

AWS Academy Data Engineering [107814]

Rohith Yellambalase Vijayakumar

G26286080

The George Washington University

Department of Computer Science,

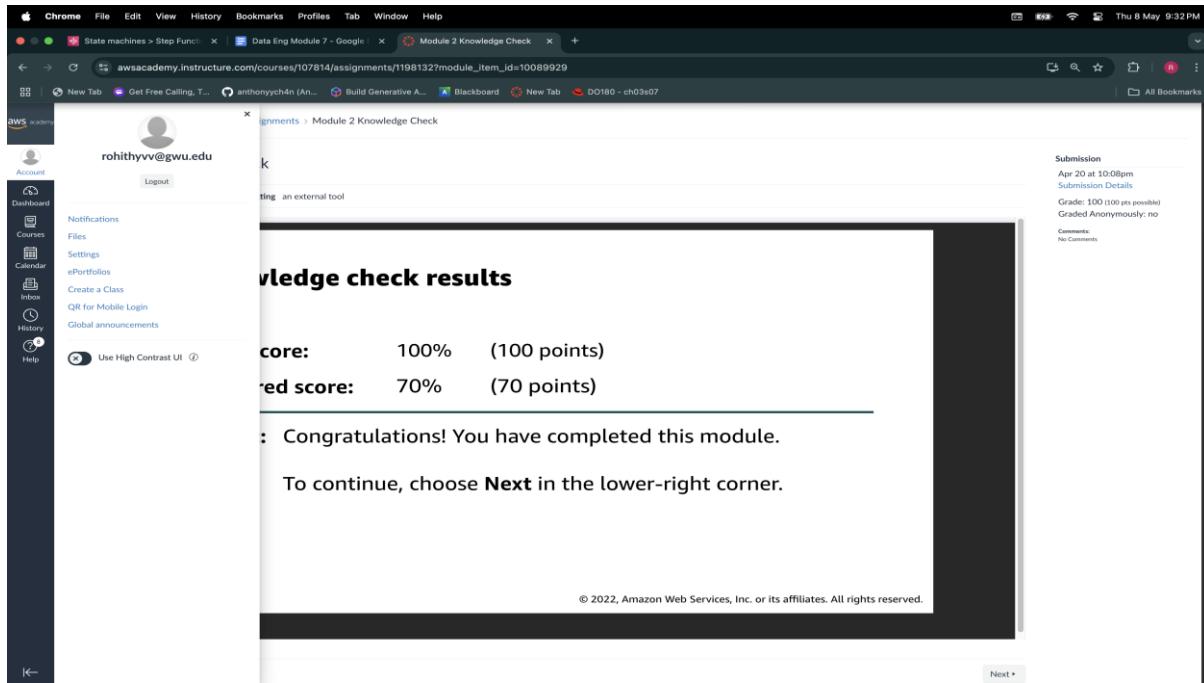
CSCI\_6442\_10: Database Systems-II

Prof. Rolando Fernandez

**30<sup>th</sup> April 2025**

## Module 2 Data-Driven Organizations

*Fig 2.1: End of module 2*



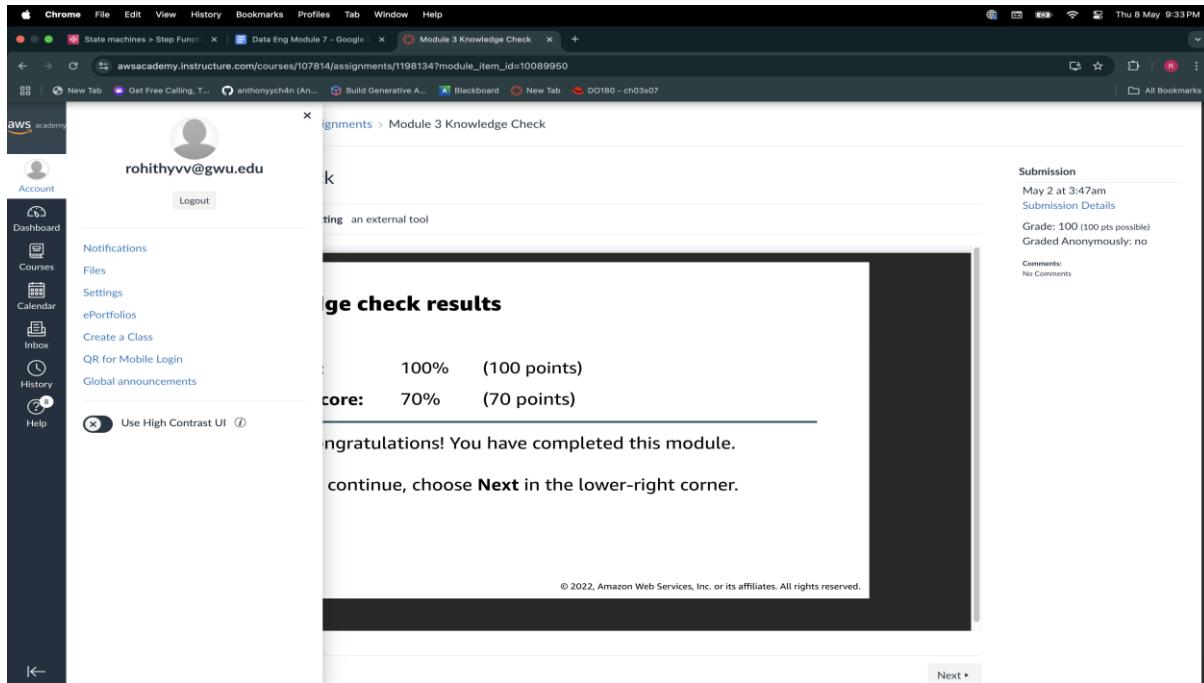
*Note: This image shows the completion of module 2*

### 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

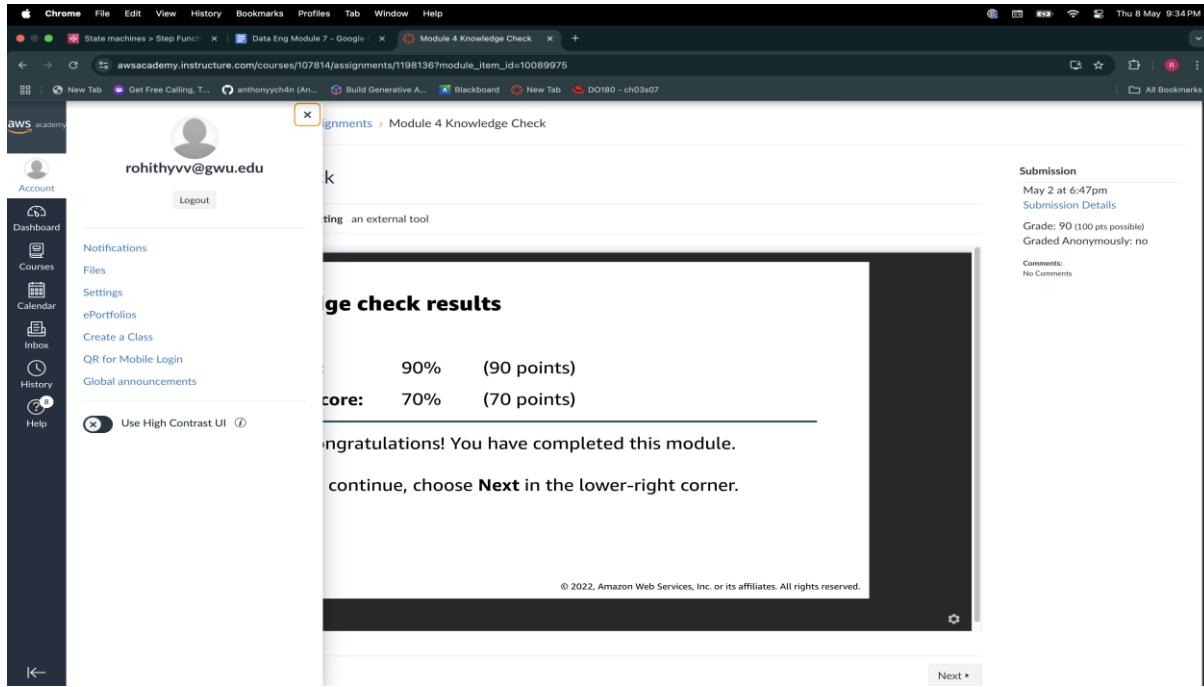
#### Summary:

- **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.*

```

1 AWSTemplateFormatVersion: 2010-09-09
2 Resources:
3   AthenaNamedQuery:
4     Type: AWS::Athena::NamedQuery
5     Properties:
6       Name: "taxidolo"
7       Description: "A query that selects all fares over $100.00 (US)"
8       Name: "FaresOver100DollarsUS"
9     QueryString:
10       SELECT distance, paytype, fare, tip, tolls, surcharge, total
11       FROM yellow
12       WHERE total >= 100.0
13       ORDER BY total DESC
14     Outputs:
15       AthenaNamedQueryQuery:
16         Value: !Ref AthenaNamedQuery
    
```

```

aws -l ip-10-16-10-55.ec2.x immediate
voclabs:~/environment $ !KcKxAKTAVRIVTLVKNZTKX6
voclabs:~/environment $ !SAKxKvTB8THR73fpvIEv3/MwTFB180e0r7v5Keuag
voclabs:~/environment $ AWS_ACCESS_KEY_ID=$AK AWS_SECRET_ACCESS_KEY=$AK aws athena get-named-query --named-query-id $NQ
{
  "NamedQuery": {
    "Name": "FaresOver100DollarsUS",
    "Description": "A query that selects all fares over $100.00 (US)",
    "Database": "taxidolo",
    "QueryString": "SELECT distance, paytype, fare, tip, tolls, surcharge, total FROM yellow WHERE total >= 100.0 ORDER BY total DESC\\n",
    "NamedQueryId": "68fcac9c-9ade-45bd-97f5-9a88cccf59e7",
    "WorkGroup": "primary"
  }
}
voclabs:~/environment $ 
    
```

*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*

| Total score                                  | 45/45 |
|--|-------|
| [Task 1a] Database created                   | 5/5   |
| [Task 1b] Database table created             | 5/5   |
| [Task 1c] Database table previewed           | 5/5   |
| [Task 2] Database table jan exists           | 5/5   |
| [Task 4] Student queried using views         | 5/5   |
| [Task 5a] Student tested the query in Athena | 5/5   |
| [Task 5b] CloudFormation stack created       | 5/5   |
| [Task 5c] CreateNamedQuery event recorded    | 5/5   |
| [Task 7] GetNamedQuery event recorded        | 5/5   |

**Lab complete**  
Congratulations! You have completed the lab.

34. To find detailed feedback about your work, choose **Submission Report**.

35. At the top of this page, choose **End Lab**, and then choose **Yes** to confirm that you want to end the lab.  
A message panel indicates that the lab is terminating.

36. To close the panel, choose **Close** in the upper-right corner.

**Additional resources**  
For more information about the services and concepts covered in this lab, see the following resources:

- Amazon Athena Documentation
- Getting Started with AWS Glue
- Presto Documentation
- DML Queries, Functions, and Operators
- Amazon Athena Pricing
- Using a SerDe
- Columnar Storage Formats

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

*Fig 4.4 : End of module 4*

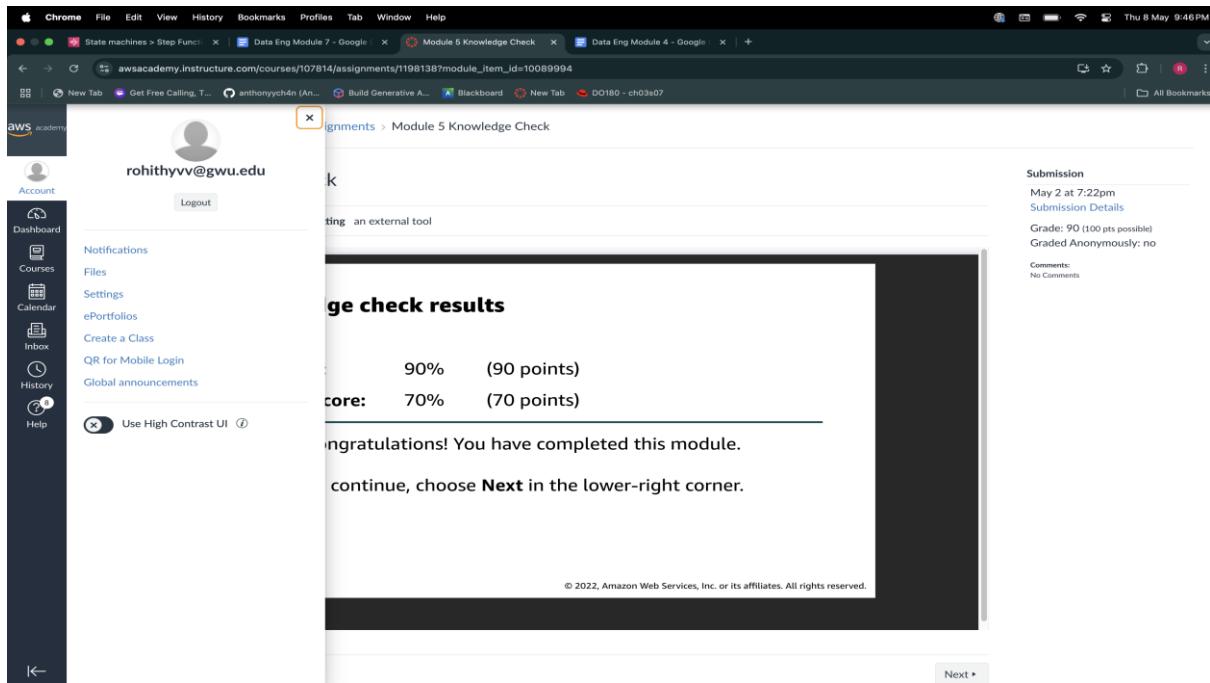
The screenshot shows a browser window with the AWS Academy Data Engineering module 4 knowledge check results. The page header includes the AWS logo and the user's email address, rohitvyyv@gwu.edu. On the left, there is a sidebar with various navigation links such as Dashboard, Courses, Calendar, Inbox, History, and Help. The main content area displays the 'Module 4 Knowledge Check' results. It shows two scores: 'Grade: 90% (90 points)' and 'Core: 70% (70 points)'. Below these scores, a message reads: 'Congratulations! You have completed this module. To continue, choose Next in the lower-right corner.' In the top right corner, there is a 'Submission' section showing the date and time of submission (May 2 at 6:47pm), the grade (90), and the fact that it was graded anonymously. There are also 'Comments' and 'No Comments' sections.

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

The screenshot shows a browser window with multiple tabs open. The active tab is titled "Module 6 Knowledge Check" from the AWS Academy website. The main content area displays the results of a knowledge check. It shows a score of 90% (90 points) out of 100, with a required score of 70%. A message below the scores says, "Result: Congratulations! You have completed this module." At the bottom, it says, "To continue, choose Next in the lower-right corner." On the left side, there is a sidebar with icons for Account, Dashboard, Courses, Calendar, Inbox, History, and Help. On the right side, there is a sidebar with submission details: Grade: 90 (100 pts possible), Graded Anonymously: no, and Comments: No Comments. The date and time of submission are also listed.

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

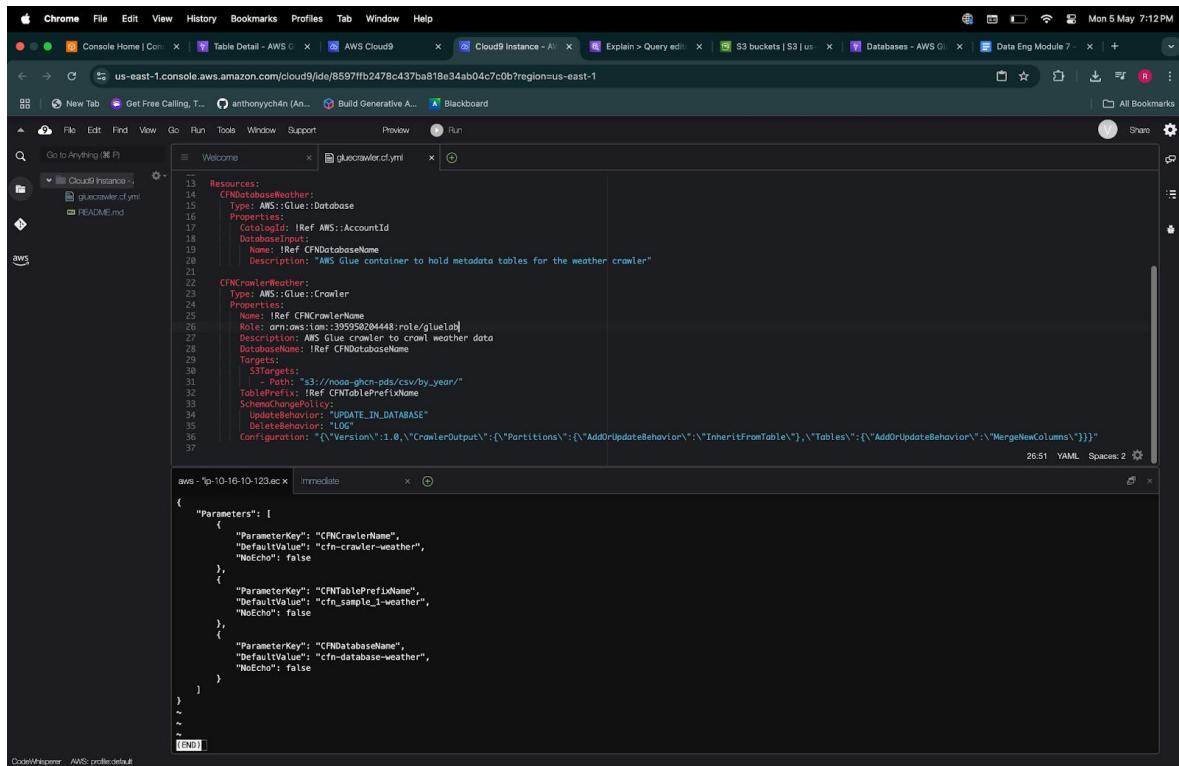
*Fig 7.1: Analyzing Historical Temperature Data with Amazon Athena*

The screenshot shows the AWS Lambda function configuration for the 'lambda-athena' function. It includes tabs for 'Overview', 'Code', 'Logs', 'Environment', 'Permissions', and 'Metrics'. The 'Environment' tab displays environment variables like AWS\_REGION, AWS\_DEFAULT\_REGION, and AWS\_LAMBDA\_FUNCTION\_NAME. The 'Logs' tab shows log entries from CloudWatch Logs.

| #  | Year | Max                |
|----|------|--------------------|
| 1  | 1950 | 16.734594971312752 |
| 2  | 1951 | 16.806771769217043 |
| 3  | 1952 | 17.280697862535543 |
| 4  | 1953 | 17.919086201985372 |
| 5  | 1954 | 17.478765473399562 |
| 6  | 1955 | 17.104405749839186 |
| 7  | 1956 | 17.15493545441965  |
| 8  | 1957 | 17.403568467007226 |
| 9  | 1958 | 17.30067325208722  |
| 10 | 1959 | 17.379008960362224 |
| 11 | 1960 | 16.93149079351014  |
| 12 | 1961 | 17.624064898282302 |
| 13 | 1962 | 17.54085890725994  |
| 14 | 1963 | 17.71521783661604  |
| 15 | 1964 | 17.601848333210782 |

*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*



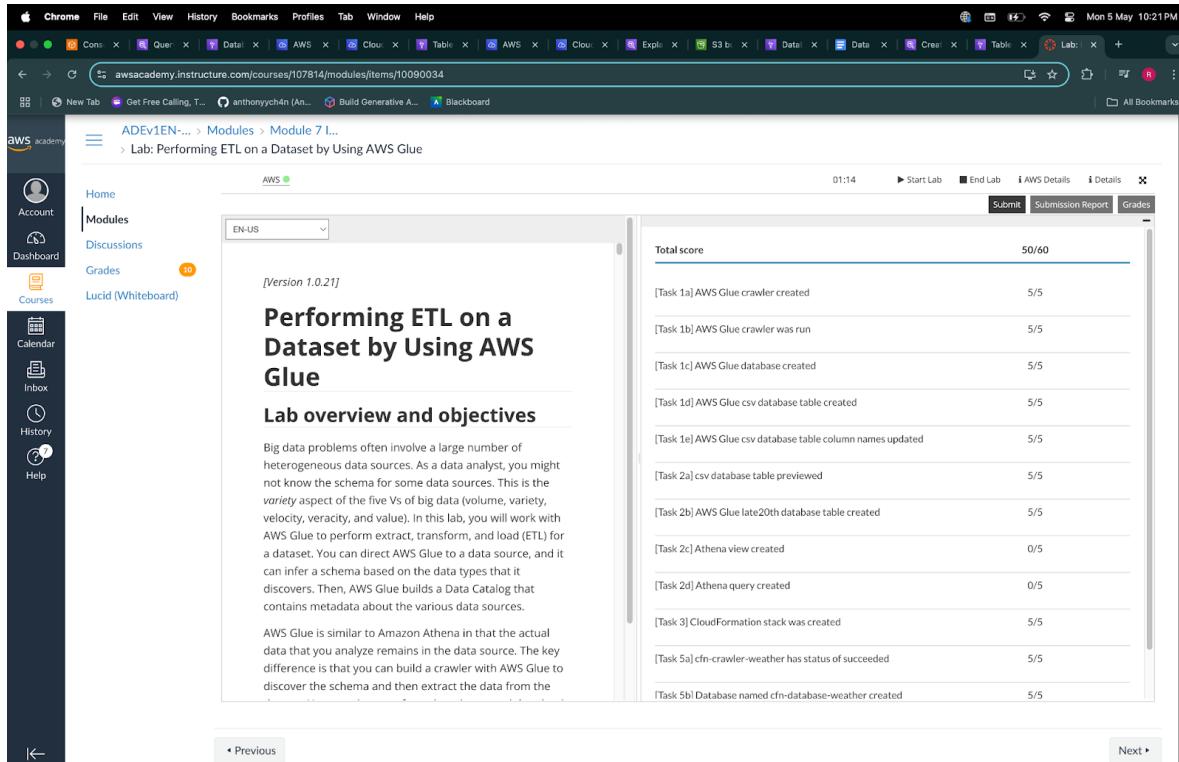
```

Resources:
  CFNDatabaseWeather:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: "AWS Glue container to hold metadata tables for the weather crawler"
  CFNCrawlerWeather:
    Type: AWS::Glue::Crawler
    Properties:
      Name: !Ref CFNCrawlerName
      Role: arn:aws:iam::395990204448:role/gluelab
      Description: AWS Glue crawler to crawl weather data
      DatabaseName: !Ref CFNDatabaseName
      Targets:
        S3Targets:
          Path: "s3://noaa-ghcn-pds/csv/by_year/"
      TablePrefix: !Ref CFNTablePrefixName
      SchemaUpdateStrategy: "UPDATE_IN_DATABASE"
      DeleteBehavior: "LOG"
      Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":\"MergeNewColumns\"}}}"
  26:51 YAML Spaces: 2
  
```

The screenshot shows a browser window with multiple tabs open, including 'Cloud Instance - AI', 'AWS Cloud9', 'Table Detail - AWS Glue', 'Cloud9 Instance - AI', 'Explain > Query edit...', 'S3 buckets | S3 | us', 'Databases - AWS Glue', and 'Data Eng Module 7...'. The main content area displays CloudFormation YAML code for creating a database and a crawler. The database is named 'CFNDatabaseWeather' and has a description of 'AWS Glue container to hold metadata tables for the weather crawler'. The crawler is named 'CFNCrawlerWeather' and is configured to crawl data from 's3://noaa-ghcn-pds/csv/by\_year/'. The configuration section includes a schema update strategy of 'UPDATE\_IN\_DATABASE' and a delete behavior of 'LOG'.

*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



The screenshot shows the AWS Academy course interface for module 7. The left sidebar shows navigation links for Home, Modules, Discussions, Grades, and Lucid (Whiteboard). The main content area displays a lab titled 'Lab: Performing ETL on a Dataset by Using AWS Glue'. The lab title is 'Performing ETL on a Dataset by Using AWS Glue'. Below it is a 'Lab overview and objectives' section. The right side of the screen shows a detailed grading breakdown for the lab tasks:

| Task   | Score |
|--|-------|
| [Task 1a] AWS Glue crawler created                         | 5/5   |
| [Task 1b] AWS Glue crawler was run                         | 5/5   |
| [Task 1c] AWS Glue database created                        | 5/5   |
| [Task 1d] AWS Glue csv database table created              | 5/5   |
| [Task 1e] AWS Glue csv database table column names updated | 5/5   |
| [Task 2a] csv database table previewed                     | 5/5   |
| [Task 2b] AWS Glue late20th database table created         | 5/5   |
| [Task 2c] Athena view created                              | 0/5   |
| [Task 2d] Athena query created                             | 0/5   |
| [Task 3] CloudFormation stack was created                  | 5/5   |
| [Task 5a] cfn-crawler-weather has status of succeeded      | 5/5   |
| [Task 5b] Database named cfn-database-weather created      | 5/5   |

The total score is 50/60.

*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

**Knowledge check results**

**Your score:** 90% (90 points)  
**Required score:** 70% (70 points)

**Result:** Congratulations! You have completed this module.

To continue, choose **Next** in the lower-right corner.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

*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*

The screenshot shows the AWS EC2 Security Groups console. A success message at the top states: "Security group (sg-072f02d6bb053c91d | Redshift security group) was created successfully". Below it, the "sg-072f02d6bb053c91d - Redshift security group" details are shown. The "Inbound rules" tab is selected, displaying one rule: "sgr-070a57418ba5f8be4" (IP version IPv4, Type Redshift, Protocol TCP, Port range 5439, Source 0.0.0.0/0).

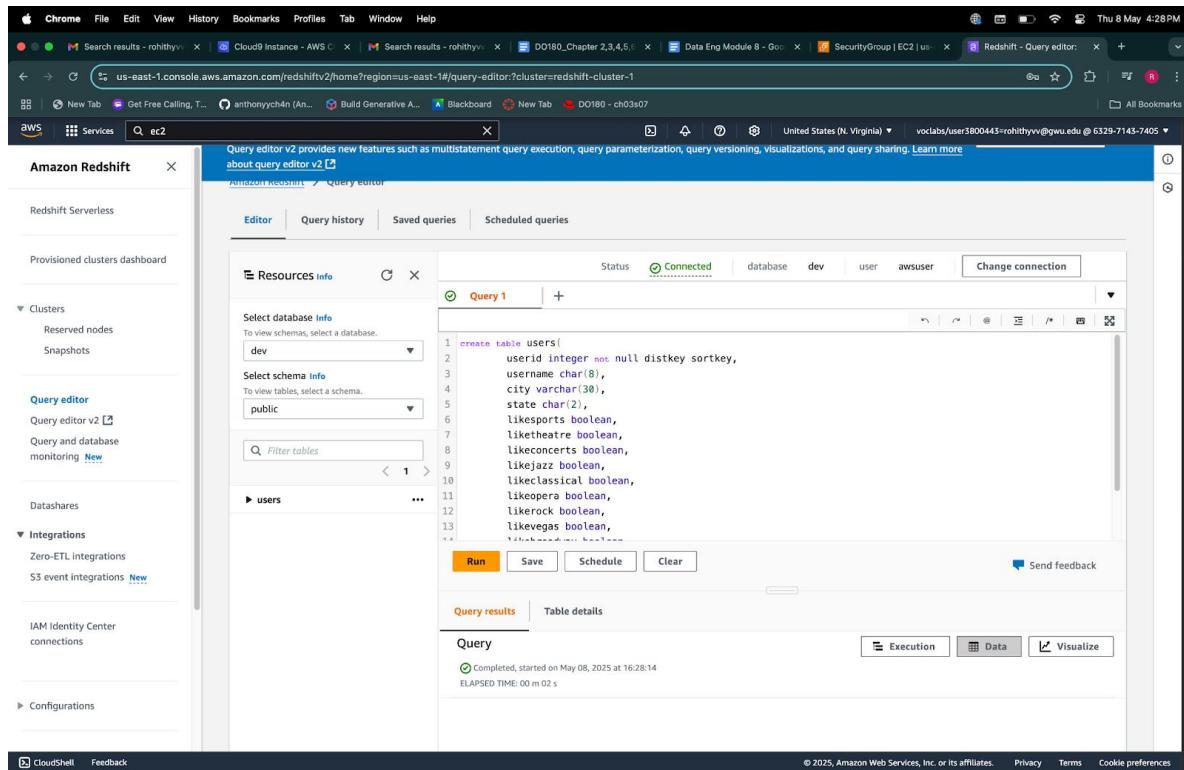
*Note: Created a security group for the cluster so that other resources, such as Amazon EC2 instances, can access it through port 5439. And then added the cluster to the VPC by associating it with the security group created.*

*Fig 8.2: Redshift Cluster IAM Role Association*

The screenshot shows the "Cluster permissions" section of the Amazon Redshift Cluster details page. It displays the "Associated IAM roles" table, which includes a single row for "MyRedshiftRole" (Status: in-sync). The "Granted accounts" section below shows no granted accounts.

*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



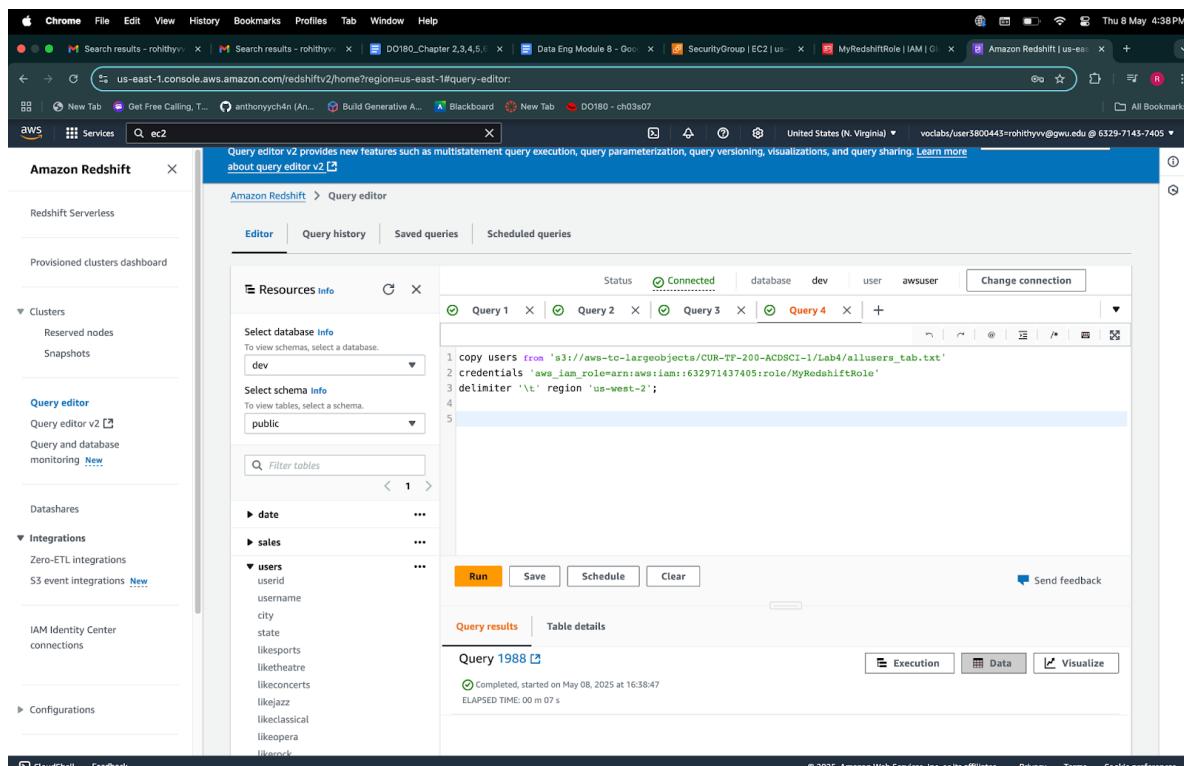
The screenshot shows the Amazon Redshift Query Editor interface. On the left, the navigation sidebar includes 'Amazon Redshift', 'Redshift Serverless', 'Provisioned clusters dashboard', 'Clusters' (with 'Reserved nodes' and 'Snapshots' sub-options), 'Query editor' (selected), 'Query editor v2', 'Query and database monitoring', 'Datastores', 'Integrations' (with 'Zero-ETL integrations' and 'S3 event integrations' sub-options), 'IAM Identity Center connections', and 'Configurations'. The main area has tabs for 'Editor', 'Query history', 'Saved queries', and 'Scheduled queries'. The 'Editor' tab is active, showing a 'Resources info' panel with dropdowns for 'Select database info' (set to 'dev') and 'Select schema info' (set to 'public'). Below this is a table structure for 'users' with columns: userid (integer), username (char(8)), city (varchar(30)), state (char(2)), likesports (boolean), liketheatre (boolean), likeconcerts (boolean), likejazz (boolean), likeclassical (boolean), likeopera (boolean), likerock (boolean), and likevegas (boolean). The 'Status' bar at the top right shows 'Connected' to 'dev', 'user awuser', and 'awsuser'. The 'Query results' section shows the query was completed on May 08, 2025, at 16:28:14, with an elapsed time of 00 m 02 s.

```

1 create table users(
2     userid integer not null distkey sortkey,
3     username char(8),
4     city varchar(30),
5     state char(2),
6     likesports boolean,
7     liketheatre boolean,
8     likeconcerts boolean,
9     likejazz boolean,
10    likeclassical boolean,
11    likeopera boolean,
12    likerock boolean,
13    likevegas boolean,
14);
  
```

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



The screenshot shows the Amazon Redshift Query Editor interface, similar to Fig 8.3. The 'Query editor' tab is selected in the sidebar. The main area shows four tabs: 'Query 1', 'Query 2', 'Query 3', and 'Query 4'. 'Query 1' contains the following COPY command:

```

1 copy users from 's3://aws-tc-largeobjects/CUR-TF-200-ACDBSCI-1/Lab4/allusers_tab.txt'
2 credentials 'aws_iam_role=arn:aws:iam::632971437405:role/MyRedshiftRole'
3 delimiter '\t' region 'us-west-2';
  
```

The 'Status' bar at the top right shows 'Connected' to 'dev', 'user awuser', and 'awsuser'. The 'Query results' section for 'Query 1' shows it was completed on May 08, 2025, at 16:38:47, with an elapsed time of 00 m 07 s.

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

The screenshot shows the Amazon Redshift Query Editor v2 interface. On the left, the navigation sidebar includes options like Redshift Serverless, Provisioned clusters dashboard, Clusters, Query editor (selected), Datasources, Integrations, IAM identity Center, and Configurations. The main area has tabs for Resources info, Status (Connected), database (dev), user (awsuser), and Change connection. There are five tabs for queries labeled Query 1 through Query 5. The Query 1 tab contains the following SQL code:

```

1 SELECT sum(qtysold)
2 FROM sales, date
3 WHERE sales.dateid = date.dateid
4 AND caldate = '2008-01-05';

```

Below the code are buttons for Run, Save, Schedule, and Clear. The Run button is highlighted in orange. To the right, there's a 'Send feedback' link. The 'Query results' tab is selected, showing a summary: Query 2159 completed on May 08, 2025 at 16:46:50, with an elapsed time of 00 m 02 s. It shows 1 row returned, with a sum value. Buttons for Execution, Data, and Visualize are available. At the bottom, there are links for cloudShell and Feedback.

*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

The screenshot shows the 'Lab: Storing and Analyzing Data by Using Amazon Redshift' page. The top navigation bar includes links for Search results, DO180 Chapter 2,3,4,5,6, Data Eng Module 8 - Go!, SecurityGroup | EC2 | us..., MyRedshiftRole | IAM | G..., and Amazon Redshift | us-east-1. The main content area displays a lab summary and a scoring table.

**Module 8 Storing an... > Lab: Storing and Analyzing Data by Using Amazon Redshift**

retrieve data from a dataset that is stored in Amazon S3, extract the data to a Redshift database, and then query the data for analysis.

After completing this lab, you should be able to do the following:

- Review an AWS Identity and Access Management (IAM) role's permissions to access and configure Amazon Redshift.
- Create and configure a Redshift cluster.
- Create a security group for a Redshift cluster.
- Create a database, schemas, and tables with the Redshift cluster.
- Load data into tables in a Redshift cluster.
- Query data within a Redshift cluster by using the Amazon Redshift console.
- Query data within a Redshift cluster by using the API and AWS Command Line Interface (AWS CLI) in AWS Cloud9.
- Review an IAM policy with permissions to run queries on a Redshift cluster.
- Confirm that a data science team member can run queries on a Redshift cluster.

**Duration**

This lab requires **90 minutes** to complete. The lab will remain active for **180 minutes** so that you can complete all steps.

**AWS service restrictions**

| Total score                             | 30/30 |
|---|-------|
| TASK 2 - Cluster created                | 5/5   |
| TASK 2b - Security group associated     | 5/5   |
| TASK 3 - Tables created and data loaded | 5/5   |
| TASK 4 - Table data queried             | 5/5   |
| TASK 5 - Queried by using the AWS CLI   | 5/5   |
| TASK 7 - IAM user queried data          | 5/5   |

**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

The screenshot shows a browser window with multiple tabs open. The active tab is titled "Module 8 Knowledge Check" from the AWS Academy website. The page displays the results of a knowledge check. The score is listed as 90% (90 points) out of 100, with a required score of 70%. A message below the score says "Result: Congratulations! You have completed this module." There are "Previous" and "Next" buttons at the bottom. On the left, there's a sidebar with icons for Account, Dashboard, Courses, Calendar, Inbox, History, and Help. On the right, there's a sidebar with submission details: Grade: 90 (100 pts possible), Graded Anonymously: no, and Comments: No Comments.

*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

The screenshot shows the AWS Glue Studio interface. On the left, there is a sidebar with various navigation options like ETL jobs, Data Catalog, and Data Integration and ETL. The main area is titled "hudi-connection" under "Connection details". It shows the connector type as "MARKETPLACE" and the class name as "org.apache.hudi". There are sections for "Tags" and "Your jobs". A message at the bottom states: "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."

*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

The screenshot shows the AWS Glue Studio interface. The main area displays a message: "Successfully started job Hudi\_Streaming\_Job-98c369e0. Navigate to Run details for more details." Below this, there is a "Create job" section with three options: "Visual ETL" (selected), "Notebook", and "Script editor". Under "Example jobs", there is a table for "Your jobs" with one entry: "Hudi\_Streaming\_Job-98c369e0" (Glue Streaming, Script, 07/05/2025, 21:56:08, AWS Glue version 3.0). A message at the bottom states: "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."

*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

The screenshot shows the AWS Glue console with the 'Runs' tab selected for the 'Hudi\_Streaming\_Job-98c369e0'. The job is currently running. The 'Run details' section provides information such as the job name, start time (05/07/2025 21:57:57), end time (05/07/2025 21:58:01), and worker type (G.1X). It also shows the number of retries (0), execution time (44 seconds), and Cloudwatch logs.

*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

The screenshot shows the Amazon Kinesis Data Generator interface. A modal window titled 'Sending Data to Kinesis' displays the message '172 records sent to Kinesis.' Below the modal, the 'Schema 1' template is shown:

```
{
  "name": "{{random.arrayElement(["Sensor1","Sensor2","Sensor3", "Sensor4"])}}",
  "date": "{{date.utc('YYYY-MM-DD')}}",
  "year": "{{date.utc('YYYY')}}",
  "month": "{{date.utc('MM')}}",
  "day": "{{date.utc('DD')}}",
  "column_to_update_Integer": {{random.number(1000000000)}},
  "column_to_update_String": "{{random.arrayElement(['45f","47f","44f", "48f'])}}"
}
```

*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

9. Run the AWS Glue job.

- In the search box to the right of Services, search for and choose **AWS Glue** to open the AWS Glue console.
- In the navigation pane, under **ETL**, choose **Jobs**.
- In the Your jobs section, choose the link for the job name that contains **Hudi\_Streaming\_Job**.
- Note:** The CloudFormation template that you used in the previous step created this job. You can review the Python code on the **Script** tab.
- To start the job, choose **Run** in the upper-right corner.
- To view the status of the job, choose the **Runs** tab.

Before you continue, make sure that the **Run status** is **Running**. You might need to refresh the page.

In this task, you configured and started the AWS Glue job.

### Task 5: Using the KDG to send data to Kinesis

In this task, you will use the Kinesis Data Generator (KDG) tool to generate and send random data to Kinesis. The tool will simulate IoT devices sending data from sensors.

10. Access the KDG and start sending data.

- From the outputs that you recorded from CloudFormation, find the **KinesisDataGeneratorUrl** value. Paste that URL in a new browser tab or window to open the KDG tool.
- The URL is similar to <https://aws-labs.s3.us-west-2.amazonaws.com/amazon-kinesis-data-generator/web/producer.html?void>

*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

Your cluster 'Hive EMR Cluster' has been successfully created.

### Hive EMR Cluster

| Cluster info   |                                  | Applications                                       | Cluster management  | Status and time                               |
|--|----------------------------------|--|---|---|
| Cluster ID<br>j-1DONNZ756DKGR  | Amazon EMR version<br>emr-5.29.0 | Installed applications<br>Hadoop 2.8.5, Hive 2.3.6 | Log destination in Amazon S3<br>s3://hive-output-563d29zd | Status<br>Starting                            |
| Cluster ARN<br><a href="#">arn:aws:elasticmapreduce:us-east-1:986657421547:cluster/j-1DONNZ756DKGR</a> |                                  |  | Primary node public DNS                                   | Creation time<br>7 May 2023 18:24 (UTC-04:00) |
| Cluster configuration<br>Instance groups   |                                  |  |   | Elapsed time<br>0 seconds                     |
| Capacity<br>1 Primary   2 Core   1 Task  |                                  |  |   |   |

**Properties** Bootstrap actions Instances (hardware) Steps Applications Configurations Monitoring Events Tags (0)

**Cluster logs** Info Archive log files to Amazon S3 Turned on Amazon S3 location <s3://hive-output-563d29zd/>

**Turn on encryption for logs** Turned off

**Cluster termination and node replacement** Info Edit Termination option Manually terminate cluster Idle time - Termination protection Off Unhealthy node replacement On

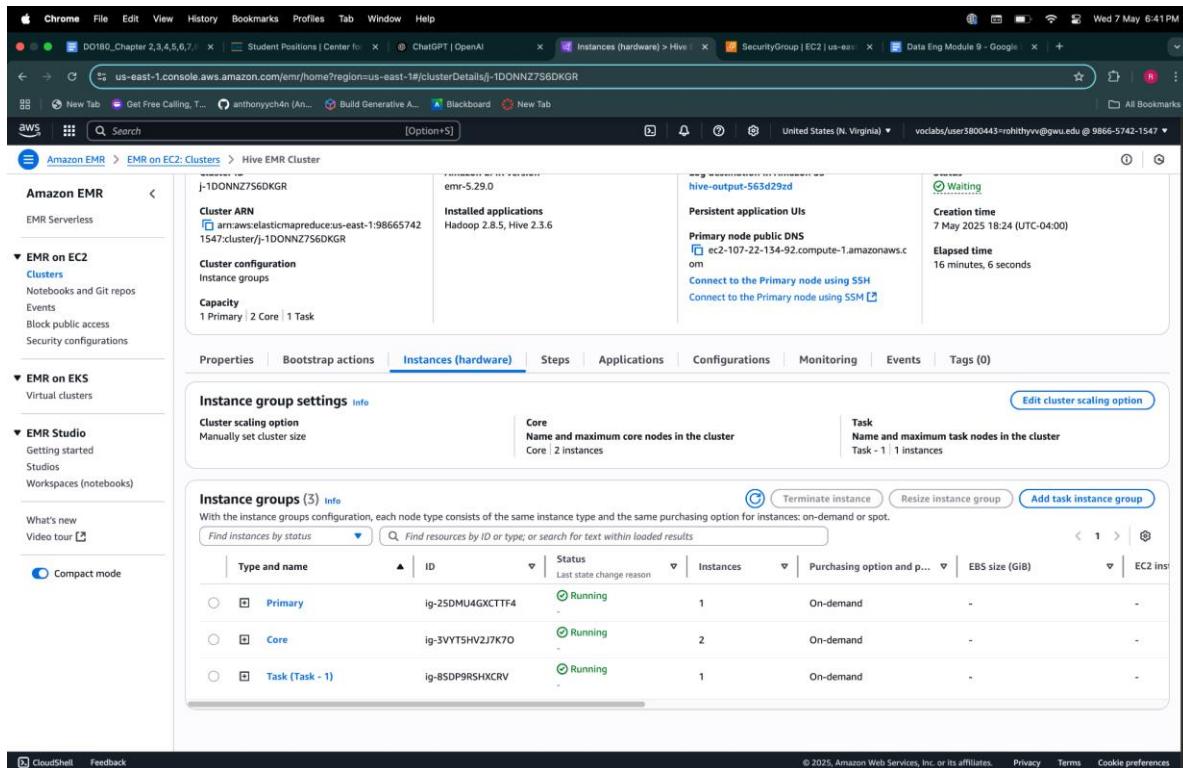
**Network and security** Info Network Virtual Private Cloud (VPC) vpc-08a6ff03f5ae9e8ad Subnet ID and Availability Zone (AZ)

**Security configuration** Security configuration: none EC2 key pair: None

**Permissions** Service role for Amazon EMR: EMR\_DefaultRole EC2 instance profile: None

*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*

The screenshot shows a terminal window with several tabs open, including "Properties > Hive", "SecurityGroup", "EC2 us-east-1", "Applications > H", "Cloud9 Instance", "SecurityGroup", and "Data Eng Module". The main terminal window displays the following command and its output:

```
ssh -T hadoop@ip-10-16-10-158.compute-1.amazonaws.com
The authenticity of host 'ec2-10-16-10-158.compute-1.amazonaws.com (10.16.10.158)' can't be established.
ED25519 key fingerprint is SHA256:05NBIA01WYR35ILopz6SpUuQoJ0pmn6gRTWw.
EDDSA key fingerprint is MD5:22:32:07:03:de:86:a5:83:66:2b:8f:f7:77:09:5d:8b:ff.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-10-16-10-158.compute-1.amazonaws.com,10.16.10.158' (EDDSA) to the list of known hosts.
Last login: Wed May 7 23:23:52 2015
[Amazon Linux AMI]
https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
55 package(s) needed for security, out of 82 available
Run "sudo yum update" to apply all updates.

[REDACTED]
[hadoop@ip-10-16-10-158 ~]$ ls
bash -> ip-10-16-10-34.ec2.x immediate
voclabs:~/environment $
```

*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*

The screenshot shows a Chrome browser window with multiple tabs open. The active tab is a terminal session on an AWS Cloud9 Instance, displaying Hive SQL commands being executed. The commands include creating a temporary table, inserting data from a Clicks table, and performing a query. The terminal output shows the execution time and job status.

```
Time taken: 40.968 seconds
hive> CREATE TABLE tmp_clicks (
    >     impressionId string,
    >     addId string
    > )
    > STORED AS SEQUENCEFILE;
OK
Time taken: 0.068 seconds
hive> INSERT OVERWRITE TABLE tmp_clicks
    >     SELECT
    >         impressionId,
    >         addId
    >     FROM Clicks C
    >     WHERE c.dt >= '${DAY}-00'
    >     AND c.dt < '${NEXT_DAY}-00'
Query ID = hadoop_20250508002342_320b4742-023a-483a-a8c3-21e92c9851ba
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1746656950902_0003)

VERTICES      MODE      STATUS      TOTAL      COMPLETED      RUNNING      PENDING      FAILED      KILLED
Map 1      container      RUNNING      1      0      1      0      0      0
VERTICES: 00/01 [>>>-----] 0%   ELAPSED TIME: 12.36 s
```

The bottom of the terminal window shows a bash prompt: `vocabs:~/environment $`.

A sidebar on the right is the Google Chrome sync interface, showing profile information and sync status.

*Note: Temporary Hive table tmp\_clicks created, populated with filtered click data, and query execution initiated.*

Fig 9.6: End of guided lab

This method queries the AWS Glue Data Catalog for each to-be-ingested record and gets the current Hudi table schema. It then merges the Hudi table schema with the schema of the to-be-ingested record and enriches that schema with null values for `new_column`. This enables Athena to query the Hudi dataset without any issues.

In this task, you reverted the schema and observed that records were updated in place.

### Update from the team

Congratulations! In this lab, you created the Hudi connection and then used a custom Python script to read data from Kinesis Data Streams. You used Athena to run queries on the data and see the changes in near real time. You also changed the schema and observed that Athena queries ran seamlessly and returned the expected data.

Your POC was successful to demonstrate how to process dynamic data changes and accommodate changes to the data structures.

### Submitting your work

16. To record your progress, choose **Submit** at the top of these instructions.

17. When prompted, choose **Yes**.

After a couple of minutes, the grades panel appears and shows you how many points you earned for each task. If the results don't display after a couple of minutes, choose **Grades** at the top of these instructions.

**Tip:** You can submit your work multiple times. After you change your work, choose **Submit** again.

Your last submission is recorded for this lab.

| Total score                              | 30/30 |
|--|-------|
| [TASK 2] Hudi Connection created         | 5/5   |
| [TASK 3] Glue Job Files configured       | 5/5   |
| [TASK 4] Glue Job Configured and Running | 5/5   |
| [TASK 5] Kinesis Data Generation Started | 5/5   |
| [TASK 6] Athena Configuration Verified   | 5/5   |
| [TASK 7] Schema Changes verified         | 5/5   |

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

Module 9 Knowledge Check

Due No Due Date Points 100 Submitting an external tool

**Knowledge check results**

Your score: 100% (100 points)  
Required score: 70% (70 points)

**Result:** Congratulations! You have completed this module.

To continue, choose **Next** in the lower-right corner.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

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*

The screenshot shows a Chrome browser window with multiple tabs open. The active tab is titled "Module 10 Knowledge Check". The page content displays the results of the knowledge check:

- Knowledge check results**
- Your score:** 100% (100 points)
- Required score:** 70% (70 points)
- Result:** Congratulations! You have completed this module.
- To continue, choose **Next** in the lower-right corner.

On the right side of the page, there is a "Submission" section showing the submission date (May 3 at 1:10am), grade (100), and that it was graded anonymously. There are also "Comments" and "Grade Details" sections.

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

The screenshot shows the AWS IAM Roles page. The role 'OsDemoWebserverIAMRole' is selected. The 'Permissions' tab is active, showing three inline policies: 'OsDemoWebserverIAMPolicy1', 'OsDemoWebserverIAMPolicy2', and 'OsDemoWebserverIAMPolicy3'. Each policy is listed under 'Customer inline' and has an attached entity count of 0. There are also sections for 'Permissions boundary' (not set) and 'Generate policy based on CloudTrail events'.

*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

The screenshot shows the AWS Lambda Functions page. The function 'os-demo-lambda-function' is selected. The 'Function overview' tab is active, displaying basic information like the function name, ARN, and last modified time (55 minutes ago). The 'Code source' tab is open, showing the code editor with 'Lambda.py' containing the following Python code:

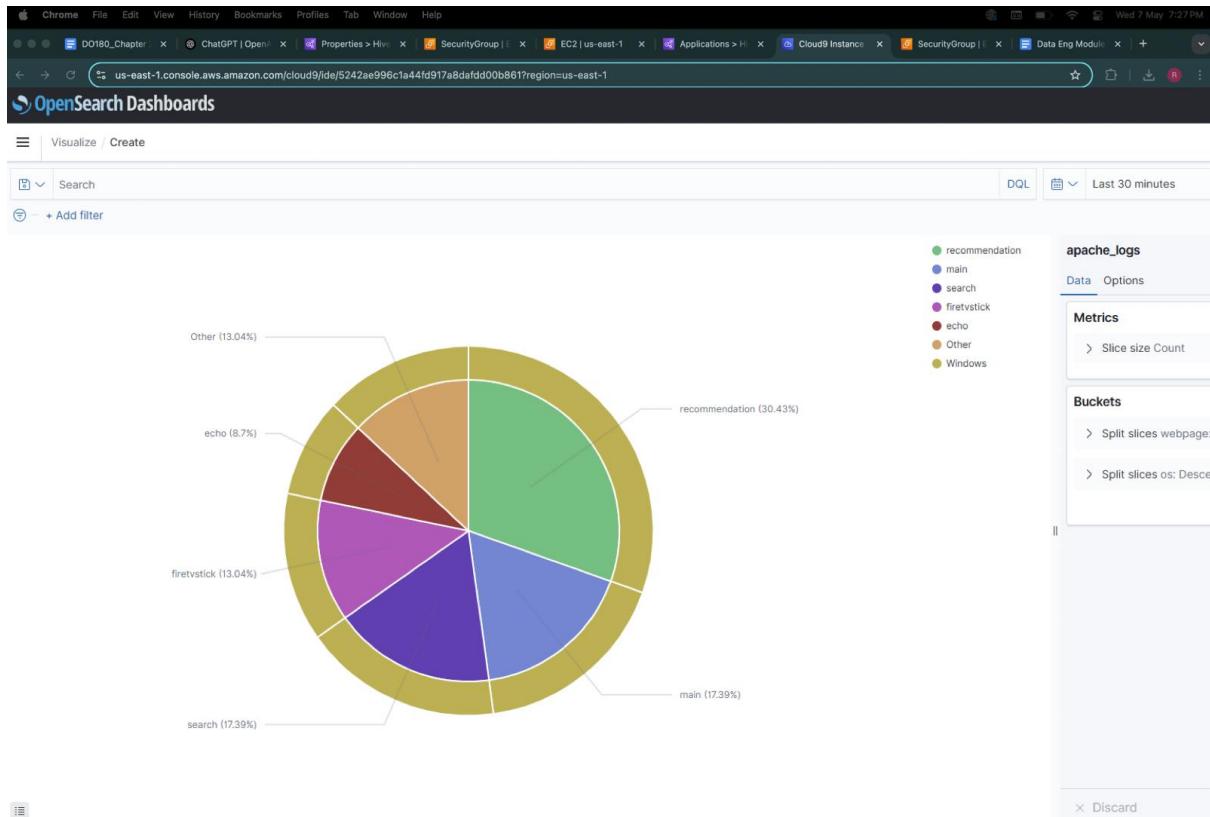
```

1 #!/usr/bin/python
2 # -*- coding: utf-8 -*-

```

*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

The screenshot shows a web browser window with the URL <https://us-east-1.console.aws.amazon.com/cloud9/ide/5242ae996c1a44fd917a8dafdd00b861?region=us-east-1>. The browser has several tabs open, including 'Properties > HW...', 'SecurityGroup | us-east-1', 'EC2 | us-east-1', 'Cloud9 Instance', 'SecurityGroup | us-east-1', and 'Data Eng Module'. The main content area displays a heatmap visualization titled 'Analyzing and Visualizing Streaming Data' with the subtitle 'Using Kinesis Data Firehose, OpenSearch Service, and OpenSearch Dashboards'. Below the visualization, there is an 'Analysis' section stating: 'Analysis: Based on this image of the visualization, visitors accessed the Echo and FireStick product pages more often from the search page than the recommendations page. The team could infer that the search page is more effective than the recommendations page at directing users to the product pages.' A congratulatory message follows: 'Congratulations! In this task, you created a heat map to gain insights into whether more customers were referred to product pages from the search page or the recommendations page. Your POC to demonstrate how to use Kinesis Data Firehose and OpenSearch Service to analyze streaming data from a website was successful.' On the right side, there is a summary table:

| Total score                             | 20/20 |
|---|-------|
| [TASK 1] EC2 Instance type was reviewed | 5/5   |
| [TASK 1b] IAM Policy was reviewed       | 5/5   |
| [TASK 2] OpenSearch user authenticated  | 5/5   |
| [TASK 4] CloudWatch logs reviewed       | 5/5   |

At the bottom left, there is an 'Update from the team' section with the following text: 'You share the POC results with the administrator for the university bookstore's website. She is excited to implement the infrastructure to analyze the streaming data from the web server access logs.' At the bottom right, there is a 'Next ▶' button.

*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

The screenshot shows a Chrome browser window with multiple tabs open. The active tab is titled "Module 11 Knowledge Check". The page content displays the results of the knowledge check:

- Your score:** 100% (100 points)
- Required score:** 70% (70 points)
- Result:** Congratulations! You have completed this module.
- Text below result:** To continue, choose **Next** in the lower-right corner.

On the right side of the page, there is a "Submission" section with the following details:

- Date: Apr 28 at 12:43am
- Grade: 100 (100 pts possible)
- Graded Anonymously: no
- Comments: No Comments

The browser interface includes a sidebar with icons for Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The address bar shows the URL: [awsacademy.instructure.com/courses/107814/assignments/1198128?module\\_item\\_id=10090149](https://awsacademy.instructure.com/courses/107814/assignments/1198128?module_item_id=10090149).

Note: This image shows the completion of module 11

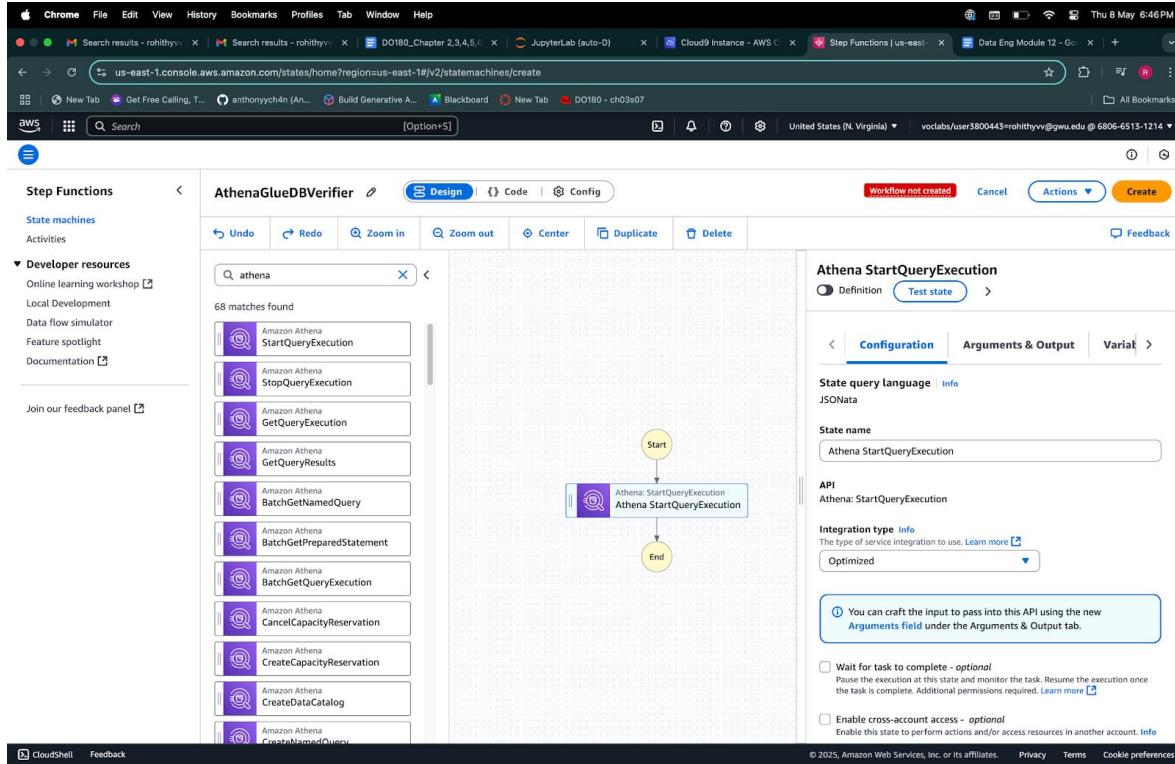
## Summary:

- **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

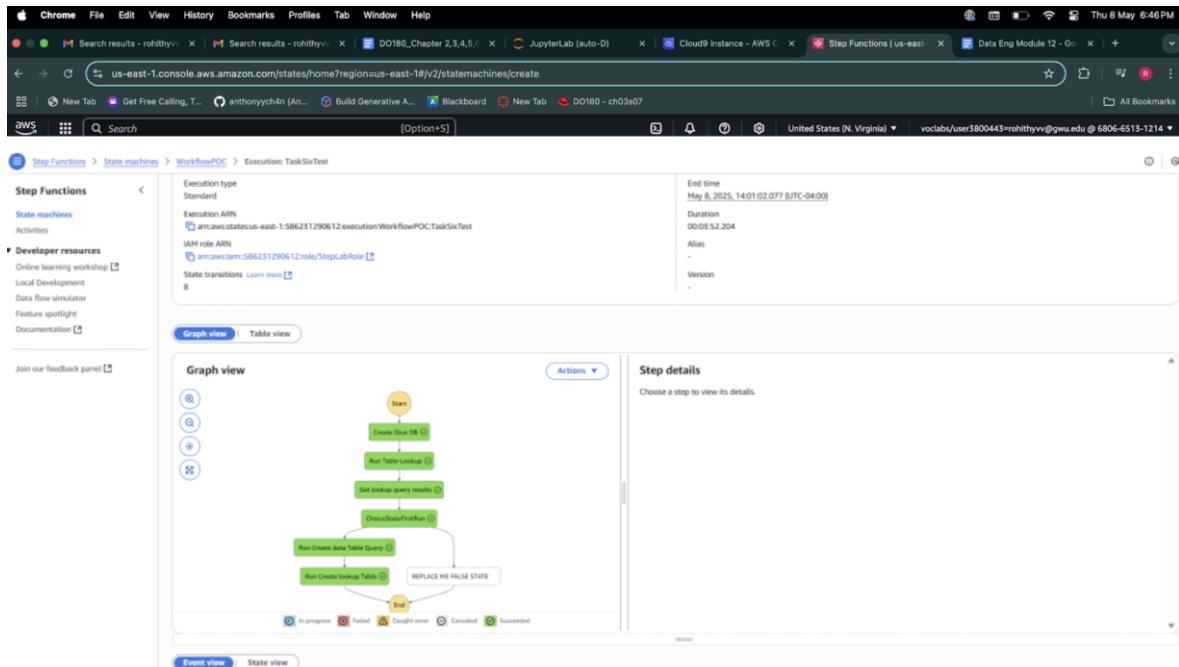
## Lab: Building and Orchestrating ETL Pipelines by Using Athena and Step Functions

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.

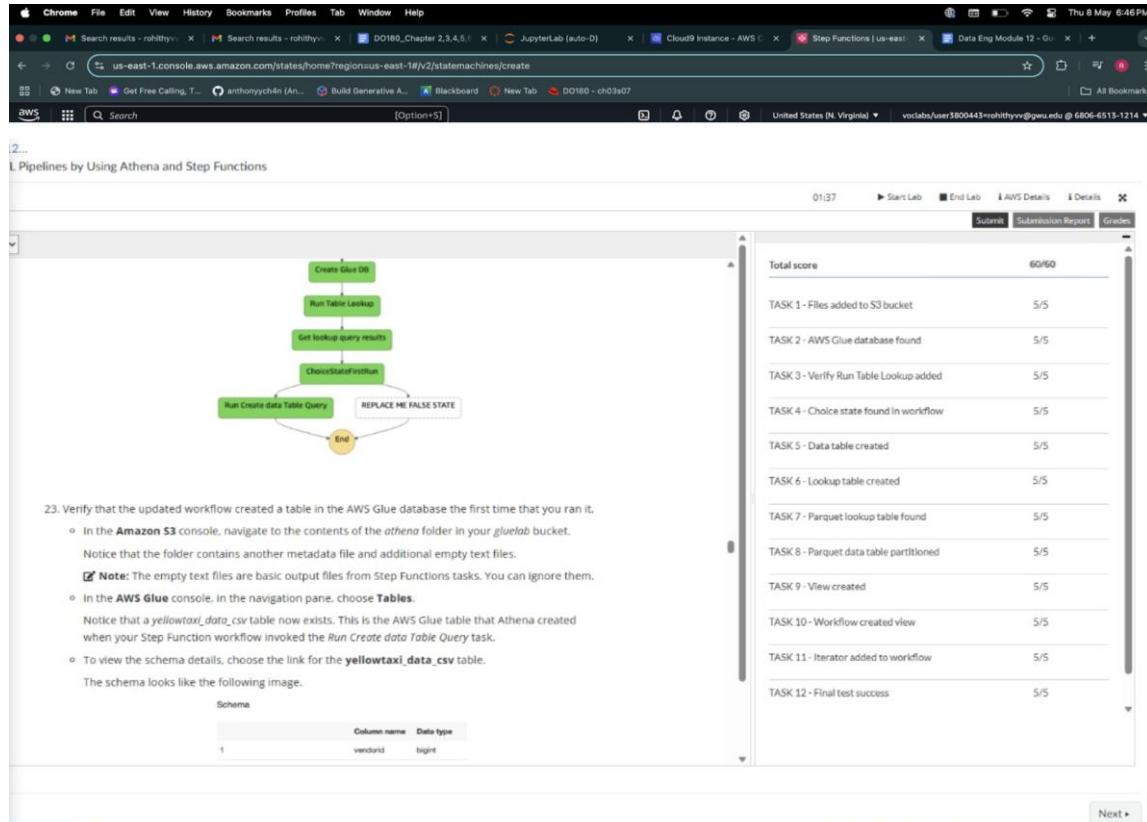
Fig 12.2: Creating the AWS Glue table for location lookup data



Note: Created another AWS Glue table by updating and running the Step Functions workflow. The new table will reference the `taxis_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

The screenshot shows a browser window with multiple tabs open. The active tab is titled "Module 12 Knowledge Check". The page content is as follows:

**Knowledge check results**

**Your score:** 100% (100 points)  
**Required score:** 70% (70 points)

---

**Result:** Congratulations! You have completed this module.

To continue, choose **Next** in the lower-right corner.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Navigation buttons at the bottom include "Previous" and "Next".

*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.