AWS Academy Data Engineering [107814]

Rohith Yellambalase Vijayakumar

G26286080

The George Washington University

Department of Computer Science,

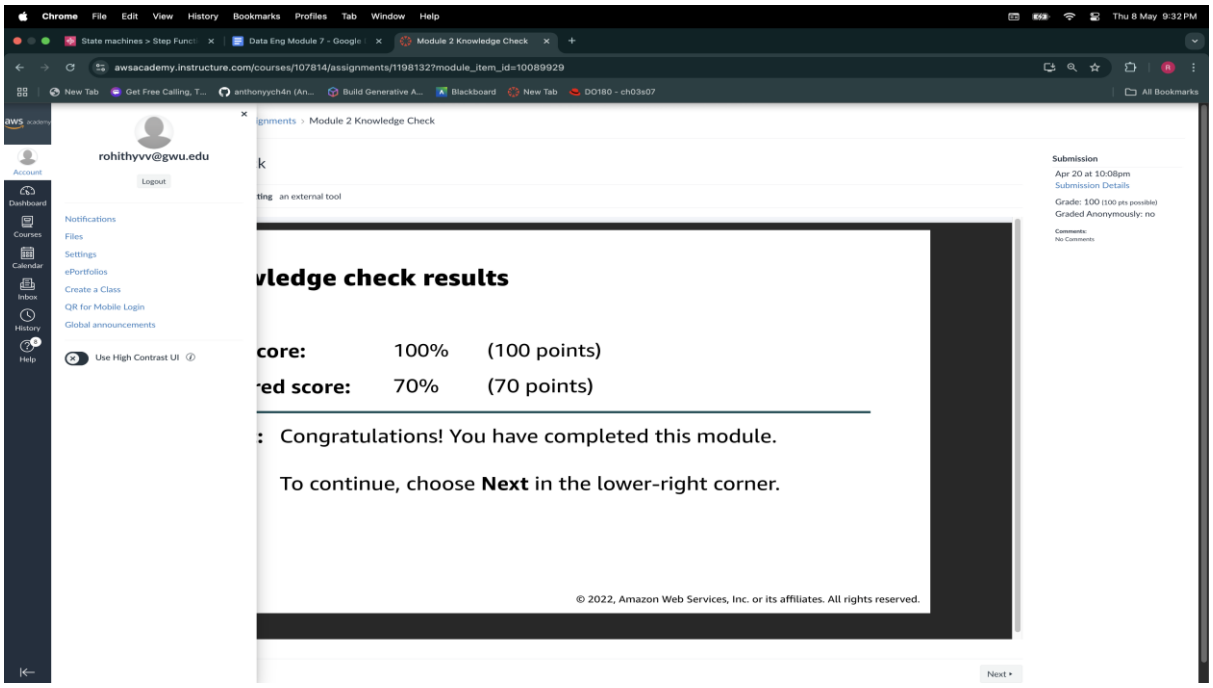
CSCI_6442_10: Database Systems-II

Prof. Rolando Fernandez

30th April 2025

Module 2 Data-Driven Organizations

Fig 2.1: End of module 2



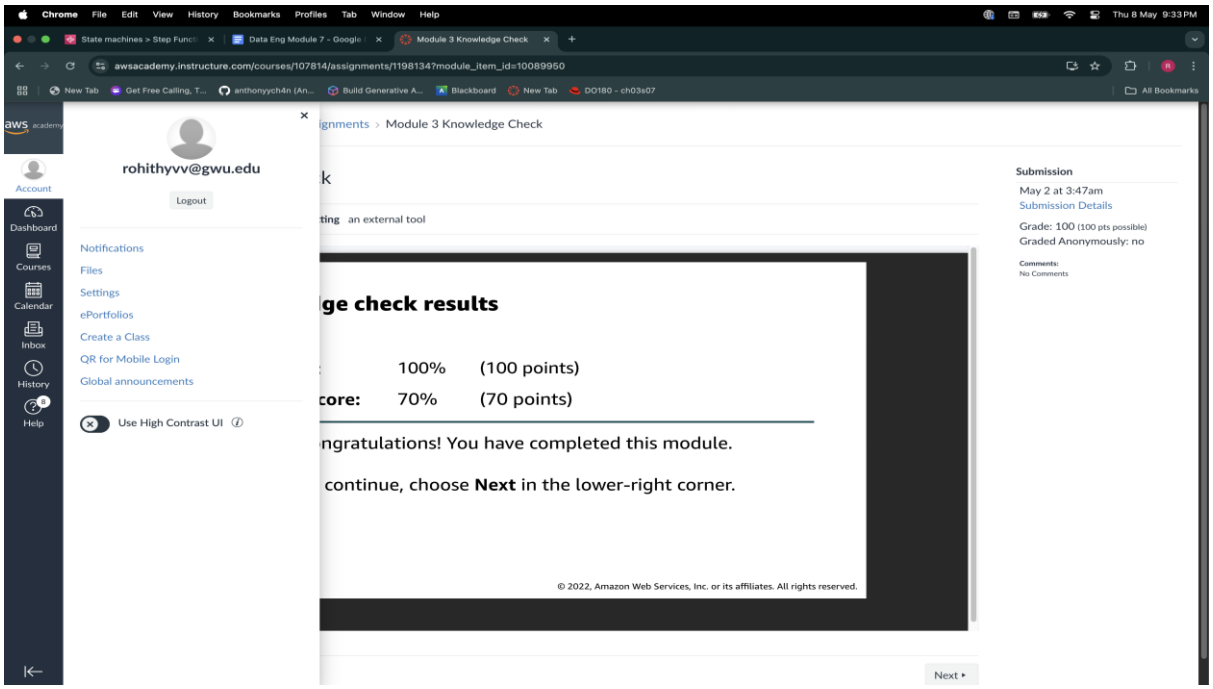
Note: This image shows the completion of module 2

Summary:

- **Centralized Data Lakes**: Modern data engineering focuses on building scalable **data lakes** (e.g., using Amazon S3) that consolidate structured and unstructured data from multiple sources into a single, accessible repository.
- **ETL and ELT Pipelines**: Data engineers design and manage **ETL (Extract, Transform, Load)** or **ELT** workflows (e.g., with AWS Glue) to ingest, clean, and prepare data for analytics, ensuring quality and consistency.
- **Data Democratization**: By providing governed, secure access to data (via tools like AWS Lake Formation), data engineers enable **self-service analytics** for stakeholders without relying on IT for every request.
- **Purpose-Built Data Stores**: While raw data resides in the lake, **purpose-built databases** (e.g., Amazon Redshift for analytics, DynamoDB for NoSQL) are used for specific workloads requiring optimized performance.
- **Data Governance & Security**: Data engineering ensures **compliance, lineage, and access control** across the entire data lifecycle using AWS IAM, encryption, and monitoring tools (e.g., AWS CloudTrail, AWS KMS).

Module 3: The Elements of Data

Fig 3.1: End of module 3



Note: This image shows the completion of module 3

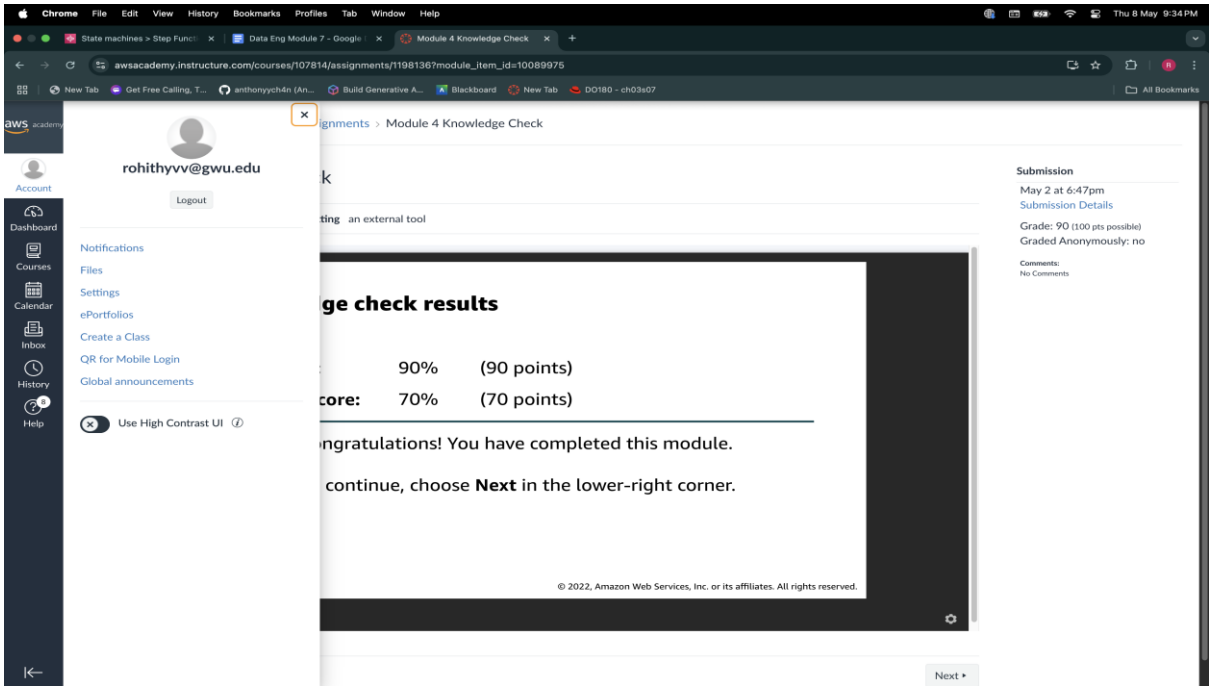
Summary:

- **Volume**: Refers to the **amount of data** generated and stored — impacts storage architecture and processing frameworks (e.g., batch vs distributed).
- **Velocity**: Measures the **speed of data generation and movement** — determines the need for **real-time streaming** vs **batch ingestion** solutions.
- **Variety**: Describes the **different types and formats** of data (structured, semi-structured, unstructured) — affects data modeling, storage choices, and transformation logic.
- **Veracity**: Represents the **accuracy, reliability, and trustworthiness** of data — essential for data quality, cleansing, and governance practices.
- **Value**: The **usefulness and business relevance** of the data — focuses on extracting insights through analytics, visualization, and decision-making processes.

Module 4: **Design Principles and Patterns for Data Pipelines**

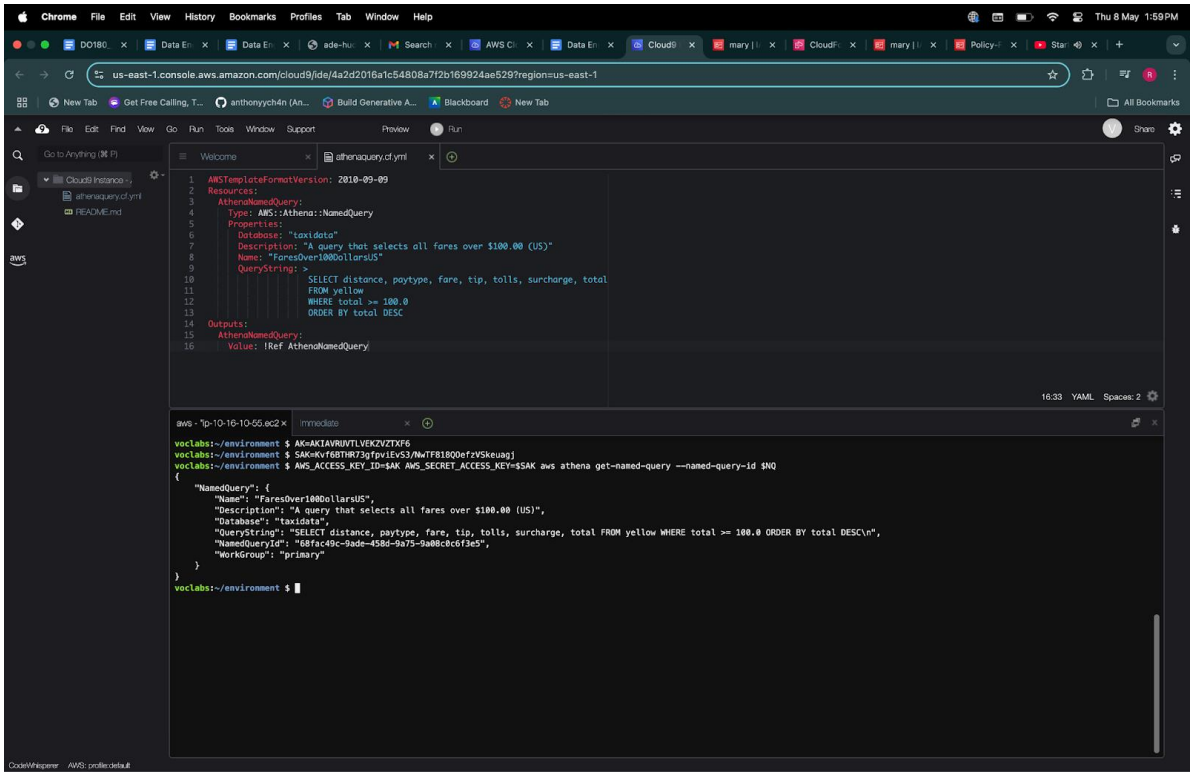
Lab: Querying Data by Using Athena

Fig 4.1: Preview data in the AWS Glue table



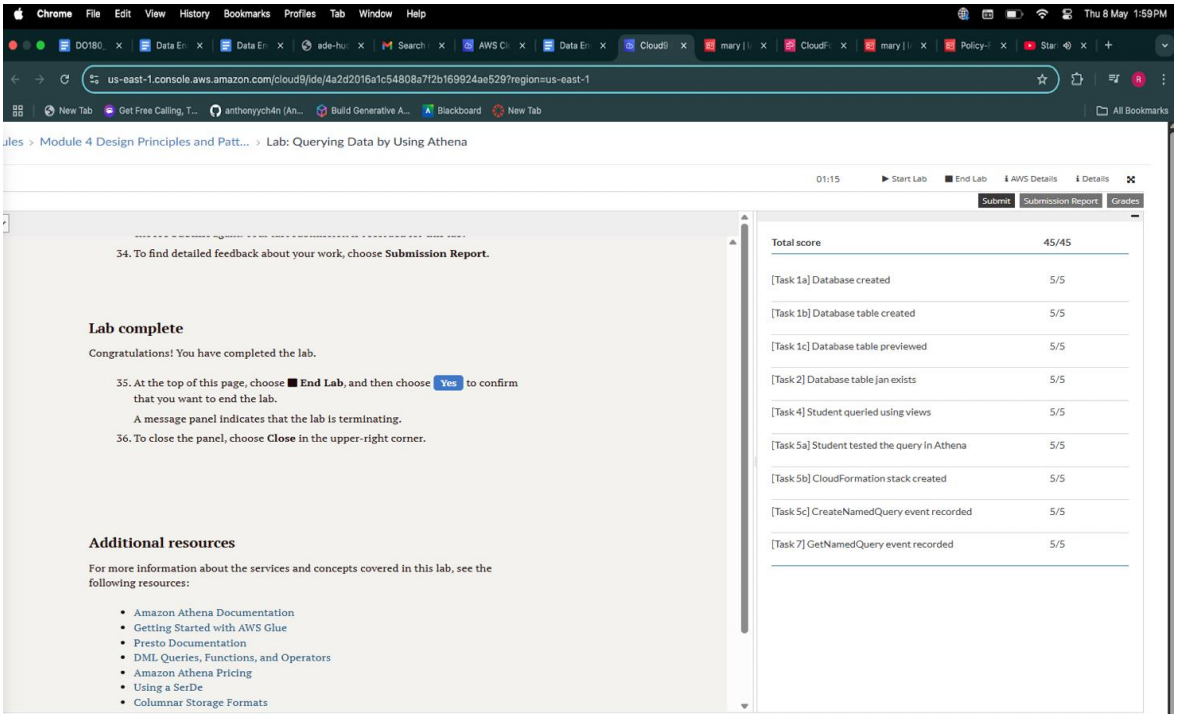
Note: I connected an Amazon S3 dataset to the table and defined the schema for the table using the bulk add columns feature. After creating the table, I learned how to preview the data by using the preview table feature.

Fig 4.2 : Create cloudformation template to easier access to SQL query for other departments.



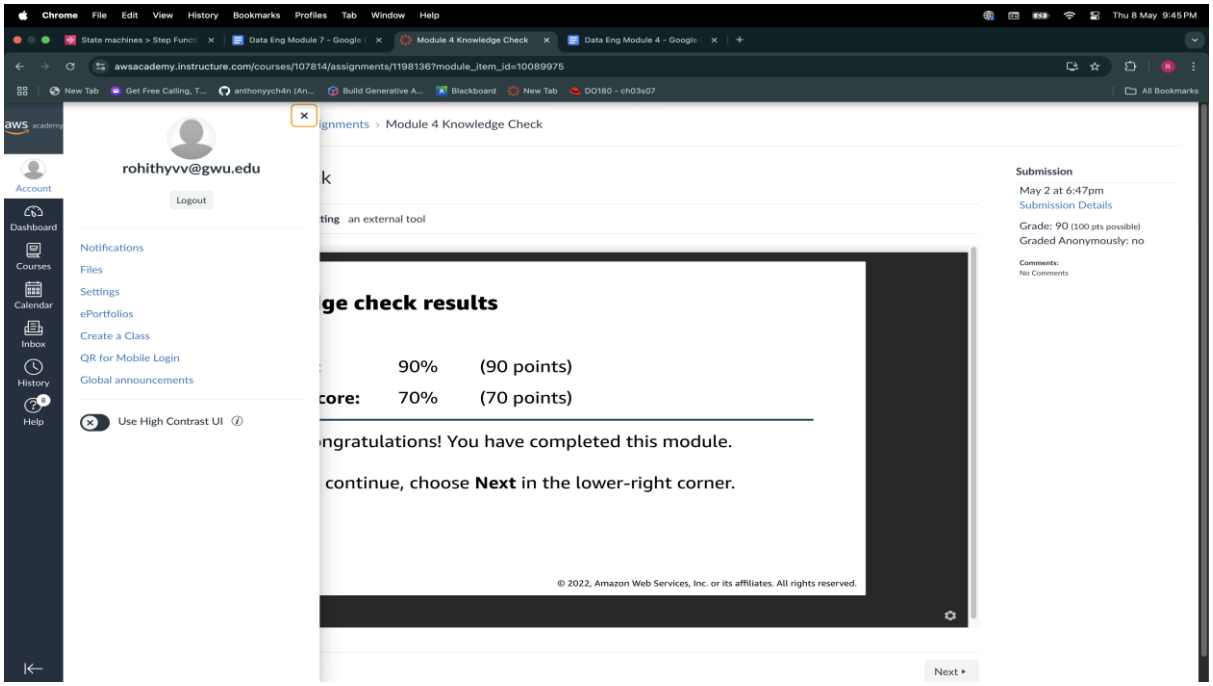
Note: Building queries in a development account and then testing them in a controlled account with production data can help to ensure that the query is designed as intended and generates the necessary insights. Here in the image, the query is created and stored cloudformation stack for easier access who are unaware of the SQL queries.

Fig 4.3: End of guided lab



Note: This image shows the end of guided lab Querying Data by Using Athena.

Fig 4.4 : *End of module 4*



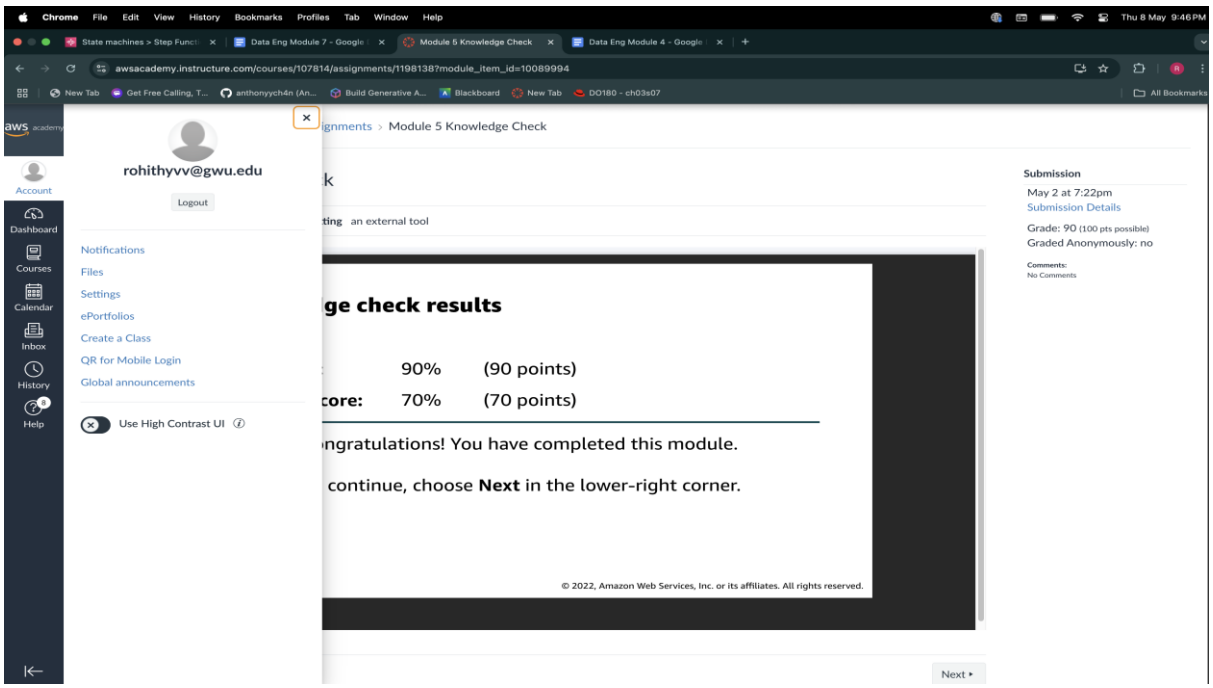
Note: This image shows the completion of module 4

Summary

- **Modern data architecture** enables centralized storage (data lakes) while supporting flexible movement of data between warehouses, analytics tools, and services for AI/ML applications.
- **Data movement patterns** include:
- *Outside In*: Moving structured data from databases or warehouses into a data lake for broader analysis.
- *Inside Out*: Moving curated data from a data lake into specialized tools like Redshift for reporting.
- *Around the Perimeter*: Direct transfers between services without touching the data lake.
- **AWS Glue** automates the ETL process by detecting schema, cataloging metadata, and transforming data across S3, Redshift, and other sources. It creates a centralized metadata catalog.
- **AWS Lake Formation** builds on top of AWS Glue to enforce governance by applying access policies and fine-grained controls over who can access which data at the table, column, or row level.
- **Stream processing pipelines** use services like CloudWatch, Kinesis Data Streams, and Apache Flink to process and analyze data in real time, sending results to destinations like OpenSearch for dashboards or to S3/Redshift for long-term analytics.

Module 5 : **Securing and Scaling the Data Pipeline**

Fig 5.1: End of module 5



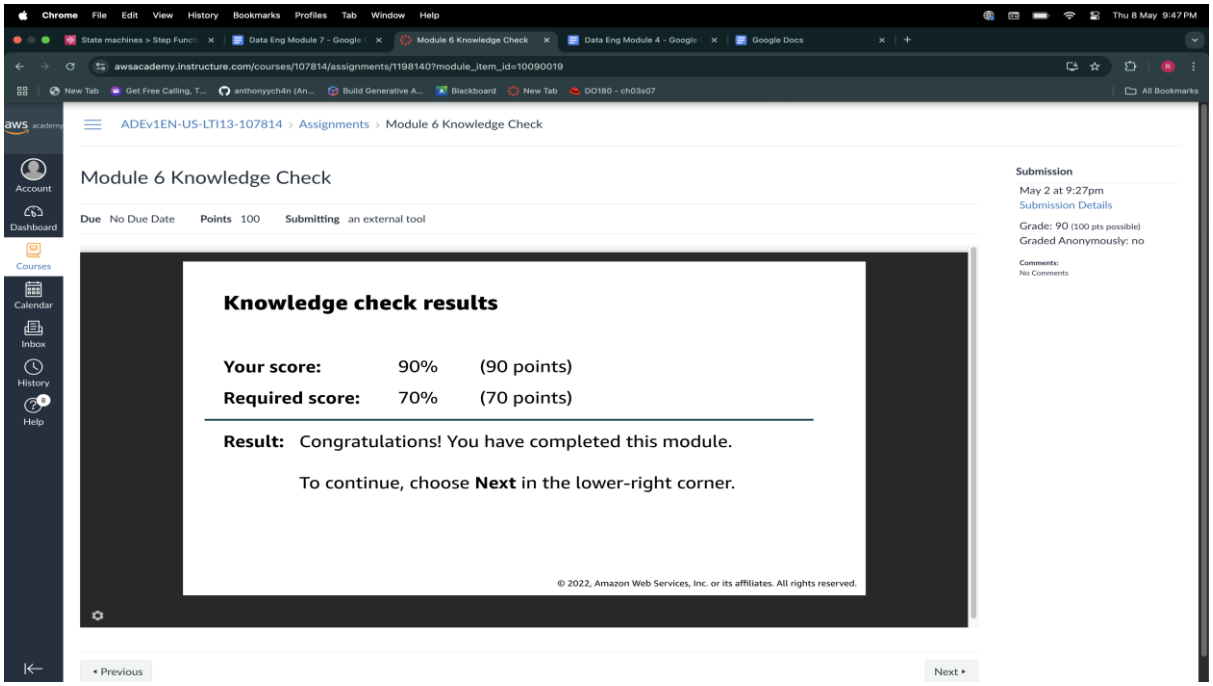
Note: This image shows the completion of module 5

Summary:

- **Implement Least Privilege Access**: Use IAM roles and policies to grant only the permissions necessary for each pipeline component, minimizing security risks and following the principle of least privilege.
- **Secure Data at Rest and in Transit**: Encrypt data using AWS Key Management Service (KMS) for both storage (e.g., S3, Redshift) and transmission (e.g., TLS for APIs and streaming).
- **Enable Scalability with Serverless and Distributed Services**: Use scalable services like AWS Glue, Kinesis, Lambda, and EMR that automatically adjust to data volume and processing needs without manual provisioning.
- **Monitor and Audit with CloudTrail and CloudWatch**: Track pipeline activities and set up alerts for unusual access or performance issues using CloudWatch metrics, logs, and AWS CloudTrail auditing.

Module 6: **Ingesting and Preparing Data**

Fig 6.1: End of module 6



Note: This image shows the completion of module 6

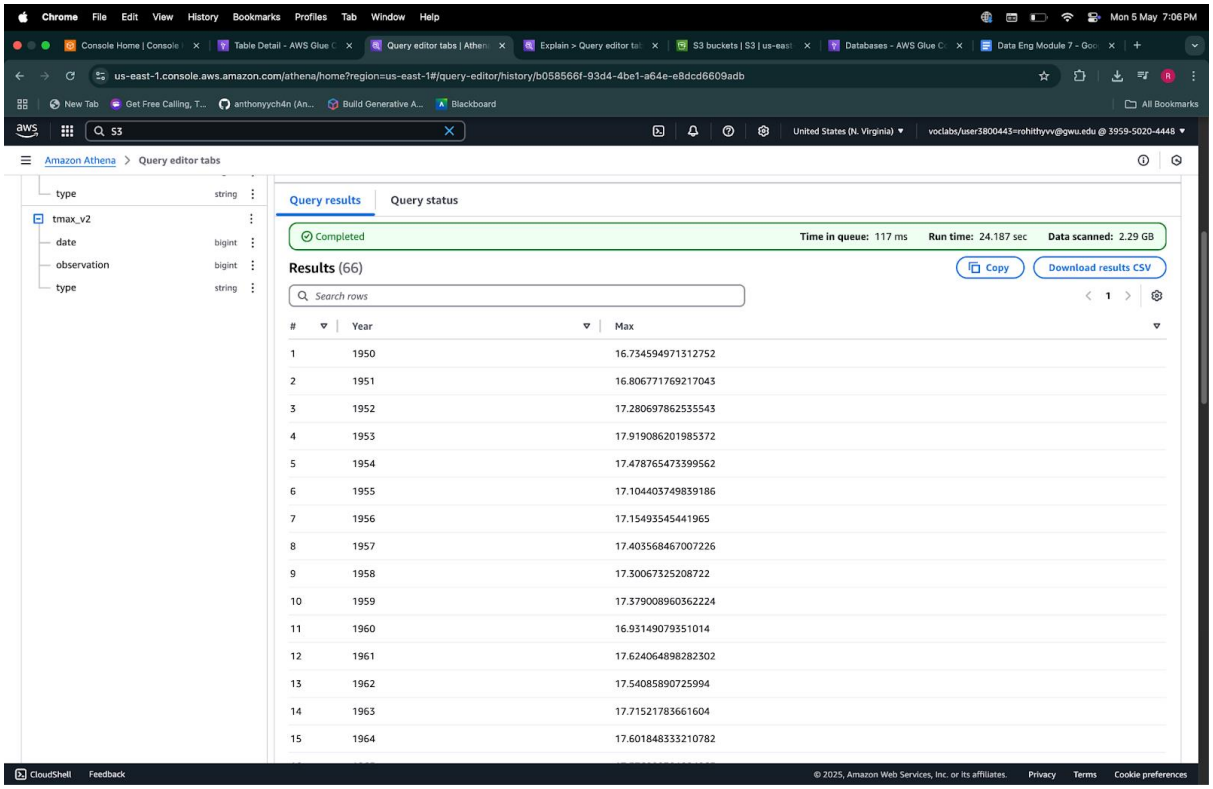
Summary:

- ETL (Extract, Transform, Load): Used for structured data. Data is cleaned and transformed before loading into a warehouse like Redshift for efficient querying.
- ELT (Extract, Load, Transform): Used for unstructured/semi-structured data. Raw data is loaded into a data lake (e.g., S3), then transformed as needed for analytics or ML.
- The term **data wrangling** reflects the complexity and messiness of transforming large amounts of unstructured data or sets of structured data with multiple schemas across multiple sources into a meaningful set of data that has value for downstream processes or users.
- Structuring: Organizes raw data into a consistent format or schema, making it easier to process and analyze downstream.
- Enriching: Combines data from multiple sources or adds calculated fields to improve data quality and usefulness.
- Validating: Checks data accuracy and completeness after transformation to ensure it meets expected standards before loading or publishing.

Module 7: **Ingesting by Batch or by Stream**

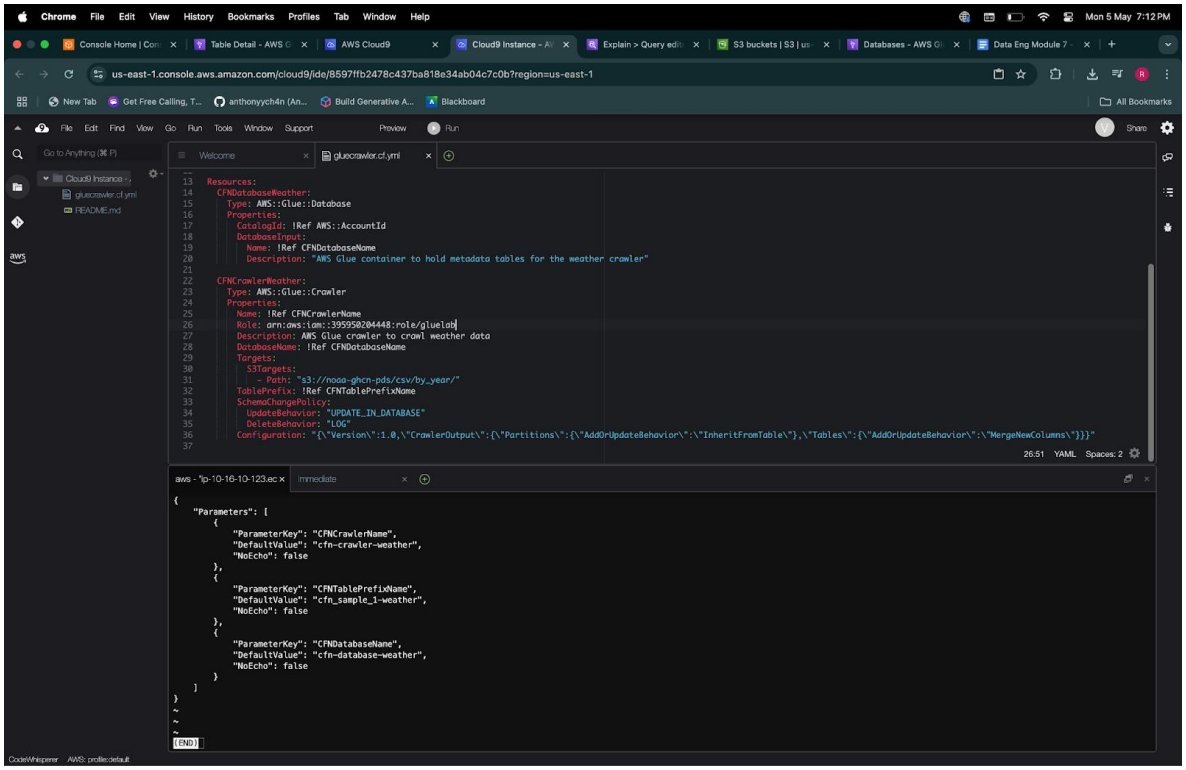
Lab: Performing ETL on a Dataset by Using AWS Glue

Fig 7.1: Analyzing Historical Temperature Data with Amazon Athena



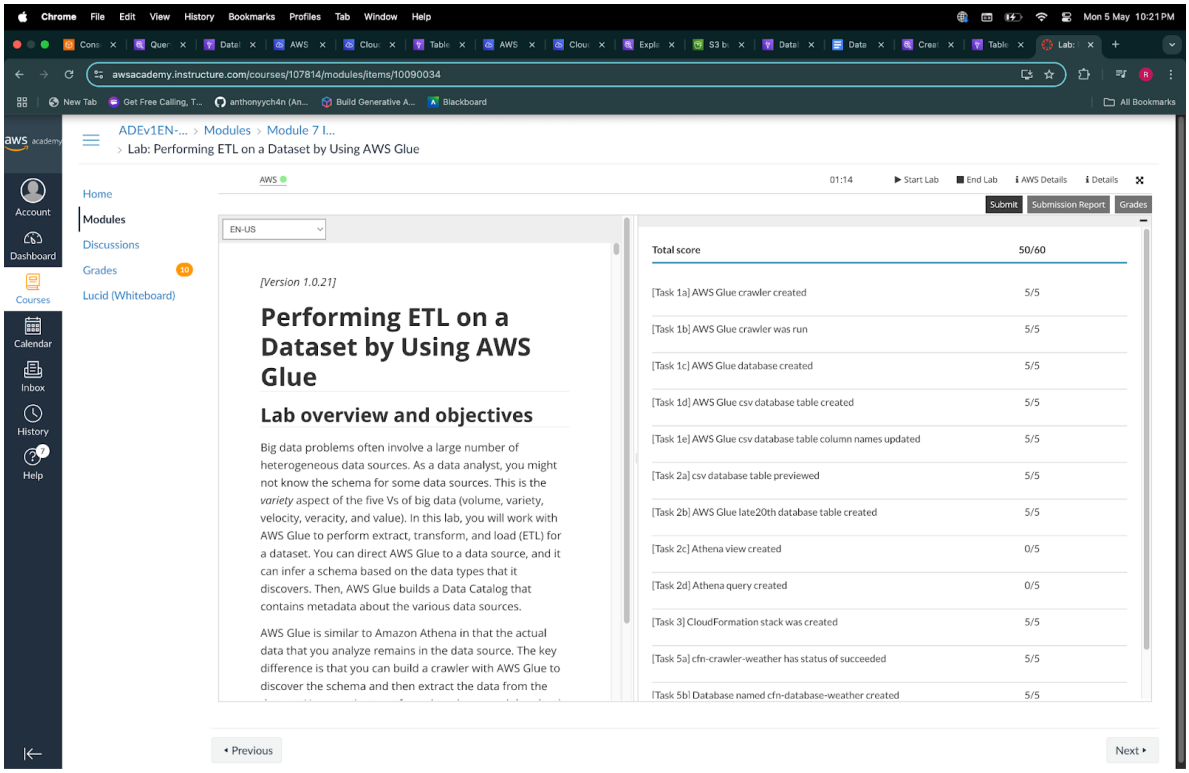
Note: This lab demonstrates querying historical temperature datasets using Amazon Athena. The analysis includes maximum temperature observations across multiple years, showcasing Athena's capability to query large datasets stored in Amazon S3 efficiently.

Fig 7.2: Confirm that the crawler is finished running



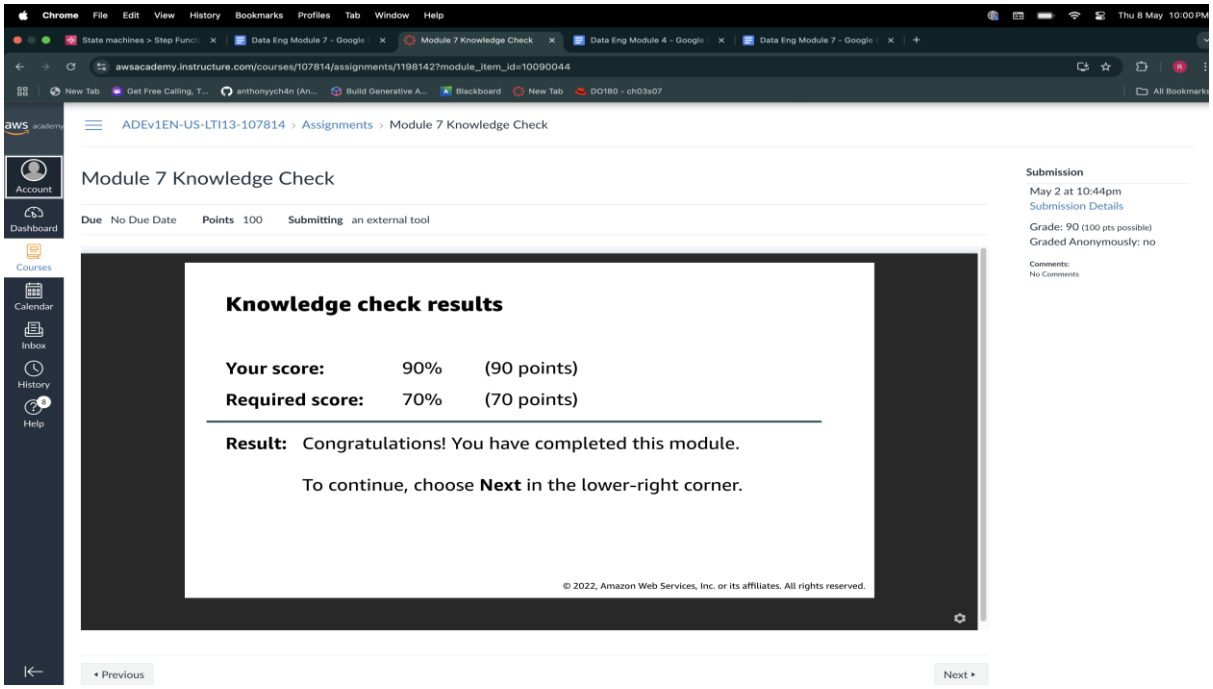
*Note: Notice that the **LastCrawl** section is included, and the status in that section is SUCCEEDED. This means that Mary was able to run the crawler successfully. This result confirms that Mary has access to the AWS Glue crawler that you created and deployed with CloudFormation.*

Fig 7.3: End of guided lab



Note: This image shows the end of guided lab Performing ETL on a Dataset by Using AWS Glue

Fig 7.4: End of module 7



Note: This image shows the completion of module 7

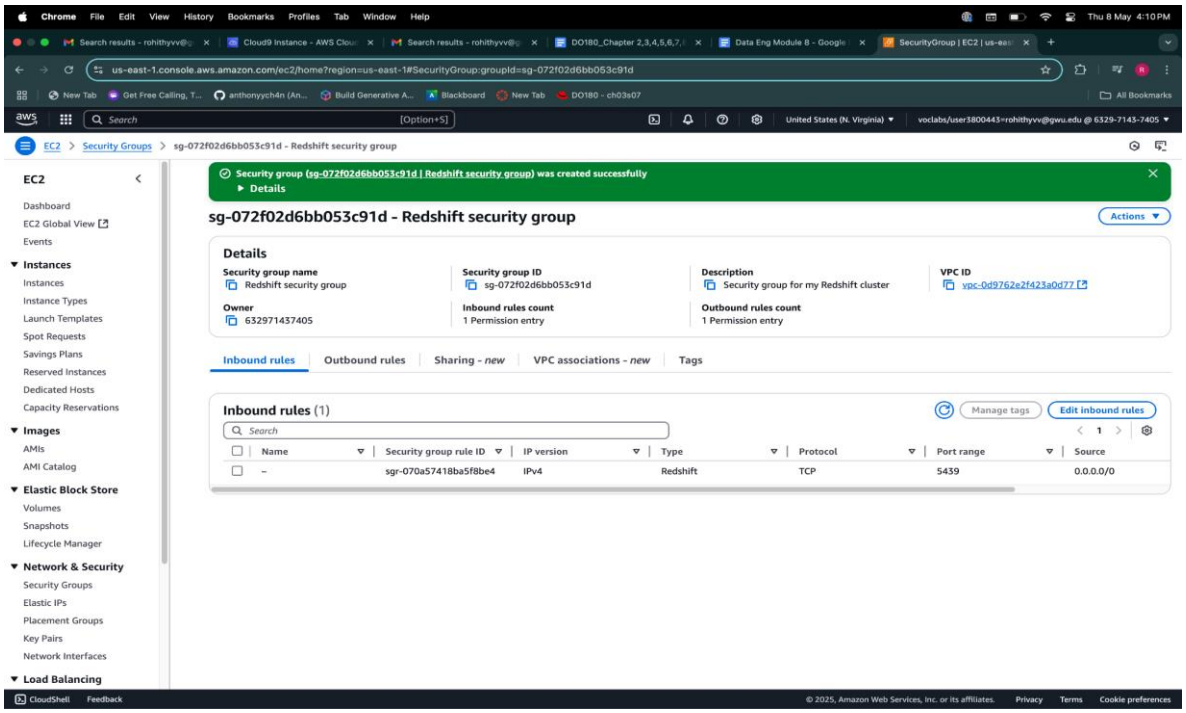
Summary:

- ETL Automation with AWS Glue: Instead of manually preparing data every month, you automate the process using ETL scripts (extract → transform → load), scheduled via batch jobs or workflows (e.g., AWS Glue workflows or cron).
- Glue Crawler + Data Catalog: Glue Crawlers scan data (e.g., in S3), detect schema and format, and create or update metadata entries in the Glue Data Catalog. The Catalog acts as a metadata store used by Athena, Redshift, and Glue jobs.
- AWS Glue Studio & Apache Spark: Glue Studio allows visual authoring of ETL jobs, running on a Spark-based serverless engine for fast, scalable data processing, with integrated job monitoring and orchestration.
- Kinesis and IoT Streaming: Kinesis Data Streams enable real-time ingestion and processing of streaming data from sources like sensors and apps. The MQTT pub/sub model in AWS IoT Core helps devices publish data even with intermittent connectivity.
- IoT Analytics Components: AWS IoT Analytics uses channels, pipelines, data stores, and datasets to ingest, transform, store, and query IoT data—typically writing results to S3 for analysis or integration with data lakes.

Module 8: **Storing and Organizing Data**

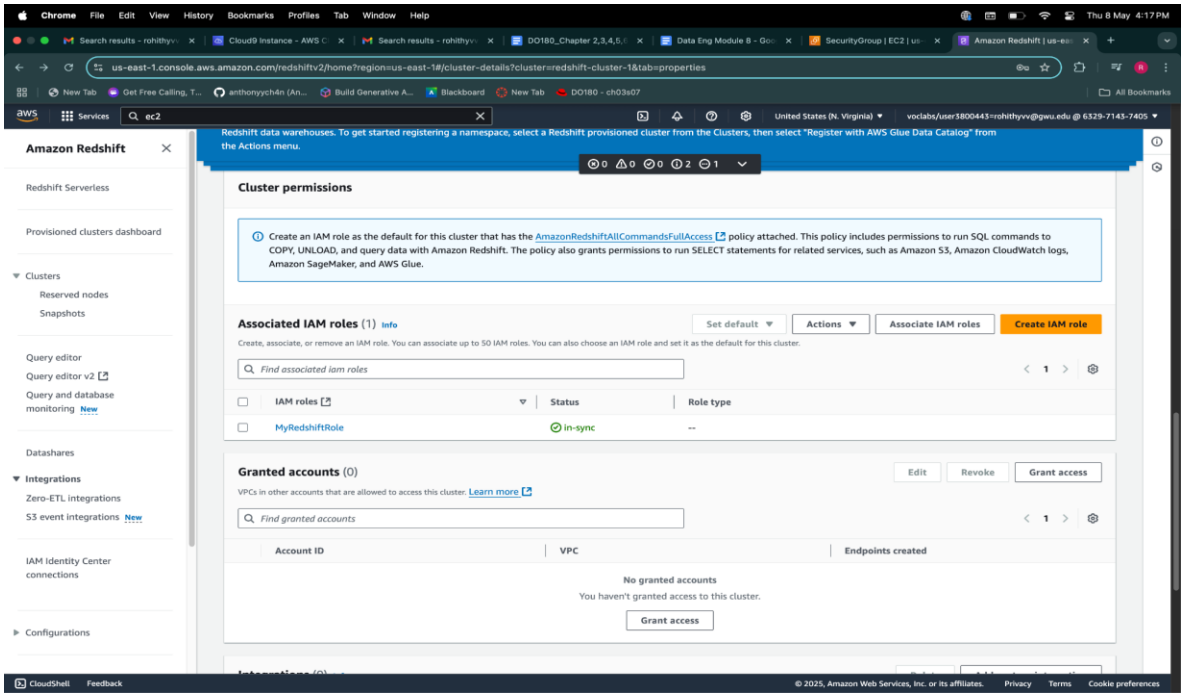
Lab: Storing and Analyzing Data by Using Amazon Redshift

Fig 8.1: Configure Redshift Security Group for Inbound Access



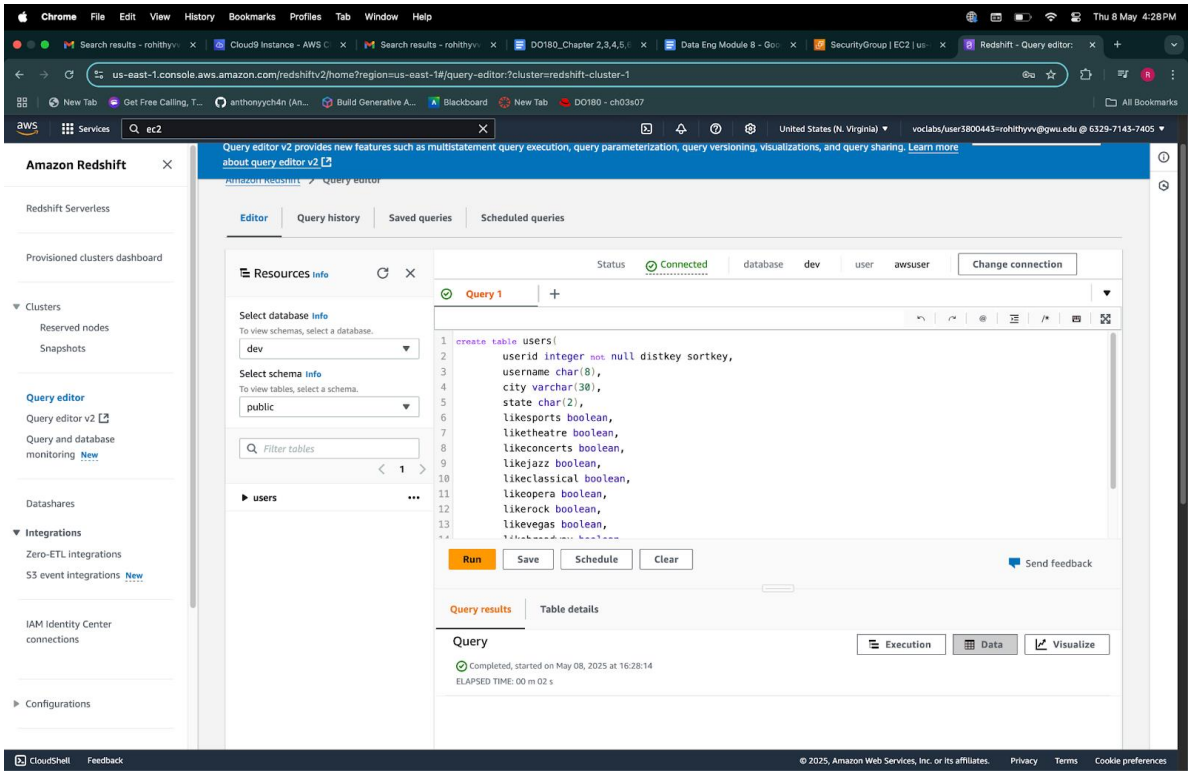
Note: Created a security group for the cluster so that other resources, such as Amazon EC2 instances, can access it through port 5439. And then added the cluster to the VPC by associating it with the security group created.

Fig 8.2: Redshift Cluster IAM Role Association



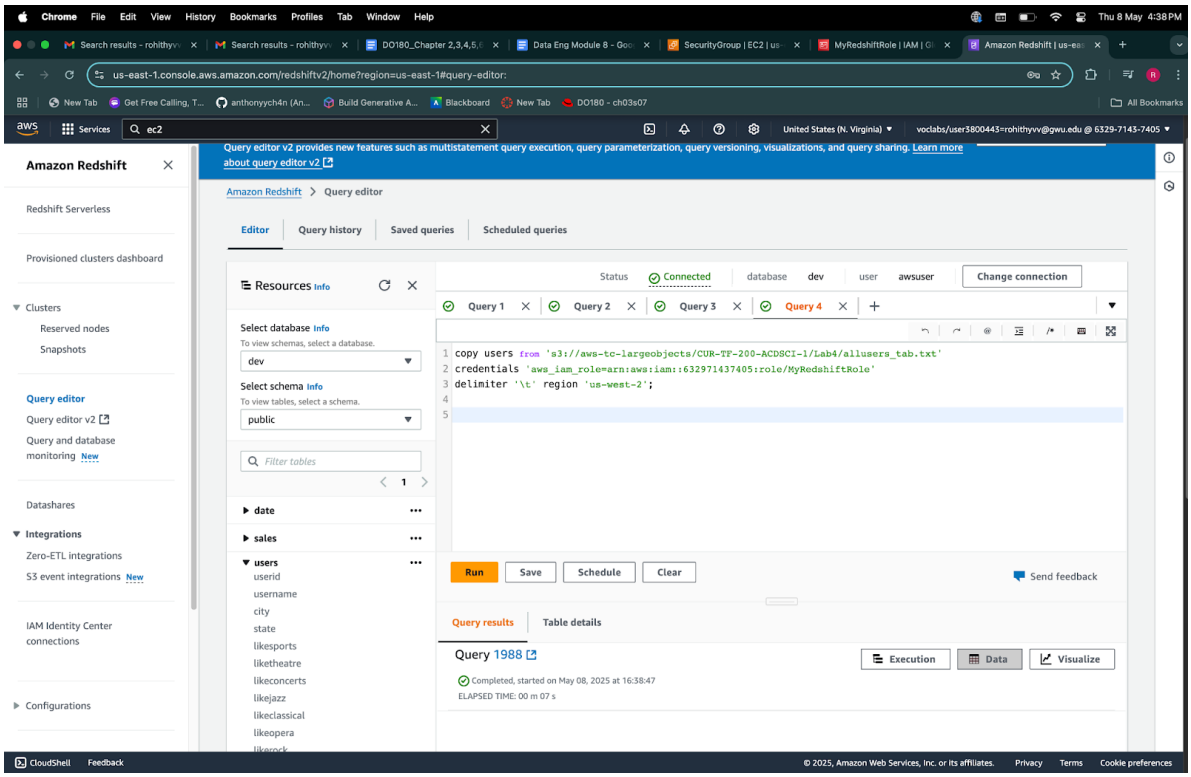
Note: IAM Role `MyRedshiftRole` is associated with the Redshift cluster, granting access to Amazon S3, CloudWatch Logs, and AWS Glue.

Fig 8.3: Redshift Table Creation in Query Editor



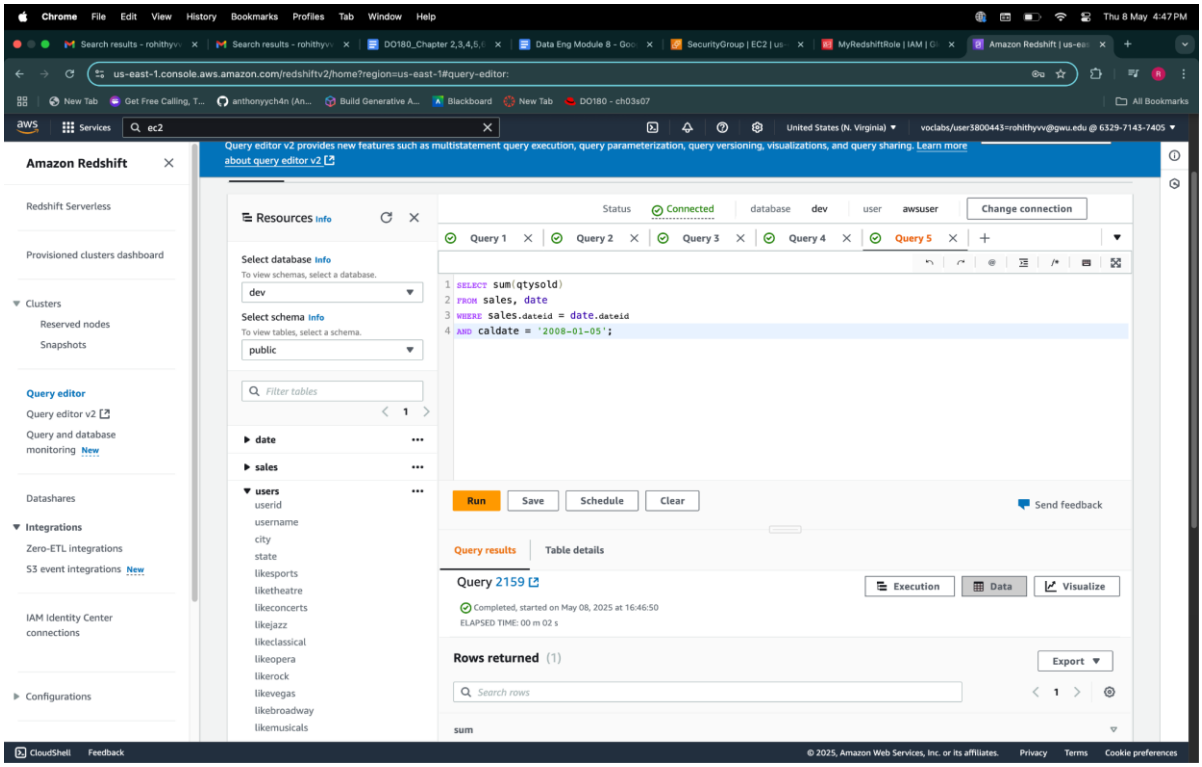
Note: A `users` table is created in the Redshift Query Editor with various boolean fields representing user preferences and a distkey sortkey.

Fig 8.4: Redshift Data Copy from S3 to Users Table



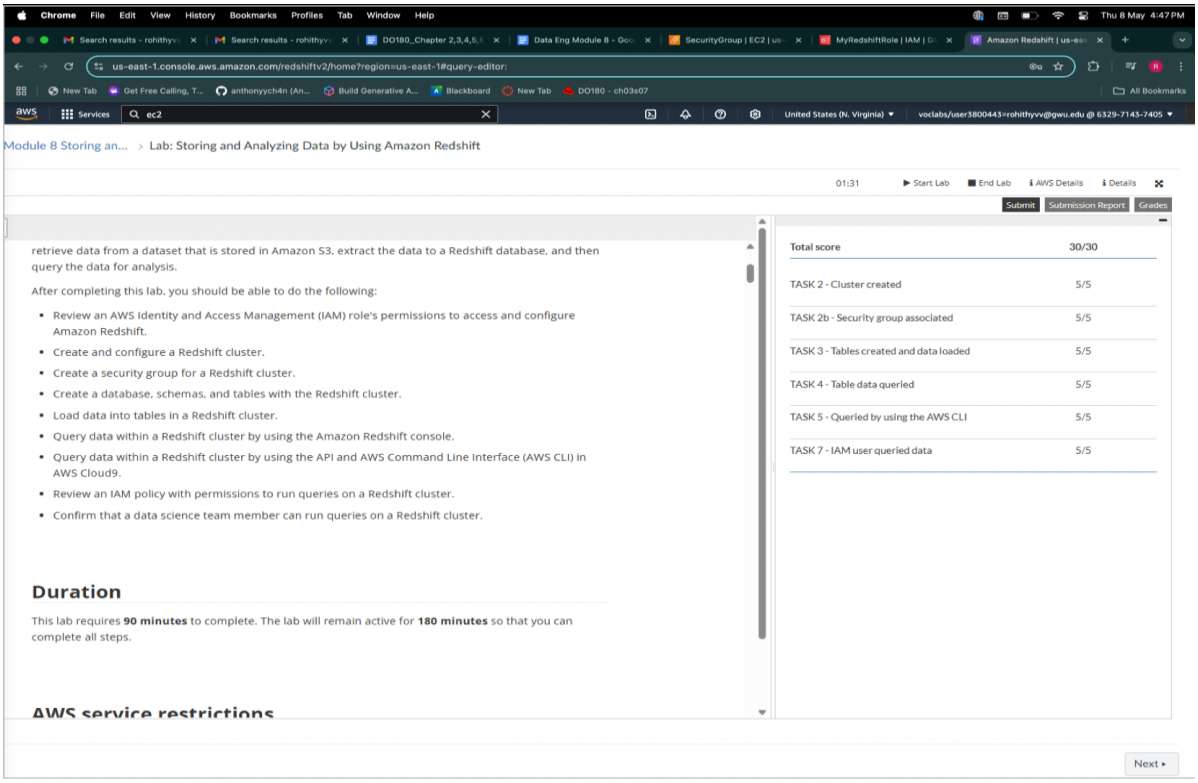
Note: Data is copied from S3 to the `users` table using `COPY` command with IAM role and tab-delimited file configuration.

Fig 8.5: Querying Sales Data in Amazon Redshift



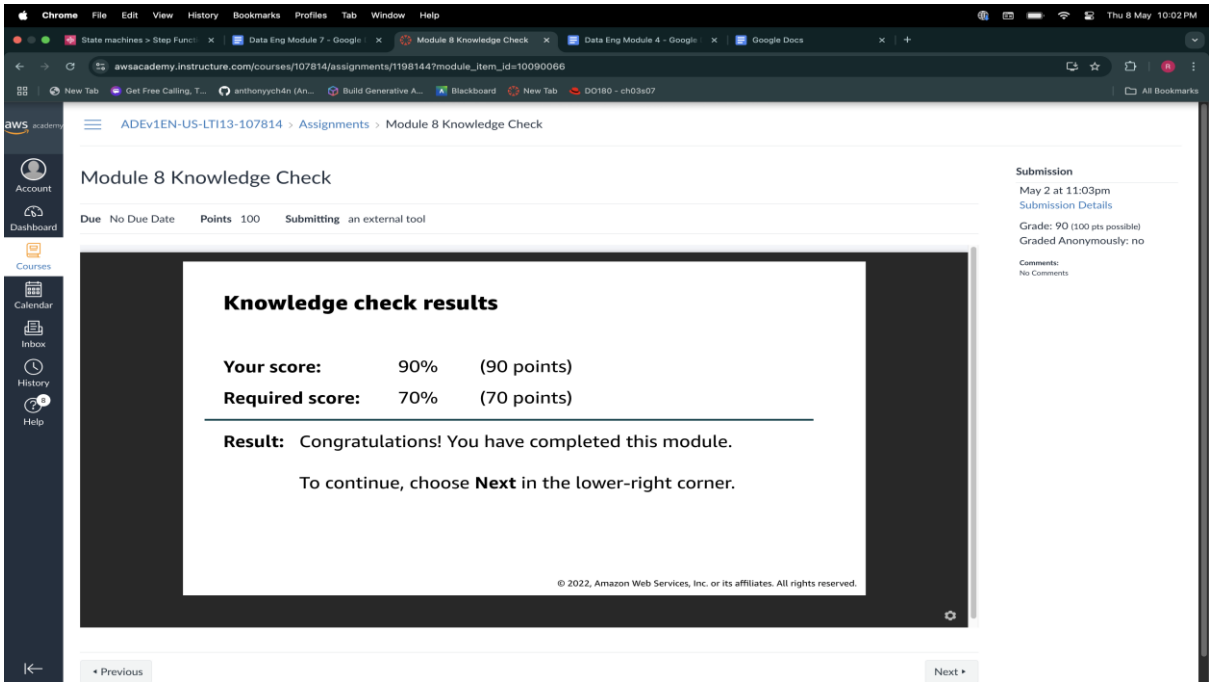
Note: This lab demonstrates querying sales data from the Redshift cluster using SQL. It calculates the total quantity sold on a specific date using the `SUM` function and joins tables by date ID.

Fig 8.6: End of guided lab



Note: This image shows the end of guided lab Performing ETL on a Dataset by Using AWS Glue

Fig 8.7: End of module 8



Note: This image shows the completion of module 8

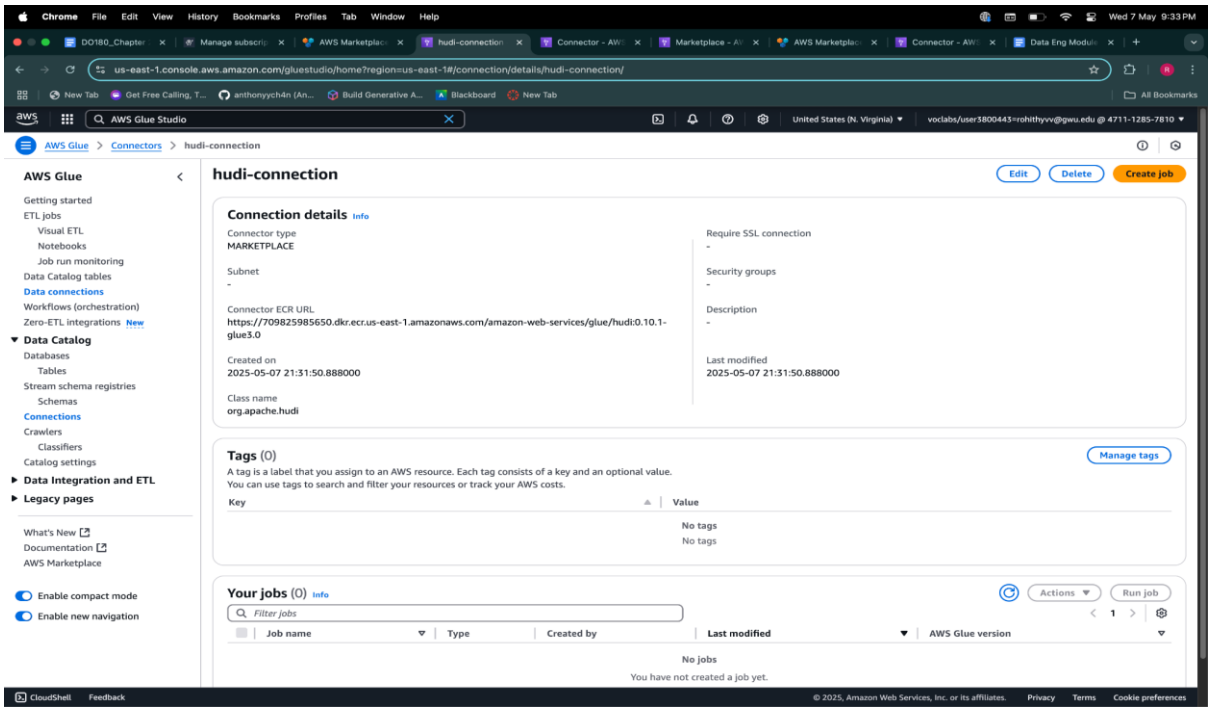
Summary:

- Data lakes and data warehouses serve different purposes — data lakes handle raw, semi-structured, and unstructured data (like IoT or logs), while data warehouses handle structured data optimized for fast analytics (like Redshift).
- AWS Lake Formation simplifies data lake management — it automates schema discovery, sets granular security (column/row/cell level), performs ML-based data cleaning (FindMatches), and optimizes storage via compaction.
- Amazon Redshift architecture is tiered and parallel — client apps interact with the leader node, which plans queries and distributes work to compute nodes, which process in parallel using multiple node slices.
- Leader node in Redshift handles query parsing, planning, and final result aggregation — compute nodes never interact with users directly; they just execute parts of the plan.
- ETL and ELT pipelines integrate both lake and warehouse — raw data lands in Amazon S3 (data lake), gets structured and enriched, and then loaded into Amazon Redshift for analytics and reporting.

Module 9: **Processing Big Data**

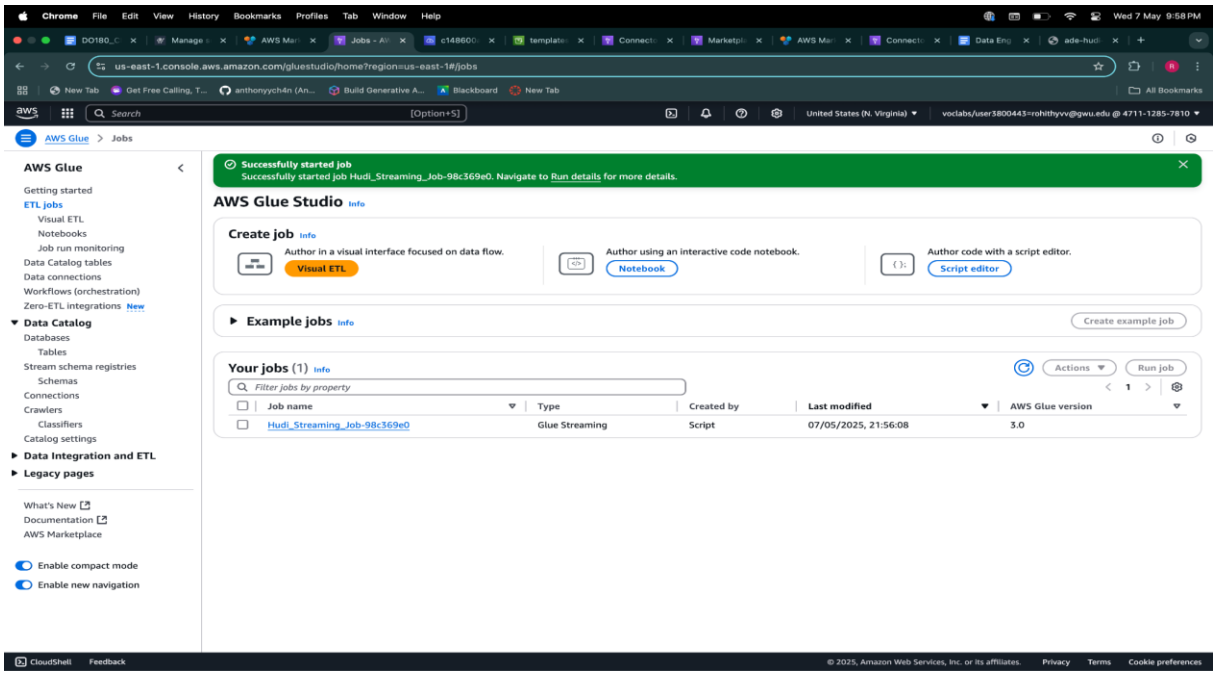
Lab: Processing Logs by Using Amazon EMR

Fig 9.1: AWS Glue Hudi Connection Setup



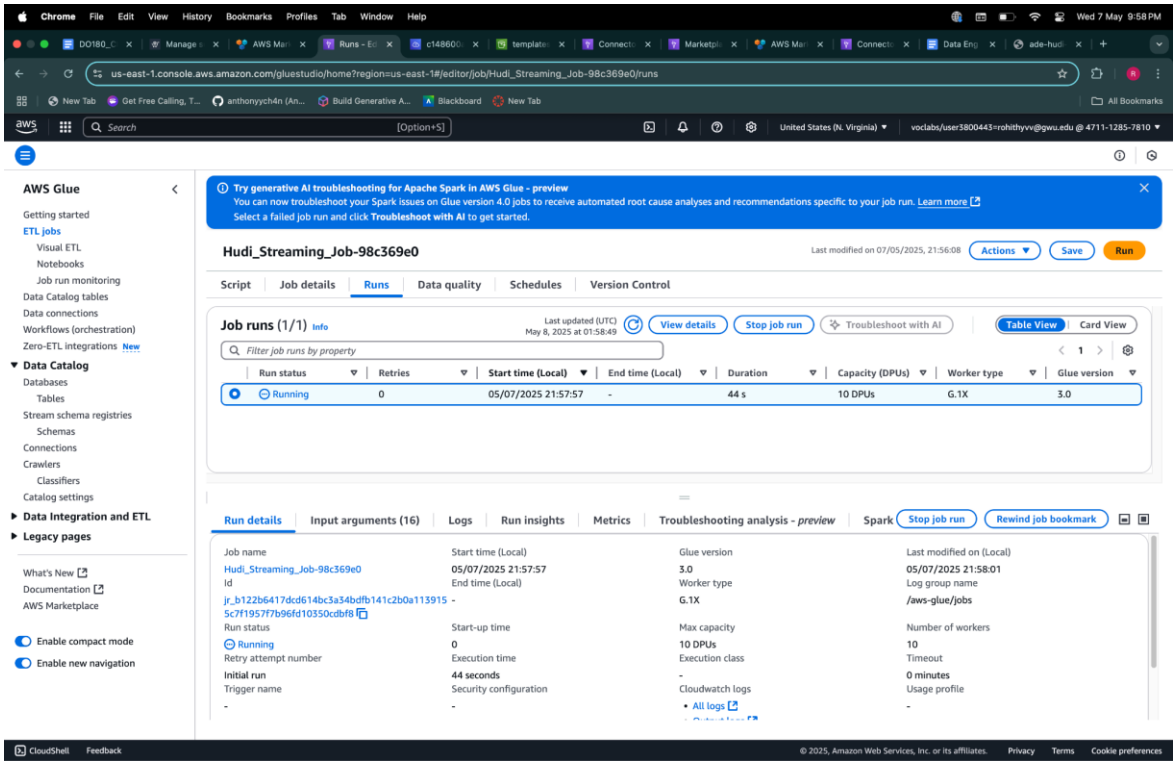
Note: The Hudi connector is successfully created in AWS Glue, enabling integration with Hudi-managed datasets for real-time data processing. The ECR URL is linked to the Glue 3.0 version.

Fig 9.2: AWS Glue Job Initialization



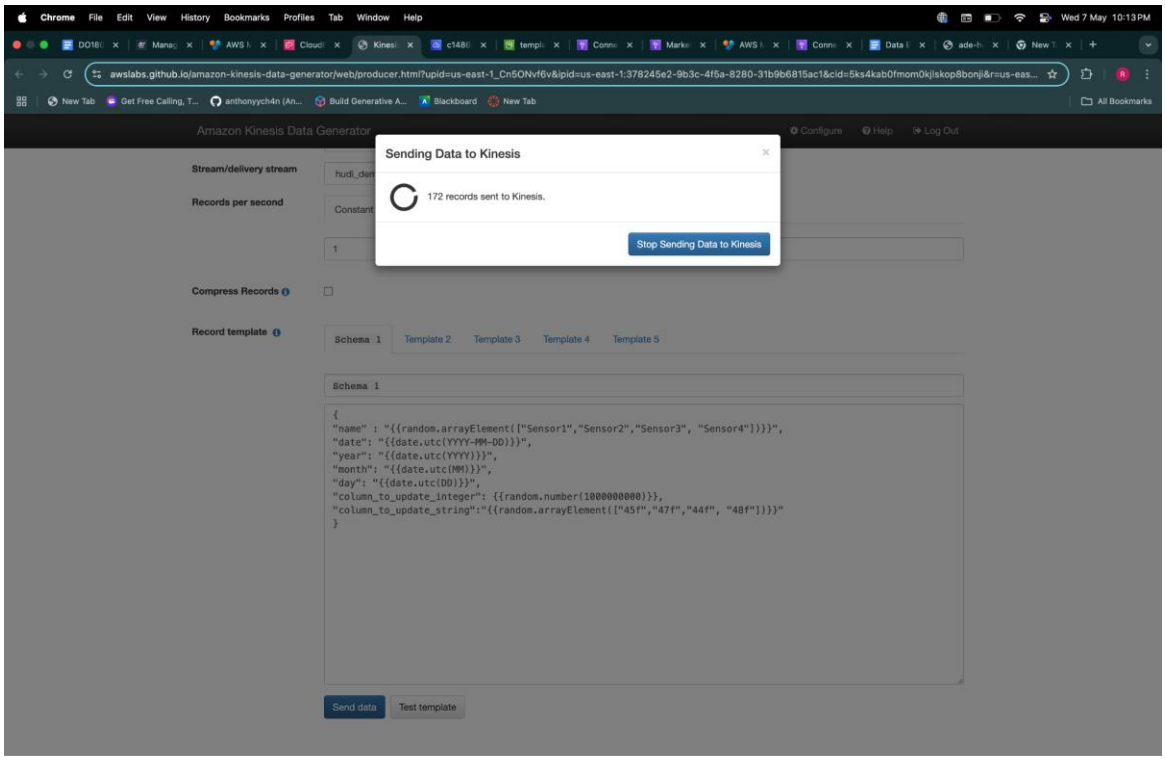
Note: Hudi_Streaming_Job is initiated successfully in AWS Glue Studio. The job is configured as a Glue Streaming script, ready for data transformation and processing in real-time.

Fig 9.3: AWS Glue Job Monitoring



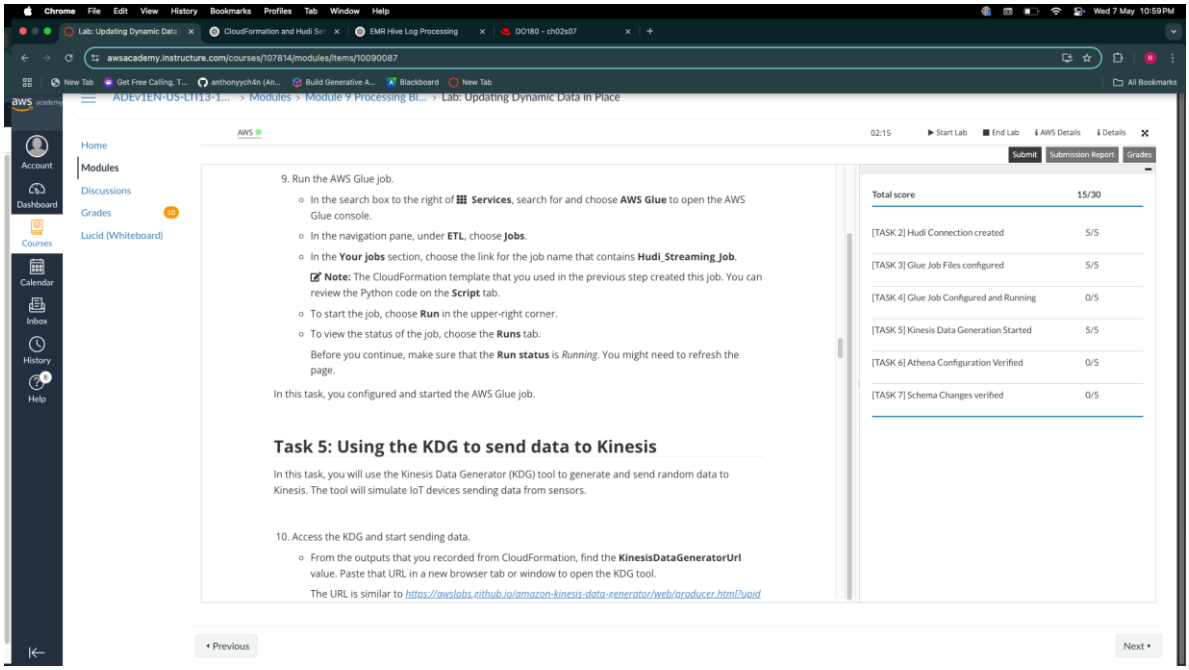
Note: The Glue Streaming job is currently running with 10 DPUs and a G.1X worker type. Real-time status, logs, and metrics are being tracked for monitoring.

Fig 9.4: Kinesis Data Stream Generation



Note: Kinesis Data Generator (KDG) is sending data to the "hudi_demo_stream" in real-time, simulating IoT device metrics. The generator is producing constant data streams for processing.

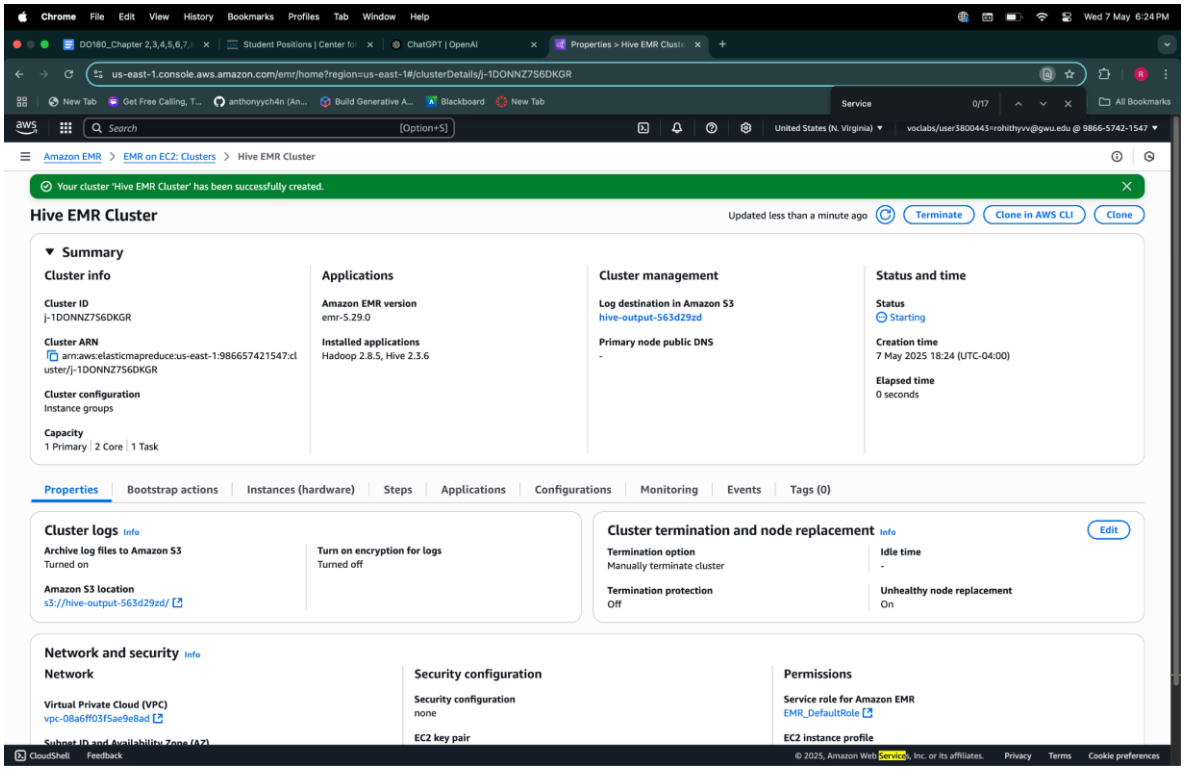
Fig 9.5: End of guided lab



Note: This image shows the end of guided lab Performing ETL on a Dataset by Using AWS Glue

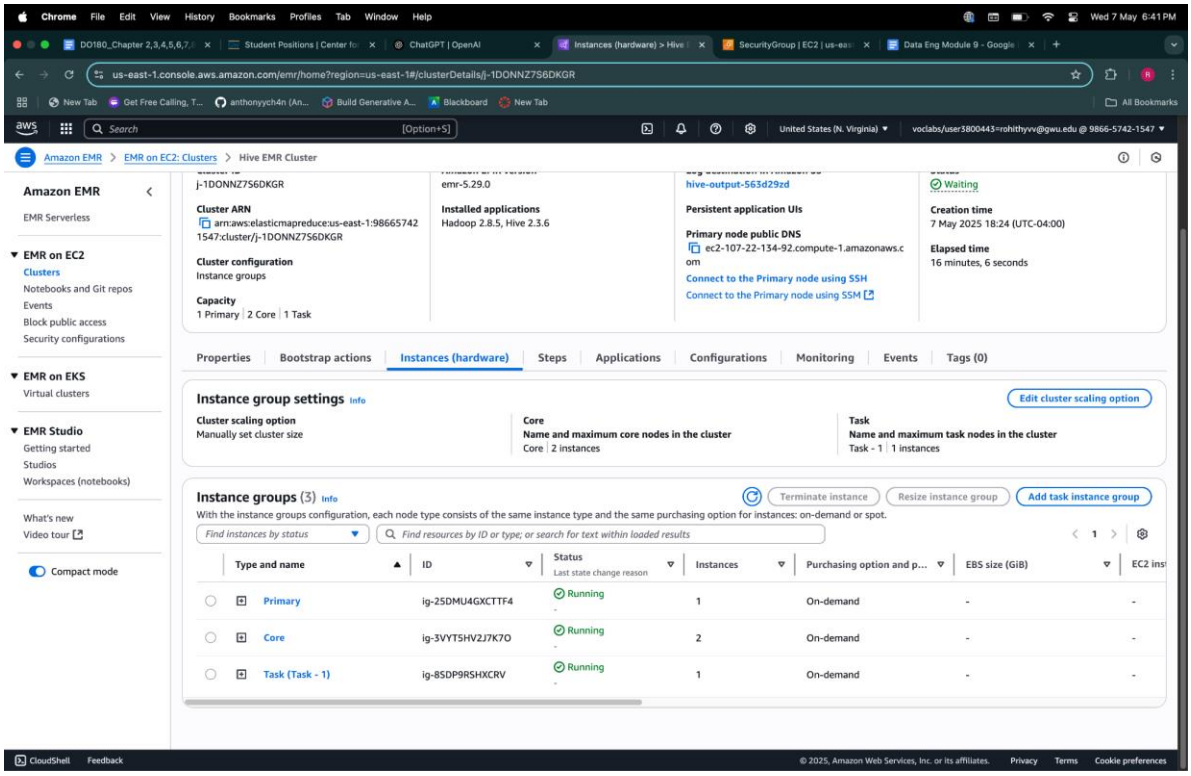
Lab 2: Updating Dynamic Data in Place

Fig 9.6: Hive EMR Cluster Initialization



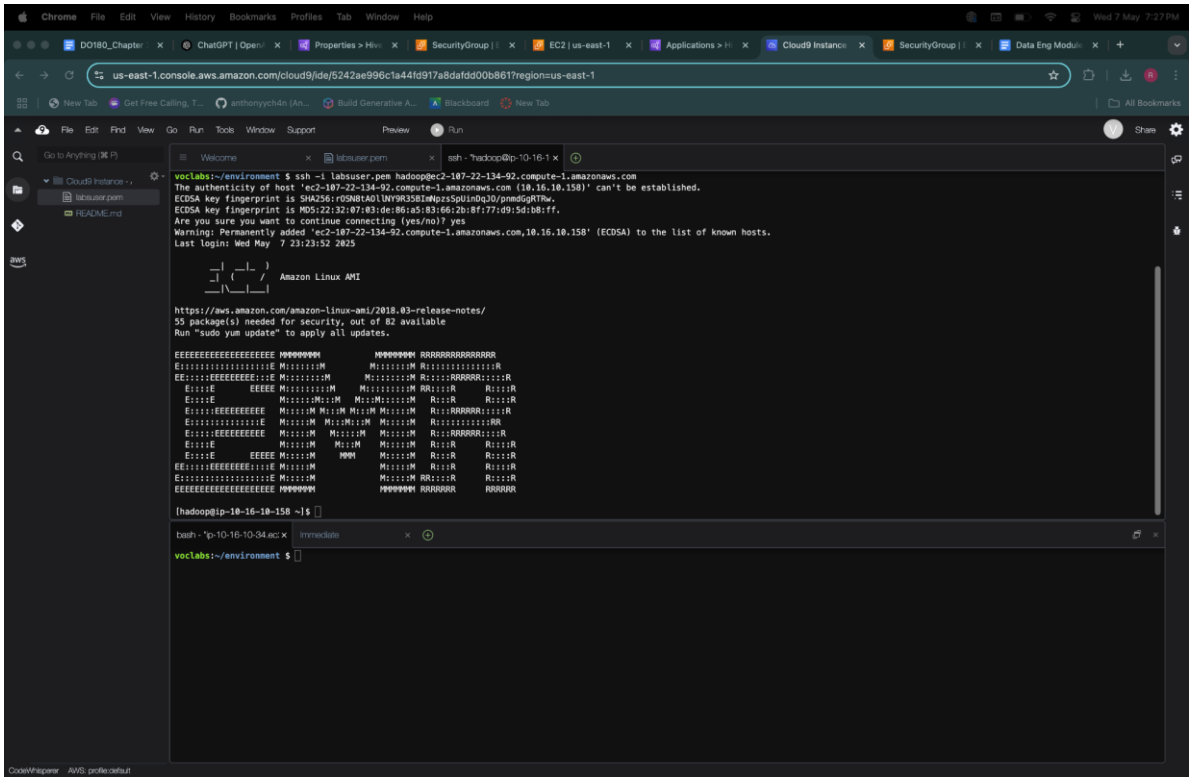
Note: EMR cluster successfully created with two core instances and one task instance, running Hadoop 2.8.5 and Hive 2.3.6.

Fig 9.7: EMR Instance Group Configuration



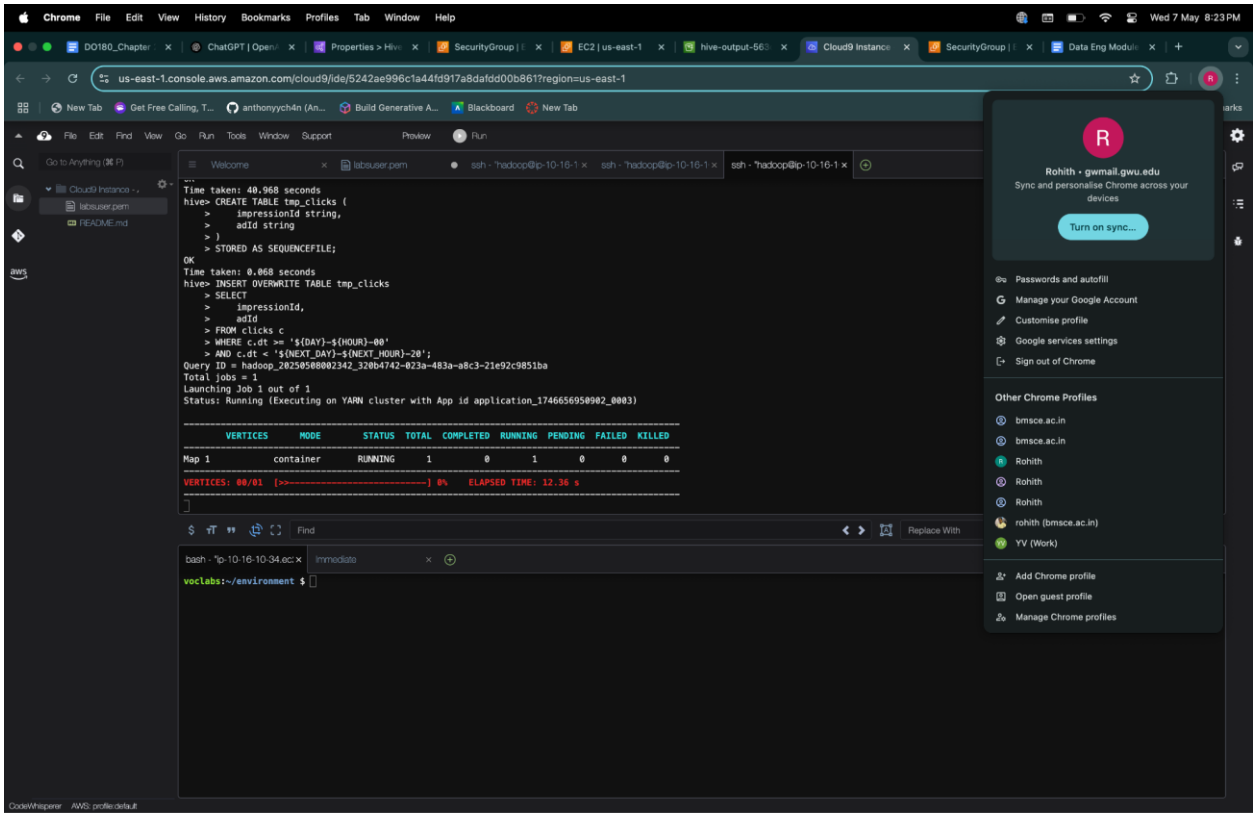
Note: EMR cluster displays instance groups for Primary, Core, and Task nodes, all running on-demand with healthy statuses.

Fig 9.8:SSH Connection to EMR Master Node



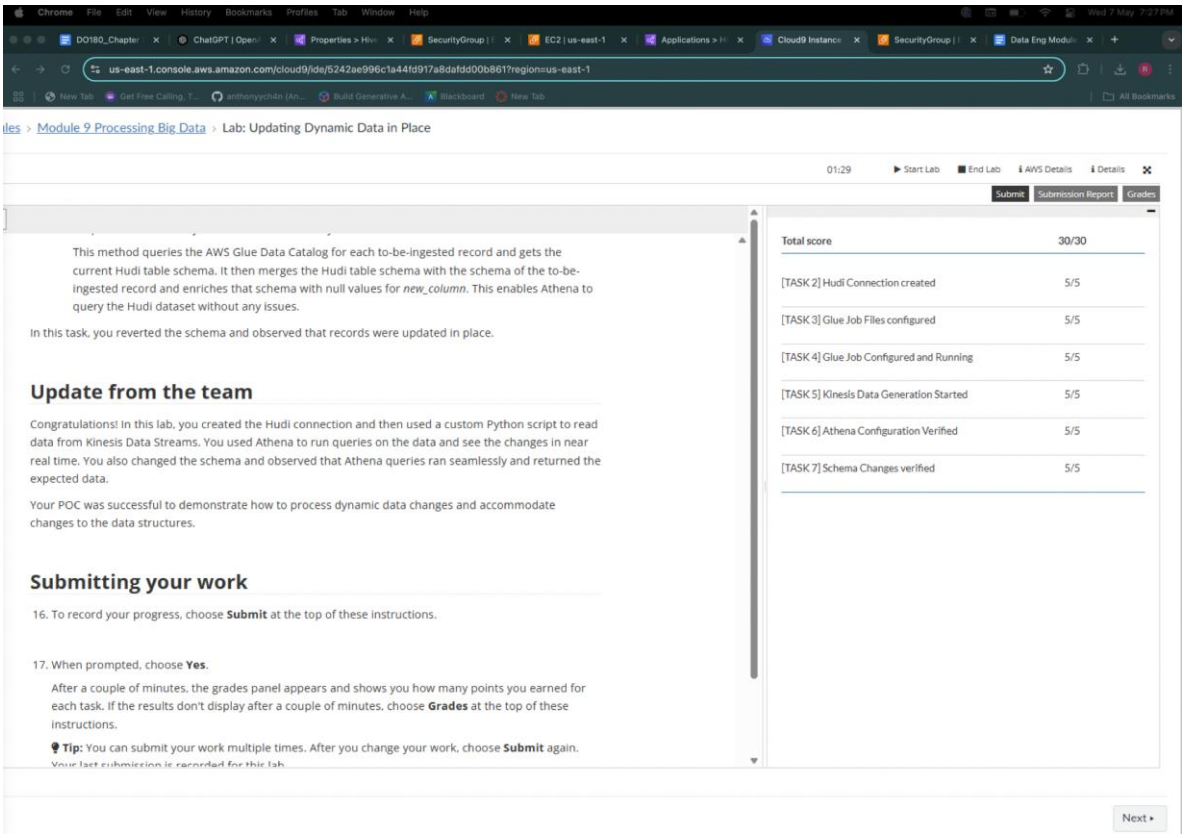
Note: Secure SSH connection established to EMR master node, ready for Hive queries and data processing.

Fig 9.9:Hive Table Creation and Query Execution



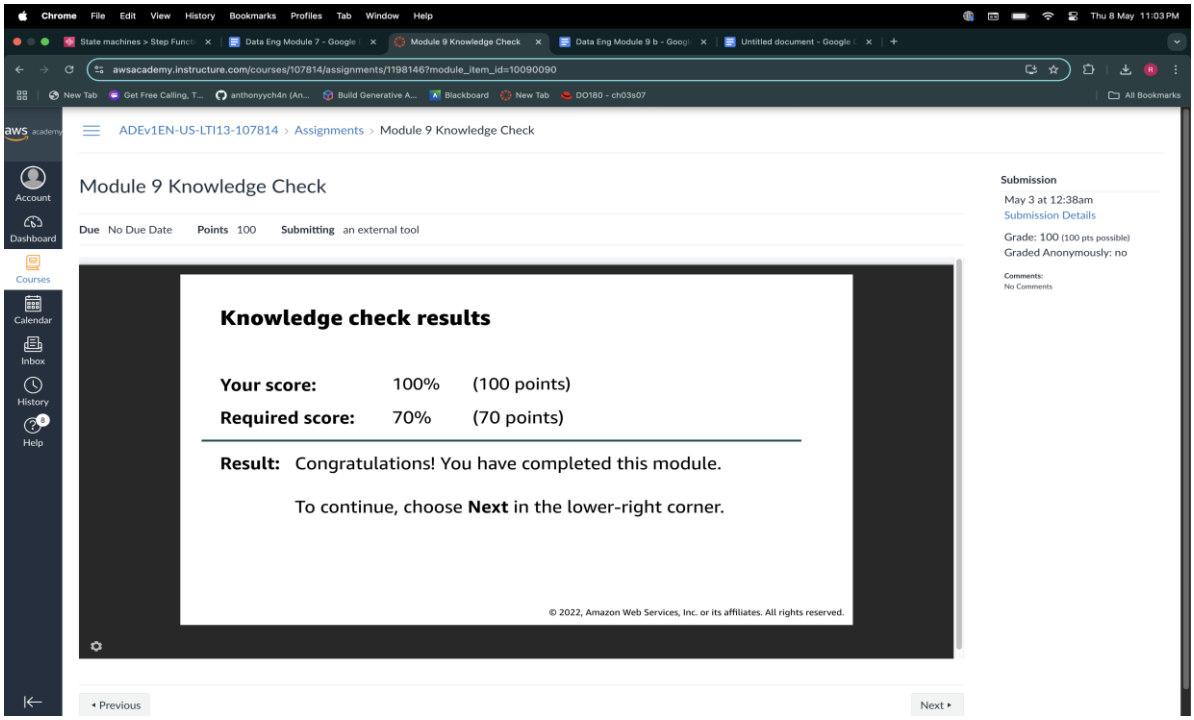
Note: Temporary Hive table `tmp_clicks` created, populated with filtered click data, and query execution initiated.

Fig 9.6: End of guided lab



Note: This image shows the end of guided lab Performing ETL on a Dataset by Using AWS Glue

Fig 9.1: End of module 9



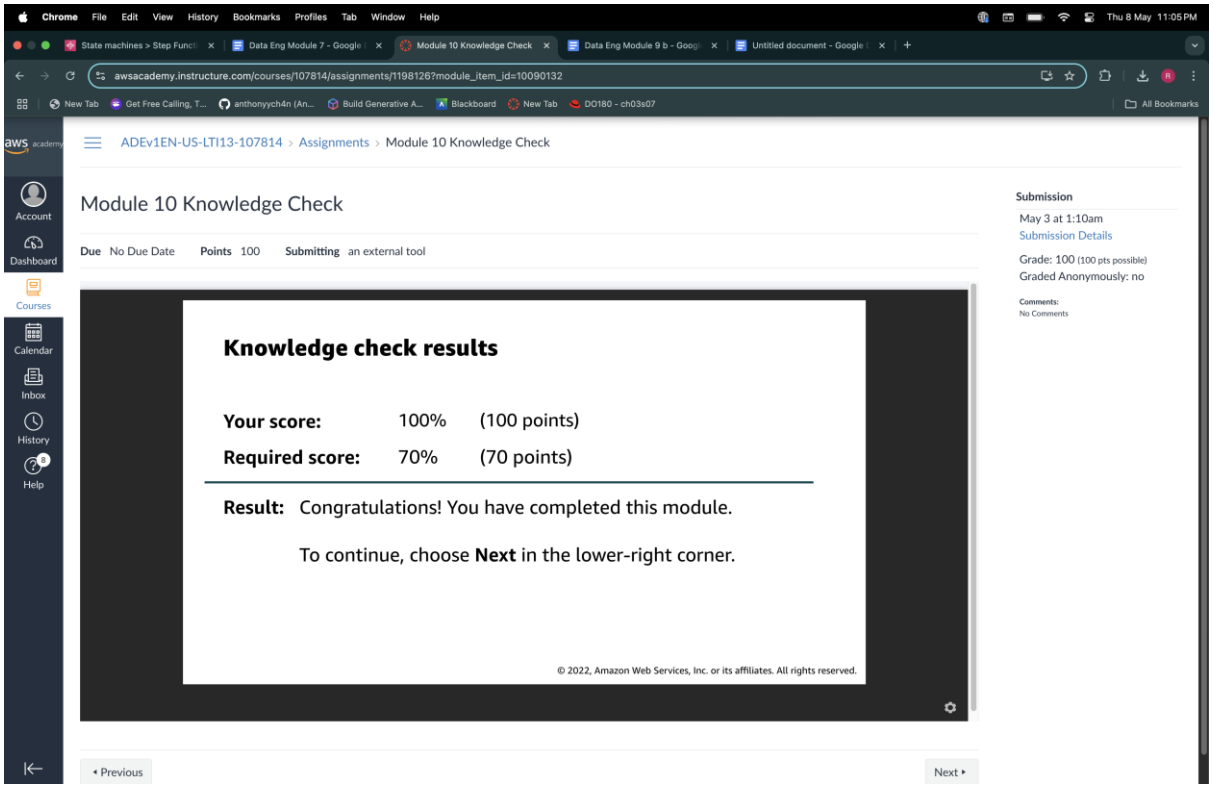
Note: This image shows the completion of module 9

Summary:

- Hadoop processes massive data across many computers using HDFS for storage, YARN for resource management, and MapReduce for data processing.
- MapReduce splits data into chunks (Map) and aggregates results (Reduce), enabling parallel processing and fault tolerance.
- Spark improves over MapReduce with in-memory processing, faster execution, and supports real-time, batch, and machine learning tasks.
- Tools like Hive, Pig, Flink, and Presto simplify querying and analyzing big data through SQL-like interfaces or streaming capabilities.
- Apache Hudi enables real-time data updates, deletes, and incremental processing in data lakes like Amazon S3, supporting use cases like GDPR and ETL.

Module 10: **Processing Data for ML**

Fig 10.1: End of module 10



Note: This image shows the completion of module 10

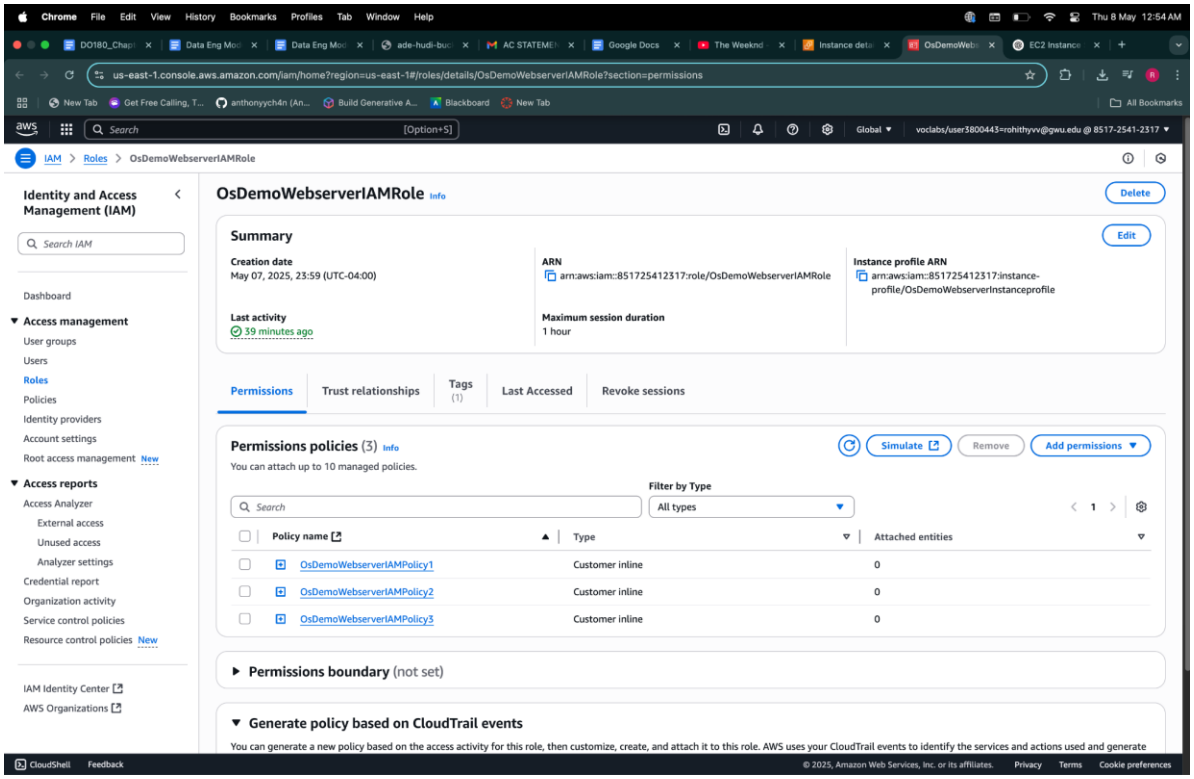
Summary:

- **ML Lifecycle Roles**: Key roles in the machine learning lifecycle include data scientists, AI/ML architects, data engineers, ML engineers, and domain experts, all collaborating across stages like data collection, preprocessing, model development, and deployment.
- **ML Use Case Evaluation**: Before adopting ML, assess if business rules or statistical modeling suffice. ML is suitable when large, quality data is available, and it offers better accuracy or insights than traditional approaches.
- **Preprocessing Techniques**: Strategies like outlier removal, data partitioning, normalization, augmentation, and handling class imbalance are critical for enhancing model training quality and reliability.
- **Model Accuracy Metrics**: Evaluate models using metrics like accuracy, precision, recall, and F1 score, and understand classification outcomes like true/false positives and negatives.
- **MLOps & AWS Tools**: AWS provides tools like SageMaker Studio, Data Wrangler, and SageMaker Processing to support MLOps practices—streamlining model training, versioning, deployment, and monitoring in production.

Module 11: **Analyzing and Visualizing Data**

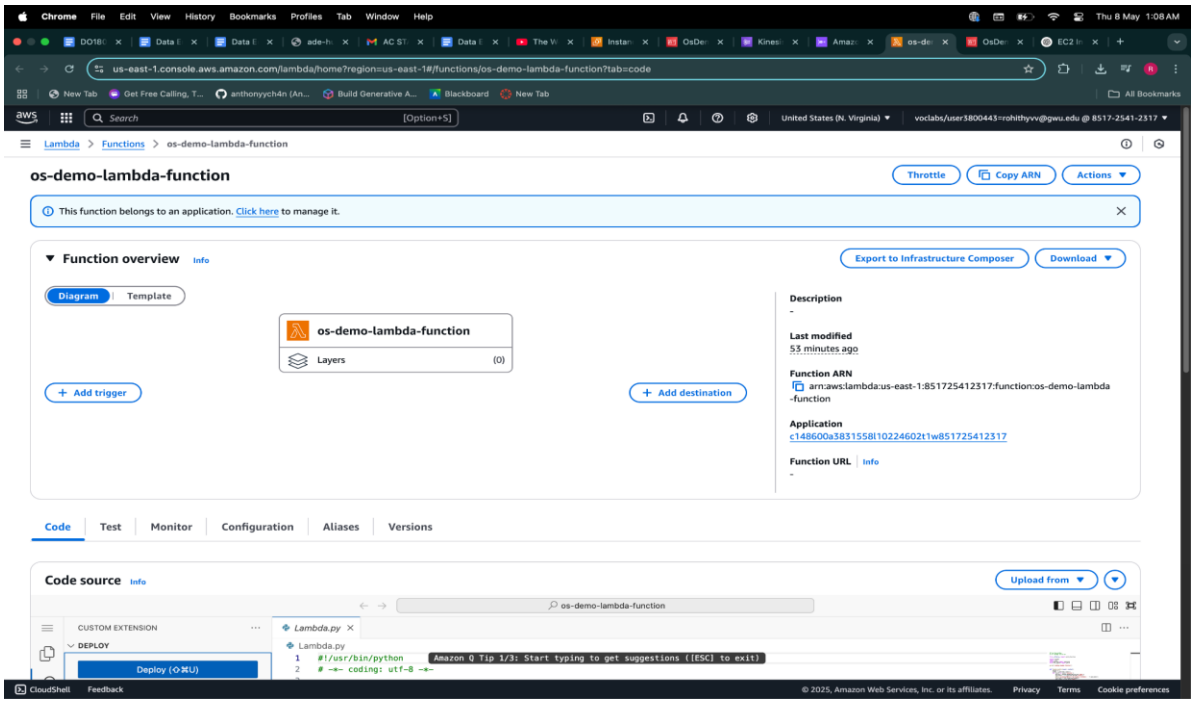
Lab: IAM Role Configuration for Web Server

Fig 11.1: OpenSearch Dashboards



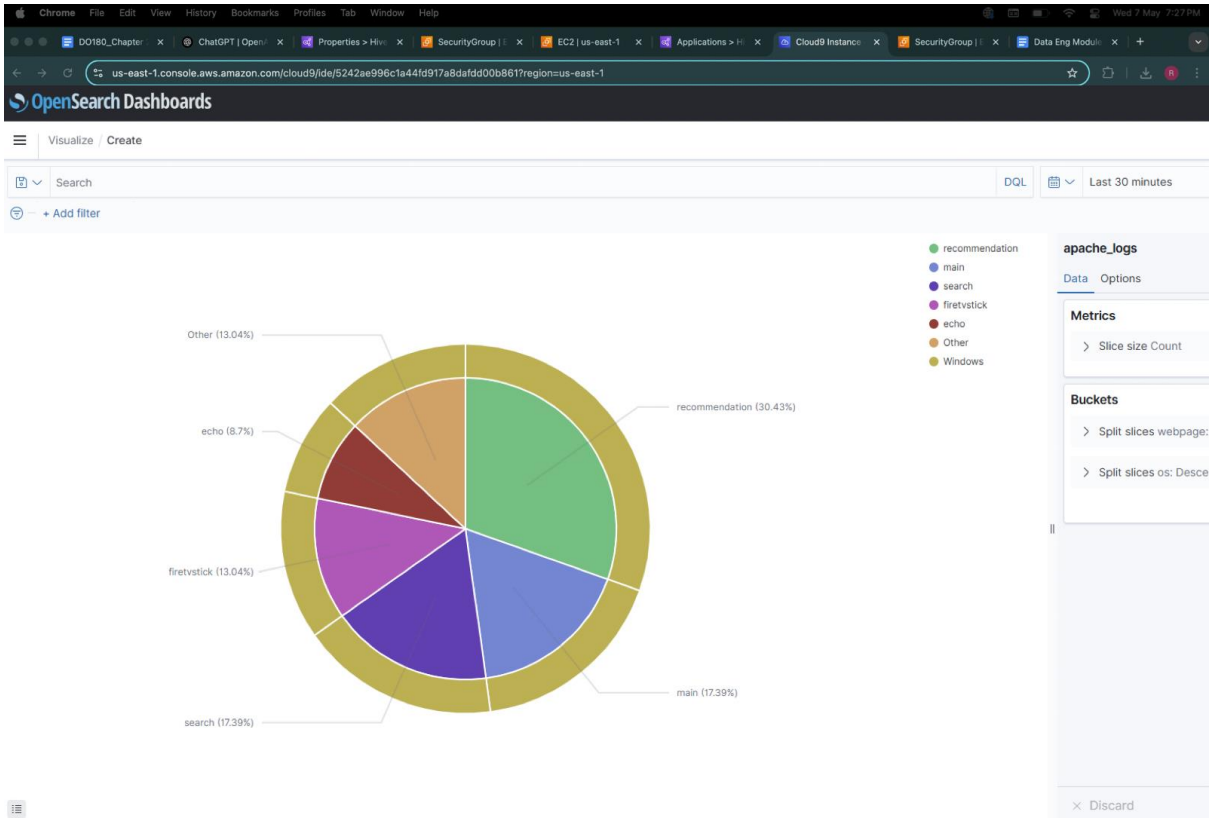
Note: The IAM Role OsDemoWebserverIAMRole is configured with three inline policies for controlling access to AWS services. It is associated with an instance profile for streamlined role-based access.

Fig 11.2: Lambda Function Overview



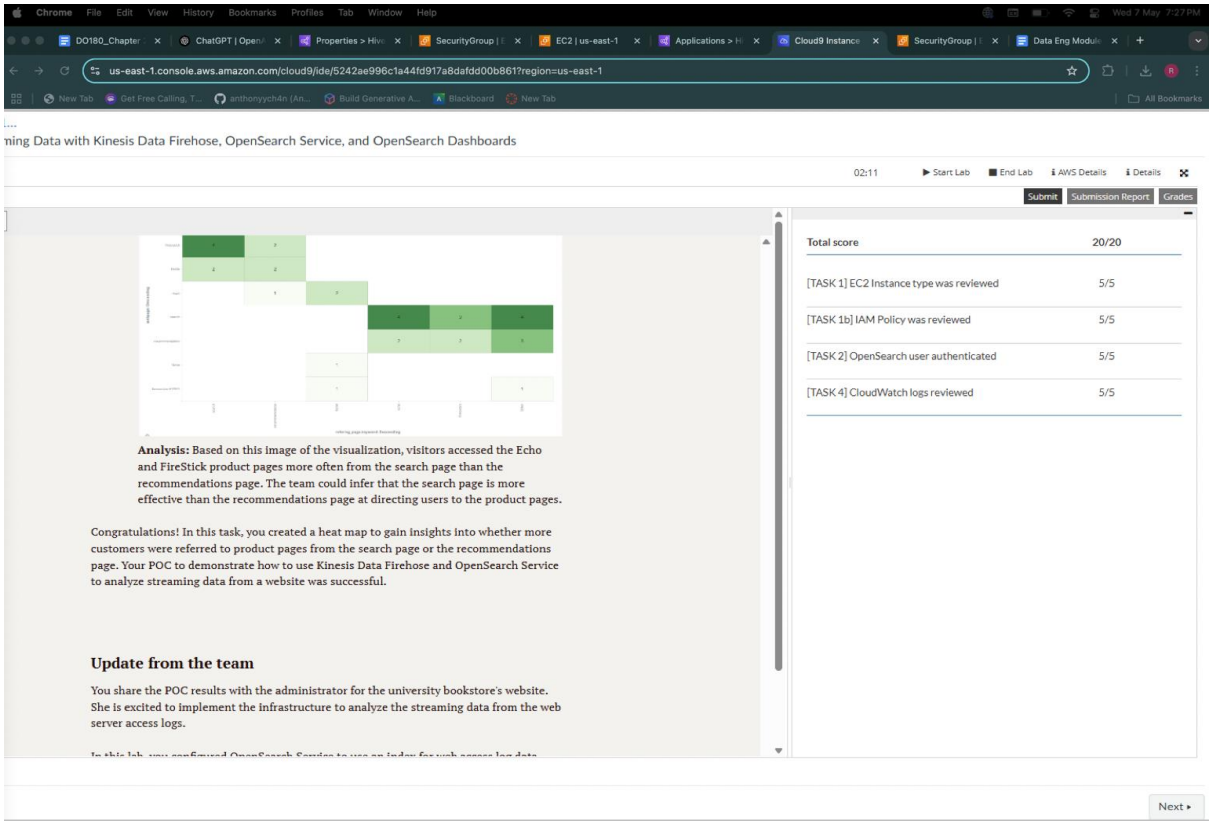
Note: Image shows log in to OpenSearch Dashboards and configure an index for the web server log data that is streamed from Kinesis Data Firehose.

Fig 11.2: Creating a pie chart visualization



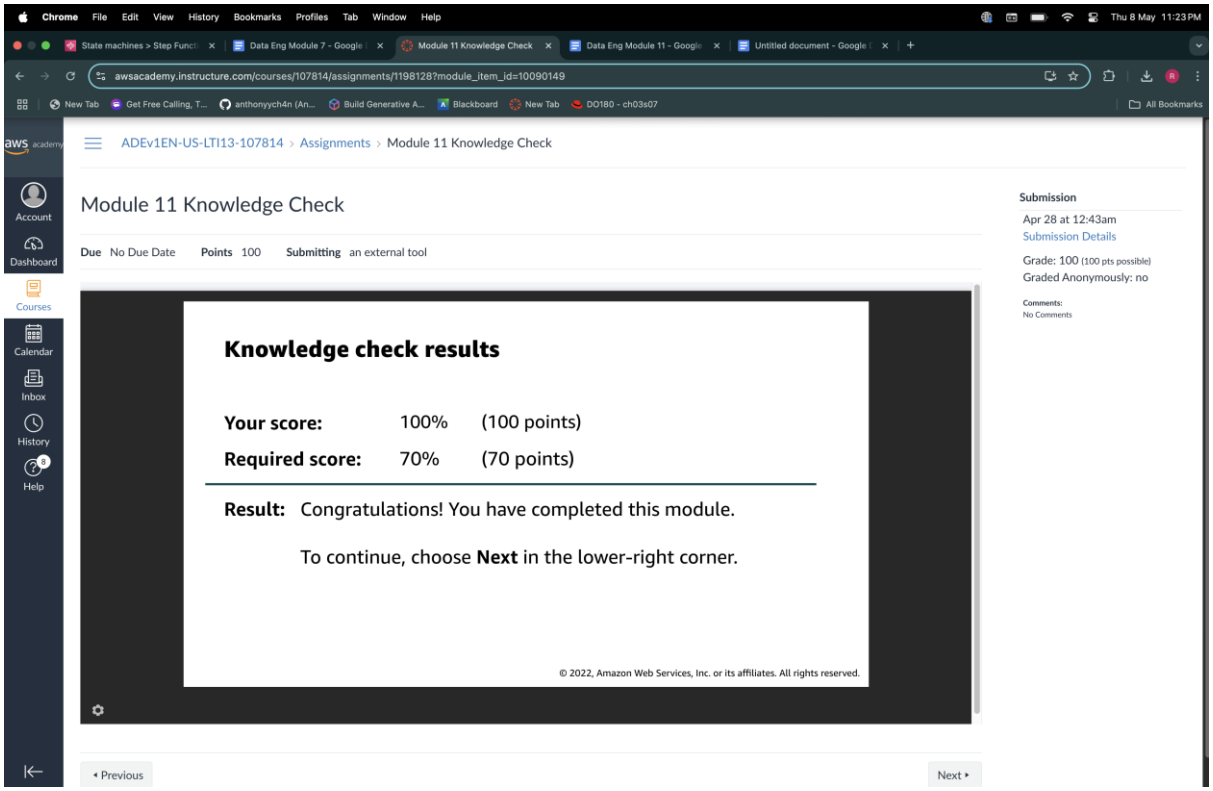
Note: First visualisation will address the question of whether there is a relationship between the products that customers select and the operating system or web browser that they use, created a stacked pie chart to gain insights to answer this question.

Fig 11.3: End of guided lab



Note: This image shows the end of guided lab Analyzing and Visualizing Streaming Data with Kinesis Data Firehose, OpenSearch Service, and OpenSearch Dashboards

Fig 11.4: End of module 11



Note: This image shows the completion of module 11

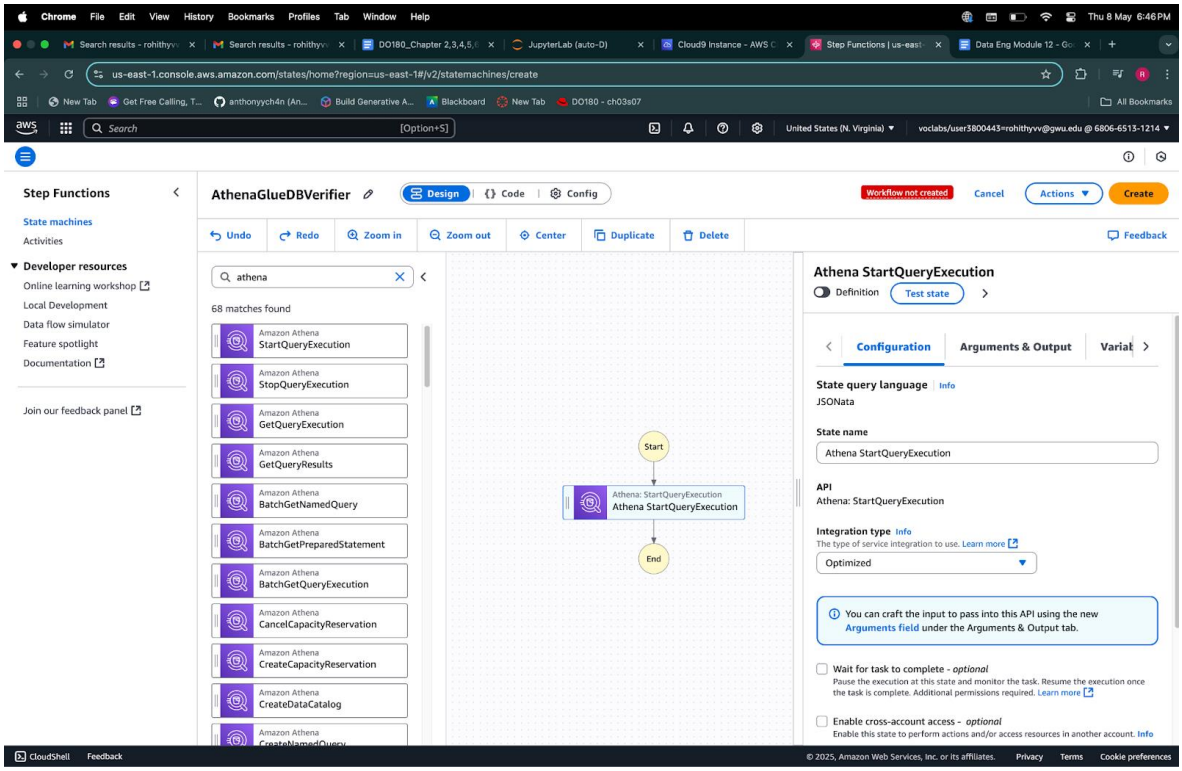
Summary:

- **Amazon Athena**: A serverless service that allows interactive analysis of data using SQL; you can analyze data directly from Amazon S3 and start querying instantly.
- **Amazon QuickSight**: A serverless business intelligence service used to build dashboards and visualizations for business analytics, enabling insights through easy-to-use visuals.
- **Amazon OpenSearch Service**: A fully managed service for operational analytics; it enables search, filtering, aggregation, and real-time data visualization.
- **OpenSearch Engine**: Open-source search and analytics engine powering the Amazon OpenSearch Service, designed for scalable and easy-to-operate cluster deployments.
- **Use Case Differentiation**: Athena is ideal for SQL-based queries, QuickSight for business dashboards, and OpenSearch Service for real-time operational data exploration and visualization.

Module 12: **Automating the Pipeline**

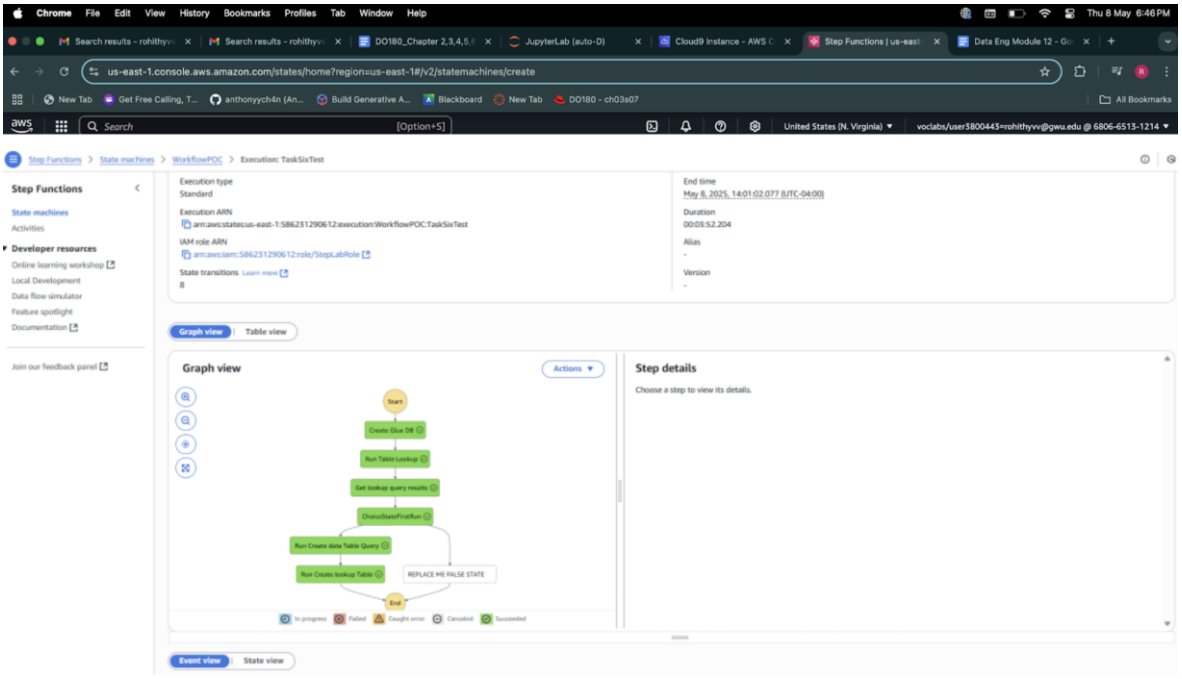
Lab: Building and Orchestrating ETL Pipelines by Using Athena and Step Functions

Fig 12.1: AWS Step Functions - Athena Query Execution Workflow



Note: The state machine `AthenaGlueDBVerifier` is designed to execute an Athena query using `Athena StartQueryExecution`. This workflow initiates a query in Athena and progresses from Start to End, providing automated query execution as part of a larger data verification or processing pipeline.

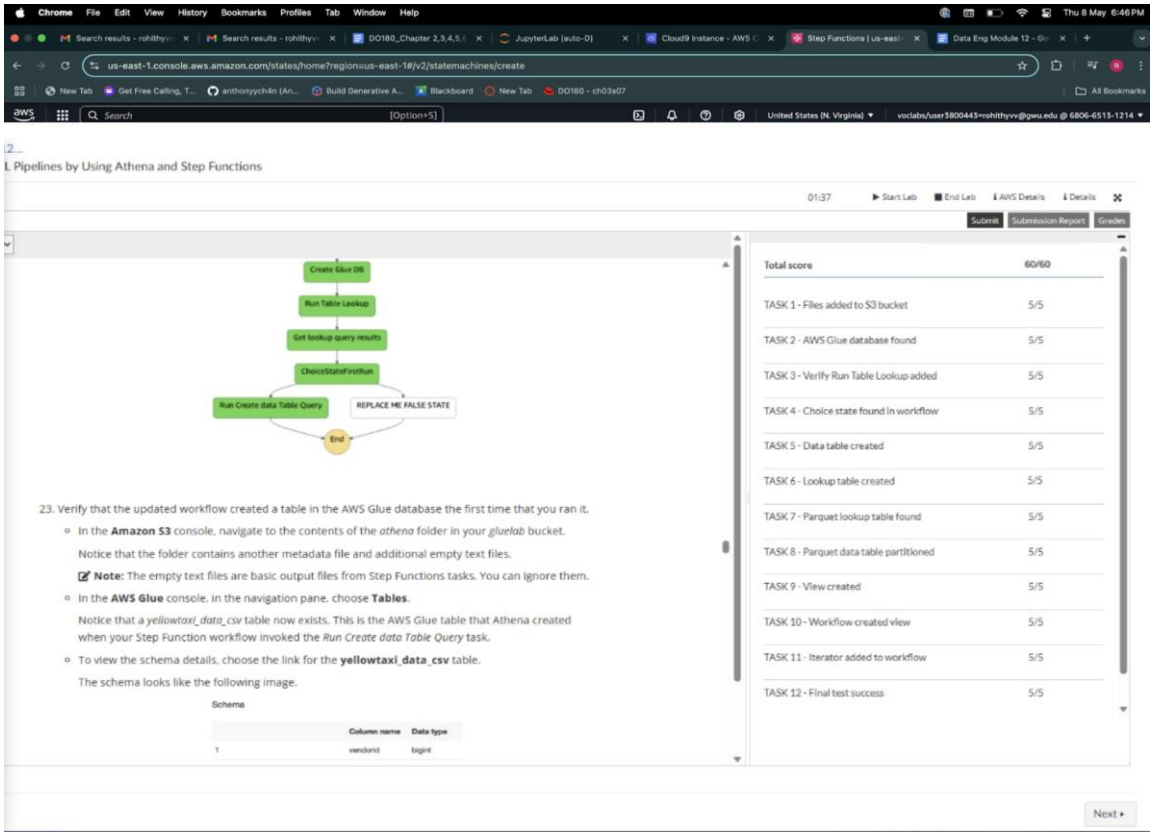
Fig 12.2: Creating the AWS Glue table for location lookup data



Note: Created another AWS Glue table by updating and running the Step Functions workflow. The new table will reference the taxi _zone_lookup.csv source data file in Amazon S3. After creating this table, will be able to

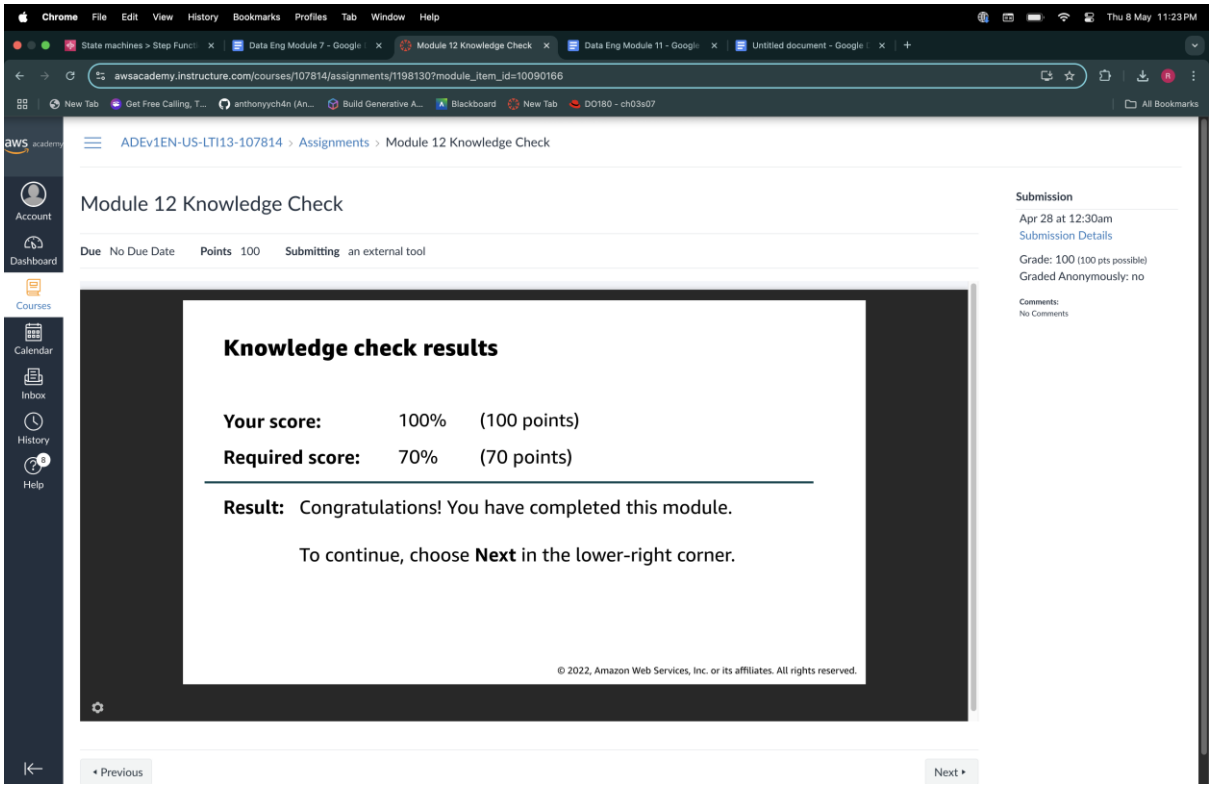
join the yellow taxi data table with the lookup table in a later task. Joining the two tables will help to make more sense of the data.

Fig 12.3: End of guided lab



Note: This image shows the end of guided lab Analyzing and Visualizing Streaming Data with Kinesis Data Firehose, OpenSearch Service, and OpenSearch Dashboards

Fig 12.4: End of module 12



Note: This image shows the completion of module 12

Summary:

- **CI/CD Overview**: Continuous Integration (CI) involves frequently merging code into a shared repository, while Continuous Delivery (CD) automates the building, testing, and preparing of code for production deployment.
- **DevOps Philosophy**: DevOps blends cultural practices and tools to accelerate application delivery, enabling faster product iteration compared to traditional development methods.
- **AWS Step Functions**: A service that uses visual workflows to coordinate distributed systems and microservices, handling retries, state tracking, and integration with services like Amazon Athena.
- **State Types in Step Functions**: Workflow states include tasks (units of work), choices (branching), parallel (concurrent paths), wait (delays), map (iteration), and terminal states like succeed or fail.
- **Data Pipeline Execution**: After designing the workflow visually, AWS generates JSON that, when deployed, orchestrates logic using services like AWS Lambda, Glue, S3, and SNS.