OPUS - SQL Queries

AWS MySQL Database Connection Credentials:

username: guest
password: w3AkpA55word
host: cis550-project.cpmaujgv1bce.us-east-2.rds.amazonaws.com
port: 3306

1. Company Info Page (Dummy symbol *AAPL* – To be supplied by JavaScript via embedded SQL):

```
SELECT *
FROM CompanyInformation
WHERE symbol = 'AAPL';
```

The query will retrieve all the required information for each company like (but not limited to) Company Name, Sector, Industry, CEO Name, Number of Employees etc. which are contained within the CompanyInformation table. This information will be displayed on the **Company Page**. This will enable the user to get a broad overview of the company in terms of its size and the work it does.

2. Company Info Page, suggest peers:

```
SELECT CI.symbol, CI.CompanyName
FROM Peers P
JOIN CompanyInformation CI ON P.peerID = CI.symbol
WHERE P.symbol = 'AAPL';
```

This query will return the list of peers of the company with their symbols and names to be displayed on the **Company Page**. This will provide the user with a set of companies who do similar work. This feature will enable them to get to know about more companies which are likely to be engaged in work matching their interests.

3. Job Search Filter Temporary Table (Dummy entries - *$Dummy* – To be supplied by JavaScript via embedded SQL):

```
CREATE TEMPORARY TABLE TT (
SELECT *
FROM CompanyInformation CI
JOIN Indeed I ON I.company = CI.symbol
WHERE CI.industry LIKE '$Dummy' AND CI.sector LIKE '$Dummy' AND
CI.companyName LIKE '$Dummy' AND I.jobType LIKE '$Dummy' AND
(I.rating BETWEEN $DummyVAL1 AND $DummyVAL2)
ORDER BY I.DaysPosted
```

)

This execution of this query will create a temporary table containing the information of jobs fetched from Indeed corelated with the company information. This is achieved through joining the CompanyInformation table with IndeedJobs table on the trading symbol of the company. This intermediate table will be used by the subsequent queries for displaying information on the **Job Search Page.**

4. Job search :

```
SELECT * FROM TT;
```

This query will return information on all the jobs matching the user specified filters, which is stored in the temporary table TT. This information will be accessed by the user from the **Job Search Page**.

5. Job Stats by industry and sector:

```
WITH CMP_AVG AS (
SELECT symbol, MAX(price) as price, MAX(volAvg) as volAvg, MAX(mktCap) as mktCap, MAX(industry) as industry, MAX(sector) as sector, MAX(fullTimeEmployees) as fullTimeEmployees, COUNT(JobLink) as num_jobs, MAX(CompanyRating) AS CompanyRating
FROM TT
GROUP BY symbol),
SENT_CMP AS (SELECT price, fullTimeEmployees, sentiment, absoluteIndex, num_jobs, sector, industry FROM CMP_AVG C LEFT JOIN SENTIMENT S ON C.Symbol=S.Symbol)
SELECT AVG(price) as avg_price, MAX(fullTimeEmployees) as max_FTE, MIN(fullTimeEmployees) AS min_FTE, AVG(fullTimeEmployees) as avg_FTE, MIN(sentiment) as min_sentiment, MAX(sentiment) as max_sentiment, MAX(absoluteIndex) as max_abs_index, MIN(absoluteIndex) as min_abs_index, AVG(absoluteIndex) as avg_abs_index, AVG(sentiment) as avg_sentiment, SUM(num_jobs) as total_jobs, COUNT(DISTINCT sector) as num_sectors, COUNT(DISTINCT industry) as num_industries
FROM SENT_CMP;
```

This query will calculate average statistics of companies and jobs on the basis of the search results obtained after placing the user filters. The statistics calculated for each search include (but not limited to) average/minimum/maximum sentiment of all the companies in the search result, average/minimum/maximum absoluteIndex (popularity on social media) of the companies, number of sectors & industries the results are spread across, average/minimum/maximum rating of the companies in the search result on Indeed etc. These statistics will be show in the form of creative visualizations to the end user along with the Indeed job application links and relevant information on the **Job Search Page.**

6. News for a Company

```
WITH tmp1 as (SELECT symbol
FROM CompanyInformation
WHERE companyName LIKE 'value')
SELECT *
FROM CompanyNews c
JOIN tmp1
ON c.symbol=tmp1.symbol
ORDER BY publishedDate DESC;
```

Gets the news for a company with name like ‘value’. This query will be used on the **News Page**.

7. News for Peers of the Company

```
With tmp1 AS
(SELECT symbol
FROM CompanyInformation
WHERE companyName LIKE 'value'),
tmp2 AS
(SELECT peerID
FROM Peers
JOIN tmp1
ON Peers.symbol=tmp1.symbol)
SELECT *
FROM CompanyNews cn
JOIN tmp2
ON tmp2.peerID=cn.symbol
ORDER BY publishedDate DESC;
```

Gets the news for the peers of a company with name like ‘value’. This query will be used on the **News Page**.

8. Company Search on Home Page

```
WITH tmp1 AS
(SELECT symbol, companyName
FROM CompanyInformation
WHERE companyName LIKE '%value%' AND
fullTimeEmployees BETWEEN ‘$Dummylow’ AND ‘$Dummyhigh’
AND mktCap BETWEEN ‘$Dummylow’ AND ‘$Dummyhigh’),
tmp2 AS
(SELECT s.symbol, tmp1.companyName
FROM CompanySentiments s
JOIN tmp1
ON tmp1.symbol= s.symbol
```

```
WHERE sentiment BETWEEN '$Dummylow' AND '$Dummyhigh')
SELECT *, COUNT (jobLink)
FROM IndeedJobs i
JOIN tmp2
ON tmp2.symbol=i.companySymbol
GROUP BY i.companySymbol
HAVING COUNT (jobLink)> '$Dummy';
```

The query will display companies according to the search result obtained after placing the filters provided by the users like: Name of company, Range of MarketCap, Sentiment, No of full time employees and the minimum number of jobs that should be available for that company. This query will be utilized on the **Home Page**.
(Dummy values to be provided by JavaScript via embedded SQL)

9. Sentiment for a company

```
WITH T1 AS (SELECT name, ID
FROM Companies
WHERE name LIKE 'value')
SELECT T.name, S.sentiment
FROM Sentiment S JOIN T1 T ON S.companyID = T.ID;
```

Gets sentiment for a company with name like 'value'. For use on the **Company Information** page. This will allow the user to see how a company is perceived on social media.

10. Sentiment for a company's peers

```
WITH T1 AS (SELECT peerID as ID
FROM Peers
WHERE companyID = 'value'),
T2 AS (SELECT C.name, C.symbol, T1.ID
FROM Companies C JOIN T1 ON C.Id = T1.ID),
T3 AS (SELECT S.sentiment, T1.ID
FROM Sentiment S JOIN T1 ON S.companyID = T1.ID)
SELECT C.name, C.symbol, S.sentiment
FROM T2 JOIN T3 ON T2.ID = T3.ID;
```

Get sentiment and peers of a company given its ID as 'value'. For use on the **Sentiment page.** This will allow the user to compare the general perception of a company to its peers directly

11. Average sentiment for a company's peers

```
WITH T1 AS (SELECT peerID as ID
FROM Peers
WHERE companyID = 'value')
```

```
SELECT AVG(S.sentiment), T1.ID
FROM Sentiment S JOIN T1 ON S.companyID = T1.ID;
```

Get average sentiment for peers of a company given its ID as ‘value’. For use on the **Sentiment page.** This will allow the user to compare the general perception of a company to its peers more generally than the above

12. Average sentiment for companies in the same industry/sector

```
WITH T1 AS (SELECT Id AS ID
FROM CompanyInformation
WHERE [sector/industry] LIKE ‘value’)
SELECT AVG(S.sentiment)
FROM Sentiment S JOIN T1 ON T1.ID = S.companyID;
```

Get average sentiment for companies given an industry as ‘value’. A similar query can be made on sector instead of industry. For use in the **Sentiment** page. This will allow the user to determine the general perception of companies in the same industry/sector as the one they’re interested in

13. Average sentiment across all companies in every industry

```
SELECT CI.[industry/sector], AVG(S.sentiment)
FROM CompanyInformation CI JOIN Sentiment S ON CI.Id = S.companyID
GROUP BY CI.[industry/sector]
ORDER BY AVG(S.sentiment) DESC;
```

Get average sentiment across each industry, in descending order. Can be done for sector as well. For use on the **Sentiment page.** This will allow the user to see which industry/sector has the best general perception