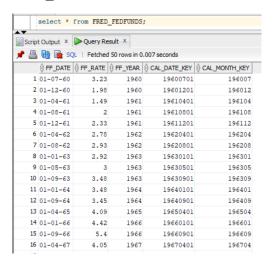
P2 Assignment: Analytical SQL

Query 1: Computing RANKs

Assign a RANK and DENSE_RANK to each year based on the federal funds rate (using the FRED_FEDFUNDS data) or a data set of your choice. How does the RANK differ from the DENSE_RANK assigned in the query?

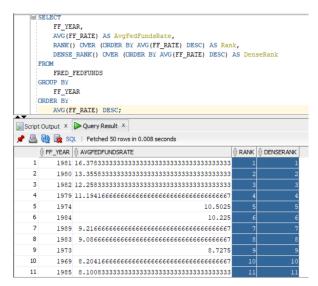
Ans:

FRED_FEDFUNDS Data:



Rank and Dense Rank:

The below query will provide a yearly breakdown of the Federal Funds Rate, ranked by the average rate for each year, showcasing the differences between RANK and DENSE_RANK in handling ties.



FF YEAR: Groups the data by year.

AVG(FF_RATE) AS AvgFedFundsRate: Calculates the average Federal Funds Rate for each year.

RANK() OVER (ORDER BY AVG(FF_RATE) DESC) AS Rank: Assigns a rank to each year based on the average Federal Funds Rate, with gaps in the ranking for ties.

DENSE_RANK() OVER (ORDER BY AVG(FF_RATE) DESC) AS DenseRank: Similar to RANK(), but without gaps for ties, ensuring a continuous sequence of rank numbers.

Difference between Rank and Dense Rank:

The key difference between the RANK and DENSE_RANK functions in SQL is how they handle duplicate values. The RANK function may skip positions after equal rankings, resulting in gaps in the ranking, while the DENSE_RANK function does not skip any positions, ensuring there are no gaps in the ranking.

Query 2: Creating Bins with NTILE

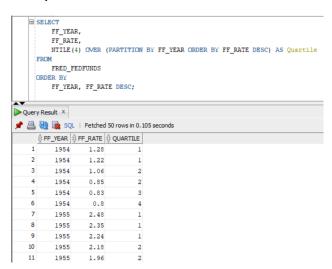
It is often useful to assign bins or otherwise categorize data in a warehouse environment with functions like NTILE.

Use the NTILE function to bin indicator values, such as the federal funds rate (FRED_FEDFUNDS) or data set of your choice using quartiles or even deciles.

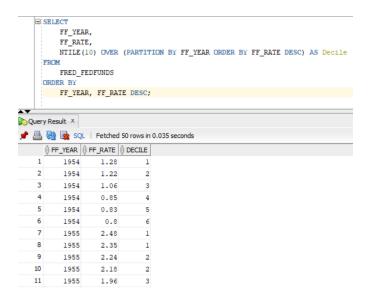
Group the data by year or decade and use NTILE within the partitions to assign values.

Ans: To use the NTILE function for binning the federal funds rate (FF_RATE) into quartiles within each year, we can modify the query to partition the data by FF_YEAR and then apply NTILE(4) to create four bins (quartiles) for each year. We can do create 10 bins and do the partition with deciles as well.

For Quartiles:



For Deciles:



Query 3: Correlations (CORR)

The correlations between different economic indicators are often important. The whole idea of diversification is based on holding investments that are somewhat uncorrelated, so their values do not move in lock step. A diversified portfolio should do better under a variety of economic conditions, thereby reducing the risk of large swings in value.

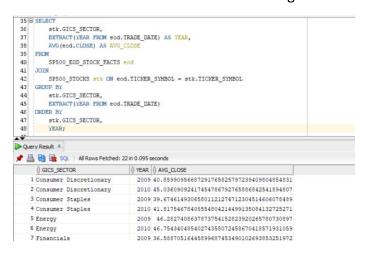
Use analytic SQL to assess the correlation between some selected stocks within and across sectors.

Assess some correlations on other financial data.

Ans:

Aggregate Stock Performance by Sector and Year

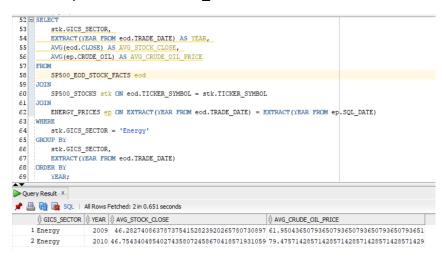
First, aggregate the average closing prices of stocks within each GICS sector for each year. This will serve as a foundation for assessing correlations across sectors.



Comparative Analysis for Correlation

Incorporating Other Financial Data

To incorporate energy prices or political party data, we can join these tables on the year and compare trends. For example, comparing average stock prices in the Energy sector with crude oil prices from ENERGY PRICE.



Query 4: Leading and Lagging Indicators

Economic indicators are often characterized as "leading or lagging" indicators. We can use analytic SQL to investigate the temporal relationships between indicators, especially using the LAG/LEAD functions.

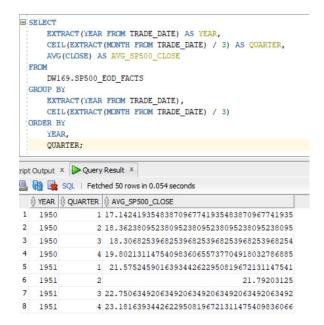
For instance, the stock market is often considering a leading indicator of more general economic activity. You might look at a stock index such as the S&P 500 (SP500_EOD_FACTS) or the DJIA (DBERNDT.DJIA_WEEKLY_FACTS) and compare that with a general measure like Gross Domestic Product (FRED_GDP).

Investigate other leading and lagging indicator relationships.

Ans:

Quarterly Aggregation of S&P 500 Index

First, we aggregate the S&P 500 index data on a quarterly basis to match the reporting period of GDP data.SP500_EOD_FACTS contains daily closing values, we can aggregate these values to get an average or closing value for each quarter.



Analyzing GDP Data with S&P 500 Performance

Next, joining this aggregated S&P 500 data with the FRED_GDP table on the year and quarter to analyze their relationship. We can use the LAG or LEAD functions to compare the quarter-over-quarter changes in GDP with the S&P 500 performance.

Here's the below query:

```
WITH AnnualSP500 AS (
 SELECT
 EXTRACT(YEAR FROM TRADE_DATE) AS YEAR,
 AVG(CLOSE) AS AVG_CLOSE
 FROM
 DW169.SP500_EOD_FACTS
 GROUP BY
 EXTRACT(YEAR FROM TRADE_DATE)
),
AnnualGDP AS (
 SELECT
 EXTRACT(YEAR FROM GDP_DATE) AS YEAR,
 AVG(GDP_VALUE) AS AVG_GDP
 FROM
 DW169.FRED_GDP
 GROUP BY
 EXTRACT(YEAR FROM GDP_DATE)
SELECT
 sp.YEAR,
 sp.AVG_CLOSE AS SP500_AVG_CLOSE,
```

```
gd.AVG_GDP AS GDP_AVG,

sp.AVG_CLOSE - LAG(sp.AVG_CLOSE) OVER (ORDER BY sp.YEAR) AS SP500_YOY_CHANGE,

gd.AVG_GDP - LAG(gd.AVG_GDP) OVER (ORDER BY gd.YEAR) AS GDP_YOY_CHANGE

FROM

AnnualSP500 sp

JOIN

AnnualGDP gd ON sp.YEAR = gd.YEAR

ORDER BY

sp.YEAR;
```

1	YEAR		GDP_AVG	\$P500_YOY_CHANGE	
1	1950	18.39726907630522088353413654618473895582	293.725	(null)	(null)
2	1951	22.32188755020080321285140562248995983936	339.25	3.92461847389558232931726907630522088354	45.525
3	1952	24.49616	358.3	2.17427244979919678714859437751004016064	19.05
1	1953	24.72258964143426294820717131474103585657	379.35	0.22642964143426294820717131474103585657	21.05
5	1954	29.7240873015873015873015873015873	380.35	5.00149766015303863909441598684626573073	1
5	1955	40.49884920634920634920634920634921	414.725	10.77476190476190476190476190476190476191	34.375
7	1956	46.63952191235059760956175298804780876494	437.45	6.14067270600139126035540378169860241573	22.725
3	1957	44.42337301587301587301587301587301587302	461.075	-2.21614889647758173654587997217479289192	23.625
9	1958	46.20345238095238095238095238095238	467.15	1.78007936507936507936507936507936507936	6.075
)	1959	57.41818181818181818181818181818181818182	506.625	11.21472943722943722943722943722943722944	39.475
L	1960	55.84575396825396825396825396825396	526.475	-1.57242784992784992784992784992784992785	19.85
2	1961	66.26632	544.775	10.42056603174603174603174603174603174603	18.3
3	1962	62.32075396825396825396825396825397	585.65	-3.94556603174603174603174603174603174603	40.875
4	1963	69.85936254980079681274900398406374501992	617.775	7.53860858154682855878075001580977676595	32.125

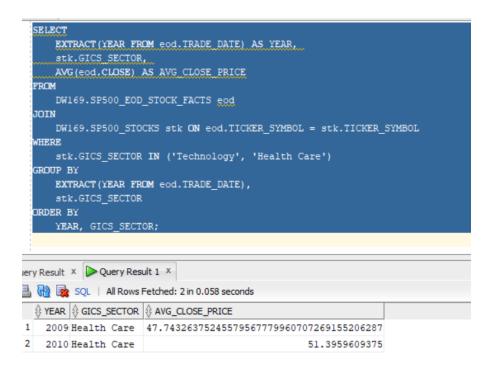
Annual Aggregation: This approach simplifies the analysis by focusing on the annual average closing price of the S&P 500 and the annual GDP, avoiding the complexities of quarterly data.

Year-over-Year Change: By using the LAG function, this query calculates the year-over-year change in both the S&P 500 average closing prices and GDP values. It allows for a simplified comparison of how changes in the stock market might relate to changes in the broader economy without detailed correlation analysis.

Additional Interesting Queries

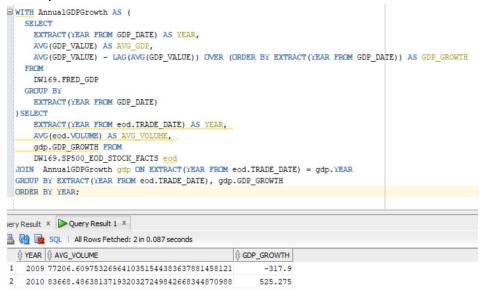
Query 1: Sector Performance Comparison

Description: Compare the annual average performance of two distinct sectors within the S&P 500 to understand how different sectors react to economic conditions.



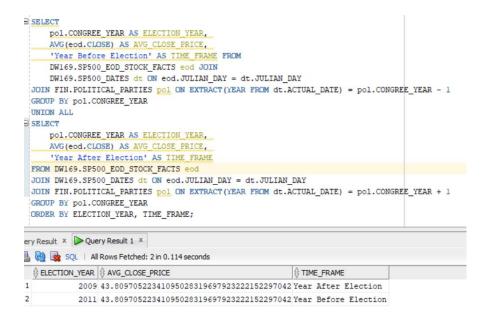
Query 2: GDP Growth vs. Stock Market Volume

Description: Analyze the relationship between GDP growth and stock market trading volume on an annual basis to see if higher economic growth correlates with increased trading activity.



Query 3: Identifying Top Performing Stocks

Description: Identify the top 5 performing stocks in terms of yearly price increase within the technology sector.



Query 4: Analysing Stock Market Volatility by Month

Description: This query calculates the monthly volatility of the S&P 500 by measuring the standard deviation of daily closing prices. High volatility may indicate uncertain market conditions or investor sentiment.

