

Uploading Files to S3 and Querying Using AWS Athena

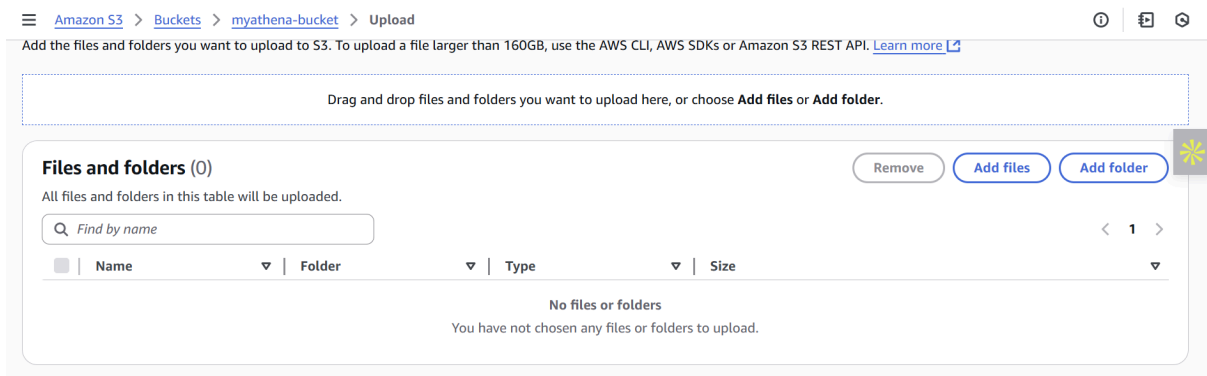
Purpose:

This SOP outlines the steps to upload data to S3, define a table schema, and run queries in AWS Athena.

Procedures:

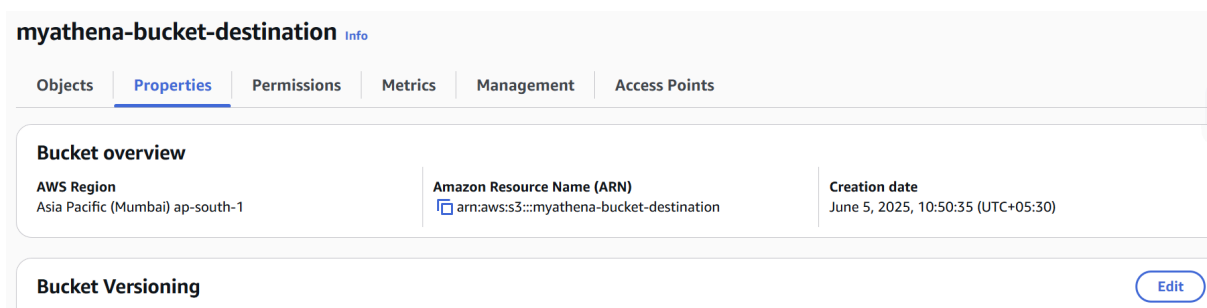
Step 1. Uploading files to S3

1. Log in to the AWS Console.
2. Go to S3 and click [Create bucket](#) to create a new bucket.
3. In the General configuration, add bucket name (eg: new-bucket) and keep other configuration the same for now, and click on create bucket.
4. Click on the newly created bucket and click upload.
5. Click on Add files and select the file that needs to be uploaded, and click upload (eg: apache.log)



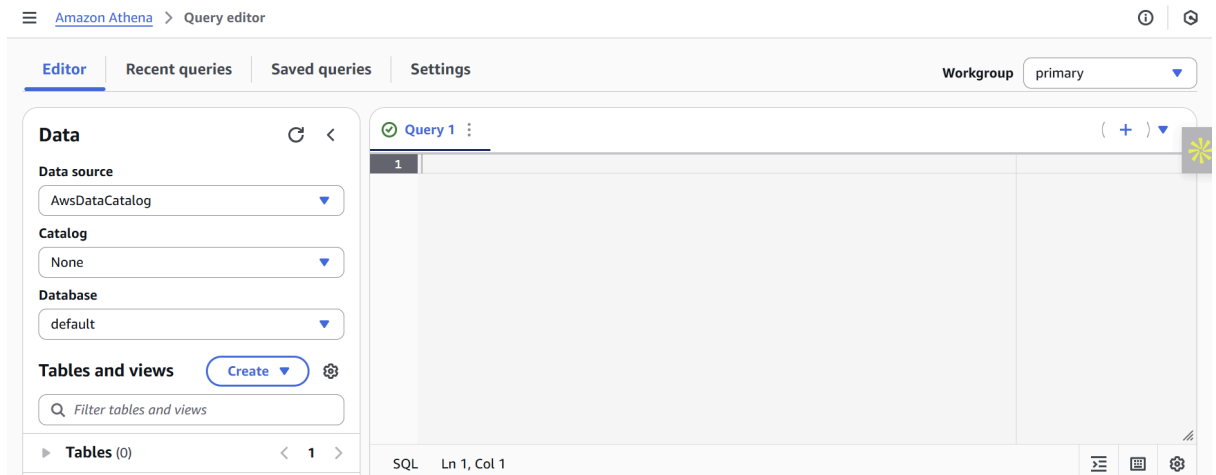
Step 2. Creating a destination bucket

1. Go to S3 and create a new bucket (eg: new-bucket-destination)
2. Click on this bucket and navigate to properties
3. Copy the ARN (Amazon Resource Name)

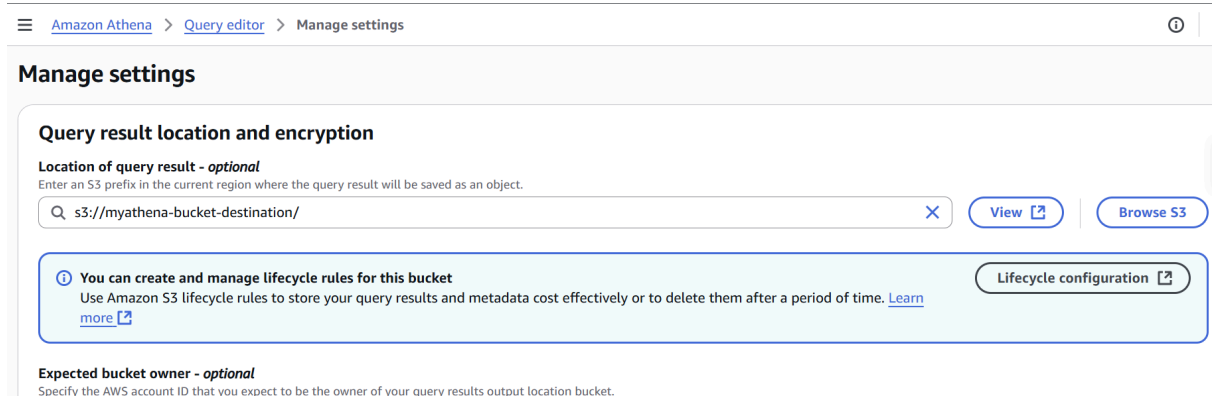


Step 3. Configuring Athena

1. Go to Athena and click on “[Query your data with Trino SQL](#)”, Use Query editor to analyze data on S3, and click on launch query editor.



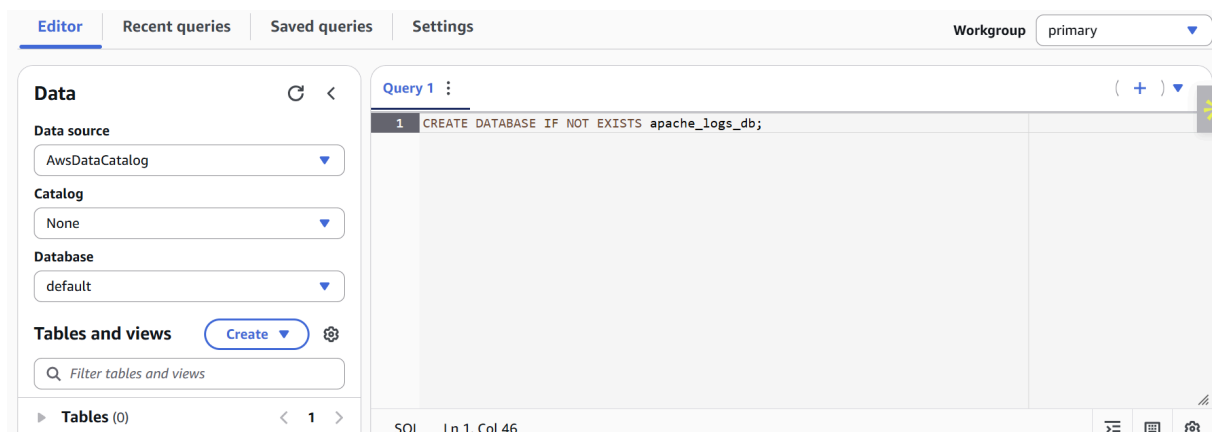
2. Select settings and click on manage.



3. In "Location of query result" paste the ARN of the destination bucket and save.

Step 4. Creating a database

1. Go to the Query editor in Amazon Athena.
2. And run the query for creating the database.
Eg: **CREATE DATABASE IF NOT EXISTS apache_logs_db;**



3. To use the db (By default, it will be selected)

Eg: **USE apache_logs_db**

Step 5. Creating a table

1. Run the Query for creating a table.

Eg: **CREATE EXTERNAL TABLE IF NOT EXISTS apache_access_logs (**
 host STRING,
 ident STRING,
 authuser STRING,
 datetime STRING,
 request STRING,
 status INT,
 bytes INT,
 referrer STRING,
 useragent STRING
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
 "input.regex" = "^(\S+) (\S+) (\S+) \[[^\]]+\] \"(.*)\" (\d{3}) (\d+|-) \"(.*)\"
\"(.*)\""
)
LOCATION 's3://myathena-bucket/'
TBLPROPERTIES ('has_encrypted_data'='false');

- **CREATE EXTERNAL TABLE:** Used to create a table
- (host STRING, ident STRING, authuser STRING, datetime STRING, request STRING, status INT, bytes INT, referrer STRING, useragent STRING): These lines define the schema of the table (i.e., the column names and data types). Here, I have used Webserver logs as an example, so these fields match the structure of a standard Apache web server log (combined log format).
- **ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe':** Specifies how to serialize/deserialize the text data into structured table rows.
- **WITH SERDEPROPERTIES ("input.regex" = "^(\S+) (\S+) (\S+) \[[^\]]+\] \"(.*)\" (\d{3}) (\d+|-) \"(.*)\" \"(.*)\"") :** Supplies configuration to the SerDe. In this case, a regular expression (input.regex) to match each log line.
- **LOCATION "s3://bucket-name/" :** ARN of source bucket
- **TBLPROPERTIES ('has_encrypted_data' = ' false') :** the data is not encrypted in S3.

Step 6. Running Queries

1. Create SQL queries and run them in the editor.

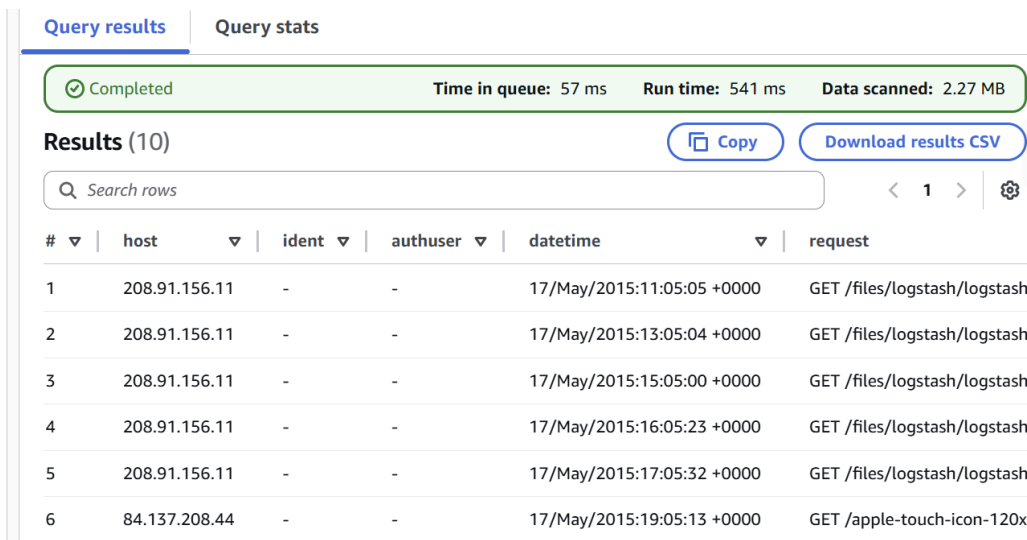
samples:

```
SELECT * FROM apache_access_logs where bytes=324 limit 10;
```

```
SELECT * FROM apache_access_logs where status=404;
```

```
SELECT * FROM apache_access_logs;
```

2. By scrolling down, you can see the query results and download CSV.



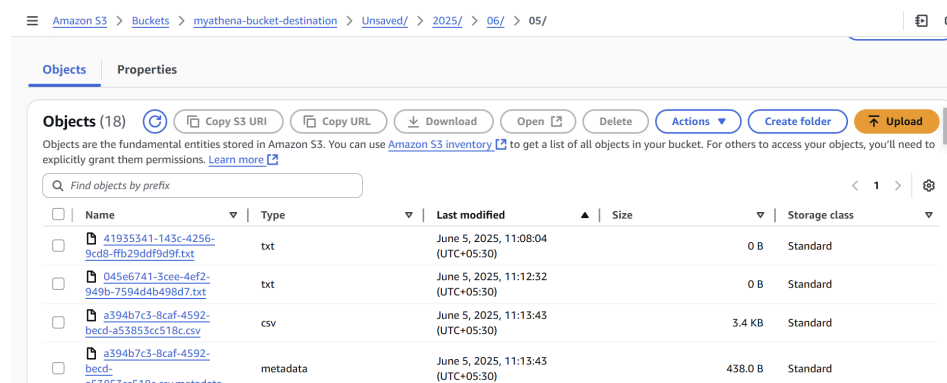
#	host	ident	authuser	datetime	request
1	208.91.156.11	-	-	17/May/2015:11:05:05 +0000	GET /files/logstash/logstash
2	208.91.156.11	-	-	17/May/2015:13:05:04 +0000	GET /files/logstash/logstash
3	208.91.156.11	-	-	17/May/2015:15:05:00 +0000	GET /files/logstash/logstash
4	208.91.156.11	-	-	17/May/2015:16:05:23 +0000	GET /files/logstash/logstash
5	208.91.156.11	-	-	17/May/2015:17:05:32 +0000	GET /files/logstash/logstash
6	84.137.208.44	-	-	17/May/2015:19:05:13 +0000	GET /apple-touch-icon-120x

Step 7. Checking the result in the S3 Destination bucket and downloading the output.

1. Go to S3 and select the destination bucket

2. From there, navigate to recently created folders (eg:if you execut query on 5/6/2025 folder structure would be /unsaved/2025/06/05/)

3. Filter last modified and select the file by clicking the Object URL, you can download the CSV file (output)



Name	Type	Last modified	Size	Storage class
41935341-143c-4256-9cd8-ffb29ddfd9f9.txt	txt	June 5, 2025, 11:08:04 (UTC+05:30)	0 B	Standard
045e6741-3cee-4ef2-949b-7594d4b498d7.txt	txt	June 5, 2025, 11:12:32 (UTC+05:30)	0 B	Standard
a394b7c3-8caf-4592-becd-a53853cc518c.csv	csv	June 5, 2025, 11:13:43 (UTC+05:30)	3.4 KB	Standard
a394b7c3-8caf-4592-becd-a53853cc518c.csv.metadata	metadata	June 5, 2025, 11:13:43 (UTC+05:30)	438.0 B	Standard

Common Errors :

1. Access Denied in S3: Check bucket policy and Block Public Access setting.