

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

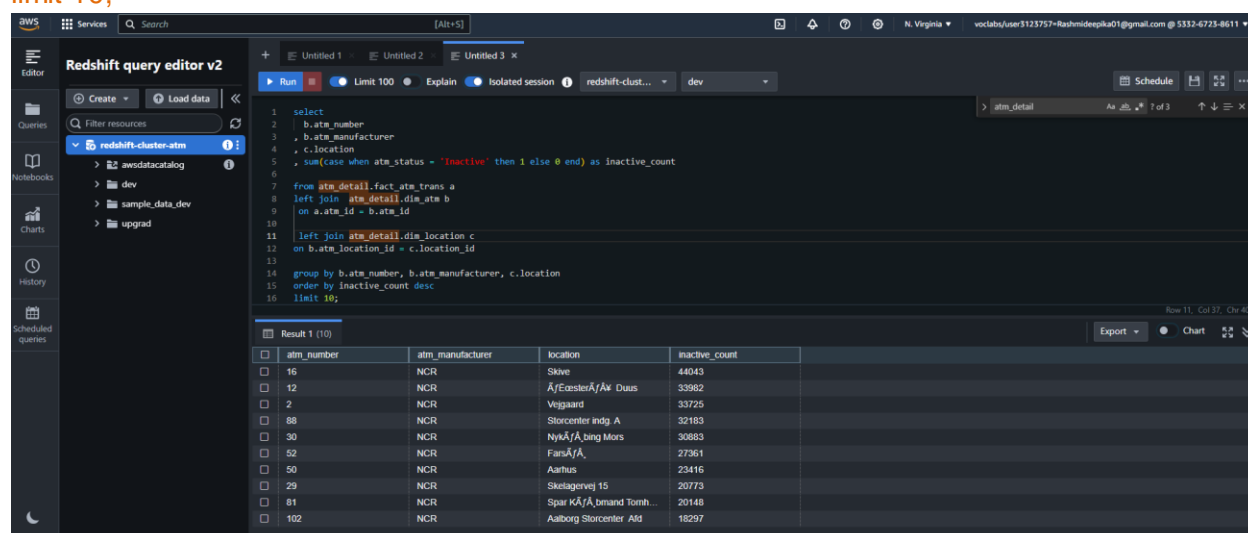
Query:

```
select
  b.atm_number
, b.atm_manufacturer
, c.location
, sum(case when atm_status = 'Inactive' then 1 else 0 end) as inactive_count

from atm_detail.fact_atm_trans a
left join atm_detail.dim_atm b
on a.atm_id = b.atm_id

left join atm_detail.dim_location c
on b.atm_location_id = c.location_id

group by b.atm_number, b.atm_manufacturer, c.location
order by inactive_count desc
limit 10;
```



The screenshot shows the AWS Redshift Query Editor v2 interface. The query is written in the editor and has been executed. The results are displayed in a table with 4 columns: atm_number, atm_manufacturer, location, and inactive_count. The table shows the top 10 ATMs with the highest number of inactive transactions.

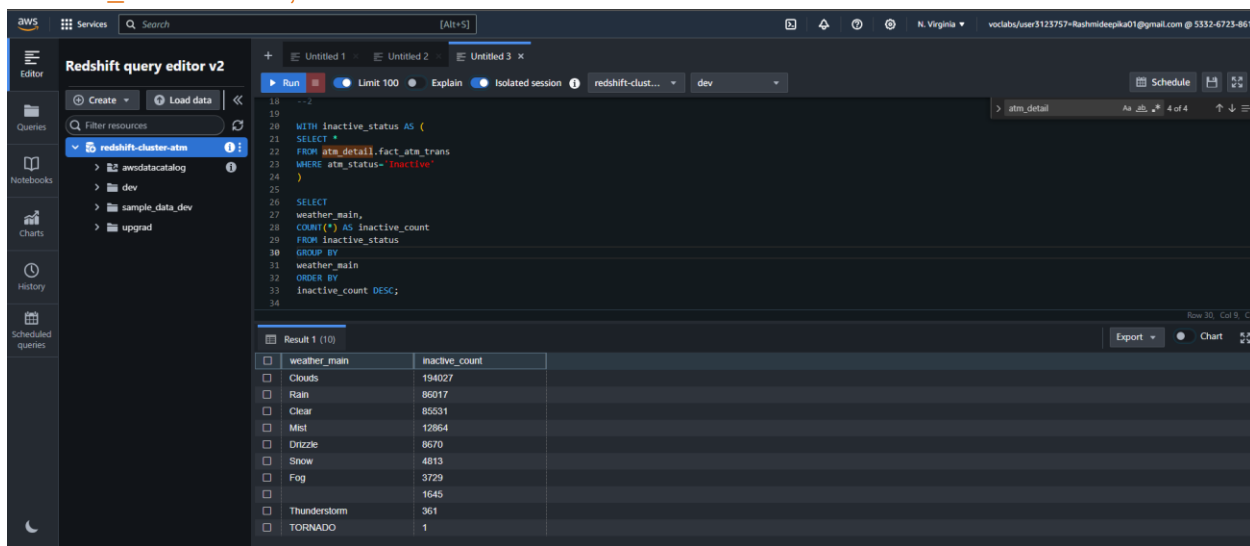
atm_number	atm_manufacturer	location	inactive_count
16	NCR	Skive	44043
12	NCR	Århus	33982
2	NCR	Vejsgaard	33725
88	NCR	Storcenter indg. A	32183
30	NCR	Nykj. A. Bing Mors	30883
52	NCR	Farsk. A.	27361
50	NCR	Aarhus	23416
29	NCR	Skjelagervej 15	20773
81	NCR	Spar K&A. Demand Torsh...	20148
102	NCR	Aalborg Storcenter Ald	18297

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

Query:

```
WITH inactive_status AS (
SELECT *
FROM atm_detail.fact_atm_trans
WHERE atm_status='Inactive'
)
```

```
SELECT
weather_main,
COUNT(*) AS inactive_count
FROM inactive_status
GROUP BY
weather_main
ORDER BY
inactive_count DESC;
```



The screenshot shows the AWS Redshift Query Editor v2 interface. The query editor displays the following SQL query:

```
WITH inactive_status AS (
SELECT *
FROM atm_detail.fact_atm_trans
WHERE atm_status='Inactive'
)

SELECT
weather_main,
COUNT(*) AS inactive_count
FROM inactive_status
GROUP BY
weather_main
ORDER BY
inactive_count DESC;
```

The query has been executed, and the results are displayed in a table with 10 rows. The table has two columns: `weather_main` and `inactive_count`.

weather_main	inactive_count
Clouds	194027
Rain	86017
Clear	85531
Mist	12864
Drizzle	8670
Snow	4813
Fog	3729
	1645
Thunderstorm	361
TORNADO	1

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

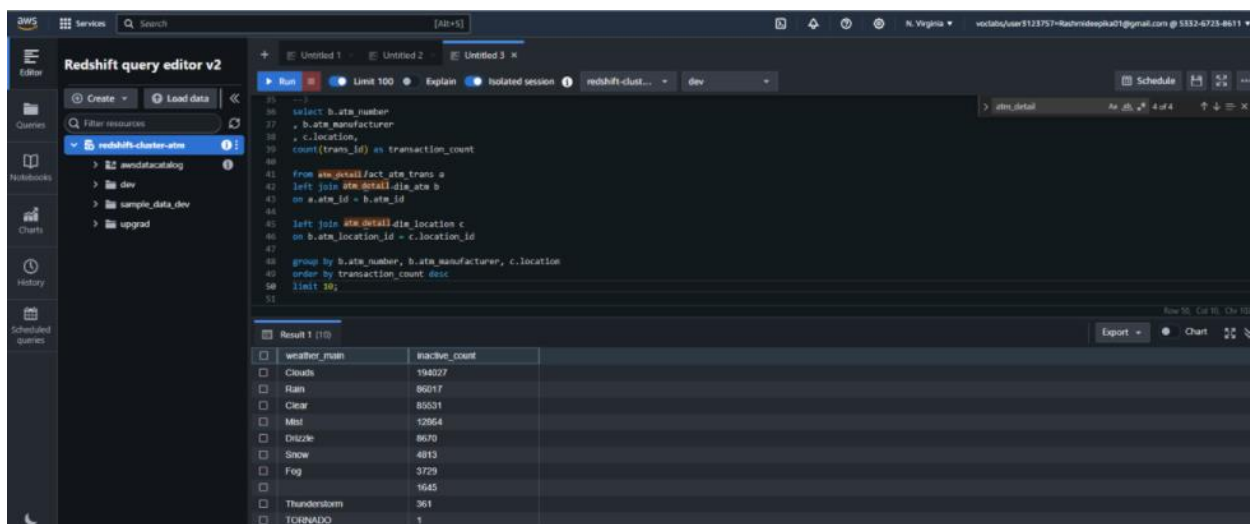
Query:

```
select b.atm_number
, b.atm_manufacturer
, c.location,
count(trans_id) as transaction_count
```

```
from atm_detail.fact_atm_trans a
left join atm_detail.dim_atm b
on a.atm_id = b.atm_id
```

```
left join atm_detail.dim_location c
on b.atm_location_id = c.location_id
```

```
group by b.atm_number, b.atm_manufacturer, c.location
order by transaction_count desc
limit 10;
```



The screenshot shows the AWS Redshift query editor v2 interface. The query is written in the editor and has been executed. The results are displayed in a table with 10 rows and 3 columns: atm_number, atm_manufacturer, and location. The results are ordered by transaction_count in descending order.

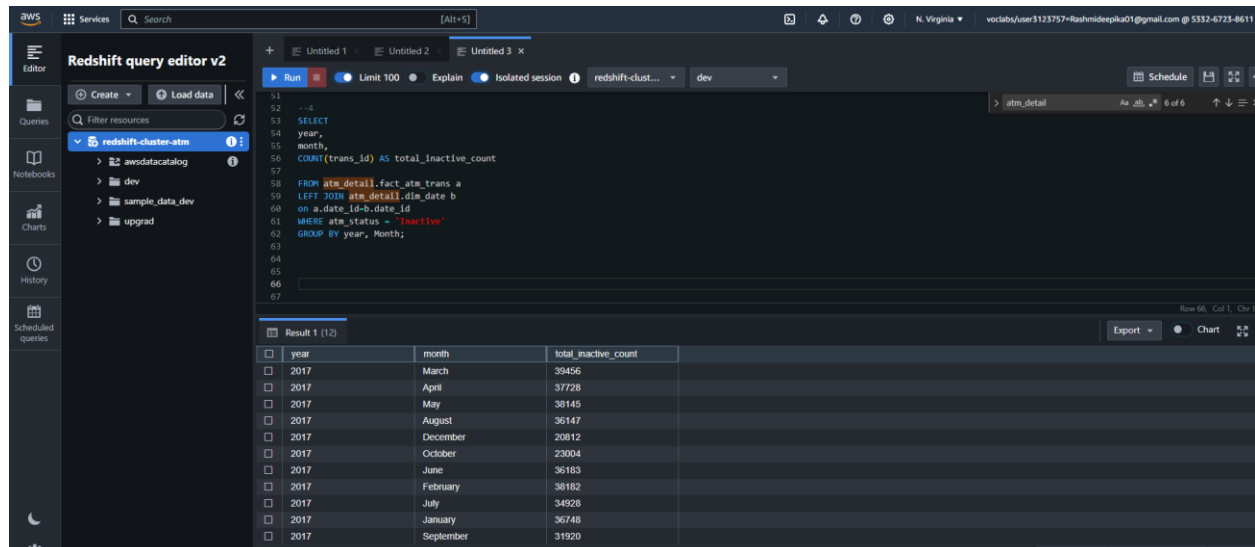
atm_number	atm_manufacturer	location	transaction_count
194027	Weather	Clouds	194027
86017	Rain	Rain	86017
80501	Clear	Clear	80501
12864	Mist	Mist	12864
8670	Drizzle	Drizzle	8670
4813	Snow	Snow	4813
3729	Fog	Fog	3729
1645	Thunderstorm	Thunderstorm	1645
361	Thunderstorm	Thunderstorm	361
1	TORNADO	TORNADO	1

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

Query:

```
SELECT
year,
month,
COUNT(trans_id) AS total_inactive_count
```

```
FROM atm_detail.fact_atm_trans a
LEFT JOIN atm_detail.dim_date b
on a.date_id=b.date_id
WHERE atm_status = 'Inactive'
GROUP BY year, Month;
```



The screenshot shows the AWS Redshift query editor v2 interface. The query is executed, and the results are displayed in a table format. The table has three columns: year, month, and total_inactive_count. The results show the number of inactive ATM transactions for each month in 2017.

year	month	total_inactive_count
2017	March	39456
2017	April	37728
2017	May	38145
2017	August	36147
2017	December	20812
2017	October	23004
2017	June	36183
2017	February	38182
2017	July	34928
2017	January	36748
2017	September	31920

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

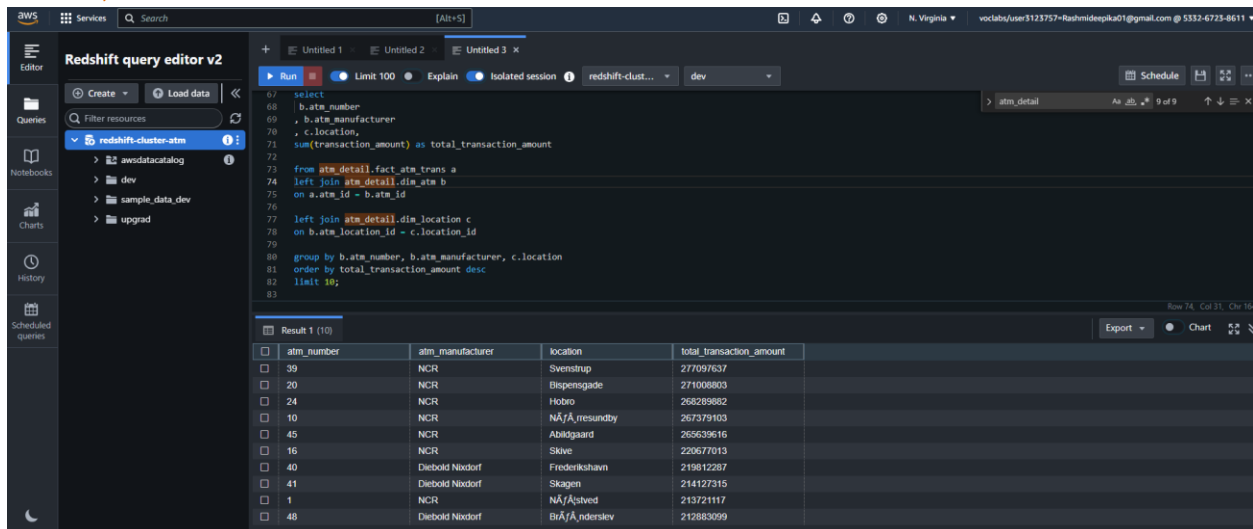
Query:

```
select
  b.atm_number
, b.atm_manufacturer
, c.location,
sum(transaction_amount) as total_transaction_amount
```

```
from atm_detail.fact_atm_trans a
left join atm_detail.dim_atm b
on a.atm_id = b.atm_id
```

```
left join atm_detail.dim_location c
on b.atm_location_id = c.location_id
```

```
group by b.atm_number, b.atm_manufacturer, c.location
order by total_transaction_amount 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 4 columns: atm_number, atm_manufacturer, location, and total_transaction_amount. The results are sorted in descending order of total_transaction_amount, showing the top 10 ATMs.

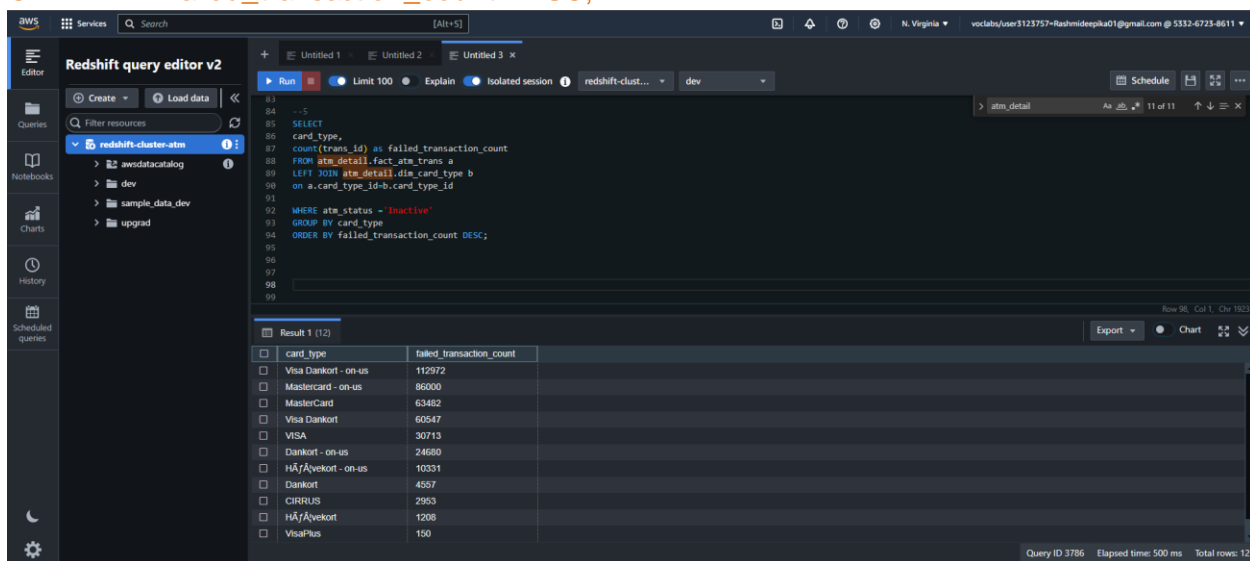
atm_number	atm_manufacturer	location	total_transaction_amount
39	NCR	Svensstrup	277097637
20	NCR	Bispensgade	271008803
24	NCR	Hobro	268289882
10	NCR	NÅ/Å resundby	267379103
45	NCR	Abildgaard	265639616
16	NCR	Sløve	220677013
40	Diebold Nixdorf	Frederikshavn	219812287
41	Diebold Nixdorf	Shagen	214127315
1	NCR	NÅ/Åsted	213721117
48	Diebold Nixdorf	BrÅ/Å nderslev	212883099

6. Number of failed ATM transactions across various card types

Query:

```
SELECT
card_type,
count(trans_id) as failed_transaction_count
FROM atm_detail.fact_atm_trans a
LEFT JOIN atm_detail.dim_card_type b
on a.card_type_id=b.card_type_id
```

```
WHERE atm_status ='Inactive'
GROUP BY card_type
ORDER BY failed_transaction_count DESC;
```



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 12 rows and 2 columns: card_type and failed_transaction_count.

card_type	failed_transaction_count
Visa Dankort - on-us	112972
Mastercard - on-us	86000
MasterCard	63482
Visa Dankort	60547
VISA	30713
Dankort - on-us	24680
HÅ/Åvekort - on-us	10331
Dankort	4557
CHIRRIUS	2953
HÅ/Åvekort	1208
VisaPlus	150

Query ID: 3786 Elapsed time: 500 ms Total rows: 12

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

Query:

```
WITH weekday_weekend AS (  
  SELECT  
    date_id,  
    CASE WHEN weekday IN ('Saturday', 'Sunday') THEN 1 ELSE  
    0 END AS weekday_weekend_flag  
  FROM  
    atm_detail.dim_date  
)  
SELECT  
  atm.ATM_number,  
  atm.ATM_manufacturer,  
  loc.location,  
  wd.weekday_weekend_flag,  
  COUNT(t.trans_id) AS total_transaction_count  
FROM  
  atm_detail.fact_atm_trans t  
LEFT JOIN  
  atm_detail.dim_atm atm USING(atm_id)  
LEFT JOIN  
  weekday_weekend wd USING (date_id)  
LEFT JOIN  
  atm_data.dim_location loc ON loc.location_id=  
  t.weather_loc_id  
GROUP BY  
  atm.ATM_number,  
  atm.ATM_manufacturer,  
  loc.location,  
  wd.weekday_weekend_flag  
ORDER BY  
  atm.ATM_number,  
  atm.ATM_manufacturer,  
  loc.location,  
  wd.weekday_weekend_flag, total transaction count;
```

Run Limit 100 Explain Isolated session etl-sn-bank-atm dev Schedule

```

75 REWRITE: COLUMNS: @UR_MUNIT@C LUFER TO @UR_MUNIT@C LUFER;
76
77
78
79
80
81 with weekday_weekend as (select date_id,
82 | case when weekday in ('Saturday','Sunday') then 1 else 0 end as weekday_weekend_flag
83 from atm_detail.dim_date)
84 select ATM_number, ATM_manufacturer, location, weekday_weekend_flag , count(trans_id) as total_transaction_count
85 from atm_detail.fact_atm_trans t left join atm_detail.dim_atm using(atm_id) left join weekday_weekend using(date_id) left join atm_detail.dim_location dd on dd.location_id=t.weather_loc_id
86 group by ATM_number, ATM_manufacturer, location, weekday_weekend_flag
87 order by ATM_number, ATM_manufacturer, location, weekday_weekend_flag,total_transaction_count
88
89
90
91
92
93

```

Result 1 (100) Export Chart

atm_number	atm_manufactures	location	weekday_weekend_flag	total_transaction_count
1	NCR	NÄ/rÄstved	0	32711
1	NCR	NÄ/rÄstved	1	19076
10	NCR	NÄ/rÄ_resundby	0	41667
10	NCR	NÄ/rÄ_resundby	1	12127

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

Query:

```

WITH ranks AS (
SELECT DISTINCT
atm_id,
atm_number,
location,
weekday,
COUNT(trans_id) AS total_transaction_count,
ROW_NUMBER() OVER (PARTITION BY atm_id ORDER BY COUNT(trans_id) DESC) AS
highest
FROM
atm_detail.fact_atm_trans f
LEFT JOIN atm_detail.dim_atm
USING (atm_id)
LEFT JOIN
atm_detail.dim_location al
ON f.weather_loc_id = al.location_id
LEFT JOIN
atm_detail.dim_date USING (date_id)
WHERE
location = 'Vejgaard'
GROUP BY
weekday,
atm_id,
atm_number,
location

```


)
SELECT * FROM ranks WHERE highest=1;

+
Untitled 1 x
Untitled 2 x
dim_atm table x

Run
Limit 100
Explain
Isolated session
atl-sn-bank-atm
dev

```

92
93
94
95
96
97
98 with ranks as (select distinct(atm_id),atm_number,location, weekday,count(trans_id) as total_transaction_count,
99   row_number() over (PARTITION by atm_id order by count(trans_id) desc) as highest
100 from atm_detail.fact_atm_trans f left join atm_detail.dim_atm using(atm_id)
101 left join atm_detail.dim_location al on f.weather_loc_id=al.location_id left join atm_detail.dim_date using(date_id)
102 where location='vejgaard'
103 group by weekday,atm_id,atm_number,location
104 )
105 select * from ranks where highest=1;
106
107
108
109

```

Result 1 (2)

atm_id	atm_number	location	weekday	total_transaction_count
10	103	Vejgaard	Friday	4757
39	2	Vejgaard	Friday	6290