**USECASE 1**

Diagram

Description automatically generated

**Needed AWS Services:**

S3

IAM

EC2

Glue

DynamoDB

Redshift

Data Pipelines

**Create Necessary IAM Roles:**

1. For Lambda – create a role and attach ‘AdministratorAccess’ policy.
2. For Redshift – create a role and attach ‘[AmazonS3FullAccess](https://console.aws.amazon.com/iam/home?region=us-east-1#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonS3FullAccess)’, ‘[AmazonAthenaFullAccess](https://console.aws.amazon.com/iam/home?region=us-east-1" \l "/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonAthenaFullAccess)’, ‘AWSGlueConsoleFullAccess’ policies.
3. For Glue- create a role and attach ‘[AmazonS3FullAccess](https://console.aws.amazon.com/iam/home?region=us-east-1#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonS3FullAccess)’, ‘AdministratorAccess’, ‘AWSGlueConsoleFullAccess’ policies.

**Before begin,**

1. Launch t2.micro (free tier) EC2 instance and install python3 and necessary libraries.
2. Make sure to create all the services within a region that supports all the AWS services.
3. **Upload raw data files to S3 (bucket name: input data)**

Navigate to S3 and create a bucket to upload input data. Upload all the raw data files into the bucket.

1. **Upload Address data to dynamo db from S3 (‘input data’)**
2. Navigate to Dynamo DB and create a table named ‘Address’ and add ‘CustomerID’ as the Primary partition key.
3. Then go to Lambda and create a function using Python 3.7 as the runtime (Give Lambda Execution role created for the permission section).
4. Write a python code to insert data from S3 to DynamoDB , deploy it and test. (Refer: [https://www.youtube.com/watch?v=Y18HF5ALXew](https://www.youtube.com/watch?v=Y18HF5ALXew%20))
5. **Upload customer, products, product\_category, orders data from S3 (‘input data’) to Redshift source table.**
6. Create a Redshift cluster providing a database name, master username and password.
7. Once the cluster is available, navigate to ‘Query Editor’ tab.
8. Run SQL commands to create Customer, Orders, Product, Category tables (inside public schema). Make sure to have the same column order in both S3 file and in the Redshift table.
9. Run ‘Copy’ commands to each RS. Table. It will copy data from the S3 files into the Redshift tables. (Add ‘IGNOREHEADER 1’ at the end of the command, since the input file contains headers)
10. Check whether the data has been inserted into the source tables.
11. **Unload data from dynamodb to S3 data lake (bucket name: sales data)**
12. Write a python script to decode nested json address data into csv (With 2 records per customer – AddressLine1 and AddressLine2) and store it in a new S3 bucket (‘Sales data’).
13. **Unload data from Redshift source tables to S3 data lake(‘sales data’)**
14. Write a python script to call ‘Unload’ commands and store the data in CSV format. Create another 2 files to store SQL commands and Redshift/Dynamodb credentials separately.
15. **Copy product, product\_category from S3 data lake (‘sales data’) to Redshift staging tables using Glue**
16. Create a new schema ‘Sales’ in Redshift and create stg\_customer, stg\_orders, stg\_products, stg\_category, stg\_address tables.
17. Navigate to Glue and create a database. Then create a connection for Redshift and test it.
18. Create a crawler (‘fromS3Crawler’) to get the metadata **from the ‘products.csv’** in S3 data lake.
19. Run the crawler and check whether it has created a table.
20. Create another crawler (‘toRSCrawler’) to get the metadata **from the ‘stg\_products’** table in the Redshift Target schema and run it as well.
21. Once the tables are created, navigate to ‘Jobs’ under ETL section. Create a new job and choose the table created using ‘fromS3Crawler’ as the data source and the other as the data target.
22. Finally save the job and edit the script as needed and run it.

Refer for all the steps : <https://www.youtube.com/watch?v=B6Wz9A_rZ1Q> and

<https://www.youtube.com/watch?v=luKULJIl4qw&t=1071s>

1. **Copy customer, orders, address data from S3 data lake (‘sales data’) to Redshift staging tables**
2. Write a python script to call ‘Copy’ commands. (Keep the SQL commands and the Redshift connection string in the SQL file and the credentials file respectively)