CloudWatch Logs Transformation and Querying Workflow Report

Objective

The goal of this task was to build an automated workflow to extract, clean, and transform raw CloudWatch logs into a structured format that enables easy querying and analysis in Amazon Athena.

This involved:

1. Using AWS Glue (PySpark) for ETL (Extract, Transform, Load).
2. Storing cleaned logs in S3.
3. Querying structured logs with Amazon Athena.

Raw Input

The raw dataset was a CSV file exported from CloudWatch Logs into a S3 Bucket in this format:

timestamp,message

1757511342804,"INFO 2025-09-10T13:35:42,804 10153 org.apache.spark.metrics.source.StageSkewness [Thread-10] 29 [Observability] Skewness metric using Skewness Factor = 5"

The challenge with this raw data:

- The message column contained multiple attributes mixed together (timestamp, log level, class name, thread, etc.).
- The structure was not suitable for querying (e.g., filtering by ERROR, aggregating by date).

Feature Engineering & Transformations

To make the logs query able, we applied **regex-based parsing** inside an AWS Glue PySpark job.

Steps Taken

1. **Regex Extraction**
   Extracted key components from the message field:
   - **timestamp_raw** → extracted event timestamp string.
   - **log_level** → captured INFO, ERROR, DEBUG, etc.
   - **thread** → extracted thread name (e.g., Thread-10).

  - **class** → Java/Python class that logged the event.
  - **line** → line number in source file.
  - **message_clean** → actual log message text.

2. **Type Conversion**
  - Converted timestamp_raw → **timestamp** column (timestamp type).
  - Extracted **date** (date type) from timestamp for partitioning and easy filtering.

3. **Filtering Noise**
  - Removed rows where message_clean was empty or null (avoided blank entries).

Final Schema

After transformations, the structured dataset contained the following columns:

Column	Type	Description
timestamp	timestamp	Event timestamp with full precision.
log_level	string	Log level (INFO, ERROR, DEBUG, WARN, TRACE).
thread	string	Thread name that produced the log.
class	string	Class/package name of the log source.
line	int	Line number in the source file.
message_clean	string	Cleaned human-readable log message.
date	date	Event date (extracted from timestamp).

Workflow Overview

Extract

  - Source: CloudWatch logs exported to S3 (log-events-viewer-result.csv).
  - Format: CSV.

Transform (Glue Job / PySpark)

  - Parsed message using regex.
  - Added structured fields (log_level, class, etc.).
  - Converted timestamps into proper types.
  - Removed empty rows.

Load

  - Saved cleaned data as CSV in S3 under s3://cloudwatch-cleaned-logs-aip-71/cleaned-logs/.
  - Ensured single file output (part-0000*.csv).

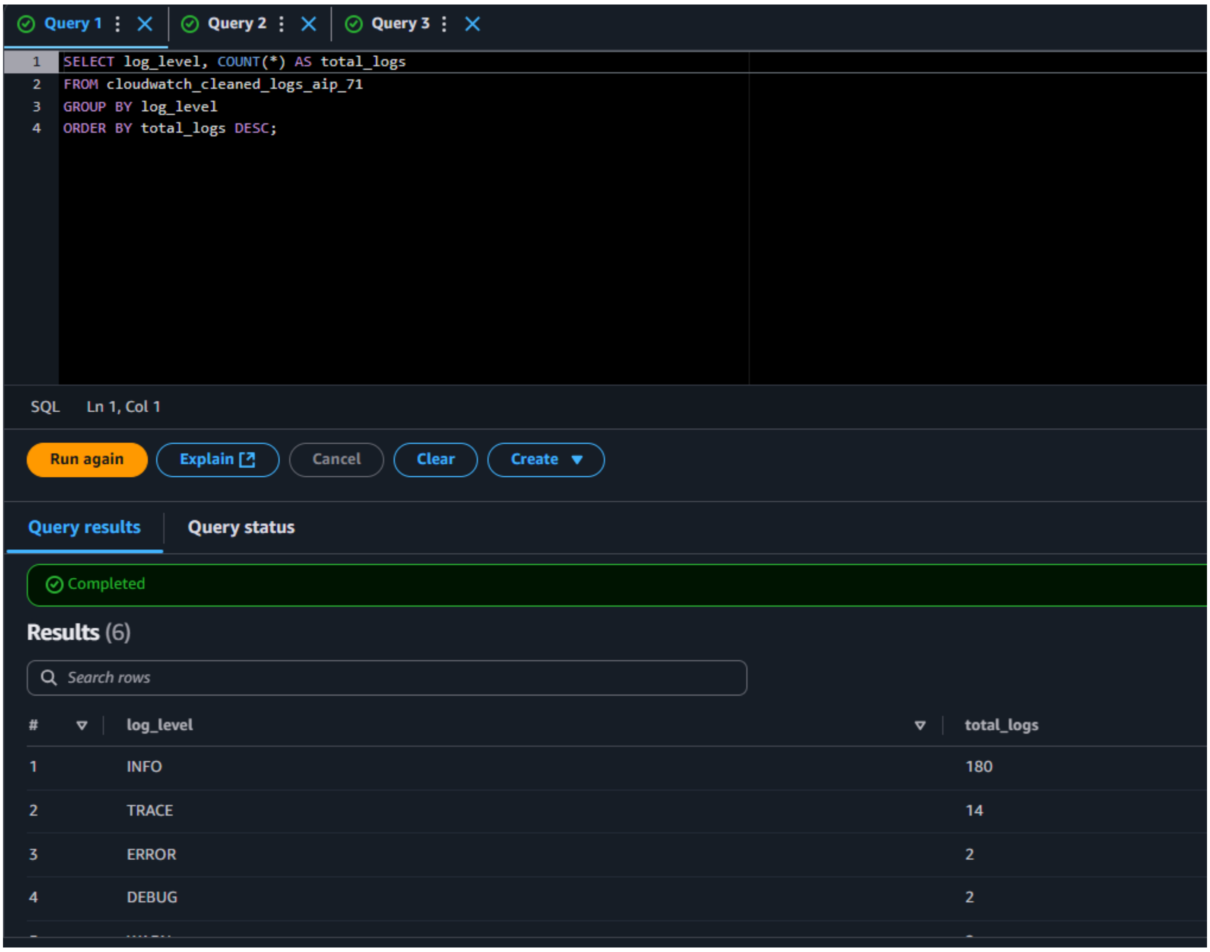
Query (Athena)

  - Created Athena external table over the cleaned logs.
  - Queried logs by date, log_level, and keywords.

Why These Features Were Created

- **timestamp** → Enables time-series queries (e.g., logs per hour/day).
- **date** → Optimized partitioning in Athena, allows fast date filtering.
- **log_level** → Separates ERROR/INFO logs for reliability monitoring.
- **thread** → Useful for debugging concurrency or thread-specific issues.
- **class** → Helps trace which component of the system generated logs.
- **line** → Assists in pinpointing exact code lines for debugging.
- **message_clean** → Human-readable message for deeper insights and keyword search.

Example Athena Queries



Query 1 Query 2 Query 3

```sql
SELECT log_level, COUNT(*) AS total_logs
FROM cloudwatch_cleaned_logs_aip_71
GROUP BY log_level
ORDER BY total_logs DESC;
```

SQL Ln 1, Col 1

Run again Explain Cancel Clear Create

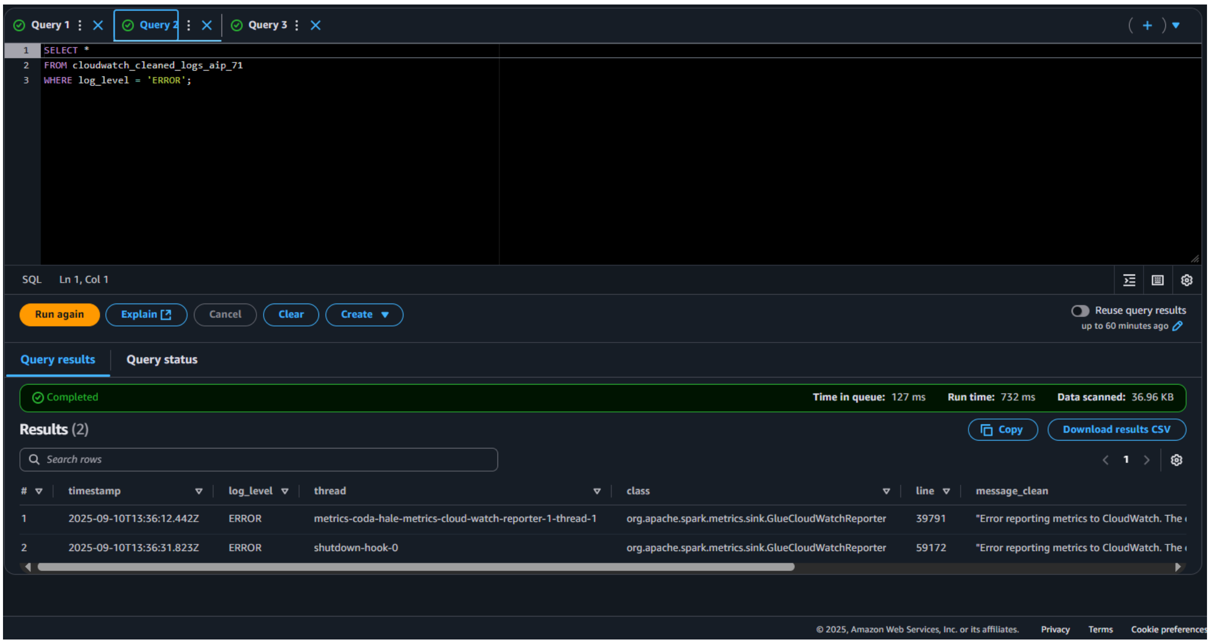
Query results Query status

Completed

Results (6)

Search rows

#	log_level	total_logs
1	INFO	180
2	TRACE	14
3	ERROR	2
4	DEBUG	2



Query 1 Query 2 Query 3

```sql
SELECT *
FROM cloudwatch_cleaned_logs_aip_71
WHERE log_level = 'ERROR';
```

SQL Ln 1, Col 1

Run again Explain Cancel Clear Create

Reuse query results
up to 60 minutes ago

Query results Query status

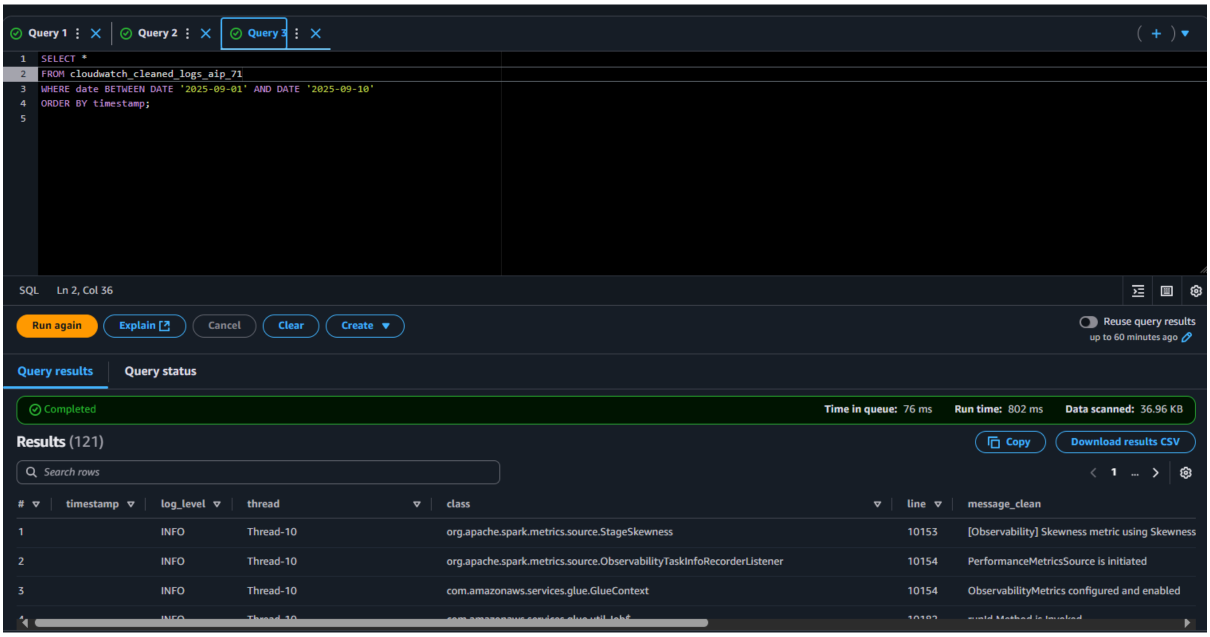
Completed Time in queue: 127 ms Run time: 732 ms Data scanned: 36.96 KB

Results (2)

Copy Download results CSV

Search rows

#	timestamp	log_level	thread	class	line	message_clean
1	2025-09-10T13:36:12.442Z	ERROR	metrics-coda-hale-metrics-cloud-watch-reporter-1-thread-1	org.apache.spark.metrics.sink.GlueCloudWatchReporter	39791	"Error reporting metrics to CloudWatch. The
2	2025-09-10T13:36:31.823Z	ERROR	shutdown-hook-0	org.apache.spark.metrics.sink.GlueCloudWatchReporter	59172	"Error reporting metrics to CloudWatch. The



Conclusion

By engineering new features from the unstructured message column, we converted raw CloudWatch logs into a structured dataset suitable for analytical querying.
This workflow:

- Improves log observability.
- Enables faster debugging and monitoring via Athena.
- Provides flexibility for future extensions (e.g., partitioning by log_level or date for performance).