**AWS Glue Homework**

This report summarizes the AWS Glue data analysis tasks completed using MSK Streaming and Titanic database. It includes step-by-step processes, code snippets, analytical insights, and reflections on challenges encountered.

**Part 1. Setting up an MSK Cluster and Producer Configuration**

**Step-by-step Process and Code Snippets**

1. **AWS MSK CloudFormation Template**

Build up a file named HW1-AWS\_MSK.yaml with Security Groups, MSK Configuration, CloudWatch Group, MSK Cluster, with public access permission.

1. **Deploy Steps**

Upload the created HW1-AWS\_MSK.yaml template to AWS CloudFormation to create the resources. Wait until the resources are correctly created. Then, direct to the MSK cluster to open the public access and acquire the public access bootstraps.

A screenshot of a computer

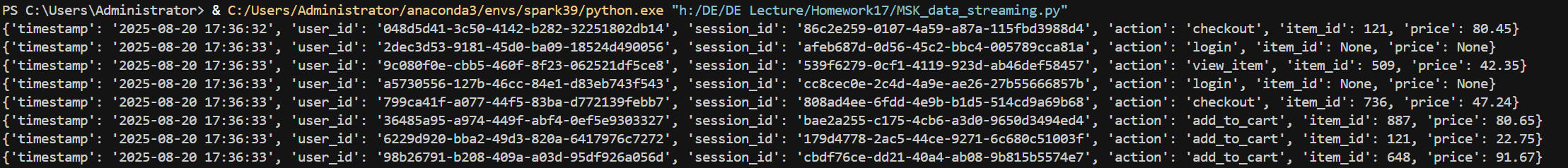
AI-generated content may be incorrect.

1. **Data Streaming Application**

Create an MSK topic named Part1 (MSK\_Topic.py) using the bootstraps acquired in the previous step with IAM authentication.

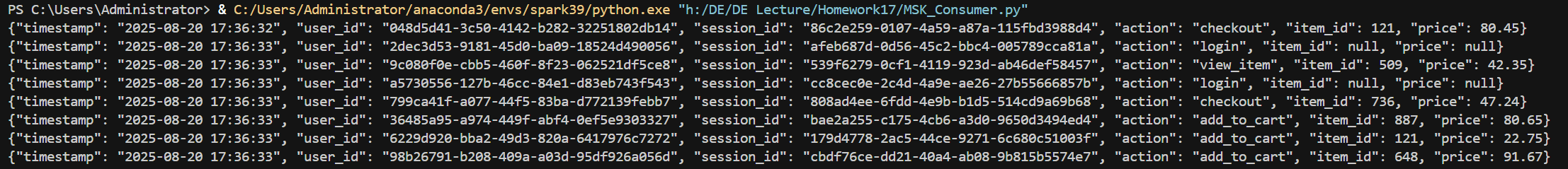


Then, create an MSK data streaming python file. The MSK data streaming script includes a Kafka producer, which can generate fake records and send them to the previously created topic.



1. **Data Streaming Verification and Monitoring**

To verify whether the data streamed into the MSK topic, an MSK customer script is created and implemented. If the script runs with the produced MSK data, it signifies that the data is streamed successfully to the topic.



Additionally, CloudWatch metrics can be applied to further verify whether it flows to the MSK topic. BytesInPerSec is a significant metric to confirm the data flow.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Glue Streaming ETL**

**5.1 S3 Bucket and ETL IAM Role**

To begin with the ETL Streaming job, creating an S3 bucket named hw17-msk-part1 serves as the destination of the data.

Then, an IAM pass role policy (HW1-AWS\_IAM-role.yaml) should be created to ensure the successful execution of the ETL script. Besides, the policy enables the creation of the log group and the transformation to the S3 bucket.

Upload the created IAM pass role policy file.

* 1. **ETL Job Creation and Execution**

To begin with the ETL job, we set up a connection between AWS Glue and AWS MSK. Then, open the AWS Glue Studio to create a new ETL job named HW17-ETL-MSK. In this script, transformation is done to store the data from MSK topic and filter out the Null values in price column.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Data Storage**

After the transformation is done, the cleaned data is stored in the S3 bucket created in part 5.1.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Trouble Shooting**

7.1 Public Access

The first question comes from the MSK public access issue. If public access is denied, the MSK python client cannot be connected to the Personal Computer.

Solution:

In the configuration yaml file, set up proper configurations to grant public access. In this scenario, we will only enable IAM and SCRAM authentication to ensure public access. Then, turn on public access in the AWS MSK console.

Code Configuration Snippets:

ClientAuthentication:

        Unauthenticated:

          Enabled: false

        Sasl:

          Iam:

            Enabled: true

          Scram:

            Enabled: true

* 1. Data Acquisition in ETL

When trying to use Infer Schema function, the job cannot be executed as it cannot detect the columns in the streaming dataset.

Solution:

When performing the ETL job using the streaming service, it will generate a temporary column to store all the information of the dataset. In this situation, extract the information from the column based on the known schema, and the streaming data can be acquired and stored in the S3 bucket.

**Part 2. Data Transformation and Analysis with the Titanic Dataset**

**Step-by-step Process and Code Snippets**

1. **Download the dataset**

Visit the Titanic dataset on Kaggle and download it. Then, upload the dataset to the created S3 bucket named hw17-part2.

The link: <https://www.kaggle.com/datasets/yasserh/titanic-dataset>

1. **Glue Studio Job**

To perform the glue studio ETL job, the primary task is to establish the IAM policy accordingly based on the policy template in Part1. Then, deploy the template and begin the ETL job.

In the Glue Studio, an ETL job script (HW2-ETL-Clean) is created to clean and transform the data. The null value in the age column is filtered out, and a new column, family size, is added, which is calculated based on the sum of the sibling numbers and parent numbers. After the data transformation job is completed, write the data in parquet format to the S3 bucket (hw17-part2) in the processed\_data file for further analysis.

Code Snippet:

*# get the dataset from the S3 bucket*

df = spark.read.option("header", "true").option("inferSchema", "true").csv(S3\_INPUT)

*# perform dataset clean and transformations*

df = df.select("PassengerID", "Survived", "Pclass", "Name", "Sex", "Age", "SibSp", "Parch", "Fare", "Embarked")

df = df.withColumn("family\_size", f.col("SibSp") + f.col("Parch") + 1).filter(f.col("age").isNotNull())

df.show()

*# write the data to the S3 bucket*

df.write.mode("append").parquet(S3\_OUTPUT)

1. **Data Analysis in Glue Interactive Session**

In this section, four analyses are conducted focusing on survival rates across different dimensions: age groups, passenger classes, gender, and family size. The data analysis code can be found in HW17-DA-Part2.ipynb file.

The first analysis is the survival rate in different age groups. The dataset is divided into different age groups and the survival rate in each age group is calculated. Based on the survival rate result, it can be concluded from the result that baby, teenagers, and adults have relatively high survival rates, while elderly passengers exhibit low survival rates.

Age group classification code snippet:

A computer screen with white text

AI-generated content may be incorrect.

Survival rate result:

A screenshot of a computer screen

AI-generated content may be incorrect.

The second analysis examines survival rates across different passenger classes. The results indicate that first-class passengers had the highest survival rate at 65.59%, suggesting a strong relationship between passenger class and the likelihood of survival.A screen shot of a computer

AI-generated content may be incorrect.

The third analysis examines the survival rate by gender. The results demonstrate that female has the highest survival rate at 75.48%, suggesting that female passengers were more likely to be assisted during the disaster.

A black screen with white text

AI-generated content may be incorrect.

The final analysis examines the survival rate across different family sizes. The results indicate that passengers with a family size of four have the highest survival rate at 77.78%, indicating that traveling in small family groups may increase the chance of survival.

A black screen with white text

AI-generated content may be incorrect.

1. **Data Storage**

After the data analyses are performed, the results are stored in the S3 bucket (hw17-part2), under the DA\_results directory.

A screenshot of a computer

AI-generated content may be incorrect.

**Part 3. Designing an ETL Workflow with AWS Glue Studio and Blueprints**

**Step-by-step Process and Code Snippets**

1. **Blueprint Configuration**

The first step is to set up the configuration file (blueprint.cfg) for the AWS Glue blueprint. In this scenario, the blueprint is designed to create an ETL workflow that ingests the Titanic dataset from an S3 bucket, performs data transformation, and stores the processed results back into another S3 location.

The configuration file defines all the required parameters for the blueprint, including:

WorkflowName – The name of the Glue workflow.

WorkerType and DPU – Specify the type and number of workers for the ETL job.

S3Path – The source S3 location where the raw dataset is stored.

TargetS3Path – The destination S3 location to store the transformed data.

SourceDatabase and TargetDatabase – Databases used by crawlers for schema inference and catalog updates.

PassRole – In this part, we use the PassRole defined in part2.

1. **Layout Code**

The second step is to develop the layout.py script. In this step, the source crawler, target crawler, Glue ETL job, and triggers are defined and integrated into a single workflow. The source crawler is triggered manually to extract the schema in the source dataset. The ETL job can be performed once the source crawler successfully acquires the schema of the dataset. The target crawler is triggered after the successful execution of ETL job to acquire the schema after the dataset transformation. This target crawler also helps address potential schema evolution issues.

1. **Conversion Code**

The third step is to develop the conversion.py script. In this step, the dataset is cleaned, transformed, and validated to ensure data quality. The transformation process involves selecting the required columns, adding a family\_size column based on the number of siblings and parents, and categorizing passengers into different age groups. After the transformation, a data quality check is performed. In this scenario, the DQDL (Data Quality Definition Language) is used to define a set of validation rules, including verifying the existence of required columns, checking column completeness, and ensuring that the data values fall within the expected ranges. Finally, only the records that pass all data quality checks are selected and written to the specified S3 bucket for storage.

1. **Deployment**

To create the blueprint workflow, the first step is to upload the blueprint cfg, layout.py, and conversion.py zip file into the designated S3 bucket under the blueprint\_script directory.

After uploading the zip file, the AWS Glue console is used to register the blueprint and create the workflow. The screenshot below shows that the blueprint deployed successfully, and the workflow is created.

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

1. **Workflow Execution**

After the deployment of the blueprint, the next step is to execute the generate workflow in AWS Glue.

The first step of the workflow execution is to extract the schema using source crawler. The inferred schema is stored in the AWS Glue Data Catalog. The screenshot below shows the extracted dataset schema.

A screenshot of a computer

AI-generated content may be incorrect.

After the successful execution of the crawler, the ETL job can be executed. The ETL job cleans and transforms the dataset and stores the transformed dataset in the target S3 location. The screenshot below shows the successful execution of the ETL job.

A screenshot of a computer

AI-generated content may be incorrect.

After the ETL job is completed, the target crawler runs to scan the transformed dataset and update the Data Catalog schema. This step ensures proper schema evolution handling when the structure of the dataset changes. The screenshots indicate that the crawler captures the schema evolution.

A screenshot of a computer

AI-generated content may be incorrect.

The transformed dataset is stored in the S3 bucket in parquet format.

A screenshot of a computer

AI-generated content may be incorrect.