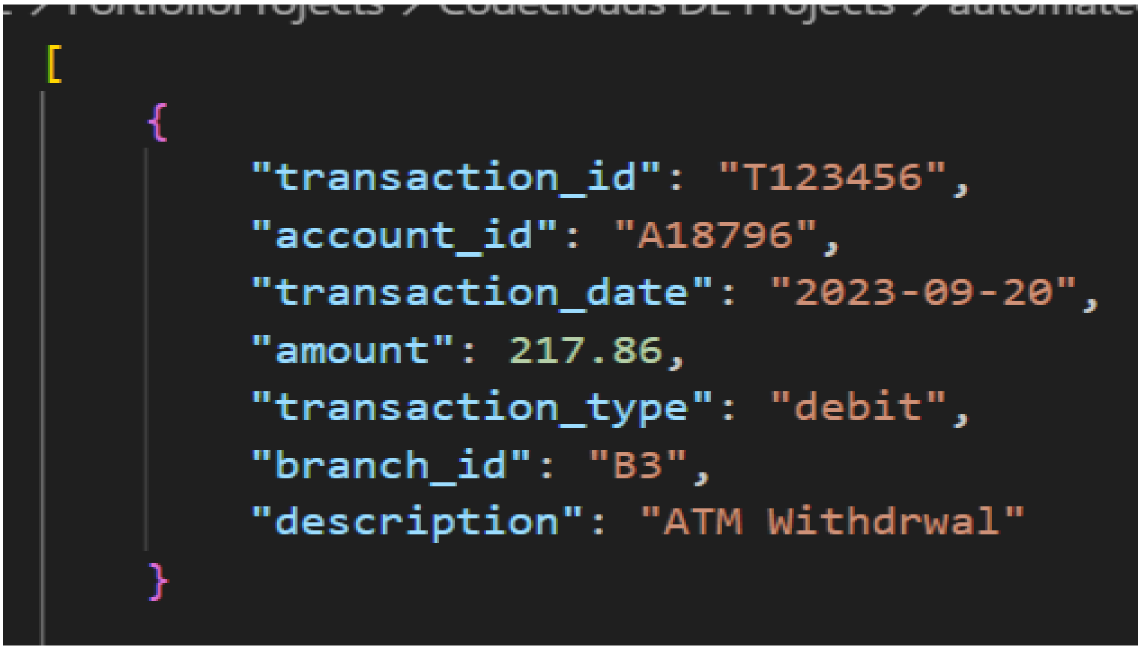
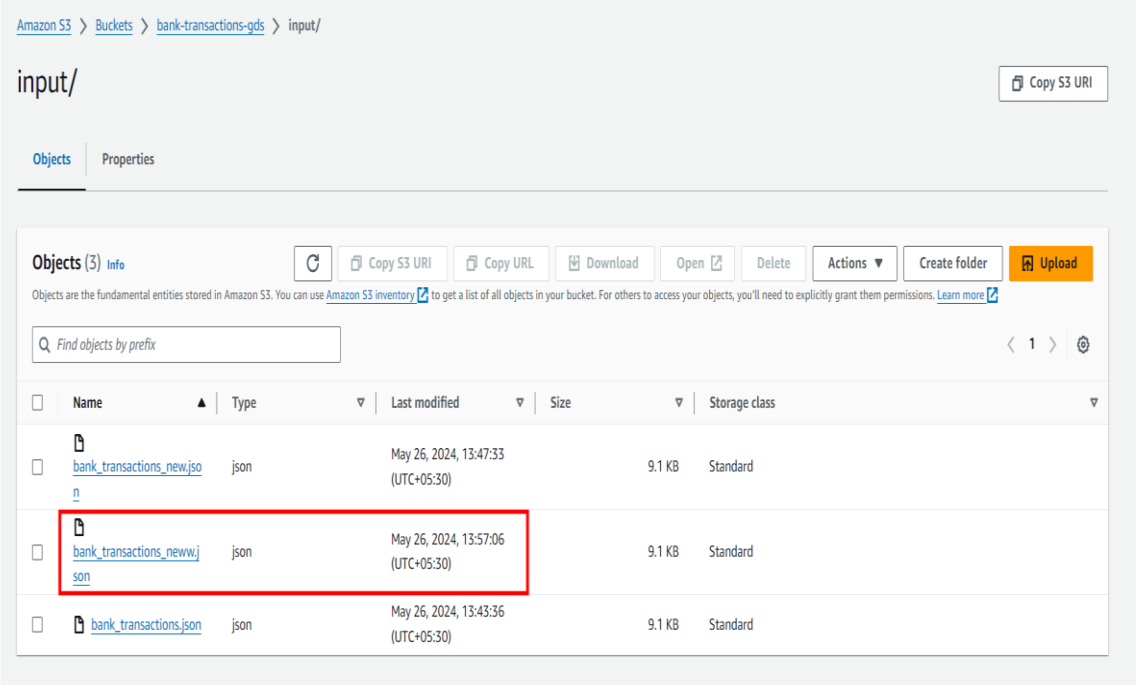
Bank Transactions Batch Data Processing with AWS

1. This project was about implementing a batch data processing pipeline in AWS to process daily bank transactions stored in JSON files. Goal was to set up an AWS data processing pipeline that automatically processes this data as soon as it lands on S3, transforming and storing it for querying.

2. A daily JSON file containing bank transactions was being dropped into an S3 bucket. Once new file uploaded, data processing triggered automatically with AWS Lambda function.



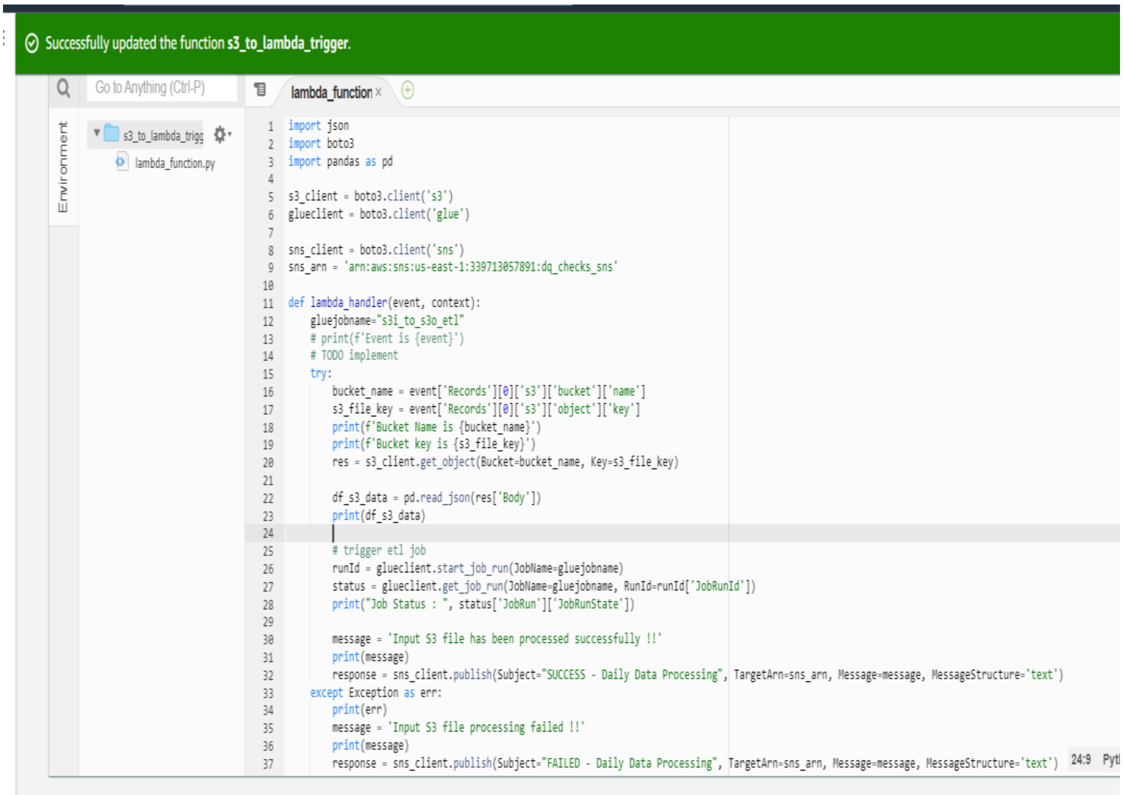
```
[
    {
        "transaction_id": "T123456",
        "account_id": "A18796",
        "transaction_date": "2023-09-20",
        "amount": 217.86,
        "transaction_type": "debit",
        "branch_id": "B3",
        "description": "ATM Withdrwal"
    }
```



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name ▲	Type	Last modified	Size	Storage class
□	bank_transactions_new.json	json	May 26, 2024, 13:47:33 (UTC+05:30)	9.1 KB	Standard
□	bank_transactions_neww.json	json	May 26, 2024, 13:57:06 (UTC+05:30)	9.1 KB	Standard
□	bank_transactions.json	json	May 26, 2024, 13:43:36 (UTC+05:30)	9.1 KB	Standard

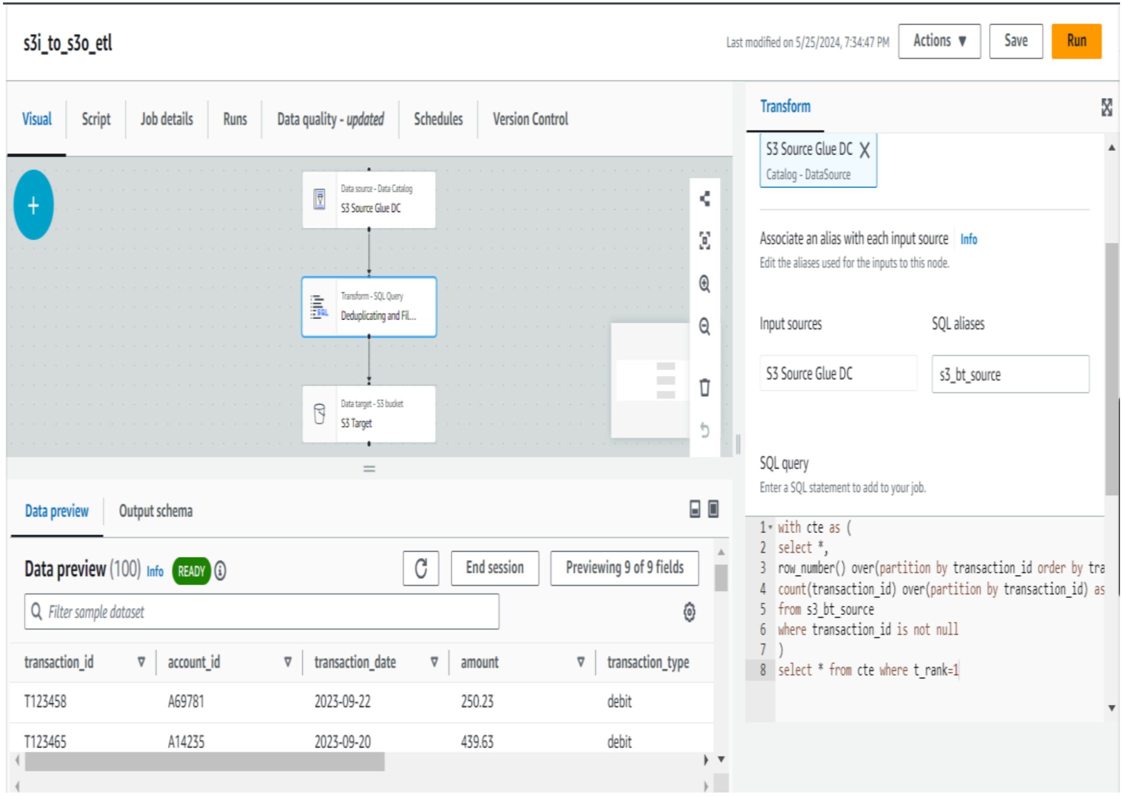
3. Within Lambda, I invoked the Glue job. Following step by step starting, used AWS Glue to read the JSON file from the S3 bucket



Successfully updated the function **s3_to_lambda_trigger.**

```python
import json
import boto3
import pandas as pd

s3_client = boto3.client('s3')
glueclient = boto3.client('glue')

sns_client = boto3.client('sns')
sns_arn = 'arn:aws:sns:us-east-1:339713057891:dq_checks_sns'

def lambda_handler(event, context):
    gluejobname="s3i_to_s3o_etl"
    # print(f'Event is {event}')
    # TODO implement
    try:
        bucket_name = event['Records'][0]['s3']['bucket']['name']
        s3_file_key = event['Records'][0]['s3']['object']['key']
        print(f'Bucket Name is {bucket_name}')
        print(f'Bucket key is {s3_file_key}')
        res = s3_client.get_object(Bucket=bucket_name, Key=s3_file_key)

        df_s3_data = pd.read_json(res['Body'])
        print(df_s3_data)
        
        # trigger etl job
        runId = glueclient.start_job_run(JobName=gluejobname)
        status = glueclient.get_job_run(JobName=gluejobname, RunId=runId['JobRunId'])
        print("Job Status : ", status['JobRun']['JobRunState'])

        message = 'Input S3 file has been processed successfully !!'
        print(message)
        response = sns_client.publish(Subject="SUCCESS - Daily Data Processing", TargetArn=sns_arn, Message=message, MessageStructure='text')
    except Exception as err:
        print(err)
        message = 'Input S3 file processing failed !!'
        print(message)
        response = sns_client.publish(Subject="FAILED - Daily Data Processing", TargetArn=sns_arn, Message=message, MessageStructure='text')
```

4. Implemented transformations such as filtering out any transactions with null values, and deduplicating any repeated transactions based on transaction_ID. Then converted the JSON format into a columnar format which is considered more optimized for querying



Associate an alias with each input source Info

Edit the aliases used for the inputs to this node.

Input sources	SQL aliases
S3 Source Glue DC	s3_bt_source

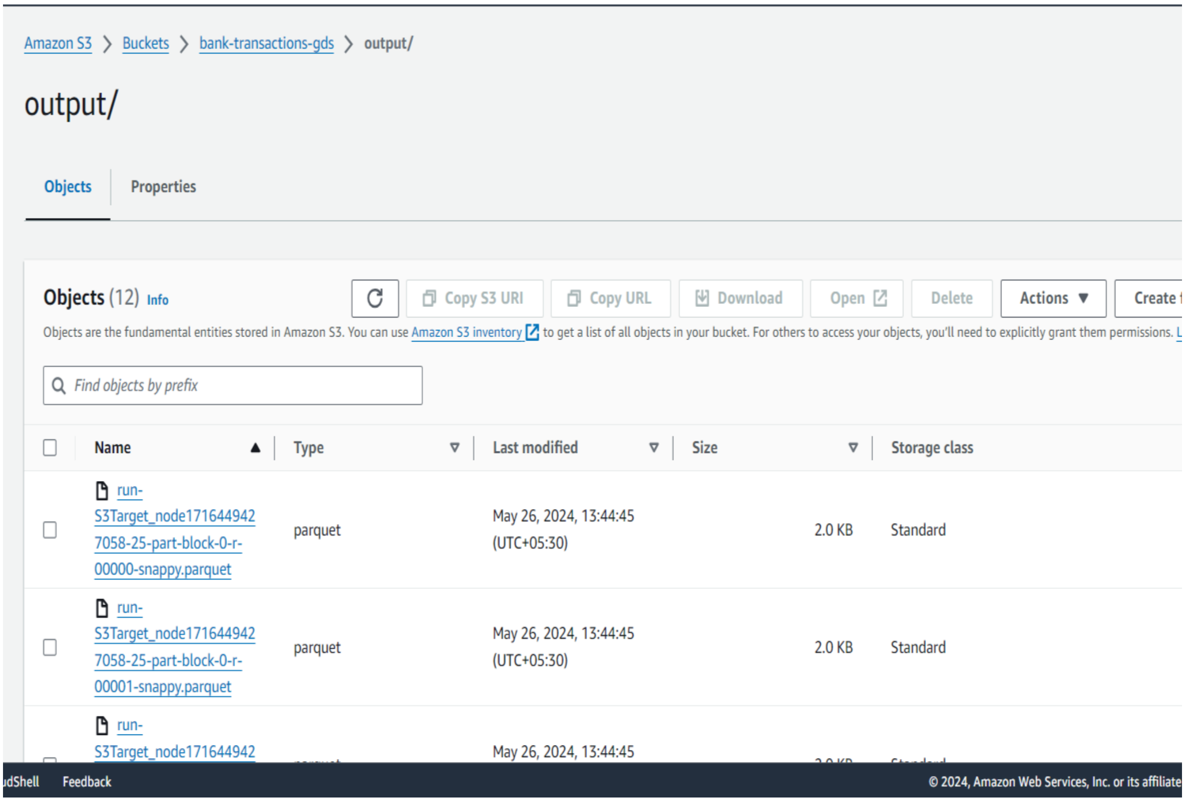
SQL query

Enter a SQL statement to add to your job.

```sql
with cte as (
select *,
row_number() over(partition by transaction_id order by tra
count(transaction_id) over(partition by transaction_id) as
from s3_bt_source
where transaction_id is not null
)
select * from cte where t_rank=1
```

transaction_id	account_id	transaction_date	amount	transaction_type
T123458	A69781	2023-09-22	250.23	debit
T123465	A14235	2023-09-20	439.63	debit

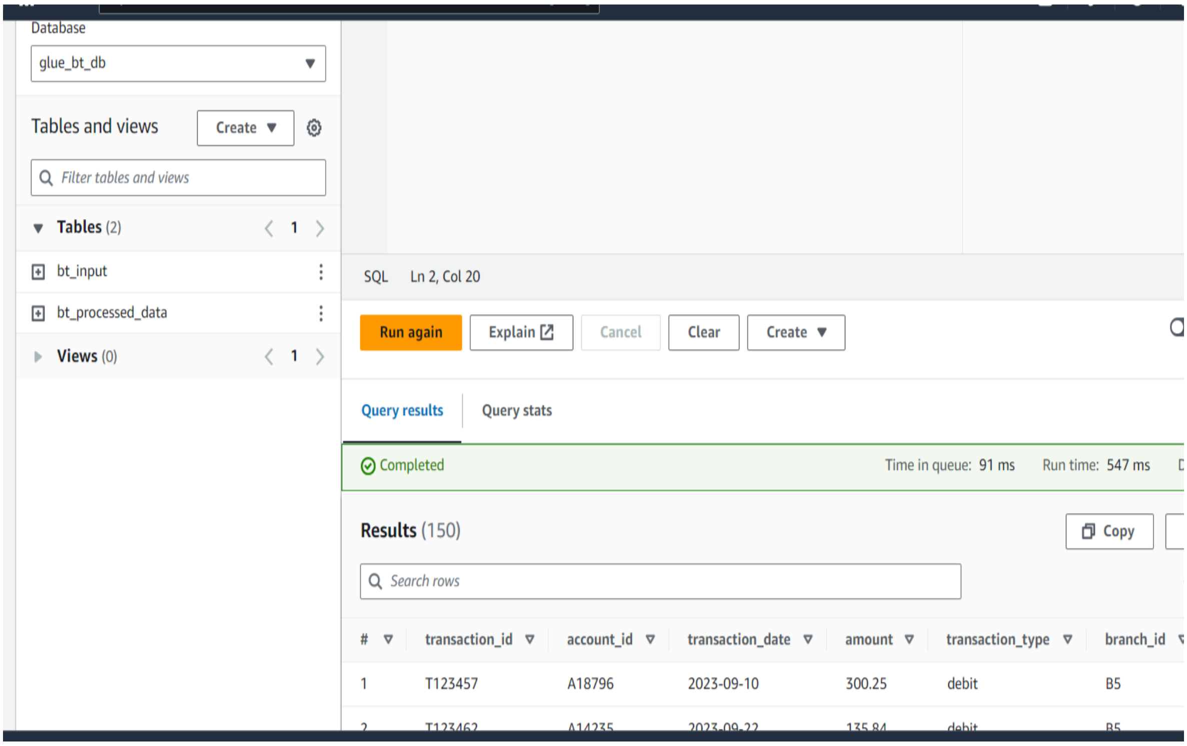
5. Further storing the transformed data back into a separate S3 bucket in parquet form. Also enabled the etl job bookmark to get only new or updated data.



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. L

□	Name ▲	Type	Last modified	Size	Storage class
□	run-S3Target_node1716449427058-25-part-block-0-r-00000-snappy.parquet	parquet	May 26, 2024, 13:44:45 (UTC+05:30)	2.0 KB	Standard
□	run-S3Target_node1716449427058-25-part-block-0-r-00001-snappy.parquet	parquet	May 26, 2024, 13:44:45 (UTC+05:30)	2.0 KB	Standard
□	run-S3Target_node171644942		May 26, 2024, 13:44:45		

6. At last I set up AWS Athena to set up a table and query on the top of transformed data stored in S3.

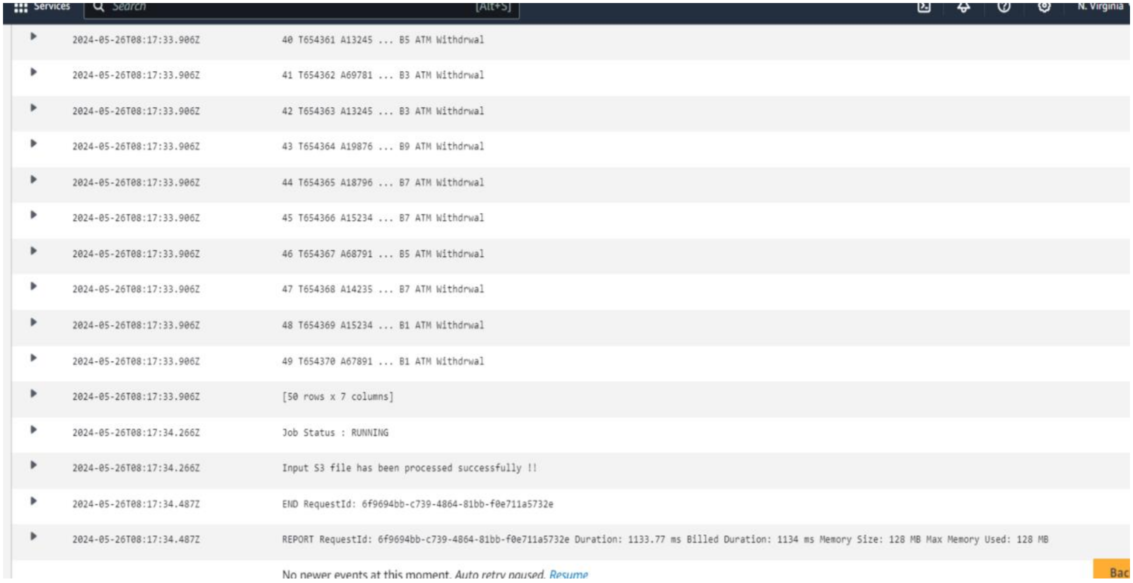


Completed Time in queue: 91 ms Run time: 547 ms

Results (150)

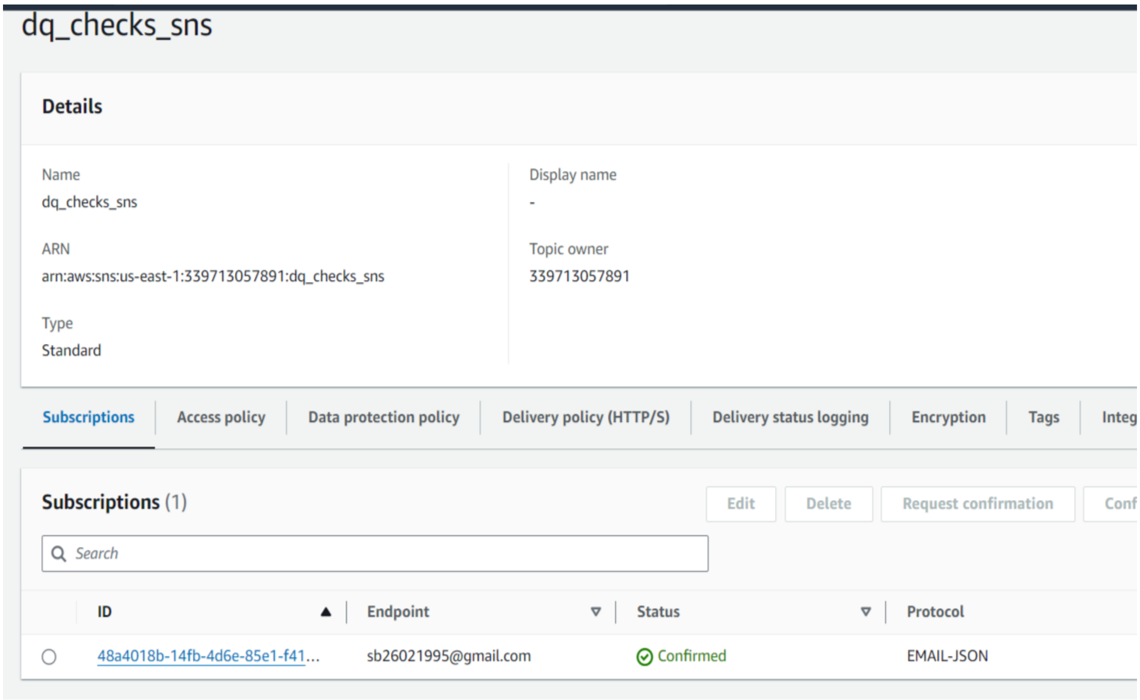
#	transaction_id	account_id	transaction_date	amount	transaction_type	branch_id
1	T123457	A18796	2023-09-10	300.25	debit	B5
2	T123462	A14235	2023-09-22	135.84	debit	B5

7. Used aws cloudWatch to monitor the data processing tasks



```
2024-05-26T08:17:33.906Z    40 T654361 A13245 ... B5 ATM Withdrwal
2024-05-26T08:17:33.906Z    41 T654362 A69781 ... B3 ATM Withdrwal
2024-05-26T08:17:33.906Z    42 T654363 A13245 ... B3 ATM Withdrwal
2024-05-26T08:17:33.906Z    43 T654364 A19876 ... B9 ATM Withdrwal
2024-05-26T08:17:33.906Z    44 T654365 A18796 ... B7 ATM Withdrwal
2024-05-26T08:17:33.906Z    45 T654366 A15234 ... B7 ATM Withdrwal
2024-05-26T08:17:33.906Z    46 T654367 A68791 ... B5 ATM Withdrwal
2024-05-26T08:17:33.906Z    47 T654368 A14235 ... B7 ATM Withdrwal
2024-05-26T08:17:33.906Z    48 T654369 A15234 ... B1 ATM Withdrwal
2024-05-26T08:17:33.906Z    49 T654370 A67891 ... B1 ATM Withdrwal
2024-05-26T08:17:33.906Z    [50 rows x 7 columns]
2024-05-26T08:17:34.266Z    Job Status : RUNNING
2024-05-26T08:17:34.266Z    Input S3 file has been processed successfully !!
2024-05-26T08:17:34.487Z    END RequestId: 6f9694bb-c739-4864-81bb-f0e711a5732e
2024-05-26T08:17:34.487Z    REPORT RequestId: 6f9694bb-c739-4864-81bb-f0e711a5732e Duration: 1133.77 ms Billed Duration: 1134 ms Memory Size: 128 MB Max Memory Used: 128 MB
                            No newer events at this moment. Auto retry paused. Resume
```

8. set up SNS notifications for any failures in the pipeline or if any data quality checks fail



dq_checks_sns

Details

Name
dq_checks_sns

Display name
-

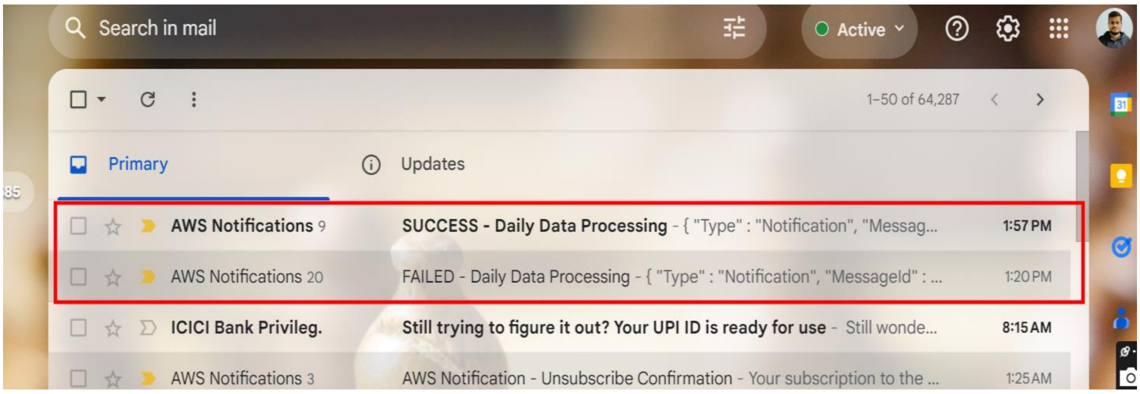
ARN
arn:aws:sns:us-east-1:339713057891:dq_checks_sns

Topic owner
339713057891

Type
Standard

Subscriptions (1)

	ID ▲	Endpoint ▽	Status ▽	Protocol
○	48a4018b-14fb-4d6e-85e1-f41...	sb26021995@gmail.com	✓ Confirmed	EMAIL-JSON



AWS Notifications 9	SUCCESS - Daily Data Processing - { "Type" : "Notification", "Messag...	1:57 PM
AWS Notifications 20	FAILED - Daily Data Processing - { "Type" : "Notification", "MessageId" : ...	1:20 PM
ICICI Bank Privileg.	Still trying to figure it out? Your UPI ID is ready for use - Still wonde...	8:15 AM
AWS Notifications 3	AWS Notification - Unsubscribe Confirmation - Your subscription to the ...	1:25 AM