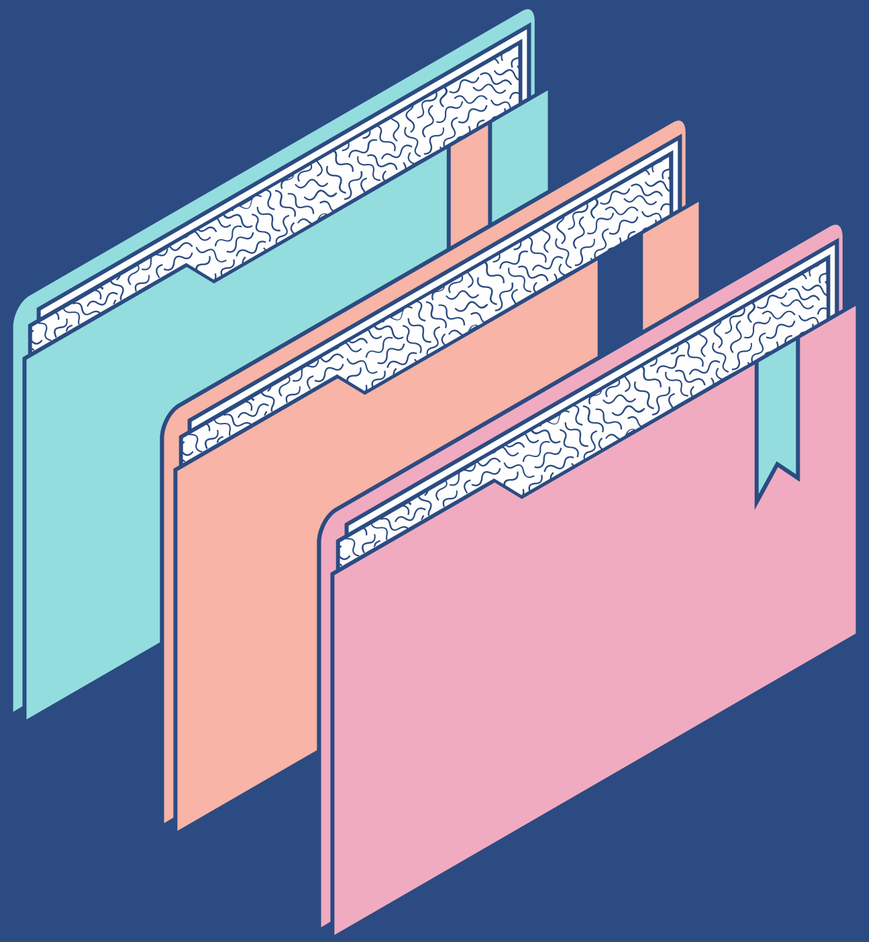
Airbnb Ratings & Pricing Insights using AWS Big Data Pipeline

Saurabh Ghumnar



Agenda

KEY TOPICS DISCUSSED IN
THIS PRESENTATION

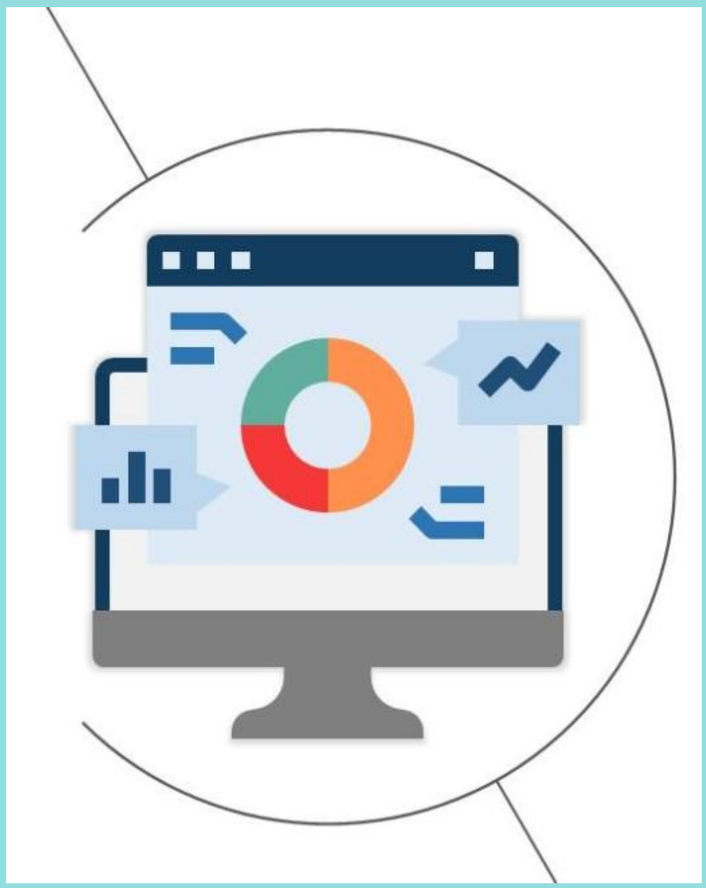
- Problem Statement
- Dataset Overview
- Dataset Schema
- AWS Architecture Diagram
- Lambda Automation
- Glue Crawler Setup
- Athena Queries and Insights
- Machine Learning with EMR
- Final Outputs
- Challenges and Resolutions

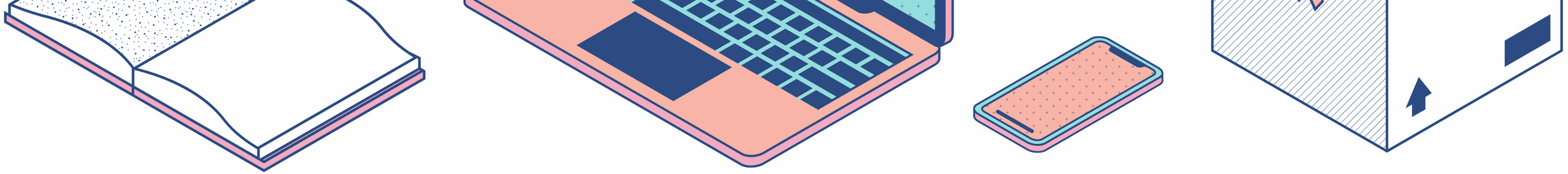
Problem Statement

Objective: **Discover actionable insights from Airbnb data using cloud-native analytics.**

This project focuses on answering key questions like:

- What factors influence customer ratings across cities?
- How do pricing trends correlate with customer satisfaction?
- Can we identify overpriced underperformers?
- How do we scale this analysis using modern cloud tools?



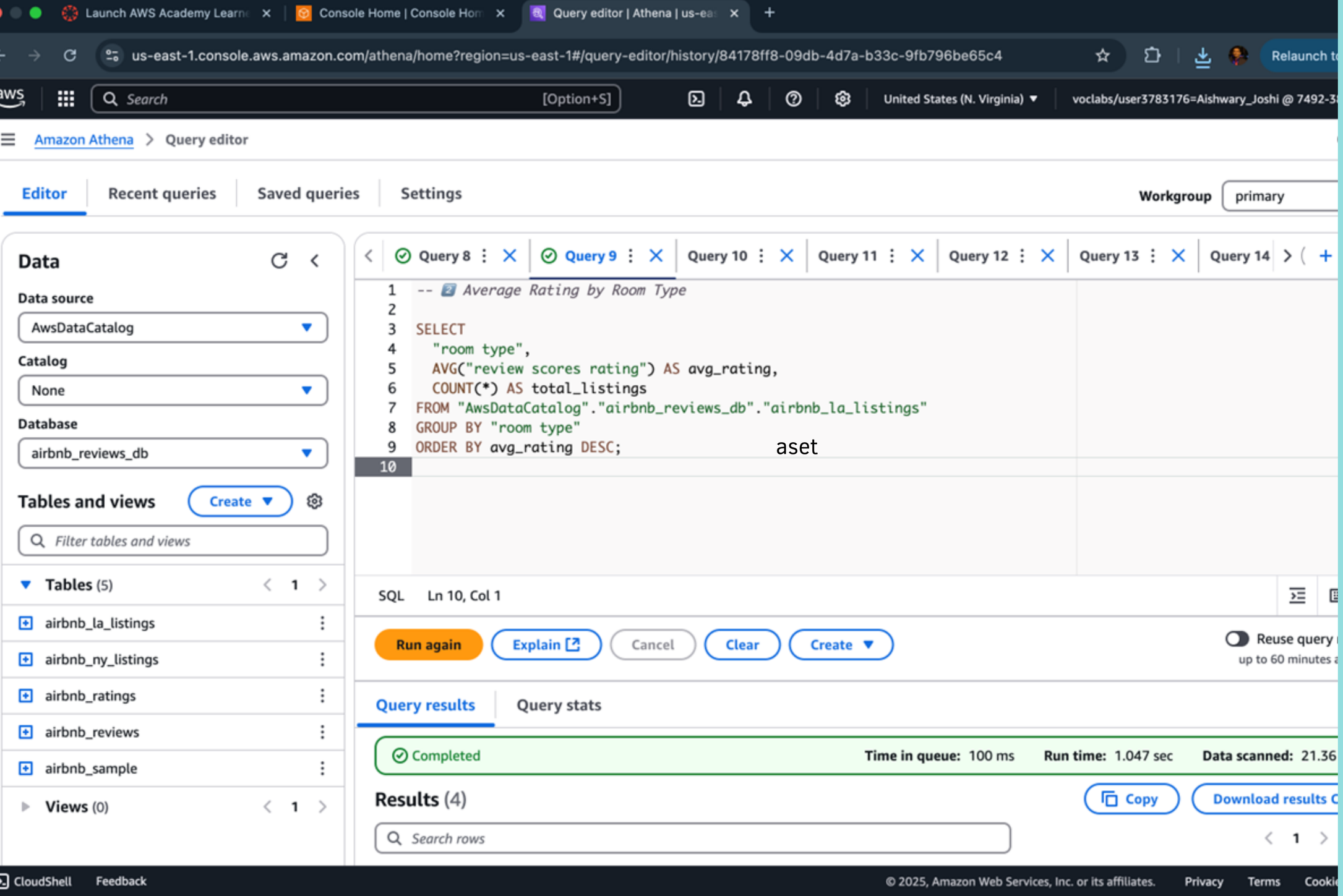


Dataset Overview

- We used publicly available Airbnb datasets covering listings and reviews from:
- Los Angeles
- New York City These were uploaded to:
- S3 Bucket: airbnb/raw/
- Subfolders: LA_Listings/, NY_Listings/, Reviews/, Ratings/

Dataset Schema

- Key attributes across the datasets include:
- listing id, host name, room type, price, review scores rating
- Reviews metadata with timestamps and sentiment-rich text This structured schema enabled meaningful groupings (by room type, neighborhood, etc.) and served as the basis for ML feature engineering.


Launch AWS Academy Learn
Console Home | Console Hon
Query editor | Athena | us-ea
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/84178ff8-09db-4d7a-b33c-9fb796be65c4
Search [Option+S]
United States (N. Virginia)
Amazon Athena > Query editor
Editor
Recent queries
Saved queries
Settings
Workgroup primary
Data
Data source
AwsDataCatalog
Catalog
None
Database
airbnb_reviews_db
Tables and views
Create
Filter tables and views
Tables (5)
airbnb_la_listings
airbnb_ny_listings
airbnb_ratings
airbnb_reviews
airbnb_sample
Views (0)
Query 8
Query 9
Query 10
Query 11
Query 12
Query 13
Query 14
-- Average Rating by Room Type
SELECT
"room type",
AVG("review scores rating") AS avg_rating,
COUNT(*) AS total_listings
FROM "AwsDataCatalog"."airbnb_reviews_db"."airbnb_la_listings"
GROUP BY "room type"
ORDER BY avg_rating DESC;
aset
SQL Ln 10, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query
up to 60 minutes
Query results
Query stats
Completed
Time in queue: 100 ms
Run time: 1.047 sec
Data scanned: 21.36
Results (4)
Copy
Download results
Search rows
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms

Dataset

AWS Architecture Diagram

Amazon S3
(Raw CSV Upload)

AWS Lambda
(Auto Trigger)

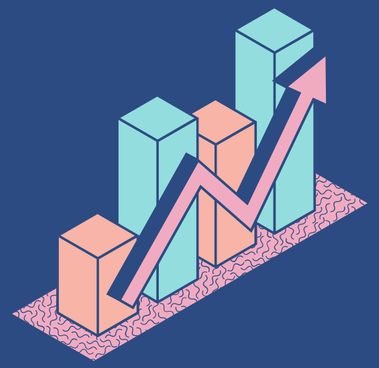
AWS Glue
(Database/Table Creation)

AWS Athena
(SQL-based Queries)

amazon EMR
(ML Modeling) - Optional

LAMBDA AUTOMATION

- Lambda triggers on file upload in S3
- Script handles:
  1. Glue DB: airbnb_reviews_db
  2. Dynamic crawler setup
  3. Table creation & monitoring



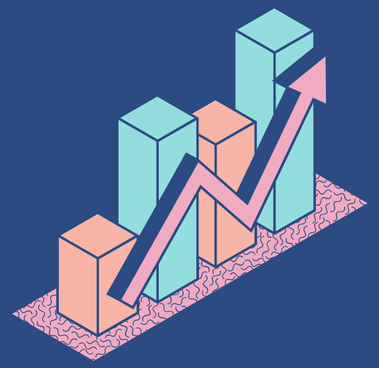
Lambda Code Snippet

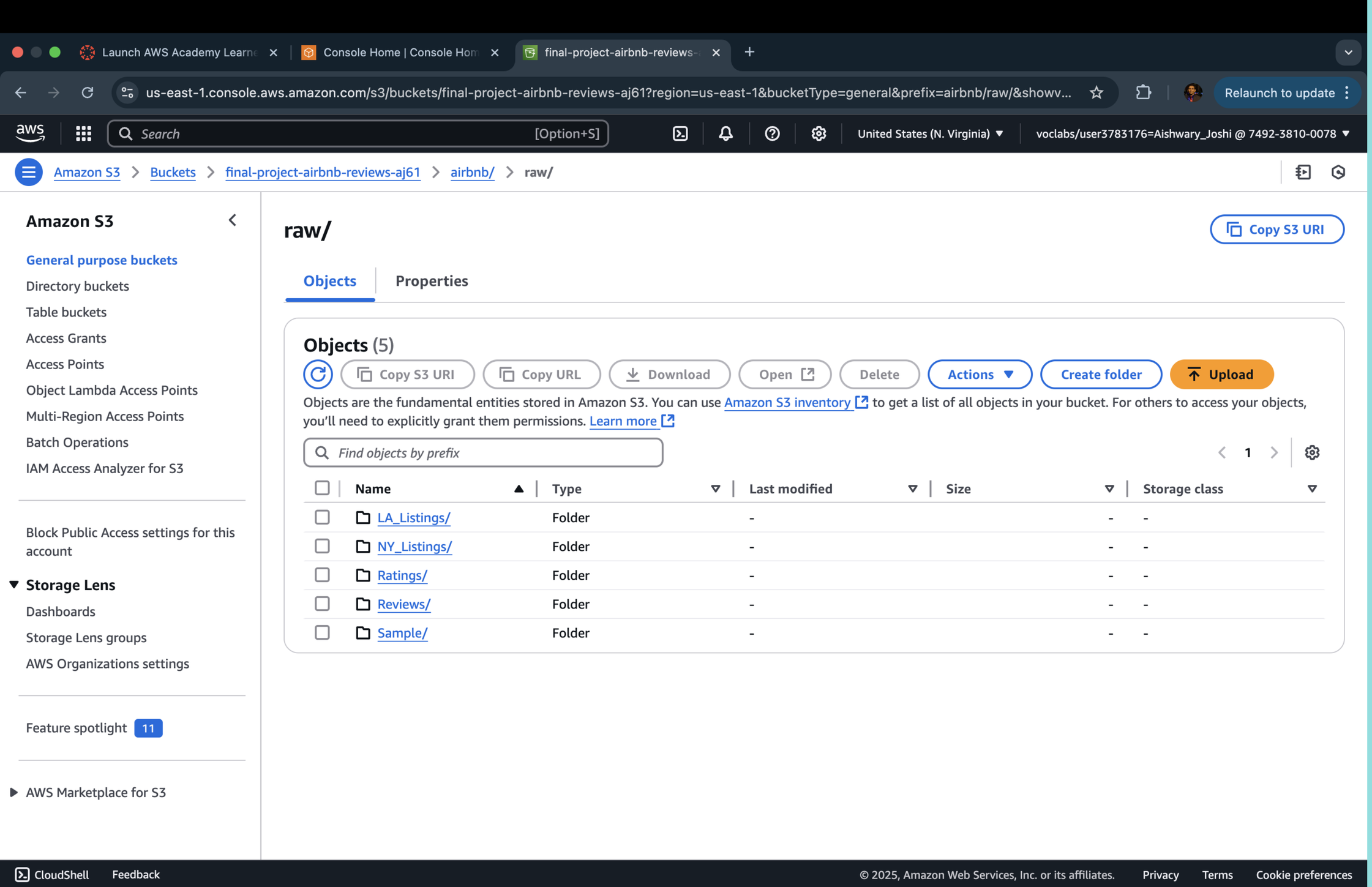
(lambda_code.py)

```python
import boto3
import urllib.parse
import time

s3 = boto3.client('s3')
glue = boto3.client('glue')

GLUE_DATABASE_NAME = "airbnb_reviews_db"
GLUE_ROLE_NAME = "LabRole"
CRAWLER_PREFIX = "crawler_for_"
TABLE_PREFIX = "airbnb_"

def lambda_handler(event, context):
    for record in event['Records']:
        bucket = record['s3']['bucket']['name']
        key = urllib.parse.unquote_plus(record['s3']['object']['key'])

        print(f"✅ File uploaded: s3://{bucket}/{key}")

        key_parts = key.split('/')
        if len(key_parts) < 4:
            print("⚠️ File not under expected path /airbnb/raw/<folder>/<file.csv>")
            return

        table_folder = key_parts[2]  # FIXED: correctly gets folder name
        table_name_suffix = table_folder.replace('-', '_').lower()
        final_table_name = TABLE_PREFIX + table_name_suffix
        crawler_name = CRAWLER_PREFIX + table_name_suffix
        s3_target_path = f"s3://{bucket}/airbnb/raw/{table_folder}/"

        try:
            glue.create_database(DatabaseInput={'Name': GLUE_DATABASE_NAME})
            print(f"✅ Created Glue database: {GLUE_DATABASE_NAME}")
        except glue.exceptions.AlreadyExistsException:
            print(f"ℹ️ Glue database {GLUE_DATABASE_NAME} already exists")

        try:
            glue.create_crawler(
                Name=crawler_name,
                Role=GLUE_ROLE_NAME,
                DatabaseName=GLUE_DATABASE_NAME,
                Targets={'S3Targets': [{'Path': s3_target_path}]},
                TablePrefix=TABLE_PREFIX,
                SchemaChangePolicy={
                    'UpdateBehavior': 'UPDATE_IN_DATABASE',
                    'DeleteBehavior': 'DEPRECATE_IN_DATABASE'
                }
            )
            print(f"✅ Created crawler: {crawler_name}")
        except glue.exceptions.AlreadyExistsException:
            print(f"ℹ️ Crawler {crawler_name} already exists")

        try:
            glue.start_crawler(Name=crawler_name)
            print(f"🚀 Started crawler: {crawler_name}")
        except glue.exceptions.CrawlerRunningException:
            print(f"⚠️ Crawler {crawler_name} is already running")

        # Wait for crawler to finish
        print("⏳ Waiting for crawler to finish...")
        while True:
            status = glue.get_crawler(Name=crawler_name)['Crawler']['State']
            if status == 'READY':
                break
            time.sleep(5)

        print(f"✅ Crawler finished. Looking for table: {final_table_name} in DB: {GLUE_DATABASE_NAME}")

        try:
            try:
                glue.start_crawler(Name=crawler_name)
                print(f"🚀 Started crawler: {crawler_name}")
            except glue.exceptions.CrawlerRunningException:
                print(f"⚠️ Crawler {crawler_name} is already running")

            # Wait for crawler to finish
            print("⏳ Waiting for crawler to finish...")
            while True:
                status = glue.get_crawler(Name=crawler_name)['Crawler']['State']
                if status == 'READY':
                    break
                time.sleep(5)

            print(f"✅ Crawler finished. Looking for table: {final_table_name} in DB: {GLUE_DATABASE_NAME}")

            try:
                response = glue.get_table(
                    DatabaseName=GLUE_DATABASE_NAME,
                    Name=final_table_name
                )
                print(f"✅ Table found: {final_table_name}")
            except glue.exceptions.EntityNotFoundException:
                print(f"❌ Table {final_table_name} NOT found in database {GLUE_DATABASE_NAME}.")
                print(f"🔍 Check if the file at {s3_target_path} is correctly formatted and non-empty.")

    return {"status": "done"}
```

Glue Crawler Setup

- 5 Glue tables auto-created
- Crawlers run on individual S3 folders
- Tables: LA, NY Listings, Ratings, Reviews, Sample

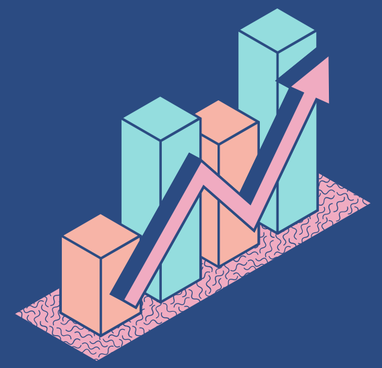


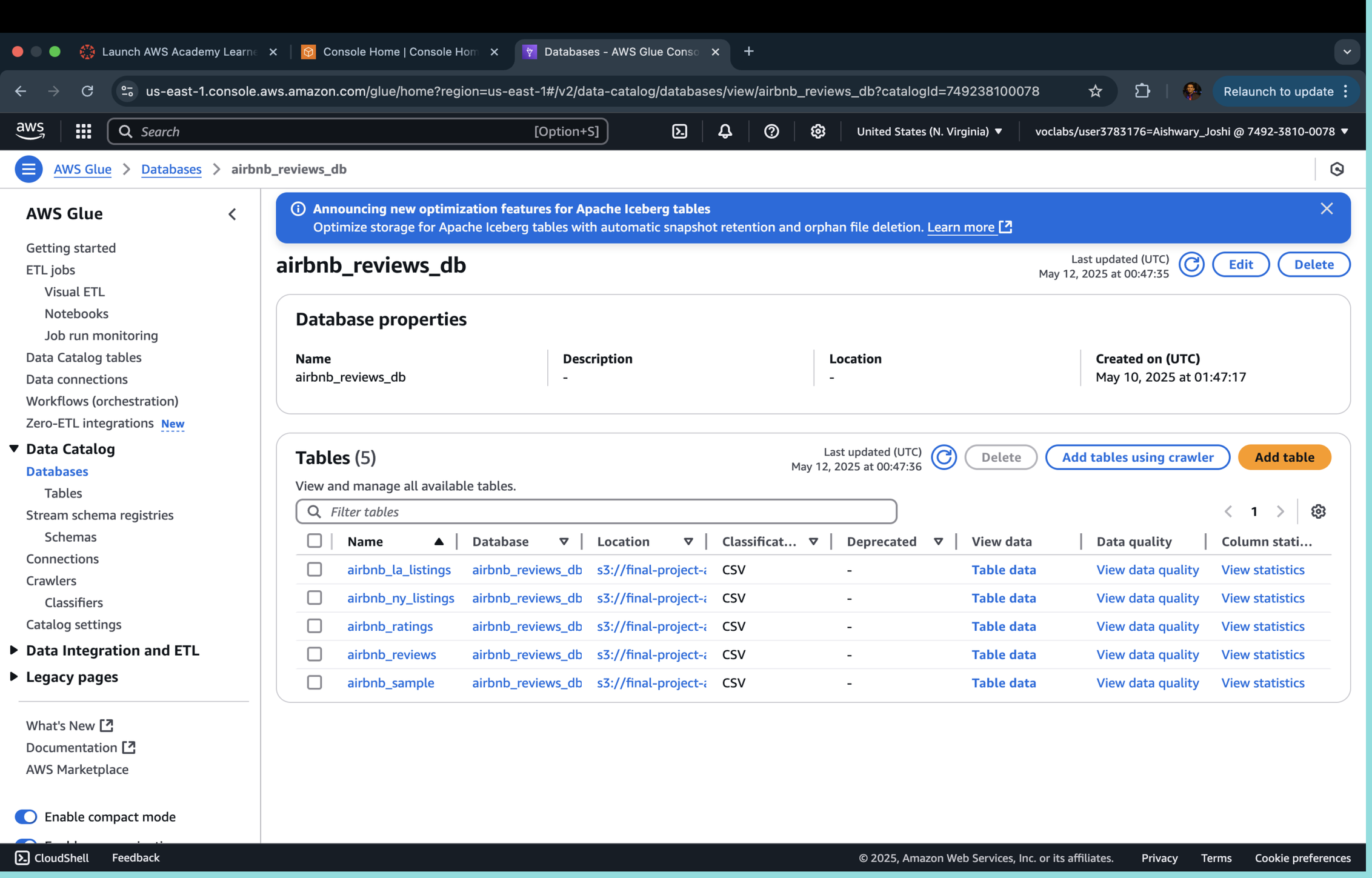

Launch AWS Academy Learne
Console Home | Console Hom
final-project-airbnb-reviews-
us-east-1.console.aws.amazon.com/s3/buckets/final-project-airbnb-reviews-aj61?region=us-east-1&bucketType=general&prefix=airbnb/raw/&showv...
Relaunch to update
aws
Search
[Option+S]
United States (N. Virginia)
voclabs/user3783176=Aishwary_Joshi @ 7492-3810-0078
Amazon S3 > Buckets > final-project-airbnb-reviews-aj61 > airbnb/ > raw/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 11
AWS Marketplace for S3
raw/
Copy S3 URI
Objects
Properties
Objects (5)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
LA_Listings/ Folder - - -
NY_Listings/ Folder - - -
Ratings/ Folder - - -
Reviews/ Folder - - -
Sample/ Folder - - -
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

OBJECTS

Glue Databases And Tables

- Database: airbnb_reviews_db
- Location mapped to: s3://final-project-airbnb-reviews...
- Format: CSV with structured schema

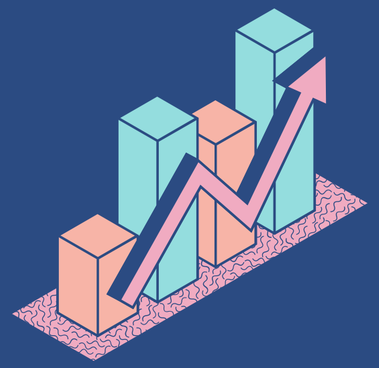


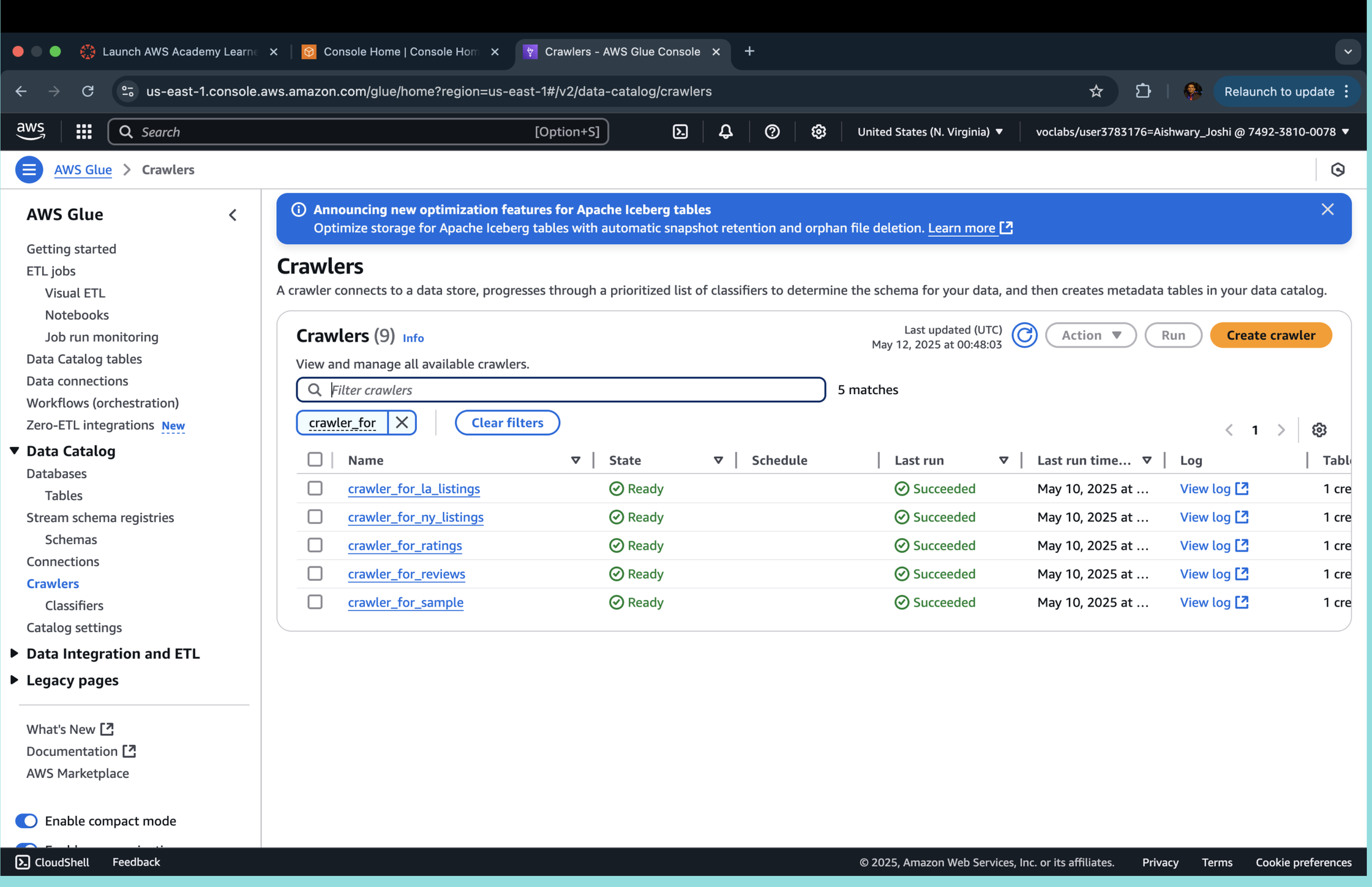

Launch AWS Academy Learne
Console Home | Console Hom
Databases - AWS Glue Conso
us-east-1.console.aws.amazon.com/glue/home?region=us-east-1#/v2/data-catalog/databases/view/airbnb_reviews_db?catalogId=749238100078
Relaunch to update
Search [Option+S]
United States (N. Virginia)
voclabs/user3783176=Aishwary_Joshi @ 7492-3810-0078
AWS Glue > Databases > airbnb_reviews_db
AWS Glue
Getting started
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Data Catalog tables
Data connections
Workflows (orchestration)
Zero-ETL integrations New
Data Catalog
Databases
Tables
Stream schema registries
Schemas
Connections
Crawlers
Classifiers
Catalog settings
Data Integration and ETL
Legacy pages
What's New
Documentation
AWS Marketplace
Enable compact mode
Announcing new optimization features for Apache Iceberg tables
Optimize storage for Apache Iceberg tables with automatic snapshot retention and orphan file deletion. Learn more
airbnb_reviews_db
Last updated (UTC)
May 12, 2025 at 00:47:35
Edit
Delete
Database properties
Name
airbnb_reviews_db
Description
-
Location
-
Created on (UTC)
May 10, 2025 at 01:47:17
Tables (5)
Last updated (UTC)
May 12, 2025 at 00:47:36
Delete
Add tables using crawler
Add table
View and manage all available tables.
Filter tables
1
Name
Database
Location
Classificat...
Deprecated
View data
Data quality
Column stati...
airbnb_la_listings airbnb_reviews_db s3://final-project-a CSV - Table data View data quality View statistics
airbnb_ny_listings airbnb_reviews_db s3://final-project-a CSV - Table data View data quality View statistics
airbnb_ratings airbnb_reviews_db s3://final-project-a CSV - Table data View data quality View statistics
airbnb_reviews airbnb_reviews_db s3://final-project-a CSV - Table data View data quality View statistics
airbnb_sample airbnb_reviews_db s3://final-project-a CSV - Table data View data quality View statistics
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

TABLES

Querying with Amazon Athena

- SQL queries run directly on S3 through Glue tables
- Queries written to answer specific business questions

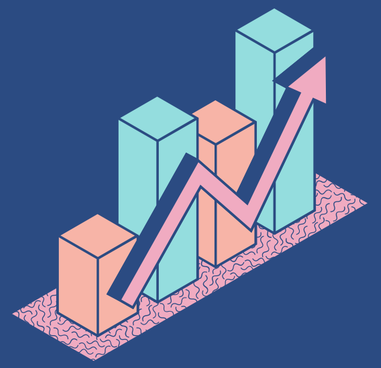



Launch AWS Academy Learne
Console Home | Console Hom
Crawlers - AWS Glue Console
us-east-1.console.aws.amazon.com/glue/home?region=us-east-1#/v2/data-catalog/crawlers
Relaunch to update
Search [Option+S]
United States (N. Virginia)
voclabs/user3783176=Aishwary_Joshi @ 7492-3810-0078
AWS Glue > Crawlers
AWS Glue
Getting started
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Data Catalog tables
Data connections
Workflows (orchestration)
Zero-ETL integrations New
Data Catalog
Databases
Tables
Stream schema registries
Schemas
Connections
Crawlers
Classifiers
Catalog settings
Data Integration and ETL
Legacy pages
What's New
Documentation
AWS Marketplace
Enable compact mode
Announcing new optimization features for Apache Iceberg tables
Optimize storage for Apache Iceberg tables with automatic snapshot retention and orphan file deletion. Learn more
Crawlers
A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in your data catalog.
Crawlers (9) Info
Last updated (UTC)
May 12, 2025 at 00:48:03
Action
Run
Create crawler
View and manage all available crawlers.
Filter crawlers
5 matches
crawler_for
Clear filters
1
Name
State
Schedule
Last run
Last run time...
Log
crawler_for_la_listings Ready Succeeded May 10, 2025 at ... View log 1 cre
crawler_for_ny_listings Ready Succeeded May 10, 2025 at ... View log 1 cre
crawler_for_ratings Ready Succeeded May 10, 2025 at ... View log 1 cre
crawler_for_reviews Ready Succeeded May 10, 2025 at ... View log 1 cre
crawler_for_sample Ready Succeeded May 10, 2025 at ... View log 1 cre
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

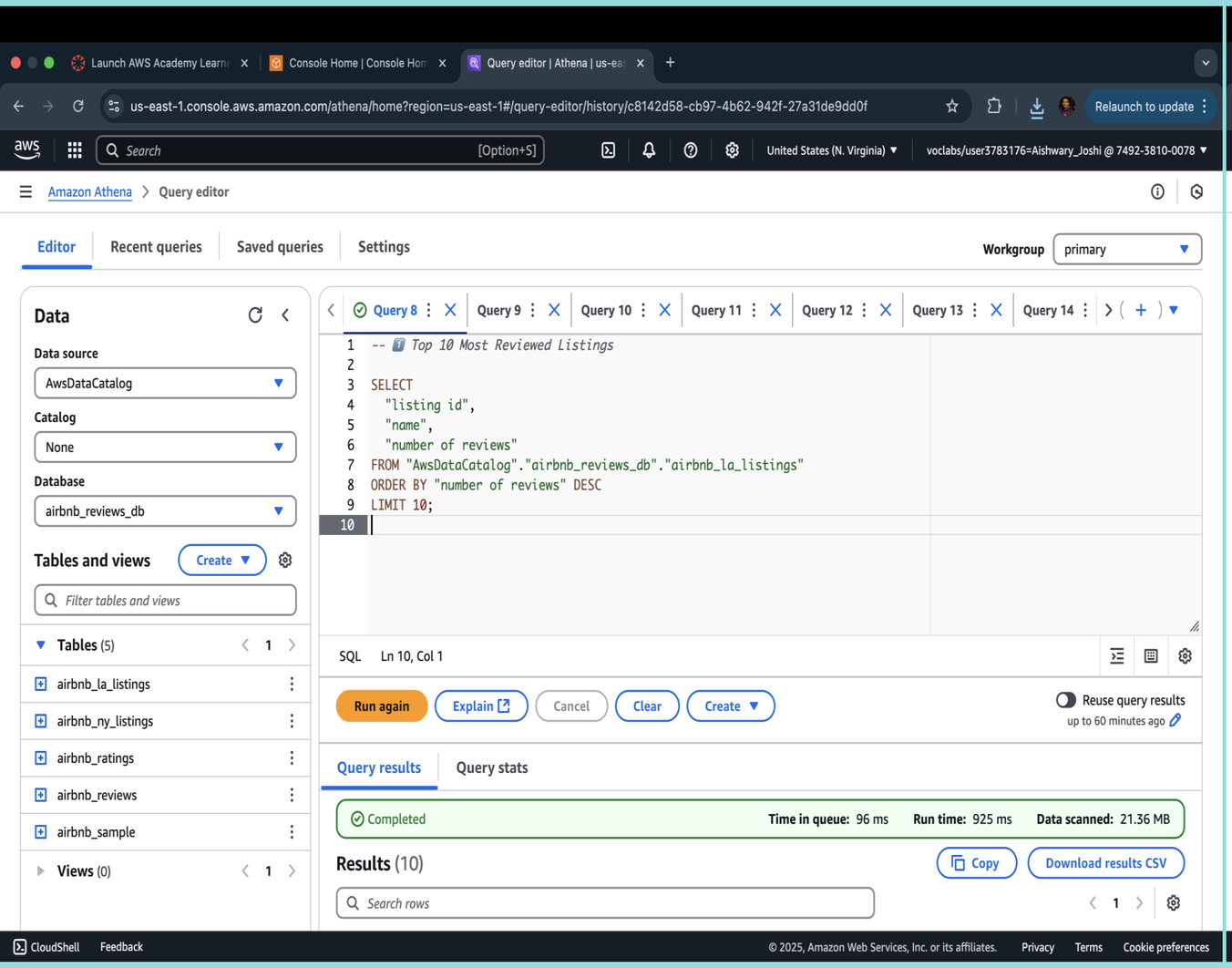
CRAWLERS

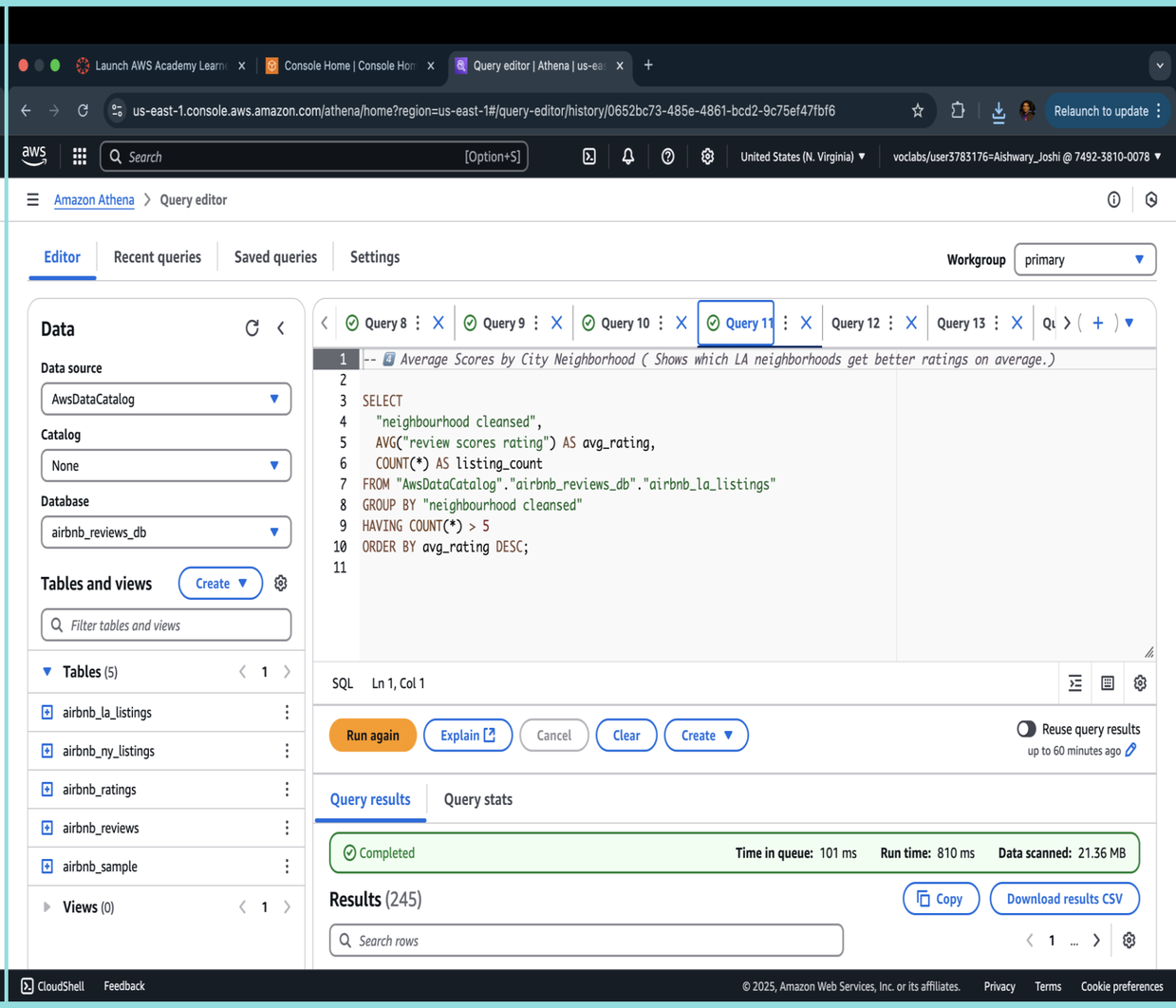
Insights from SQL Queries

- Avg. Rating by Room Type (Query 9)
- Avg. Rating by Neighborhood (Query 11)
- Top Reviewed Listings (Query 10)



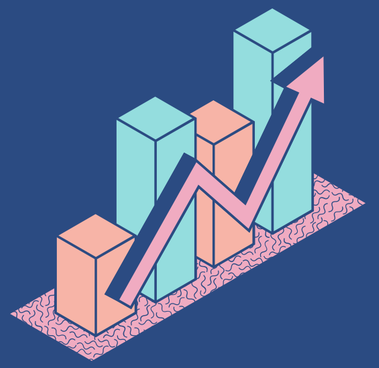
SQL Queries

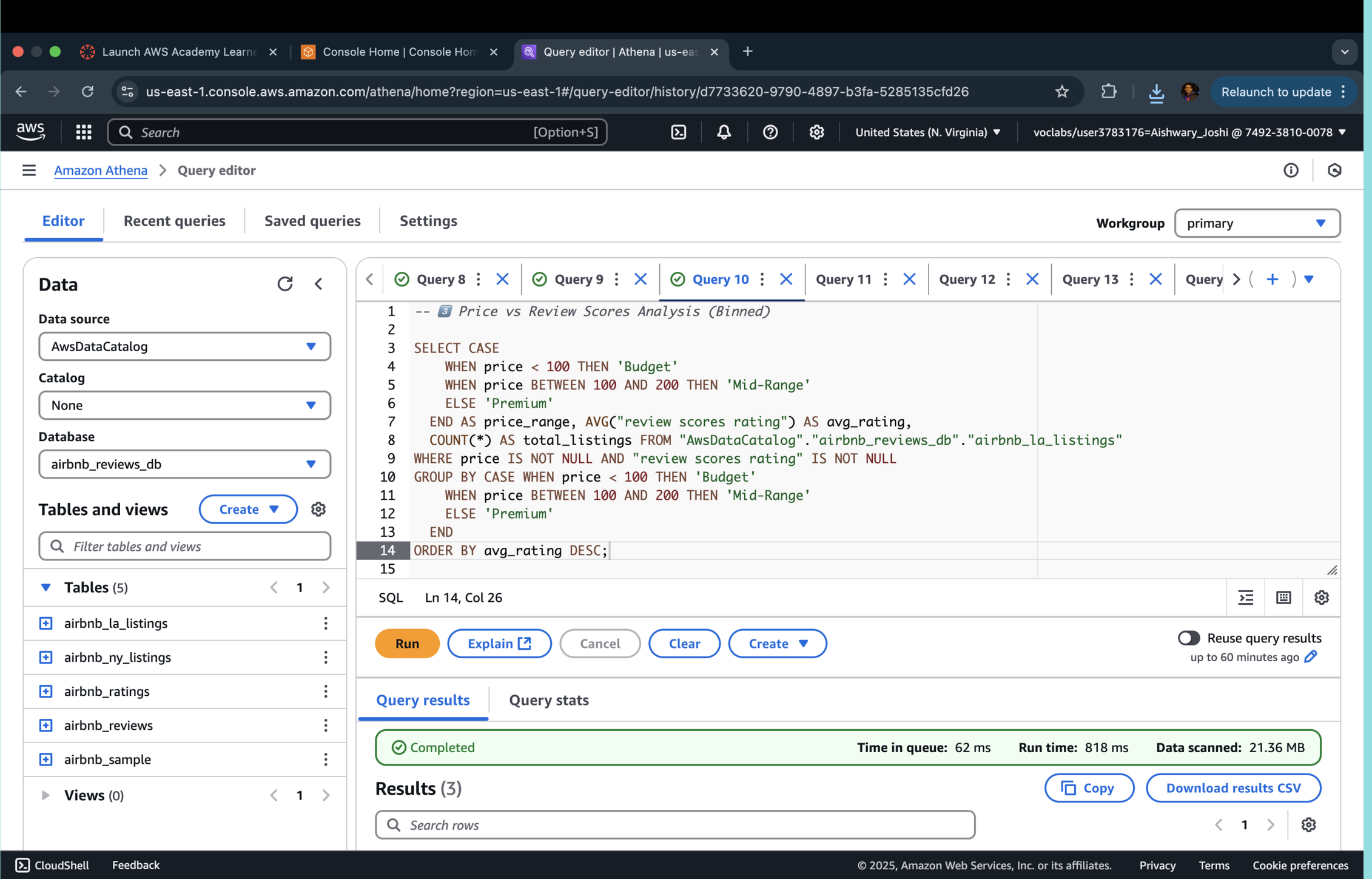

Launch AWS Academy Learn
Console Home | Console Hom
Query editor | Athena | us-ea
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/c8142d58-cb97-4b62-942f-27a31de9dd0f
Relaunch to update
Search [Option+S]
United States (N. Virginia)
voclabs/user3783176=Aishwary_Joshi @ 7492-3810-0078
Amazon Athena > Query editor
Editor
Recent queries
Saved queries
Settings
Workgroup primary
Data
Data source
AwsDataCatalog
Catalog
None
Database
airbnb_reviews_db
Tables and views
Create
Filter tables and views
Tables (5)
airbnb_la_listings
airbnb_ny_listings
airbnb_ratings
airbnb_reviews
airbnb_sample
Views (0)
Query 8
Query 9
Query 10
Query 11
Query 12
Query 13
Query 14
1 -- Top 10 Most Reviewed Listings
2
3 SELECT
4 "listing id",
5 "name",
6 "number of reviews"
7 FROM "AwsDataCatalog"."airbnb_reviews_db"."airbnb_la_listings"
8 ORDER BY "number of reviews" DESC
9 LIMIT 10;
10
SQL Ln 10, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
up to 60 minutes ago
Query results
Query stats
Completed
Time in queue: 96 ms
Run time: 925 ms
Data scanned: 21.36 MB
Results (10)
Copy
Download results CSV
Search rows
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Launch AWS Academy Learn
Console Home | Console Hom
Query editor | Athena | us-ea
us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/0652bc73-485e-4861-bcd2-9c75ef47fbf6
Relaunch to update
Search [Option+S]
United States (N. Virginia)
voclabs/user3783176=Aishwary_Joshi @ 7492-3810-0078
Amazon Athena > Query editor
Editor
Recent queries
Saved queries
Settings
Workgroup primary
Data
Data source
AwsDataCatalog
Catalog
None
Database
airbnb_reviews_db
Tables and views
Create
Filter tables and views
Tables (5)
airbnb_la_listings
airbnb_ny_listings
airbnb_ratings
airbnb_reviews
airbnb_sample
Views (0)
Query 8
Query 9
Query 10
Query 11
Query 12
Query 13
1 -- Average Scores by City Neighborhood (Shows which LA neighborhoods get better ratings on average.)
2
3 SELECT
4 "neighbourhood cleansed",
5 AVG("review scores rating") AS avg_rating,
6 COUNT(*) AS listing_count
7 FROM "AwsDataCatalog"."airbnb_reviews_db"."airbnb_la_listings"
8 GROUP BY "neighbourhood cleansed"
9 HAVING COUNT(*) > 5
10 ORDER BY avg_rating DESC;
11
SQL Ln 1, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
up to 60 minutes ago
Query results
Query stats
Completed
Time in queue: 101 ms
Run time: 810 ms
Data scanned: 21.36 MB
Results (245)
Copy
Download results CSV
Search rows
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

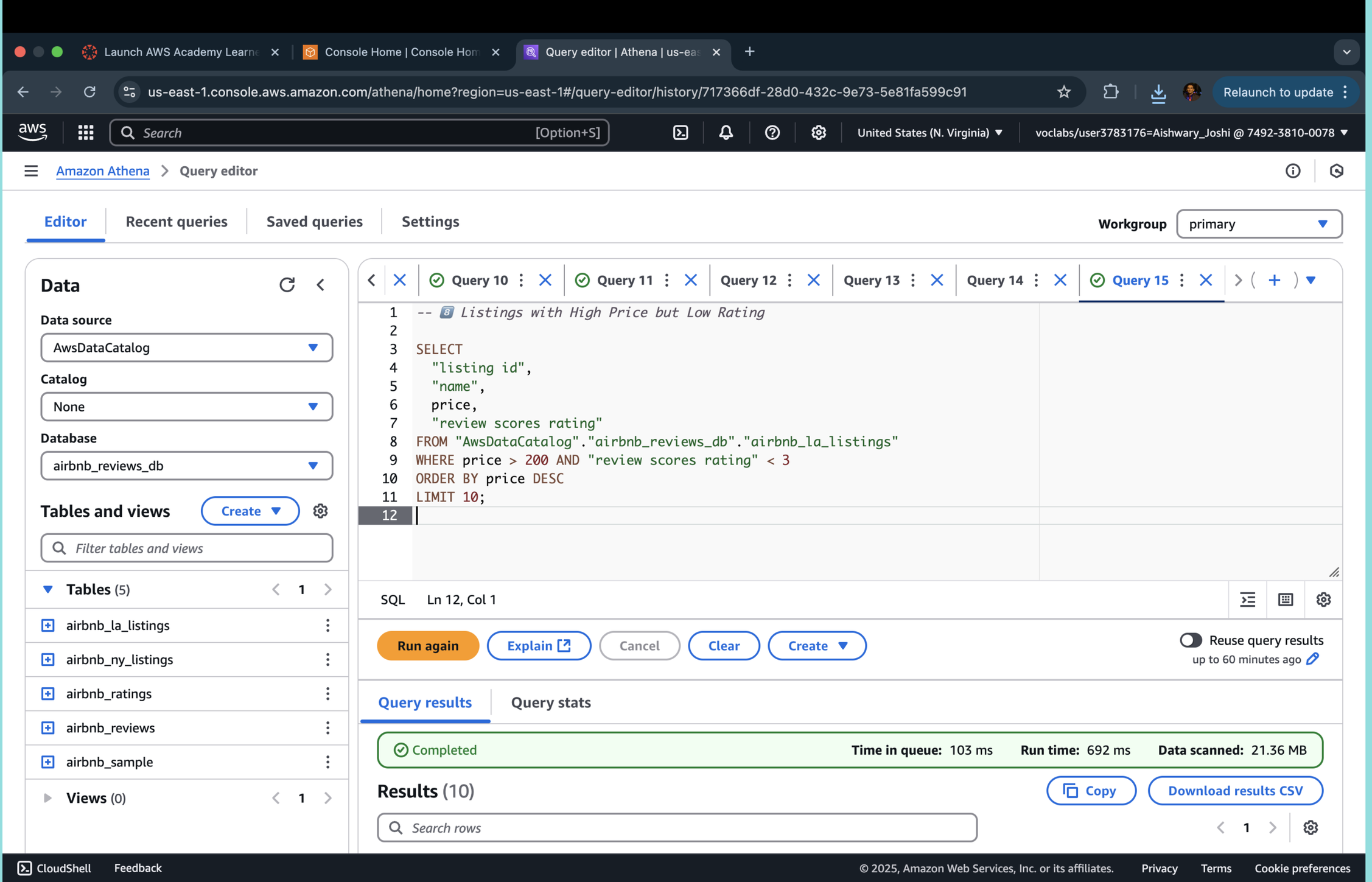
Advanced Analytics Using Athena

- Diving deeper, we ran queries like:
  1. Price vs Review Score (binned) (Query 10)
  2. Listings with High Price but Low Rating: potential anomalies (Query 15)
  3. Top Hosts by Rating (Query 12)

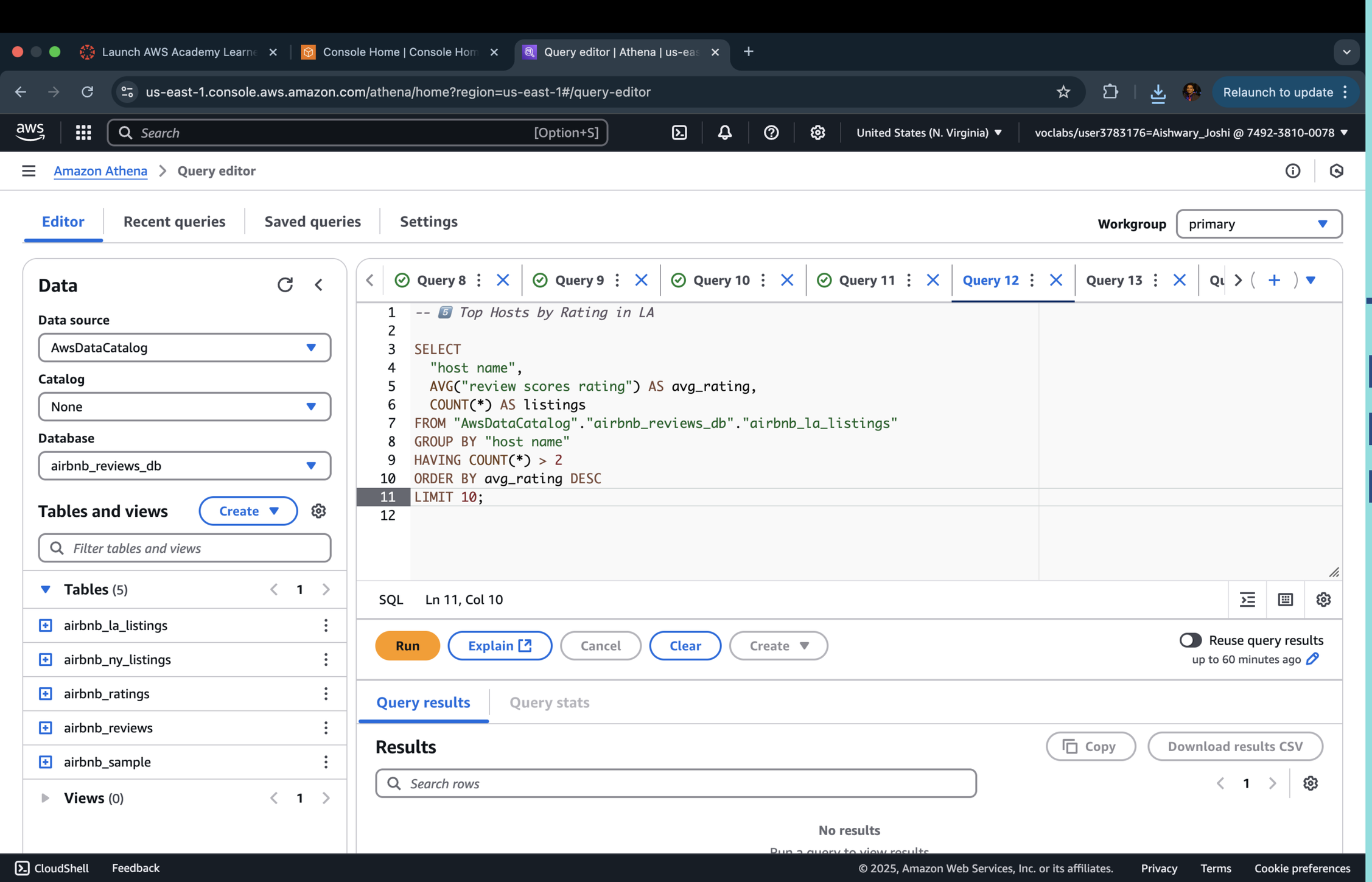




PRICE Vs REVIEW



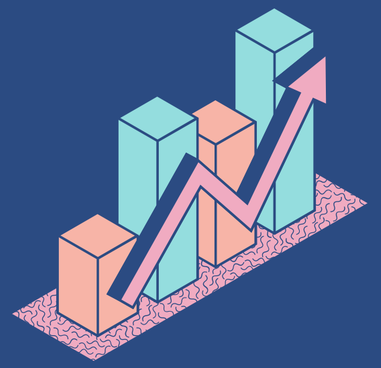
HIGH PRICE BUT LOW RATING

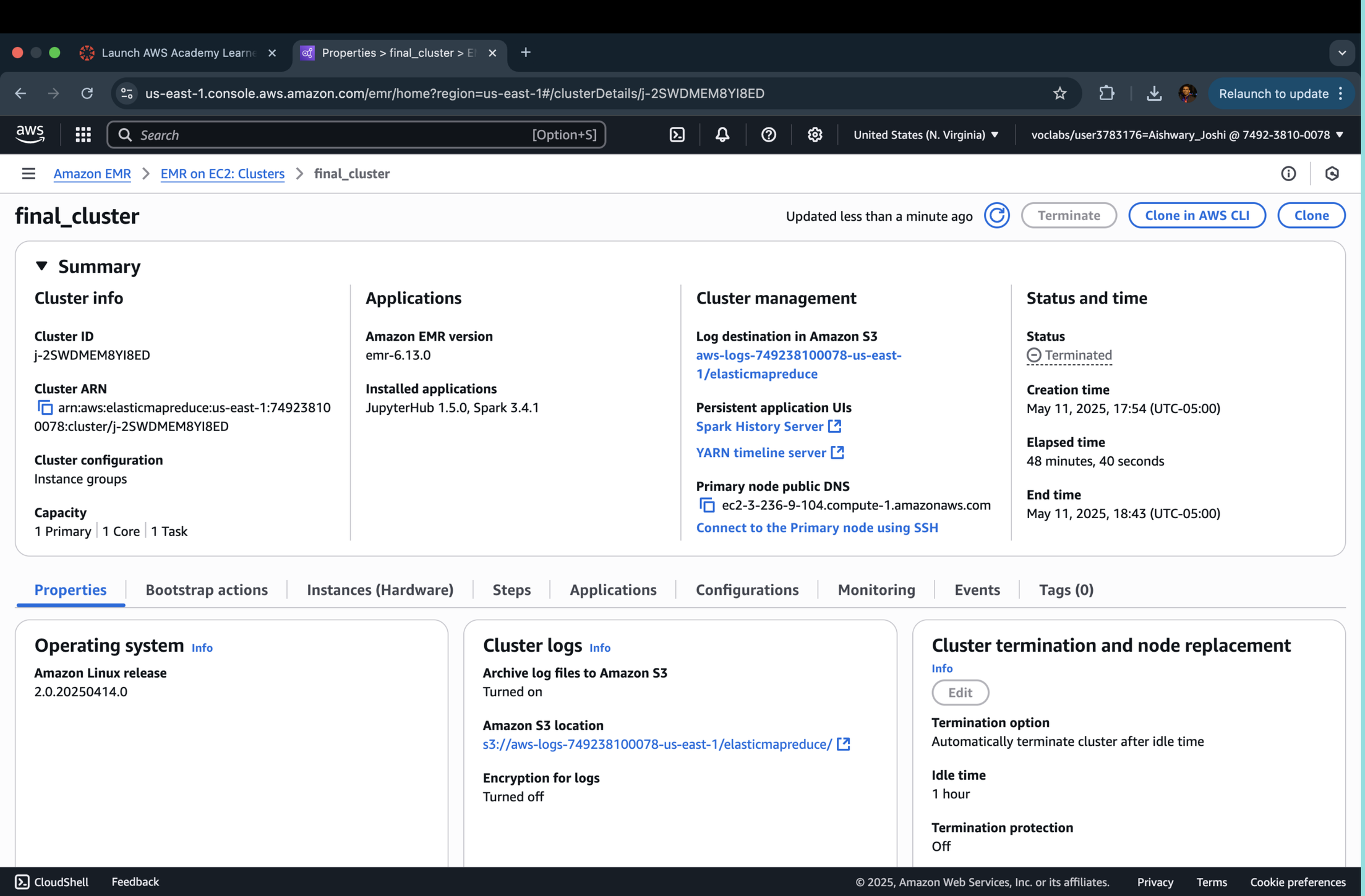


TOP HOSTS BY RATINGS

Machine Learning With EMR

- To conduct ML experiments:
- Launched Amazon EMR Cluster (JupyterHub + Spark)
- Used Docker to deploy a Jupyter Notebook
- Cluster ID: j-2SWDMEM8YI8ED
- Docker-based Jupyter Notebook setup
- Installed sklearn, xgboost, pandas





EMR ON EC2 CLUSTER

EMR Setup Process

1 STEP

SSH into master node using .pem file

2 STEP

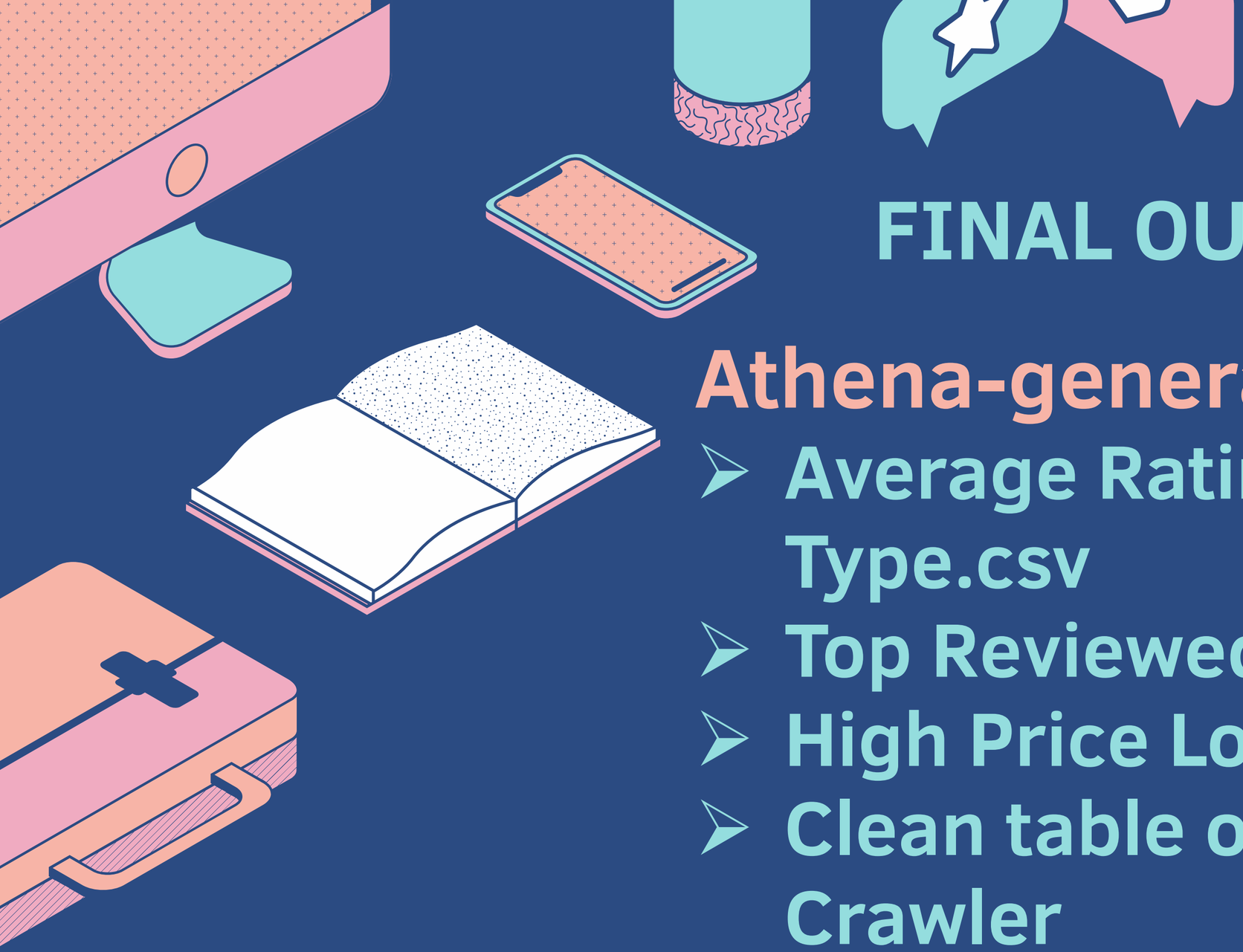
Docker installation and permission setup

3 STEP

Container launch of Jupyter Lab

4 STEP

Secure token-based access on localhost:8888 Referenced from emr_setup.docx, which explains setup and fixes



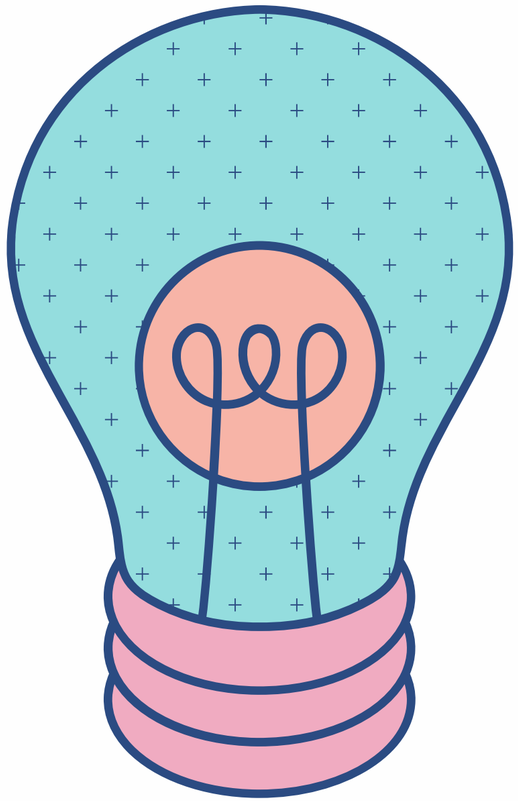
FINAL OUTPUTS

Athena-generated CSVs:

- Average Rating by Room Type.csv
- Top Reviewed Listings.csv
- High Price Low Rating.csv
- Clean table outputs from Glue Crawler

Challenges and Resolutions

Challenge	Resolution
Lambda IAM issues	Solved by assigning correct LabRole
Docker runtime errors	Resolved via manual install
Missing Python packages	• During initial model experimentation in the Jupyter Notebook hosted on EMR, we encountered errors related to missing essential Python libraries such as scikit-learn, xgboost, and pandas. • These libraries were not pre-installed in the Docker-based environment we had configured within the EMR master node. • We accessed the Docker container's shell environment and used the pip package manager to manually install the required libraries. • After installation, we verified that each package was correctly recognized and usable by the active Jupyter kernel. • This process was essential for ensuring that the ML code could execute without interruption and remain compatible with the Spark and Hadoop dependencies present on the EMR cluster.
Schema inconsistency	Cleaned during preprocessing phase. Described using lambda_code.py and emr_setup.docx



FINAL TAKEAWAYS

✅ Fully automated AWS pipeline for structured insights

✅ Integrated SQL, Serverless, and ML in one workflow

✅ Production-ready setup with scalability potential

Future Enhancements

- Add Stream Ingestion via Kinesis
- Use SageMaker for model training
- Add BI Layer: Amazon QuickSight for real-time dashboards

THANK YOU!