# **Report: User Click Analysis(GA - 3)**

## **Introduction**

This report details the process of generating a CSV file with user IDs and timestamps, processing the data using PySpark, and analyzing the user clicks within specified time intervals.

## **Data Generation**

We generated a dataset containing 50 user IDs with random timestamps. The timestamps span across various dates and times to simulate user click activities. The following Python script was used to create the dataset:

python

Copy code

import csv

import random

from datetime import timedelta, datetime

def generate\_random\_timestamp(base\_date=datetime(2024, 1, 1)):

return base\_date + timedelta(

days=random.randint(1, 365),

hours=random.randint(0, 23),

minutes=random.randint(0, 59),

seconds=random.randint(0, 59)

)

data = [

(f"User\_{i + 1}", generate\_random\_timestamp()) for i in range(50)

]

output\_file = "generated\_data.csv"

with open(output\_file, "w", newline="") as file:

writer = csv.writer(file)

writer.writerow(["id", "timestamp", "date"])

writer.writerows((user\_id, ts.strftime("%Y-%m-%d %H:%M:%S"), ts.strftime("%Y-%m-%d")) for user\_id, ts in data)

This script generates a CSV file named generated\_data.csv with the following structure:

bash

Copy code

id,timestamp,date

User\_1,2024-03-16 08:47:23,2024-03-16

User\_2,2024-06-11 12:15:42,2024-06-11

...

## **Data Processing with PySpark**

We used PySpark to process the generated data. The main goal was to count the number of user clicks within different time intervals: 00-06, 06-12, 12-18, and 18-24 hours. The following PySpark script was executed on Google Cloud Platform (GCP):

python

Copy code

from pyspark.sql import SparkSession

from pyspark.sql.functions import col, hour, when

# Initialize Spark session

spark = SparkSession.builder.appName("user\_click\_counter").getOrCreate()

# Input and output file paths

input\_file = "gs://your-bucket/generated\_data.csv" # Update with your GCS bucket path

output\_file = "output.txt"

# Read the data

data = spark.read.option("header", "true").option("delimiter", ",").csv(input\_file)

# Extract hour from timestamp

data = data.withColumn("Hour", hour(data["timestamp"]))

# Define time intervals

data = data.withColumn(

"time\_interval",

when(col("Hour") < 6, "00-06").

when(col("Hour") < 12, "06-12").

when(col("Hour") < 18, "12-18").

when(col("Hour") < 24, "18-24").

otherwise("Invalid timestamp")

)

# Group and count the data by time interval

result = data.groupBy("time\_interval").count().sort("time\_interval")

result.show()

# Save the result to a file

result.toPandas().to\_csv(output\_file, sep="\t", index=False, header=True)

# Stop the Spark session

spark.stop()

## **Results**

The PySpark script processes the CSV file, categorizes the timestamps into the defined time intervals, and counts the number of clicks in each interval. The output is saved in output.txt with the following format:

Copy code

time\_interval count

00-06 12

06-12 15

12-18 10

18-24 13

This output shows the distribution of user clicks across the different time intervals, providing insights into peak activity periods.

## **Conclusion**

This project involved generating a random dataset of user IDs and timestamps, processing the data with PySpark to categorize and count user clicks within specified time intervals, and analyzing the results. The process demonstrated effective use of data generation, cloud-based data processing, and result analysis techniques.

## **Appendix**

### **Generated Data Sample**

Below is a sample of the generated data:

bash

Copy code

id,timestamp,date

User\_1,2024-03-16 08:47:23,2024-03-16

User\_2,2024-06-11 12:15:42,2024-06-11

User\_3,2024-10-29 22:03:05,2024-10-29

...

### **PySpark Output Sample**

Below is a sample of the PySpark output:

Copy code

time\_interval count

00-06 12

06-12 15

12-18 10

18-24 13

This detailed report provides a comprehensive overview of the project, from data generation to processing and analysis.