Retail Data Management Project Report

Name: Ali Hamza Shaikh
Project Title: Retail Data Processing using AWS Glue

---

1. Project Overview

This project was developed to enhance ABC Retail's data management by using AWS Glue to automate ETL (Extract, Transform, Load) processes. The key objectives included cleaning and transforming raw data, joining datasets, extracting numerical values from text, and producing a summary of average sales by product category and shipping mode.

---

2. AWS Services Used

- **Amazon S3**: To store input and output data files.
- **AWS Glue**: For data cataloguing, transformation, and ETL job orchestration.
- **IAM**: To manage permissions for the Glue job.

---

3. Datasets Used

- **Product Details CSV**: Contains information about product categories, IDs, and descriptions.
- **Transactions CSV**: Contains sales records, including dates, quantities, discounts, profits, and shipping modes.

---

4. Steps Performed

Step 1: S3 Bucket Setup

- Created bucket: etl-cep-01-ali
- Subfolders:

    - transaction-files/ (uploaded transactions.csv)
    - product-files/ (uploaded product details.csv)
- Created output bucket: etl-cep-output-01

Step 2: Create AWS Glue Database

- Name: abc-retail

Step 3: Create Classifiers

- **cust_classifier**: For product data (CSV with headers)
- **txnClass**: For transaction data with manual headers

Step 4: Create IAM Role

- Role Name: glue-role
- Permissions: AdministratorAccess

Step 5: Create Crawlers

- Crawler for transactions:
    - Source: etl-cep-01-ali/transaction-files/
    - Classifier: txnClass
- Crawler for product details:
    - Source: etl-cep-01-ali/product-files/
    - Classifier: cust_classifier

Step 6: Create Visual ETL Job

- **Join**: Inner join on Product ID
- **Drop Fields**: Removed duplicate Product ID
- **Regex Extractor**: Extracted numeric value from Sales column using regex (\d+(\.\d+)?)
- **Aggregate**:
    - Grouped by: Product Category, Ship Mode

    - Aggregated field: Sales
    - Aggregation: Average
- **Output**: Saved as .parquet to etl-cep-output-01-ali

Step 7: Output Verification

- Used S3 Select to query .parquet output file
- Set output format to CSV

---

5. Key Learnings

- Understood the role of Glue classifiers in schema detection
- Applied regular expressions for data cleaning
- Used Visual ETL to create a no-code data pipeline
- Implemented grouping and aggregation for summary reporting

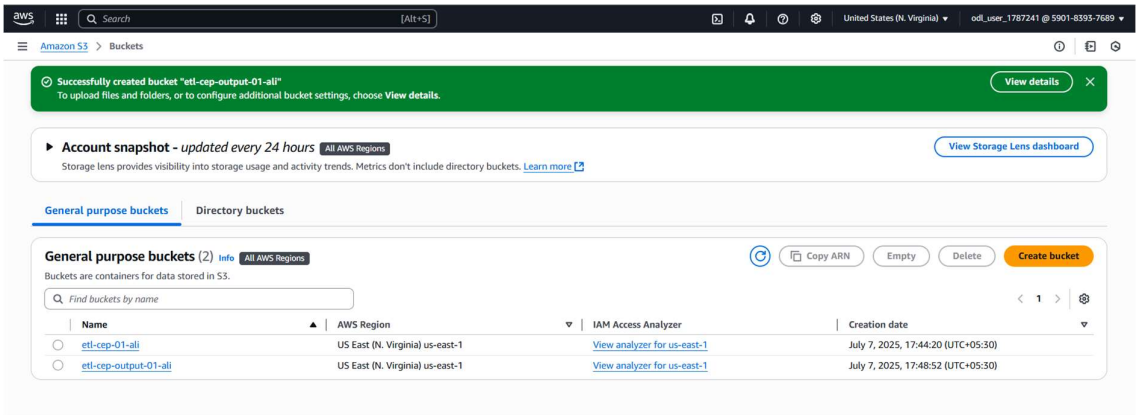
---

6. Screenshots

📸 Screenshot Checklist for AWS Glue ETL Retail Project

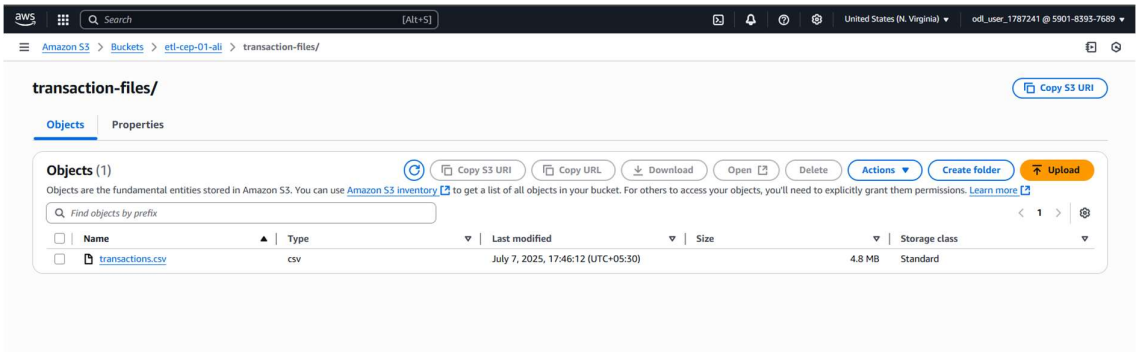
🔹 S3 Setup

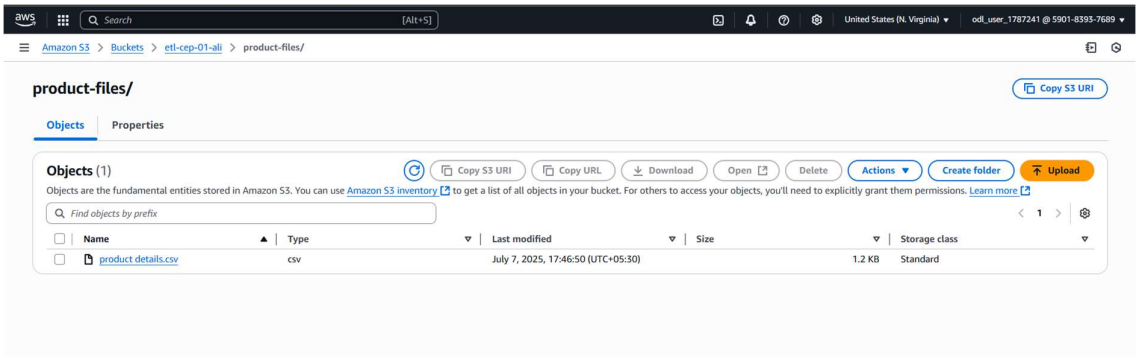
1. ✅ **S3 Bucket Created**
   Screenshot of the bucket list showing etl-cep-01-ali and etl-cep-output-01-ali.



2. ✅ **S3 Subfolders Uploaded**

Inside etl-cep-01, show product-files/ and transaction-files/ each containing their CSV file.

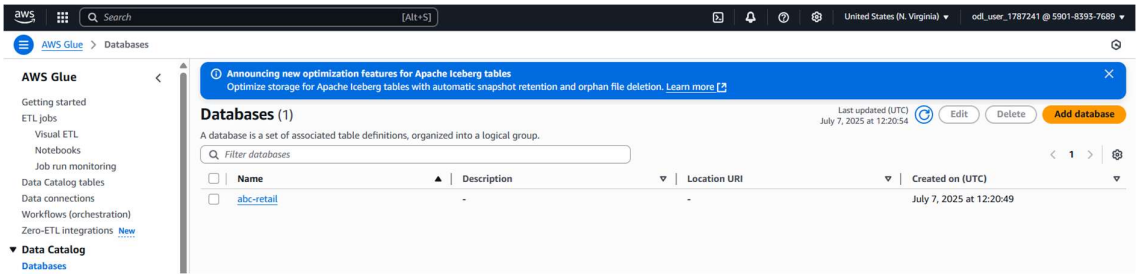




🔹 Glue Database and Classifiers

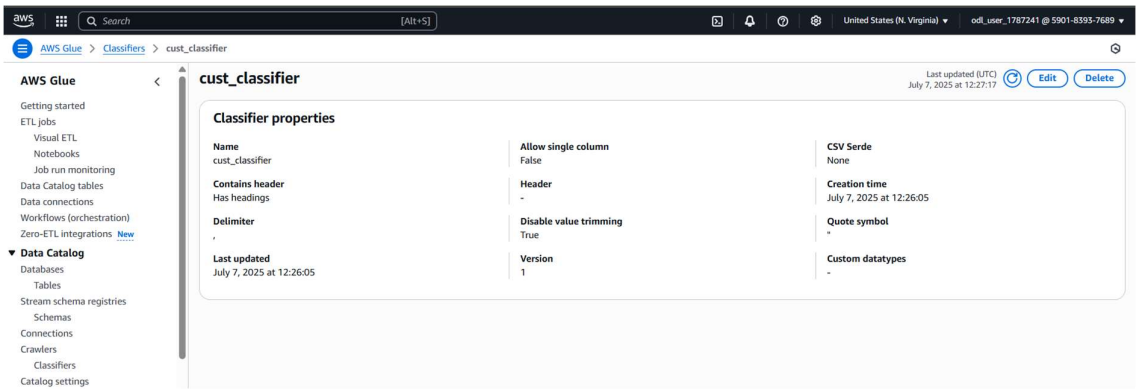
3. ✅ **Glue Database Created**

Screenshot of database named abc-retail in AWS Glue > Databases.



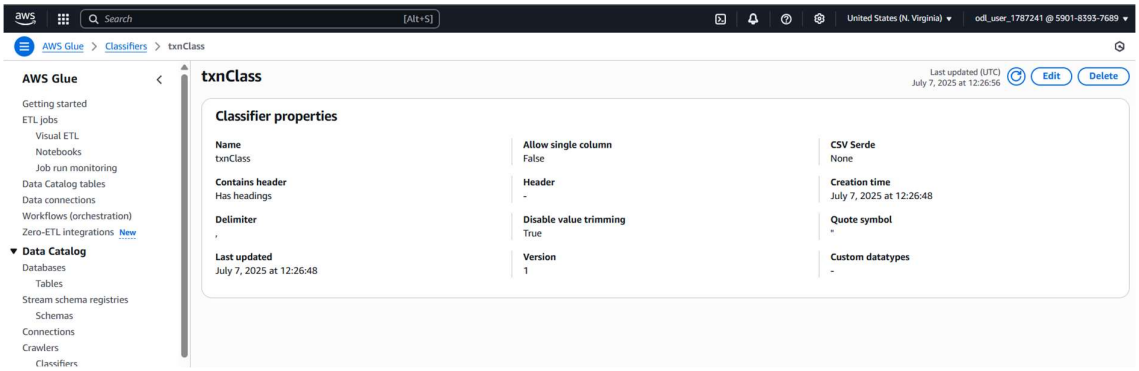
4. ✅ **Classifier: cust_classifier Created**

Screenshot of classifier for product data in AWS Glue > Classifiers.



5. ✅ **Classifier: txnClass Created**

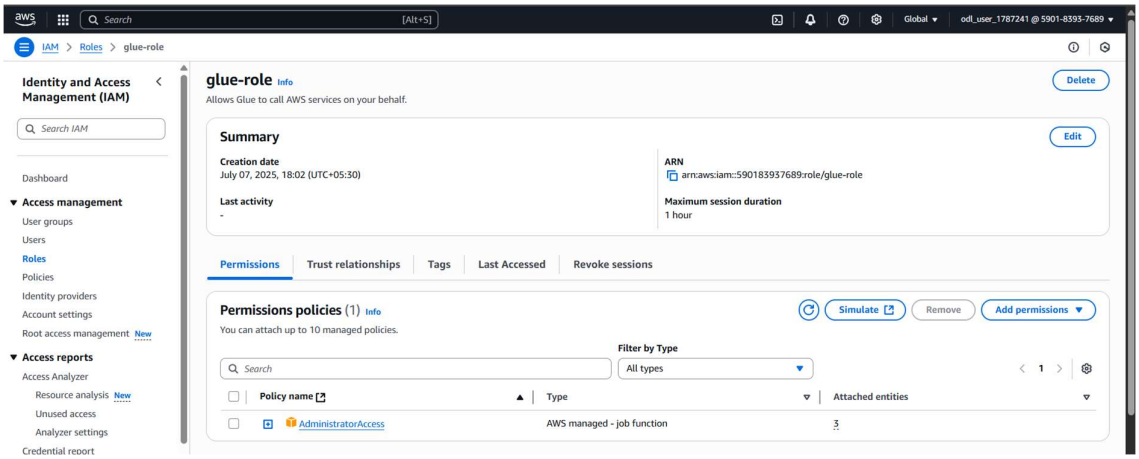
Screenshot of classifier for transaction data.



◆ IAM Role

6. ✅ IAM Role: glue-role Created

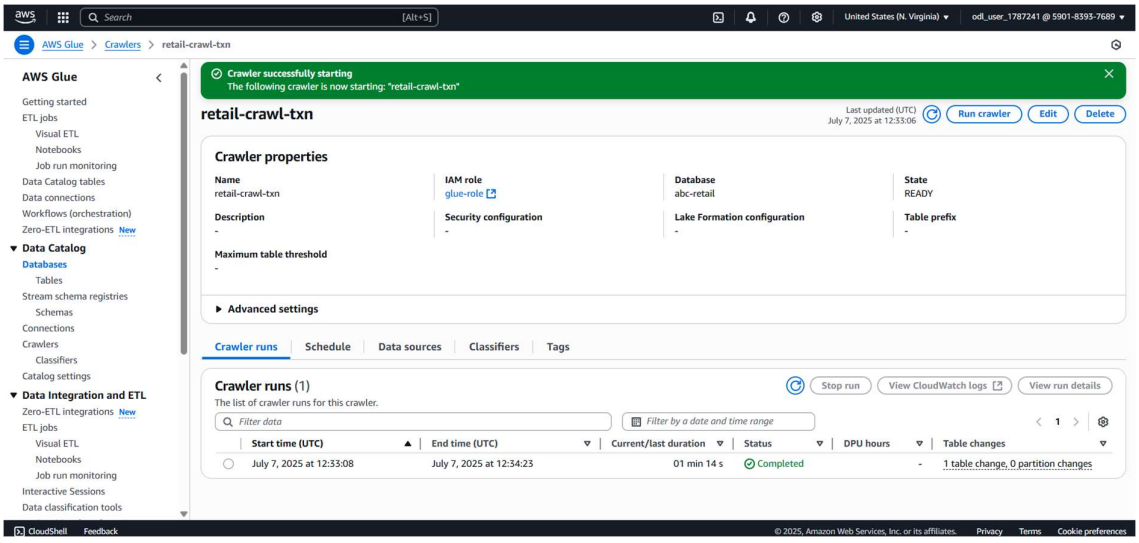
Screenshot showing the new IAM role and the AdministratorAccess policy attached.



◆ Crawlers

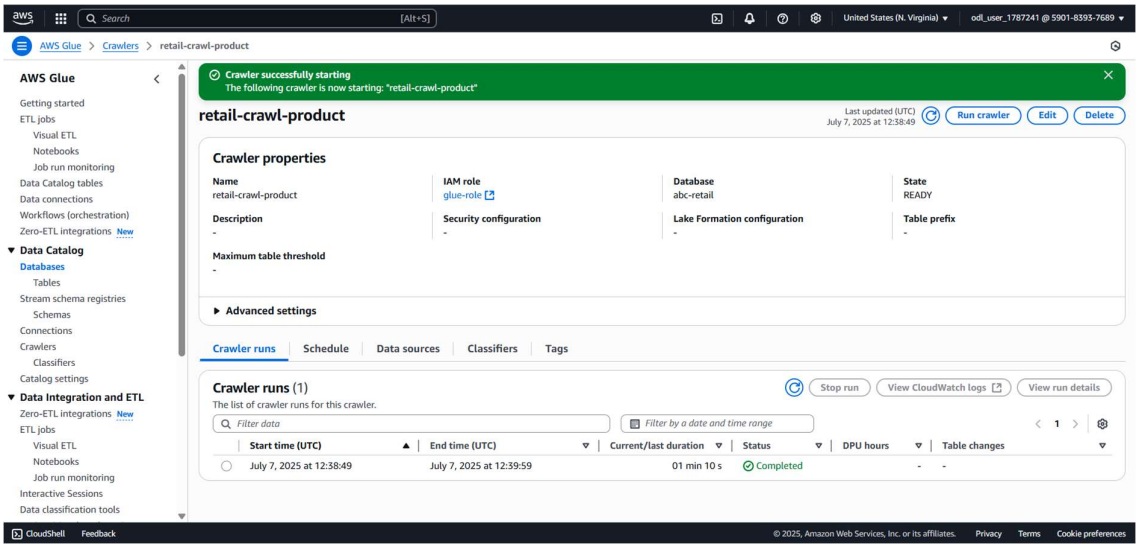
7. ✅ Crawler for Transactions Created

Screenshot of crawler settings with txnClass, transaction path, and abc-retail DB selected.



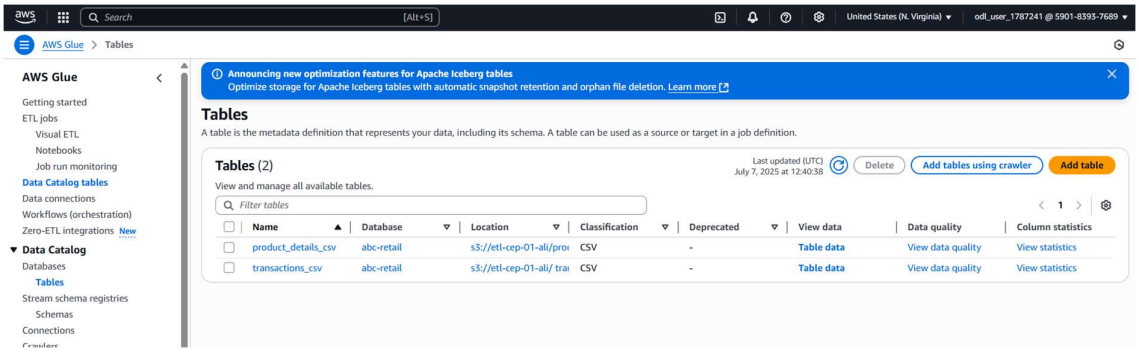
8. ✅ **Crawler for Product Details Created**

Screenshot of second crawler settings with cust_classifier, product path selected.



9. ✅ **Crawler Run Completed**

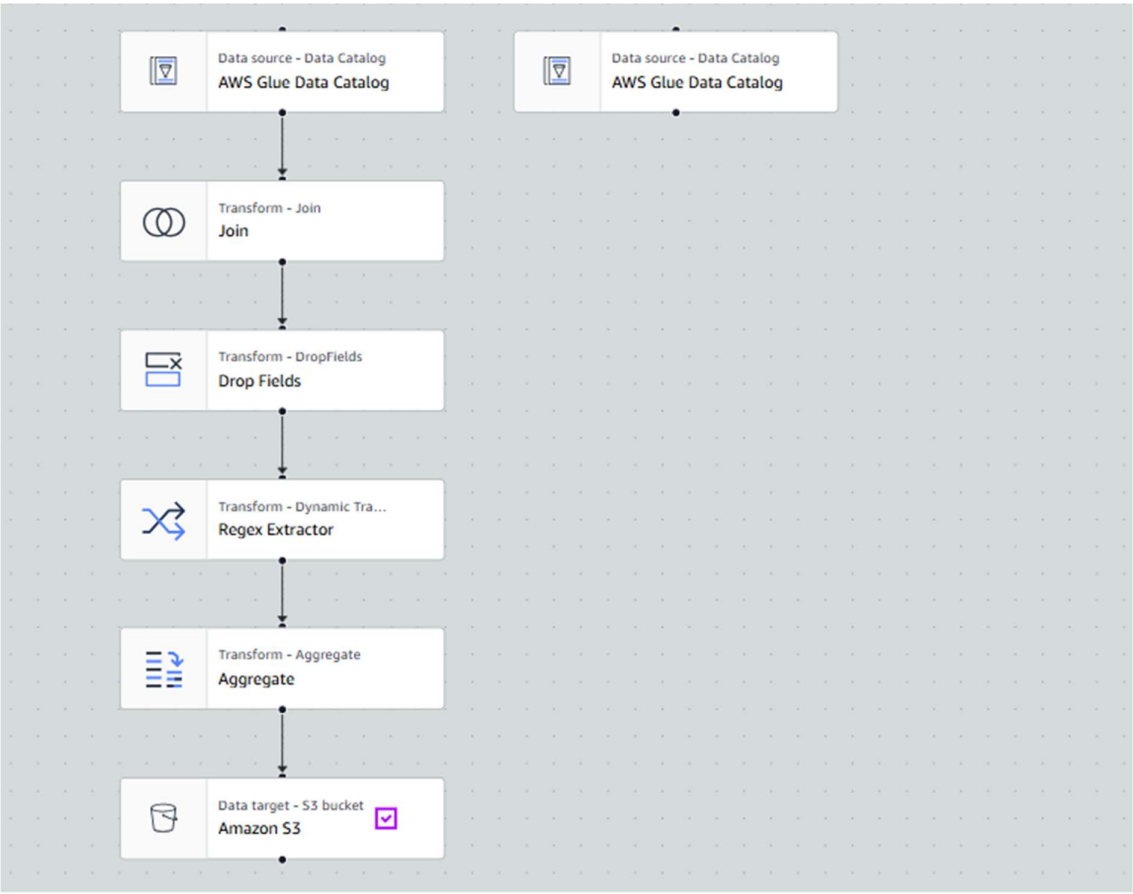
Screenshot showing success of both crawler runs and tables created in the Data Catalog.



- **ETL Visual Job**

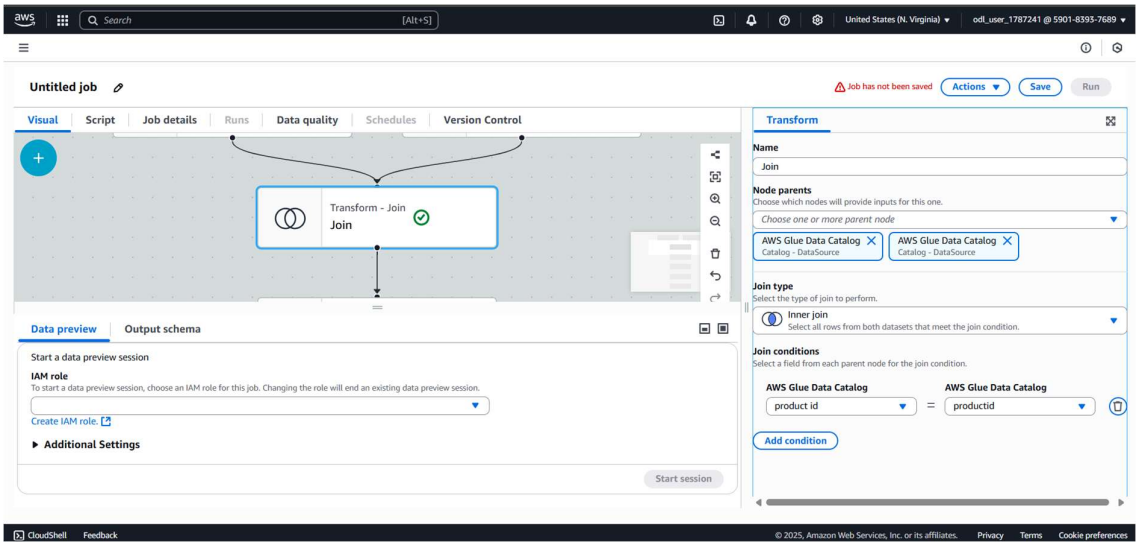
10. ✅ **Visual ETL Canvas (Before Linking)**

Screenshot of AWS Glue Visual job canvas with unlinked nodes: 2 Data Catalog nodes, Join, Drop Fields, Regex Extractor, Aggregate, and S3 Target.



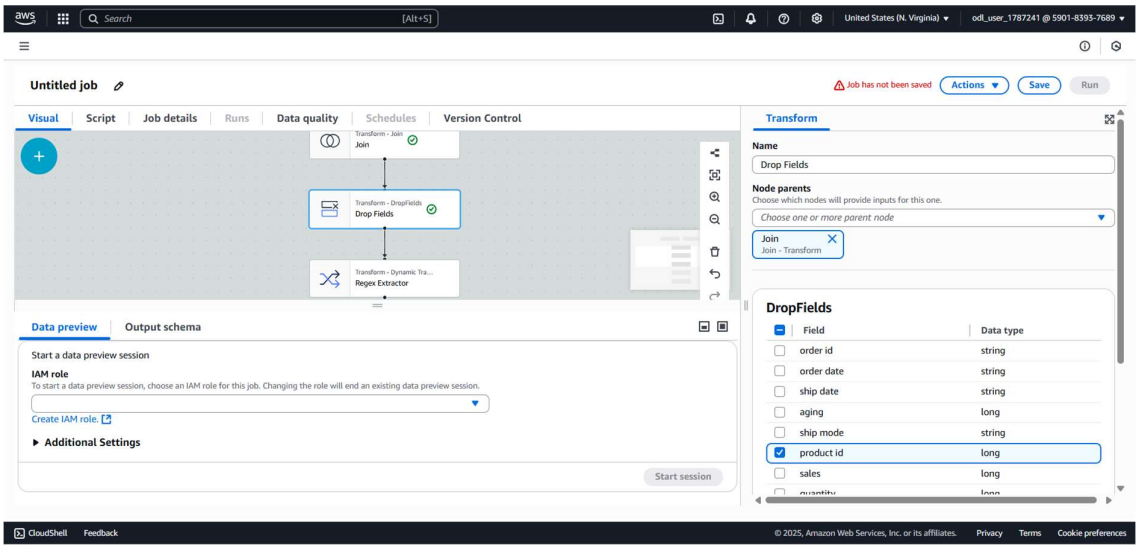
11. ✅ **Join Node Configuration**

Screenshot showing Join type as “Inner Join” with Product ID fields selected from both sides.



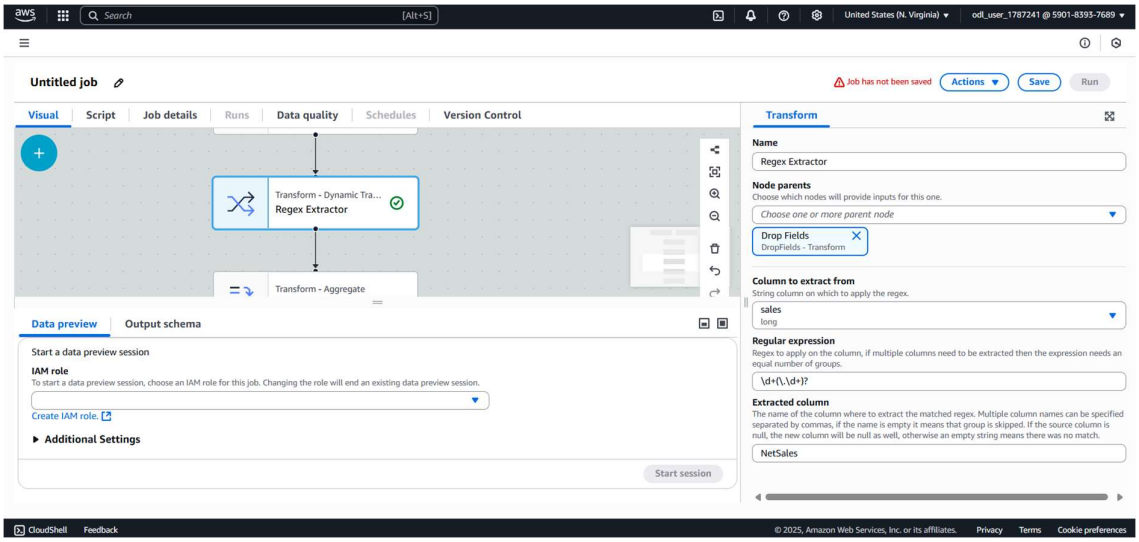
12. ✅ **Drop Fields Configuration**

Screenshot where one Product ID is dropped to remove duplicates.



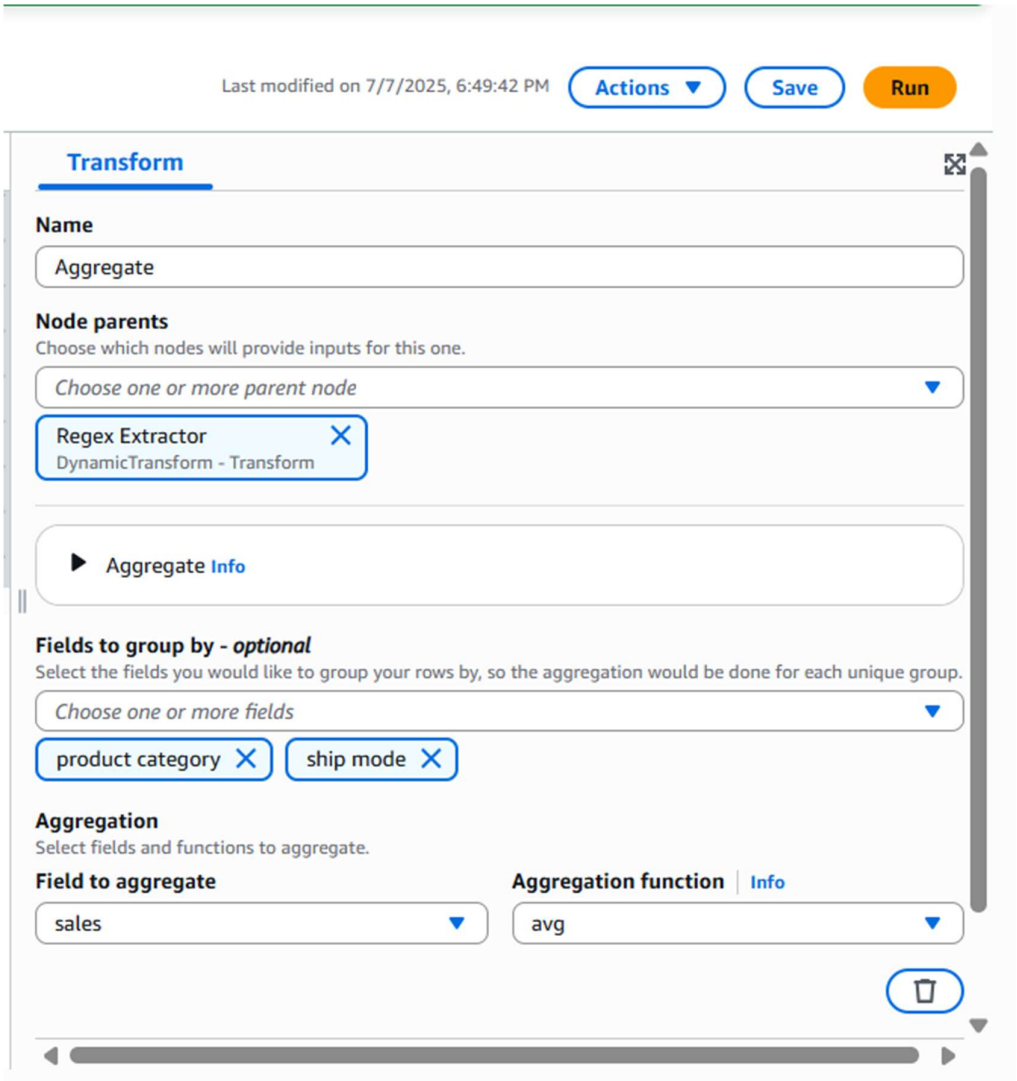
13. ✅ **Regex Extractor Node Settings**

Screenshot showing Sales column cleaned using regex: \\d+(\\.\\d+)?, and new column as NetSales.



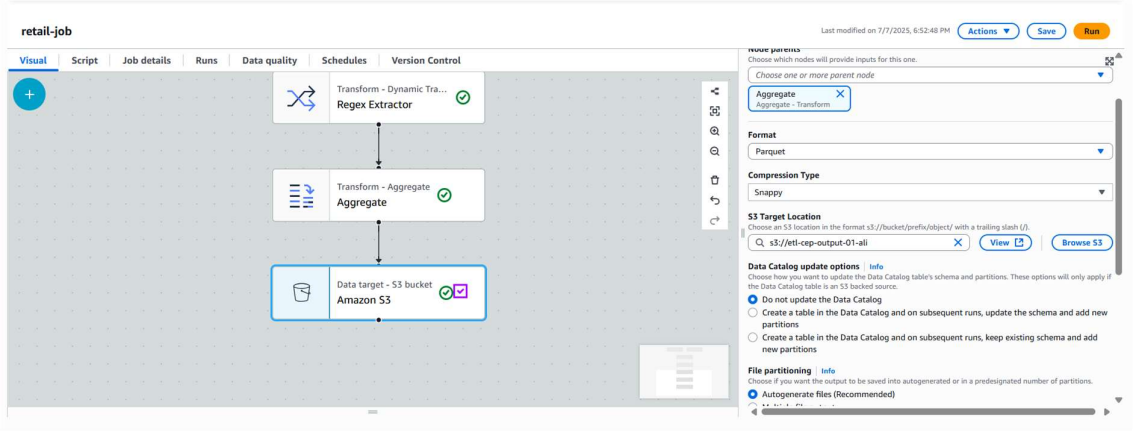
14. ✅ **Aggregate Node Settings**

Screenshot showing group by: Product Category, Ship Mode; aggregate: AVG(Sales).



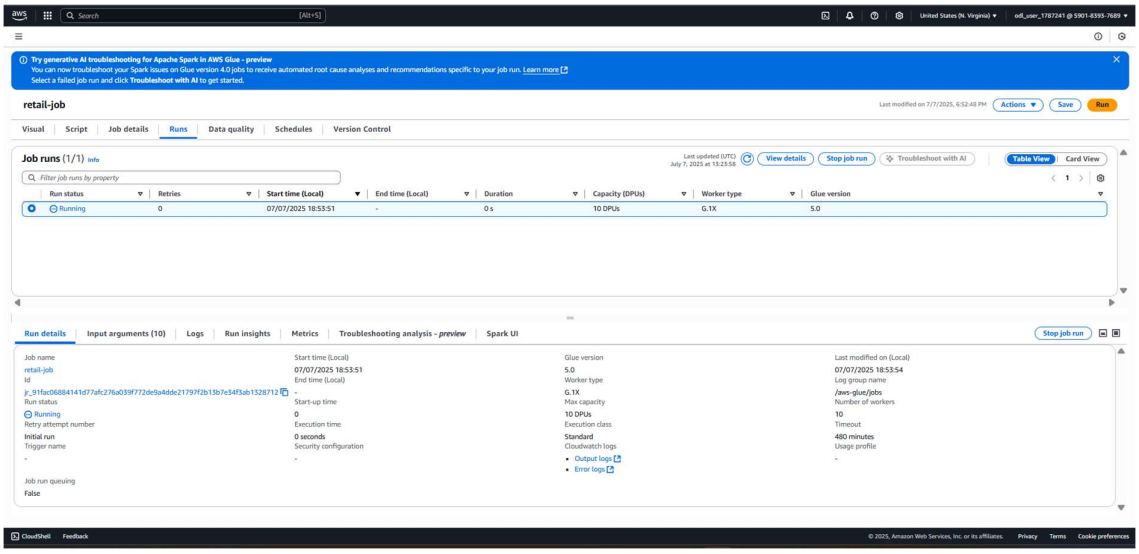
15. ✅ **ETL Output Node Setup**

Screenshot showing target S3 bucket selected: etl-cep-output-01-ali.



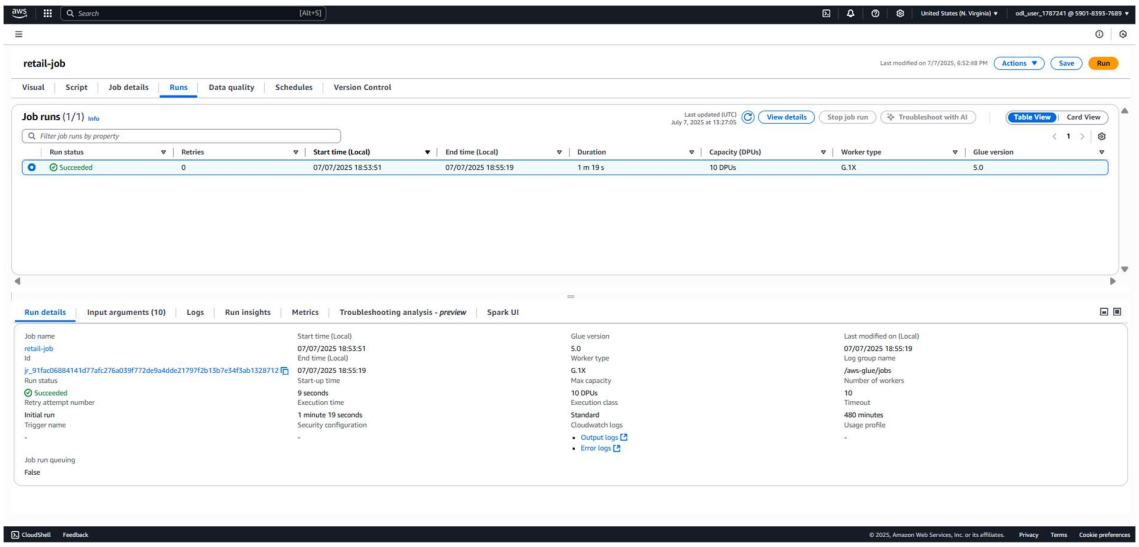
16. ✅ **Job Saved and Run Started**

Screenshot showing job name as etl-cep-job and job run in progress.



17. ✅ **Job Run Completed Successfully**

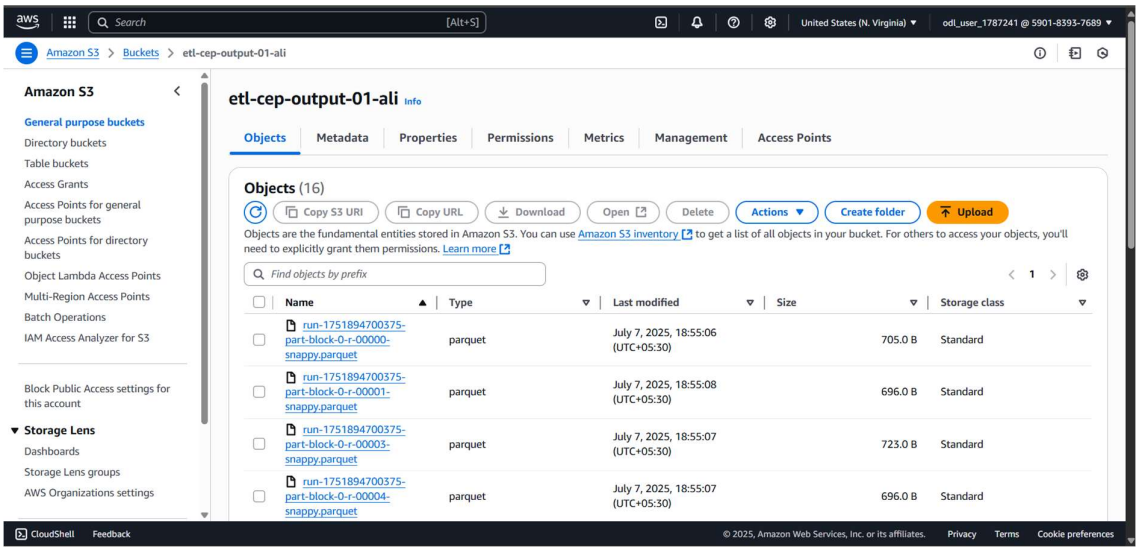
Screenshot of job run status: "Succeeded".



◆ Output and SQL Query

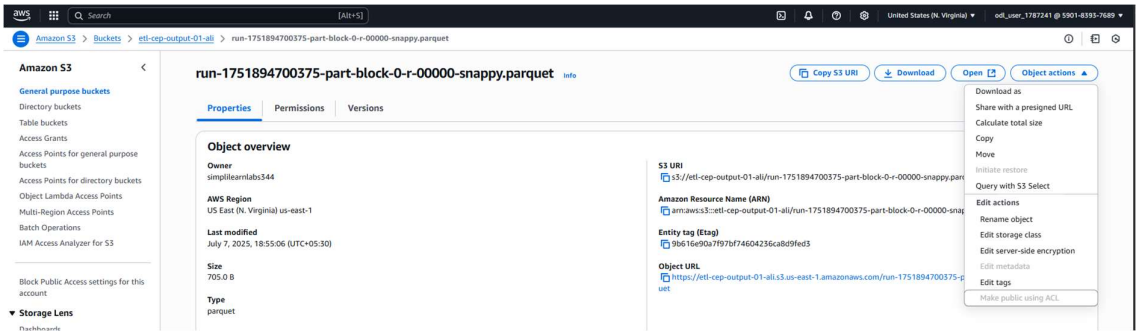
18. ✅ **Output File in Output Bucket**

Screenshot showing the .parquet file inside etl-cep-output-01-ali.



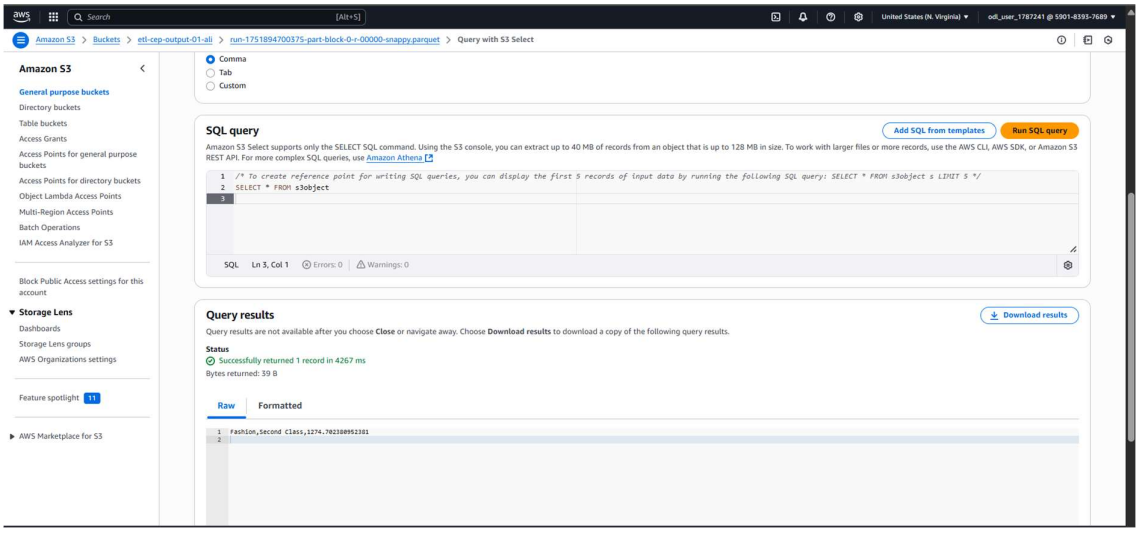
19. ✅ **S3 Select Query Panel**

Screenshot of "Query with S3 Select" screen, with format set to CSV and query executed.



20. ✅ **Query Output Displayed**

Screenshot showing sample rows of output (average sales by product category and ship mode).



7. Conclusion

This project demonstrated how AWS Glue can be effectively used to automate and manage ETL workflows. By integrating and transforming retail datasets, valuable insights were generated to support data-driven decision-making.

[End of Report]