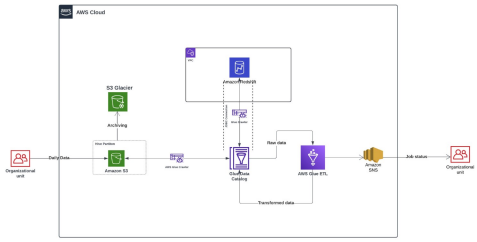
**PS/Objective:**
Implement an end-to-end, event-driven ETL (Extract, Transform, Load) pipeline on AWS to manage the daily processing of ecommerce transactional data. The solution should automate data extraction, transformation, validation, loading into a Redshift data warehouse, and archiving of outdated data, with process updates communicated via SNS. Additionally, establish a JDBC connection for data extraction or loading processes where necessary.

Develop an automated, event-driven ETL pipeline on AWS to handle daily ecommerce transaction data stored in an S3 bucket with Hive partitions (partitioned by day). The pipeline should filter out customers and products not existing in the current table, transform and load the data into a transactions table, notify stakeholders upon successful data load, and archive data older than a week in S3.



**Step 1: Create a Redshift cluster with the Following Parameters**

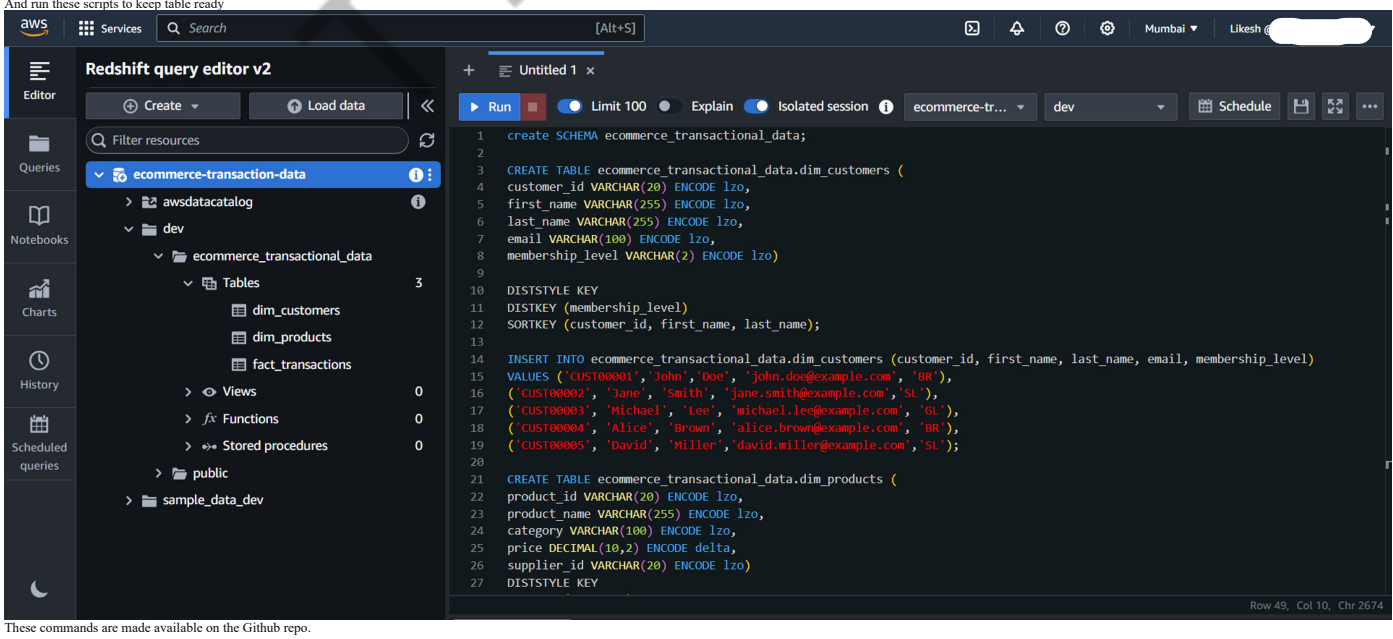
1. Select Redshift service.
2. Click on "Create cluster"
3. Give cluster a name
4. Choose the size of the cluster -> I'll choose
5. Node type -> dc2.large(cuz free trial)
6. Number of nodes -> 2
7. Admin user name -> Give a username
8. Check "Manually add the admin password" -> Add a password
9. Create a IAM role with following permissions
   a.

| Policy name | Type | Attached entities |
|---|---|---|
| AmazonRedshift-CommandsAccessPolicy-202406... | Customer managed | 1 |
| AmazonRedshiftAllCommandsFullAccess | AWS managed | 3 |
| AmazonRedshiftFullAccess | AWS managed | 7 |
| AmazonS3FullAccess | AWS managed | 19 |
| AWSGlueConsoleFullAccess | AWS managed | 5 |

10. Click on "Create Cluster"

**Step2: Prep the Data warehouse**

1. Create and Load customers, products table.
2. Create fact_transactions table.

1. Once the cluster is created
2. Select the cluster
3. Open Query editor v2
4. Log in using the above credentials
5. And run these scripts to keep table ready
6. 

7. These commands are made available on the Github repo.

**Step3: Configuring Security Group for Access**

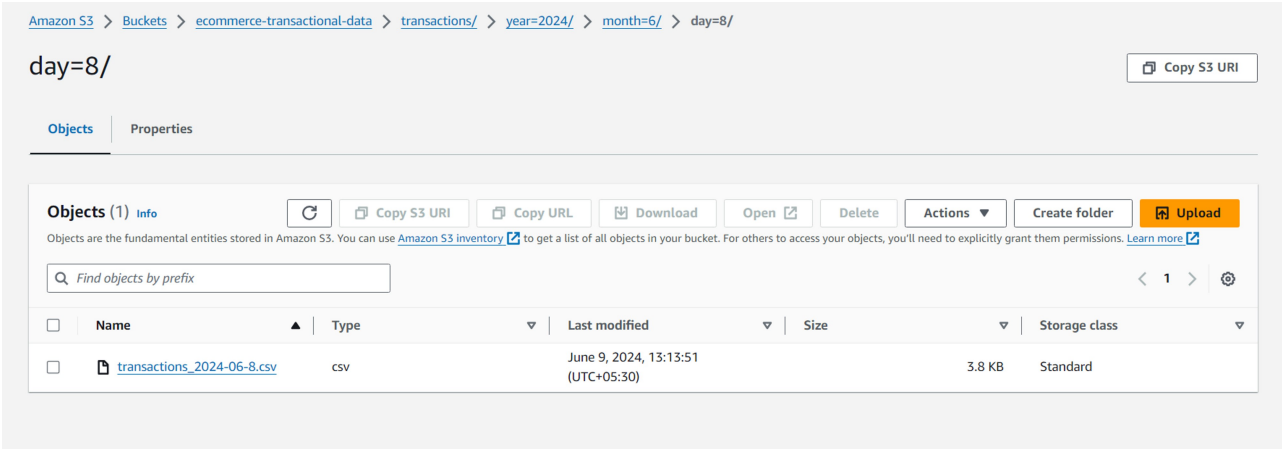
We need to open the port on which the database service is running to allow inbound traffic.
Our Redshift is running in a VPC, which has a security group. Within this security group, we need to allow inbound traffic on port 3306, the port on which the datawarehouse is running.

1. Under the Connectivity & security section of the database instance, click on the VPC link.
2. Click on the security group associated with the VPC.
3. Click Edit inbound rules.
4. Click Add rule.
5. For Type, select Redshift.
6. For Port range, enter 3306.

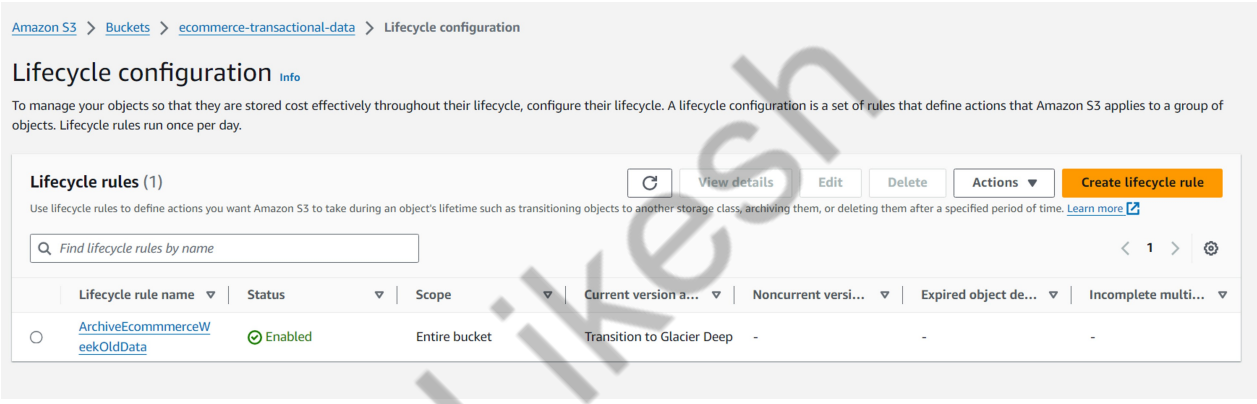
7. For Source, enter 0.0.0.0/0 (to allow any IP).
8. Click Save rules.

**Step4: Setup the Data Lake - create an S3 Bucket for Raw Data and a lifecycle rule to archive week old data**

Create an S3 bucket where the client can upload CSV data in Hive partitions (using the date as the partition key).

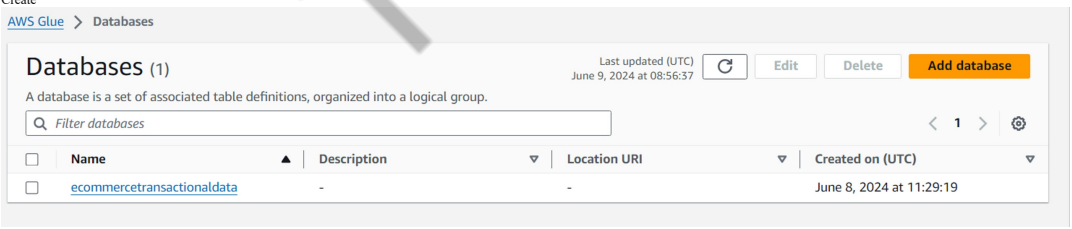



To create life cycle rules
1. Select the s3 bucket
2. Go to management tab
3. Click on LifeCycle rules
4. Give a name
5. Choose a rule scope -> Apply to all objects in the bucket
6. Check -> Move current versions of objects between storage classes option
7. Choose storage class transitions -> Glacier Deep Archive
8. Days -> 7
9. Save
10.




**Step5: Create a database in AWS Glue for the data catalog.**

1. Select aws Glue
2. Select Databases
3. Add database
4. Give it a name
5. Create
6.




**Step6: Create a JDBC connection to the RedShift.**

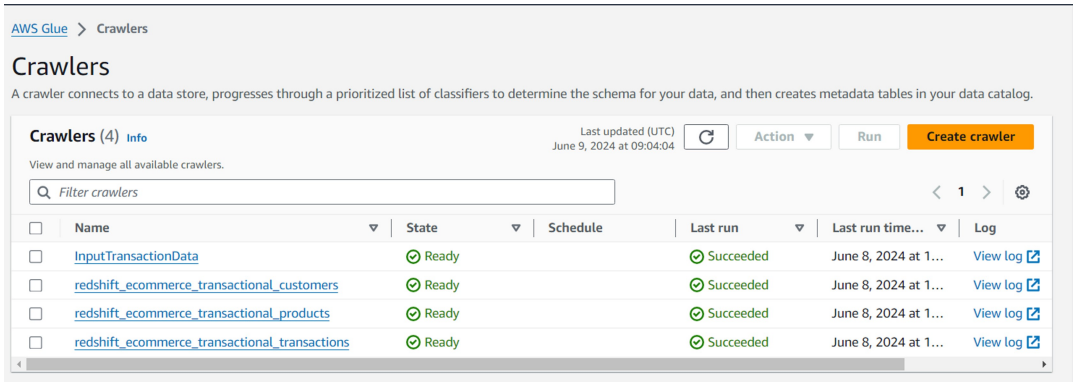
This connection will be utilized in our crawler to access the redshift tables.

To create the connection:
- Navigate to "Connections" under Data Catalog.
- Choose "Create connection."
- Select "Redshift"
- Choose the Redshift instance for the connection.
- Specify the database you want to crawl in the RDS.
- Input Redshift credentials.
- Provide a name and create the connection.

After creating the connection, it's essential to test it:
- Select the created connection.
- Under actions, click "Test connection."
- Grant permission to connect to Redshift by selecting the previous IAM role and adding the Redshift policy to it.
- Click "Confirm." If the connection is successful, you will receive confirmation.

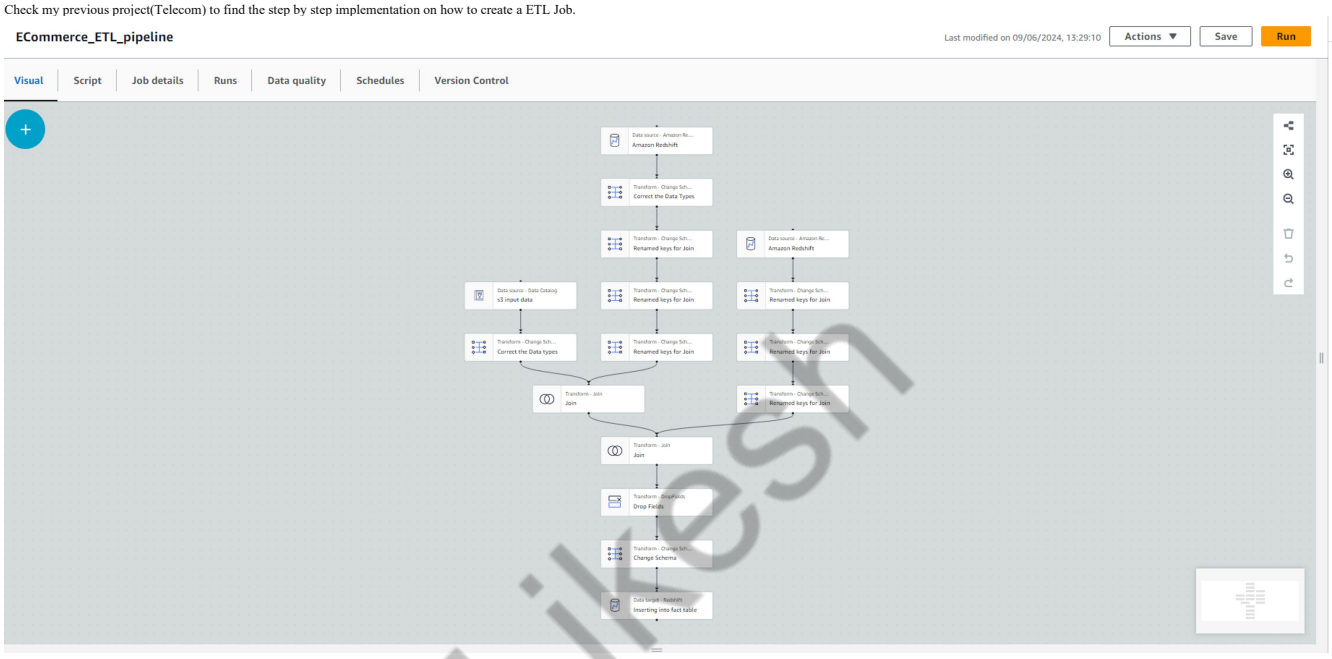
**Step6: Create crawlers**

Since, we will be applying our business logic with 4 data pieces we need to create 4 data tables on the glue catalog.
Please reffer to my previous project to check how to create crawlers on S3 and JDBC connections.

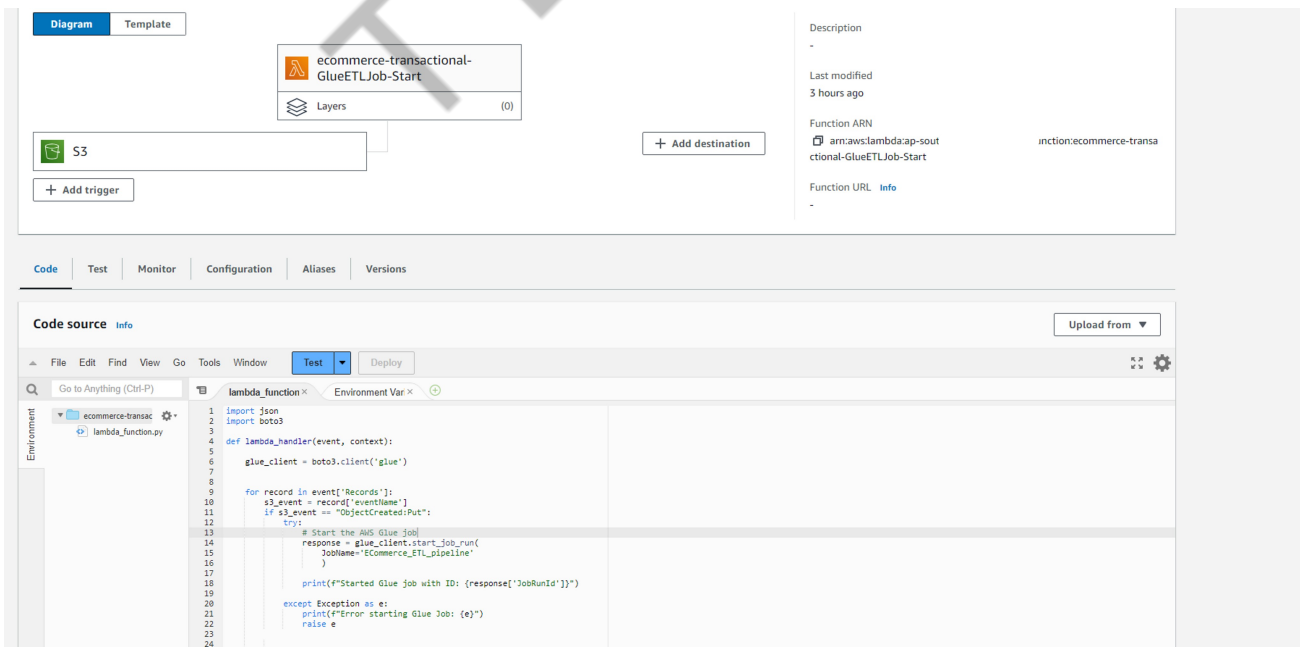



**Step7: Create a ETL job in Glue**

Check my previous project(Telecom) to find the step by step implementation on how to create a ETL Job.

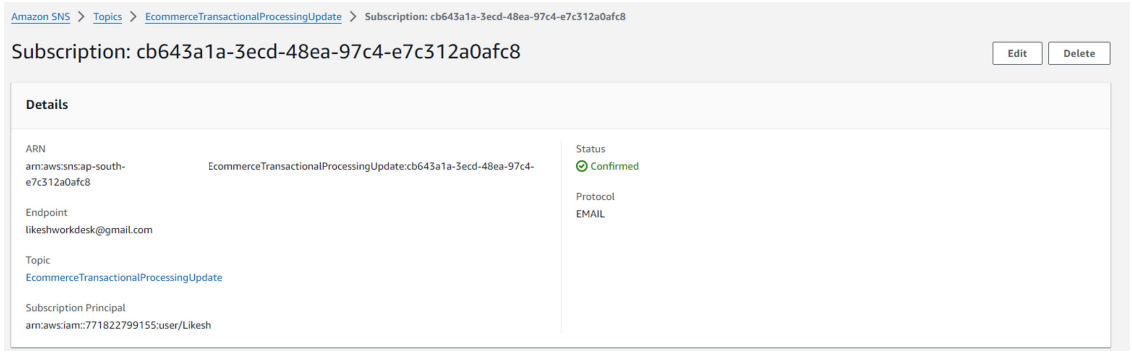



**Step8: Create a Lambda function to invoke the Job upon data arrival on s3**

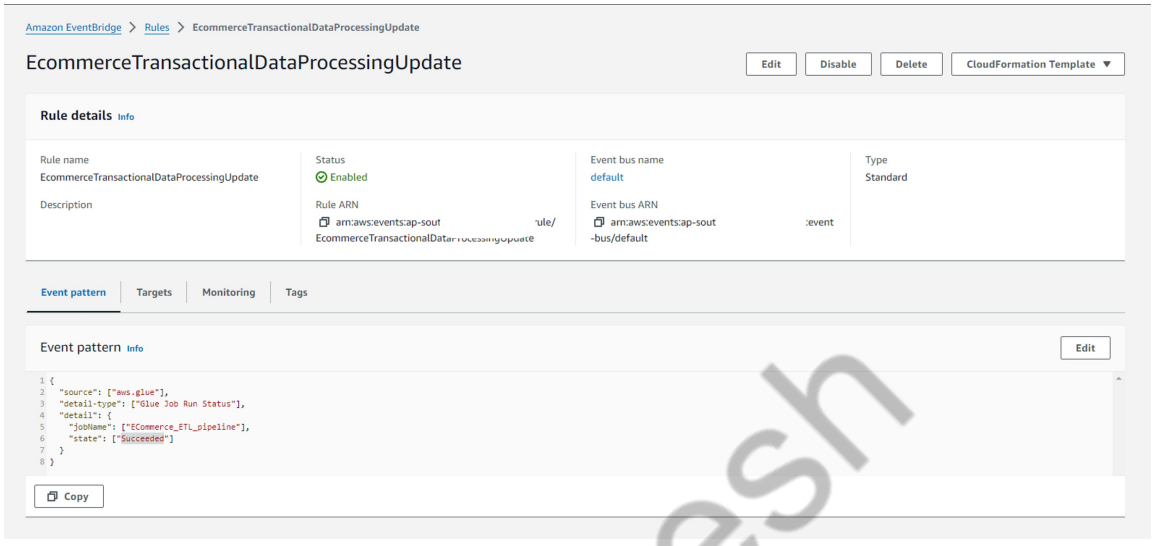



**Step9: Create a SNS to update the stakeholder via Email**

1. Select SNS service
2. Create a topic
3. Choose standard type
4. Give it a name
5. Click on create
6. Choose Email protocol and add subscriptions.



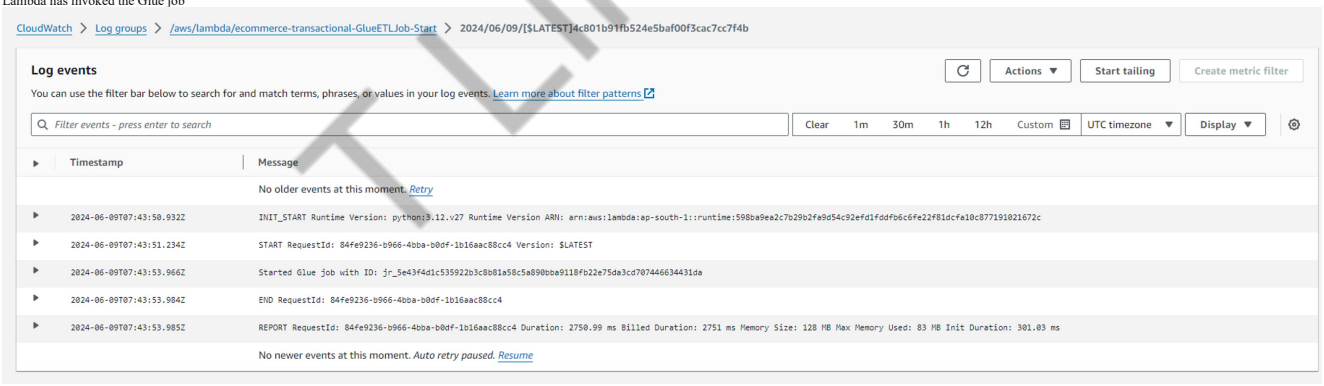

**Step10: Create Event Bridge Rule to trigger SNS when Glue Job Run is Succeeded**

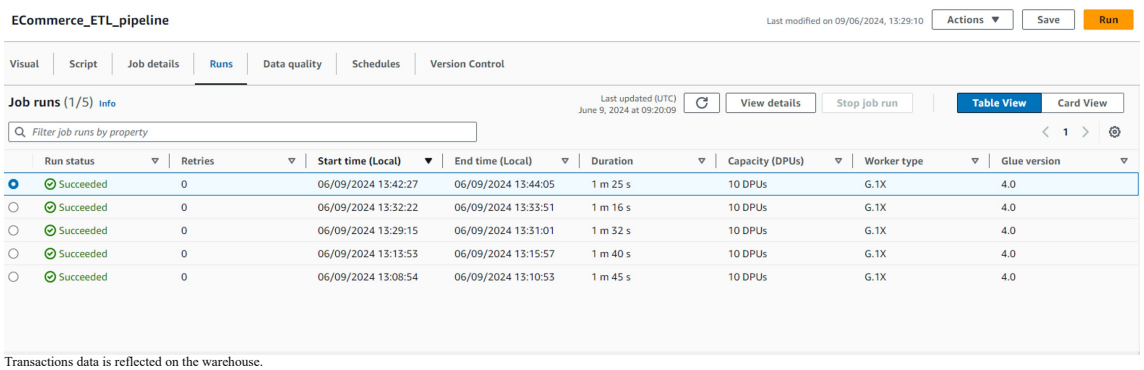





**Execute the pipeline now,**

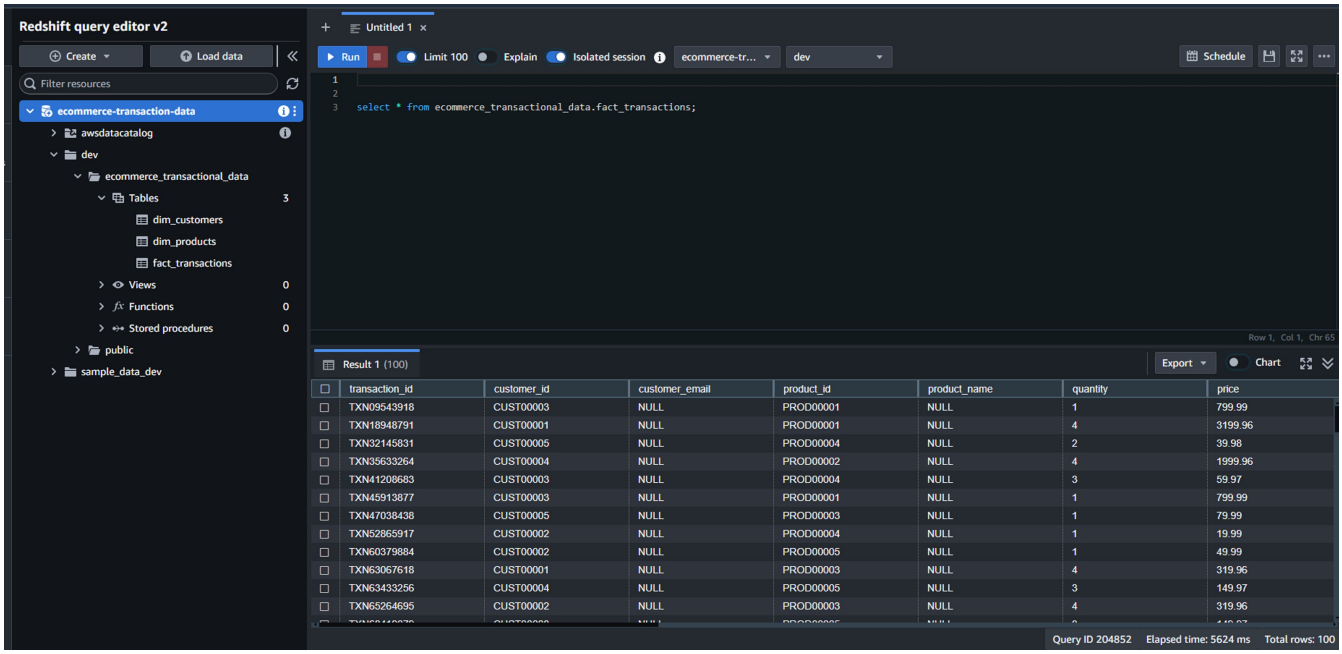
Upload the csv in S3 and check the logs

1. Lambda has invoked the Glue job
2. 

3. Glue Job has been successfully executed.
4. 

5. Transactions data is reflected on the warehouse.

6.


Redshift query editor v2
Create
Load data
Filter resources
ecommerce-transaction-data
awsdatacatalog
dev
ecommerce_transactional_data
Tables 3
dim_customers
dim_products
fact_transactions
Views 0
Functions 0
Stored procedures 0
public
sample_data_dev
Untitled 1
Run
Limit 100
Explain
Isolated session
ecommerce-tr...
dev
Schedule
1
2
3
select * from ecommerce_transactional_data.fact_transactions;
Row 1, Col 1, Chr 65
Result 1 (100)
Export
Chart
transaction_id customer_id customer_email product_id product_name quantity price
TXN09543918 CUST00003 NULL PROD00001 NULL 1 799.99
TXN18948791 CUST00001 NULL PROD00001 NULL 4 3199.96
TXN32145831 CUST00005 NULL PROD00004 NULL 2 39.98
TXN35633264 CUST00004 NULL PROD00002 NULL 4 1999.96
TXN41208683 CUST00003 NULL PROD00004 NULL 3 59.97
TXN45913877 CUST00003 NULL PROD00001 NULL 1 799.99
TXN47038438 CUST00005 NULL PROD00003 NULL 1 79.99
TXN52865917 CUST00002 NULL PROD00004 NULL 1 19.99
TXN60379884 CUST00002 NULL PROD00005 NULL 1 49.99
TXN63067618 CUST00001 NULL PROD00003 NULL 4 319.96
TXN63433256 CUST00004 NULL PROD00005 NULL 3 149.97
TXN65264695 CUST00002 NULL PROD00003 NULL 4 319.96
Query ID 204852 Elapsed time: 5624 ms Total rows: 100