Athena Queries Guide

This document describes the Athena saved queries automatically provisioned by the CloudFormation stack. All queries are created in the stack's Athena workgroup and target the Glue databases/tables that back the Datalake S3 bucket.

Queries use partition projection for efficient scanning – the account, region, bucket, year, month, date partition keys are resolved at query time without requiring a Glue crawler.

1. Workgroup and Databases

Workgroup: <stackname>-workgroup
Query results are stored in s3://<datalake-bucket>/query-results/ with optional KMS encryption.

Databases (one per index type):

- <stackname>_app – application log events
- <stackname>_audit – audit log events

Each database contains a single table named logs with columns: @timestamp, accountid, loggroup, logstream, message, metadata (JSON string).

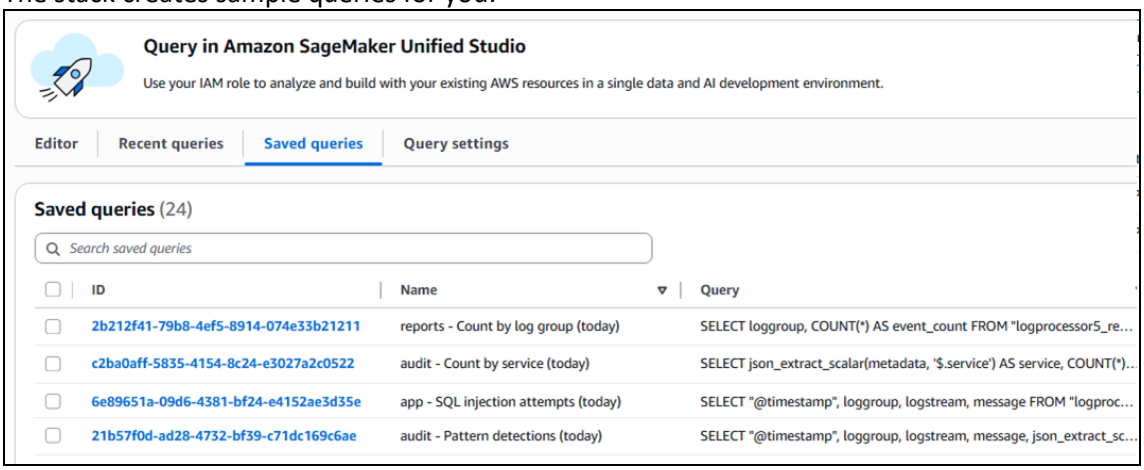
Access log database: <stackname>_access_logs
Contains a single table access_logs for S3 server access logs. The bucket partition is required in all queries.

2. Per-Index Saved Queries

The following queries are created for each index type (app and audit). Replace <type> with the index type in the query name.

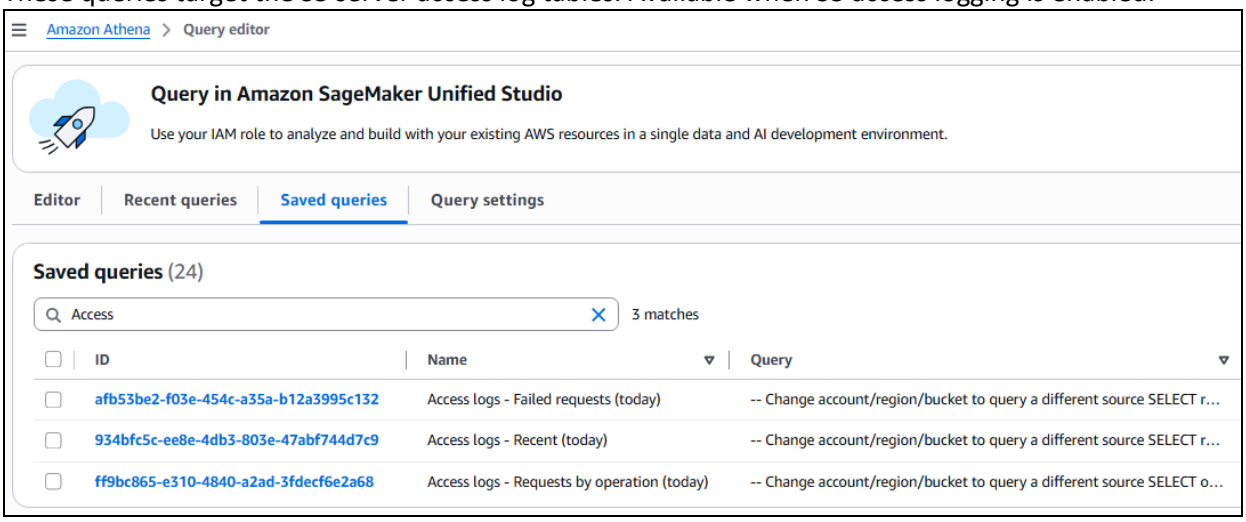
Note: For @timestamp use from_iso8601_timestamp() at query time. It's negligible cost and keeps the schema compatible with the raw data format, e.g.
SELECT from_iso8601_timestamp("@timestamp") AT TIME ZONE 'America/New_York' as timestamp …

The stack creates sample queries for you:



3. Access Log Saved Queries

These queries target the S3 server access log tables. Available when S3 access logging is enabled.



4. Running Saved Queries

1. Open the **Athena** console in your AWS region
2. Select the workgroup <stackname>-workgroup from the dropdown
3. Click the **Saved queries** tab
4. Select a query and click **Run**

5. Querying Pattern Detections

Pattern detection fields (pattern_detected, pattern_types) are stored inside the metadata JSON string column. Use json_extract_scalar to access them:

```
-- All pattern detections
json_extract_scalar(metadata, '$.pattern_detected') = 'true'

-- Specific pattern type
json_extract_scalar(metadata, '$.pattern_types') LIKE '%ssn%'

-- Custom metadata fields
json_extract_scalar(metadata, '$.service') = 'my-service'
```

6. Cost Considerations

Athena charges $5 per TB of data scanned. To minimize costs:

- Always include partition filters (account, year, month, day, hour) to limit scanned data.
- Use LIMIT to cap result rows.
- Select only the columns you need rather than SELECT *. Refer to the saved sample queries.
- The workgroup enforces result location and encryption – no additional configuration needed.

e.g.

```
SELECT requestdatetime, bucket_name, remoteip, requester, operation, key, httpstatus
FROM "logprocessor5_access_logs"."access_logs"
WHERE account = '000000000000'
  AND region = 'us-east-1'
  AND bucket = 'log-bucket-logprocessor5-000000000000-us-east-1'
  AND date = date_format(current_date, '%Y/%m/%d')
ORDER BY requestdatetime DESC
LIMIT 100
```

7. Notes

- Consider Athena be a primary target for logging where log volume is too high for OpenSearch.
- Saved queries are recreated on every stack update. Custom queries should be saved with different names to avoid being overwritten.
- The <stackname> and <accountid> placeholders in this document are replaced with your actual stack name and AWS account ID in the provisioned queries.
- Glue tables use partition projection – no crawler is needed. New partitions are available immediately as data arrives.
- The access_logs table uses bucket and date partitions. The bucket partition is required (injected type). Additional buckets can be added via provided (**.cmd/.sh**) script, e.g. add-access-logging.cmd add <bucket-name> (refer to readme.txt in the downloadable logprocessor-scripts.zip).
- Use the provided (**.cmd/.sh**) script, e.g. setup-replication.cmd to configures an external account's S3 access bucket to replicate S3 access logs to a stack's access log bucket.

Cross-Region S3 Access Logs

Available on essential tier and above. Replicate S3 access logs from buckets in other regions to the stack's access log bucket. Logs are replicated via S3 cross-region replication and queryable in Athena alongside your primary region's access logs.

Example steps leveraging provided (**.cmd/.sh**) scripts:

1. From the source region, run the one-time setup pointing to your stack's access log bucket:

   setup-replication.cmd setup logprocessor-access-logs-000000000000-us-east-1 us-west-2

2. Add each source bucket you want to monitor:

   setup-replication.cmd add my-app-bucket us-west-2

3. Verify replication is working:

   setup-replication.cmd verify my-app-bucket us-west-2

Here is a sample cross-account Athena query:

```
-- Cross-region:
SELECT requestdatetime, bucket_name, remoteip, requester, operation, key, httpstatus
FROM "logprocessor_access_logs"."access_logs"
WHERE account = '000000000000'
   AND region = 'us-west-2'
   AND bucket = 'my-app-bucket'
   AND date = date_format(current_date, '%Y/%m/%d')
ORDER BY requestdatetime DESC
```

Note: Logs appear in Athena under the same access_logs table with the source region and bucket as partition keys. First logs may take 15–60 minutes to appear.

Cross-Account S3 Access Log Replication

Available on enterprise tier leveraging provided (**.cmd/.sh**) scripts. Centralize S3 access logs from multiple AWS accounts into a single pipeline.

Step 1 – Run:
setup-replication.cmd setup <target-bucket> [--expire-days n] [--noncurrent-days 3]
e.g.

```
setup-replication.cmd setup logprocessor-access-logs-000000000000-us-east-1 us-east-1 --expire-days 7
--noncurrent-days 3
```

```
Account:    111111111111
Region:     us-east-1
Log bucket: access-logs-111111111111-us-east-1
Target:     logprocessor-access-logs-000000000000-us-east-1

[1/5] Creating shared log bucket access-logs-111111111111-us-east-1 ...
{
      "Location": "/access-logs-111111111111-us-east-1",
      "BucketArn": "arn:aws:s3:::access-logs-111111111111-us-east-1"
}
   Done.
[2/5] Enabling versioning on access-logs-111111111111-us-east-1 (required for replication) ...
   Done.
[3/5] Creating IAM replication role s3-access-log-replication-111111111111...
   Done.
   Waiting for IAM role propagation ...
[4/5] Configuring S3 replication to logprocessor-access-logs-000000000000-us-east-1 ...
   Done.
[5/5] Configuring lifecycle policy ...
{
      "TransitionDefaultMinimumObjectSize": "all_storage_classes_128K"
}
```

```
  Objects expire after 7 days.
  Noncurrent versions expire after 3 days.
  Done.

Setup complete. Shared log bucket: access-logs-111111111111-us-east-1
Now add source buckets with:
  add <source-bucket> [region]
```

Step 2 – Add source buckets using: setup-replication.cmd add <source-bucket>
e.g.

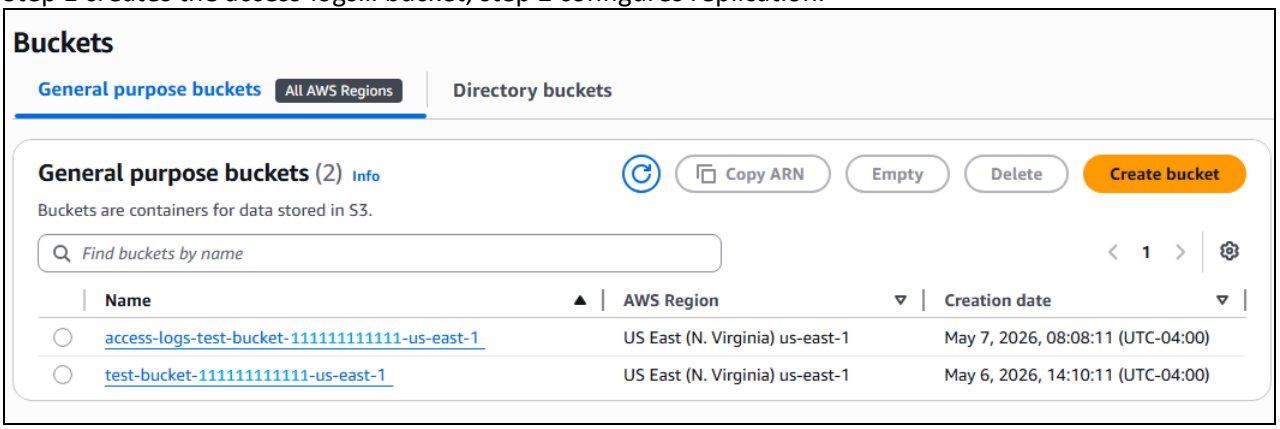
```
setup-replication.cmd add test-bucket-111111111111-us-east-1
Account:    111111111111
Region:     us-east-1
Log bucket: access-logs-111111111111-us-east-1
Source:     test-bucket-111111111111-us-east-1

[1/1] Enabling partitioned access logging on test-bucket-639381489303-us-east-1 ...
   Done.

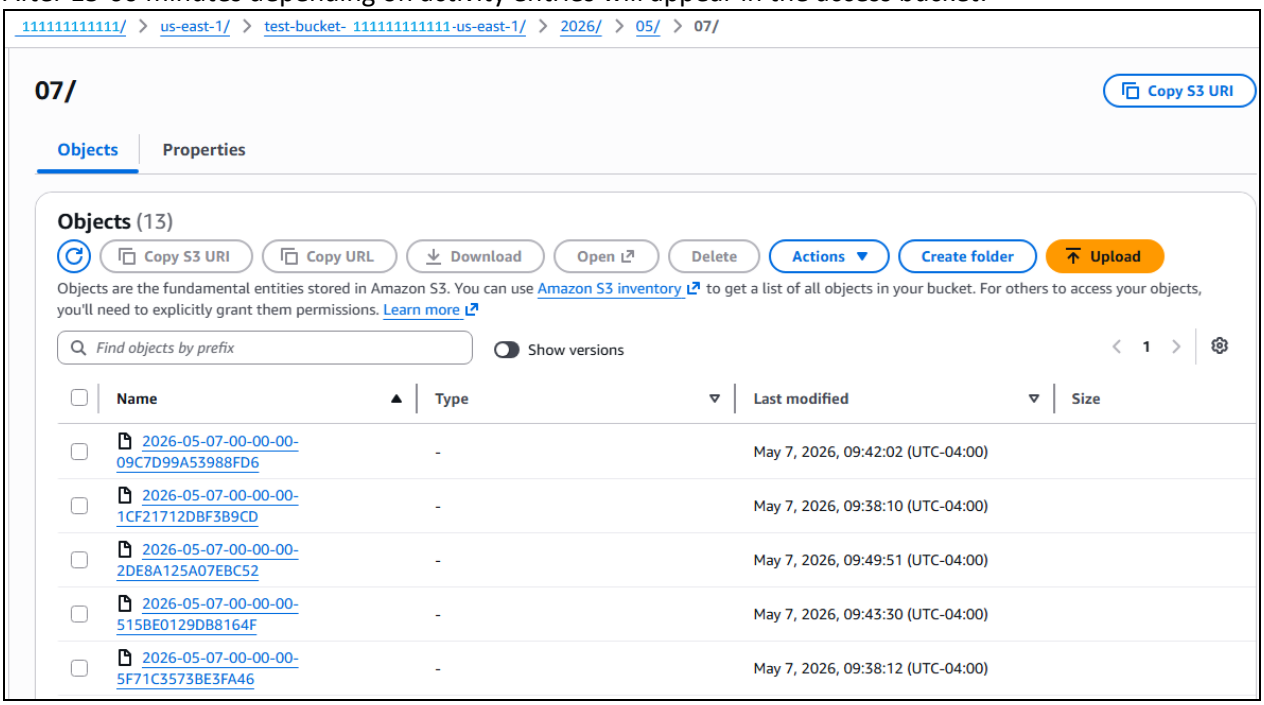
Access logs from test-bucket-111111111111-us-east-1 will appear at:
   s3://access-logs-000000000000-us-east-1/access/111111111111/us-east-1
\/test-bucket-111111111111-us-east-1/YYYY/MM/DD/

Logs replicate to the target bucket configured during setup.
First logs may take 15-60 minutes to appear.
```

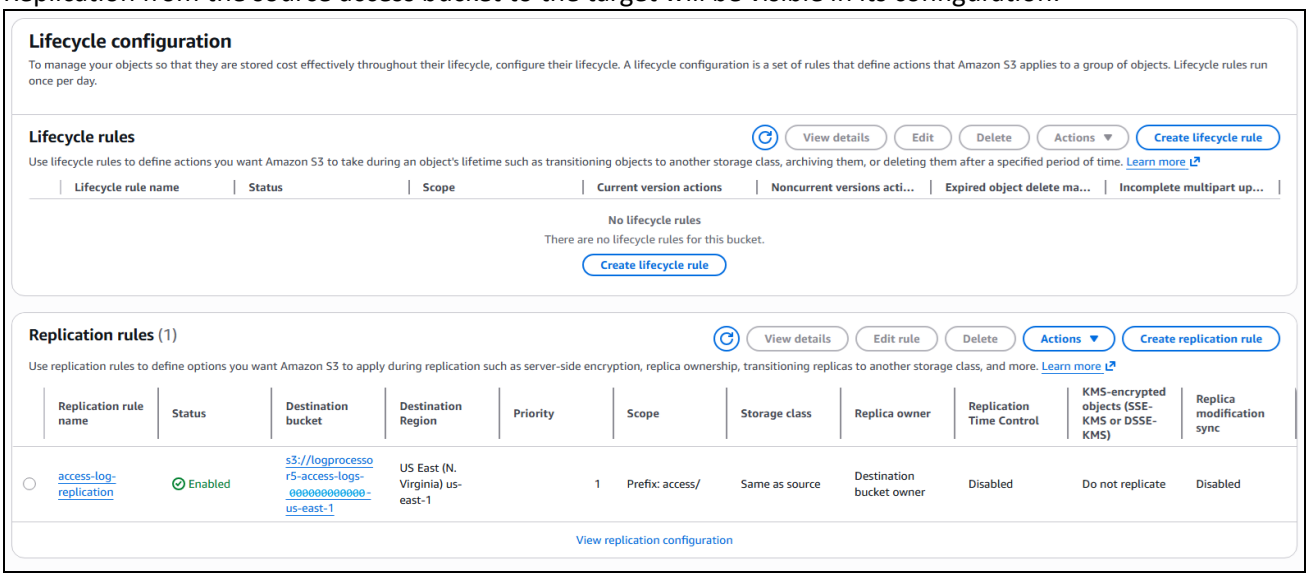
Step 1 creates the access-logs… bucket, step 2 configures replication.



After 15-60 minutes depending on activity entries will appear in the access bucket:



Replication from the source access bucket to the target will be visible in its configuration:



To list buckets hosted in access log S3 bucket use: add-access-logging.cmd <stack> list

add-access-logging.cmd LogProcessor list
Monitored buckets in logprocessor-access-logs-000000000000-us-east-1: test-bucket-111111111111-us-east-1 datalake-logprocessor-000000000000-us-east-1 log-bucket-logprocessor-000000000000-us-east-1
Or … aws s3 ls s3://logprocessor-access-logs-000000000000-us-east-1/access/ --region us-east-1

Here is a sample cross-account Athena query:

```
-- Cross-account:
SELECT requestdatetime, bucket_name, remoteip, requester, operation, key, httpstatus
FROM "logprocessor_access_logs"."access_logs"
WHERE account = '111111111111'
    AND region = 'us-east-1'
    AND bucket = 'test-bucket-111111111111-us-east-1'
    AND date = date_format(current_date, '%Y/%m/%d')
ORDER BY requestdatetime DESC
```

