DS203: Exercise 9

Nirav Bhattad

Last updated November 8, 2024

Task 1

Task: 1

Do a manual review of the table **nsedata** and describe its contents

The table `nsedata` contains daily stock market data for various companies. Each row in the table represents a stock identified by its symbol, detailing the performance during a specific trading session. The columns in the table are as follows:

Column Name	**Data Type**	**Description**
symbol	varchar(50)	Unique identifier for the stock/company (e.g., AAPL).
series	varchar(50)	Indicates the type of equity, all entries marked as EQ (common equity).
open	decimal(20,6)	The price at the start of the trading session.
high	decimal(20,6)	The highest price the stock reached during the day.
low	decimal(20,6)	The lowest price recorded during the day.
close	decimal(20,6)	The price at the end of the trading session.
last	decimal(20,6)	The last traded price, which may differ from the closing price.
prevclose	decimal(20,6)	The closing price from the previous trading session.
tottrdqty	int	The total traded quantity of the stock.
tottrdval	decimal(20,6)	The total traded value of the stock.
timestamp	varchar(50)	The timestamp of when the data was recorded.
anum	mediumint	An additional numeric field that may be used for internal purposes (nullable).
isin	varchar(50)	The International Securities Identification Number (nullable).
extra	varchar(50)	Any extra information related to the stock (nullable).

Constraints:

- All columns, except `anum`, `isin`, and `extra`, are required to have non-null values.
- None of the columns are marked as keys.
- No default values have been set for any column.

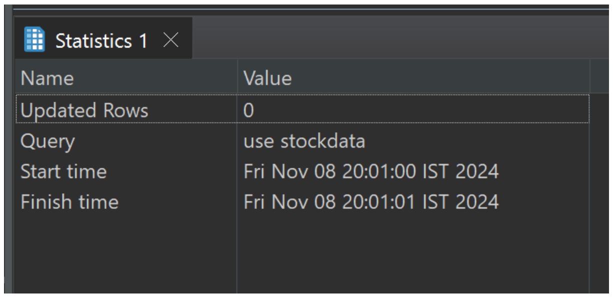
Task 2

Task: 2

Select the database **stockdata** using SQL

```
USE stockdata;
```

Listing 1: Selecting the Database



```
Name            Value
Updated Rows    0
Query           use stockdata
Start time      Fri Nov 08 20:01:00 IST 2024
Finish time     Fri Nov 08 20:01:01 IST 2024
```

Figure 1: Output of Task 2

Task 3

Task: 3

Get a schema dump of the table nsedata using SQL

```
SHOW CREATE TABLE nsedata;
```

Listing 2: Schema Dump of Table nsedata

```
CREATE TABLE `nsedata` (
  `symbol` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `series` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `open` decimal(20,6) NOT NULL,
  `high` decimal(20,6) NOT NULL,
  `low` decimal(20,6) NOT NULL,
  `close` decimal(20,6) NOT NULL,
  `last` decimal(20,6) NOT NULL,
  `prevclose` decimal(20,6) NOT NULL,
  `tottrdqty` int NOT NULL,
  `tottrdval` decimal(20,6) NOT NULL,
  `timestamp` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `anum` mediumint DEFAULT NULL,
  `isin` varchar(50) DEFAULT NULL,
  `extra` varchar(50) DEFAULT NULL
)
ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci COMMENT='Table containing stock
    data obtained from NSE'
```

Listing 3: Schema Dump of Table nsedata

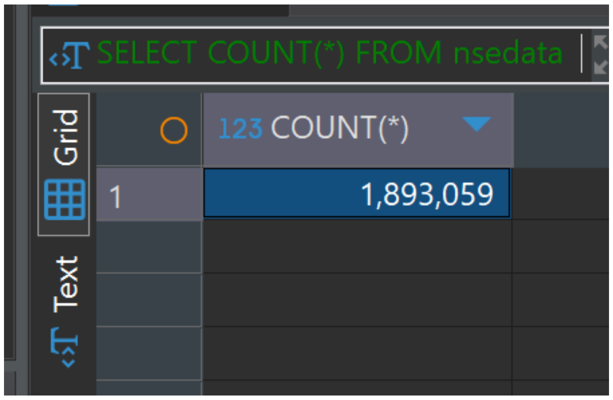
Task 4

Task: 4

Get a count of the total number of records in nsedata

```sql
SELECT COUNT(*) FROM nsedata;
```

Listing 4: Counting the Total Number of Records



```
SELECT COUNT(*) FROM nsedata
   COUNT(*)
1  1,893,059
```

Figure 2: Output of Task 4

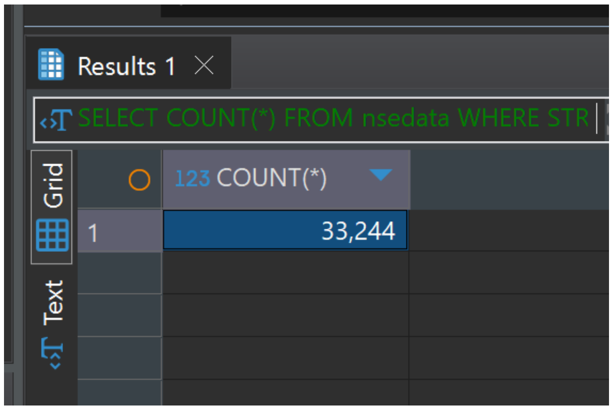
Task 5

Task: 5

Get the total count of the records for the month "October 2012"

```sql
SELECT COUNT(*)
FROM nsedata
WHERE STR_TO_DATE(timestamp,'%d-%b-%Y')
BETWEEN '2012-10-01'
AND '2012-10-31';
```

Listing 5: Counting the Total Number of Records for October 2012



```
SELECT COUNT(*) FROM nsedata WHERE STR
   COUNT(*)
1  33,244
```

Figure 3: Output of Task 5

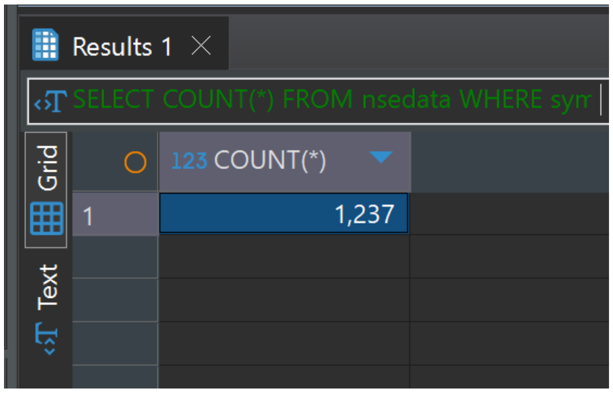
Task 6

Task: 6

Repeat '4', but only for the stock with symbol "GEOMETRIC"

```sql
SELECT COUNT(*)
FROM nsedata
WHERE symbol = 'GEOMETRIC';
```

Listing 6: Counting the Total Number of Records for GEOMETRIC



```
SELECT COUNT(*) FROM nsedata WHERE sym
```

	COUNT(*)
1	1,237

Figure 4: Output of Task 6

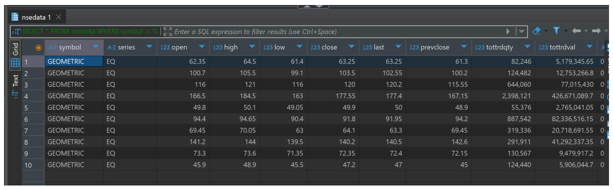
Task 7

Task: 7

Repeat '6', but only display the first 10 records

```sql
SELECT *
FROM nsedata
WHERE symbol = 'GEOMETRIC'
LIMIT 10;
```

Listing 7: Displaying the First 10 Records for GEOMETRIC



```
SELECT * FROM nsedata WHERE symbol = 'G
Enter a SQL expression to filter results (use Ctrl+Space)
```

	symbol	series	open	high	low	close	last	prevclose	tottrdqty	tottrdval	
1	GEOMETRIC	EQ	62.35	64.5	61.4	63.25	63.25	61.3	82,246	5,179,345.65	0
2	GEOMETRIC	EQ	100.7	105.5	99.1	103.5	102.55	100.2	124,482	12,753,266.8	0
3	GEOMETRIC	EQ	116	121	116	120	120.2	115.55	644,060	77,015,430	0
4	GEOMETRIC	EQ	166.5	184.5	163	177.55	177.4	167.15	2,398,121	426,671,089.7	0
5	GEOMETRIC	EQ	49.8	50.1	49.05	49.9	50	48.9	55,376	2,765,041.05	0
6	GEOMETRIC	EQ	94.4	94.65	90.4	91.8	91.95	94.2	887,542	82,336,516.15	0
7	GEOMETRIC	EQ	69.45	70.05	63	64.1	63.3	69.45	319,336	20,718,691.55	0
8	GEOMETRIC	EQ	141.2	144	139.5	140.2	140.5	142.6	291,911	41,292,337.35	0
9	GEOMETRIC	EQ	73.3	73.6	71.35	72.35	72.4	72.15	130,567	9,479,917.2	0
10	GEOMETRIC	EQ	45.9	48.9	45.5	47.2	47	45	124,440	5,906,044.7	0

Figure 5: Output of Task 7

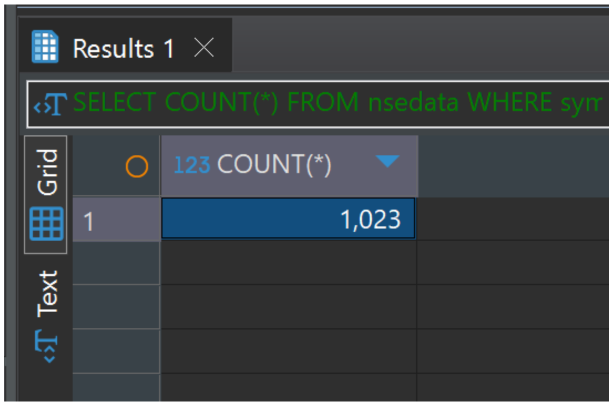
Task 8

Task: 8

Totally, how many records of "INFY" does the table contain?

```sql
SELECT COUNT(*)
FROM nsedata
WHERE symbol = 'INFY';
```

Listing 8: Counting the Total Number of Records for INFY



```
SELECT COUNT(*) FROM nsedata WHERE sym
```

	COUNT(*)
1	1,023

Figure 6: Output of Task 8

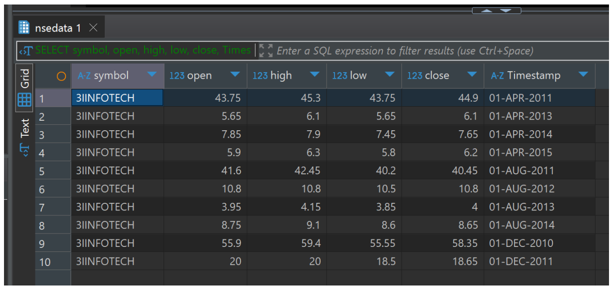
Task 9

Task: 9

Get a listing of the first 10 records of "3IINFOTECH", but the listing should contain only the following columns: symbol, open, high, low, close, and timestamp

```sql
SELECT symbol, open, high, low, close,
Timestamp
FROM nsedata
WHERE symbol = '3IINFOTECH'
LIMIT 10;
```

Listing 9: Displaying the First 10 Records for 3IINFOTECH



```
SELECT symbol, open, high, low, close, Times
```

	symbol	open	high	low	close	Timestamp
1	3IINFOTECH	43.75	45.3	43.75	44.9	01-APR-2011
2	3IINFOTECH	5.65	6.1	5.65	6.1	01-APR-2013
3	3IINFOTECH	7.85	7.9	7.45	7.65	01-APR-2014
4	3IINFOTECH	5.9	6.3	5.8	6.2	01-APR-2015
5	3IINFOTECH	41.6	42.45	40.2	40.45	01-AUG-2011
6	3IINFOTECH	10.8	10.8	10.5	10.8	01-AUG-2012
7	3IINFOTECH	3.95	4.15	3.85	4	01-AUG-2013
8	3IINFOTECH	8.75	9.1	8.6	8.65	01-AUG-2014
9	3IINFOTECH	55.9	59.4	55.55	58.35	01-DEC-2010
10	3IINFOTECH	20	20	18.5	18.65	01-DEC-2011

Figure 7: Output of Task 9

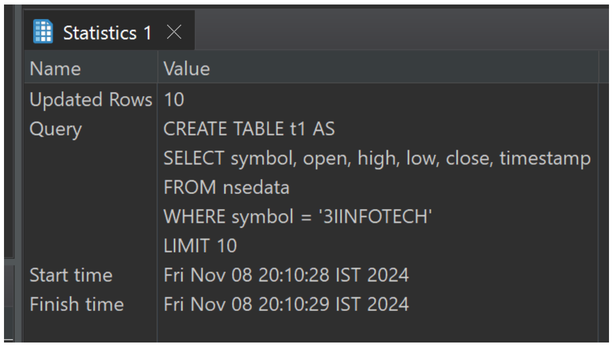
Task 10

Task: 10

Repeat '9', but this time use the results to create a table **t1** in the **stockdata** database

```sql
CREATE TABLE t1 AS
SELECT symbol, open, high, low, close, timestamp
FROM nsedata
WHERE symbol = '3IINFOTECH'
LIMIT 10;
SHOW TABLES;
```

Listing 10: Creating Table t1

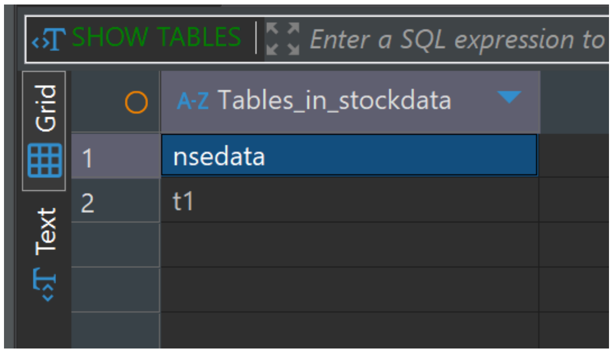


Name	Value
Updated Rows	10

Query	CREATE TABLE t1 AS
	SELECT symbol, open, high, low, close, timestamp
	FROM nsedata
	WHERE symbol = '3IINFOTECH'
	LIMIT 10

Start time	Fri Nov 08 20:10:28 IST 2024
Finish time	Fri Nov 08 20:10:29 IST 2024

Figure 8: Output of Task 10



	Tables_in_stockdata
1	nsedata
2	t1

Figure 9: Output of Task 10

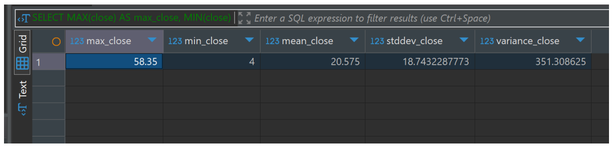
Task 11

Task: 11

Using **t1** find out the following for the column **close**: max, min, mean. standard deviation and variance

```sql
SELECT MAX(close) AS max_close, MIN(close) AS min_close, AVG(close) AS mean_close, STDDEV(close) AS
     stddev_close, VARIANCE(close) AS variance_close
FROM t1;
```

Listing 11: Finding Statistics for the close Column



SELECT MAX(close) AS max_close, MIN(close)

	max_close	min_close	mean_close	stddev_close	variance_close
1	58.35	4	20.575	18.7432287773	351.308625

Figure 10: Output of Task 11

Task 12

Task: 12

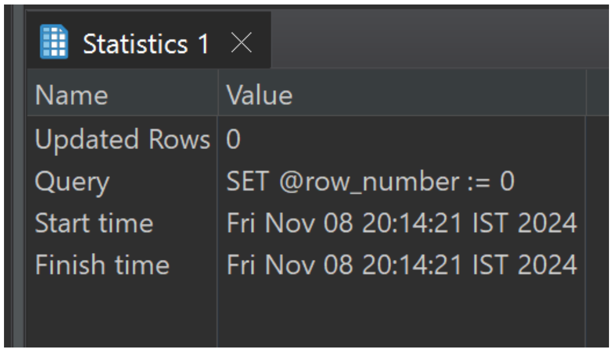
How will you find out the value of the median, if that is also required?

```sql
SET @row_number := 0;

SELECT AVG(close) AS median_close
FROM (
  SELECT close,
     @row_number := @row_number + 1 AS row_num
  FROM t1
  ORDER BY close
) AS sorted_data

WHERE row_num IN (
  FLOOR((@row_number + 1) / 2),
  CEIL((@row_number + 1) / 2)
);
```

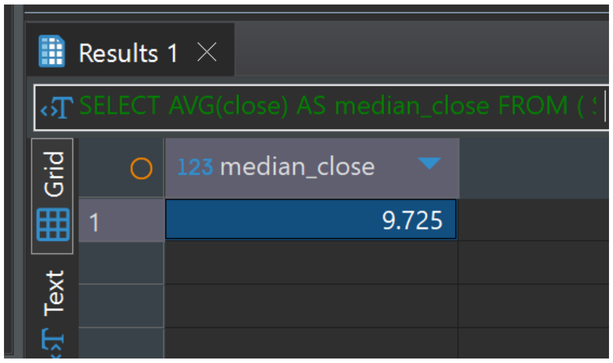
Listing 12: Finding the Median for the close Column



Statistics 1

Name	Value
Updated Rows	0
Query	SET @row_number := 0
Start time	Fri Nov 08 20:14:21 IST 2024
Finish time	Fri Nov 08 20:14:21 IST 2024

Figure 11: Output of Task 12



```
Results 1
SELECT AVG(close) AS median_close FROM (
   median_close
1  9.725
```

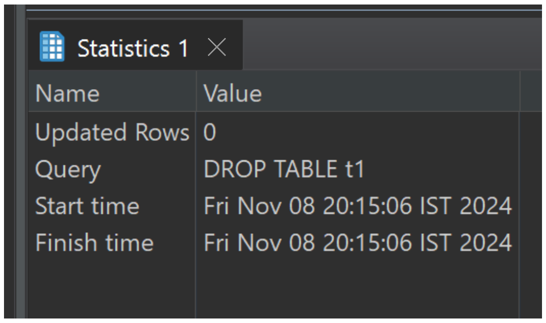
Figure 12: Output of Task 12

Task 13

Task: 13
Delete table **t1**

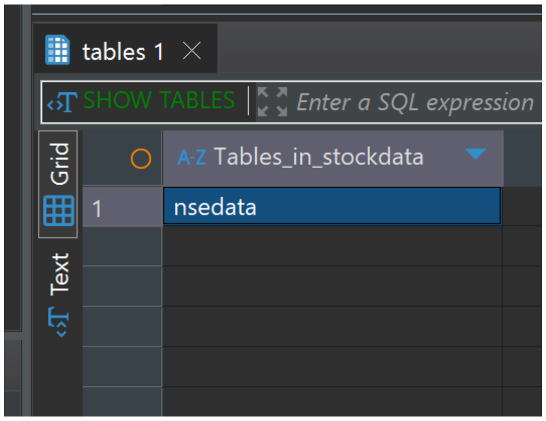
```sql
DROP TABLE t1;
SHOW TABLES;
```

Listing 13: Deleting Table t1



```
Statistics 1
Name          Value
Updated Rows  0
Query         DROP TABLE t1
Start time    Fri Nov 08 20:15:06 IST 2024
Finish time   Fri Nov 08 20:15:06 IST 2024
```

Figure 13: Output of Task 13



```
tables 1
SHOW TABLES
   Tables_in_stockdata
1  nsedata
```

Figure 14: Output of Task 13

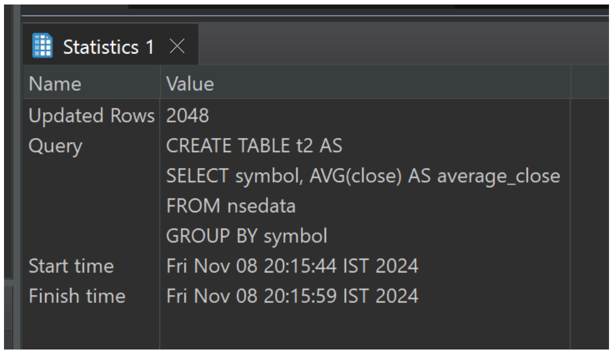
Task 14

Task: 14

Switch back to using nsedata. Using the GROUP BY functionality of SQL create a table **t2** containing the average value of **close** for every symbol in the table. Hint: the table will have the columns: **symbol**, **average**

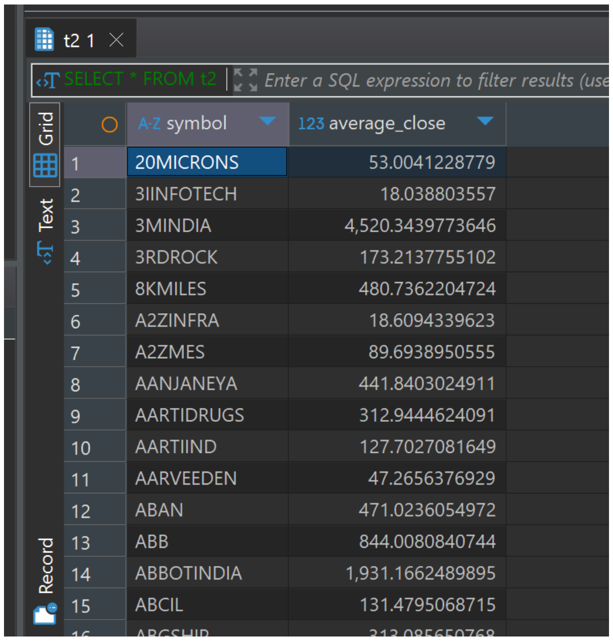
```sql
CREATE TABLE t2 AS
SELECT symbol , AVG(close) AS average_close
FROM nsedata
GROUP BY symbol;
SELECT * FROM t2;
```

Listing 14: Creating Table t2



```
Statistics 1
Name          Value
Updated Rows  2048
Query         CREATE TABLE t2 AS
              SELECT symbol, AVG(close) AS average_close
              FROM nsedata
              GROUP BY symbol
Start time    Fri Nov 08 20:15:44 IST 2024
Finish time   Fri Nov 08 20:15:59 IST 2024
```

Figure 15: Output of Task 14



```
     symbol        average_close
1    20MICRONS     53.0041228779
2    3IINFOTECH    18.038803557
3    3MINDIA       4,520.3439773646
4    3RDROCK       173.2137755102
5    8KMILES       480.7362204724
6    A2ZINFRA      18.6094339623
7    A2ZMES        89.6938950555
8    AANJANEYA     441.8403024911
9    AARTIDRUGS    312.9444624091
10   AARTIIND      127.7027081649
11   AARVEEDEN     47.2656376929
12   ABAN          471.0236054972
13   ABB           844.0080840744
14   ABBOTINDIA    1,931.1662489895
15   ABCIL         131.4795068715
```

Figure 16: Output of Task 14

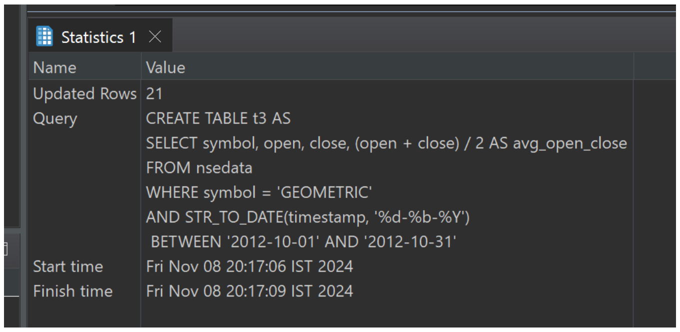
Task 15

Task: 15

Create a table **t3** such that it contains the following columns: symbol, open, close, "average of open and close". Fill up this table for the company GEOMETRIC, for the month of October 2012

```sql
CREATE TABLE t3 AS
SELECT symbol , open , close , (open + close) / 2 AS avg_open_close
FROM nsedata
WHERE symbol = 'GEOMETRIC'
AND STR_TO_DATE(timestamp , '%d-%b-%Y')
BETWEEN '2012-10-01' AND '2012-10-31';
SELECT * FROM t3;
```

Listing 15: Creating Table t3

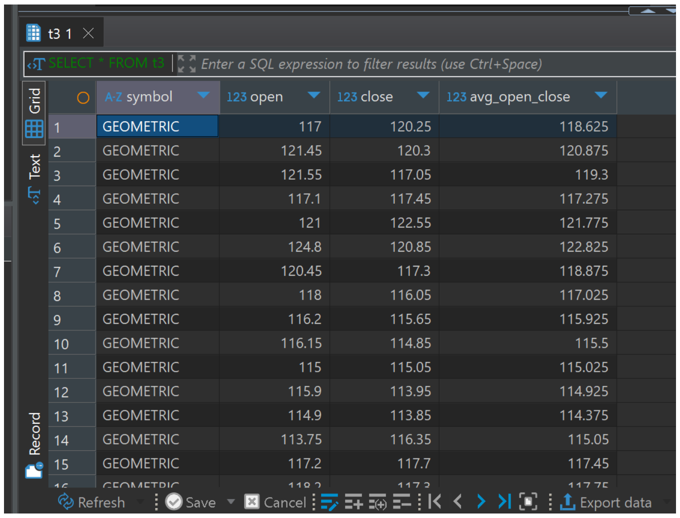


Name	Value
Updated Rows	21

Query	CREATE TABLE t3 AS
	SELECT symbol, open, close, (open + close) / 2 AS avg_open_close
	FROM nsedata
	WHERE symbol = 'GEOMETRIC'
	AND STR_TO_DATE(timestamp, '%d-%b-%Y')
	BETWEEN '2012-10-01' AND '2012-10-31'

Start time	Fri Nov 08 20:17:06 IST 2024
Finish time	Fri Nov 08 20:17:09 IST 2024

Figure 17: Output of Task 15



SELECT * FROM t3

	symbol	open	close	avg_open_close
1	GEOMETRIC	117	120.25	118.625
2	GEOMETRIC	121.45	120.3	120.875
3	GEOMETRIC	121.55	117.05	119.3
4	GEOMETRIC	117.1	117.45	117.275
5	GEOMETRIC	121	122.55	121.775
6	GEOMETRIC	124.8	120.85	122.825
7	GEOMETRIC	120.45	117.3	118.875
8	GEOMETRIC	118	116.05	117.025
9	GEOMETRIC	116.2	115.65	115.925
10	GEOMETRIC	116.15	114.85	115.5
11	GEOMETRIC	115	115.05	115.025
12	GEOMETRIC	115.9	113.95	114.925
13	GEOMETRIC	114.9	113.85	114.375
14	GEOMETRIC	113.75	116.35	115.05
15	GEOMETRIC	117.2	117.7	117.45
16	GEOMETRIC	118.2	117.3	117.75

Figure 18: Output of Task 15

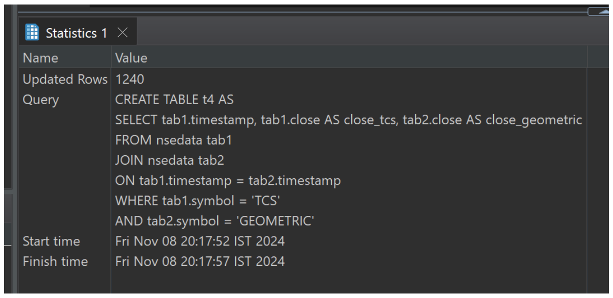
Task 16

Task: 16

It is required to create a table **t4** such that it contains the data for two companies **GEOMETRIC** and **TCS**. The columns of this table should be as follows: timestamp, close_tcs, close_geometric.
Hint: use JOIN

```sql
CREATE TABLE t4 AS
SELECT tab1.timestamp, tab1.close AS close_tcs, tab2.close AS close_geometric
FROM nsedata tab1
JOIN nsedata tab2
ON tab1.timestamp = tab2.timestamp
WHERE tab1.symbol = 'TCS'
AND tab2.symbol = 'GEOMETRIC';
SELECT * FROM t4;
```

Listing 16: Creating Table t4

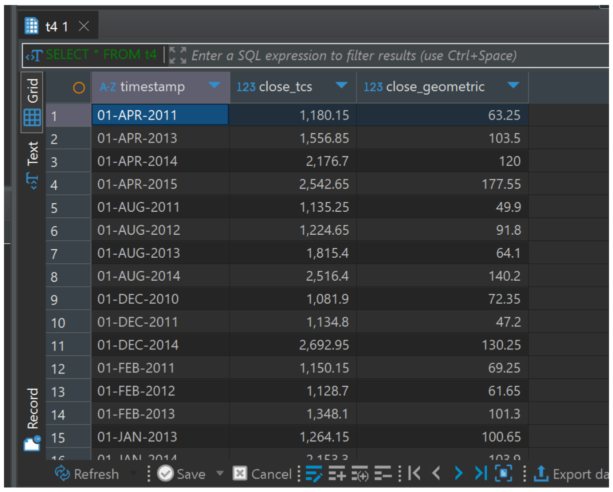


Name	Value
Updated Rows	1240

Query	CREATE TABLE t4 AS
	SELECT tab1.timestamp, tab1.close AS close_tcs, tab2.close AS close_geometric
	FROM nsedata tab1
	JOIN nsedata tab2
	ON tab1.timestamp = tab2.timestamp
	WHERE tab1.symbol = 'TCS'
	AND tab2.symbol = 'GEOMETRIC'

Start time	Fri Nov 08 20:17:52 IST 2024
Finish time	Fri Nov 08 20:17:57 IST 2024

Figure 19: Output of Task 16



	timestamp	close_tcs	close_geometric
1	01-APR-2011	1,180.15	63.25
2	01-APR-2013	1,556.85	103.5
3	01-APR-2014	2,176.7	120
4	01-APR-2015	2,542.65	177.55
5	01-AUG-2011	1,135.25	49.9
6	01-AUG-2012	1,224.65	91.8
7	01-AUG-2013	1,815.4	64.1
8	01-AUG-2014	2,516.4	140.2
9	01-DEC-2010	1,081.9	72.35
10	01-DEC-2011	1,134.8	47.2
11	01-DEC-2014	2,692.95	130.25
12	01-FEB-2011	1,150.15	69.25
13	01-FEB-2012	1,128.7	61.65
14	01-FEB-2013	1,348.1	101.3
15	01-JAN-2013	1,264.15	100.65
16	01-JAN-2014	2,153.3	103.9

Figure 20: Output of Task 16

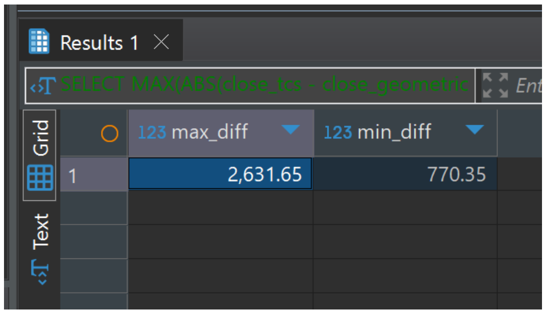
Task 17

Task: 17

Find out the maximum and minimum difference in the daily closing prices of these two companies.

```sql
SELECT MAX(ABS(close_tcs - close_geometric)) AS max_diff, MIN(ABS(close_tcs - close_geometric)) AS
    min_diff
FROM t4;
```

Listing 17: Finding the Maximum and Minimum Difference in Closing Prices



SELECT MAX(ABS(close_tcs - close_geometric

	max_diff	min_diff
1	2,631.65	770.35

Figure 21: Output of Task 17

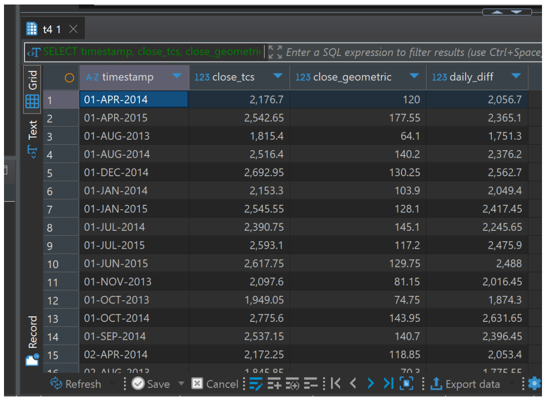
Task 18

Task: 18

Based on **t4** can you identify those days on which the difference in their closing price was more than the average of the minimum and maximum differences of their closing prices.

```sql
SELECT timestamp, close_tcs, close_geometric, ABS(close_tcs - close_geometric) AS daily_diff
FROM t4
WHERE ABS(close_tcs - close_geometric) > (SELECT (MAX(ABS(close_tcs - close_geometric)) + MIN(ABS(
    close_tcs - close_geometric))) / 2
FROM t4
);
```

Listing 18: Identifying Days with Closing Price Difference More than Average



SELECT timestamp, close_tcs, close_geometri

	timestamp	close_tcs	close_geometric	daily_diff
1	01-APR-2014	2,176.7	120	2,056.7
2	01-APR-2015	2,542.65	177.55	2,365.1
3	01-AUG-2013	1,815.4	64.1	1,751.3
4	01-AUG-2014	2,516.4	140.2	2,376.2
5	01-DEC-2014	2,692.95	130.25	2,562.7
6	01-JAN-2014	2,153.3	103.9	2,049.4
7	01-JAN-2015	2,545.55	128.1	2,417.45
8	01-JUL-2014	2,390.75	145.1	2,245.65
9	01-JUL-2015	2,593.1	117.2	2,475.9
10	01-JUN-2015	2,617.75	129.75	2,488
11	01-NOV-2013	2,097.6	81.15	2,016.45
12	01-OCT-2013	1,949.05	74.75	1,874.3
13	01-OCT-2014	2,775.6	143.95	2,631.65
14	01-SEP-2014	2,537.15	140.7	2,396.45
15	02-APR-2014	2,172.25	118.85	2,053.4

Figure 22: Output of Task 18

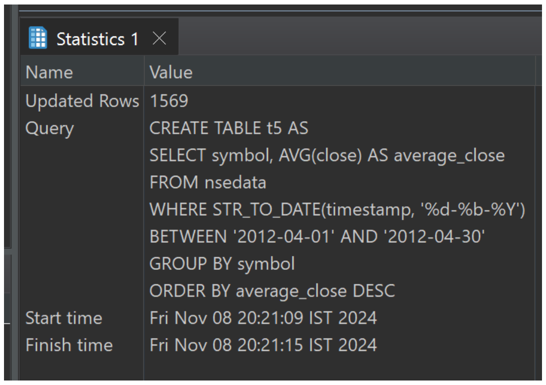
Task 19

Task: 19

Based on **nsedata**, create table **t5** such that it contains the average **close** price of each company traded in the month of April 2012. The table should be sorted in descending order of the average close price.

```sql
CREATE TABLE t5 AS
SELECT symbol, AVG(close) AS average_close
FROM nsedata
WHERE STR_TO_DATE(timestamp, '%d-%b-%Y')
BETWEEN '2012-04-01' AND '2012-04-30'
GROUP BY symbol
ORDER BY average_close DESC;
SELECT * FROM t5;
```

Listing 19: Creating Table t5



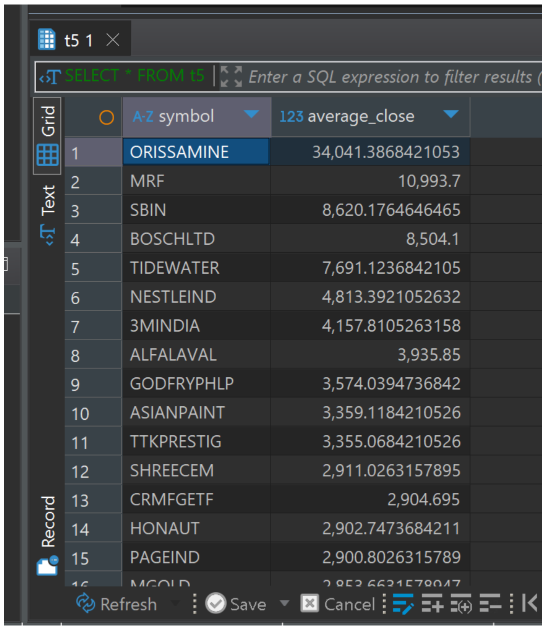
Statistics 1

Name	Value
Updated Rows	1569

Query	CREATE TABLE t5 AS
	SELECT symbol, AVG(close) AS average_close
	FROM nsedata
	WHERE STR_TO_DATE(timestamp, '%d-%b-%Y')
	BETWEEN '2012-04-01' AND '2012-04-30'
	GROUP BY symbol
	ORDER BY average_close DESC

Start time	Fri Nov 08 20:21:09 IST 2024
Finish time	Fri Nov 08 20:21:15 IST 2024

Figure 23: Output of Task 19



t5 1

SELECT * FROM t5

	symbol	average_close
1	ORISSAMINE	34,041.3868421053
2	MRF	10,993.7
3	SBIN	8,620.1764646465
4	BOSCHLTD	8,504.1
5	TIDEWATER	7,691.1236842105
6	NESTLEIND	4,813.3921052632
7	3MINDIA	4,157.8105263158
8	ALFALAVAL	3,935.85
9	GODFRYPHLP	3,574.0394736842
10	ASIANPAINT	3,359.1184210526
11	TTKPRESTIG	3,355.0684210526
12	SHREECEM	2,911.0263157895
13	CRMFGETF	2,904.695
14	HONAUT	2,902.7473684211
15	PAGEIND	2,900.8026315789
16	MGOLD	2,853.6631578947

Figure 24: Output of Task 19

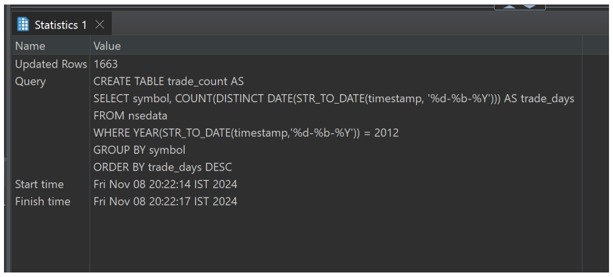
Task 20

Task: 20

Not all companies are traded every day. It is required to create a table that contains a count of the days each company has been traded in a selected year - say 2012. The table should be sorted in descending order of the count.

```sql
CREATE TABLE trade_count AS
SELECT symbol, COUNT(DISTINCT DATE(STR_TO_DATE(timestamp, '%d-%b-%Y'))) AS trade_days
FROM nsedata
WHERE YEAR(STR_TO_DATE(timestamp,'%d-%b-%Y')) = 2012
GROUP BY symbol
ORDER BY trade_days DESC;
SELECT * FROM trade_count;
```

Listing 20: Creating Table trade_count

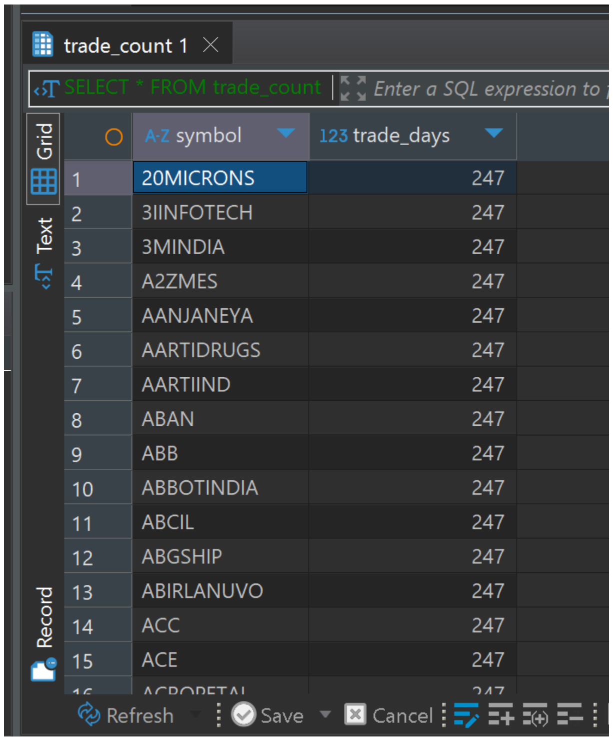


Name	Value
Updated Rows	1663

Query	CREATE TABLE trade_count AS
	SELECT symbol, COUNT(DISTINCT DATE(STR_TO_DATE(timestamp, '%d-%b-%Y'))) AS trade_days
	FROM nsedata
	WHERE YEAR(STR_TO_DATE(timestamp,'%d-%b-%Y')) = 2012
	GROUP BY symbol
	ORDER BY trade_days DESC

Start time	Fri Nov 08 20:22:14 IST 2024
Finish time	Fri Nov 08 20:22:17 IST 2024

Figure 25: Output of Task 20



	symbol	trade_days
1	20MICRONS	247
2	3IINFOTECH	247
3	3MINDIA	247
4	A2ZMES	247
5	AANJANEYA	247
6	AARTIDRUGS	247
7	AARTIIND	247
8	ABAN	247
9	ABB	247
10	ABBOTINDIA	247
11	ABCIL	247
12	ABGSHIP	247
13	ABIRLANUVO	247
14	ACC	247
15	ACE	247
16	ACROPETAL	247

Figure 26: Output of Task 20