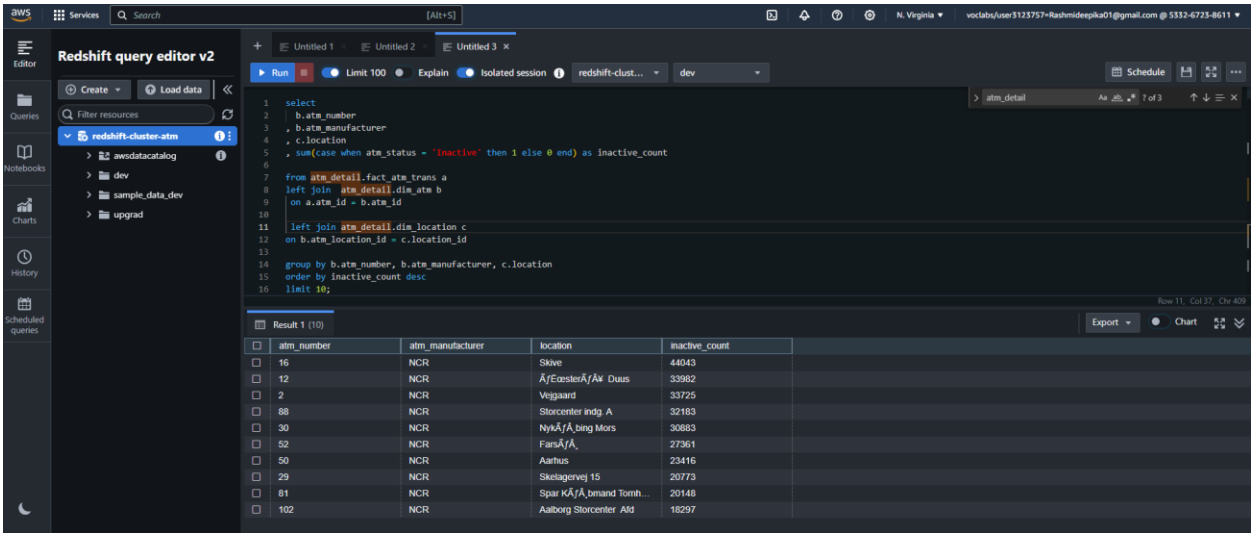
Solving analytical queries on Redshift Cluster

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:

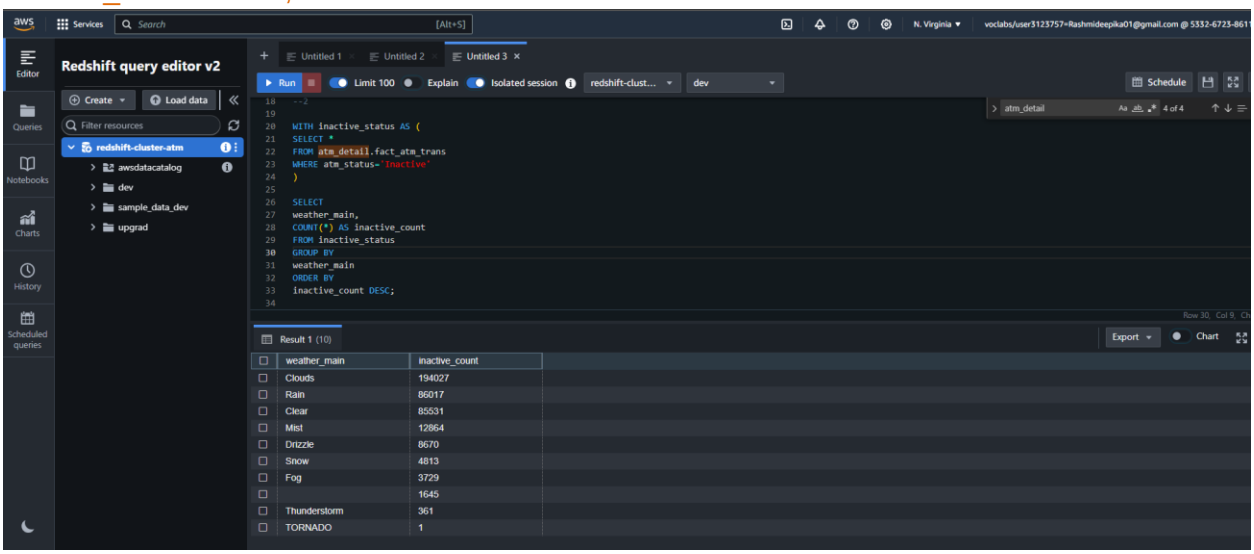
```
select
  b.atm_number
, b.atm_manufacturer
, c.location
, sum(case when atm_status = 'Inactive' then 1 else 0 end) as inactive_count

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;
```



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

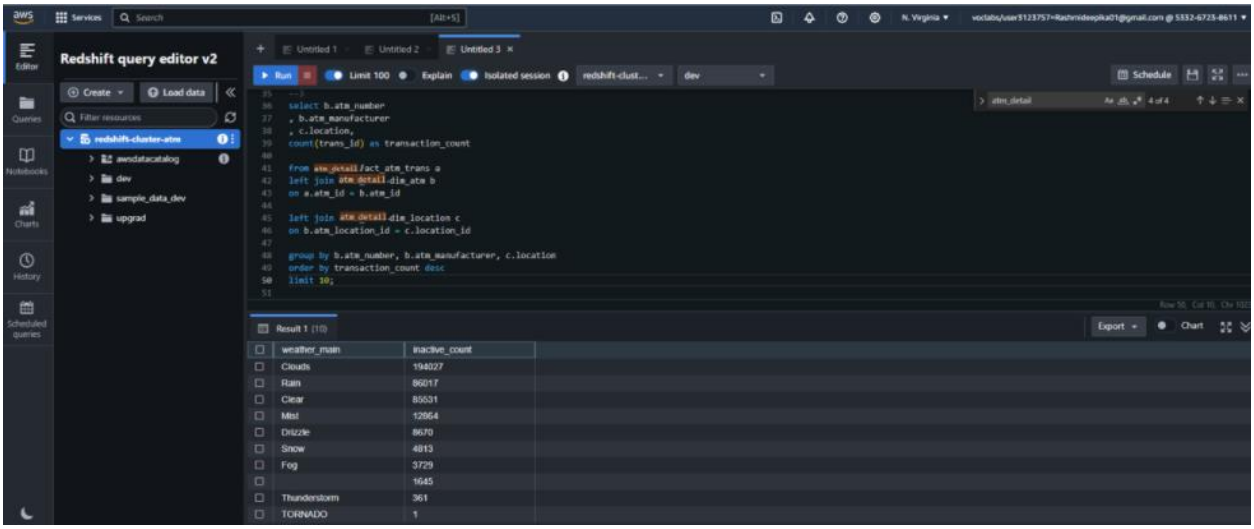
Query:

```
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;
```



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

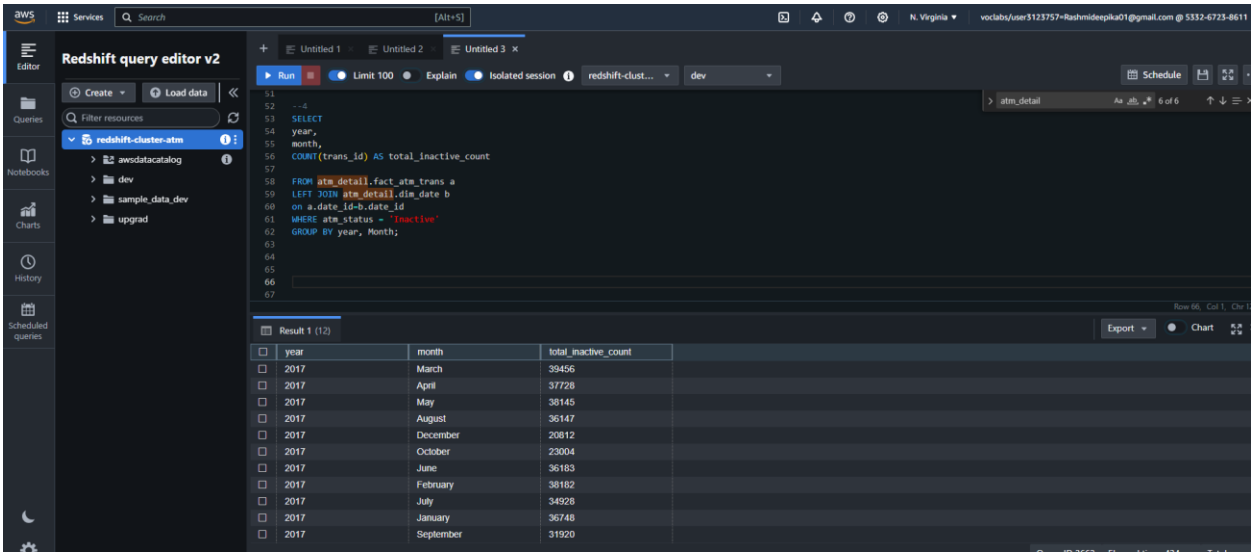
```
select b.atm_number
, b.atm_manufacturer
, c.location,
count(trans_id) as transaction_count

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;
```



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

Query:

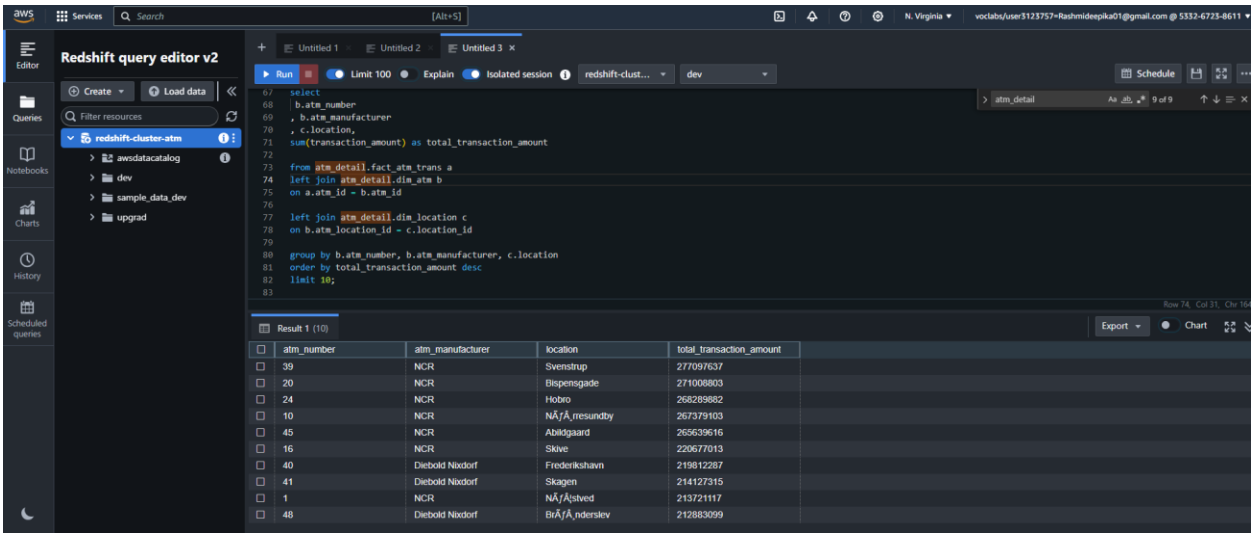
```
SELECT
year,
month,
COUNT(trans_id) AS total_inactive_count

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;
```



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 **Top 10 ATMs with the highest total withdrawn amount throughout the year**

Query:

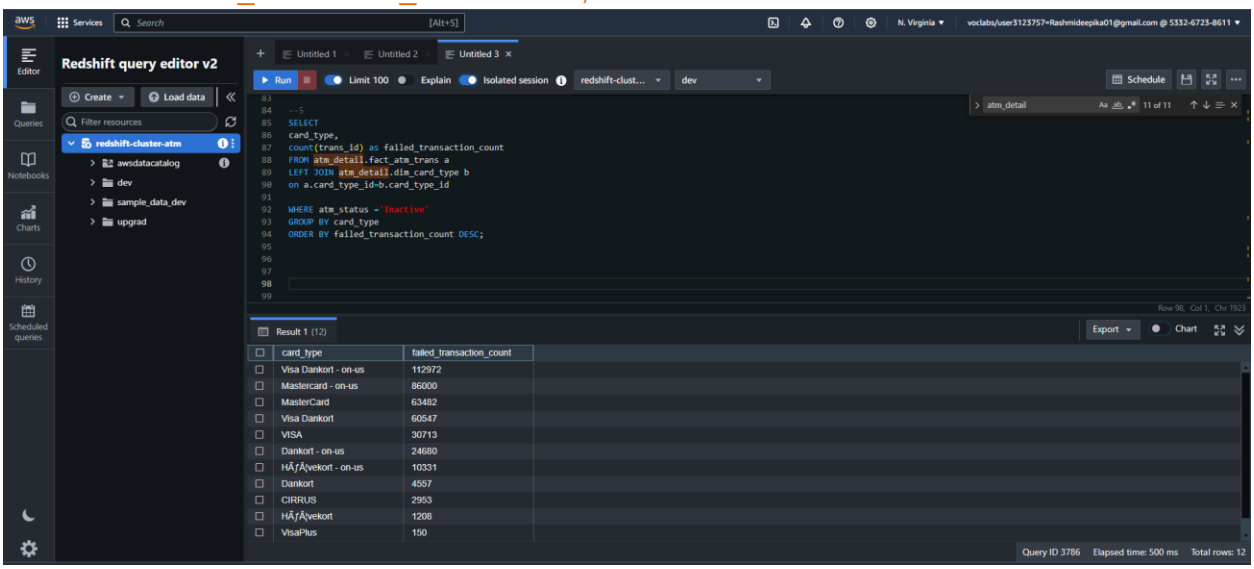
```
select
 b.atm_number
, b.atm_manufacturer
, c.location,
sum(transaction_amount) as total_transaction_amount

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;
```



6. **Number of failed ATM transactions across various card types**

Query:

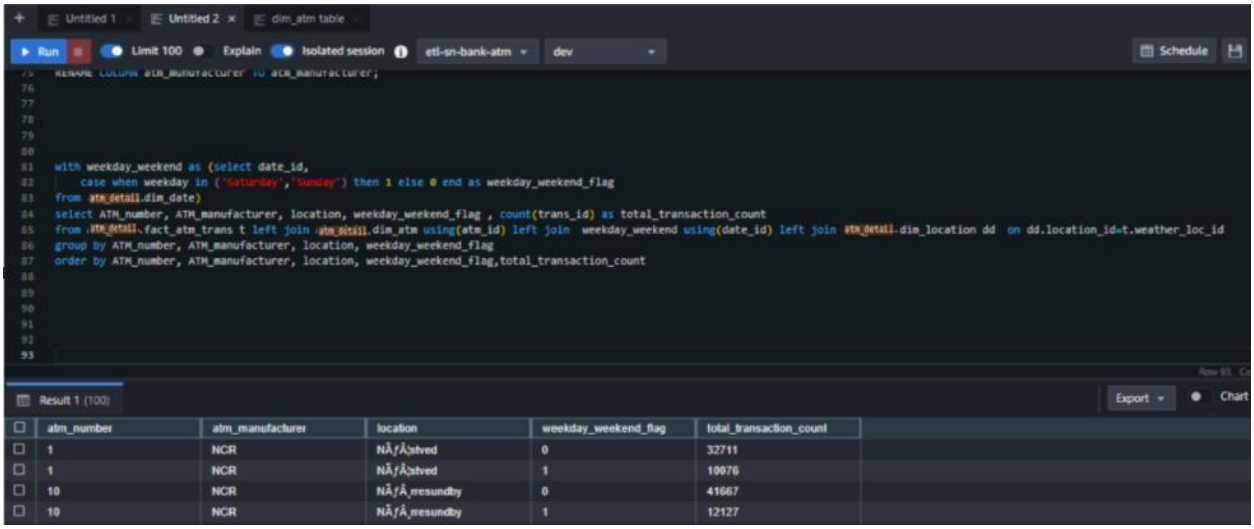
```
SELECT
card_type,
count(trans_id) as failed_transaction_count
FROM atm_detail.fact_atm_trans a
LEFT JOIN atm_detail.dim_card_type b
on a.card_type_id=b.card_type_id

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



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;
```



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;