

Case Study 1

Amazon Books Catalog Search

Data Cleansing, Analytics, Pipeline & Reporting using Apache Beam, Python pySpark on a Google Data Proc Cluster.

PROBLEM STATEMENT:

1. Find the average user rating for Fiction books in pyspark.
2. Find the average user rating for Non-fiction books and for all books together. Please write the result in separate files. – python (analytics), PySpark
3. Sort the last 5 books in alphabetical order and print the output – Pyspark

Kaggle Dataset - <https://www.kaggle.com/sootersaalu/amazon-top-50-bestselling-books-2009-2019>

Notes - This is a dataset on Amazon's Top 50 bestselling books from 2009 to 2019., it contains 550+ books, and has been categorized into fiction and non-fiction using Goodreads.

[Note, one can add/multiple the data set to about 5000+] books to utilize power of GCP Cluster, Analytics, Spark Job, BigQuery, BigTable capabilities.

Implementation Steps -

- **Pre-requisite** – One should download this csv and do data cleaning (replace the commas in the Name column with semicolons). [Upload this csv in the GCP bucket] – Pyspark, Hive [10% credits]
- One has to build a DataProc Cluster [at least 2 nodes] to run Python pySpark batch jobs [Kaggle dataset, from GCP bucket] on Google Cloud. – DataProc [15% credits]
- Advisable to connect to the nodes using CLI and provision the VMs with required configurations using [15% credits] o GCP Cloud SDK o gsutil tool
- **Create/Configure a pipeline** [10% credits] using Command Line
- Pipeline = Reading input + Transforming Data + Writing Output
- Output results should be stored as raw data in Hive[**BigQuery**] before stored finally into HBase[BigTable] for further processing & reporting to Analytics Engines. – Hbase, Hive [15% Credits]

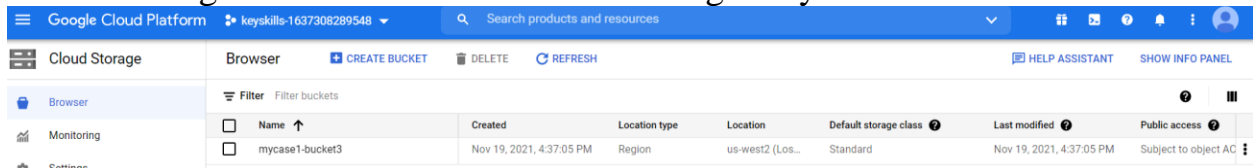
- **Run the pipeline – Apache Beam [10% credit]**
- Print the results to console and store for further analytics – Hbase [10% credits]
- Extract of the pipeline execution and output is here – (good to use HBase) – [10% credits]
- Engineering Best Practices to be followed [30% credits]

1. Logging has to be implemented in all the Python Components
2. Exception handling framework need to be implemented
3. Unit testing of each Python component/service using Pytest/UnitTest/Mock Objects
4. Unit Testing of Spark Cluster, Data Pipeline [is desirable]
5. Usage of 4 to 5 Big Data Design Patterns using Python [is desirable]

Answer:

Bucket creation using CLI:

- `gsutil mb -c standard -l us-west2 gs://mycase1-bucket3`



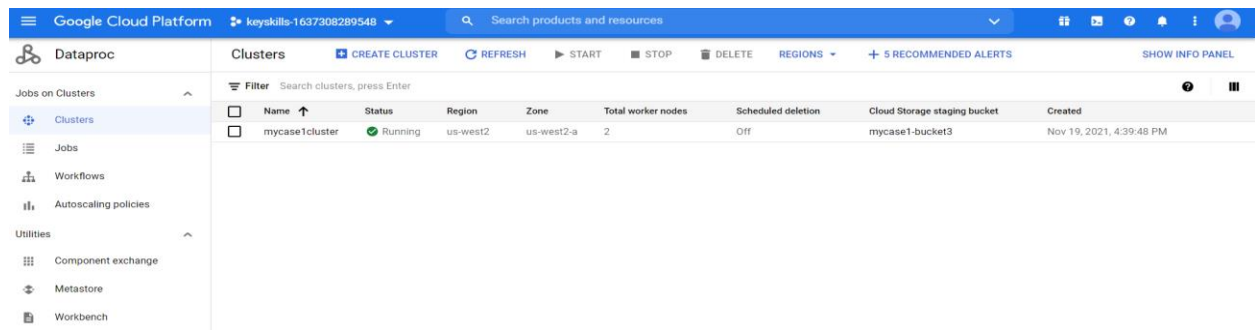
Name	Created	Location type	Location	Default storage class	Last modified	Public access
mycase1-bucket3	Nov 19, 2021, 4:37:05 PM	Region	us-west2 (Los...	Standard	Nov 19, 2021, 4:37:05 PM	Subject to object AC

Cluster creation using CLI:

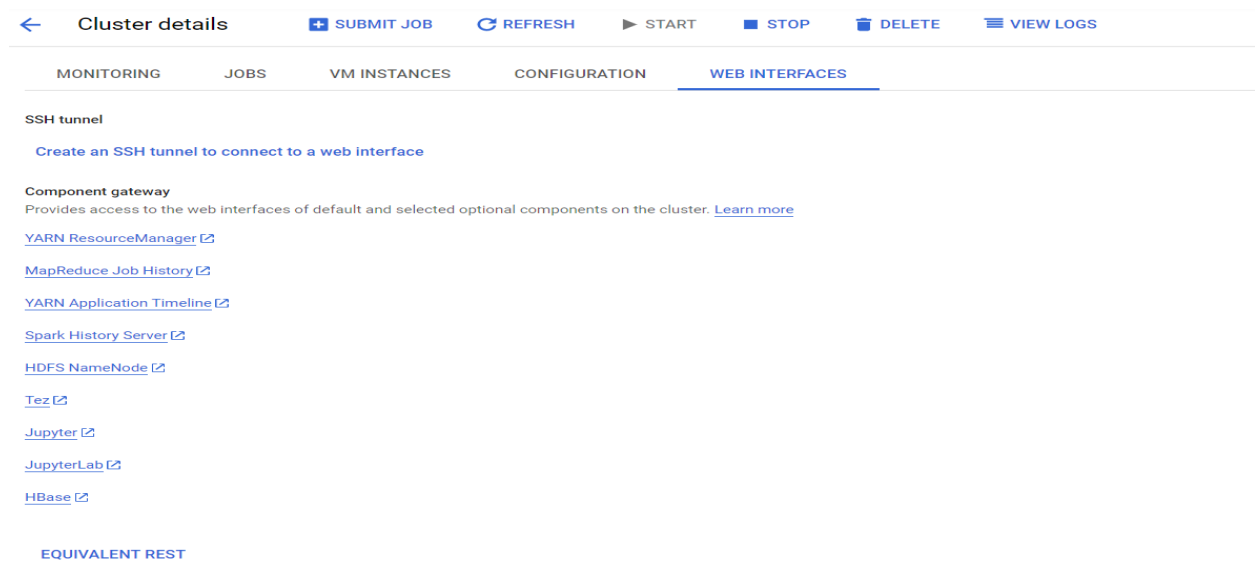
- `gcloud dataproc clusters create mycase1cluster --region us-west2 --zone us-west2-a --master-machine-type n1-standard-2 --master-boot-disk-size 50 --num-workers 2 --worker-machine-type n1-standard-2 --worker-boot-disk-size 50 --image-version 2.0-ubuntu18 --optional-components JUPYTER,ZOOKEEPER,HBASE --bucket mycase1-bucket3 --enable-component-gateway --project keyskills-1637308289548`

```
kshgcp_userid01@cloudshell:~ (keyskills-1637308289548) $ gsutil mb -c standard -l us-west2 gs://mycase1-bucket3
Creating gs://mycase1-bucket3/...
kshgcp_userid01@cloudshell:~ (keyskills-1637308289548) $ gcloud dataproc clusters create mycase1cluster --region us-west2 --zone us-west2-a --master-machine-type n1-standard-2 --master-boot-disk-size 50 --num-workers 2 --worker-machine-type n1-standard-2 --worker-boot-disk-size 50 --image-version 2.0-ubuntu18 --optional-components JUPYTER,ZOOKEEPER,HBASE --bucket mycase1-bucket3 --enable-component-gateway --project keyskills-1637308289548
API [dataproc.googleapis.com] not enabled on project [397076816615]. Would you like to enable and retry (this will take a few minutes)? (y/N)? y
Enabling service [dataproc.googleapis.com] on project [397076816615]...
Operation "operations/acf.p2-397076816615-2b754e45-0572-4661-98c0-3ba3e49bd296" finished successfully.
Waiting on operation [projects/keyskills-1637308289548/regions/us-west2/operations/4559a072-a397-3296-9696-c082b97b0607].
Waiting for cluster creation operation...
WARNING: 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.
Waiting for cluster creation operation...done.
Created [https://dataproc.googleapis.com/v1/projects/keyskills-1637308289548/regions/us-west2/clusters/mycase1cluster] Cluster placed in zone [us-west2-a].
kshgcp_userid01@cloudshell:~ (keyskills-1637308289548) $
```

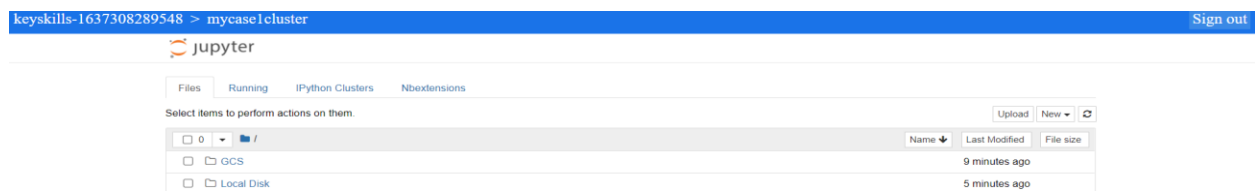
Cluster Created Using Dataproc



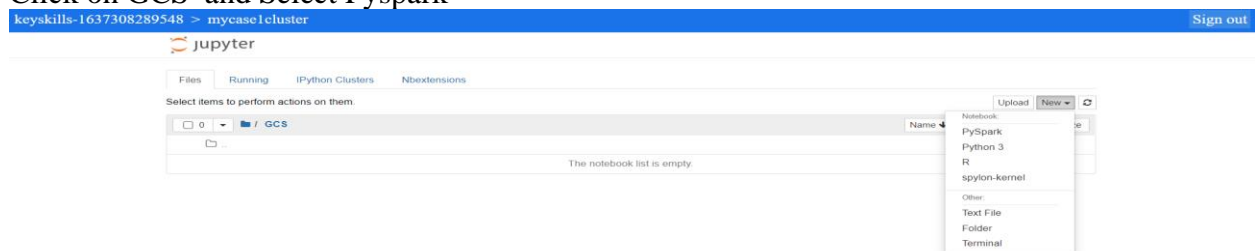
Cluster has all the web interfaces we opted



Click on Jupyter



Click on GCS and Select Pyspark



Upload the kaggle data set into the GCS bucket and copy its path

← Bucket details REFRESH HELP ASSISTANT LEARN

mycase1-bucket3

Location: us-west2 (Los Angeles) Storage class: Standard Public access: Subject to object ACLs Protection: None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > mycase1-bucket3

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter objects and folders Show deleted data

<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention
<input type="checkbox"/>	bestsellers with categories.csv	50 KB	text/csv	Nov 19, 2...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—	—

Upload the csv and perform data cleaning i.e replace the commas in the Name column with semicolons:

- import pyspark
- from pyspark.sql import SparkSession
- spark = SparkSession.builder.appName('temp').getOrCreate()
- df = spark.read.csv("gs://mycase1-bucket3/bestsellers with categories.csv", header=True, inferSchema = True)
- df.show(truncate=False)

```
File Edit View Insert Cell Kernel Widgets Help Trusted | PySpark
```

```
In [1]: import pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('temp').getOrCreate()
df = spark.read.csv("gs://mycase1-bucket3/bestsellers with categories.csv", header=True, inferSchema = True)
df.show(truncate=False)
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|Name|User Rating|Reviews|Price|Year|Genre|Author|
+-----+-----+-----+-----+-----+-----+-----+
|10-Day Green Smoothie Cleanse|4.7|17350|8|2016|Non Fiction|JJ Smith|
|11/22/63: A Novel King|4.6|2052|22|2011|Fiction|Stephen|
|12 Rules for Life: An Antidote to Chaos|4.7|18979|15|2018|Non Fiction|Jordan|
|B. Peterson|4.7|18979|15|2018|Non Fiction|Jordan|
|1984 (Signet Classics)|4.7|21424|6|2017|Fiction|George O|
|5,000 Awesome Facts (About Everything!) (National Geographic Kids)|4.8|7665|12|2019|Non Fiction|National|
|Geographic Kids|4.8|7665|12|2019|Non Fiction|National|
|A Dance with Dragons (A Song of Ice and Fire)|4.4|12643|11|2011|Fiction|George|
|R. R. Martin|4.4|12643|11|2011|Fiction|George|
|A Game of Thrones / A Clash of Kings / A Storm of Swords / A Feast of Crows / A Dance with Dragons|4.7|19735|30|2014|Fiction|George|
|R. R. Martin|4.7|19735|30|2014|Fiction|George|
|A Gentleman in Moscow: A Novel|4.7|19699|15|2017|Fiction|Amor Tow|
|les|4.7|19699|15|2017|Fiction|Amor Tow|
|A Higher Loyalty: Truth, Lies, and Leadership|4.7|5983|3|2018|Non Fiction|James Co|
|mey|4.7|5983|3|2018|Non Fiction|James Co|
|A Man Called Ove: A Novel|4.6|23848|8|2016|Fiction|Fredrik|
|Backman|4.6|23848|8|2016|Fiction|Fredrik|
|A Man Called Ove: A Novel|4.6|23848|8|2017|Fiction|Fredrik|
|Backman|4.6|23848|8|2017|Fiction|Fredrik|
|A Patriot's History of the United States: From Columbus's Great Discovery to the War on Terror|4.6|460|2|2010|Non Fiction|Larry Sc|
|hweikart|4.6|460|2|2010|Non Fiction|Larry Sc|
|A Stolen Life: A Memoir|4.6|4149|32|2011|Non Fiction|Jaycee D|
|ugard|4.6|4149|32|2011|Non Fiction|Jaycee D|
|A Wrinkle in Time (Time Quintet)|4.5|5153|5|2018|Fiction|Madelein|
|e L'Engle|4.5|5153|5|2018|Fiction|Madelein|
|Act Like a Lady, Think Like a Man: What Men Really Think About Love, Relationships, Intimacy, and Commitment|4.5|5153|5|2018|Fiction|Steve Ha|
```

- from pyspark.sql.functions import translate

- `df = df.withColumn('Name', translate('Name', ',', ';'))`
- `df.show(truncate=False)`

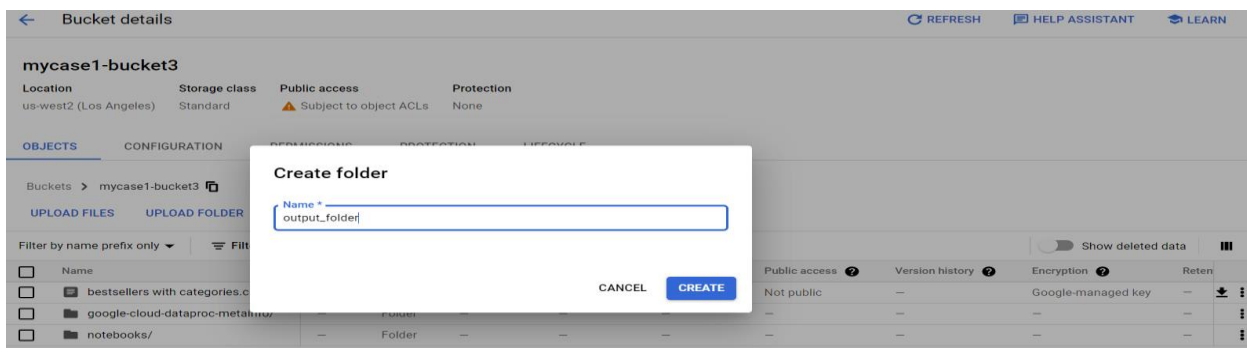
```
File Edit View Insert Cell Kernel Widgets Help Trusted | PySpark O
```

```
In [2]: from pyspark.sql.functions import translate
df = df.withColumn('Name', translate('Name', ',', ';'))
df.show(truncate=False)
```

Name	User Rating	Reviews	Price	Year	Genre	Author
10-Day Green Smoothie Cleanse	4.7	17350	8	2016	Non Fiction	JJ Smith
11/22/63: A Novel	4.6	12643	11	2011	Fiction	Stephen King
12 Rules for Life: An Antidote to Chaos	4.7	18979	15	2018	Non Fiction	Jordan B. Peterson
1984 (Signet Classics)	4.7	21424	6	2017	Fiction	George Orwell
5,000 Awesome Facts (About Everything!) (National Geographic Kids)	4.8	7665	12	2019	Non Fiction	National Geographic Kids
A Dance with Dragons (A Song of Ice and Fire)	4.4	12643	11	2011	Fiction	George R. R. Martin
A Game of Thrones / A Clash of Kings / A Storm of Swords / A Feast of Crows / A Dance with Dragons	4.7	19735	30	2014	Fiction	George R. R. Martin
A Gentleman in Moscow: A Novel	4.7	19699	15	2017	Fiction	Amor Towles
A Higher Loyalty: Truth, Lies, and Leadership	4.7	5983	3	2018	Non Fiction	James Comey
A Man Called Ove: A Novel	4.6	23848	8	2016	Fiction	Fredrik Backman
A Man Called Ove: A Novel	4.6	23848	8	2017	Fiction	Fredrik Backman
A Patriot's History of the United States: From Columbus's Great Discovery to the War on Terror	4.6	460	2	2010	Non Fiction	Larry Schweikart
A Stolen Life: A Memoir	4.6	4149	32	2011	Non Fiction	Jaycee Dugard
A Wrinkle in Time (Time Quintet)	4.5	5153	5	2018	Fiction	Madeleine L'Engle
Act Like a Lady, Think Like a Man: What Men Really Think About Love, Relationships, Intimacy, and Commitment	4.6	5013	17	2009	Non Fiction	Steve Harvey
Adult Coloring Book Designs: Stress Relief Coloring Book: Garden Designs; Mandalas; Animals; and Paisley Patterns	4.5	2313	4	2016	Non Fiction	Adult Coloring Book Designs

1. Finding average of user ratings for Genre = Fiction and saving the file in output_folder :

- `df_fiction = df.filter("Genre == 'Fiction'")`
- `df_fiction.show(10)`
- `fiction_avg = df_fiction.agg({'User Rating': 'avg'})`
- `fiction_avg.show()`
- `fiction_avg.write.save("gs://mycase1-bucket3/output_folder/fiction_avg.csv", format = 'csv', header = True)`



```
In [3]: df_fiction = df.filter("Genre == 'Fiction'")
df_fiction.show(10)
fiction_avg = df_fiction.agg({'User Rating': 'avg'})
fiction_avg.show()
fiction_avg.write.save("gs://mycase1-bucket3/output_folder/fiction_avg.csv", format = 'csv', header = True)
```

Name	Author	User Rating	Reviews	Price	Year	Genre
11/22/63: A Novel	Stephen King	4.6	2052	22	2011	Fiction
1984 (Signet Clas...	George Orwell	4.7	21424	6	2017	Fiction
A Dance with Drag...	George R. R. Martin	4.4	12643	11	2011	Fiction
A Game of Thrones...	George R. R. Martin	4.7	19735	30	2014	Fiction
A Gentleman in Mo...	Amor Towles	4.7	19699	15	2017	Fiction
A Man Called Ove:...	Fredrik Backman	4.6	23848	8	2016	Fiction
A Man Called Ove:...	Fredrik Backman	4.6	23848	8	2017	Fiction
A Wrinkle in Time...	Madeleine L'Engle	4.5	5153	5	2018	Fiction
All the Light We ...	Anthony Doerr	4.6	36348	14	2014	Fiction
All the Light We ...	Anthony Doerr	4.6	36348	14	2015	Fiction

only showing top 10 rows

```
-----+
| avg(User Rating)|
-----+
|4.648333333333326|
-----+
```

2. Finding average of user ratings for Genre =Non Fiction and saving the file in output_folder :

- `df_non_fiction = df.filter("Genre == 'Non Fiction'")`
- `df_non_fiction.show(10)`
- `non_fiction_avg = df_non_fiction.agg({'User Rating': 'avg'})`
- `non_fiction_avg.show()`
- `non_fiction_avg.write.save("gs://mycase1-bucket3/output_folder/non_fiction_avg.csv", format = 'csv', header = True)`

```
In [4]: df_non_fiction = df.filter("Genre == 'Non Fiction'")
df_non_fiction.show(10)
non_fiction_avg = df_non_fiction.agg({'User Rating': 'avg'})
non_fiction_avg.show()
non_fiction_avg.write.save("gs://mycase1-bucket3/output_folder/non_fiction_avg.csv", format = 'csv', header = True)
```

Name	Author	User Rating	Reviews	Price	Year	Genre
10-Day Green Smoo...	JJ Smith	4.7	17350	8	2016	Non Fiction
12 Rules for Life...	Jordan B. Peterson	4.7	18979	15	2018	Non Fiction
5,000 Awesome Fac...	National Geograph...	4.8	7665	12	2019	Non Fiction
A Higher Loyalty:...	James Comey	4.7	5983	3	2018	Non Fiction
A Patriot's Histo...	Larry Schweikart	4.6	460	2	2010	Non Fiction
A Stolen Life: A ...	Jaycee Dugard	4.6	4149	32	2011	Non Fiction
Act Like a Lady; ...	Steve Harvey	4.6	5013	17	2009	Non Fiction
Adult Coloring Bo...	Adult Coloring Bo...	4.5	2313	4	2016	Non Fiction
Adult Coloring Bo...	Blue Star Coloring	4.6	2925	6	2015	Non Fiction
Adult Coloring Bo...	Blue Star Coloring	4.4	2951	6	2015	Non Fiction

only showing top 10 rows

```
-----+
| avg(User Rating)|
-----+
|4.595161290322579|
-----+
```

Finding average of user ratings for all the books and saving the file in output_folder :

- `avg_of_all = df.agg({'User Rating': 'avg'})`

- avg_of_all.show()
- avg_of_all.write.save("gs://mycase1-bucket3/output_folder/avg_of_all.csv", format = 'csv', header = True)

```
In [5]: avg_of_all = df.agg({'User Rating': 'avg'})
avg_of_all.show()
avg_of_all.write.save("gs://mycase1-bucket3/output_folder/avg_of_all.csv", format = 'csv', header = True)
```

```
+-----+
| avg(User Rating)|
+-----+
|4.6183636363641|
+-----+
```

3. Sorting the last 5 books in alphabetical order and printing the output:

- df.orderBy(df.Name.desc()).limit(5).show()
- df.write.csv('gs://mycase1-bucket3/output_folder/cleaned_books_amazon.csv')

```
In [6]: df.orderBy(df.Name.desc()).limit(5).show()
df.write.csv('gs://mycase1-bucket3/output_folder/cleaned_books_amazon.csv')
```

```
+-----+-----+-----+-----+-----+-----+-----+
|          Name|   Author|User Rating|Reviews|Price|Year|   Genre|
+-----+-----+-----+-----+-----+-----+-----+
|You Are a Badass:...|Jen Sincero|      4.7| 14331|  8|2018|Non Fiction|
|You Are a Badass:...|Jen Sincero|      4.7| 14331|  8|2016|Non Fiction|
|You Are a Badass:...|Jen Sincero|      4.7| 14331|  8|2017|Non Fiction|
|You Are a Badass:...|Jen Sincero|      4.7| 14331|  8|2019|Non Fiction|
|Wrecking Ball (Di...|Jeff Kinney|      4.9|  9413|  8|2019|  Fiction|
+-----+-----+-----+-----+-----+-----+-----+
```

Output folder with all the output files:

mycase1-bucket3

Location	Storage class	Public access	Protection
us-west2 (Los Angeles)	Standard	⚠ Subject to object ACLs	None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > mycase1-bucket3 > output_folder 📁

[UPLOAD FILES](#) [UPLOAD FOLDER](#) [CREATE FOLDER](#) [MANAGE HOLDS](#) [DOWNLOAD](#) [DELETE](#)

Filter by name prefix only ▼ **Filter** filter objects and folders Show deleted data ⌵

<input type="checkbox"/>	Name	Size	Type	Created ?	Storage class	Last modified	Public access ?	Version history ?	Encryption ?	Retention expiration
<input type="checkbox"/>	avg_of_all.csv/	—	Folder	—	—	—	—	—	—	⋮
<input type="checkbox"/>	cleaned_books_amazon.csv/	—	Folder	—	—	—	—	—	—	⋮
<input type="checkbox"/>	fiction_avg.csv/	—	Folder	—	—	—	—	—	—	⋮
<input type="checkbox"/>	non_fiction_avg.csv/	—	Folder	—	—	—	—	—	—	⋮

Job Running:

Save the Pyspark notebook file as clean_n_store.py and upload it in the mycase1-bucket3 bucket.

mycase1-bucket3

Location	Storage class	Public access	Protection
us-west2 (Los Angeles)	Standard	Subject to object ACLs	None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > mycase1-bucket3

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter objects and folders

Show deleted data

<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention	Actions
<input type="checkbox"/>	bestsellers with categories.csv	50 KB	text/csv	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—	
<input type="checkbox"/>	clean_n_store.py	1.3 KB	text/plain	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—	
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—	—	
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—	—	
<input type="checkbox"/>	output_folder/	—	Folder	—	—	—	—	—	—	—	

Running CLI command for Dataproc job:

- gcloud dataproc jobs submit pyspark gs://mycase1-bucket3/clean_n_store.py --cluster=mycase1cluster --region=us-west2

Dataproc

Jobs on Clusters

- Clusters
- Jobs**
- Workflows
- Autoscaling policies

Utilities

- Component exchange
- Metastore
- Workbench

Jobs

SUBMIT JOB REFRESH STOP DELETE REGIONS + 2 RECOMMENDED ALERTS SHOW INFO PANEL

Filter Filter jobs

<input type="checkbox"/>	Job ID	Status	Region	Type	Cluster	Start time	Elapsed time	Labels
<input type="checkbox"/>	a339cf73656c403485b738345ed88792	Succeeded	us-west2	PySpark	mycase1cluster	Nov 19, 2021, 5:04:03 PM	54 sec	None

Outputs obtained from Dataproc job:

Average of Fiction:

Job details

CLONE DELETE STOP REFRESH

Job ID a339cf73656c403485b738345ed88792

Job UUID cf8f58d3-7c80-311f-bc82-6fd17105c0d2

Type Dataproc Job

Status Succeeded

Output

LINE WRAP: OFF

```
|-----|
|      Name      | Author | User Rating | Reviews | Price | Year | Genre |
|-----|-----|-----|-----|-----|-----|-----|
| 11/22/63: A Novel | Stephen King | 4.6 | 2052 | 22 | 2011 | Fiction |
| 1984 (Signet Clas... | George Orwell | 4.7 | 21424 | 6 | 2017 | Fiction |
| A Dance with Drag... | George R. R. Martin | 4.4 | 12643 | 11 | 2011 | Fiction |
| A Game of Thrones... | George R. R. Martin | 4.7 | 19735 | 30 | 2014 | Fiction |
| A Gentleman in Mo... | Amor Towles | 4.7 | 19699 | 15 | 2017 | Fiction |
| A Man Called Ove:... | Fredrik Backman | 4.6 | 23848 | 8 | 2016 | Fiction |
| A Man Called Ove:... | Fredrik Backman | 4.6 | 23848 | 8 | 2017 | Fiction |
| A Wrinkle in Time... | Madeleine L'Engle | 4.5 | 5153 | 5 | 2018 | Fiction |
| All the Light We ... | Anthony Doerr | 4.6 | 36348 | 14 | 2014 | Fiction |
| All the Light We ... | Anthony Doerr | 4.6 | 36348 | 14 | 2015 | Fiction |
|-----|-----|-----|-----|-----|-----|
only showing top 10 rows

+-----+
| avg(User Rating) |
+-----+
| 4.64833333333326 |
+-----+
```


Average of Non-Fiction:

Output

LINE WRAP: OFF

Name	Author	User Rating	Reviews	Price	Year	Genre
10-Day Green Smoo...	JJ Smith	4.7	17350	8	2016	Non Fiction
12 Rules for Life...	Jordan B. Peterson	4.7	18979	15	2018	Non Fiction
5,000 Awesome Fac...	National Geograph...	4.8	7665	12	2019	Non Fiction
A Higher Loyalty...	James Comey	4.7	5983	3	2018	Non Fiction
A Patriot's Histo...	Larry Schweikart	4.6	460	2	2010	Non Fiction
A Stolen Life: A ...	Jaycee Dugard	4.6	4149	32	2011	Non Fiction
Act Like a Lady; ...	Steve Harvey	4.6	5013	17	2009	Non Fiction
Adult Coloring Bo...	Adult Coloring Bo...	4.5	2313	4	2016	Non Fiction
Adult Coloring Bo...	Blue Star Coloring	4.6	2925	6	2015	Non Fiction
Adult Coloring Bo...	Blue Star Coloring	4.4	2951	6	2015	Non Fiction

only showing top 10 rows

```
-----+
| avg(User Rating)|
+-----+
| 4.595161290322579|
+-----+
```

Average of all:

Output

LINE WRAP: OFF

```
21/11/19 11:34:48 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageFileSystem: Successfully repaired 'gs://mycase1-bucket3/output_folder/non_fict
| avg(User Rating)|
+-----+
| 4.6183636363641|
+-----+
```

```
21/11/19 11:34:51 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageFileSystem: Successfully repaired 'gs://mycase1-bucket3/output_folder/avg_of_i
+-----+
| Name| Author|User Rating|Reviews|Price|Year| Genre|
+-----+
|You Are a Badass:...|Jen Sincero| 4.7| 14331| 8|2018|Non Fiction|
|You Are a Badass:...|Jen Sincero| 4.7| 14331| 8|2016|Non Fiction|
|You Are a Badass:...|Jen Sincero| 4.7| 14331| 8|2017|Non Fiction|
|You Are a Badass:...|Jen Sincero| 4.7| 14331| 8|2019|Non Fiction|
|Wrecking Ball (Di...|Jeff Kinney| 4.9| 9413| 8|2019| Fiction|
+-----+
```

```
21/11/19 11:34:55 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageFileSystem: Successfully repaired 'gs://mycase1-bucket3/output_folder/cleaned_
21/11/19 11:34:55 INFO org.sparkproject.jetty.server.AbstractConnector: Stopped Spark@3fee8cd1(HTTP/1.1, (http/1.1)){0.0.0.0:0}
```

Output files in the output folder:

mycase1-bucket3

Location

Storage class

Public access

Protection

us-west2 (Los Angeles)

Standard

Subject to object ACLs

None

OBJECTS

CONFIGURATION

PERMISSIONS

PROTECTION

LIFECYCLE

Buckets > mycase1-bucket3 > output_folder

UPLOAD FILES

UPLOAD FOLDER

CREATE FOLDER

MANAGE HOLDS

DOWNLOAD

DELETE

Filter by name prefix only

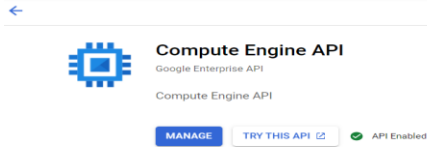
Filter objects and folders

Show deleted data

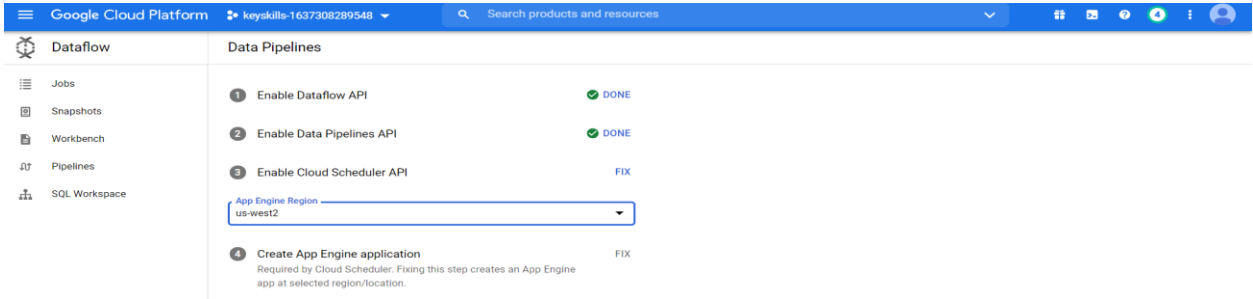
Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention expiration
<input type="checkbox"/> avg_of_all.csv/	—	Folder	—	—	—	—	—	—	—
<input checked="" type="checkbox"/> avg_of_all.csv2/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/> cleaned_books_amazon.csv/	—	Folder	—	—	—	—	—	—	—
<input checked="" type="checkbox"/> cleaned_books_amazon2.csv/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/> fiction_avg.csv/	—	Folder	—	—	—	—	—	—	—
<input checked="" type="checkbox"/> fiction_avg2.csv/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/> non_fiction_avg.csv/	—	Folder	—	—	—	—	—	—	—
<input checked="" type="checkbox"/> non_fiction_avg2.csv/	—	Folder	—	—	—	—	—	—	—

Create/Configure a pipeline:

Enable the compute engine API:

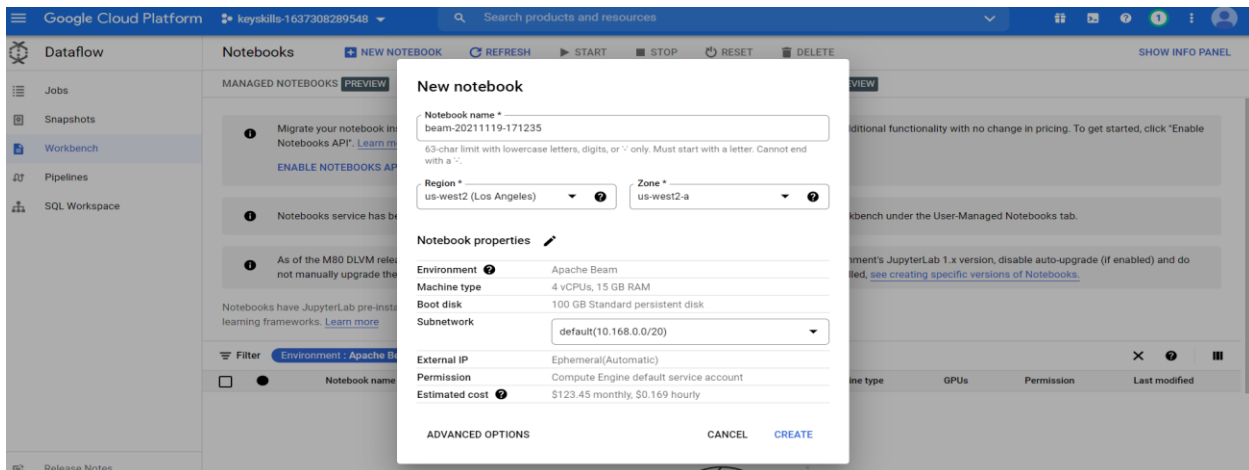
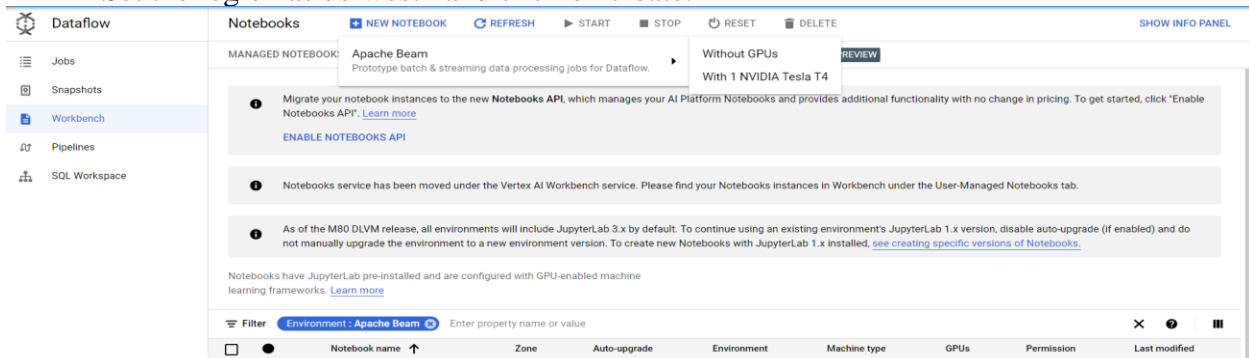


Enable the Dataflow API:



Configuring the Data Pipeline:

- Go to dataflow, click on workbench.
- Click on new notebook → Apache Beam → Without GPUs.
- Set the region as us-west2 and click on create.



Click on open Jupyter lab:

Filter

Environment : Apache Beam

Enter property name or value

X ?

<input type="checkbox"/>	<input checked="" type="checkbox"/>	Notebook name ↑	Zone	Auto-upgrade	Environment	Machine type	GPUs	Permission	Last modified
<input type="checkbox"/>	<input checked="" type="checkbox"/>	beam-20211119-171235	OPEN JUPYTERLAB	us-west2-a	—	Apache Beam	4 vCPUs, 15 GB RAM	None	Service account

Before running the pipeline code we need to create a BigQuery view of the given dataset.

- Download the cleaned dataset file obtained in the output_folder after running the PySpark commands in notebook.
- Upload it in the mycase1-bucket.

← Bucket details REFRESH HELP ASSISTANT LEARN

mycase1-bucket3

Location: us-west2 (Los Angeles) Storage class: Standard Public access: Subject to object ACLs Protection: None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > mycase1-bucket3 📁

[UPLOAD FILES](#) [UPLOAD FOLDER](#) [CREATE FOLDER](#) [MANAGE HOLDS](#) [DOWNLOAD](#) [DELETE](#)

Filter by name prefix only ▼ Filter Filter objects and folders Show deleted data

<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention
<input type="checkbox"/>	.ipynb_checkpoints/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/>	bestsellers with categories.csv	50 KB	text/csv	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	clean_n_store.py	1.3 KB	text/plain	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	cleaned_books_amazon.csv	49.2 KB	text/csv	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—	—
<input type="checkbox"/>	output_folder/	—	Folder	—	—	—	—	—	—	—

- Go to BigQuery and create a dataset with dataset id case1_amz.
- Create table view for bestsellers with categories.csv file. Here the table name is set at fic1.

🔍 FEATURES & INFO SHORTCUT DISABLE EDITOR TABS

Explorer + ADD DATA

🔍 Type to search

Viewing pinned projects.

keyskills-1637308289548

case1_amz

fic1 MORE RESULTS

COMPOSE NEW QUERY

fic1 QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW

Table schema

Filter Enter property name or value

Field name	Type	Mode	Policy Tags	Description
Name	STRING	NULLABLE		
Author	STRING	NULLABLE		
User_Rating	FLOAT	NULLABLE		
Reviews	INTEGER	NULLABLE		
Price	INTEGER	NULLABLE		
Year	INTEGER	NULLABLE		
Genre	STRING	NULLABLE		

[EDIT SCHEMA](#) [VIEW ROW ACCESS POLICIES](#)

COMPOSE NEW QUERY

fic1 QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS **PREVIEW**

Row	Name	Author	User_Rating	Reviews	Price	Year	Genre
1	Gone Girl	Gillian Flynn	4.0	57271	10	2012	Fiction
2	Gone Girl	Gillian Flynn	4.0	57271	10	2013	Fiction
3	Gone Girl	Gillian Flynn	4.0	57271	9	2014	Fiction
4	Harry Potter and the Cursed Child, Parts 1 & 2, Special Rehearsal Edition Script	J.K. Rowling	4.0	23973	12	2016	Fiction
5	The Elegance of the Hedgehog	Muriel Barbery	4.0	1859	11	2009	Fiction
6	A Wrinkle in Time (Time Quintet)	Madeleine L'Engle	4.5	5153	5	2018	Fiction
7	Divergent / Insurgent	Veronica Roth	4.5	17684	6	2014	Fiction
8	Fifty Shades Freed: Book Three of the Fifty Shades Trilogy (Fifty Shades of Grey Series) (English Edition)	E L James	4.5	20262	11	2012	Fiction
9	Fifty Shades Trilogy (Fifty Shades of Grey / Fifty Shades Darker / Fifty Shades Freed)	E L James	4.5	13964	32	2012	Fiction
10	Joyland (Hard Case Crime)	Stephen King	4.5	4748	12	2013	Fiction
11	Little Fires Everywhere	Celeste Ng	4.5	25706	12	2018	Fiction
12	Looking for Alaska	John Green	4.5	8491	7	2014	Fiction
13	Mockingjay (The Hunger Games)	Suzanne Collins	4.5	36744	6	2010	Fiction

Rows per page: 100 1 - 100 of 550 First page < > Last page

- Create a output bucket to store the outputs by name case_output_bucket1.

Run the pipeline code in jupyter lab:

```
from __future__ import absolute_import

import argparse

import logging

import re

import apache_beam as beam

from apache_beam.options.pipeline_options import PipelineOptions

import os

import google.auth

from apache_beam.options import pipeline_options

from apache_beam.options.pipeline_options import GoogleCloudOptions

from apache_beam.runners import DataflowRunner

from google.cloud import bigquery


def Split(element):

    return element.split(",")

#outputs text in a readable format

def FormatText(elem):

    return 'AVERAGE RATING OF BOOKS:'+str(elem[1])

#finds the average of given list of sets

class AverageFn(beam.CombineFn):

    def create_accumulator(self):

        return (0.0, 0) # initialize (sum, count)

    def add_input(self, sum_count, inputt):
```

```

(summ, count) = sum_count

(summ2, c2) = inputt

return summ + float(summ2)*c2, count + c2

options = PipelineOptions()

#p = beam.Pipeline(options=options)

p = beam.Pipeline(InteractiveRunner())

p2 = beam.Pipeline(InteractiveRunner())

# Setting up the Apache Beam pipeline options.

options = pipeline_options.PipelineOptions(flags=[])

# Sets the project to the default project in your current Google Cloud environment.

_, options.view_as(GoogleCloudOptions).project = google.auth.default()

# Sets the Google Cloud Region in which Cloud Dataflow runs.

options.view_as(GoogleCloudOptions).region = 'us-west2'

dataflow_gcs_location = 'gs://case_output_bucket1/dataflow'

options.view_as(GoogleCloudOptions).staging_location = '%s/staging' % dataflow_gcs_location

# Dataflow Temp Location. This location is used to store temporary files or intermediate results before finally outputting to the
sink.

options.view_as(GoogleCloudOptions).temp_location = '%s/temp' % dataflow_gcs_location

client = bigquery.Client()

dataset_id = "keyskills-1637308289548.case1_amz"

#dataset = bigquery.Dataset(dataset_id)

dataset.location = "us-west2"

dataset.description = "dataset books"

#dataset_ref = client.create_dataset(dataset, timeout = 30)

# split and Convert to json

def to_json(csv_str):

    fields = csv_str.split(',')

```

```

    json_str = {"Name":fields[0],

                "Author": fields[1],

                "User_Rating": fields[2],

                "Reviews": fields[3],

                "Price": fields[4],

                "Year": fields[5],

                "Genre": fields[6]

                }

    return json_str

table_schema =
'Name:STRING,Author:STRING,User_Rating:FLOAT,Reviews:INTEGER,Price:Integer,Year:Integer,Genre:STRING'

bs = (p2 | beam.io.ReadFromText("gs://mycase1-bucket3/cleaned_books_amazon.csv"))

(bs | 'cleaned_data to json' >> beam.Map(to_json)

| 'write to bigquery' >> beam.io.WriteToBigQuery(

"keyskills-1637308289548:case1_amz.fic1",

schema=table_schema,

create_disposition=beam.io.BigQueryDisposition.CREATE_IF_NEEDED,

write_disposition=beam.io.BigQueryDisposition.WRITE_APPEND,

custom_gcs_temp_location="gs://case_output_bucket1/dataflow/temp"

))

from apache_beam.runners.runner import PipelineState

ret = p2.run()

if ret.state == PipelineState.DONE:

    print('Success!!!')

else:

    print('Error Running beam pipeline')

#read data and split based on ','

```

```
books = ( p | beam.io.ReadFromText("gs://mycase1-bucket3/cleaned_books_amazon.csv") | beam.Map(Split) )
```

#Filter records having fiction , map each rating as a set (rating,1), use combineperkey to count the number of each rating, run the average function, write result to Fiction_avg

```
res1 = (
```

```
  books
```

```
  | beam.Filter(lambda rec : rec[6]=="Fiction")
```

```
  | beam.Map(lambda rec: (rec[2], 1))
```

```
  | "Grouping keys" >> beam.CombinePerKey(sum)
```

```
  | "Combine Globally" >> beam.CombineGlobally(AverageFn())
```

```
  | "write" >> beam.io.WriteToText("gs://case_output_bucket1/Fiction_avg") )
```

#Filter records having Non Fiction , map each rating as a set (rating,1), use combineperkey to count the number of each rating, run the average function, write result to N_Fiction_avg

```
Res2 = (
```

```
  books
```

```
  | beam.Filter(lambda rec : rec[6]=="Non Fiction")
```

```
  | beam.Map(lambda rec: (rec[2], 1))
```

```
  | "Grouping keys" >> beam.CombinePerKey(sum)
```

```
  | "Combine Globally" >> beam.CombineGlobally(AverageFn())
```

```
  | "write" >> beam.io.WriteToText("gs://case_output_bucket1/N_Fiction_avg")
```

```
)
```

map each rating as a set (rating,1), use combineperkey to count the number of each rating, run the average function, write result to All_avg

```
Res3 = (
```

```
  books
```

```
  | beam.Map(lambda rec: (rec[2], 1))
```

```
  | "Grouping keys" >> beam.CombinePerKey(sum)
```

```
  | "Combine Globally" >> beam.CombineGlobally(AverageFn())
```

```
  | "write" >> beam.io.WriteToText("gs://case_output_bucket1/All_avg") )
```

#map each record's 0th column that is name with value 1 , run distinct function to get the distinct values of name, run top.of(5) to sort and get the last5 books alphabetically and store in storage bucket last5

```
f_res = (books | beam.Map(lambda rec: (rec[0], 1)) | beam.Distinct() | beam.combiners.Top.Of(5) |  
beam.io.WriteToText("gs://case_output_bucket1/Last_5_books"))
```

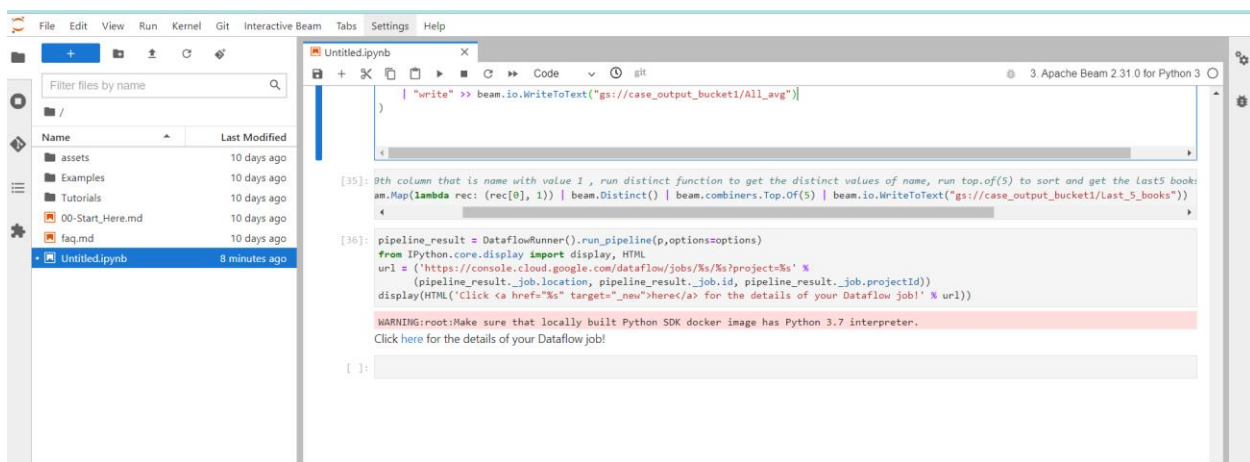
```
pipeline_result = DataflowRunner().run_pipeline(p,options=options)
```

```
from IPython.core.display import display, HTML
```

```
url = ('https://console.cloud.google.com/dataflow/jobs/%s/%s?project=%s' %
```

```
(pipeline_result._job.location, pipeline_result._job.id, pipeline_result._job.projectId))
```

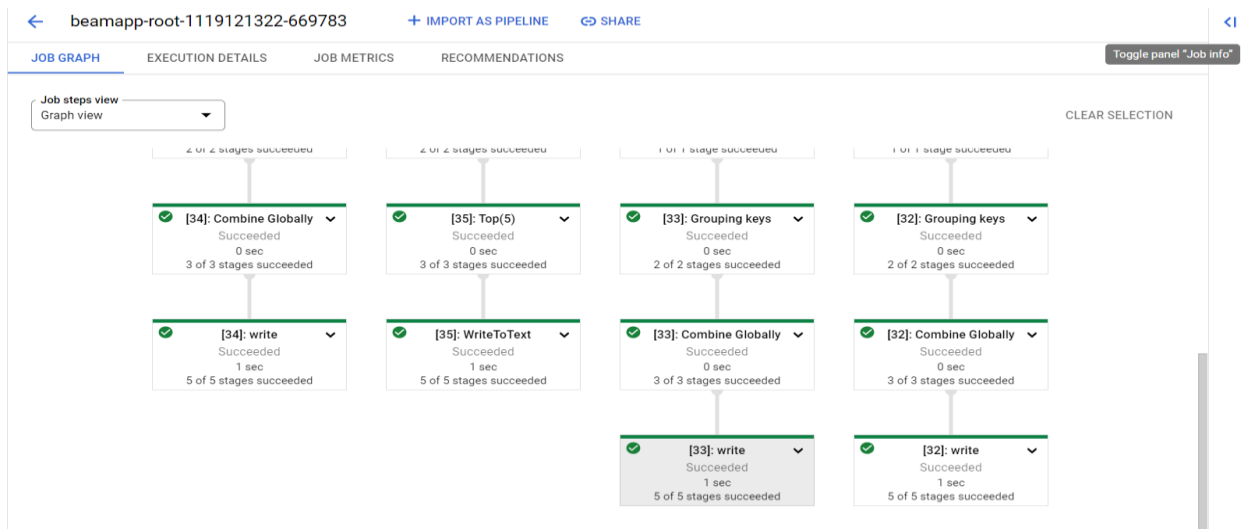
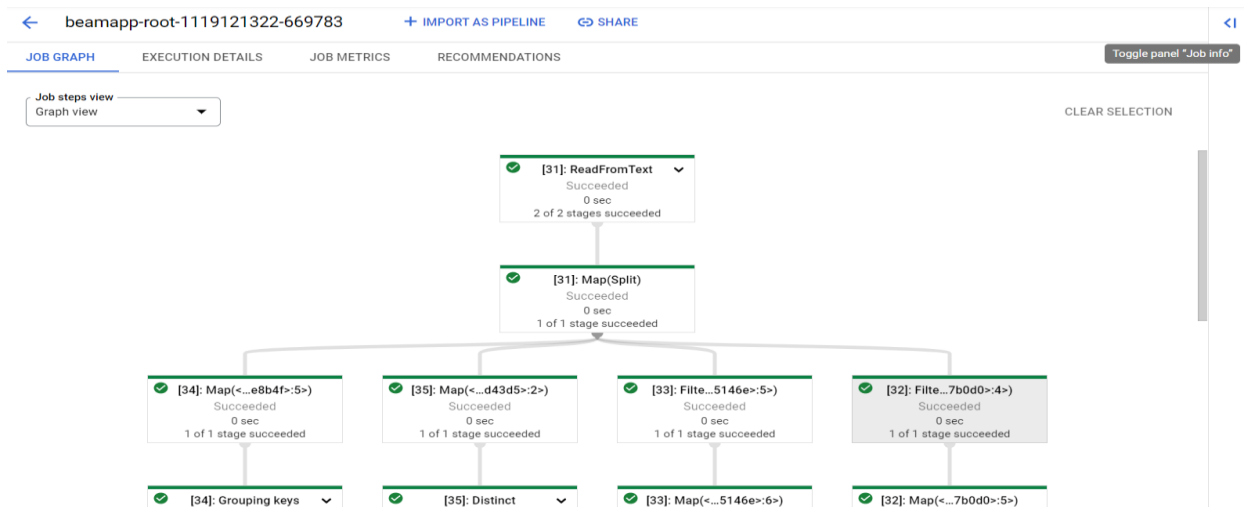
```
display(HTML('Click <a href="%s" target="_new">here</a> for the details of your Dataflow job!' % url))
```



Dataflow job succeeded:

Google Cloud Platform									
Dataflow									
Jobs									
Filter jobs									
Refresh, Updated just now									
Name	Type	End time	Elapsed time	Start time	Status	SDK version	ID	Region	
beamapp-root-1119121322-669783	Batch	Nov 19, 2021, 5:49:56 PM	6 min 30 sec	Nov 19, 2021, 5:43:26 PM	Succeeded	2.31.0	2021-11-19_04_13_25-1525247603527737833	us-west2	
beamapp-root-1119120928-376293	Batch	Nov 19, 2021, 5:41:02 PM	4 sec	Nov 19, 2021, 5:40:58 PM	Failed	2.31.0	2021-11-19_04_10_57-471101722300970550	us-west2	

Below figures show the dataflow pipeline obtained after running the pipeline code in Apache beam for Python environment in Jupyter Lab.



Output files obtained in the case_output_bucket1:

case_output_bucket1

Location: us-west2 (Los Angeles) Storage class: Standard Public access: Not public Protection: None

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

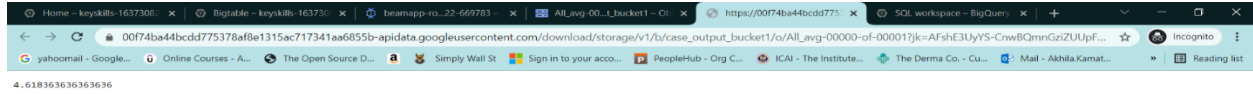
Buckets > case_output_bucket1

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER MANAGE HOLDS DOWNLOAD DELETE

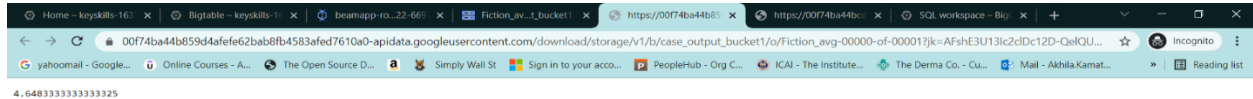
Filter by name prefix only Filter Filter objects and folders Show deleted data

	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention
<input type="checkbox"/>	All_avg-00000-of-00001	18 B	text/plain	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	Fiction_avg-00000-of-00001	19 B	text/plain	Nov 19, 20...	Standard	Nov 19, 2021, 5:46:54 PM	Not public	—	Google-managed key	—
<input type="checkbox"/>	Last_5_books-00000-of-00001	293 B	text/plain	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	N_Fiction_avg-00000-of-00001	17 B	text/plain	Nov 19, 20...	Standard	Nov 19, 20...	Not public	—	Google-managed key	—
<input type="checkbox"/>	dataflow/	—	Folder	—	—	—	—	—	—	—

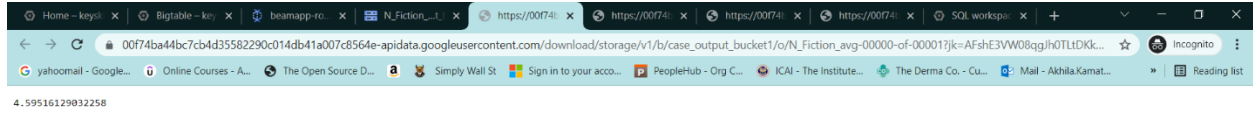
Average of all books :



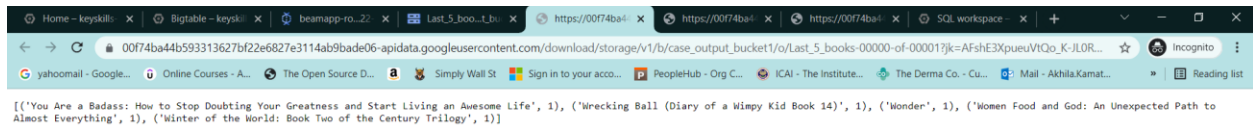
Average of Fiction Books:



Average of Non Fiction Books:



Last 5 Books:



Case study 2:

Big Data Analytics with Java using Cloud Dataproc, Google's Fully-Managed Spark and Hadoop Service, IBRD Statement Of Loans Data dataset, from Kaggle is used as input.

The International Bank for Reconstruction and Development (IBRD) loans are public and publicly guaranteed debt extended by the World Bank Group. IBRD loans are made to, or guaranteed by, countries that are members of IBRD. This dataset contains historical snapshots of the Statement of Loans including the latest available snapshots.

There are two data files available. The Statement of Loans latest available snapshots data file contains 8,713 rows of loan data (~3 MB), ideal for development and testing. The Statement of Loans historic data file contains approximately 750,000 rows of data (~265

MB). Although not exactly 'big data', the historic dataset is large enough to sufficiently explore Dataproc. Both IBRD files have an identical schema with 33 columns of data

PROBLEM STATEMENT:

For the analysis, one has to

1. Ascertain the top 25 historic IBRD borrower;
2. Determine their total disbursements, current obligations, and the average interest rates they were charged.
3. Output/Persist the results in asked target endpoints*

Implementation Steps –

- Note – To show the capabilities of Cloud Dataproc, one has to create a three-node cluster, upload Java- analytics jobs and data to Google Cloud Storage, and execute the jobs on the Spark cluster. One has use Stackdriver for monitoring and notifications for the Dataproc clusters and the jobs running on the clusters. [15% credits]
- One has to demonstrate the use of the Google Cloud Console, as well as Google's Cloud SDK's command line tools, for all tasks.[5% credits]
Analysis has to be done using Spark's SQL capabilities. The results of the

analysis, a Spark DataFrame containing 25 Rows, will be saved as a CSV file then to be pushed to HBase [BigTable] for further processing and reporting [20% credits]

- One has to leverage Google Cloud Storage to load local Spark Jobs before submitting to DataProc for running in Cluster.
- Ensure we are using the dataproc clusters create command, to create the 3 node cluster using CLI. [10% Credits]
- Cluster details – The three node Linux cluster [a single master node and two worker nodes]. All three nodes use the n1-standard-4 machine type, with 4 vCPU and 15 GB of memory. Although still considered a minimally-sized cluster, this cluster represents a significant increase in compute power

1. Uploading Job Resources to Cloud Storage [10% credits]

In total, we need to upload four items to the new Cloud Storage bucket. The items include the two Kaggle IBRD CSV files, the compiled Java JAR file from the dataproc-java project, and any dataset needed to support the work. Using the Google Cloud Console, upload the four files to the new Google Storage bucket, as shown below. Make sure you unzip the two Kaggle IBRD CSV data files before uploading.

2. Running Jobs on Dataproc[15% credits]

The easiest way to run a job on the Dataproc cluster is by submitting a job through the Dataproc Jobs UI, part of the Google Cloud Console.

3. Spark Jobs [15% credits]

To run a Spark job using the JAR file, select Job type Spark. The Region will match your Dataproc cluster and bucket locations, us-east-1 in my case. You should have a choice of both clusters in your chosen region..

4. File Output [Most critical step to follow][10% credits]

During development and testing, outputting results to the console is useful. However, in Production, the output from jobs is most often written to Apache Avro, CSV, JSON, or XML format files, persisted Apache Hive, SQL, or NoSQL database, or streamed to another system for post-processing, using technologies such as Apache Kafka.

*Target endpoints - [one has to demonstrate usage of Apache Avro, JSON, Apache Hive[BigQuery], SQL or a NoSQL DB.]

Once the Java jobs have run successfully on the Dataproc cluster, you should observe the results have been saved back to the Storage bucket as a CSV file.

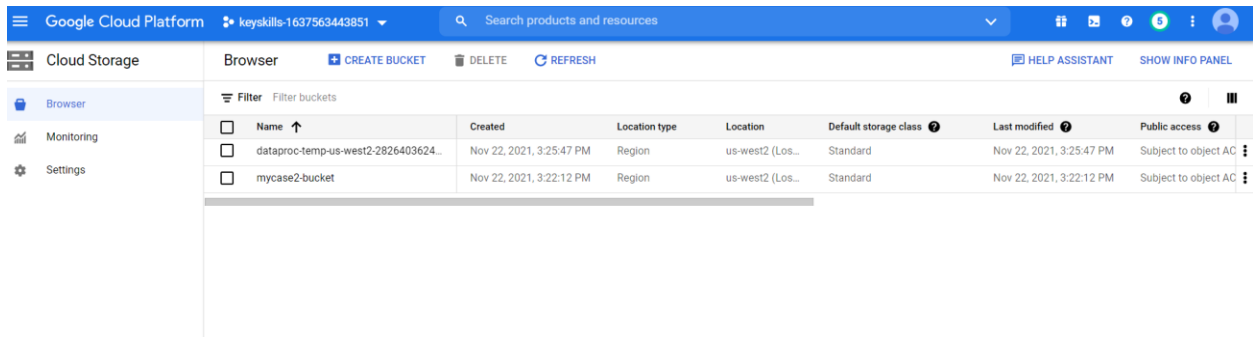
Engineering Best Practices to be followed [20% credits]

6. Logging has to be implemented in all the Java Components
7. Exception handling framework need to be implemented
8. Unit testing of each component/service using JUnit/Mokito Framework
9. Unit Testing of Spark Cluster, Data Pipeline [is desirable]
10. Usage of 4 to 5 Big Data Design Patterns using Java [is desirable]

Answer:

Bucket creation using CLI:

- `gsutil mb -c standard -l us-west2 gs://mycase2-bucket`



The screenshot shows the Google Cloud Platform interface for Cloud Storage. The 'Buckets' tab is selected, displaying a table of buckets. The table has columns for Name, Created, Location type, Location, Default storage class, Last modified, and Public access. Two buckets are listed: 'dataproc-temp-us-west2-2826403624...' and 'mycase2-bucket'.

Name	Created	Location type	Location	Default storage class	Last modified	Public access
dataproc-temp-us-west2-2826403624...	Nov 22, 2021, 3:25:47 PM	Region	us-west2 (Los...	Standard	Nov 22, 2021, 3:25:47 PM	Subject to object AC
mycase2-bucket	Nov 22, 2021, 3:22:12 PM	Region	us-west2 (Los...	Standard	Nov 22, 2021, 3:22:12 PM	Subject to object AC

Cluster creation using CLI:

- `gcloud dataproc clusters create mycase2cluster --bucket mycase2-bucket --region us-west2 --zone us-west2-a --master-machine-type n1-standard-4 --master-boot-disk-size 50 --num-workers 3 --worker-machine-type n1-standard-4 --worker-boot-disk-size 50 --image-version 1.4-ubuntu18 --optional-components ANACONDA,JUPYTER,ZOOKEEPER --enable-component-gateway --project keyskills-1637563443851`

```
kshgcp_userid01@cloudshell:~ (keyskills-1637563443851) $ gcloud dataproc clusters create mycase2cluster --bucket mycase2-bucket --region us-west2 --zone us-west2-a --master-machine-type n1-standard-4 --master-boot-disk-size 50 --num-workers 3 --worker-machine-type n1-standard-4 --worker-boot-disk-size 50 --image-version 1.4-ubuntu18 --optional-components ANACONDA,JUPYTER,ZOOKEEPER --enable-component-gateway --project keyskills-1637563443851
Waiting on operation [projects/keyskills-1637563443851/regions/us-west2/operations/3e4089c9-b95a-38f6-b12d-8c316f740951].
Waiting for cluster creation operation...
WARNING: 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.
Waiting for cluster creation operation... done.
Created [https://dataproc.googleapis.com/91/projects/keyskills-1637563443851/regions/us-west2/clusters/mycase2cluster] Cluster placed in zone [us-west2-a].
kshgcp_userid01@cloudshell:~ (keyskills-1637563443851) $
```

Dataproc cluster with 3 worker nodes:

Google Cloud Platform keyskills-1637563443851 Search products and resources									
Dataproc	Clusters	CREATE CLUSTER	REFRESH	START	STOP	DELETE	REGIONS	+ 5 RECOMMENDED ALERTS	SHOW INFO PANEL
Jobs on Clusters	Filter Search clusters, press Enter								
Clusters	<input type="checkbox"/>	Name ↑	Status	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket	Created
Jobs	<input type="checkbox"/>	mycase2cluster	Running	us-west2	us-west2-a	3	Off	mycase2-bucket	Nov 22, 2021, 3:36:08 PM
Workflows									
Autoscaling policies									
Utilities									
Component exchange									
Metastore									
Workbench									

Uploading the required csv files into the bucket:

Bucket details

REFRESH

HELP ASSISTANT

LEARN

mycase2-bucket

Location

Storage class

Public access

Protection

us-west2 (Los Angeles)

Standard

Subject to object ACLs

None

OBJECTS

CONFIGURATION

PERMISSIONS

PROTECTION

LIFECYCLE

Buckets

>

mycase2-bucket

UPLOAD FILES

UPLOAD FOLDER

CREATE FOLDER

MANAGE HOLDS

DOWNLOAD

DELETE

Filter by name prefix only

Filter

Filter objects and folders

Show deleted data

<input type="checkbox"/>	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	...
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—	
<input type="checkbox"/>	ibrd-statement-of-loans-historical-d	353.1 MB	text/csv	Nov 22, 2...	Standard	Nov 22, 20...	Not public	—	Google-managed key	
<input type="checkbox"/>	ibrd-statement-of-loans-latest-avail	3.3 MB	text/csv	Nov 22, 2...	Standard	Nov 22, 20...	Not public	—	Google-managed key	
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—	

Uploads and keyskills-1637563443851 operations

Uploading 2 files

ibrd-statement-of-loans-historical-data.csv

Complete

ibrd-statement-of-loans-latest-available-snapshot.csv

Complete

Creating the JAR file:

- Java code:

package dataproc;

import org.apache.spark.sql.Dataset;

import org.apache.spark.sql.Row;

```

import org.apache.spark.sql.SaveMode;

import org.apache.spark.sql.SparkSession;

public class InternationalLoansAppDataprocLarge {

    public static void main(String[] args) {        InternationalLoansAppDataprocLarge app = new
InternationalLoansAppDataprocLarge();

    app.start();

    }

    private void start() {

        SparkSession spark = SparkSession.builder()

            .appName("my-java-dataproc")

            .master("yarn")

            .getOrCreate();

        spark.sparkContext().setLogLevel("WARN");

// Loading the CSV file from GCS Bucket

        Dataset<Row> dfLoans = spark.read()

            .format("csv")

            .option("header", "true")

            .option("inferSchema", true)

            .load("gs://mycase2-bucket/ibrd-statement-of-loans-historical-data.csv");

// Creates temporary view using DataFrame

        dfLoans.withColumnRenamed("Country", "country")

            .withColumnRenamed("Country Code", "country_code")

            .withColumnRenamed("Disbursed Amount", "disbursed")

            .withColumnRenamed("Borrower's Obligation", "obligation")

            .withColumnRenamed("Interest Rate", "interest_rate")

            .createOrReplaceTempView("loans");

// Performs basic analysis of dataset

        Dataset<Row> dfDisbursement = spark.sql(

```



```

"SELECT country, country_code, "
    + "format_number(total_disbursement, 0) AS total_disbursement, "
    + "format_number(ABS(total_obligation), 0) AS total_obligation, "
    + "format_number(avg_interest_rate, 2) AS avg_interest_rate "
    + "FROM ( "
    + "SELECT country, country_code, "
    + "SUM(disbursed) AS total_disbursement, "
    + "SUM(obligation) AS total_obligation, "
    + "AVG(interest_rate) AS avg_interest_rate "
    + "FROM loans "
    + "GROUP BY country, country_code "
    + "ORDER BY total_disbursement DESC "
    + "LIMIT 25)"
);

// Saves results to single CSV file in GCS Bucket
dfDisbursement.repartition(1)
    .write()
    .mode(SaveMode.Overwrite)
    .format("csv")
    .option("header", "true")
    .save("gs://mycase2-bucket/ibrd-loan-summary-large");

System.out.println("Results successfully written to CSV file");
}
}

```

- The java code is saved as csfinal and exported as a jar file

Uploading the JAR file in the bucket:

← Bucket details REFRESH HELP ASSISTANT LEARN

mycase2-bucket

Location: us-west2 (Los Angeles) | Storage class: Standard | Public access: Subject to object ACLs | Protection: None

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE

Buckets > mycase2-bucket

UPLOAD FILES | UPLOAD FOLDER | CREATE FOLDER | MANAGE HOLDS | DOWNLOAD | DELETE

Filter by name prefix only | Filter | Filter objects and folders | Show deleted data

	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption
<input type="checkbox"/>	csfinal.jar	265.8 MB	application/octet-stream	Nov 22, 2...	Standard	Nov 22, 20...	Not public	—	Google-mar
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—
<input type="checkbox"/>	lbrd-statement-of-loans-historical-d	353.1 MB	text/csv	Nov 22, 2...	Standard	Nov 22, 20...	Not public	—	Google-mar
<input type="checkbox"/>	lbrd-statement-of-loans-latest-avall	3.3 MB	text/csv	Nov 22, 2...	Standard	Nov 22, 20...	Not public	—	Google-mar
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—

Uploads and keyskills-1637563443851 operations

- csfinal.jar Complete
- Uploading 2 files
- lbrd-statement-of-loans-historical-data.csv Complete
- lbrd-statement-of-loans-latest-available-snapshot.csv Complete

Submitting Dataproc job using CLI command:

- gcloud dataproc jobs submit spark --cluster=mycase2cluster --region=us-west2 --class=dataproc.InternationalLoansAppDataprocLarge --jars=gs://mycase2-bucket/csfinal.jar

Google Cloud Platform | keyskills-1637563443851 | Search products and resources

Dataproc | Jobs | SUBMIT JOB | REFRESH | STOP | DELETE | REGIONS | + 2 RECOMMENDED ALERTS | SHOW INFO PANEL

Jobs on Clusters

- Clusters
- Jobs**
- Workflows
- Autoscaling policies

Utilities

- Component exchange
- Metastore
- Workbench

Job ID	Status	Region	Type	Cluster	Start time	Elapsed time	Labels
6ff79de8a6bc4a77846a9947d761b204	Succeeded	us-west2	Spark	mycase2cluster	Nov 22, 2021, 4:08:46 PM	1 min 8 sec	None

Dataproc job succeeded and the data is saved in the CSV file:

← Job details | CLONE | DELETE | STOP | REFRESH

Job ID: 6ff79de8a6bc4a77846a9947d761b204
Job UUID: c003333b-1370-3c08-b3c7-33d27ce131bc
Type: Dataproc Job
Status: Succeeded

Output | LINE WRAP: OFF

```
21/11/22 10:38:54 INFO org.spark_project.jetty.util.log: Logging initialized @3236ms to org.spark_project.jetty.util.log.Slf4jLog
21/11/22 10:38:54 INFO org.spark_project.jetty.server.Server: jetty-9.4.z-SNAPSHOT; built: unknown; git: unknown; jvm 1.8.0_275-Bu275-b01-0ubuntu1-18.04-b01
21/11/22 10:38:54 INFO org.spark_project.jetty.server.Server: Started @3388ms
21/11/22 10:38:54 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@7ce7e83c(HTTP/1.1, (http/1.1)){0.0.0.0:4040}
21/11/22 10:38:54 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair Scheduler configuration file not found so jobs will be scheduled in FIFO order. To use fair scheduling, confi
21/11/22 10:38:54 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at mycase2cluster-m/10.168.0.9:8032
21/11/22 10:38:55 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at mycase2cluster-m/10.168.0.9:10200
21/11/22 10:38:58 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1637575646572_0001
21/11/22 10:39:42 WARN org.apache.spark.util.Utils: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.maxToStringf
Results successfully written to CSV file
```

Output file added to the bucket:

[←](#) Bucket details [REFRESH](#) [HELP ASSISTANT](#) [LEARN](#)

mycase2-bucket

Location: us-west2 (Los Angeles) Storage class: Standard Public access: Subject to object ACLs Protection: None

[OBJECTS](#) [CONFIGURATION](#) [PERMISSIONS](#) [PROTECTION](#) [LIFECYCLE](#)

Buckets > mycase2-bucket

[UPLOAD FILES](#) [UPLOAD FOLDER](#) [CREATE FOLDER](#) [MANAGE HOLDS](#) [DOWNLOAD](#) [DELETE](#)

Filter by name prefix only Filter Filter objects and folders Show deleted data

	Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption
<input type="checkbox"/>	csfinal.jar	265.8 MB	application/octet-stream	Nov 22, 20...	Standard	Nov 22, 20...	Not public	—	Google-man
<input type="checkbox"/>	google-cloud-dataproc-metainfo/	—	Folder	—	—	—	—	—	—
<input checked="" type="checkbox"/>	ibrd-loan-summary-large/	—	Folder	—	—	—	—	—	—
<input type="checkbox"/>	ibrd-statement-of-loans-historical-d	353.1 MB	text/csv	Nov 22, 20...	Standard	Nov 22, 20...	Not public	—	Google-man
<input type="checkbox"/>	ibrd-statement-of-loans-latest-avail	3.3 MB	text/csv	Nov 22, 20...	Standard	Nov 22, 20...	Not public	—	Google-man
<input type="checkbox"/>	notebooks/	—	Folder	—	—	—	—	—	—

Creating a BigQuery table:

- Go to BigQuery and create a table under the project name's dataset (i.e output1) and select the output file from GCS and load it into table.
- Give the table name, in this case its loansummary.

Create table

Source

Create table from

Google Cloud Storage

Select file from GCS bucket *

mycase2-bucket/ibrd-loan-summary-large/part-00000-919dc47 [BROWSE](#)

File format

CSV

☐ Source Data Partitioning

Destination

Project *

keyskills-1637563443851 [BROWSE](#)

Dataset ID *

output1

Table name *

loansummary

Unicode letters, marks, numbers, connectors, dashes or spaces allowed.

Table type

Native table

[CREATE TABLE](#) [CANCEL](#)

BigQuery table for output file is created:

Google Cloud Platform | keyskills-1637563443851 | Search products and resources

FEATURES & INFO | SHORTCUT | DISABLE EDITOR TABS

Explorer | + ADD DATA

loansummary

SCHEMA | DETAILS | PREVIEW

Table schema

Filter Enter property name or value

Field name	Type	Mode	Policy Tags	Description
country	STRING	NULLABLE		
country_code	STRING	NULLABLE		
total_disbursement	INTEGER	NULLABLE		
total_obligation	INTEGER	NULLABLE		
avg_interest_rate	FLOAT	NULLABLE		

EDIT SCHEMA | VIEW ROW ACCESS POLICIES

Click on preview to see the output:

- Output showing the top 25 historic IBRD borrowers.
- Also shows the total disbursements, current obligations, and the average interest rates they were charged.

Google Cloud Platform | keyskills-1637563443851 | Search products and resources

FEATURES & INFO | SHORTCUT | DISABLE EDITOR TABS

Explorer | + ADD DATA

loansummary

SCHEMA | DETAILS | **PREVIEW**

Row	country	country_code	total_disbursement	total_obligation	avg_interest_rate
1	Brazil	BR	5722173997528	1687497391153	4.01
2	Mexico	MX	5548466349768	1696743402361	4.87
3	Indonesia	ID	4424650929974	1636279345733	4.6
4	China	CN	4180040102521	1574714083399	2.99
5	India	IN	4155693388268	1501966913537	3.7
6	Turkey	TR	3729186036275	1422497908202	4.78
7	Argentina	AR	3022864423101	714017670701	3.3
8	Colombia	CO	2333039710369	1040444472300	4.4
9	Korea, Republic of	KR	1752500020587	9609765857	6.8
10	Philippines	PH	1604975603970	543174877765	5.33
11	Poland	PL	1547580713649	871199374874	2.85
12	Morocco	MA	1444432749026	533022489730	4.29
13	Dominican Republic	DO	1276386650648	104779057766	4.25

Rows per page: 100 | 1 - 25 of 25 | First page | < | > | Last page

