

[TYPE THE COMPANY NAME]

Reddit Comments ETL Pipeline

Using AWS and Airflow

Dhanyamol Devassy

Table of Contents

1. Introduction	3
2. Architecture Diagram	4
3. Data Source: Reddit API	4
3.1 Why Reddit API?	4
3.2 API Authentication.....	4
3.3 Helper Script (reddit_api.py)	5
4. Storage: Amazon S3 (Data Lake)	6
4.1 Bronze Layer	7
4.2 Silver Layer	7
4.3 Gold Layer	8
5. AWS Glue Resources.....	9
5.1 IAM Role for Glue	9
5.2 Create a Glue Database.....	9
5.3 Create Glue Crawlers	10
5.4 Create Glue Jobs	12
6. Airflow Orchestration	19
7. Athena Validation Queries (Samples).....	20
8. QuickSight Visuals	21
8.1 Top Subreddits by Comment Count (Bar Chart).....	21
8.2 Average Sentiment by Subreddit (Colored Bar Chart)	22
8.3 Trend of Comments Over Time (Line Chart)	23
8.4 Sentiment Distribution (Pie Chart)	24
8.5 Heatmap (Subreddit × Day).....	25
8.6 Scatter Plot (Sentiment vs Comments)	26
9. Troubleshooting steps	27
10. Cost Optimization Guide.....	29
11. Git Repo	30
12. Conclusion.....	30

1. Introduction

In the era of big data, Reddit represents a valuable platform for analyzing public conversations. With millions of daily active users, Reddit offers real-time insights into community opinions, discussions, and sentiment across domains like Python, Machine Learning, AWS, and Data Science.

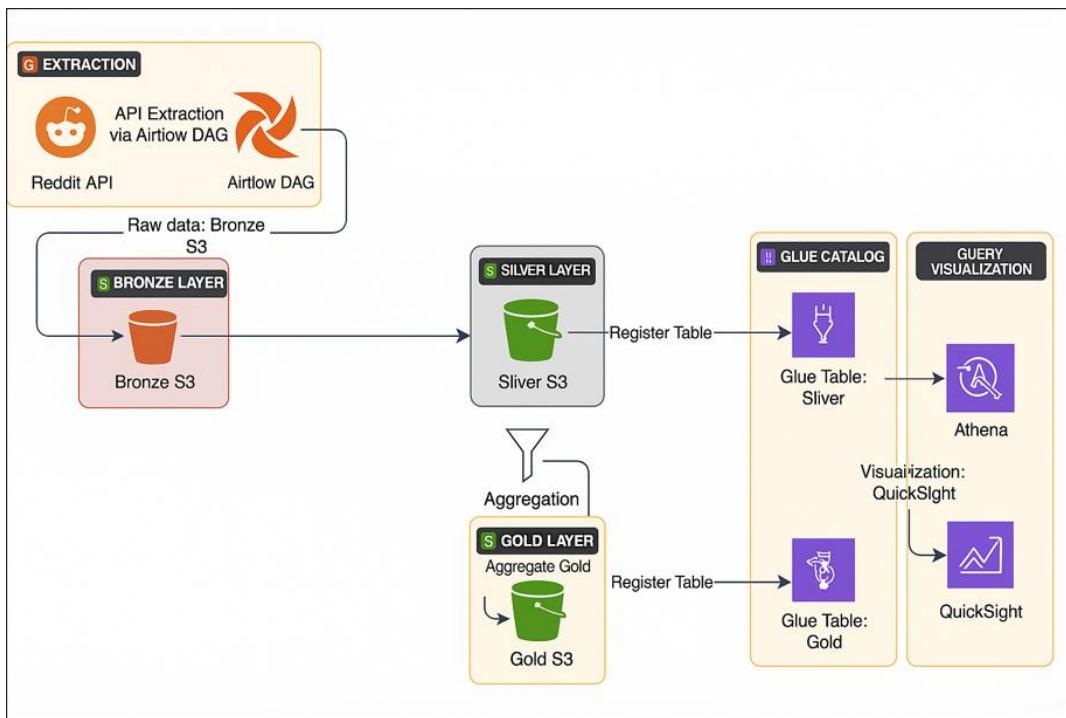
This project implements a cloud-native data pipeline for Reddit comments using AWS services (S3, Glue, Athena, QuickSight) orchestrated with Apache Airflow. It follows the medallion architecture (Bronze → Silver → Gold) for structured ETL and produces business-ready dashboards in QuickSight.

The pipeline automatically:

1. Extracts Reddit comments using the Reddit API.
2. Stores raw data in Amazon S3 (Bronze).
3. Cleans & normalizes data via Glue ETL jobs (Silver).
4. Aggregates & enriches subreddit metrics (Gold).
5. Validates & queries datasets using Athena.
6. Visualizes insights with QuickSight dashboards.

Reddit Comments AWS Data Pipeline

2. Architecture Diagram



3. Data Source: Reddit API

3.1 Why Reddit API?

Reddit provides an official API for fetching posts, comments, votes, and metadata from subreddits. For this project, the comment stream was chosen since it represents direct user interaction and sentiment.

3.2 API Authentication

The pipeline uses OAuth2-based credentials stored in Airflow .env:

```
airflow.env X
config > airflow.env
1  REDDIT_CLIENT_ID=[REDACTED]
2  REDDIT_CLIENT_SECRET=[REDACTED]
3  REDDIT_USER_AGENT=DataPipeline
```

Reddit Comments AWS Data Pipeline

3.3 Helper Script (reddit_api.py)

The script fetches comments from multiple subreddits and saves them to CSV for downstream ingestion.

```
reddit_api.py 1 ×
utils > reddit_api.py > fetch_reddit_comments
1 import os
2 import praw
3 import pandas as pd
4
5 def fetch_reddit_comments(output_path="data/raw/reddit_raw.csv"):
6     reddit = praw.Reddit(
7         client_id=os.getenv("CLIENT_ID"),
8         client_secret=os.getenv("CLIENT_SECRET"),
9         user_agent=os.getenv("USER_AGENT")
10    )
11
12     subreddits = ["datascience", "python", "aws", "machinelearning"]
13     comments = []
14
15     for sub in subreddits:
16         for comment in reddit.subreddit(sub).comments(limit=500):
17             comments.append({
18                 "id": comment.id,
19                 "author": str(comment.author),
20                 " subreddit": comment.subreddit.display_name,
21                 "body": comment.body,
22                 "created_utc": comment.created_utc
23             })
24
25     df = pd.DataFrame(comments)
26     os.makedirs(os.path.dirname(output_path), exist_ok=True)
27     df.to_csv(output_path, index=False)
28     print(f"Saved {len(df)} comments to {output_path}")
29
```

3.4 Data Collected

- Subreddit name
- Comment ID
- Comment text
- Timestamp (created_utc)
- Derived sentiment score

The Python script reddit_api.py fetches data and stores daily dumps into data/raw/reddit_raw.csv.

Output: data/raw/reddit_raw.csv (daily).

Reddit Comments AWS Data Pipeline

4. Storage: Amazon S3 (Data Lake)

The project follows partitioned storage for scalability.

```
C:\Users\Owner>aws s3 mb s3://reddit-data-bucket-dhanya --region us-east-1
make_bucket: reddit-data-bucket-dhanya

C:\Users\Owner>
```

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with 'Amazon S3' and sections for 'General purpose buckets' (listing various AWS services like Directory buckets, Table buckets, Vector buckets, etc.) and 'Storage Lens'. The main area is titled 'General purpose buckets' and shows one bucket named 'reddit-data-bucket-dhanya' created on September 15, 2025. To the right, there are two informational boxes: 'Account snapshot' (updated daily) and 'External access summary - new' (updated daily).

The screenshot shows the contents of the 'reddit-data-bucket-dhanya' bucket. The 'Objects' tab is selected, displaying four folder objects: 'athena_results/' (Folder), 'bronze/' (Folder), 'gold/' (Folder), and 'silver/' (Folder). Each folder has a small icon, a name, a type, last modified date, size, and storage class (all listed as '-'). There are also buttons for Actions, Create folder, and Upload.

Reddit Comments AWS Data Pipeline

4.1 Bronze Layer

- Raw CSV dumps, stored daily.
- Path format:

s3://reddit-data-bucket-dhanya/reddit/bronze/reddit_raw.csv

Purpose: Preserve original Reddit API data for traceability

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with navigation links like 'Amazon S3', 'General purpose buckets', 'AWS Region', 'Last modified', 'Size', 'Type', and 'Key'. The main area displays the properties of the 'reddit_raw.csv' object. The 'Properties' tab is selected. Key details shown include:

- S3 URI:** s3://reddit-data-bucket-dhanya/bronze/reddit_raw.csv
- Amazon Resource Name (ARN):** arnaws:s3:::reddit-data-bucket-dhanya/bronze/reddit_raw.csv
- Entity tag (Etag):** 1e3a4b363f4e30297d146bf4a4f3cec0
- Object URL:** https://reddit-data-bucket-dhanya.s3.us-east-1.amazonaws.com/bronze/reddit_ra w.csv

4.2 Silver Layer

- ETL job cleans raw CSVs:
 - Extracts required fields (subreddit, comment, sentiment).
 - Removes malformed/empty records.
 - Normalizes timestamps into year, month, day.
- Stored in Parquet format for query efficiency.

Reddit Comments AWS Data Pipeline

Amazon S3

General purpose buckets

Objects (2)

Name	Type	Last modified	Size	Storage class
part-00000-7c908005-6309-41a0-bca8-8a72d86eb3fa.c000.snappy.parquet	parquet	September 16, 2025, 17:12:08 (UTC-04:00)	135.8 KB	Standard
part-00001-7c908005-6309-41a0-bca8-8a72d86eb3fa.c000.snappy.parquet	parquet	September 16, 2025, 17:12:07 (UTC-04:00)	75.8 KB	Standard

4.3 Gold Layer

- Aggregation layer:
 - Groups by subreddit.
 - Computes comment_count and avg_sentiment.
- Stored in Parquet, partitioned by year/month/day.
- Directly queried by Athena and used in QuickSight.

Amazon S3

General purpose buckets

Objects (1)

Name	Type	Last modified	Size	Storage class
part-00000-1dfe37c0-afcf-42c9-832b-bb5e6c0e511c.c000.snappy.parquet	parquet	September 16, 2025, 17:17:33 (UTC-04:00)	2.1 KB	Standard

5. AWS Glue Resources

5.1 IAM Role for Glue

- Role Name: reddit-role.
- Permissions:
 - AmazonS3FullAccess (to dataset bucket).
 - AWSGlueServiceRole managed policy.
 - AmazonAthenaFullAccess (optional for validation).

The screenshot shows the AWS IAM Roles page. The search bar at the top contains 'redd'. Below it, a table lists roles, with one row highlighted: 'reddit-role' under 'Role name'. To the right of the table, there are sections for 'Trusted entities' and 'Last activity'. At the bottom of the page, there are three cards: 'Access AWS from your non AWS workloads', 'X.509 Standard', and 'Temporary credentials'.

5.2 Create a Glue Database

1. Go to AWS Glue Console → Databases → Add Database.
2. Name: redditdb.
3. Used as metadata catalog for Athena & QuickSight.

Reddit Comments AWS Data Pipeline

The screenshot shows the AWS Glue Data Catalog Databases page. On the left, there's a navigation sidebar with sections for AWS Glue (Getting started, ETL jobs, etc.) and Data Catalog (Databases, Tables, Schemas, etc.). The main content area is titled "Databases (1)". It displays a table with one row for "redditdb". The table columns are Name, Description, Location URI, and Created on (UTC). The "Name" column shows "redditdb", the "Created on (UTC)" column shows "September 16, 2025 at 01:01:12", and the "Last updated (UTC)" column shows "September 16, 2025 at 17:22:13". There are buttons for Edit, Delete, and Add database.

5.3 Create Glue Crawlers

Crawler for Bronze Layer

- Name: reddit-bronze-crawler.
- Data store: s3://reddit-data-dhanya/reddit/bronze/.
- IAM Role: reddit-role.
- Target Database: redditdb.
- Schedule: On-demand (triggered via Airflow).

The screenshot shows the AWS Glue Crawlers page. The left sidebar includes sections for AWS Glue (Getting started, ETL jobs, etc.) and Data Catalog (Databases, Tables, Schemas, etc.). A green notification bar at the top says "One crawler successfully created" and "The following crawler is now created: *reddit_bronze_crawler*". The main content area is titled "Crawlers". It contains a table with one row for "reddit_bronze_cra...". The table columns are Name, State, Schedule, Last run, Last run times..., Log, and Table changes fr.... The "Name" column shows "reddit_bronze_cra...", the "State" column shows "Ready", and the "Last run" column shows "-". There are buttons for Action, Run, and Create crawler.

Ran the crawler

Reddit Comments AWS Data Pipeline

The screenshot shows the AWS Glue Crawler interface. A green success message at the top states: "Crawler successfully starting. The following crawler is now starting: 'reddit_bronze_crawler'". Below this, the "Crawlers" section lists one crawler named "reddit_bronze_cra..." which is "Ready" and has "Succeeded" with a timestamp of "September 16, 2025 at 17:27:34". There are buttons for "Action", "Run", and "Create crawler". The left sidebar includes sections for AWS Glue (Getting started, ETL jobs, Data Catalog tables, Data connections, Workflows (orchestration), Zero-ETL integrations), Data Catalog (Databases, Tables, Stream schema registries, Schemas, Connections, Crawlers, Classifiers, Catalog settings), Data Integration and ETL, and Legacy pages.

Table is created in redditdb

The screenshot shows the AWS Glue Table interface. A blue info bar at the top says: "Announcing new optimization features for Apache Iceberg tables. Optimize storage for Apache Iceberg tables with automatic snapshot retention and orphan file deletion. Learn more". Below this, the "Tables" section lists one table named "bronze" located in "redditdb" with "s3://reddit-data-bu" as the location and "CSV" as the classification. There are buttons for "Delete", "Add tables using crawler", and "Add table". The left sidebar includes sections for AWS Glue (Getting started, ETL jobs, Data Catalog tables, Data connections, Workflows (orchestration), Zero-ETL integrations), Data Catalog (Databases, Tables, Stream schema registries, Schemas, Connections, Crawlers, Classifiers, Catalog settings).

Reddit Comments AWS Data Pipeline

The screenshot shows the AWS Glue Data Catalog interface. On the left, there's a sidebar with navigation links like 'AWS Glue', 'Data Catalog tables', 'Data Catalog', 'Data Integration and ETL', and 'Legacy pages'. The main area displays the schema for a table named 'bronze'. The schema table has columns for '#', 'Column name', 'Data type', 'Partition key', and 'Comment'. The data shown is:

#	Column name	Data type	Partition key	Comment
1	id	string	-	-
2	author	string	-	-
3	subreddit	string	-	-
4	body	string	-	-
5	created_utc	double	-	-

Test in Athena

The screenshot shows the Amazon Athena query editor. On the left, there's a sidebar with 'Data source' set to 'AwsDataCatalog', 'Catalogue' set to 'None', and 'Database' set to 'redditdb'. Below that, under 'Tables and views', there's a table named 'bronze' with columns 'id', 'author', 'subreddit', and 'body'. The main area shows a query editor tab titled 'Query 1' with the following SQL:

```
1 SELECT * FROM redditdb.bronze LIMIT 2;
```

The results section shows two rows of data:

#	id	author	subreddit	body
1	nega8or	chocolateandcoffee	datascience	"Have you spoken with anyone at the company about your plan? If someone showed up to an interview....."
2	neg93l0	Alpha-Centauri-C	datascience	Wow. The statistical awareness of the majority of people who use the term "A/B test" is abysmal.....

5.4 Create Glue Jobs

5.4.1 Silver Job (silver-job)

- Input: Bronze CSV.
- Transformations:
 - Select key columns (subreddit, text, sentiment, timestamp).

Reddit Comments AWS Data Pipeline

- Add partitions: year, month, day.
- Save as Parquet.

Script:

```
glue_silver_job.py 1 X
etls > glue_silver_job.py > ...
1   from awsglue.context import GlueContext
2   from pyspark.context import SparkContext
3   from pyspark.sql.functions import col, from_unixtime, year, month, dayofmonth
4
5   sc = SparkContext()
6   glueContext = GlueContext(sc)
7   spark = glueContext.spark_session
8
9   # Read Bronze (CSV)
10  df = spark.read.option("header", True).csv("s3://reddit-data-bucket-dhanya/bronze/")
11
12  # Clean + add partition columns
13  df_clean = (
14      df.filter(col("body").isNotNull())
15      .withColumn("created_ts", from_unixtime(col("created_utc")))
16      .withColumn("year", year(from_unixtime("created_utc")))
17      .withColumn("month", month(from_unixtime("created_utc")))
18      .withColumn("day", dayofmonth(from_unixtime("created_utc")))
19  )
20
21  # Write to Silver as partitioned Parquet
22  df_clean.write.mode("overwrite").partitionBy("year", "month", "day") \
23      .parquet("s3://reddit-data-bucket-dhanya/silver/")


```

Run the job

Reddit Comments AWS Data Pipeline

The screenshot shows the AWS Glue Studio interface for a job named "silver_job". The "Runs" tab is selected. A single run is listed under "Job runs (1/1) info". The run status is "Succeeded" with 0 retries, starting at 09/16/2025 13:33:38 and ending at 09/16/2025 13:35:05, taking 1 m 20 s. It used 2 DPUs and is a G.1X worker type, running on Glue version 4.0. The table view shows the same information.

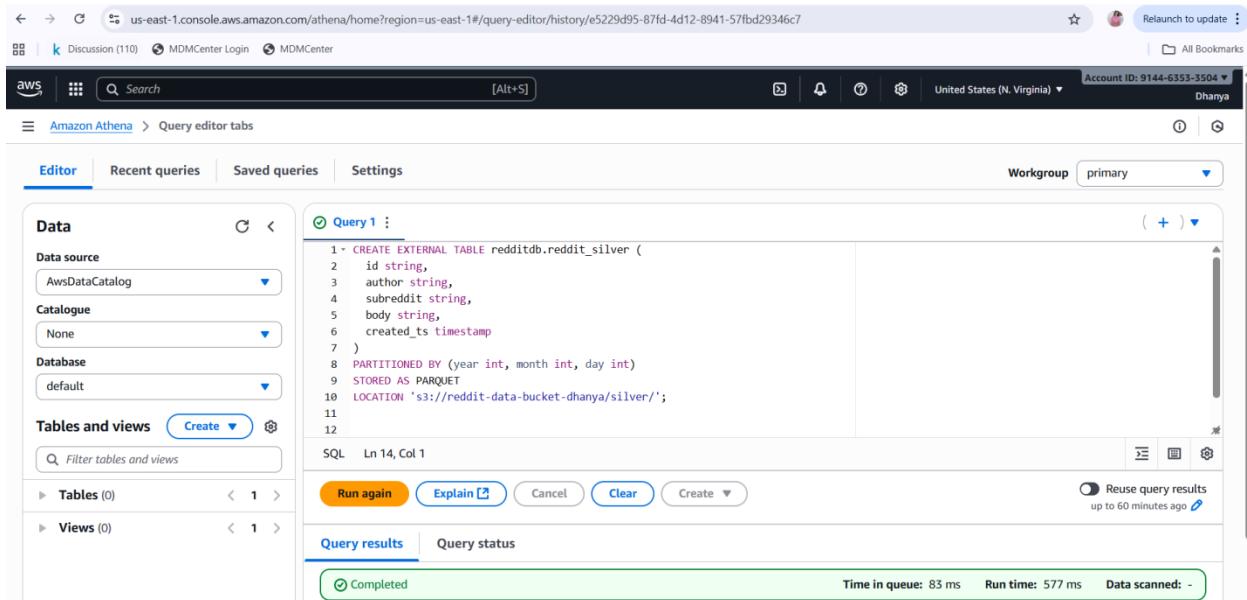
Output is generated in s3 with partitions

The screenshot shows the Amazon S3 console. The path is "Amazon S3 > Buckets > reddit-data-bucket-dhanya > silver > year=2025/ > month=9/ > day=16/". The "Objects" tab is selected, showing one object: "part-00000-a1306e2d-ac23-43ab-b1d2-f2c504cbcd8.c000.snappy.parquet". The object is a parquet file, 5.9 KB in size, last modified on September 16, 2025, at 13:34:47 (UTC-04:00). The storage class is Standard.

Validate in Athena

Create Silver Table

Reddit Comments AWS Data Pipeline

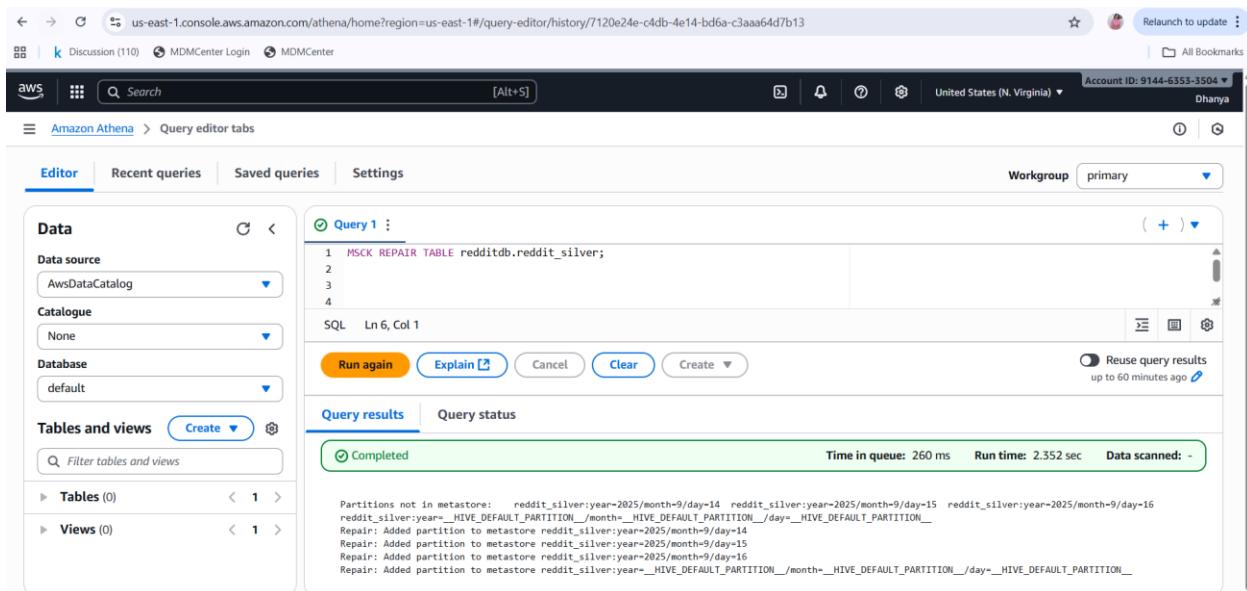


The screenshot shows the Amazon Athena Query Editor interface. On the left, the 'Data' sidebar is open, showing the configuration for a new table. The 'Data source' is set to 'AwsDataCatalog', 'Catalogue' to 'None', and 'Database' to 'default'. The 'Tables and views' section shows 0 tables and 0 views. In the main editor area, a query named 'Query 1' is being typed:

```
1 CREATE EXTERNAL TABLE redditdb.reddit_silver (
2   id string,
3   author string,
4   subreddit string,
5   body string,
6   created_ts timestamp
7 )
8 PARTITIONED BY (year int, month int, day int)
9 STORED AS PARQUET
10 LOCATION 's3://reddit-data-bucket-dhanya/silver/';
11
12
```

Below the editor, the 'Query results' tab is selected, showing a green status bar indicating the query is 'Completed'. The status bar also displays 'Time in queue: 83 ms', 'Run time: 577 ms', and 'Data scanned: -'.

Refresh partitions



The screenshot shows the Amazon Athena Query Editor interface. The 'Data' sidebar is open, showing the configuration for a new table. The 'Tables and views' section shows 0 tables and 0 views. In the main editor area, a query named 'Query 1' is being typed:

```
1 MSCK REPAIR TABLE redditdb.reddit_silver;
```

Below the editor, the 'Query results' tab is selected, showing a green status bar indicating the query is 'Completed'. The status bar displays 'Time in queue: 260 ms', 'Run time: 2.352 sec', and 'Data scanned: -'. The output of the repair command is shown in the results pane:

```
Partitions not in metastore: reddit_silver:year=2025/month=9/day=14 reddit_silver:year=2025/month=9/day=15 reddit_silver:year=2025/month=9/day=16
reddit_silver:year=__HIVE_DEFAULT_PARTITION__/month=__HIVE_DEFAULT_PARTITION__/day=__HIVE_DEFAULT_PARTITION__
Repair: Added partition to metastore reddit_silver:year=2025/month=9/day=14
Repair: Added partition to metastore reddit_silver:year=2025/month=9/day=15
Repair: Added partition to metastore reddit_silver:year=2025/month=9/day=16
Repair: Added partition to metastore reddit_silver:year=__HIVE_DEFAULT_PARTITION__/month=__HIVE_DEFAULT_PARTITION__/day=__HIVE_DEFAULT_PARTITION__
```

Test query

Reddit Comments AWS Data Pipeline

The screenshot shows the Amazon Athena Query Editor interface. On the left, there's a sidebar titled 'Data' with dropdowns for 'Data source' (AwsDataCatalog), 'Catalogue' (None), and 'Database' (default). Below it are sections for 'Tables and views' and a search bar. The main area has tabs for 'Editor' (selected), 'Recent queries', 'Saved queries', and 'Settings'. A 'Workgroup' dropdown is set to 'primary'. In the 'Editor' tab, a query named 'Query 1' is displayed:

```
1 SELECT subreddit, COUNT(*)  
2 FROM redditdb.reddit_silver  
3 WHERE year=2025 AND month=9  
4 GROUP BY subreddit  
5 ORDER BY COUNT(*) DESC;
```

The status bar at the bottom of the editor shows 'SQL Ln 1, Col 1'. Below the editor are buttons for 'Run again', 'Explain', 'Cancel', 'Clear', and 'Create'. To the right, a note says 'Reuse query results up to 60 minutes ago'. The 'Query results' tab is selected, showing a green bar indicating 'Completed' with 'Time in queue: 70 ms', 'Run time: 874 ms', and 'Data scanned: 0.21 KB'. It also shows 'Results (1)'. The results table has columns '#', 'subreddit', and '_col1'. One row is shown: '1' under '#', 'datascience' under 'subreddit', and '97' under '_col1'. Buttons for 'Copy' and 'Download results CSV' are available.

5.4.2 Gold Job (gold-job)

- Input: Silver dataset.
- Transformations:
 - Group by subreddit.
 - Calculate comment_count and avg_sentiment.

Script:

Reddit Comments AWS Data Pipeline

```
glue_gold.job.py 1 X
etls > glue_gold.job.py > ...
1  from awsglue.context import GlueContext
2  from pyspark.context import SparkContext
3  from pyspark.sql.functions import col, count, avg, udf
4  from pyspark.sql.types import DoubleType
5  sc = SparkContext()
6  glueContext = GlueContext(sc)
7  spark = glueContext.spark_session
8  # Small lexicon
9  positive_words = {"good", "great", "excellent", "happy", "love", "like", "awesome"}
10 negative_words = {"bad", "terrible", "sad", "hate", "angry", "worst", "awful"}
11 def sentiment_score(text):
12     if not text:
13         return 0.0
14     words = text.lower().split()
15     score = 0
16     for w in words:
17         if w in positive_words:
18             score += 1
19         elif w in negative_words:
20             score -= 1
21     return float(score)
22 sentiment_udf = udf(sentiment_score, DoubleType())
23 # Read Silver
24 df = spark.read.parquet("s3://reddit-data-bucket-dhanya/silver/")
25 # Add sentiment
26 df_sent = df.withColumn("sentiment", sentiment_udf(col("body")))
27 # Aggregate into Gold
28 df_gold = (
29     df_sent.groupBy(" subreddit", "year", "month", "day")
30     .agg(
31         count("id").alias("comment_count"),
32         avg("sentiment").alias("avg_sentiment")
33     )
34 )
35 df_gold.write.mode("overwrite").partitionBy("year", "month", "day") \
36     .parquet("s3://reddit-data-bucket-dhanya/gold/")
37
```

Run the job

The screenshot shows the AWS Glue Studio Job Editor interface. The top navigation bar includes links for Discussion (110), MDMCenter Login, and MDMCenter, along with account information (Account ID: 9144-6337-36) and a Relaunch to up button. The main area displays the 'gold_job' configuration. The 'Runs' tab is selected, showing a table of job runs. One run is listed under 'Job runs (1/4) Info':

Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity (DPU)	Worker type	Glue version
Succeeded	0	09/16/2025 13:54:34	09/16/2025 13:56:05	1 m 16 s	2 DPU	G.1X	4.0

Below the table, the 'Run details' section provides detailed metrics for the successful run:

Job name	Start time (Local)	Glue version
gold_job	09/16/2025 13:54:34	4.0
id	End time (Local)	Worker type
jr_f92dde89c9802bdad7b4466e368e300196c102574a	09/16/2025 13:56:05	G.1X
6cf09e44955d980991d13b2	Start-up time	Max capacity
Run status	14 seconds	2 DPU
Succeeded	Execution time	Execution class
Retrv attempt number		Number of workers
		2
		Timeout

Reddit Comments AWS Data Pipeline

Output is generated in s3

The screenshot shows the AWS S3 console interface. The left sidebar lists buckets: General purpose buckets (Amazon S3), Directory buckets, Table buckets, Vector buckets, Access Grants, Access Points (General Purpose Buckets, Fsx file system), Access Points (Directory Buckets), Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and Block Public Access settings for this account. The main area shows a list of objects in the 'day=16/' folder. There are two items:

- part-00000_a57591d4c4-refs-00000.parquet (September 16, 2025, 1.0 KB, Standard storage class)
- reddit_gold_20250916.parquet (September 16, 2025, 1.55 GB, Standard storage class)

Validate in Athena

Create gold table

The screenshot shows the AWS Athena console. The left sidebar includes Editor, Recent queries, Saved queries, and Settings. The main area displays a query editor window titled 'Query 1'. The SQL code creates an external table:

```
CREATE EXTERNAL TABLE redditdb.reddit_gold (
    subreddit string,
    comment_count bigint,
    avg_sentiment double
)
PARTITIONED BY (year int, month int, day int)
STORED AS PARQUET
LOCATION 's3://reddit-data-bucket-dhanya/gold/'$
```

The status bar at the bottom indicates the query was completed successfully.

Refresh partitions

The screenshot shows the AWS Athena console. The left sidebar includes Editor, Recent queries, Saved queries, and Settings. The main area displays a query editor window titled 'Query 1'. The SQL code runs an MSCK REPAIR TABLE command:

```
MSCK REPAIR TABLE redditdb.reddit_gold;
```

The status bar at the bottom indicates the repair process completed successfully.

Reddit Comments AWS Data Pipeline

Test query

The screenshot shows the Amazon Athena Query Editor interface. On the left, there's a sidebar with 'Data' settings (Data source: AwsDataCatalog, Catalogue: None, Database: default), 'Tables and views' (Tables: 0, Views: 0), and a search bar. The main area has tabs for 'Editor' (selected), 'Recent queries', 'Saved queries', and 'Settings'. Under 'Editor', a query named 'Query 1' is displayed:

```
1 SELECT subreddit, comment_count, avg_sentiment
2 FROM redditdb.reddit_gold
3 WHERE year=2025 AND month=9
4 ORDER BY comment_count DESC;
```

Below the query are buttons for 'Run again', 'Explain', 'Cancel', 'Clear', and 'Create'. To the right, it says 'Reuse query results up to 60 minutes ago'. The 'Query results' tab is selected, showing a green status bar with 'Completed' and metrics: Time in queue: 111 ms, Run time: 858 ms, Data scanned: 0.35 KB. The 'Results (3)' section shows a table with one row:

#	subreddit	comment_count	avg_sentiment
1	datascience	82	0.43902439024390244

Buttons for 'Copy' and 'Download results CSV' are available at the bottom of the results table.

6. Airflow Orchestration

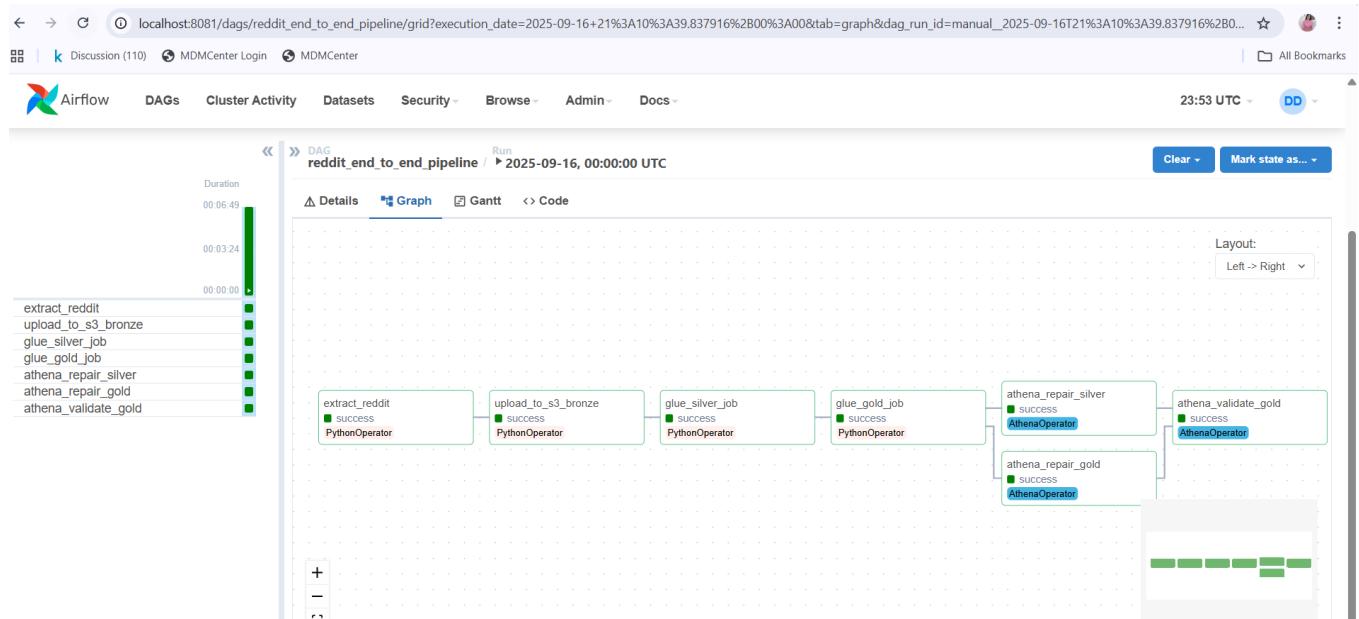
The DAG in Airflow (reddit_dag.py) runs in sequence:

1. extract_reddit → Pulls data.
2. upload_to_s3 → Push Bronze to S3.
3. trigger_glue_silver → Cleans data.
4. trigger_glue_gold → Aggregates metrics.
5. athena_repair → Updates partitions.
6. validate_data → Ensures non-empty gold tables.

Improvements made:

- .env variables for flexibility.
- Boto3 start_job_run for Glue reliability.
- Partition repair with Athena.

Reddit Comments AWS Data Pipeline



7. Athena Validation Queries (Samples)

-- Repair tables

```
MSCK REPAIR TABLE redditdb.reddit_silver;
```

```
MSCK REPAIR TABLE redditdb.reddit_gold;
```

-- Top subreddits by comments

```
SELECT subreddit, SUM(comment_count)
```

```
FROM redditdb.reddit_gold
```

```
GROUP BY subreddit
```

```
ORDER BY SUM(comment_count) DESC;
```

-- Average sentiment

```
SELECT subreddit, AVG(avg_sentiment)
```

Reddit Comments AWS Data Pipeline

```
FROM redditdb.reddit_gold
```

```
GROUP BY subreddit;
```

The screenshot shows a SQL query editor interface. At the top, there are four tabs labeled 'Query 1', 'Query 2', 'Query 3', and 'Query 4'. 'Query 4' is the active tab, containing the following SQL code:

```
1 SELECT subreddit, SUM(comment_count)
2 FROM redditdb.reddit_gold
3 GROUP BY subreddit
4 ORDER BY SUM(comment_count) DESC;
```

Below the code, it says 'SQL Ln 4, Col 34'. There are several buttons at the bottom: 'Run again' (orange), 'Explain' (blue), 'Cancel' (grey), 'Clear' (grey), and 'Create' (grey). To the right of these buttons is a checkbox for 'Reuse query results up to 60 minutes ago'.

The main area is titled 'Query results' and shows a green bar indicating 'Completed'. It displays the following information: 'Time in queue: 111 ms', 'Run time: 747 ms', and 'Data scanned: 13.23 KB'. Below this, a table header 'Results (265)' is shown with columns '# ▾' and 'subreddit'. A single row is visible, labeled '1' and 'aws'. There are buttons for 'Copy' and 'Download results CSV' to the right of the table.

8. QuickSight Visuals

The reddit_gold dataset (output of Glue Gold jobs) was used for all visualizations. This dataset contains aggregated subreddit-level metrics:

- subreddit
- comment_count (total number of comments)
- avg_sentiment (average sentiment score per subreddit)

8.1 Top Subreddits by Comment Count (Bar Chart)

Description:

- X-axis: comment_count (Sum)
- Y-axis: subreddit
- Chart type: Horizontal bar

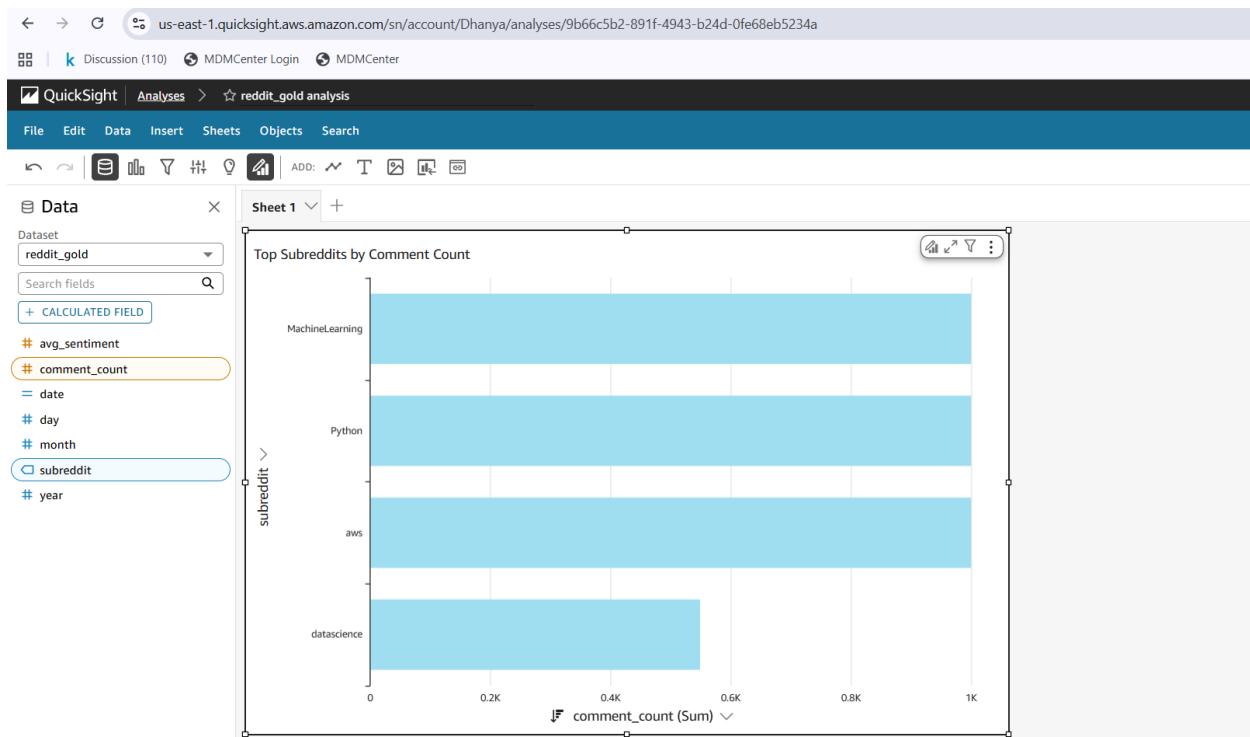
Purpose:

This chart highlights which subreddits are most active based on the volume of comments.

Reddit Comments AWS Data Pipeline

Insights:

- Subreddits like MachineLearning, Python, and AWS dominate comment activity.
- DataScience, while popular, has fewer comments compared to technical subreddits.



8.2 Average Sentiment by Subreddit (Colored Bar Chart)

Description:

- X-axis: subreddit
- Y-axis: avg_sentiment (Average)
- Color: Sentiment_Category (Negative = red, Neutral = yellow, Positive = green)

Purpose:

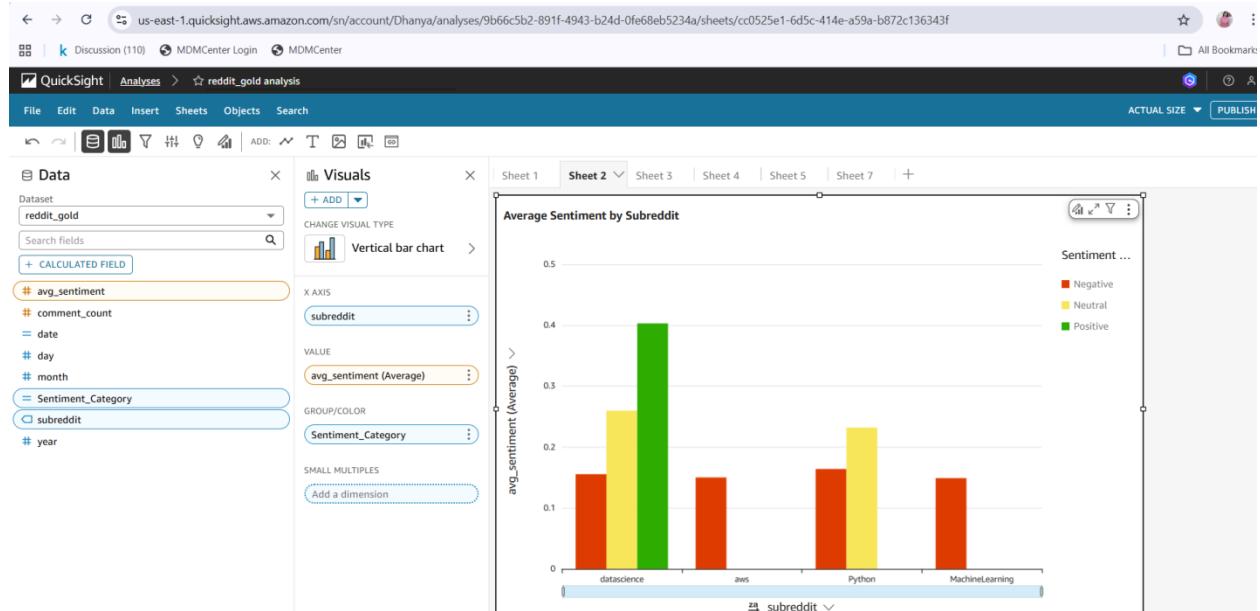
This chart compares the overall sentiment of discussions across subreddits.

Insights:

- DataScience subreddit shows more positivity (higher avg sentiment).

Reddit Comments AWS Data Pipeline

- Python and AWS communities lean neutral.
- MachineLearning has slightly more negative sentiment.
- Provides a qualitative measure of how communities feel.



8.3 Trend of Comments Over Time (Line Chart)

Description:

- X-axis: date
- Y-axis: comment_count (Sum)
- Color: subreddit

Purpose:

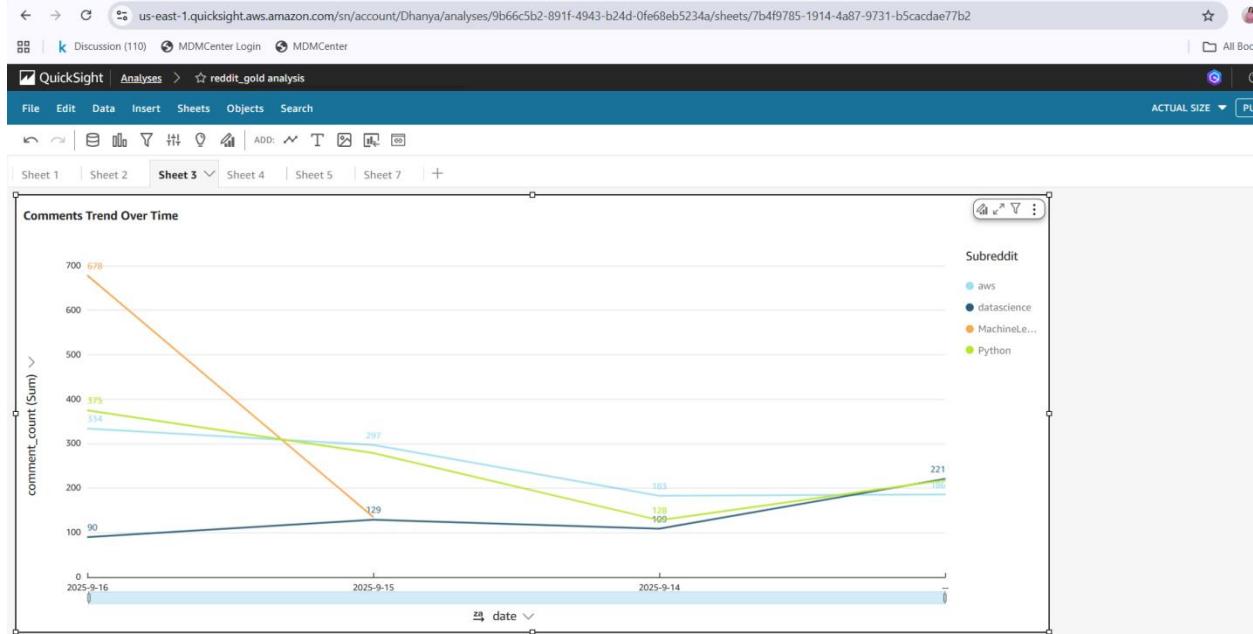
This chart reveals temporal patterns in activity.

Insights:

- Spikes in comments may align with events (e.g., AWS re:Invent, ML paper releases, Python updates).
- AWS subreddit shows sudden peaks and declines, suggesting event-driven activity.

Reddit Comments AWS Data Pipeline

- Python and MachineLearning remain steady, showing continuous community discussions.
- Useful for tracking trends and seasonality.



8.4 Sentiment Distribution (Pie Chart)

Description:

- Group/Color: Sentiment_Category
- Value: comment_count (Sum)

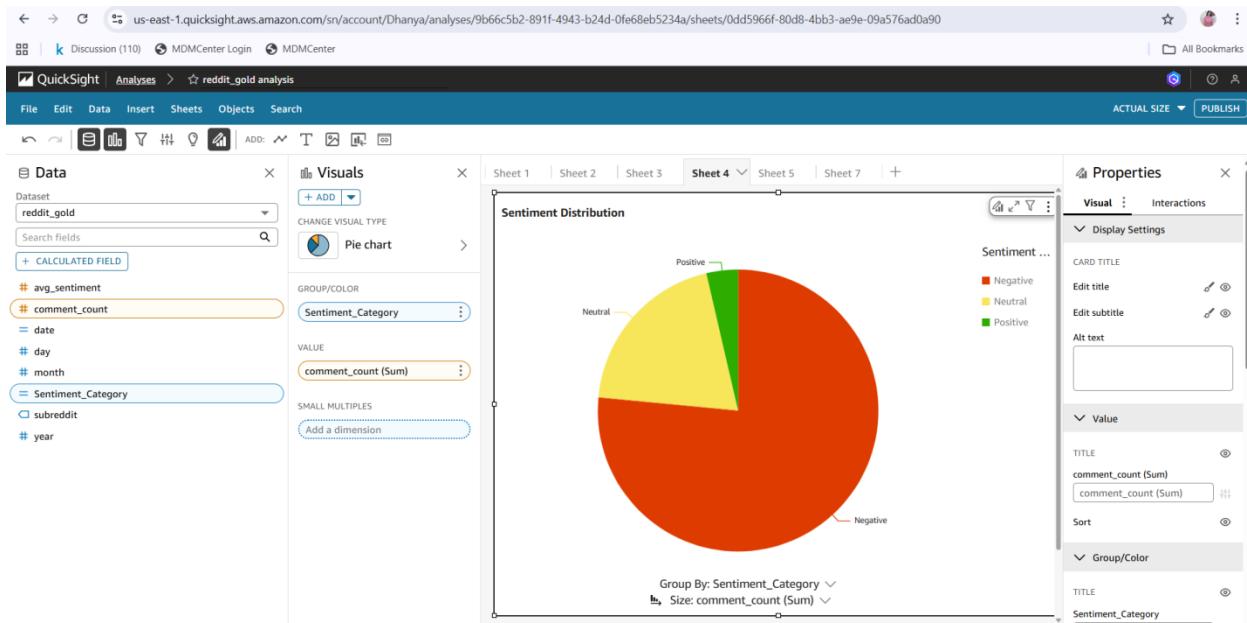
Purpose:

This visualization gives an overall breakdown of sentiment across all Reddit comments.

Insights:

- Majority of comments are Negative (red slice).
- Neutral (yellow) sentiment makes up a significant portion.
- Positive (green) sentiment is the smallest slice.
- Shows that Reddit discussions tend to be critical or cautious.

Reddit Comments AWS Data Pipeline



8.5 Heatmap (Subreddit × Day)

Description:

- Rows: subreddit
- Columns: day
- Values: comment_count (Sum)
- Color gradient: Activity intensity (light → dark blue)

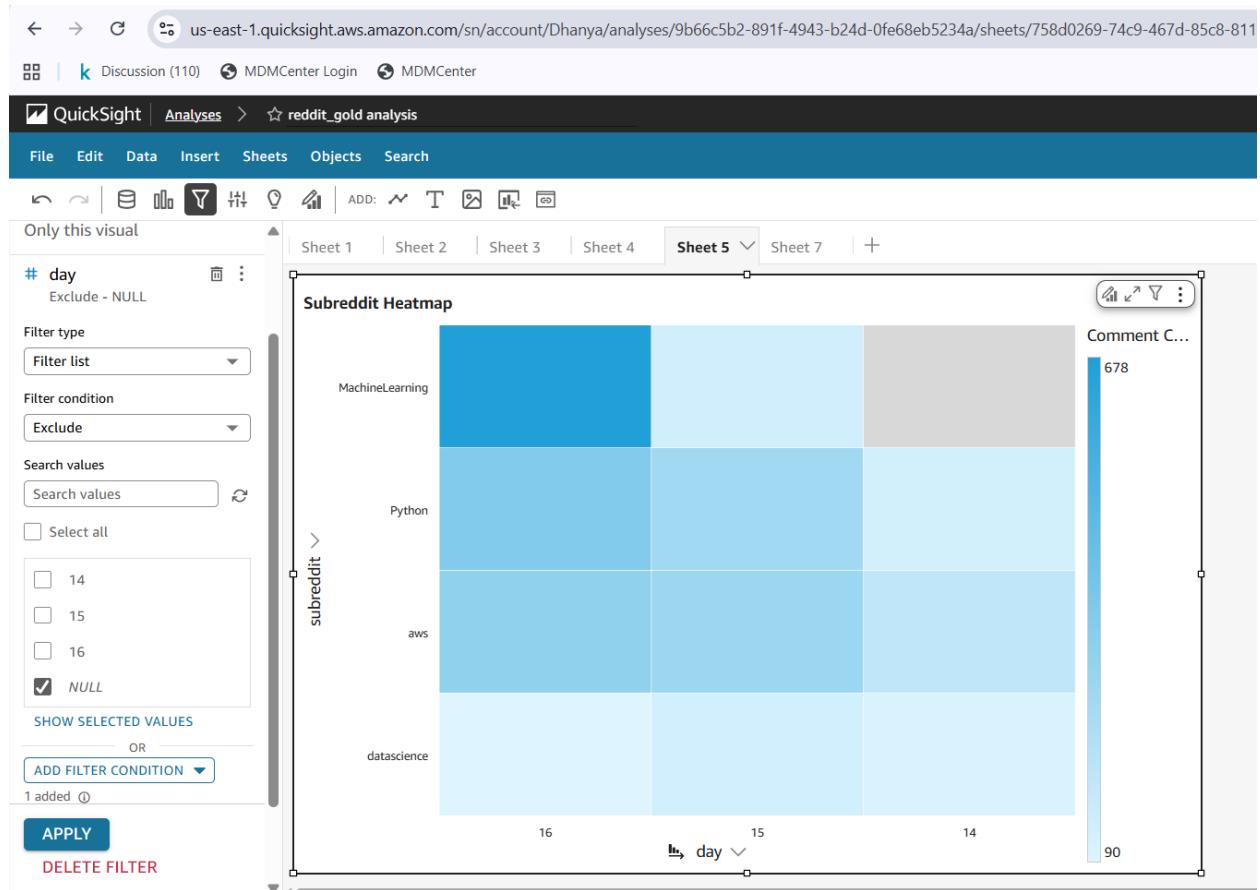
Purpose:

The heatmap provides a time vs subreddit activity view, helping detect when each subreddit is most active.

Insights:

- MachineLearning shows strong peaks on certain days.
- Python and AWS communities are consistently active.
- Some missing values (null days) suggest days with no data collection or very low activity.
- Helps in identifying daily cycles of subreddit activity.

Reddit Comments AWS Data Pipeline



8.6 Scatter Plot (Sentiment vs Comments)

Description:

- X-axis: avg_sentiment (Sum)
- Y-axis: comment_count (Sum)
- Color: subreddit

Purpose:

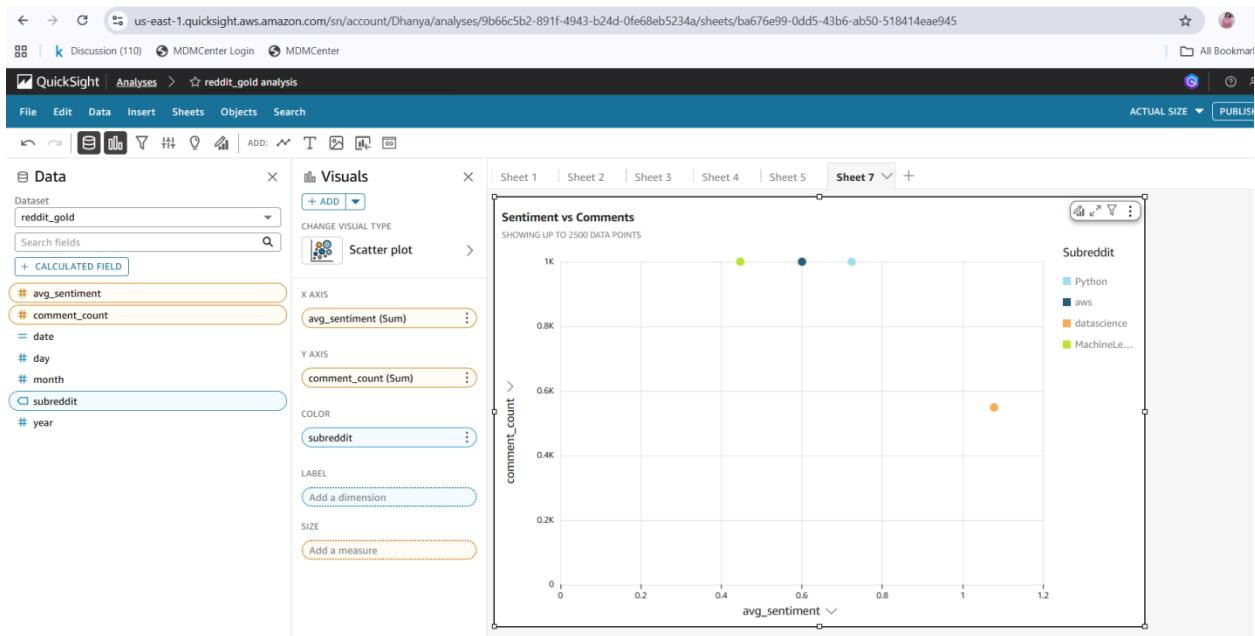
This scatter plot explores the relationship between sentiment and engagement.

Insights:

- High comment count + neutral sentiment indicates balanced discussions.
- DataScience appears with higher sentiment but fewer comments.
- AWS and Python have higher activity but lower sentiment scores.

Reddit Comments AWS Data Pipeline

- Useful for finding outliers (e.g., subreddits with high positivity but low activity).



9. Troubleshooting steps

Step/Component	Issue	Error/Mes sage	Cause	Fix/Resolution
Reddit API (PRAW)	MissingRequiredAttribut eException: client_id missing	PRAW could not find client_id	Env vars not loaded into script	Use .env with python-dotenv and pass values explicitly in DAG
Airflow DAG Import	ModuleNotFoundError: No module named 'utils'	DAG failed to import	Wrong mount path inside container	Mount project root (./opt/airflow) and place utils/ at root
Airflow Webserver Port	Bind for 0.0.0.0:8080 failed: port already allocated	Airflow container won't start	Port 8080 already in use	Change to 8081:8080 in docker-compose.yml

Reddit Comments AWS Data Pipeline

Step/Component	Issue	Error/Mes sage	Cause	Fix/Resolution
LocalFilesystemToS3Operator	S3HookUriParseFailure: Please provide a bucket name using valid format	Wrong key value	Provided reddit/bronze/... without bucket	Always pass bucket separately and only the key (e.g., bronze/...)
GlueJobOperator	Invalid type for parameter Command.ScriptLocation	Airflow tried to create a new Glue job	Glue job already exists in console	Use boto3 start_job_run instead of GlueJobOperator for existing jobs
Glue IAM Role	Invalid type for parameter RoleName, value: None	Role not passed	Env var not picked in container	Hardcode role in DAG or load from airflow.env correctly
Glue Start Job Run	EntityNotFoundException: Failed to start job run due to missing metadata	Airflow failed to trigger job	Wrong job name or region mismatch	Double-check GLUE_SILVER_JOB, GLUE_GOLD_JOB values and AWS region
Athena Queries	Table not found	Athena couldn't query	Partitions not refreshed	Run MSCK REPAIR TABLE before querying
QuickSight Date Field	parseDate syntax error	Calculated field failed	Used multi-line concat or wrong format string	Use parseDate(concat(toString({year}), '-', toString({month}), '-', toString({day})), 'yyyy-M-d')

Reddit Comments AWS Data Pipeline

Step/Component	Issue	Error/Mes sage	Cause	Fix/Resolution
QuickSight Heatmap	Null column for day	Heatmap shows “null”	Missing partition values in data	Filter nulls or ensure Glue job writes all partitions

10. Cost Optimization Guide

Service	Potential Cost Driver	Optimization Strategy
Amazon S3 (Bronze/Silver/Gold layers)	Storing large raw/processed files indefinitely	- Enable S3 Lifecycle policies (move Bronze → Glacier after 30 days)- Store Silver/Gold in Parquet instead of CSV (columnar, compressed)
AWS Glue ETL Jobs	Long job runtimes and large DPUs	- Use smaller DPU allocation (2 instead of 10) for light jobs- Enable Job Bookmarks to process only new data- Reuse scripts via script editor instead of auto-generated
AWS Glue Crawlers	Frequent unnecessary crawls	- Schedule crawlers only after new partitions arrive (daily, not hourly)- Use MSCK REPAIR TABLE instead of crawler if schema stable
Amazon Athena	Large query scans due to unpartitioned data	- Partition tables by year, month, day- Use Parquet + Snappy compression- Use SELECT columns instead of SELECT *
Amazon QuickSight	Reader session costs (\$0.30/session)	- Use SPICE datasets (in-memory, cheaper for repeated queries)- Share dashboards with limited reader accounts only
Airflow (Docker Compose local)	Running containers continuously	- Stop containers when not in use (docker-compose down)- For production: deploy on

Reddit Comments AWS Data Pipeline

Service	Potential Cost Driver	Optimization Strategy
		Fargate with auto-scaling
IAM/Permissions	Overprovisioned roles may lead to indirect costs (accidental usage)	- Use least-privilege IAM (Glue can only access required S3 buckets)- Rotate keys, disable unused creds
Logging & Monitoring (CloudWatch)	Excessive logs from Glue/Athena	- Enable log retention policy (keep 30 days)- Export important logs to S3 Glacier instead of storing indefinitely

11. Git Repo

The complete code is available in git. The repository url is:

<https://github.com/Dhanyamol-Devassy/reddit-comments-pipeline-aws/>

12. Conclusion

This project successfully demonstrated an end-to-end data engineering pipeline on AWS using Reddit comments as the data source. Starting from raw ingestion through the Reddit API, the pipeline applied modern data engineering practices — Bronze/Silver/Gold architecture, partitioned storage in S3, Glue-based ETL, Athena queries, and QuickSight visualizations.

Key outcomes:

- Built a fully automated Airflow DAG that orchestrates all stages (extract → upload → transform → validate).
- Designed Glue ETL jobs that flatten, clean, and enrich Reddit data efficiently, while leveraging partitioning and columnar formats (Parquet) to optimize performance and cost.
- Created Athena tables that enabled SQL-style queries directly on S3, minimizing the need for expensive database infrastructure.
- Delivered actionable QuickSight dashboards, offering multiple perspectives — comment volumes, sentiment distribution, temporal trends, and subreddit-level comparisons.

Reddit Comments AWS Data Pipeline

- Documented detailed troubleshooting steps and cost optimization strategies, ensuring the pipeline is both reliable and production-ready.

Ultimately, this pipeline is extensible — more subreddits or timeframes can be added easily, and the Gold dataset can support advanced analytics such as topic modeling, anomaly detection, or forecasting. This foundation proves the viability of building scalable, cost-effective, and insightful cloud-based data platforms.