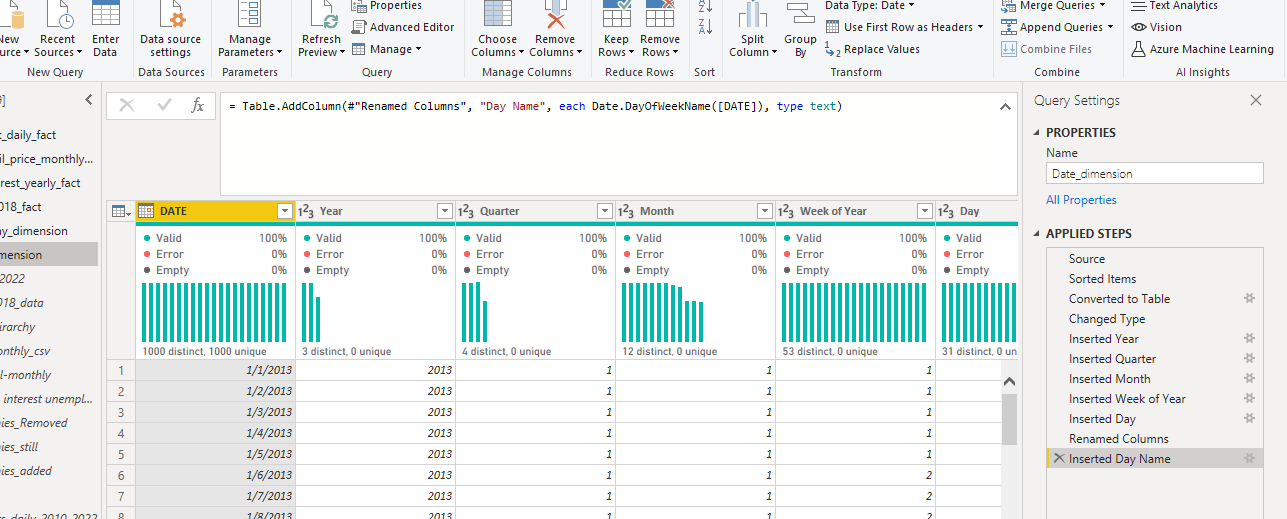
#"Inserted Day" = Table.AddColumn(#"Inserted Week of Year", "Day", each Date.Day([Column1]), Int64.Type),

#"Renamed Columns" = Table.RenameColumns(#"Inserted Day",{{"Column1", "DATE"}}),

#"Inserted Day Name" = Table.AddColumn(#"Renamed Columns", "Day Name", each Date.DayOfWeekName([DATE]), type text)

in

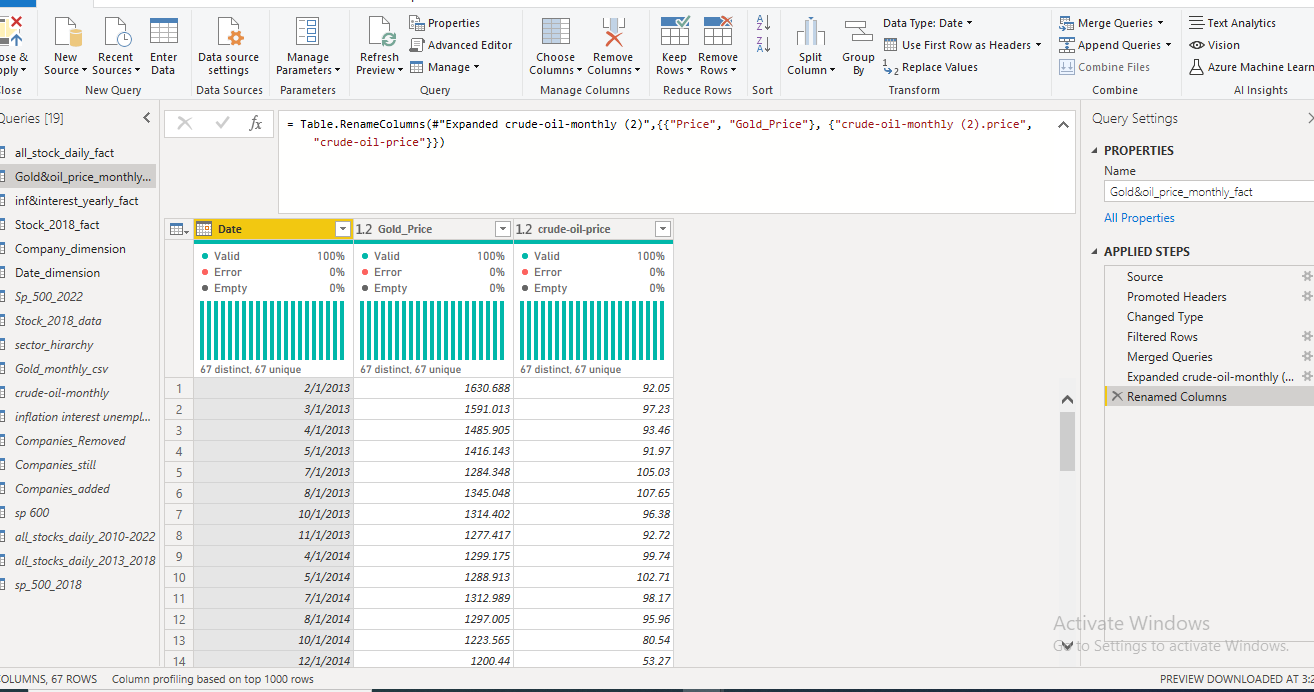
#"Inserted Day Name"



**Extra data:**

Merge between gold data table and crud\_oil data table excluding columns(change-perc change) from crud\_oil

And determined date from 2013 to2020

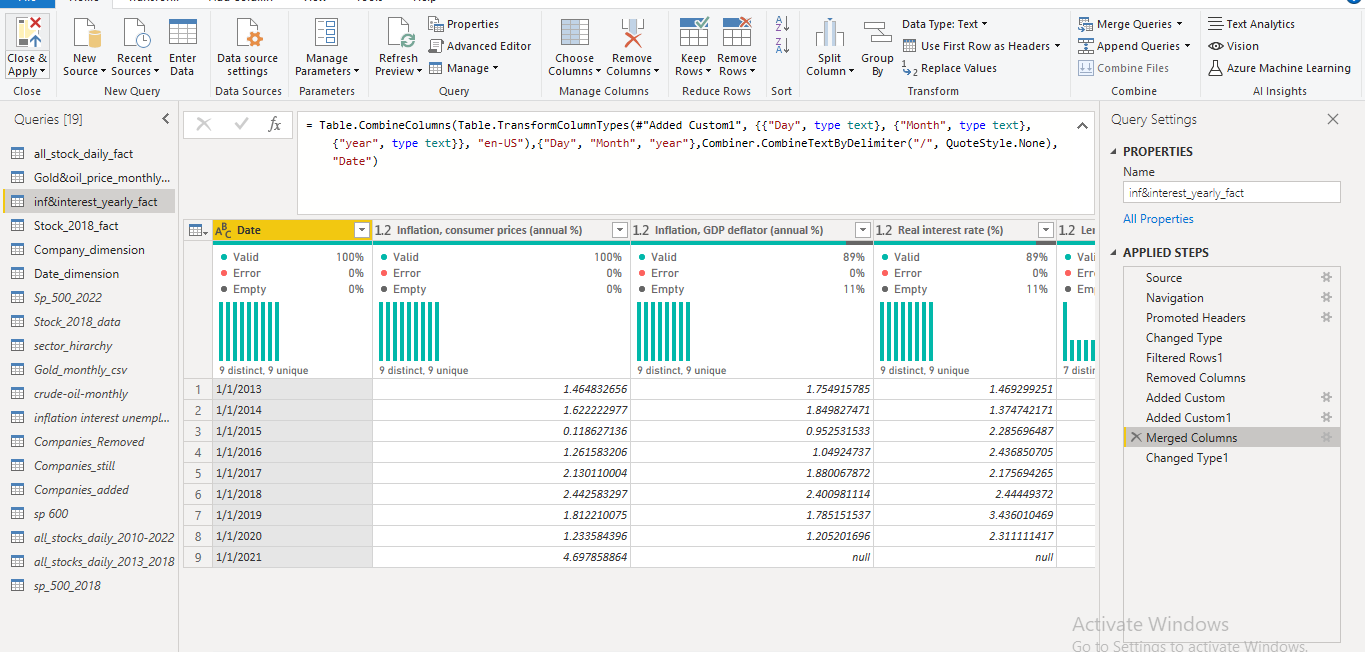
****

**Inflation and interesting data** removed columns( country,Deposit interest rate (%),,Unemployment, total (% of total labour force) (national estimate),Unemployment, total (% of total labour force) (modelled ILO estimate),iso3c,iso2c,admin region,incomeLevel)

Add custom month,and day

Merge columns with year

Change data type



**Import data:**using SQL developer

**Queries:**

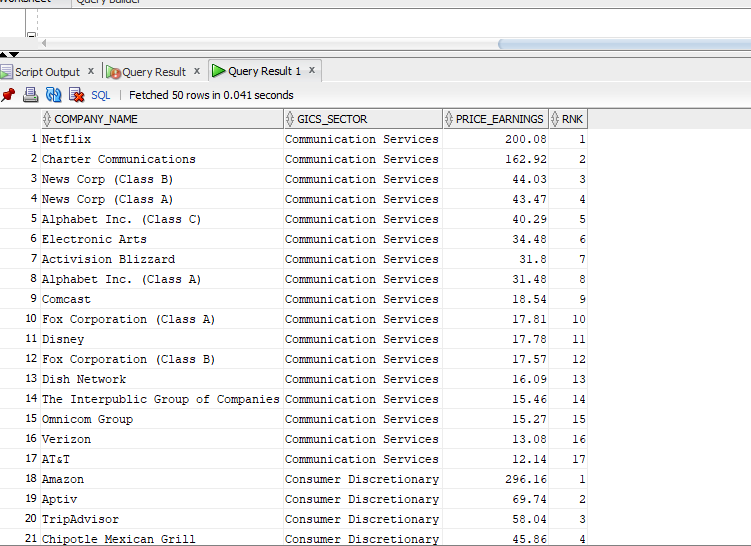
**Rank Companies according to earning price in each sector in 2018**

**Select c.company\_name , c.gis\_sector , s.price\_earnings , rnk**

**From (Select c.company\_name , c.gis\_sector , s.price\_earnings , rank() Over (partition by c.gis\_sector order by price\_earnings desc ) rnk From company\_dim c inner join stock\_2018 s on c.company\_key = s.company\_key**

**)**

**Where rnk = 1**

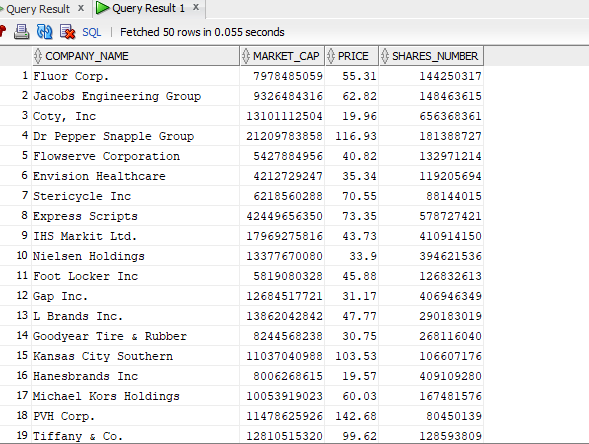
****

**how are number of shares for each company?**

**select c.company\_name,s.Market\_cap,s.price,trunc(Market\_cap/price) as shares\_number from**

**stock\_2018 s, company\_dim c**

**where c.company\_key= s.company\_key;**

****

**Top price\_earning sub\_industries;**

**select distinct GICS\_SUB\_INDUSTRY ,avg (price\_earnings)**

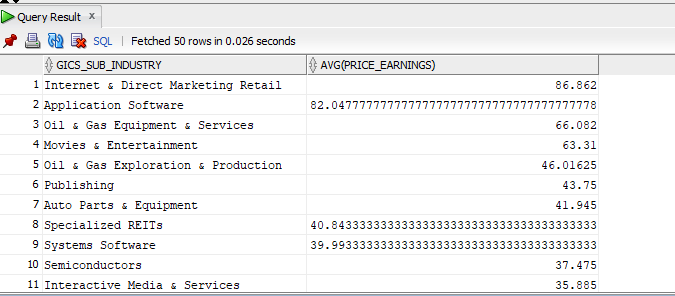
**from stock\_2018\_fact join company\_dim**

**on company\_dim.COMPANY\_KEY= stock\_2018\_fact.COMPANY\_KEY**

**where DATE\_KEY =TO\_DATE( '14/02/2018','DD-MM-YYYY')**

**group by GICS\_SUB\_INDUSTRY**

**order by avg (price\_earnings)desc ;**

****

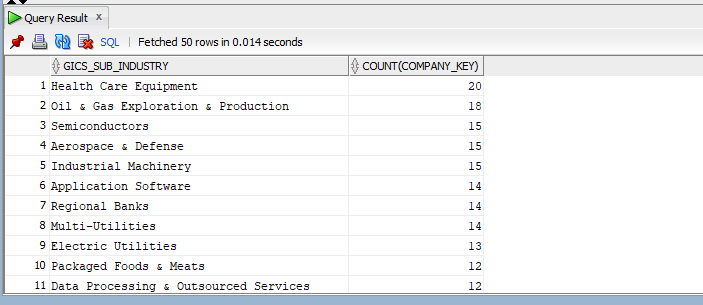
**Highest company counts sub\_industry;**

**select GICS\_SUB\_INDUSTRY ,count (COMPANY\_KEY)**

**from company\_dim**

**group by GICS\_SUB\_INDUSTRY**

**order by count (COMPANY\_KEY) desc**

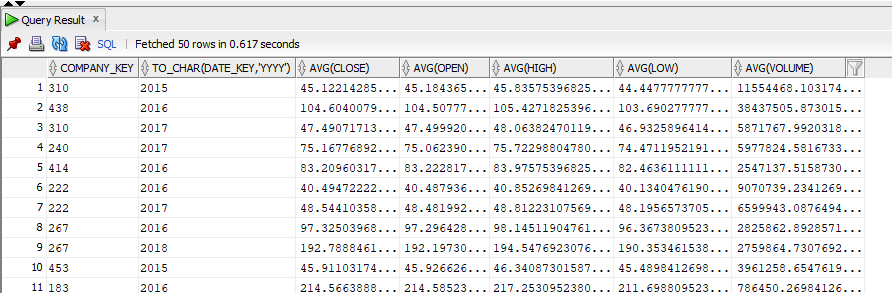
****

**Average open , close , low , high , volume per company yearly**

**select company\_key , to\_char(date\_key ,'yyyy'), avg(close),avg(open), avg(high), avg(low),avg(volume)**

**from daily\_stock\_fact**

**group by company\_key , to\_char(date\_key ,'yyyy')**

****

**Previous close, change , %change per stock**

**select company\_key , DATE\_KEY ,close ,**

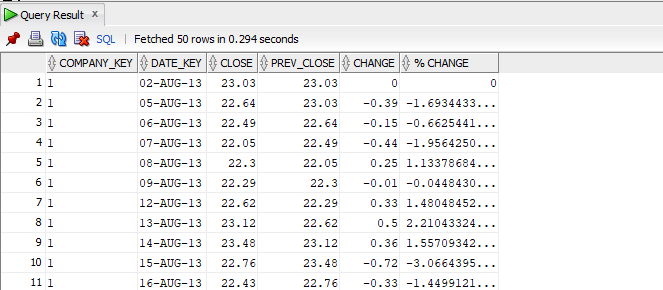
**LAG(CLOSE,1,CLOSE)OVER(partition by company\_key ORDER BY DATE\_KEY)AS PREV\_CLOSE**

**,CLOSE -(LAG(CLOSE,1,CLOSE)OVER(partition by company\_key ORDER BY DATE\_KEY))AS CHANGE**

**,(CLOSE -(LAG(CLOSE,1,CLOSE)OVER(partition by company\_key ORDER BY DATE\_KEY)))/**

**LAG(CLOSE,1,CLOSE)OVER(partition by company\_key ORDER BY DATE\_KEY)\*100 AS "% CHANGE"**

**from daily\_stock\_fact**



**Comparison of End Fiscal year to all sector quarters**

**WITH sept\_end AS (SELECT sf.close, MAX(dd.date) OVER(PARTITION BY dd.year)**

**FROM date\_dimension AS dd, all\_stocks\_daily\_fact AS sf**

**WHERE dd.date = sf.date**

**AND month = 9**

**GROUP BY sf.close, dd.year),**

**avg\_q1 AS (SELECT dd.year, AVERAGE(sf.close),**

**FROM all\_stocks\_Daily\_fact AS sf, date\_dimension dd**

**WHERE sf.date = dd.date**

**AND dd.quarter = 4**

**GROUP BY dd.year),**

**avg\_q2 AS (SELECT dd.year, AVERAGE(sf.close),**

**FROM all\_stocks\_Daily\_fact AS sf, date\_dimension dd**

**WHERE sf.date = dd.date**

**AND dd.quarter = 1**

**GROUP BY dd.year),**

**avg\_q3 AS (SELECT dd.year, AVERAGE(sf.close),**

**FROM all\_stocks\_Daily\_fact AS sf, date\_dimension dd**

**WHERE sf.date = dd.date**

**AND dd.quarter = 2**

**GROUP BY dd.year),**

**avg\_q4 AS (SELECT dd.year, AVERAGE(sf.close),**

**FROM all\_stocks\_Daily\_fact AS sf, date\_dimension dd**

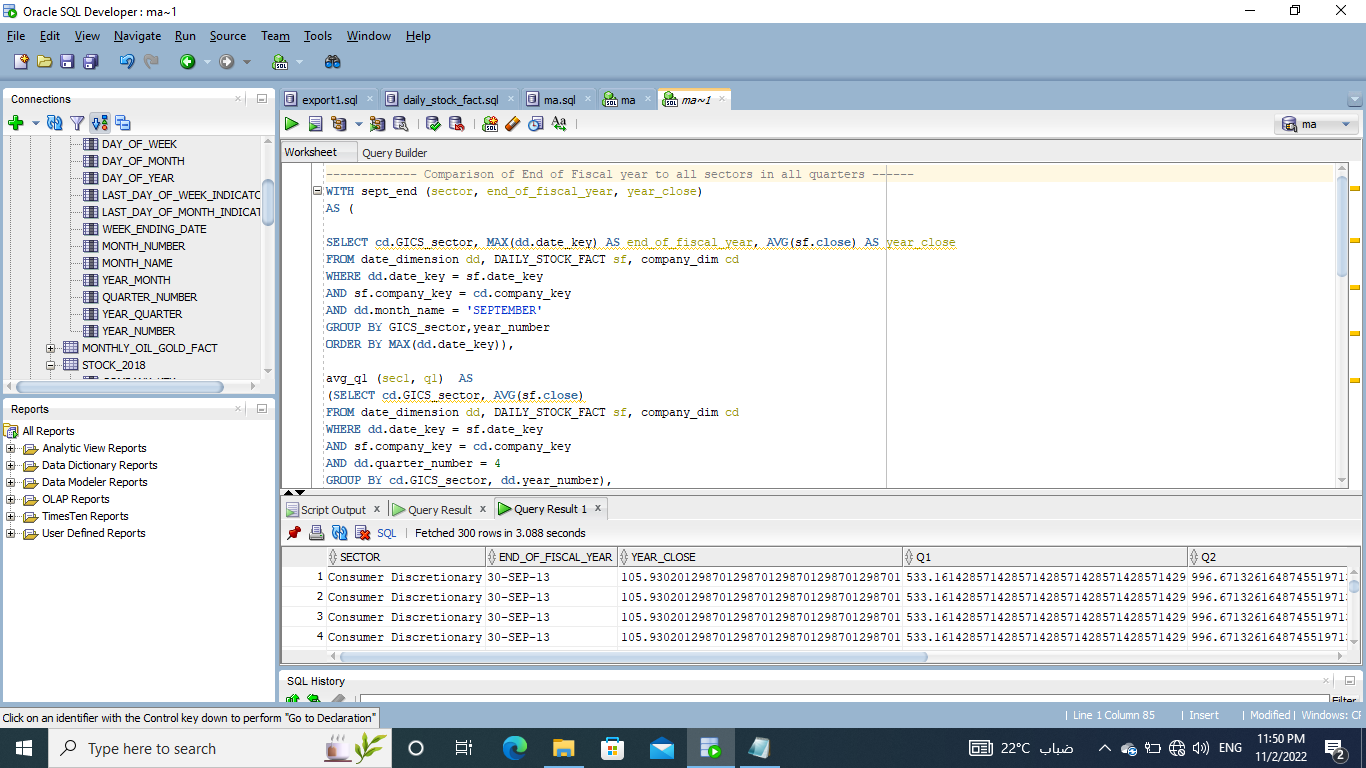
**WHERE sf.date = dd.date**

**AND dd.quarter = 3**

**GROUP BY dd.year),**

**SELECT stk.sector, sept\_end, avg\_q1, avg\_q2, avg\_q3, avg\_q4**

**FROM all\_stock\_daily\_fact AS stk**

****

**indicator-Measure = IF('indicators-Parameter'[indicators-Parameter Value]=1, MAX(Stock\_2018\_fact[Price/Earnings] , IF('indicators-Parameter'[indicators-Parameter Value] = 2 , MAX(Stock\_2018\_fact[Dividend Yield] , IF('indicators-Parameter'[indicators-Parameter Value] = 3 , MAX(Stock\_2018\_fact[Earnings/Share] , IF('indicators-Parameter'[indicators-Parameter Value] = 4 , MAX(Stock\_2018\_fact[Market Cap] , IF('indicators-Parameter'[indicators-Parameter Value] = 5 , MAX(Stock\_2018\_fact[EBITDA] , MAX(Stock\_2018\_fact[Price/Book])))))))))))**

**select \* from (**

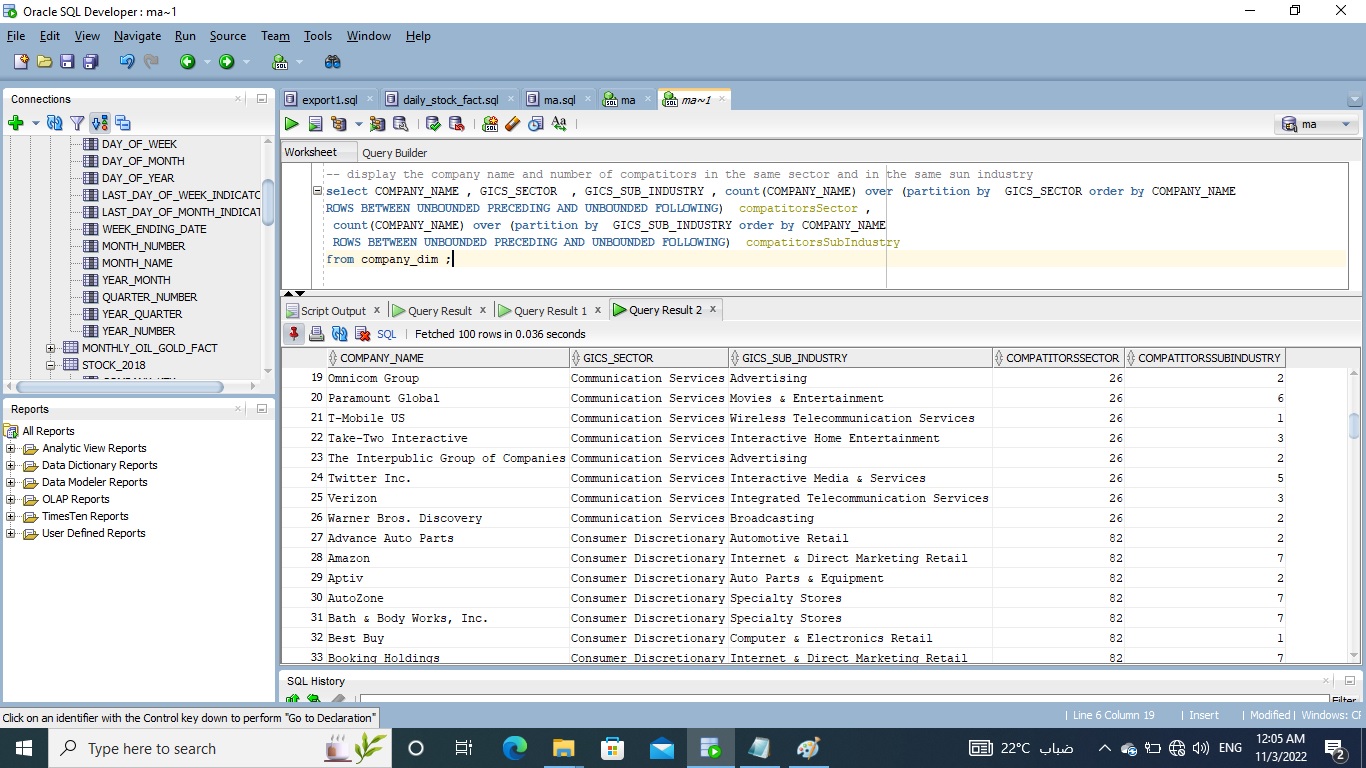
**select YEAR\_NUMBER , COMPANY\_NAME , avg(VOLUME) over() , dense\_rank() over (partition by YEAR\_NUMBER order by VOLUME desc ) rnk**

**from company\_dim c , daily\_stock\_fact s , date\_dimension d**

**where c.COMPANY\_KEY = s.COMPANY\_KEY**

**and d.DATE\_KEY = s.DATE\_KEY) new\_table**

**where new\_table.rnk < 5 ;**

**The Color pallet we Used:**

**www.Color.adobe.com**

****#005370

#7BC2DB

#8CBD0F

#736762

#9C3500

DAX Calculations:-

Added Column:-

Prev close =

VAR CurrentDate = 'all\_stock\_daily\_fact'[date]

VAR PreviousDate = CALCULATE(LASTDATE('all\_stock\_daily\_fact'[date]),FILTER('all\_stock\_daily\_fact','all\_stock\_daily\_fact'[date]<CurrentDate))

var CurrentTicker = all\_stock\_daily\_fact[Company\_key]

RETURN

CALCULATE(SUM(all\_stock\_daily\_fact[close]),

FILTER('all\_stock\_daily\_fact',

all\_stock\_daily\_fact[Company\_key] = CurrentTicker

&&

'all\_stock\_daily\_fact'[date]=PreviousDate))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Change = all\_stock\_daily\_fact[close]-all\_stock\_daily\_fact[Prev close]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

% change = DIVIDE((all\_stock\_daily\_fact[close]-all\_stock\_daily\_fact[Prev close]),all\_stock\_daily\_fact[Prev close],blank())

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Revenue = Stock\_2018\_fact[Market Cap]/Stock\_2018\_fact[Price/Sales]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Added Measures:-

1M Avg =

var interval=-1

return

CALCULATE(

sum(all\_stock\_daily\_fact[close]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

/

CALCULATE(

DISTINCTCOUNT(all\_stock\_daily\_fact[date]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3M Avg =

var interval=-3

return

CALCULATE(

sum(all\_stock\_daily\_fact[close]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

/

CALCULATE(

DISTINCTCOUNT(all\_stock\_daily\_fact[date]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6M Avg =

var interval=-6

return

CALCULATE(

sum(all\_stock\_daily\_fact[close]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

/

CALCULATE(

DISTINCTCOUNT(all\_stock\_daily\_fact[date]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

9M Avg =

var interval=-9

return

CALCULATE(

sum(all\_stock\_daily\_fact[close]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

/

CALCULATE(

DISTINCTCOUNT(all\_stock\_daily\_fact[date]),

DATESINPERIOD(all\_stock\_daily\_fact[date],LASTDATE(all\_stock\_daily\_fact[date]),interval,MONTH))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1M = SWITCH(ALLSELECTED('Table'[value]),"1M",all\_stock\_daily\_fact[1M Avg])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3M = SWITCH(ALLSELECTED('Table'[value]),"3M",all\_stock\_daily\_fact[3M Avg])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6M = SWITCH(ALLSELECTED('Table'[value]),"6M",all\_stock\_daily\_fact[6M Avg])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

9M = SWITCH(ALLSELECTED('Table'[value]),"9M",all\_stock\_daily\_fact[9M Avg])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Losers Title = "Top Losers of"& selectedvalue(Date\_dimension[Year],"All years")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Gainers Title = "Top Gainers of"& selectedvalue(Date\_dimension[Year],"All years")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title = "Daily Stock Prices on "& selectedvalue(Date\_dimension[Month],"All years")& "/"& selectedvalue(Date\_dimension[Year],"All years" )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title for bar = "Top fluidity" &'Rank Componies'[Rank Componies Value]&"Company out of total"&COUNTROWS(Company\_dimension)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Rank Componies = GENERATESERIES(0, DISTINCTCOUNT(Company\_dimension[Symbol]),1)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Rank Componies Value = SELECTEDVALUE('Rank Componies'[Rank Componies])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Quarter par = GENERATESERIES(1, 4, 1)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

September END = CALCULATE(MAXX( Date\_dimension, Date\_dimension[DATE]), Date\_dimension[Month] = 9, Date\_dimension[Day] = 30)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

End of september value = CALCULATE(AVERAGEX(all\_stock\_daily\_fact, all\_stock\_daily\_fact[close]), FILTER(Date\_dimension, Date\_dimension[DATE] = [September END]))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

par modification = SWITCH(TRUE(),'Quarter par'[Quarter par Value] = 1, 4, 'Quarter par'[Quarter par Value] -1)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

quarters calc = CALCULATE( AVERAGEX(all\_stock\_daily\_fact, all\_stock\_daily\_fact[close]),

FILTER( Date\_dimension, Date\_dimension[Quarter] = [par modification] ))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fiscal Year Title = SELECTEDVALUE(Date\_dimension[Year]) & UNICHAR(10) & "Q(" & 'Quarter par'[Quarter par Value] & ")"

fluidity = SUM(all\_stock\_daily\_fact[volume])\*SUM(all\_stock\_daily\_fact[close])

num\_shares = Stock\_2018\_fact[Market Cap]/Stock\_2018\_fact[Price]