

INFO I535 Course Project: Big Data Concepts and Implementations

**International Flight Insights: Crafting a Data Pipeline  
for Global Travel Trends**

*Submitted by:*

Ravi Sharma (ravishar)

# Table of Contents

<b><u>INTRODUCTION .....</u></b>	<b><u>3</u></b>
<b><u>BACKGROUND .....</u></b>	<b><u>4</u></b>
<b><u>METHODOLOGY .....</u></b>	<b><u>5</u></b>
<b><u>RESULTS .....</u></b>	<b><u>8</u></b>
<b><u>DISCUSSION .....</u></b>	<b><u>12</u></b>
<b><u>CONCLUSION .....</u></b>	<b><u>14</u></b>
<b><u>REFERENCES .....</u></b>	<b><u>16</u></b>

## Introduction

In this project, the aim was to practice and implement what I learned in this class and combine it with my previous experiences in a manner that suits my learning style and experience. The project focuses on building a data processing pipeline and performing analyses using big data technologies on the Google Cloud Platform on one of the suggested datasets i.e. Bureau of Transportation Statistics Flight Performance, I particularly went ahead with [the Air Carriers : T-100 International Segment dataset](#) that had like 8500+ rows and 43 features and contained all necessary information like origin city, destination city, carrier, number of passengers, distance and so on about international flights that either had their origin or destination in the United States.

The project seeks to uncover valuable insights into international flight patterns, airlines, passenger traffic, and international hotspots through the utilization of cutting-edge managed and serverless services offered by Google Cloud Platform like cloud storage, Dataproc, BigQuery and Google Looker studio in the end to create interactive and informative visualizations.

## Background

I am particularly drawn towards this topic as I have a keen interest in the Aviation industry, and I belong to the group of people who admire these flying machines. This dataset combines the best of my interests i.e. Aviation and Data Science. Additionally, the transport industry in general is not given the attention and credit it deserves, the amount of data-driven decisions that these companies make is humungous, and data is involved everywhere right from pricing decisions to scheduling and it just requires small mistakes that disrupt the entire workflow of the industry. With modern avionics and next-generation auto-pilot systems, the aviation industry is highly dependent on data in some or the other way.

# Methodology

What you will need?

1. Cloud Storage
2. Dataproc Cluster
3. BigQuery
4. Looker Studio

The workflow for this project is as follows :-

**1) Data Acquisition:** The data used for this project is one of the suggested datasets i.e. Bureau of Transportation Statistics Flight Performance. The dataset for the first quarter of 2024 was obtained by downloading the CSV file from the given link and then stored in the Google Cloud Storage bucket for further use. The dataset provided by the Bureau of Transportation Statistics is well structured and has a documentation file along with it that provides information about all the features present in the dataset. The dataset has a total of 8641 rows and 43 columns, out of which a lot columns had redundant meanings and the dataset indeed needed some cleaning like removing columns containing city ID, airline ID, airport ID, etc. which were practically of no use in this case as we also have corresponding names that makes our lives much more easier.

**2) Data processing, cleaning, validation, and modelling using Dataproc:** The hero for this project has to be the Dataproc cloud service. Google Cloud Platform's dataproc is a fully managed cloud service that provides users with a fast, easy and cost-effective way to run Apache Spark and Hadoop clusters i.e. leveraging the power of cloud computing using distributed storage and computing. Dataproc simplifies the process of deploying, managing and scaling big data processing infrastructure and helps us focus on just writing the logic of the process rather than managing the underlying infrastructure. The users still have control over the size, storage and processing power of the clusters. In this case, keeping in mind the resource quota I used a Master node with n2-standard-4 and two worker nodes with n2-standard-2 with 50GB storage on all of them. The reason why dataproc is well suited for this job is because of the Google Cloud ecosystem that lets us seamlessly connect other powerful services offered by GCP like BigQuery and BigTable.

Cluster details
SUBMIT JOB
REFRESH
START
STOP
DELETE
VIEW LOGS

For PD-Standard without local SSDs, we strongly recommend provisioning 1TB or larger to ensure consistently high I/O performance. See <https://cloud.google.com/compute/docs/disks/performance> for information on disk I/O performance.

[MORE](#)

Name	cluster-37d3
Cluster UUID	404bee3b-4862-4a30-bbb1-90a5cb965437
Type	Dataprox Cluster
Status	Stopped

MONITORING
JOBS
VM INSTANCES
CONFIGURATION
WEB INTERFACES

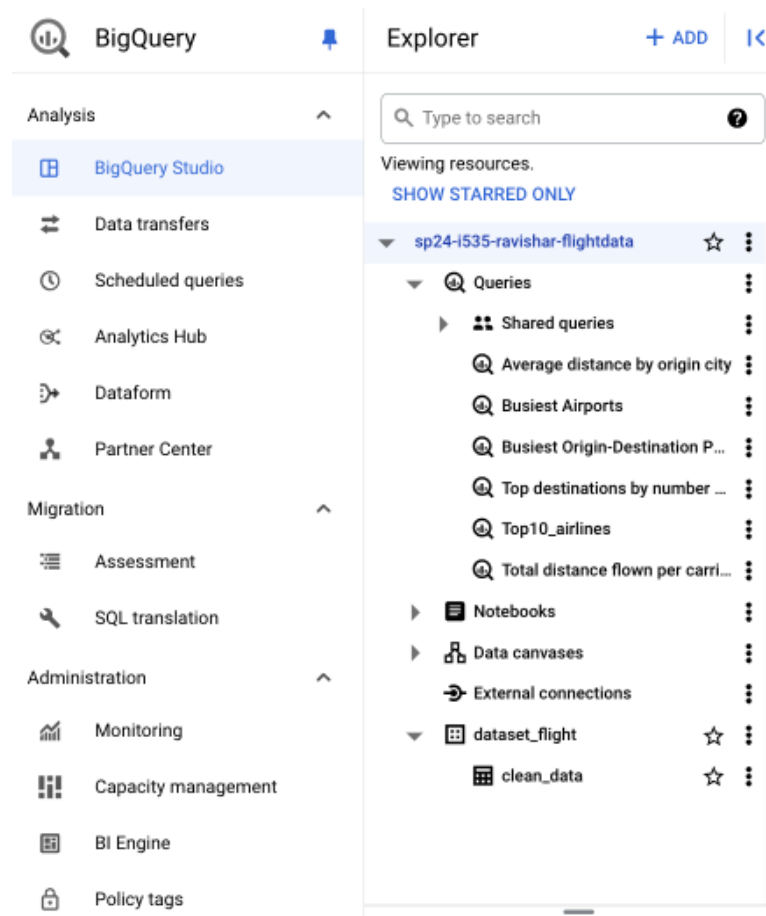
Filter Filter instances

	Name	Role	Machine type
●	<a href="#">cluster-37d3-m</a>	Master	n2-standard-4
●	<a href="#">cluster-37d3-w-0</a>	Worker	n2-standard-2
●	<a href="#">cluster-37d3-w-1</a>	Worker	n2-standard-2

IMG 1 : Screenshot with cluster details

I used the PySpark service of Dataprox to load, validate, clean, transform and write desired data into BigQuery for further analysis. The Python file, in short, reads the file from the cloud storage and validates that values such as distance, payload and passengers are non-negative and verifies that essential fields such as origin, destination, distance, etc. are not null. Then, the code proceeds to remove duplicate records and logically just keeps the columns that actually make sense and then checks if any columns have the same value for all the rows which in that case raises a ValueError. After all these preprocessing and validation steps, the final dataset is then written into a BigQuery dataset which had to be initialized before the execution of our python file. As DataProc leverages distributed computing and storage all these tasks were done on this large dataset just in 1min 2secs taking into consideration that it takes approximately 60 seconds for initialization of the spark cluster. Though all this may sound very simple and straightforward I faced a lot of challenges regarding networking and setting up the dataprox cluster in the cloud environment which I would like to discuss in the discussion section.

**3) Analysis and Visualization:** The tools that I utilized for analysis and visualization are BigQuery and Looker Studio. BigQuery is a fully managed and serverless data warehouse solution by GCP, it has been designed with the ability to handle massive datasets with superior scalability. BigQuery makes it really easy to execute SQL queries rapidly across distributed computing resources and supports a wide range of data formats. And connecting Lookers Studio to BigQuery is just a two-click process. Lookers Studio allows us to create interactive visual reports/dashboards like PowerBI and Tableau in GCP. I managed to write a lot of interesting SQL queries on the dataset using BigQuery and create a couple of visually stunning and informative dashboards on Lookers Studio.



IMG 2 : Screenshot of BigQuery console with dataset, table and queries

## Results

After all the data validation and cleaning, what truly matters is how well we managed to extract useful insights from the data. Here are some selected screenshots of SQL queries and reports created using BigQuery and Looker Studio (All files can be found [here](#)).

The screenshot shows the 'Job details' page for a job with ID 'job-f3f43b35'. The job status is 'Succeeded'. The output tab is selected, showing a log of messages from the Spark environment. The log indicates that the job successfully registered various components, connected to the ResourceManager, and completed the cleanup jobs.

```

24/04/27 03:09:48 INFO SparkEnv: Registering MapOutputTracker
24/04/27 03:09:48 INFO SparkEnv: Registering BlockManagerMaster
24/04/27 03:09:48 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
24/04/27 03:09:48 INFO SparkEnv: Registering OutputCommitCoordinator
24/04/27 03:09:49 INFO DefaultNoHARMAFailoverProxyProvider: Connecting to ResourceManager at cluster-37d3-m.c.sp24-1535-ravishar-flightdata.internal./10.142.0.7:8032
24/04/27 03:09:49 INFO AHSPProxy: Connecting to Application History server at cluster-37d3-m.c.sp24-1535-ravishar-flightdata.internal./10.142.0.7:10200
24/04/27 03:09:50 INFO Configuration: resource-types.xml not found
24/04/27 03:09:50 INFO ResourceUtils: Unable to find 'resource-types.xml'.
24/04/27 03:09:50 INFO YarnClientImpl: Submitted application application_1714184432319_0006
24/04/27 03:09:51 INFO DefaultNoHARMAFailoverProxyProvider: Connecting to ResourceManager at cluster-37d3-m.c.sp24-1535-ravishar-flightdata.internal./10.142.0.7:8038
24/04/27 03:09:53 INFO MetricsConfig: Loaded properties from hadoop-metrics2.properties
24/04/27 03:09:53 INFO MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
24/04/27 03:09:53 INFO MetricsSystemImpl: google-hadoop-file-system metrics system started
24/04/27 03:09:54 INFO GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object already exists with desired state.
24/04/27 03:09:54 INFO GoogleHadoopOutputStream: hflush(): No-op due to rate limit (RateLimiter[tableRate=0.2ops]); readers will *not* yet see flushed data for gs://dataproc-temp-us-east1-1286794815-c7xhdh
24/04/27 03:10:29 INFO SparkBigQueryConnectorModule: Registering cleanup jobs listener, should happen just once
24/04/27 03:10:30 WARN SparkStringTils: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.sql.debug.maxToStringFields'.
24/04/27 03:10:36 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired 'gs://flight_data_bucket/spark-bigquery-application_1714184432319_0006-dc12185b-e5df-4fd4-a200-1176f25234a8/' directory.
24/04/27 03:10:37 INFO BigQueryClient: Submitted job LoadJobConfiguration(type=LOAD, destinationTable=GenericData(classInfo=[datasetId, projectId, tableId], {datasetId=dataset_flight, projectId=sp24-1535-rav
24/04/27 03:10:40 INFO BigQueryClient: Done loading to sp24-1535-ravishar-flightdata.dataset_flight.clean_data. jobId: JobId(project=sp24-1535-ravishar-flightdata, job=fb1238e3-1aa8-493d-b8af-e8d48e8d3b, 1
Complete
24/04/27 03:10:40 INFO SparkBigQueryConnectorModule: In SparkListener.onApplicationEnd, going to activate cleanup jobs
24/04/27 03:10:40 INFO BigQueryClient: Running cleanup jobs. Jobs count is 0
24/04/27 03:10:40 INFO BigQueryClient: Clearing the cleanup jobs list
24/04/27 03:10:40 INFO BigQueryClient: Finished to run cleanup jobs.
  
```

IMG 3 : Screenshot of Succeeded Dataproc Job

The screenshot shows a BigQuery SQL query and its results. The query is designed to calculate the average number of passengers per flight for each carrier, ordered by the average number of passengers in descending order.

```

1 SELECT UNIQUE_CARRIER_NAME, ROUND(AVG(CAST(PASSENGERS AS FLOAT64))) AS avg_passengers_per_flight
2 FROM sp24-1535-ravishar-flightdata.dataset_flight.clean_data
3 GROUP BY UNIQUE_CARRIER_NAME
4 ORDER BY avg_passengers_per_flight DESC;
5
  
```

The query results are displayed in a table with 20 rows, showing the carrier name and the average number of passengers per flight.

Row	UNIQUE_CARRIER_NAME	avg_passengers_per_flight
1	Qatar Airways (Q.C.S.C)	12968.0
2	Kuwait Airways Corp.	9008.0
3	Jin Air Co Ltd.	8199.0
4	Eva Airways Corporation	8108.0
5	Jeju Air Co Ltd.	7935.0
6	Philippine Airlines Inc.	7311.0
7	Ethiad Airways	7003.0
8	Aerolineas Argentinas	6314.0
9	Kenya Airways PLC	6253.0
10	El Al Israel Airlines Ltd.	6249.0
11	Taca International Airlines	5880.0
12	Swiss International Airlines	5478.0
13	STARLUX AIRLINES	5246.0
14	All Nippon Airways Co.	5159.0
15	Air New Zealand	5018.0
16	Sky Airline Peru	4966.0
17	ZIPAIR Tokyo Inc.	4743.0
18	Ethiopian Airlines	4661.0
19	Asiana Airlines Inc.	4645.0
20	Compagnie Natl Air France	4492.0

A 'Load more' button is visible at the bottom of the results table.

IMG 4 : SQL Query to select total distance per carrier



🔍 **Busiest Origin-Destination Pairs by number of fli...** [▶ RUN](#) [📄 SAVE QUERY ▼](#)

```

1 SELECT ORIGIN_CITY_NAME, DEST_CITY_NAME, COUNT(*) AS num_flights
2 FROM sp24-i535-ravishar-flightdata.dataset_flight.clean_data
3 GROUP BY ORIGIN_CITY_NAME, DEST_CITY_NAME
4 ORDER BY num_flights DESC
5 LIMIT 10;

```

Query results

JOB INFORMATION	RESULTS	CHART	JSON	EXECUTION DETAILS	EXECUTION
Row	ORIGIN_CITY_NAME ▼	DEST_CITY_NAME ▼	num_flights ▼		
1	Bogota, Colombia	Miami, FL	28		
2	Miami, FL	Bogota, Colombia	26		
3	Los Angeles, CA	Tokyo, Japan	21		
4	Tokyo, Japan	Los Angeles, CA	21		
5	Miami, FL	Sao Paulo, Brazil	19		
6	Chicago, IL	Cancun, Mexico	18		
7	Seoul, South Korea	Anchorage, AK	18		
8	Hong Kong, Hong Kong	Anchorage, AK	18		

[Load more](#)

IMG 5 : SQL Query to select busiest origin-destination pairs

🔍 **Busiest Airports** [▶ RUN](#) [📄 SAVE QUERY ▼](#) [📄 DOWNLOAD](#)

```

1 WITH city_counts AS (
2     SELECT ORIGIN_CITY_NAME AS city, COUNT(*) AS total_departures
3     FROM sp24-i535-ravishar-flightdata.dataset_flight.clean_data
4     GROUP BY ORIGIN_CITY_NAME
5     UNION ALL
6     SELECT DEST_CITY_NAME AS city, COUNT(*) AS total_departures
7     FROM sp24-i535-ravishar-flightdata.dataset_flight.clean_data
8     GROUP BY DEST_CITY_NAME
9 )
10 SELECT
11     city,
12     SUM(total_departures) AS total_count
13 FROM
14     city_counts
15 GROUP BY
16     city
17 ORDER BY
18     total_count DESC
19 LIMIT 10;

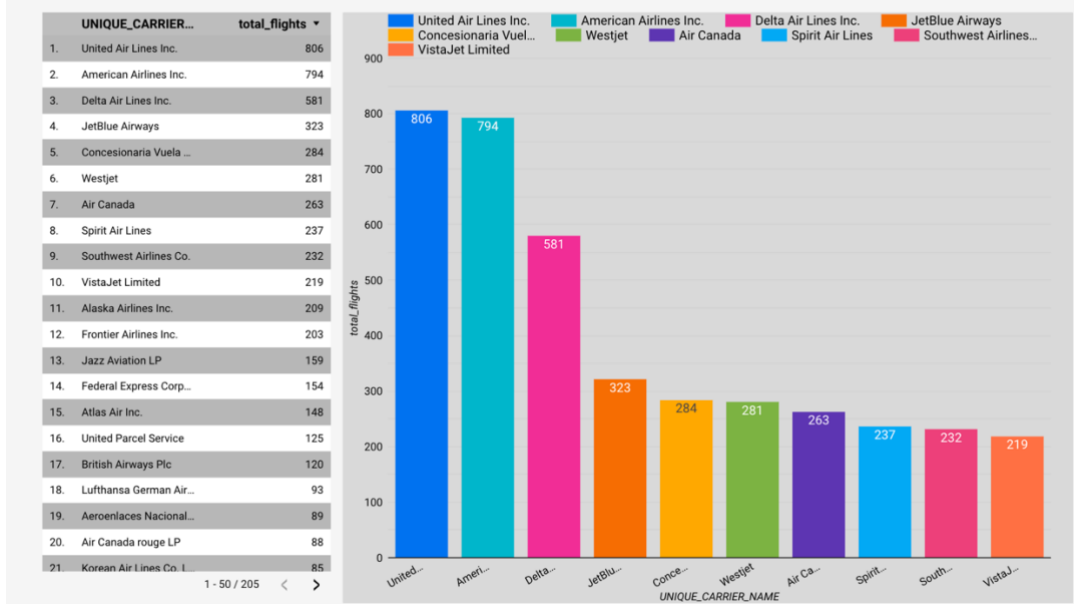
```

Query results

JOB INFORMATION	RESULTS	CHART	JSON	EXECUTION DETAILS
Row	city ▼	total_count ▼		
1	Miami, FL	956		
2	Los Angeles, CA	641		
3	New York, NY	639		
4	Toronto, Canada	496		
5	Cancun, Mexico	466		
6	Chicago, IL	439		
7	Houston, TX	430		
8	Fort Lauderdale, FL	342		
9	Atlanta, GA	313		
10	Newark, NJ	305		

IMG 6: SQL Query to select busiest airport

## Top 10 airlines by total number of flights



IMG 7 : Looker Studio Report of top 10 airlines by total number of flights

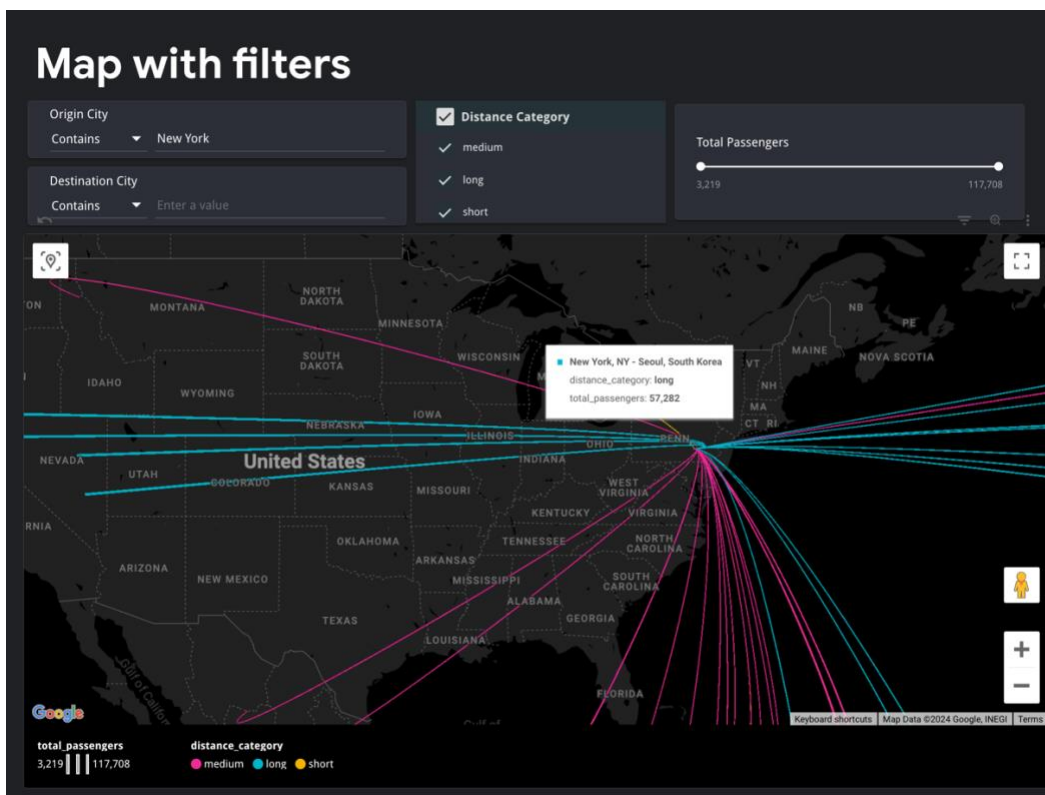
## Destination City Heat Map



IMG 8 : Looker Studio Report of Destination City Heatmap

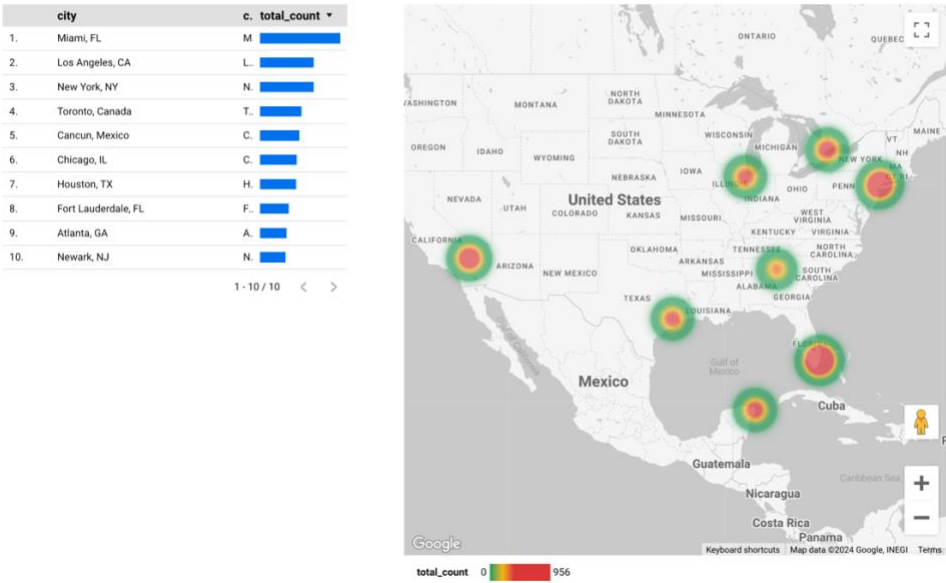


IMG 9 : Looker Studio Report of a map with filters showing all the flights with their distance category



IMG 10 : Looker Studio Report of a map with origin country set as new York showing total no. of passengers

# Top 10 Busiest Airport



IMG 11 : Looker Studio Report of top 10 busiest airport with heatmap

## Discussion

By employing cloud-based infrastructure and distributed computing frameworks, this project showcased how we extract insights from the data using SQL queries as well as interactive dashboards such as Looker Studio. From the amount of analysis, I can interpret that Miami, FL is the busiest airport for 2024 for international flights, with Bogota, Colombia to Miami, FL and Miami, FL to Bogota, Colombia being the routes with the most number of flights. I can also make out that Qatar Airways handles the most amount of passengers than any other airline as far as international flights related to the US are concerned, in contrast, United Airlines is the airline with the most number of international flights. Looking at the destination heat map report, we can see that many flights go to different places worldwide, with Mexico, Germany, and the United Kingdom being some of the most popular spots. Additionally, I noticed something interesting: no flights are operating directly to Ukraine, Russia, and their neighboring countries, likely due to current tensions and sanctions. On the other hand, I observed a lot of flights going to the islands west of the US, possibly around Hawaii.

In this project, I applied several technologies and skills learned from the course to effectively design and handle a data pipeline to help analyze international flight data. I utilized cloud-based infrastructure which was an integral part of the course with a couple of labs revolving around the Google Cloud Platform which indeed taught me how to use Dataproc as a tool to leverage concepts from distributed storage and computing.

I did face a lot of challenges while setting up this data pipeline, the failures began from not being able to create and provision a dataproc cluster because of multiple reasons that took a lot of time and effort. I spent more time correcting those errors rather than analyzing the data. Problems began when I could not even create a simple instance because the default vpc that is usually present in startup labs was missing. After all, the project was under the IU organization and the default vpc option was turned off because of security purposes and it took a lot of time to figure things out.

But, even after that the challenges didn't end, as I was constantly facing an error saying the firewall rules for specified network or subnetwork would likely not permit sufficient communication in the network or subnetwork for Dataproc to function properly, and even after following all the steps in the guide document I couldn't find a solution to this problem for a very long time.

In the end, I managed to overcome this problem by modifying the firewall rule to allow specific ports on TCP, UDP and ICMP as specified by the [Dataproc networking document](#).

After finally getting my cluster up and running, I tested a demo Python code, which worked seamlessly. However, when I ran my actual code, I encountered a strange error with the message "The billing account for the owning project is disabled in state closed." It turned out that the error was not related to billing at all; in fact, it was simply due to an error in writing the correct bucket name. However, the screenshot I've provided clearly illustrates my struggle with this misleading error message.

Name

cluster-37d3

Cluster UUID

404bee3b-4862-4a30-bbb1-90a5cb965437

Type

Dataproc Cluster

Status

Stopped

MONITORING

JOBS

VM INSTANCES

CONFIGURATION

WEB INTERFACES

Filter

Filter jobs

Job ID	Status	Region	Type	Start time	Elapsed time	Labels	
<a href="#">job-f3f43b35</a>	<div><div></div>Succeeded</div>	us-east1	PySpark	Apr 26, 2024, 11:09:42 PM	1 min 2 sec	None	<div><div></div></div>
<a href="#">job-721adb8</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 11:01:44 PM	33 sec	None	<div><div></div></div>
<a href="#">job-6bd7a8b0</a>	<div><div></div>Succeeded</div>	us-east1	PySpark	Apr 26, 2024, 10:58:45 PM	39 sec	None	<div><div></div></div>
<a href="#">job-10bc4d50</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 10:55:43 PM	28 sec	None	<div><div></div></div>
<a href="#">job-5fe87350-all-data</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 10:52:47 PM	30 sec	None	<div><div></div></div>
<a href="#">All_data_etl</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 10:48:39 PM	29 sec	None	<div><div></div></div>
<a href="#">job-824281f2</a>	<div><div></div>Succeeded</div>	us-east1	PySpark	Apr 26, 2024, 5:10:42 AM	34 sec	None	<div><div></div></div>
<a href="#">job-e8310c44</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 5:09:18 AM	34 sec	None	<div><div></div></div>
<a href="#">job-932ae0d5</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 5:04:18 AM	34 sec	None	<div><div></div></div>
<a href="#">job-4f500597</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 5:02:06 AM	32 sec	None	<div><div></div></div>
<a href="#">job-53b9d004</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 4:59:12 AM	27 sec	None	<div><div></div></div>
<a href="#">job-07c817ff</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 4:56:23 AM	26 sec	None	<div><div></div></div>
<a href="#">job-926a764a</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 4:51:11 AM	34 sec	None	<div><div></div></div>
<a href="#">job-a6015c59</a>	<div><div></div>Failed</div>	us-east1	PySpark	Apr 26, 2024, 4:48:00 AM	30 sec	None	<div><div></div></div>
<a href="#">job-bf1f9367</a>	<div><div></div>Succeeded</div>	us-east1	PySpark	Apr 26, 2024, 4:36:48 AM	27 sec	None	<div><div></div></div>

IMG 10 : Screenshot of failed jobs

## Conclusion

To conclude, this project was all about using my love for aviation and the skills I acquired in my coursework, focusing on analyzing the Bureau of Transportation Statistics International Flight dataset using advanced services on the Google Cloud Platform, leveraging distributed storage and processing capabilities I utilized Dataproc and BigQuery to handle the large dataset with ease. I ensured the dataset's integrity and relevancy for further analysis by applying various validation checks and transformations like restructuring and cleaning. The data architecture followed in this project involved designing scalable data pipelines using cloud-based services for storage, processing, warehousing and visualization. Additionally, I am maintaining a well-documented **repository** that contains all the code files along with links to the Looker Studio reports. Through this project, I not only uncovered valuable insights into international flight patterns but also demonstrated and utilized the power of cloud-based/virtual machine technologies in modern data science.

## References

- [1] GitHub repository [link](#)
- [2] Dataproc Qwik Start lab [link](#)
- [2] Dataproc cluster network configuration [link](#)
- [3] Dataproc-BigQuery connector [link](#)
- [4] Debugging – stackoverflow [link](#)
- [4] BigQuery for data analysis [link](#)
- [5] BigQuery to Looker Studio [link](#)