```
218     # 15. Partition Apple data by year and rank daily volumes within each year (6 marks)
219     # SAHIL:
220
221 •   SELECT
222     YEAR(DATE) AS YEAR , VOLUME,
223     RANK() OVER(PARTITION BY YEAR(DATE) ORDER BY VOLUME DESC)
224     AS VOLUME_RANK
225     FROM APPLE;
226
227
228
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| YEAR | VOLUME | VOLUME_RANK |
|---|---|---|
| 1980 | 469033600 | 1 |
| 1980 | 175884800 | 2 |
| 1980 | 105728000 | 3 |
| 1980 | 93161600 | 4 |
| 1980 | 86441600 | 5 |
| 1980 | 73449600 | 6 |
| 1980 | 68880000 | 7 |

```
42
43      # sahil:
44
45      # 4. Count the number of trading days in the Google dataset:
46
47  •   select count(distinct(date)) as total_trading_days
48      from google;
49
50
51
52
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| | total_trading_days |
|---|---|
| ▶ | 4041 |

```
307     # SAHIL:
308     # 18  Identify the single trading date where the difference between Google and Apple closing prices was maximum;
309     # return date, both closes, and the difference (9 marks)
310
311 •   SELECT A.DATE,A.CLOSE , G.CLOSE ,
312     ABS(A.CLOSE-G.CLOSE) AS PRICE_DIFFERENCE
313     FROM APPLE A
314     INNER JOIN GOOGLE G
315     ON A.DATE = G.DATE
316     ORDER BY PRICE_DIFFERENCE DESC
317     LIMIT 1;
318
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| DATE | CLOSE | CLOSE | PRICE_DIFFERENCE |
|---|---|---|---|
| 2020-09-01 | 134.179993 | 1655.079956 | 1520.899963 |

Toggle wrapping of cell contents

```
291     # 17. Compute and compare the average daily return for Apple and Google over the entire period; indicate which performed bette
292 •   WITH APPLE_RETURN AS
293     (SELECT DATE, (CLOSE - LAG(CLOSE,1,0) OVER(ORDER BY DATE)) / LAG(CLOSE,1,0) OVER(ORDER BY DATE)
294     AS DAILY_RETURN FROM APPLE),
295     GOOGLE_RETURN AS(
296     SELECT DATE, (CLOSE-LAG(CLOSE,1,0) OVER (ORDER BY DATE)) / LAG(CLOSE,1,0) OVER(ORDER BY DATE )
297     AS DAILY_RETURN FROM GOOGLE)
298     SELECT ROUND(AVG(A.DAILY_RETURN)*100,4) AS APPLE_AVG_RETURN_PERCENT,
299     ROUND(AVG(G.DAILY_RETURN)*100,4) AS GOOGLE_AVG_RETURN_PERECENT,
300     CASE
301     WHEN AVG(A.DAILY_RETURN) > AVG(G.DAILY_RETURN) THEN 'APPLE'
302     WHEN AVG(G.DAILY_RETURN)> AVG(A.DAILY_RETURN) THEN 'GOOGLE'
303     ELSE 'EQUAL' END AS BETTER_PERFORMER
304     FROM APPLE_RETURN A
305     JOIN GOOGLE_RETURN G ON A.DATE = G.DATE;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| APPLE_AVG_RETURN_PERCENT | GOOGLE_AVG_RETURN_PERECENT | BETTER_PERFORMER |
|---|---|---|
| 0.1591 | 0.113 | APPLE |

```
229     # 16. Compute the cumulative sum of Apple's Volume ordered by Date (SAHIL):
230
231 ●   SELECT
232     DATE , VOLUME,
233     SUM(VOLUME) OVER(ORDER BY DATE) AS CUMLATIVE_SUM_OF_VOLUME
234     FROM APPLE;
235
236
237
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| DATE | VOLUME | CUMLATIVE_SUM_OF_VOLUME |
|---|---|---|
| 1980-12-12 | 469033600 | 469033600 |
| 1980-12-15 | 175884800 | 644918400 |
| 1980-12-16 | 105728000 | 750646400 |
| 1980-12-17 | 86441600 | 837088000 |
| 1980-12-18 | 73449600 | 910537600 |
| 1980-12-19 | 48630400 | 959168000 |
| 1980-12-22 | 37363200 | 996531200 |
| 1980-12-23 | 46950400 | 1043481600 |
| 1980-12-24 | 48003200 | 1091484800 |
| 1980-12-26 | 55574400 | 1147059200 |
| 1980-12-29 | 93161600 | 1240220800 |
| 1980-12-30 | 68880000 | 1309100800 |
| 1980-12-31 | 35750400 | 1344851200 |

```
202
203     # SAHIL
204     # 14 For Google, calculate a 7-day rolling average of Close using a window function (6 marks)
205
206 •   SELECT DATE , CLOSE ,
207     ROUND(AVG(CLOSE) OVER (ORDER BY DATE ROWS BETWEEN 6 PRECEDING AND CURRENT ROW),2)
208     AS ROLLING_AVERAGE_7D
209     FROM GOOGLE;
210
211
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| DATE | CLOSE | ROLLING_AVERAGE_7D |
|---|---|---|
| 2004-08-19 | 50.220219 | 50.22 |
| 2004-08-20 | 54.209209 | 52.21 |
| 2004-08-23 | 54.754753 | 53.06 |
| 2004-08-24 | 52.487488 | 52.92 |
| 2004-08-25 | 53.053055 | 52.94 |
| 2004-08-26 | 54.00901 | 53.12 |
| 2004-08-27 | 53.128128 | 53.12 |
| 2004-08-30 | 51.056057 | 53.24 |
| 2004-08-31 | 51.236237 | 52.82 |
| 2004-09-01 | 50.175175 | 52.16 |
| 2004-09-02 | 50.805805 | 51.92 |
| 2004-09-03 | 50.055054 | 51.5 |

```
179     # SAHIL:
180     # 13. With a CTE, compute monthly average close for Apple and order descending by average close
181
182 •   WITH MONTHLY_AVERAGE AS
183     (SELECT
184     DATE_FORMAT(DATE,'%Y-%m') AS MONTH,
185     ROUND(AVG(CLOSE),2) AS AVG_CLOSE
186     FROM APPLE
187     GROUP BY DATE_FORMAT(DATE,'%Y-%m')
188     )
189     SELECT MONTH,
190     AVG_CLOSE FROM MONTHLY_AVERAGE
191     ORDER BY AVG_CLOSE DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| MONTH | AVG_CLOSE |
|---|---|
| 2020-09 | 134.18 |
| 2020-08 | 117.3 |
| 2020-07 | 95.57 |
| 2020-06 | 86.45 |
| 2020-01 | 77.98 |
| 2020-02 | 77.82 |
| 2020-05 | 77.5 |

```
155     # 12. Create a CTE that labels each Google trading day as HighVol or LowVol and select counts of each label (SAHIL)
156
157     WITH GOOGLE_LABELED AS (
158     SELECT DATE , VOLUME,
159     CASE
160     WHEN VOLUME > (SELECT AVG(VOLUME) FROM GOOGLE) THEN 'HIGHVOLUME'
161     ELSE 'LOWVOLUME'
162     END AS VOLUME_LABEL
163     FROM GOOGLE)
164
165     SELECT VOLUME_LABEL ,COUNT(VOLUME_LABEL) AS COUNTS
166     FROM GOOGLE_LABELED
167     GROUP BY VOLUME_LABEL;
168
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| VOLUME_LABEL | COUNTS |
|---|---|
| HIGHVOLUME | 1289 |
| LOWVOLUME | 2752 |

```sql
# 11. Use a CTE to calculate the daily return for Apple and select the top 5 dates with the highest returns
# SAHIL:
WITH APPLE_RETURN AS (
SELECT DATE , CLOSE , LAG(CLOSE) OVER(ORDER BY DATE) ,
ROUND(((CLOSE - LAG(CLOSE) OVER(ORDER BY DATE)) / LAG(CLOSE) OVER(ORDER BY DATE))*100,2) AS DAILY_RETURN
FROM APPLE)
SELECT DATE , CLOSE
DAILY_RETURN FROM APPLE_RETURN
WHERE DAILY_RETURN IS NOT NULL
ORDER BY DAILY_RETURN DESC
LIMIT 5;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| DATE | DAILY_RETURN |
|---|---|
| 2020-09-01 | 134.179993 |
| 2020-08-31 | 129.039993 |
| 2020-08-26 | 126.522499 |
| 2020-08-24 | 125.857498 |
| 2020-08-27 | 125.010002 |

```
115     # 10 Which company had the higher closing price on each date? Return Date, Close_Higher, and Winner:
116
117 •   SELECT A.DATE AS DATE,
118     CASE
119     WHEN A.CLOSE > G.CLOSE THEN A.CLOSE
120     ELSE G.CLOSE
121     END AS CLOSE_HIGHER,
122     CASE
123     WHEN A.CLOSE>G.CLOSE THEN 'APPLE'
124     ELSE 'GOOGLE'
125     END AS WINNER
126     FROM APPLE A
127     INNER JOIN GOOGLE G
128     ON A.DATE = G.DATE;
129
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| DATE | CLOSE_HIGHER | WINNER |
|---|---|---|
| 2004-08-19 | 50.220219 | GOOGLE |
| 2004-08-20 | 54.209209 | GOOGLE |
| 2004-08-23 | 54.754753 | GOOGLE |

```
102
103     # Sahil:
104     # 9  Using a left join, list all Apple trade dates and corresponding Google Close values:
105
106 •   SELECT A.DATE AS APPLE_TRADE_DATE ,
107     A.CLOSE AS APPLE_CLOSE_VALUE,
108     G.CLOSE AS GOOGLE_CLOSE_VALUE
109     FROM APPLE A
110     LEFT JOIN GOOGLE G
111     ON A.DATE = G.DATE;
112
113
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| APPLE_TRADE_DATE | APPLE_CLOSE_VALUE | GOOGLE_CLOSE_VALUE |
|---|---|---|
| 1980-12-12 | 0.128348 | NULL |
| 1980-12-15 | 0.121652 | NULL |
| 1980-12-16 | 0.112723 | NULL |
| 1980-12-17 | 0.115513 | NULL |
| 1980-12-18 | 0.118862 | NULL |

Result 46

```
90
91      # Sahil:
92
93      # 8.Perform an inner join on the two tables by Date and return Date, apple_close, google_close :
94
95      SELECT A.DATE,
96      A.CLOSE AS APPLE_CLOSE,
97      G.CLOSE AS GOOGLE_CLOSE
98      FROM APPLE A
99      INNER JOIN GOOGLE G
100     ON A.DATE = G.DATE;
101
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| DATE | APPLE_CLOSE | GOOGLE_CLOSE |
|---|---|---|
| 2004-08-19 | 0.548393 | 50.220219 |
| 2004-08-20 | 0.55 | 54.209209 |
| 2004-08-23 | 0.555 | 54.754753 |
| 2004-08-24 | 0.570536 | 52.487488 |
| 2004-08-25 | 0.590179 | 53.053055 |

```
80      # Sahil
81
82      # 7. For Apple, retrieve the date and Close when Close equals the maximum Close across all Google data (5 marks)
83
84      SELECT DATE , CLOSE
85      FROM APPLE
86      WHERE CLOSE=
87      (SELECT MAX(CLOSE)
88      FROM GOOGLE);
89
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| | DATE | CLOSE |
|---|---|---|

```
65
66      # Sahil:
67
68      # 6. Find the Google trade date(s) where the trading volume exceeded the average Apple volume (5 marks)
69
70      SELECT DATE FROM GOOGLE
71      WHERE VOLUME>
72      (SELECT AVG(VOLUME)
73       FROM APPLE);
74
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| | DATE |
|---|---|

```
52
53      # Sahil:
54
55      # 5. List all dates on which Apple's closing price was greater than Google's average closing price:
56
57      SELECT DATE FROM APPLE
58      WHERE CLOSE>
59      (SELECT AVG(CLOSE)
60      FROM GOOGLE);
61
62
```

```
34
35      # Sahil:
36
37      # 3. Identify the highest and lowest daily High price for Apple:
38
39 •    select max(high) as highest_daily_high_price ,
40      min(high) as lowest_daily_high_price from apple;
41
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| highest_daily_high_price | lowest_daily_high_price |
|---|---|
| 134.800003 | 0.049665 |

```
24
25      # Sahil:
26
27      # 2. Compute the average closing price for Google stock:
28
29 •    select round(avg(close)) as average_closing_price from google;
30
31
32
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| total_trading_volume |
| --- |
| 3321524160000 |

```
10
11
12      # Sahil:
13      # 1. Find the total trading volume for Apple stock:
14
15 •    select sum(volume) as total_trading_volume from apple;
16
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| total_trading_volume |
| --- |
| 3321524160000 |