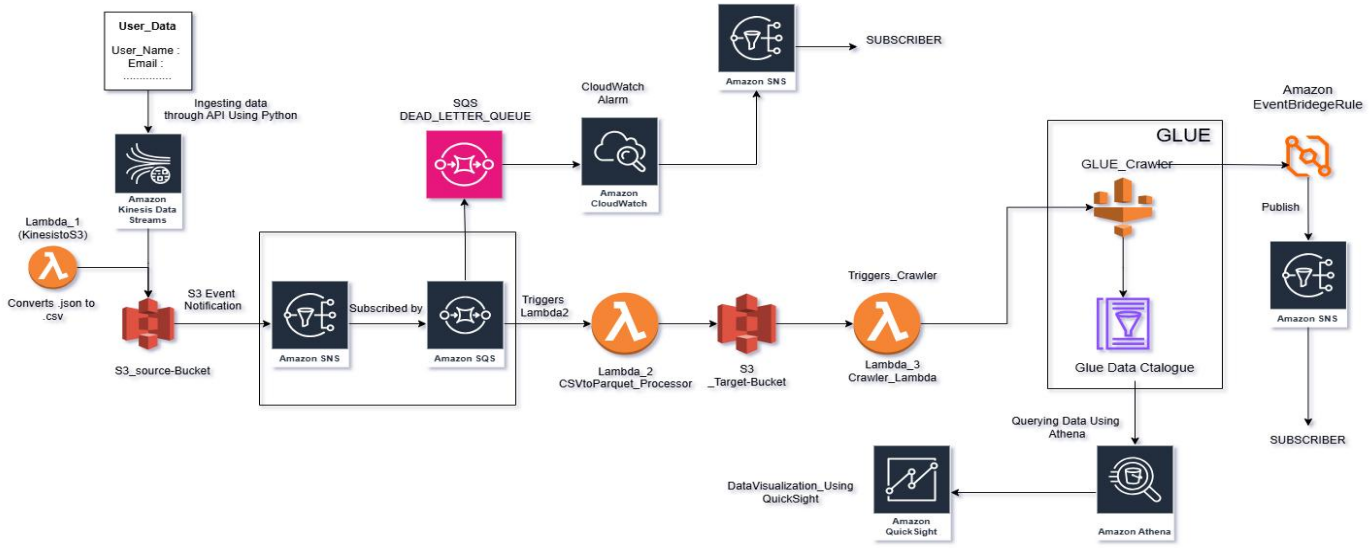
MAHIDAR REDDY PUTTA

Scalability Event-Driven Pipeline for User Engagement Analysis



Technologies Used

- **Amazon Kinesis Data Stream**
- **AWS Lambda**
- **AWS S3**
- **Amazon SNS**
- **Amazon SQS**
- **Amazon CloudWatch Alarm**
- **Amazon EventBridge**
- **AWS Glue**
- **Amazon Athena**
- **AWS QuickSight**

API: Ninjas API

Python Libraries: Boto3, PyArrow, Pandas

Overview

This project implements a scalable, event-driven pipeline for analyzing user engagement data. The pipeline ingests real-time data using AWS Kinesis, transforms the data using Lambda functions, stores the results in Amazon S3, and visualizes insights using Amazon QuickSight.

Process Breakdown

1. Extract: Real-Time Data Ingestion

- Data is extracted from an external API using Python in a local environment (PyCharm).
- AWS Kinesis Data Stream ingests the incoming data in JSON format.

2. Transform: Data Conversion

- Lambda Function 1 converts JSON data into CSV format.
- Lambda Function 2 converts CSV data into Parquet format using Pandas and PyArrow.

3. Load: Data Storage

- Transformed data is stored in the Target S3 bucket.
- AWS Glue Crawler reads the Parquet files and updates the Glue Data Catalogue.
- Data is then available for querying using Amazon Athena.

Step-by-Step Implementation

Step 1: Data Ingestion and Initial Transformation

- Created Kinesis Data Stream for real-time data ingestion.
- Lambda 1 was configured to fetch data from Kinesis, convert it from JSON to CSV, and store it in the Source S3 bucket.

Step 2: Notification and Processing

- Created an Amazon SNS Topic to receive event notifications from S3 upon CSV upload.
- Amazon SQS was configured as a subscriber to SNS to queue messages.
- Lambda 2 reads SQS messages, extracts data using Boto3, converts it into Parquet format using Pandas, and stores it in the Target S3 bucket.

Step 3: Data Cataloging and Schema Creation

- AWS Glue Data Catalogue was created to store metadata.
- Glue Crawler was configured to read Parquet files from the Target S3 bucket and generate the schema.
- Lambda 3 triggers the Glue Crawler whenever a new Parquet file is added to the bucket.

Step 4: Data Analysis and Visualization

- SQL queries were executed in Amazon Athena to analyze the data.
- Two key queries were created for:
    - User Count by Age Group
    - User Count by Company
- The query results were visualized using Amazon QuickSight by creating an interactive dashboard.

Additional Components

Dead Letter Queue (DLQ)

- Configured an Amazon SQS Dead Letter Queue for Lambda 2 to capture failed messages.
- Created a CloudWatch Alarm to monitor the DLQ.
- Set up an SNS Topic to send notifications when the alarm is triggered.

Event Monitoring

- Amazon EventBridge Rule monitors Glue Crawler failures.
- When a failure occurs, an SNS Topic sends alerts to the subscribers.

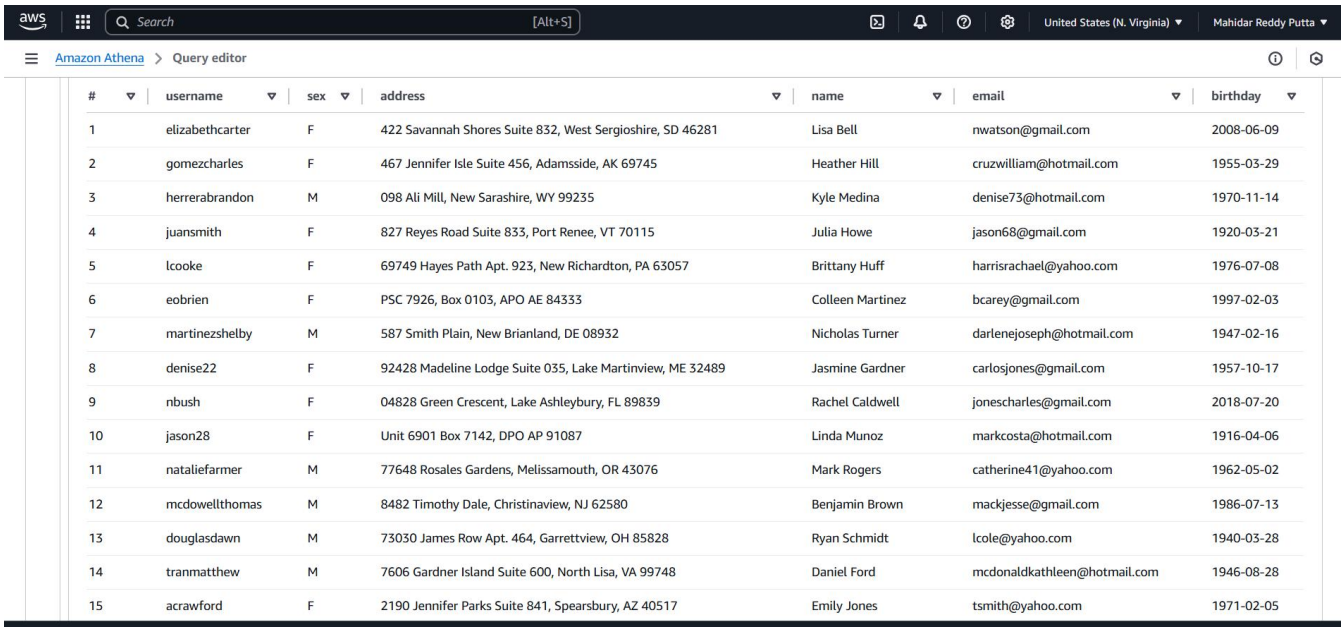
Conclusion

The implemented pipeline ensures a reliable and scalable data processing flow. With real-time data ingestion, automatic schema creation, and insightful visualizations, this project showcases the power of AWS services for large-scale data analytics.

ATHENA RESULTS

Query 1

*SELECT * FROM "user-data"."converted" ;*



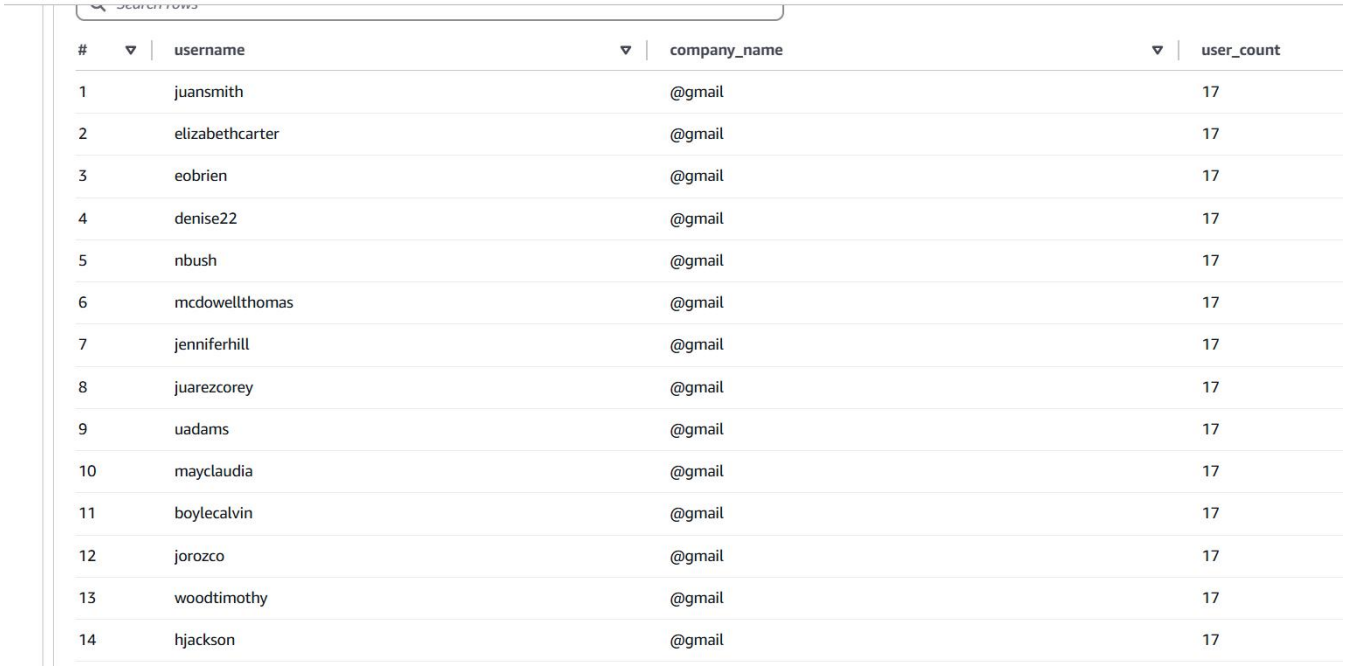
#	username	sex	address	name	email	birthday
1	elizabethcarter	F	422 Savannah Shores Suite 832, West Sergioshire, SD 46281	Lisa Bell	nwatson@gmail.com	2008-06-09
2	gomezcharles	F	467 Jennifer Isle Suite 456, Adamsside, AK 69745	Heather Hill	cruzwilliam@hotmail.com	1955-03-29
3	herrerabrandon	M	098 Ali Mill, New Sarashire, WY 99235	Kyle Medina	denise73@hotmail.com	1970-11-14
4	juansmith	F	827 Reyes Road Suite 833, Port Renee, VT 70115	Julia Howe	jason68@gmail.com	1920-03-21
5	lcooke	F	69749 Hayes Path Apt. 923, New Richardton, PA 63057	Brittany Huff	harrisrachael@yahoo.com	1976-07-08
6	eobrien	F	PSC 7926, Box 0103, APO AE 84333	Colleen Martinez	bcarey@gmail.com	1997-02-03
7	martinezshelby	M	587 Smith Plain, New Brianland, DE 08932	Nicholas Turner	darlenejoseph@hotmail.com	1947-02-16
8	denise22	F	92428 Madeline Lodge Suite 035, Lake Martinview, ME 32489	Jasmine Gardner	carlosjones@gmail.com	1957-10-17
9	nbush	F	04828 Green Crescent, Lake Ashleybury, FL 89839	Rachel Caldwell	jonescharles@gmail.com	2018-07-20
10	jason28	F	Unit 6901 Box 7142, DPO AP 91087	Linda Munoz	markcosta@hotmail.com	1916-04-06
11	nataliefarmer	M	77648 Rosales Gardens, Melissamouth, OR 43076	Mark Rogers	catherine41@yahoo.com	1962-05-02
12	mcdowellthomas	M	8482 Timothy Dale, Christinaview, NJ 62580	Benjamin Brown	mackjesse@gmail.com	1986-07-13
13	douglasdawn	M	73030 James Row Apt. 464, Garrettview, OH 85828	Ryan Schmidt	lcole@yahoo.com	1940-03-28
14	tranmatthew	M	7606 Gardner Island Suite 600, North Lisa, VA 99748	Daniel Ford	mcdonaldkathleen@hotmail.com	1946-08-28
15	acrawford	F	2190 Jennifer Parks Suite 841, Spearsbury, AZ 40517	Emily Jones	tsmith@yahoo.com	1971-02-05

Query 2

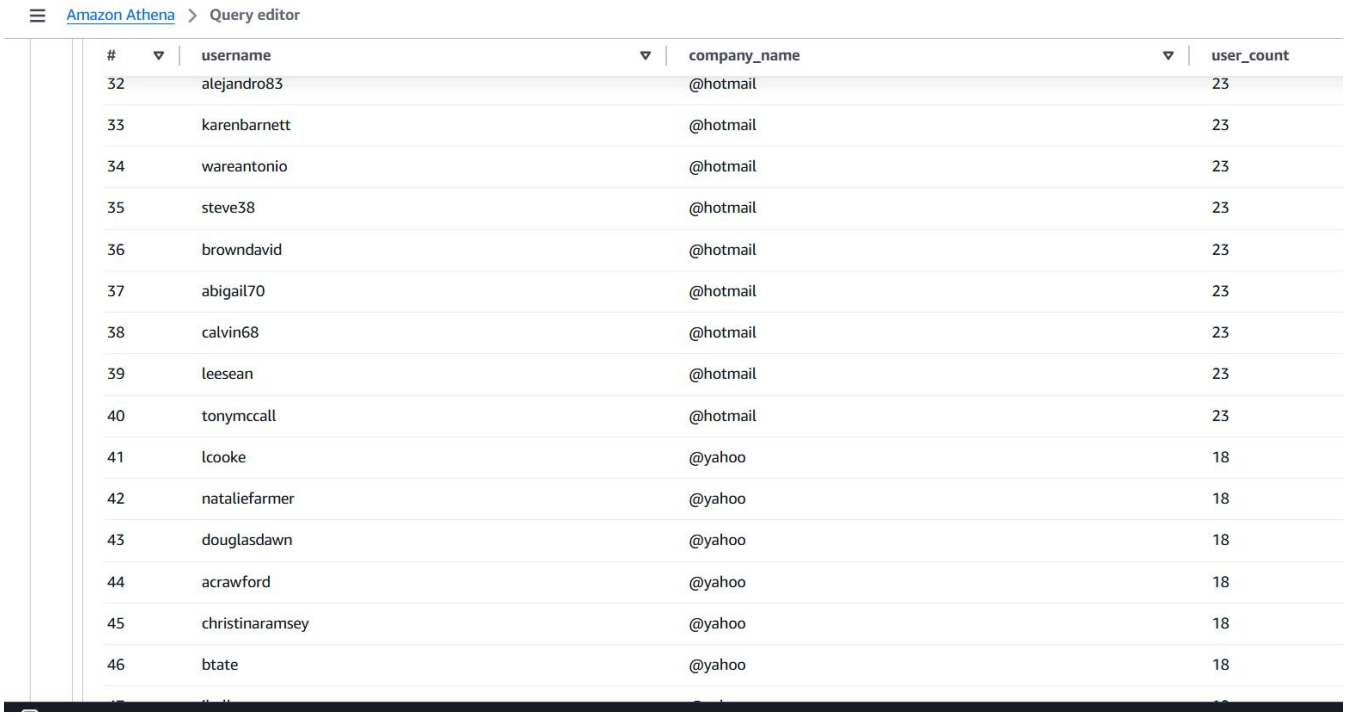
SELECT username, REGEXP_EXTRACT(email, '@([^\\.]+)') AS company_name,

COUNT(*) OVER (PARTITION BY REGEXP_EXTRACT(email, '@([^\\.]+)')) AS user_count

FROM "user-data".converted



#	username	company_name	user_count
1	juansmith	@gmail	17
2	elizabethcarter	@gmail	17
3	eobrien	@gmail	17
4	denise22	@gmail	17
5	nbush	@gmail	17
6	mcdowellthomas	@gmail	17
7	jenniferhill	@gmail	17
8	juarezcorey	@gmail	17
9	uadams	@gmail	17
10	mayclaudia	@gmail	17
11	boylecalvin	@gmail	17
12	jorozco	@gmail	17
13	woodtimothy	@gmail	17
14	hjackson	@gmail	17



#	username	company_name	user_count
32	alejandro83	@hotmail	23
33	karenbarnett	@hotmail	23
34	wareantonio	@hotmail	23
35	steve38	@hotmail	23
36	browndavid	@hotmail	23
37	abigail70	@hotmail	23
38	calvin68	@hotmail	23
39	leesean	@hotmail	23
40	tonymccall	@hotmail	23
41	lcooke	@yahoo	18
42	nataliefarmer	@yahoo	18
43	douglasdawn	@yahoo	18
44	acrawford	@yahoo	18
45	christinaramsey	@yahoo	18
46	btate	@yahoo	18

Query 3

SELECT CASE

WHEN year(current_date) - year(date_parse(birthday, '%Y-%m-%d')) BETWEEN 0 AND 20 THEN '0-20'

WHEN year(current_date) - year(date_parse(birthday, '%Y-%m-%d')) BETWEEN 21 AND 40 THEN '21-40'

WHEN year(current_date) - year(date_parse(birthday, '%Y-%m-%d')) BETWEEN 41 AND 60 THEN '41-60'

ELSE '61+' END AS age_group, COUNT(*) AS user_count FROM "user-data".converted GROUP BY 1 ORDER BY 1;

Query results | Query stats

Completed

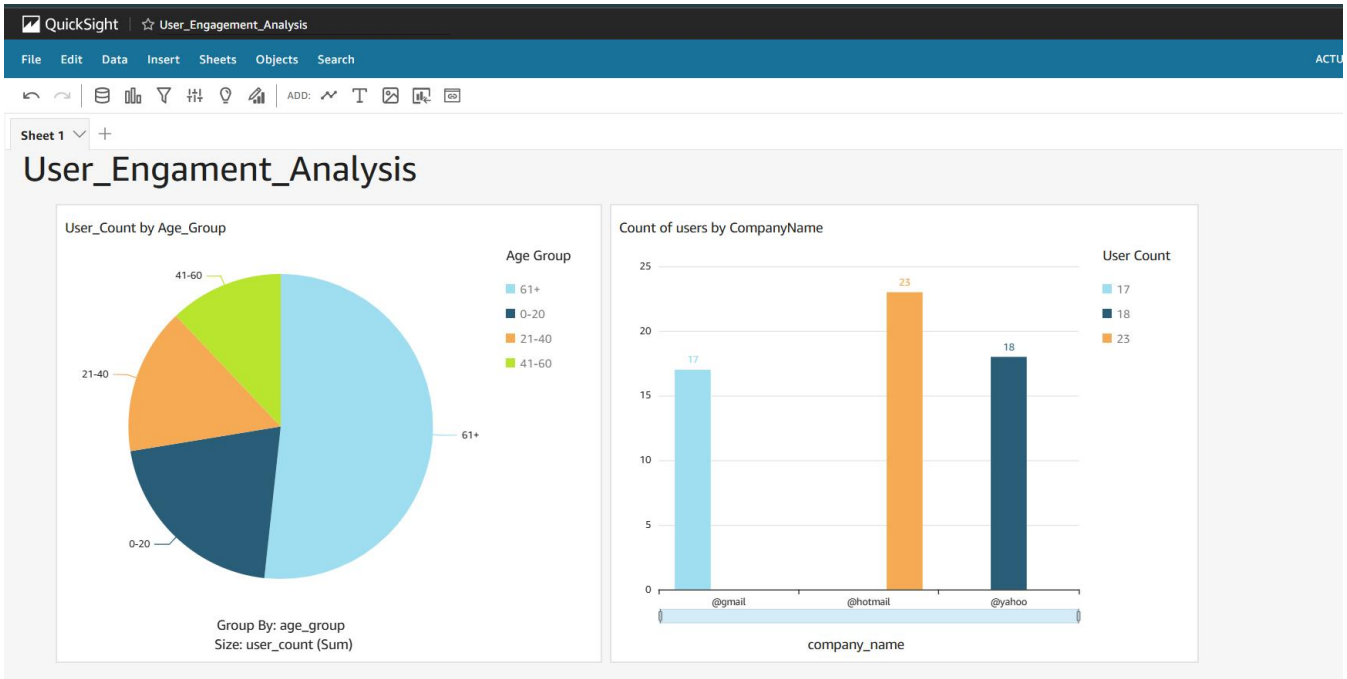
Results (4)

Search rows

#	age_group	user_count
1	0-20	12
2	21-40	9
3	41-60	7
4	61+	30

Amazon QuickSight Dashboard

URL : https://us-east-1.quicksight.aws.amazon.com/sn/accounts/120569637987/dashboards/915e3f92-2821-42dd-af07-8994edd241c3?directory_alias=mahidar120569637987



GITHUB URL : https://github.com/Mahidar1010/EventDrivenPipeline