Data Lake Implementation

Data Lake Implementation using AWS Glue and Athena

• Services: S3, Glue, Athena, IAM

• Description: Create a data lake on S3, catalog the data using AWS Glue, and perform queries using Athena. Configure IAM policies for secure access.

• Skills: Data cataloging, serverless querying, data lake architecture.

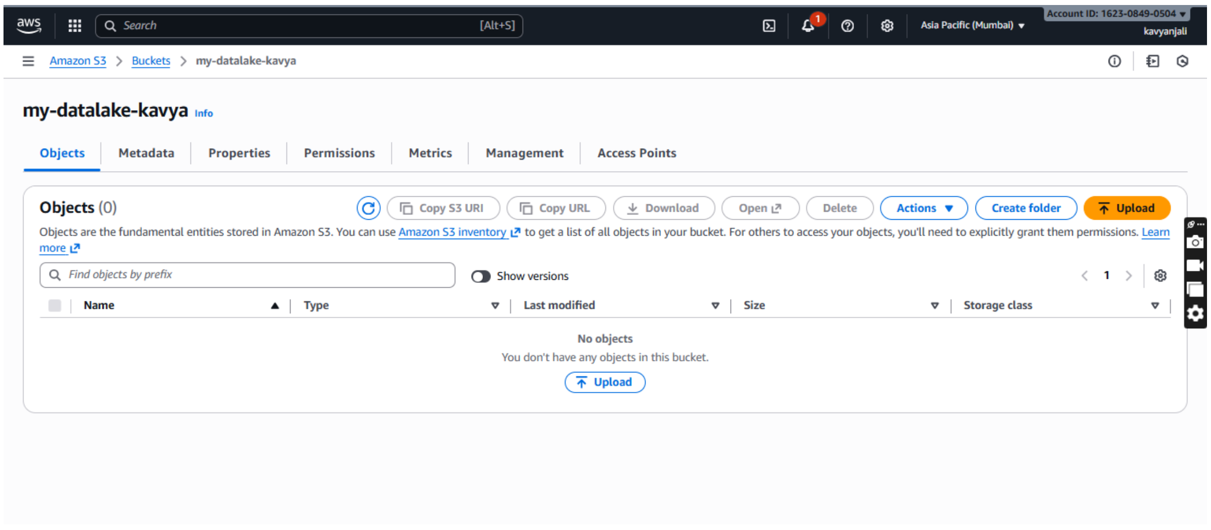
Assumptions: you have AWS Console access in one AWS account/region and an IAM user with privileges to create S3 buckets, IAM roles/policies, Glue resources and Athena workgroups

---

1) Create the S3 bucket and folder structure

1. Console: Services → S3 → **Create bucket**.
    - Bucket name: my-datalake-kavya (must be globally unique)
    - Region: Mumbai-ap-south-1
    - Uncheck "Block all public access" only if you need public — *recommended: keep block public access ON*.
2. Under **Default encryption**, choose **AWS-KMS** (SSE-KMS) or **SSE-S3**. Recommended: **SSE-KMS** with a CMK you control for auditability. (If using SSE-KMS, you may want to enable S3 Bucket Keys to reduce KMS costsUnder **Versioning**, enable versioning (useful for data recovery).
3. Create bucket.
4. In the bucket, create prefixes (folders):
    - raw/ — landing zone (original files)
    - staging/ — temporary/processing
    - curated/ — cleaned, partitioned datasets for analytics
    - athena-results/ — Athena query results (or set a dedicated results bucket/prefix)
5. Configure lifecycle rules (optional): move raw/ older than 90 days to S3 Glacier or expiration for temp data.

Why: separate prefixes help Glue crawlers and permissions. Use SSE-KMS to control keys and audit key usage.



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

No objects

You don't have any objects in this bucket.

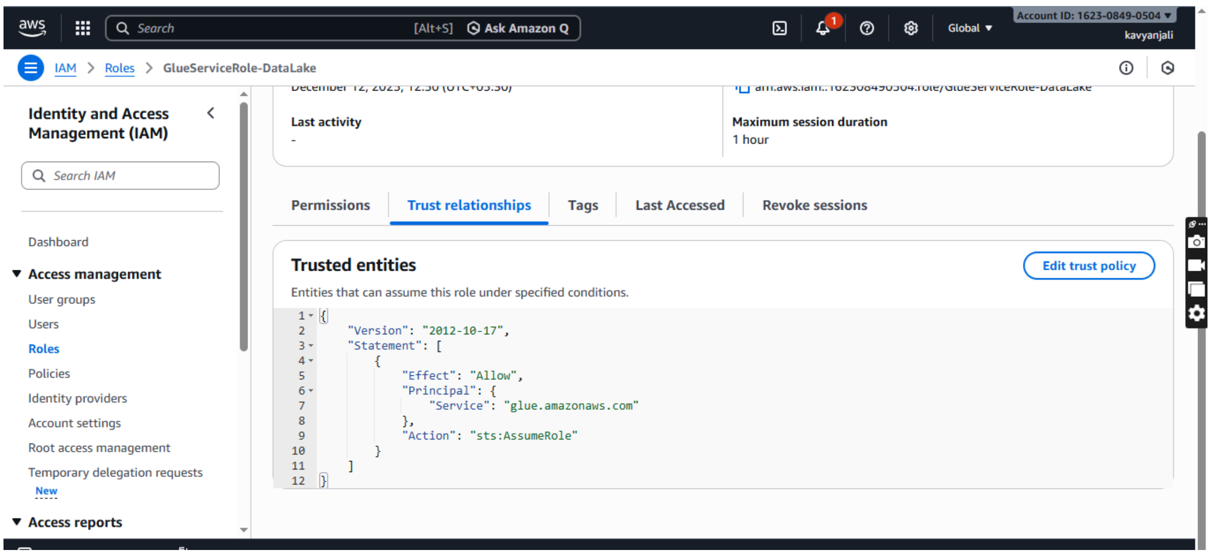
2) Create IAM role(s) for AWS Glue and Athena

You need roles/policies for:

- **Glue service role** — allows Glue to read from S3, write logs to CloudWatch, use KMS if objects encrypted.
- **User/analytic role** — permissions for analysts to run Athena queries and read Glue Data Catalog.

A. Create an IAM role for Glue (console)

1. Console: Services → IAM → Roles → **Create role**.
2. Select **Glue** as the trusted service (service that will use the role) → Next.
3. Attach policies:
   - AmazonS3ReadOnlyAccess (or a custom S3 policy limited to your bucket prefixes)
   - CloudWatchLogsFullAccess (or a narrower logging policy)
   - If using KMS: add AWSKeyManagementServicePowerUser or a narrow KMS usage policy allowing kms:Decrypt, kms:Encrypt, kms:GenerateDataKey on the CMK ARN.
4. Name: GlueServiceRole-DataLake = Create role.



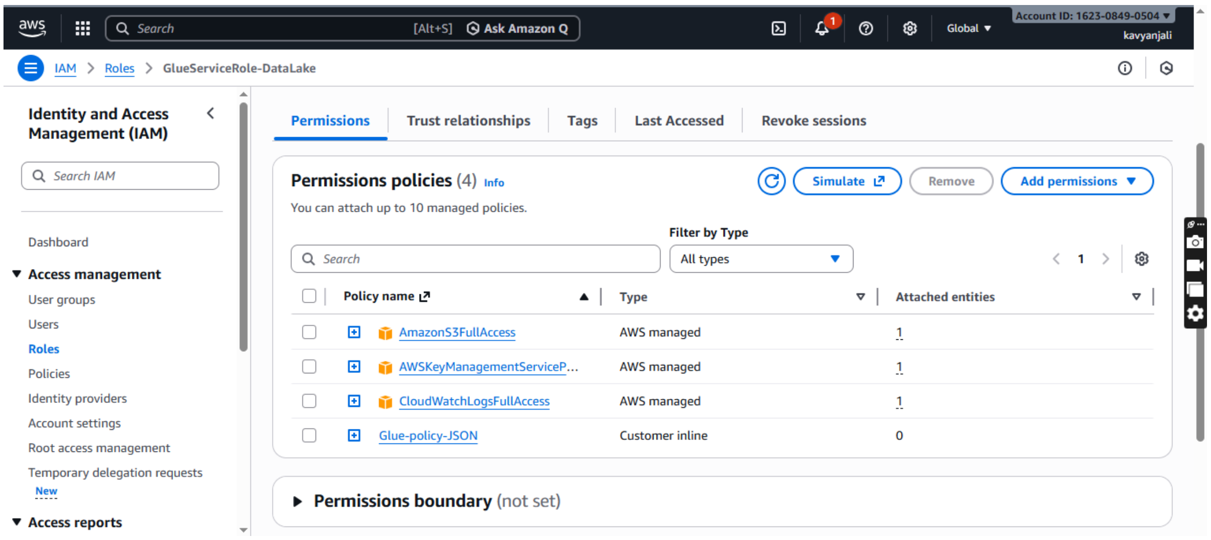
Last activity
-

Maximum session duration
1 hour

Trusted entities

Entities that can assume this role under specified conditions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```



Permissions policies (4) Info

You can attach up to 10 managed policies.

	Policy name	Type	Attached entities
☐	AmazonS3FullAccess	AWS managed	1
☐	AWSKeyManagementServiceP...	AWS managed	1
☐	CloudWatchLogsFullAccess	AWS managed	1
☐	Glue-policy-JSON	Customer inline	0

▶ **Permissions boundary** (not set)

Step 1 — Open IAM → Roles → Glue role

- Find: GlueServiceRole-DataLake
- Go to **Permissions**
- Click **Add permissions → Create Inline Policy**

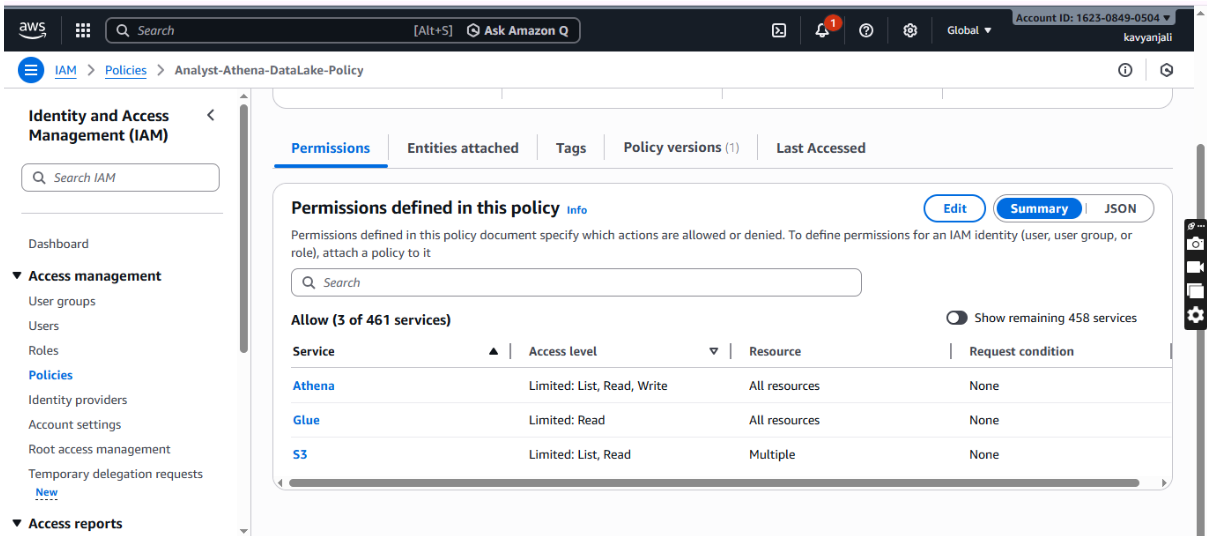
- Paste the **Glue policy JSON**
- Save

Step 2 — Create Analyst Policy

- IAM → Policies → **Create Policy**
- Paste Analyst JSON
- Save as: Analyst-Athena-DataLake-Policy
- Attach to:
  - Your analyst IAM user

Step 3 — Done

Glue, Athena, and S3 will work smoothly under SSE-S3.


IAM > Policies > Analyst-Athena-DataLake-Policy

Permissions | Entities attached | Tags | Policy versions (1) | Last Accessed

Permissions defined in this policy

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Allow (3 of 461 services)

Show remaining 458 services

Service	Access level	Resource	Request condition
Athena	Limited: List, Read, Write	All resources	None
Glue	Limited: Read	All resources	None
S3	Limited: List, Read	Multiple	None

3) Create AWS Glue Data Catalog (database) & Crawler

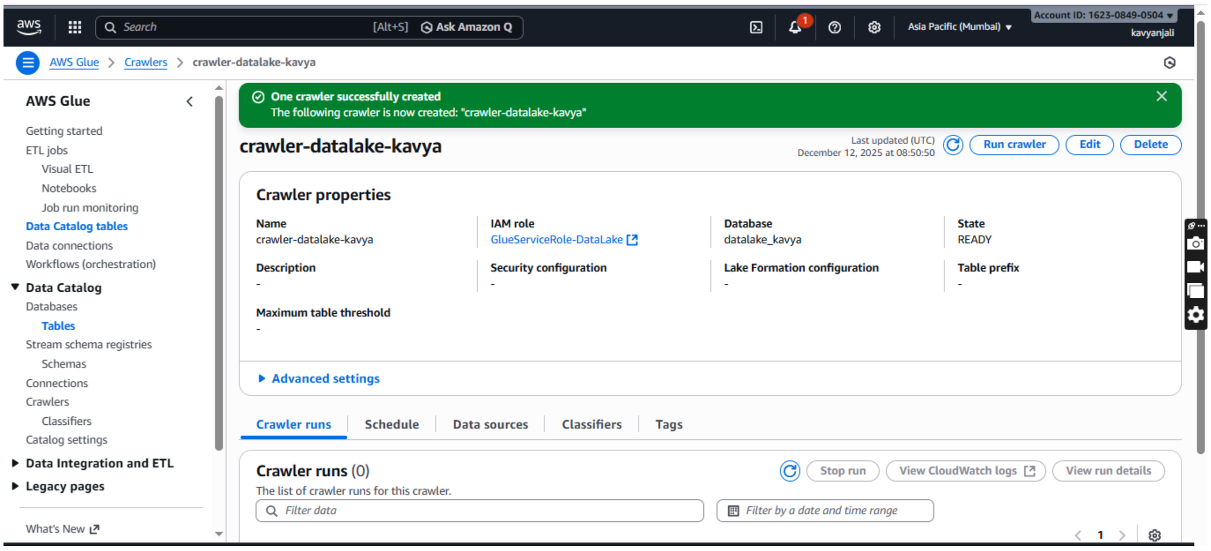
A. Create a Glue database

1. Console: Services → AWS Glue → Databases → **Add database**.
2. Database name: datalake_kavya (This is the Glue Data Catalog database used by tables.)

B. Create a Glue crawler that discovers tables from S3

1. Console: AWS Glue → Crawlers → **Create Crawler**.
2. Name: crawler-datalake-raw-kavya → Next.
3. Data source: choose **S3**, enter target path: s3://my-datalake-kavya/ → Next.
4. IAM role: choose GlueServiceRole-DataLake- (the role you created). Ensure role has permission to access S3 and KMS (if encryption).
5. Choose schedule: on-demand or periodic (e.g., run hourly/daily) depending on ingestion cadence.
6. Output: choose the Glue database datalake_ and option to add/update tables. Configure table name prefix if desired.
7. Advanced options: set classifiers (JSON, CSV, Parquet). If partitions follow folder structure like year=2025/month=12/, Glue can detect partitions. Configure crawler behavior on schema changes (update/add).
8. Create and run the crawler. After a run, the Glue Data Catalog will contain tables representing discovered datasets.

Why: Crawlers populate tables and partition metadata automatically so Athena/Glue ETL jobs can query efficiently.



AWS Glue > Crawlers > crawler-datalake-kavya

One crawler successfully created
The following crawler is now created: "crawler-datalake-kavya"

crawler-datalake-kavya
Last updated (UTC) December 12, 2025 at 08:50:50

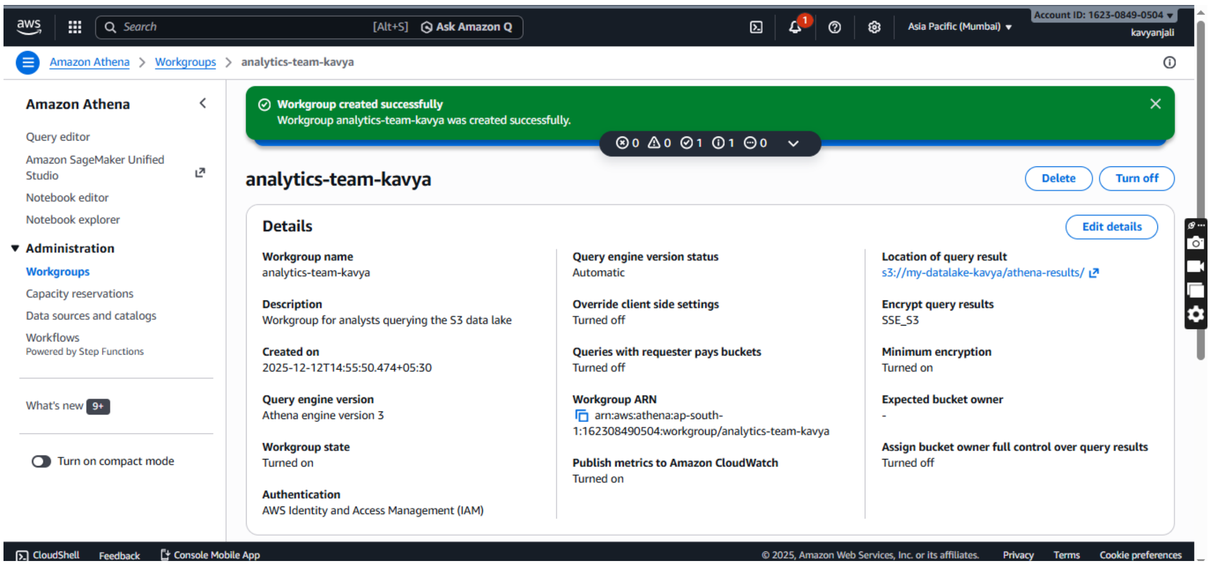
Crawler properties

Name	IAM role	Database	State
crawler-datalake-kavya	GlueServiceRole-DataLake	datalake_kavya	READY
Description	**Security configuration**	**Lake Formation configuration**	**Table prefix**
-	-	-	-
Maximum table threshold			
-			

Crawler runs (0)
The list of crawler runs for this crawler.

4) Configure Amazon Athena (workgroup & query result location)

1. Console: Services → Athena → **Get started** (or go to Query Editor).
2. Query result location: Click **Settings** (top-right) → set **Query result location** to s3://my-datalake-kavya/athena-results/. Athena needs WRITE access to this prefix.
3. Create a **Workgroup** for the team: Athena Console → Workgroups → **Create workgroup**. Use a descriptive name — e.g., analytics-team-kavya. Set:
   - Result configuration override: ensure result location is s3://.../athena-results/.
   - Encryption: enable encryption of query results (SSE-S3 or SSE-KMS) — choose same KMS key if you want centralized key control.
   - Optional: set data processing limits (bytes scanned) per query or per workgroup to prevent runaway cost.



Amazon Athena > Workgroups > analytics-team-kavya

Workgroup created successfully
Workgroup analytics-team-kavya was created successfully.

analytics-team-kavya

Details

Workgroup name	Query engine version status	Location of query result
analytics-team-kavya	Automatic	s3://my-datalake-kavya/athena-results/
Description	**Override client side settings**	**Encrypt query results**
Workgroup for analysts querying the S3 data lake	Turned off	SSE_S3
Created on	**Queries with requester pays buckets**	**Minimum encryption**
2025-12-12T14:55:50.474+05:30	Turned off	Turned on
Query engine version	**Workgroup ARN**	**Expected bucket owner**
Athena engine version 3	arn:aws:athena:ap-south-1:162308490504:workgroup/analytics-team-kavya	-
Workgroup state	**Publish metrics to Amazon CloudWatch**	**Assign bucket owner full control over query results**
Turned on	Turned on	Turned off
Authentication		
AWS Identity and Access Management (IAM)		

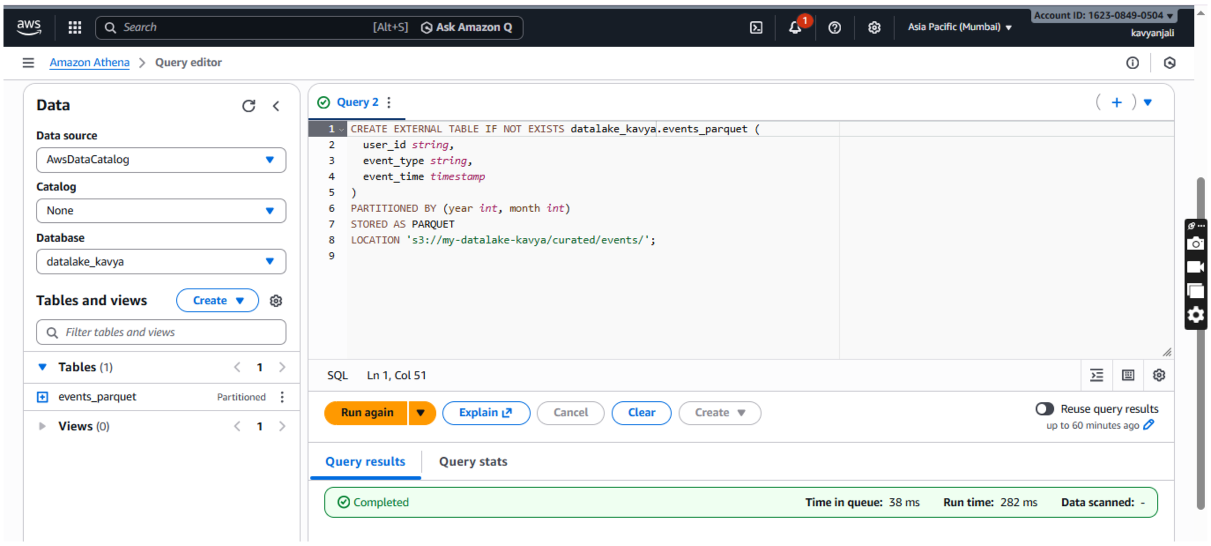
Switch to your new Workgroup

After creation: Athena → Query Editor = Top-left dropdown → Select: analytics-team-kavya

5) Querying data with Athena (examples)

- Use the Glue Data Catalog tables the crawler created. In Athena Query Editor, select the Glue database datalake_kavya
- create an external table manually (if you prefer to register manually):

```
CREATE EXTERNAL TABLE IF NOT EXISTS datalake_kavya.events_parquet (

  user_id string,

  event_type string,

  event_time timestamp

)

PARTITIONED BY (year int, month int)

STORED AS PARQUET

LOCATION 's3://my-datalake-kavya/curated/events/';
```



Data source: AwsDataCatalog
Catalog: None
Database: datalake_kavya

```
CREATE EXTERNAL TABLE IF NOT EXISTS datalake_kavya.events_parquet (
  user_id string,
  event_type string,
  event_time timestamp
)
PARTITIONED BY (year int, month int)
STORED AS PARQUET
LOCATION 's3://my-datalake-kavya/curated/events/';
```

SQL   Ln 1, Col 51

Completed   **Time in queue:** 38 ms   **Run time:** 282 ms   **Data scanned:** -

Then run MSCK REPAIR TABLE datalake_kavya.events_parquet; to pick up partitions (if Glue crawler not used for this table).

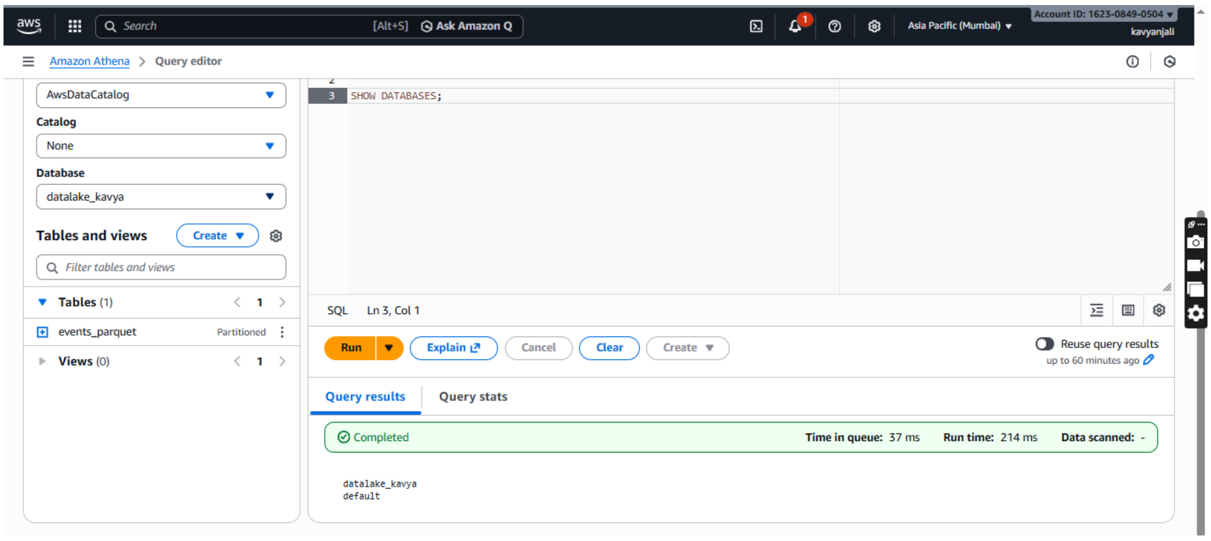
Tip: Use EXPLAIN and SHOW CREATE TABLE to inspect tables and optimize. Partitioning and Parquet significantly reduce scanned data and cost.

List databases in Athena (quick check)

Run in Athena Query Editor:

```
SHOW DATABASES;
```

Look for datalake_kavya (or your datalake_<your-prefix>).
If it’s not listed → the Glue database wasn’t created in this account/region.



AwsDataCatalog
Catalog: None
Database: datalake_kavya
Tables (1)
events_parquet Partitioned
Views (0)

```
SHOW DATABASES;
```

SQL Ln 3, Col 1

Completed Time in queue: 37 ms Run time: 214 ms Data scanned: -

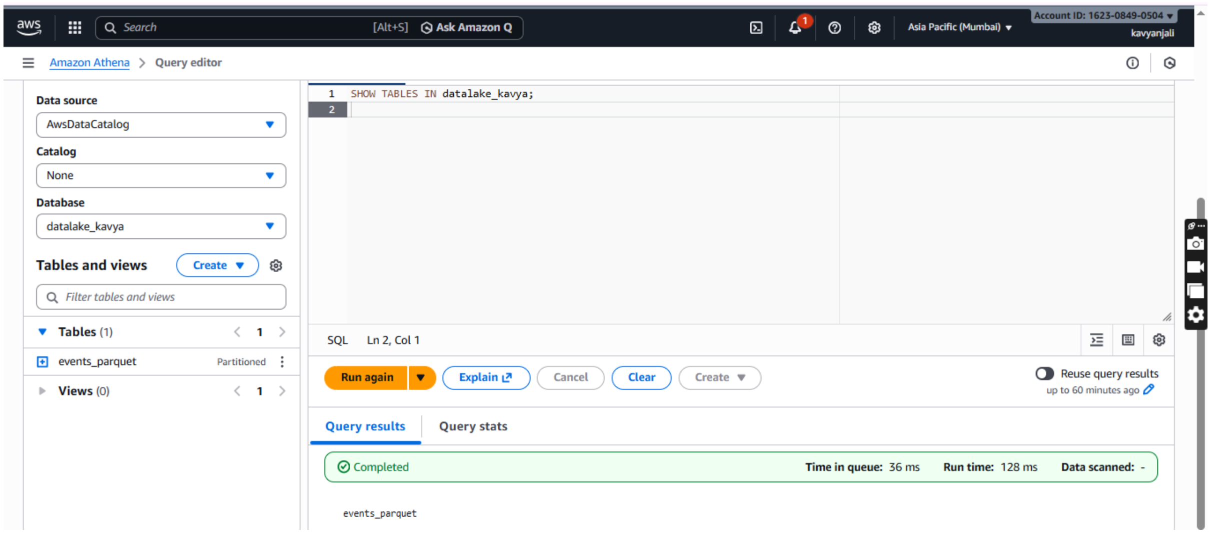
```
datalake_kavya
default
```

List tables in your Glue database

If datalake_kavya is present, run:

SHOW TABLES IN datalake_kavya;

- If the result is empty: no tables exist → crawler didn't create tables, or created them in a different database.
- If the table you expect appears but with a different name, use that exact name in your query.



Data source: AwsDataCatalog
Catalog: None
Database: datalake_kavya
Tables (1)
events_parquet Partitioned
Views (0)

```
SHOW TABLES IN datalake_kavya;
```

SQL Ln 2, Col 1

Completed Time in queue: 36 ms Run time: 128 ms Data scanned: -

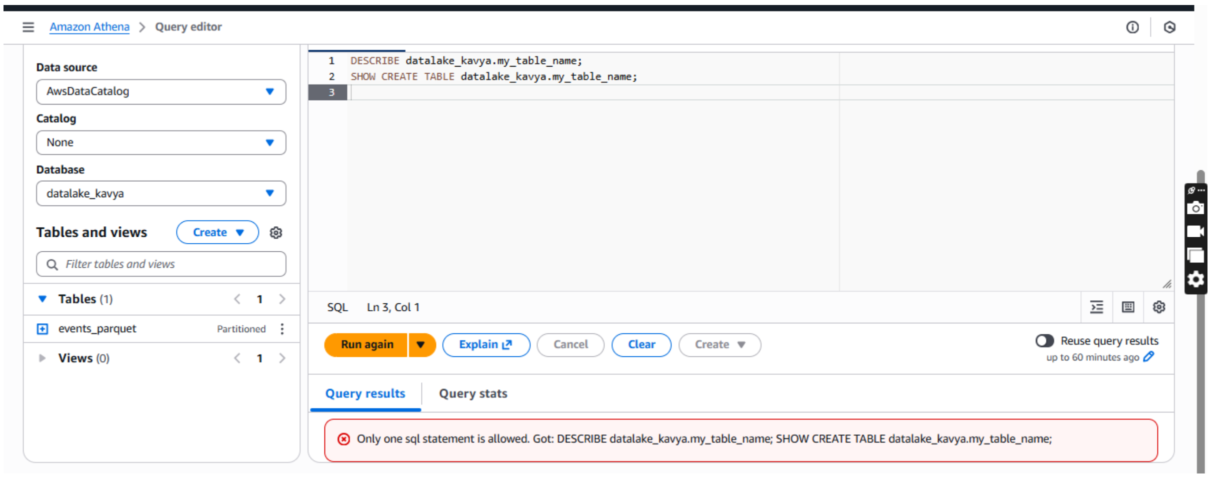
```
events_parquet
```

Check table exists and schema

If you see the table name, inspect it:

DESCRIBE datalake_kavya.my_table_name;
SHOW CREATE TABLE datalake_kavya.my_table_name;

If DESCRIBE fails with TABLE_NOT_FOUND, it truly doesn't exist.



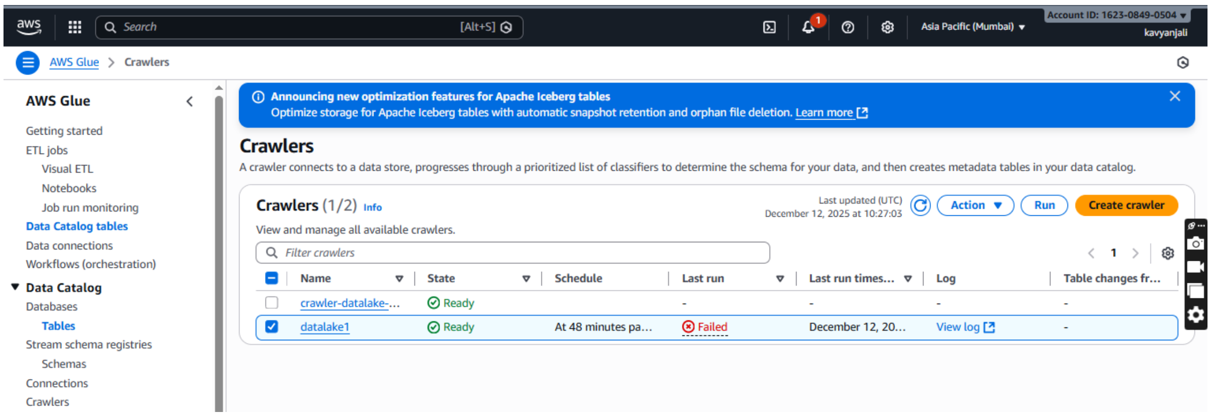
```
DESCRIBE datalake_kavya.my_table_name;
SHOW CREATE TABLE datalake_kavya.my_table_name;
```

Only one sql statement is allowed. Got: DESCRIBE datalake_kavya.my_table_name; SHOW CREATE TABLE datalake_kavya.my_table_name;

If you don't see tables → verify the Glue crawler

Go to the Console:
AWS Console → AWS Glue → Data Catalog → Crawlers

- Confirm the crawler crawler-datalake-raw-kavya ->exists.
- Check its **Last run** status: Succeeded, Failed, or Running.
- If it never ran or failed, select it and click **Run crawler**.



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 (1/2)

Last updated (UTC) December 12, 2025 at 10:27:03

Name	State	Schedule	Last run	Last run times...	Log	Table changes fr...
crawler-datalake-...	Ready		-	-	-	-
datalake1	Ready	At 48 minutes pa...	Failed	December 12, 20...	View log	-

If it failed, click the crawler and open **Logs** (there will be a link to CloudWatch logs) and inspect the error — common errors: incorrect S3 path, AccessDenied on S3, incorrect IAM role.

Verify the crawler's S3 path

Open the crawler configuration and check the S3 path the crawler is pointed at. Make sure it exactly matches where your data files are (e.g. s3://my-datalake-kavya/raw/).

Common mistake: data is in raw/events/ but crawler pointed to raw/other/.

Check IAM permissions (quick)

Ensure the **Glue role** (GlueServiceRole-...) has S3 read access for the bucket:

- **IAM → Roles → GlueServiceRole-... → Permissions**
  Policy must allow at least:
  - s3:ListBucket on arn:aws:s3:::my-datalake-kavya
  - s3:GetObject on arn:aws:s3:::my-datalake-kavya/*

Also ensure analyst user has glue:GetTable, glue:GetTables, and athena:StartQueryExecution (you added these earlier).

6) Fine-grained access control & data catalog security

- You can apply resource-level IAM permissions to Glue databases and tables and control which principals can query which Data Catalogs/tables. Athena supports fine-grained policies for glue resources. For cross-account catalogs, follow Athena's cross-account Glue catalog docs.
- Use IAM policies to restrict S3 prefixes (raw vs curated) and Athena workgroups to separate users/teams.

7) Logging, auditing, and cost control

- Enable **CloudTrail** to capture Glue/Athena/S3 API calls for auditing.
- Athena Query logs: configure CloudWatch or S3 result location and track bytes scanned. Use Athena workgroup limits to control spend.

8) Security best practices (summary)

- Use SSE-KMS + S3 Bucket Keys for encryption and KMS key policies (preferred for sensitive data).
- Apply least-privilege IAM policies and validate with IAM Access Analyzer.
- Use partitioning/columnar formats to reduce data scanned (cost) and improve query performance.
- Keep Glue crawler roles minimal and restrict Glue job role to needed S3 prefixes and KMS keys only.

Quick checklist (what to do in order)

1. Create S3 bucket my-datalake-kavya .
2. enable SSE-KMS & versioning; create prefixes.
3. Create KMS key (if SSE-KMS) and ensure Glue & Athena IAM roles have usage permissions.
4. Create IAM roles: GlueServiceRole-DataLake, attach S3/KMS/CloudWatch access. Create analyst policy for Athena + Glue read.
5. In Glue: create database datalake_kavya
6. Create & run Glue crawler(s) pointed to raw/ and curated/. Confirm tables appear in Glue Data Catalog.
7. Create Athena workgroup, set query result location, enable encryption for results.
8. Run test queries in Athena; convert curated data to Parquet and partition via Glue ETL jobs for production.
9. Configure CloudTrail & CloudWatch logs, lifecycle rules and cost limits.

Useful AWS docs (most load-bearing)

- AWS Glue: Create a crawler / Using crawlers to populate the Data Catal
- Amazon Athena: Create workgroups & query configuration.
- S3 encryption / SSE-KMS & S3 Bucket Keys.
- Glue IAM policy examples & best practices.

Conclusion

This project successfully implemented a scalable and secure data lake using AWS services. Amazon S3 served as the central storage layer, while AWS Glue automated schema discovery and created a unified data catalog. Amazon Athena enabled fast, serverless SQL queries directly on S3 without the need for infrastructure management. IAM policies ensured controlled and secure access to all components. Overall, the project demonstrates strong skills in data lake architecture, data cataloging, and serverless analytics using AWS Glue and Athena.