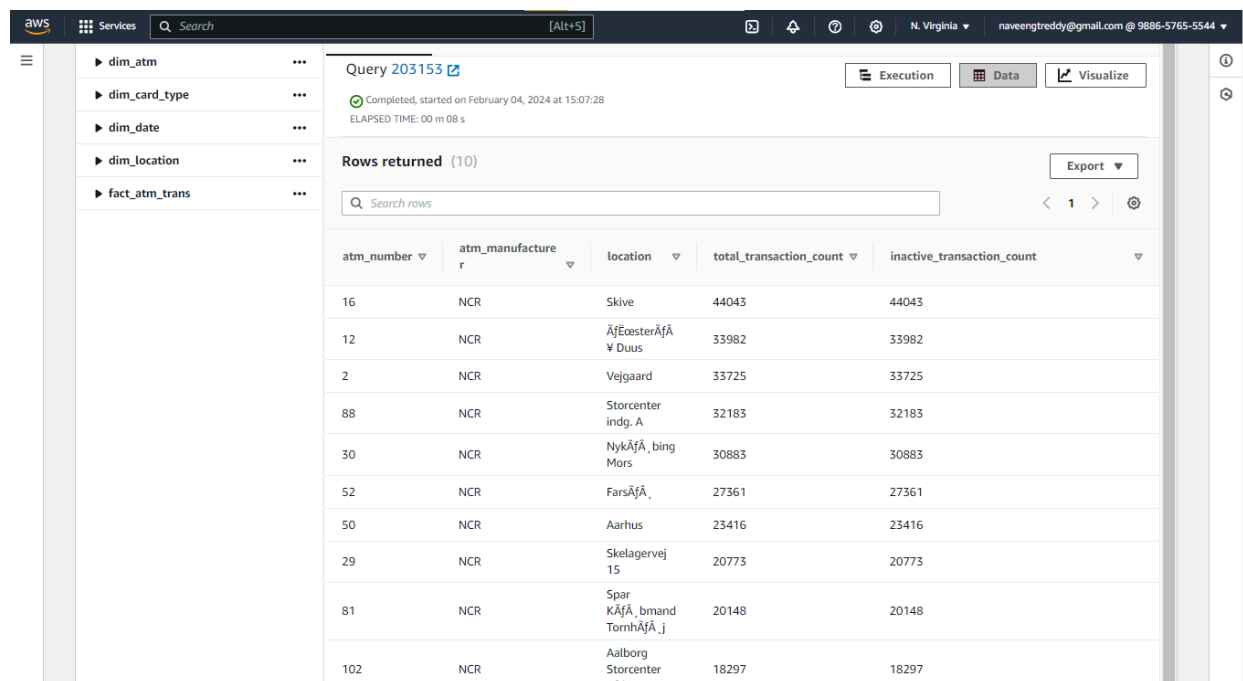


Solving analytical queries on Redshift Cluster

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

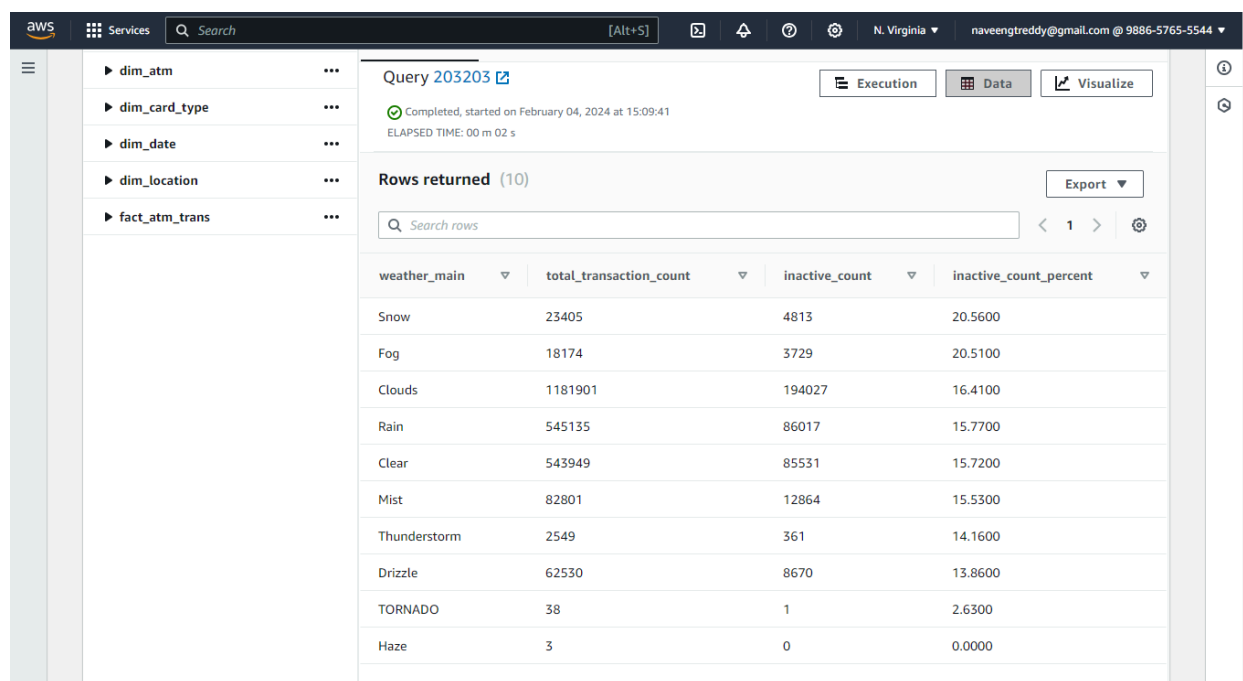
```
select a.atm_number, a.atm_manufacturer, l.location,
count(trans_id) as total_transaction_count,
sum(case when atm_status = 'Inactive' then 1 else 0 end) as
inactive_transaction_count
from elt_bank_assignment.fact_atm_trans f,
elt_bank_assignment.dim_atm a,elt_bank_assignment.dim_location l
where 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 inactive_transaction_count desc
limit 10;
```



atm_number	atm_manufacturer	location	total_transaction_count	inactive_transaction_count
16	NCR	Skive	44043	44043
12	NCR	ÅfEøsterÅfÅ v Duus	33982	33982
2	NCR	Vejgaard	33725	33725
88	NCR	Storcenter indg. A	32183	32183
30	NCR	NykÅfÅ, bing Mors	30883	30883
52	NCR	FarsÅfÅ,	27361	27361
50	NCR	Aarhus	23416	23416
29	NCR	Skelagervej 15	20773	20773
81	NCR	Spar KÅfÅ, bmand TørnhÅfÅ, j	20148	20148
102	NCR	Aalborg Storcenter ÅfÅ	18297	18297

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

```
select f.weather_main,
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 elt_bank_assignment.fact_atm_trans f
where f.weather_main != ''
group by f.weather_main
order by inactive_count_percent desc
limit 10;
```

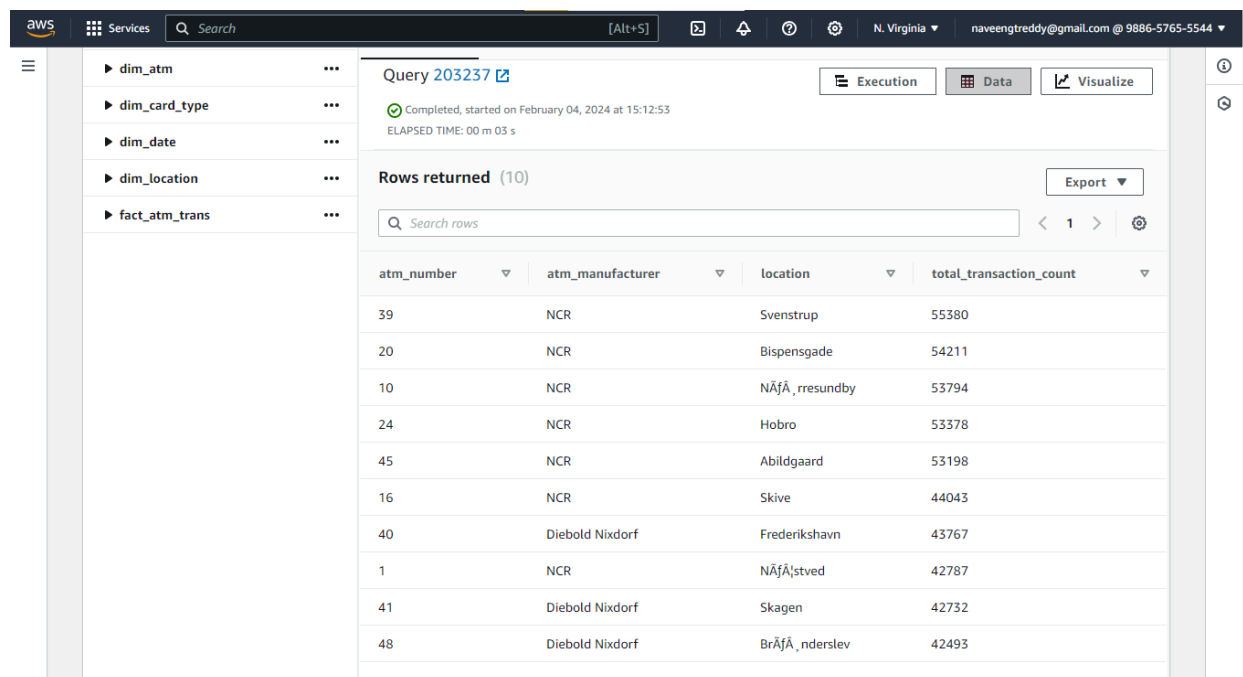


The screenshot shows the AWS Athena console interface. On the left, a sidebar lists the database schema: `dim_atm`, `dim_card_type`, `dim_date`, `dim_location`, and `fact_atm_trans`. The main panel displays the results of a SQL query (ID: 203203). The query is completed, and the results are shown in a table with 10 rows. The table columns are `weather_main`, `total_transaction_count`, `inactive_count`, and `inactive_count_percent`. The results are ordered by `inactive_count_percent` in descending order.

weather_main	total_transaction_count	inactive_count	inactive_count_percent
Snow	23405	4813	20.5600
Fog	18174	3729	20.5100
Clouds	1181901	194027	16.4100
Rain	545135	86017	15.7700
Clear	543949	85531	15.7200
Mist	82801	12864	15.5300
Thunderstorm	2549	361	14.1600
Drizzle	62530	8670	13.8600
TORNADO	38	1	2.6300
Haze	3	0	0.0000

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

```
select a.atm_number, a.atm_manufacturer, l.location,
count(trans_id) as total_transaction_count
from elt_bank_assignment.fact_atm_trans f,
elt_bank_assignment.dim_atm a,elt_bank_assignment.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_count desc
limit 10;
```

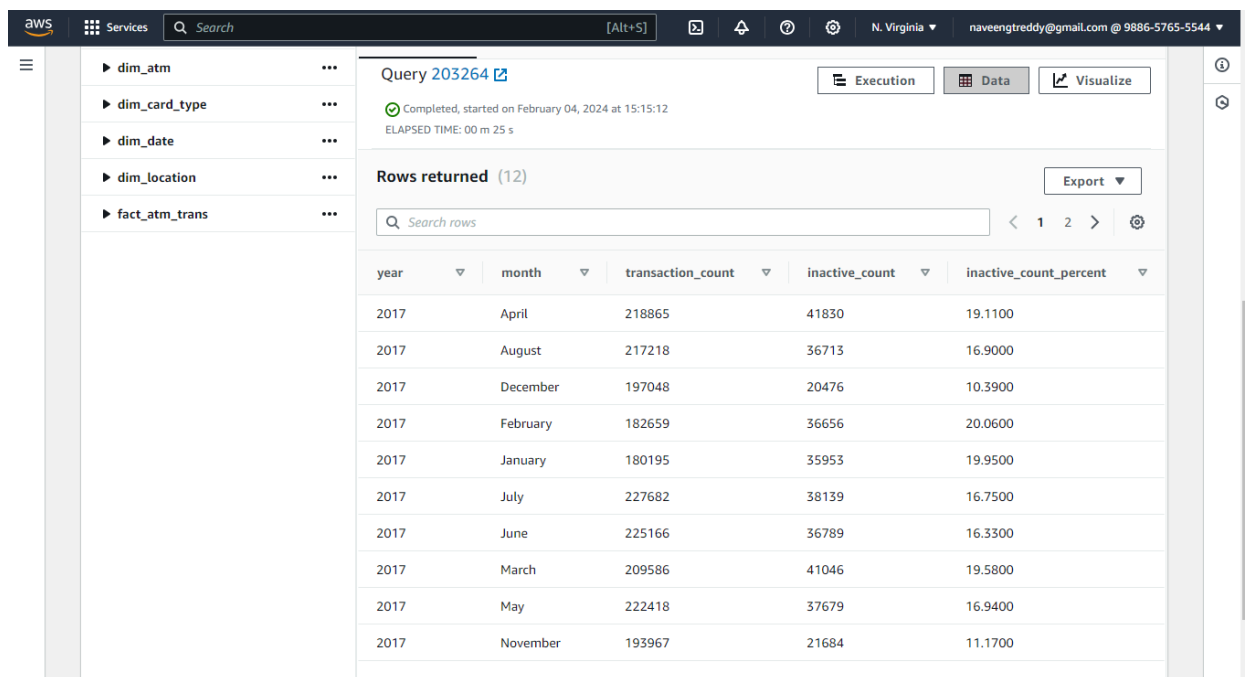


The screenshot shows the AWS Redshift console interface. On the left, a sidebar lists database schemas: `dim_atm`, `dim_card_type`, `dim_date`, `dim_location`, and `fact_atm_trans`. The main panel displays the execution details for 'Query 203237', which is completed. Below this, the 'Rows returned (10)' section shows a table of results. The table has four columns: `atm_number`, `atm_manufacturer`, `location`, and `total_transaction_count`. The data is sorted by `total_transaction_count` in descending order.

atm_number	atm_manufacturer	location	total_transaction_count
39	NCR	Svenstrup	55380
20	NCR	Bispensgade	54211
10	NCR	NÅfÅ, rresundby	53794
24	NCR	Hobro	53378
45	NCR	Abildgaard	53198
16	NCR	Skive	44043
40	Diebold Nixdorf	Frederikshavn	43767
1	NCR	NÅfÅ, stved	42787
41	Diebold Nixdorf	Skagen	42732
48	Diebold Nixdorf	BrÅfÅ, nderslev	42493

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

```
select c.year, c.month, c.transaction_count, d.inactive_count,
CAST(trunc(100.0*d.inactive_count/c.transaction_count,2) AS
NUMERIC(10,4)) as
inactive_count_percent from
(select a.year, a.month, count(b.trans_id) as transaction_count from
elt_bank_assignment.dim_date
a,elt_bank_assignment.fact_atm_trans b where a.date_id = b.date_id
group by
a.month, a.year) c
left join
(select a.year, a.month, count(b.atm_status) as inactive_count from
elt_bank_assignment.dim_date
a,elt_bank_assignment.fact_atm_trans b where a.date_id = b.date_id
and
b.atm_status='Inactive'
group by a.month, a.year) d
on c.year=d.year and c.month=d.month
order by c.year, c.month;
```



The screenshot shows the AWS Redshift console interface. On the left, a sidebar lists database schemas: dim_atm, dim_card_type, dim_date, dim_location, and fact_atm_trans. The main panel displays the results of a SQL query (Query 203264) which has completed successfully. The query results are shown in a table with 12 rows. The columns are year, month, transaction_count, inactive_count, and inactive_count_percent. The data shows transaction counts and inactive counts for various months in 2017, with the inactive count percentage calculated for each month.

year	month	transaction_count	inactive_count	inactive_count_percent
2017	April	218865	41830	19.1100
2017	August	217218	36713	16.9000
2017	December	197048	20476	10.3900
2017	February	182659	36656	20.0600
2017	January	180195	35953	19.9500
2017	July	227682	38139	16.7500
2017	June	225166	36789	16.3300
2017	March	209586	41046	19.5800
2017	May	222418	37679	16.9400
2017	November	193967	21684	11.1700

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 elt_bank_assignment.fact_atm_trans f,
elt_bank_assignment.dim_atm a,elt_bank_assignment.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;
```

aws Services Search [Alt+S] N. Virginia naveengtreddy@gmail.com @ 9886-5765-5544

Query 203302

Completed, started on February 04, 2024 at 15:18:18
ELAPSED TIME: 00 m 03 s

Execution Data Visualize

Rows returned (10)

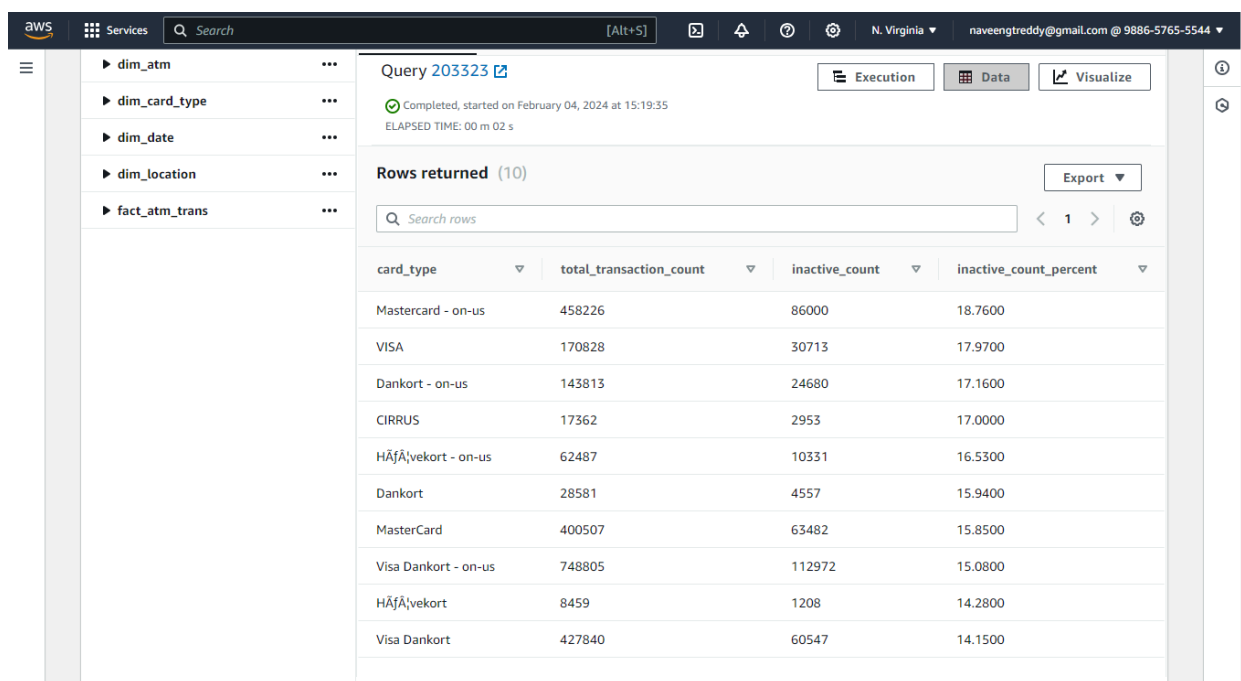
Export

Search rows

atm_number	atm_manufacturer	location	total_transaction_amount
39	NCR	Svenstrup	277097637
20	NCR	Bispensgade	271008803
24	NCR	Hobro	268289882
10	NCR	NÅfÅ, resundby	267379103
45	NCR	Abildgaard	265639616
16	NCR	Skive	220677013
40	Diebold Nixdorf	Frederikshavn	219812287
41	Diebold Nixdorf	Skagen	214127315
1	NCR	NÅfÅstved	213721117
48	Diebold Nixdorf	BrÅfÅ, nderslev	212883099

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 elt_bank_assignment.fact_atm_trans f,
elt_bank_assignment.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;
```

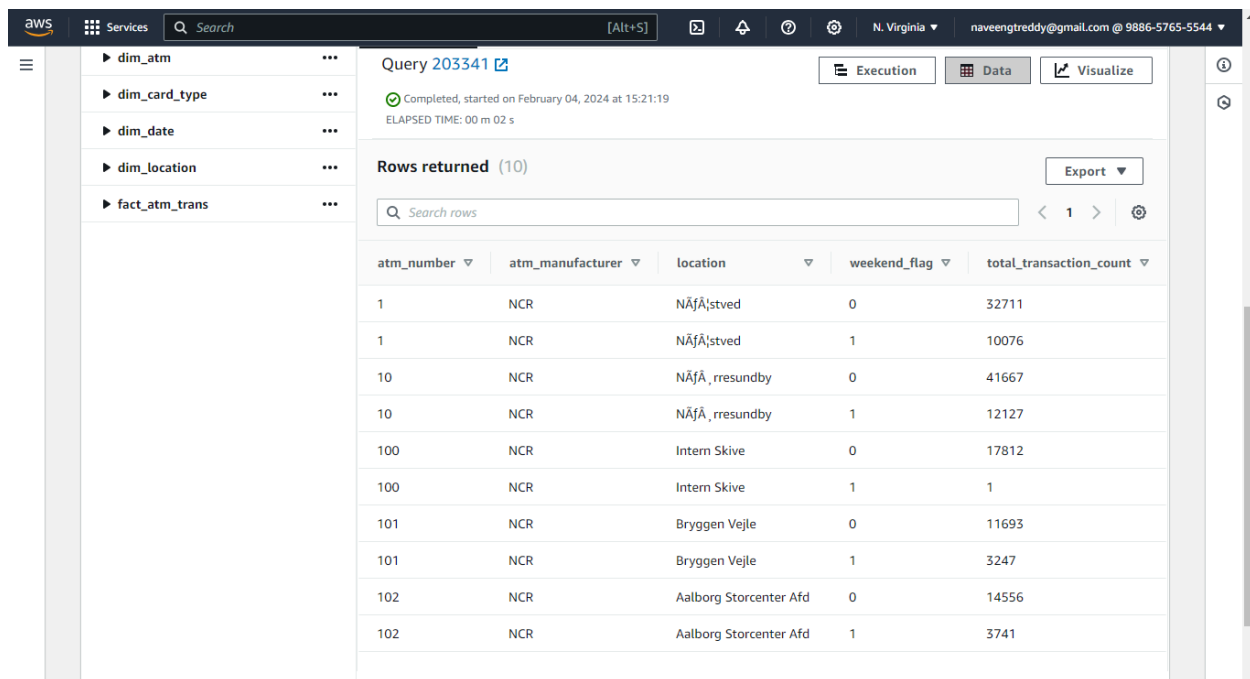


The screenshot shows the AWS Athena console interface. On the left, a sidebar lists the database schema: `dim_atm`, `dim_card_type`, `dim_date`, `dim_location`, and `fact_atm_trans`. The main panel displays the execution details for 'Query 203323', which is completed. Below this, the 'Rows returned' section shows 10 rows of data. The data is presented in a table with the following columns: `card_type`, `total_transaction_count`, `inactive_count`, and `inactive_count_percent`.

card_type	total_transaction_count	inactive_count	inactive_count_percent
Mastercard - on-us	458226	86000	18.7600
VISA	170828	30713	17.9700
Dankort - on-us	143813	24680	17.1600
CIRRUS	17362	2953	17.0000
HÅfÅ\vekort - on-us	62487	10331	16.5300
Dankort	28581	4557	15.9400
MasterCard	400507	63482	15.8500
Visa Dankort - on-us	748805	112972	15.0800
HÅfÅ\vekort	8459	1208	14.2800
Visa Dankort	427840	60547	14.1500

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 elt_bank_assignment.fact_atm_trans f,
elt_bank_assignment.dim_atm a,elt_bank_assignment.dim_location
l,elt_bank_assignment.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;
```



Query 203341

Completed, started on February 04, 2024 at 15:21:19
ELAPSED TIME: 00 m 02 s

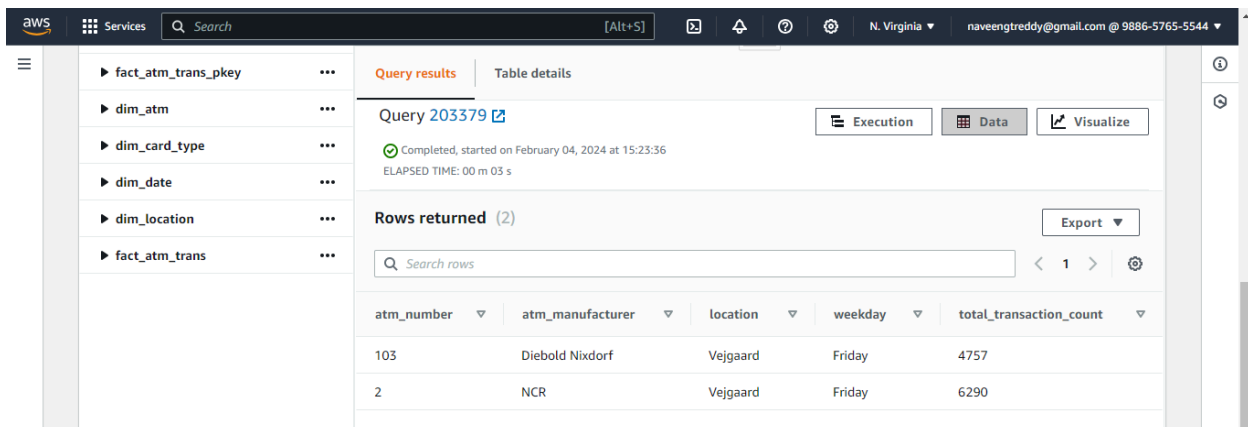
Rows returned (10)

Search rows

atm_number	atm_manufacturer	location	weekend_flag	total_transaction_count
1	NCR	NÄfÄ\stved	0	32711
1	NCR	NÄfÄ\stved	1	10076
10	NCR	NÄfÄ , rresundby	0	41667
10	NCR	NÄfÄ , rresundby	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

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 elt_bank_assignment.fact_atm_trans f inner join
elt_bank_assignment.dim_atm a on
f.atm_id =
a.atm_id
inner join elt_bank_assignment.dim_location l on a.atm_location_id =
l.location_id
inner join elt_bank_assignment.dim_date d on f.date_id = d.date_id
where l.location = 'Vejgaard' and d.weekday in
(
select d.weekday
from elt_bank_assignment.fact_atm_trans f inner join
elt_bank_assignment.dim_date d
on f.date_id = d.date_id
inner join elt_bank_assignment.dim_location l on f.location_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 Results interface. The query is identified as 'Query 203379' and has completed successfully. The results show 2 rows returned, sorted by total_transaction_count in descending order.

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