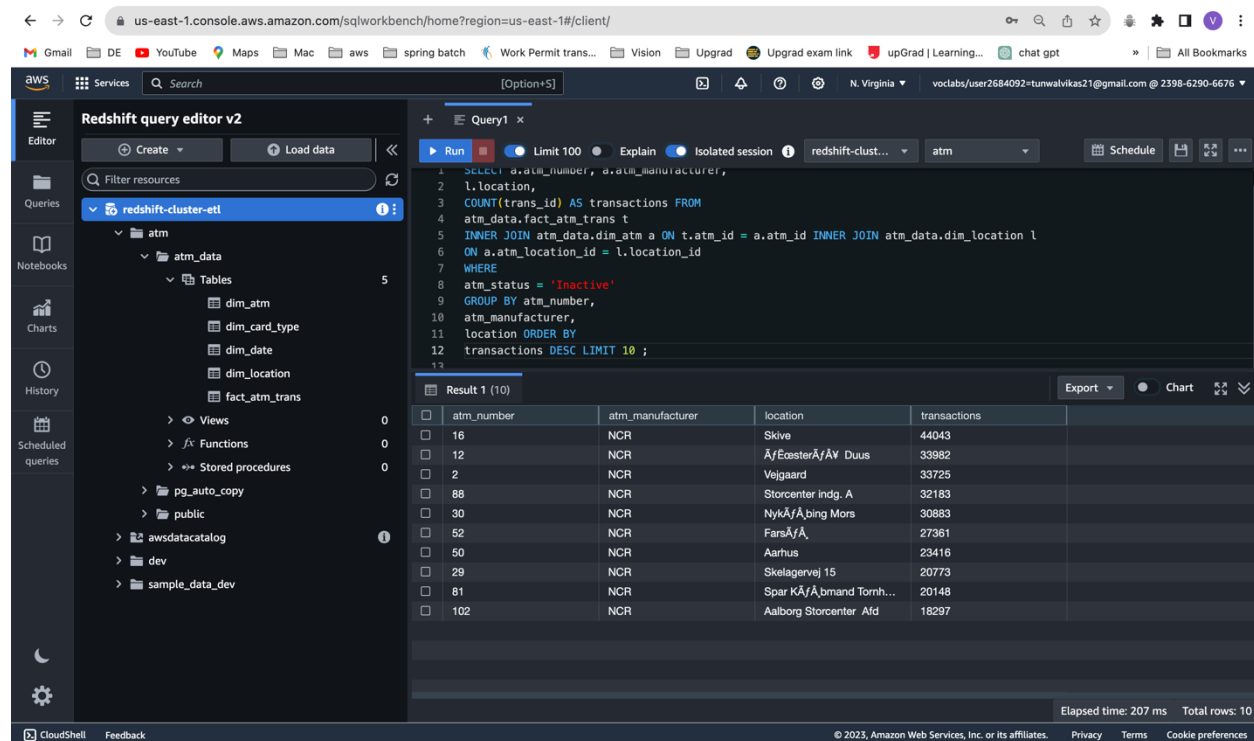


Solving analytical queries on Redshift Cluster

Here, you have to write the query used for solving the question and the screenshots of the table which is outputted after the query is run on the AWS Redshift Query editor UI.

1. Top 10 ATMs where most transactions are in the 'inactive' state

```
SELECT a.atm_number,
a.atm_manufacturer,
l.location,
COUNT(trans_id) AS transactions FROM
atm_data.fact_atm_trans t
INNER JOIN atm_data.dim_atm a ON t.atm_id = a.atm_id
INNER JOIN atm_data.dim_location l ON a.atm_location_id = l.location_id
WHERE atm_status = 'Inactive'
GROUP BY atm_number, atm_manufacturer, location
ORDER BY transactions DESC
LIMIT 10 ;
```



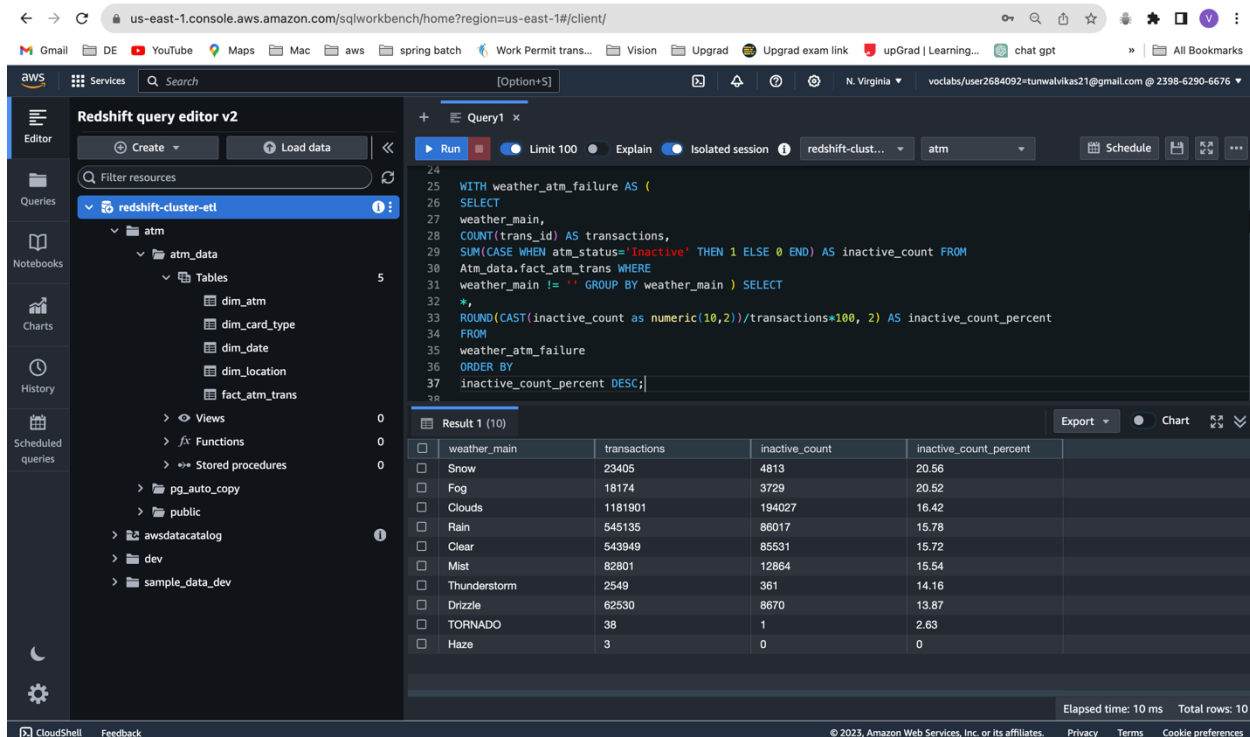
The screenshot shows the AWS Redshift Query Editor v2 interface. The query is executed, and the results are displayed in a table with 10 rows. The table has four columns: atm_number, atm_manufacturer, location, and transactions. The results are sorted by transactions in descending order.

atm_number	atm_manufacturer	location	transactions
16	NCR	Skive	44043
12	NCR	ÅrEosterÅ/ÅV Duus	33982
2	NCR	Vejgaard	33725
88	NCR	Storcenter indg. A	32183
30	NCR	NykÅ/Åbing Mors	30883
52	NCR	FarsÅ/Å	27361
50	NCR	Aarhus	23416
29	NCR	Skelagervej 15	20773
81	NCR	Spar KÅ/Åbmand Tornh...	20148
102	NCR	Aalborg Storcenter Åld	18297

Elapsed time: 207 ms Total rows: 10

2. Number of ATM failures corresponding to the different weather conditions recorded at the time of the transactions

```
WITH weather_atm_failure AS (
SELECT
weather_main,
COUNT(trans_id) AS transactions,
SUM(CASE WHEN atm_status='Inactive' THEN 1 ELSE 0 END) AS inactive_count
FROM
Atm_data.fact_atm_trans WHERE
weather_main != '' GROUP BY weather_main ) SELECT
*,
ROUND(CAST(inactive_count as numeric(10,2))/transactions*100, 2) AS
inactive_count_percent
FROM
weather_atm_failure
ORDER BY
inactive_count_percent DESC;
```



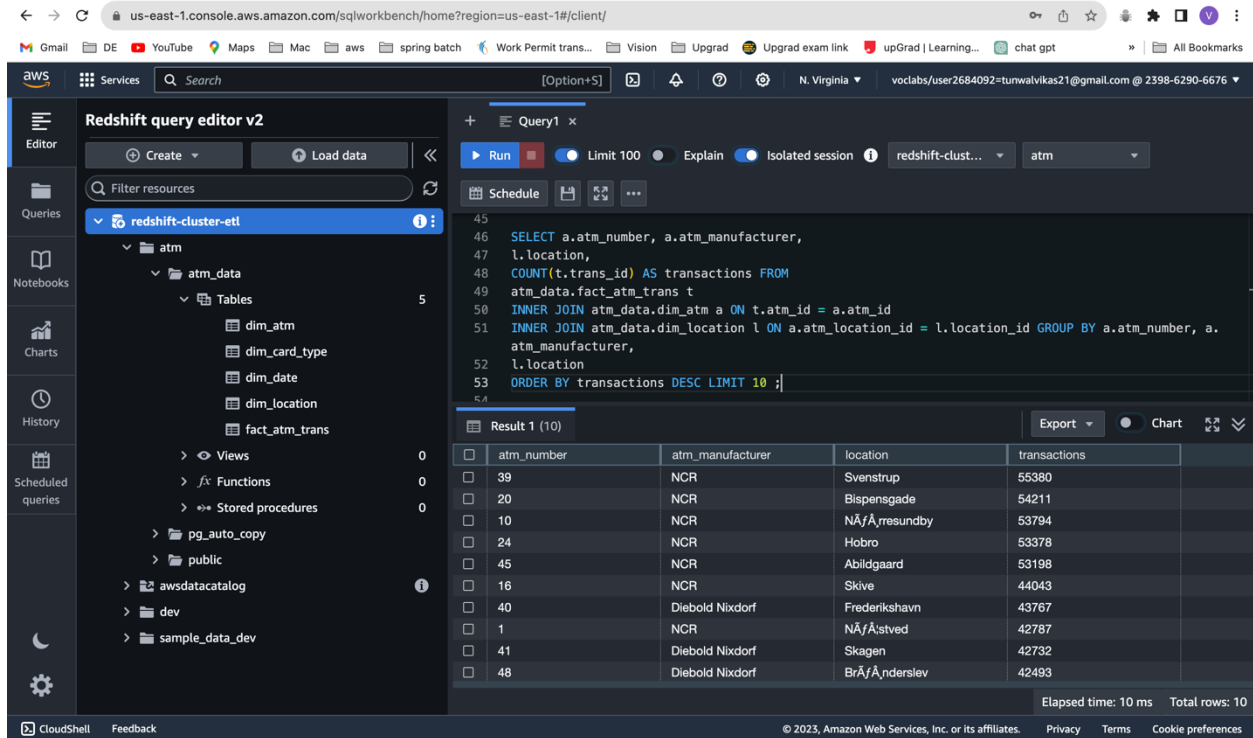
The screenshot shows the AWS Redshift query editor interface. The SQL query is displayed in the editor, and the results are shown in a table below. The table has four columns: weather_main, transactions, inactive_count, and inactive_count_percent. The results are sorted by inactive_count_percent in descending order.

weather_main	transactions	inactive_count	inactive_count_percent
Snow	23405	4813	20.56
Fog	18174	3729	20.52
Clouds	1181901	194027	16.42
Rain	545135	86017	15.78
Clear	543949	85531	15.72
Mist	82801	12864	15.54
Thunderstorm	2549	361	14.16
Drizzle	62530	8670	13.87
TORNADO	38	1	2.63
Haze	3	0	0

Elapsed time: 10 ms Total rows: 10

3. Top 10 ATMs with the most number of transactions throughout the year

```
SELECT a.atm_number,
a.atm_manufacturer,
l.location,
COUNT(t.trans_id) AS transactions
FROM
atm_data.fact_atm_trans t
INNER JOIN atm_data.dim_atm a ON t.atm_id = a.atm_id
INNER JOIN atm_data.dim_location l ON a.atm_location_id = l.location_id
GROUP BY a.atm_number, a.atm_manufacturer, l.location
ORDER BY transactions DESC LIMIT 10 ;
```



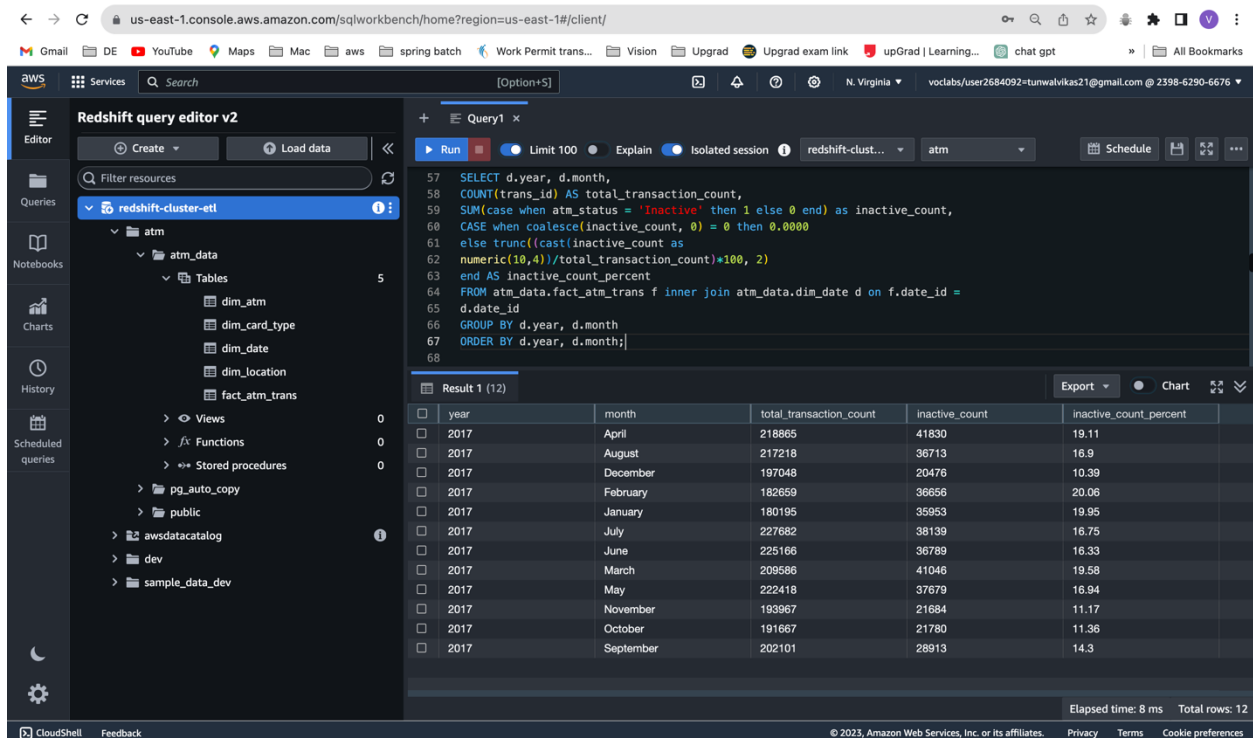
The screenshot shows the AWS Redshift Query Editor v2 interface. The SQL query is entered in the editor and has been executed. The results are displayed in a table with 10 rows, showing the top 10 ATMs by transaction count.

atm_number	atm_manufacturer	location	transactions
39	NCR	Svenstrup	55380
20	NCR	Bispensgade	54211
10	NCR	NÅfÅresundby	53794
24	NCR	Hobro	53378
45	NCR	Abildgaard	53198
16	NCR	Skive	44043
40	Diebold Nixdorf	Frederikshavn	43767
1	NCR	NÅfÅstved	42787
41	Diebold Nixdorf	Skagen	42732
48	Diebold Nixdorf	BrÅfÅnderslev	42493

Elapsed time: 10 ms Total rows: 10

4. Number of overall ATM transactions going inactive per month for each month

```
SELECT d.year, d.month,
COUNT(trans_id) AS total_transaction_count,
SUM(case when atm_status = 'Inactive' then 1 else 0 end) as
inactive_count,
CASE when coalesce(inactive_count, 0) = 0 then 0.0000
else trunc((cast(inactive_count as
numeric(10,4))/total_transaction_count)*100, 2)
end AS inactive_count_percent
FROM atm_data.fact_atm_trans f inner join atm_data.dim_date d on f.date_id =
d.date_id
GROUP BY d.year, d.month
ORDER BY d.year, d.month;
```



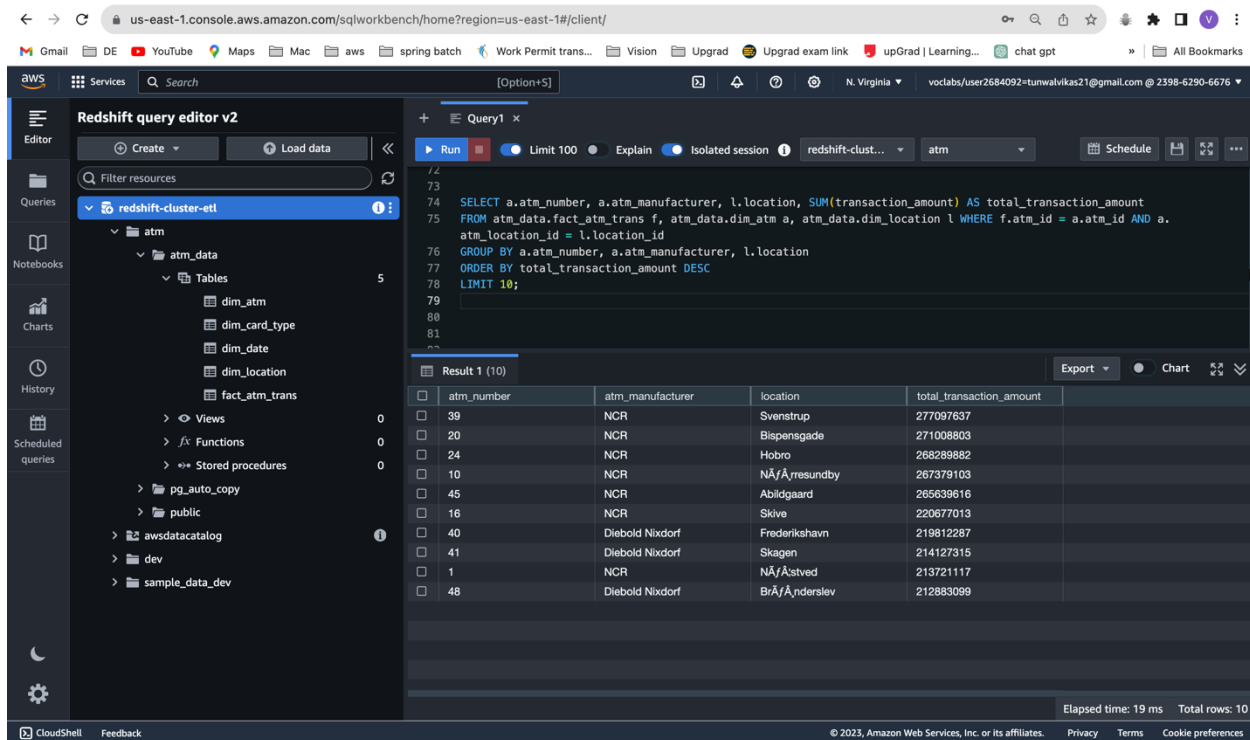
The screenshot shows the AWS Redshift Query Editor v2 interface. The SQL query is displayed in the editor, and the results are shown in a table below. The table has 6 columns: year, month, total_transaction_count, inactive_count, and inactive_count_percent. The results are grouped by year and month.

year	month	total_transaction_count	inactive_count	inactive_count_percent
2017	April	218865	41830	19.11
2017	August	217218	36713	16.9
2017	December	197048	20476	10.39
2017	February	182659	36656	20.06
2017	January	180195	35953	19.95
2017	July	227682	38139	16.75
2017	June	225166	36789	16.33
2017	March	209586	41046	19.58
2017	May	222418	37679	16.94
2017	November	193967	21684	11.17
2017	October	191667	21780	11.36
2017	September	202101	28913	14.3

Elapsed time: 8 ms Total rows: 12

5. Top 10 ATMs with the highest total withdrawn amount throughout the year

```
SELECT a.atm_number, a.atm_manufacturer, l.location,
SUM(transaction_amount) AS total_transaction_amount
FROM atm_data.fact_atm_trans f, atm_data.dim_atm a, atm_data.dim_location
l WHERE f.atm_id = a.atm_id AND a.atm_location_id = l.location_id
GROUP BY a.atm_number, a.atm_manufacturer, l.location
ORDER BY total_transaction_amount DESC
LIMIT 10;
```



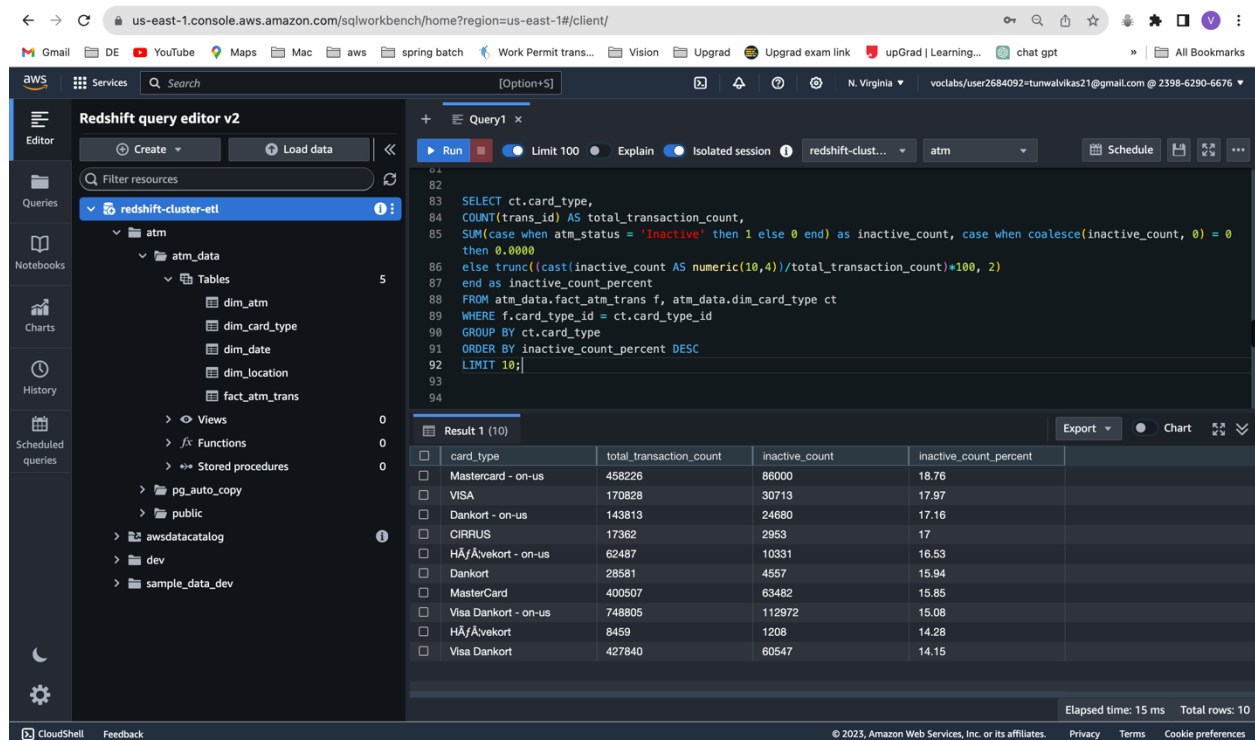
The screenshot shows the AWS Redshift query editor v2 interface. The query is executed, and the results are displayed in a table with 10 rows. The columns are atm_number, atm_manufacturer, location, and total_transaction_amount. The results are sorted by total_transaction_amount in descending order.

atm_number	atm_manufacturer	location	total_transaction_amount
39	NCR	Svenstrup	277097637
20	NCR	Bispensgade	271008803
24	NCR	Hobro	268289882
10	NCR	NÄ/Äresundby	267379103
45	NCR	Abildgaard	265639616
16	NCR	Skive	220677013
40	Diebold Nixdorf	Frederikshavn	219612287
41	Diebold Nixdorf	Skagen	214127315
1	NCR	NÄ/Ästved	213721117
48	Diebold Nixdorf	BrÄ/Änderslev	212883099

Elapsed time: 19 ms Total rows: 10

6. Number of failed ATM transactions across various card types

```
SELECT ct.card_type,
COUNT(trans_id) AS total_transaction_count,
SUM(case when atm_status = 'Inactive' then 1 else 0 end) as
inactive_count, case when coalesce(inactive_count, 0) = 0 then 0.0000
else trunc((cast(inactive_count AS
numeric(10,4))/total_transaction_count)*100, 2)
end as inactive_count_percent
FROM atm_data.fact_atm_trans f, atm_data.dim_card_type ct
WHERE f.card_type_id = ct.card_type_id
GROUP BY ct.card_type
ORDER BY inactive_count_percent DESC
LIMIT 10;
```



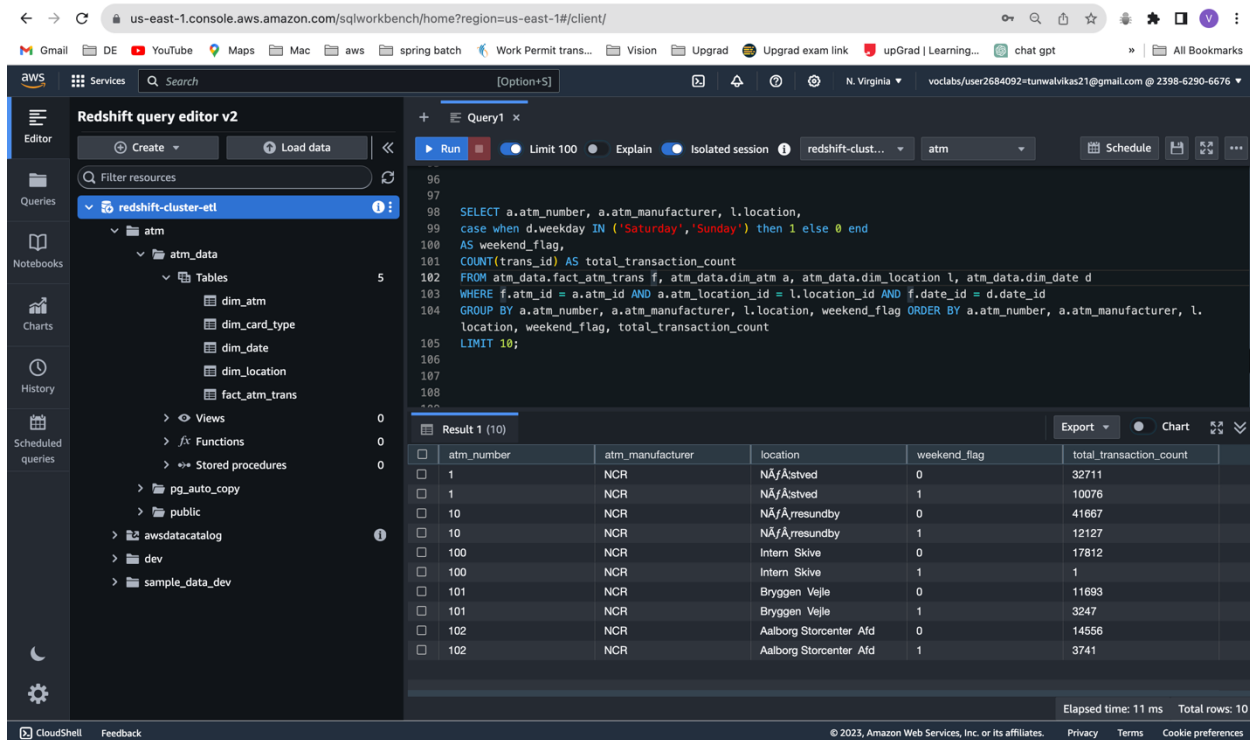
The screenshot shows the AWS Redshift Query Editor v2 interface. The SQL query is entered in the editor, and the results are displayed in a table below. The table has four columns: card_type, total_transaction_count, inactive_count, and inactive_count_percent. The results are sorted by inactive_count_percent in descending order.

card_type	total_transaction_count	inactive_count	inactive_count_percent
Mastercard - on-us	458226	86000	18.76
VISA	170828	30713	17.97
Dankort - on-us	143813	24680	17.16
CIRRUS	17362	2953	17
HÅfÅvekort - on-us	62487	10331	16.53
Dankort	28581	4557	15.94
MasterCard	400507	63482	15.85
Visa Dankort - on-us	748805	112972	15.08
HÅfÅvekort	8459	1208	14.28
Visa Dankort	427840	60547	14.15

Elapsed time: 15 ms Total rows: 10

7. Number of transactions happening on an ATM on weekdays and on weekends throughout the year. Order this by the ATM_number, ATM_manufacturer, location, weekend_flag and then total_transaction_count

```
SELECT a.atm_number, a.atm_manufacturer, l.location,
case when d.weekday IN ('Saturday','Sunday') then 1 else 0 end
AS weekend_flag,
COUNT(trans_id) AS total_transaction_count
FROM atm_data.fact_atm_trans f, atm_data.dim_atm a, atm_data.dim_location
l, atm_data.dim_date d
WHERE f.atm_id = a.atm_id AND a.atm_location_id = l.location_id AND
f.date_id = d.date_id
GROUP BY a.atm_number, a.atm_manufacturer, l.location, weekend_flag ORDER
BY a.atm_number, a.atm_manufacturer, l.location, weekend_flag,
total_transaction_count
LIMIT 10;
```



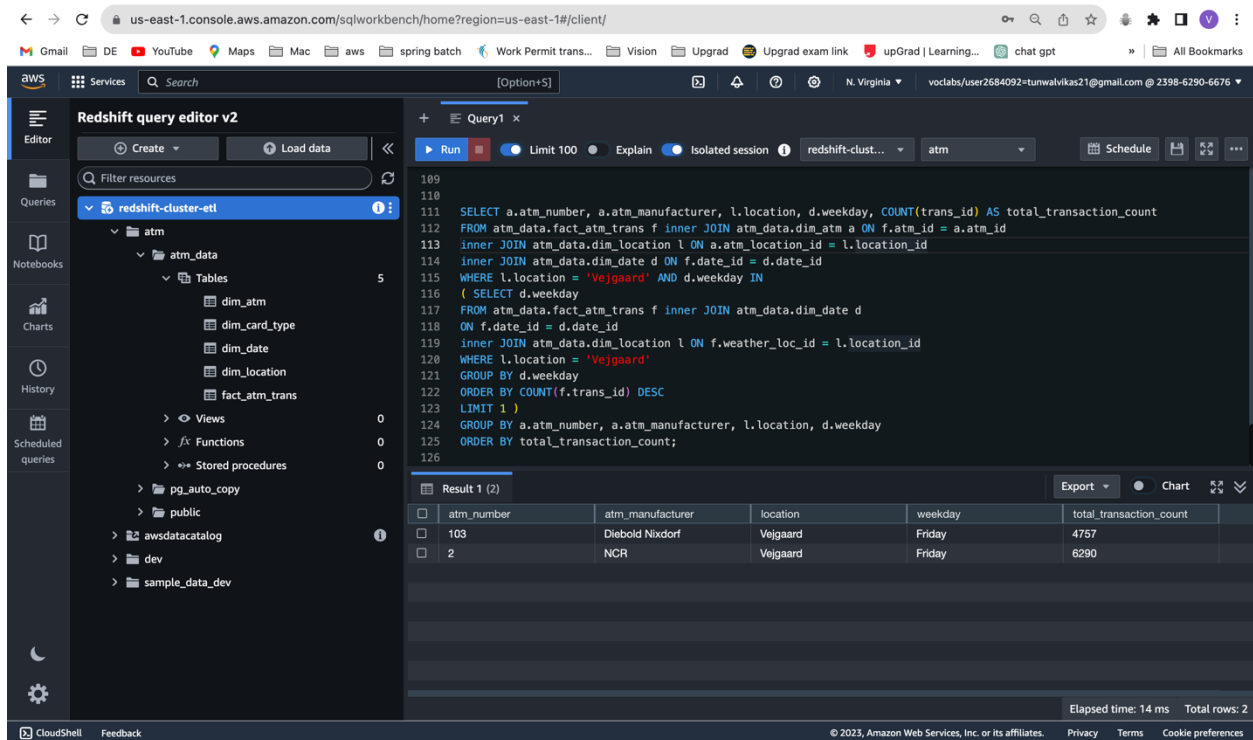
The screenshot shows the AWS Redshift query editor v2 interface. The SQL query is entered in the editor and has been executed. The results are displayed in a table with 5 columns: atm_number, atm_manufacturer, location, weekend_flag, and total_transaction_count. The results are ordered by atm_number, atm_manufacturer, location, weekend_flag, and total_transaction_count.

atm_number	atm_manufacturer	location	weekend_flag	total_transaction_count
1	NCR	NÅ/Åstved	0	32711
1	NCR	NÅ/Åstved	1	10076
10	NCR	NÅ/Åresundby	0	41667
10	NCR	NÅ/Åresundby	1	12127
100	NCR	Intern Skive	0	17812
100	NCR	Intern Skive	1	1
101	NCR	Bryggen Vejle	0	11693
101	NCR	Bryggen Vejle	1	3247
102	NCR	Aalborg Storcenter Afd	0	14556
102	NCR	Aalborg Storcenter Afd	1	3741

Elapsed time: 11 ms Total rows: 10

8. Most active day in each ATMs from location "Vejgaard"

```
SELECT a.atm_number, a.atm_manufacturer, l.location, d.weekday,
COUNT(trans_id) AS total_transaction_count
FROM atm_data.fact_atm_trans f inner JOIN atm_data.dim_atm a ON f.atm_id =
a.atm_id
inner JOIN atm_data.dim_location l ON a.atm_location_id = l.location_id
inner JOIN atm_data.dim_date d ON f.date_id = d.date_id
WHERE l.location = 'Vejgaard' AND d.weekday IN
( SELECT d.weekday
FROM atm_data.fact_atm_trans f inner JOIN atm_data.dim_date d
ON f.date_id = d.date_id
inner JOIN atm_data.dim_location l ON f.weather_loc_id = l.location_id
WHERE l.location = 'Vejgaard'
GROUP BY d.weekday
ORDER BY COUNT(f.trans_id) DESC
LIMIT 1 )
GROUP BY a.atm_number, a.atm_manufacturer, l.location, d.weekday
ORDER BY total_transaction_count;
```



The screenshot shows the AWS Redshift Query Editor v2 interface. The SQL query is pasted into the editor, and the results are displayed in a table below. The table has 5 columns: atm_number, atm_manufacturer, location, weekday, and total_transaction_count. The results show two rows of data for the location 'Vejgaard'.

atm_number	atm_manufacturer	location	weekday	total_transaction_count
103	Diebold Nixdorf	Vejgaard	Friday	4757
2	NCR	Vejgaard	Friday	6290

Elapsed time: 14 ms Total rows: 2