Parquet is more space efficient than JSON/CSV

Created "**csv_conversion_parquet/json.ipynb**" python notebook in Google Colab and executed step by step as shown in below steps:

Step1: Importing the required libraries for pyspark and for creating the spark session, datetime and finally creating sparksession and mounting the csv data from google drive into google colab.

Import required libraries

```
[38] import findspark
     findspark.init()
     from pyspark import SparkContext, SparkConf
     from pyspark.sql import SQLContext, SparkSession
     from pyspark.sql.functions import *

     from datetime import datetime
     import json
```

Create a Spark Session

```
[4]  conf = SparkConf().set('spark.ui.port', '4050')
     sc = SparkContext(conf=conf)
     spark = SparkSession.builder.master('local[*]').getOrCreate()
```

Mount GoogleDrive

```
[ ]  from google.colab import drive
     drive.mount("/content/gdrive")
```

Step2: Read the CSV dataset

Load Dataset into dataframe

```
[ ] # Load the dataset
    data = spark.read.load('/content/drive/MyDrive/PBDM_Dataset/Sample-Spreadsheet-500000-rows.csv', format='csv', inferSchema=True, header=True)

    # Print schema
    data.printSchema()
```

```
root
 |-- Eldon Base for stackable storage shelf, platinum: string (nullable = true)
 |-- Muhammed MacIntyre: string (nullable = true)
 |-- 3: string (nullable = true)
 |-- -213.25: string (nullable = true)
 |-- 38.94: string (nullable = true)
 |-- 35: string (nullable = true)
 |-- Nunavut: string (nullable = true)
 |-- Storage & Organization: string (nullable = true)
 |-- 0.8: string (nullable = true)
```

```
# The number of rows in the dataset
data.count()
```

```
59506
```

Step3: Converting CSV files to JSON, Parquet and then also converting Parquet to Parquet

Convert CSV to JSON

```
start_time = datetime.now()
print ("Reading CSV file started at : ",start_time)
df = spark.read.load('/content/drive/MyDrive/PBDM_Dataset/Sample-Spreadsheet-500000-rows.csv', format='csv', inferSchema=True, header=True)
df.write.mode("overwrite").format("json").save("/content/drive/MyDrive/PBDM_Dataset/json_output")
# df.to_json("output_data.json",sep=',',index=None)
# df.toJSON("/content/drive/MyDrive/PBDM_Dataset/json_output")
# json_output = df.toJSON()
end_time = datetime.now()
print ("Writing to JSON completed at: ",end_time)
print ("Total Duration in Converting CSV to JSON: ",end_time-start_time)
```

```
Reading CSV file started at :  2022-11-16 03:39:09.828771
Writing to JSON completed at:  2022-11-16 03:39:11.489498
Total Duration in Converting CSV to JSON:  0:00:01.660727
```

Convert CSV to Parquet

```
start_time = datetime.now()
print ("Reading CSV file started at : ",start_time)
df = spark.read.load('/content/drive/MyDrive/PBDM_Dataset/Sample-Spreadsheet-500000-rows.csv', format='csv', inferSchema=False, header=False)
df.write.parquet("/content/drive/MyDrive/PBDM_Dataset/parquet_output/")
end_time = datetime.now()
print ("Writing as Parquet completed at: ",end_time)
print ("Total Duration in Converting CSV to Parquet: ",end_time-start_time)
```

```
Reading CSV file started at :  2022-11-16 03:26:53.390479
Writing as Parquet completed at:  2022-11-16 03:26:55.352221
Total Duration in Converting CSV to Parquet:  0:00:01.961742
```

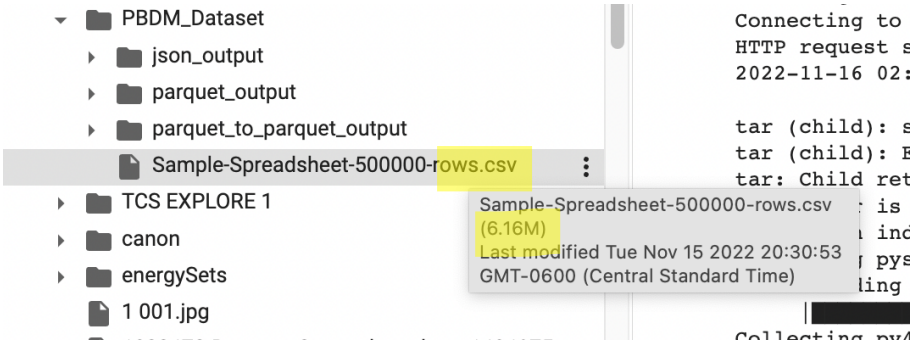
Convert Parquet to Parquet

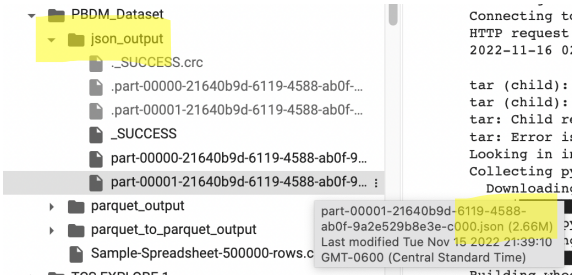
```
start_time = datetime.now()
print ("Reading Parquet file started at : ",start_time)
df = spark.read.parquet("/content/drive/MyDrive/PBDM_Dataset/parquet_output")
df.write.mode("overwrite").parquet("/content/drive/MyDrive/PBDM_Dataset/parquet_to_parquet_output")
end_time = datetime.now()
print ("Writing as Parquet completed at: ",end_time)
print ("Total Duration in Converting Parquet to Parquet: ",end_time-start_time)
```

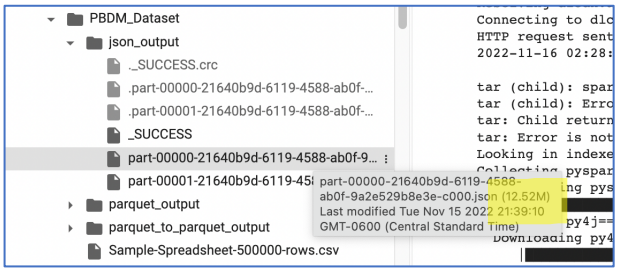
```
Reading Parquet file started at :  2022-11-16 03:30:40.972232
Writing as Parquet completed at:  2022-11-16 03:30:42.448722
Total Duration in Converting Parquet to Parquet:  0:00:01.476490
```

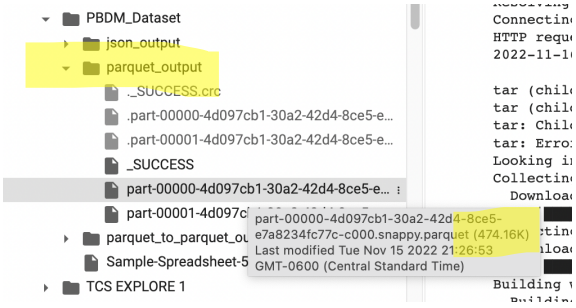
From CSV to JSON, the conversion took nearly 1.66 sec whereas CSV to Parquet, it took nearly 1.96 sec. While we also tried to convert Parquet to Parquet to check how much does it take to read and convert. Hence, it took 1.47 sec which is less from all the conversions.

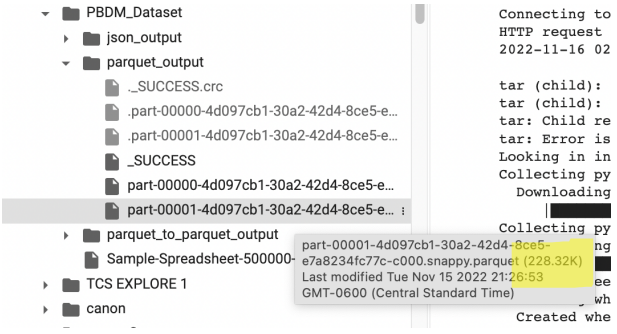
Step4: Here we are checking file sizes of both JSON and Parque and also CSV











From the above figures, we can see the actual file size of CSV is 6.16MB, whereas after converting CSV to JSON, the file sizes are 2.6MB & 12.6MB. While Parquet Conversion holds very less size in KB's as 474KB & 228KB

Step5: Here we are checking query performance on both CSV and Parquet.

Aggregations on CSV file

```
df = df = spark.read.load('/content/drive/MyDrive/PBDM_Dataset/Sample-Spreadsheet-500000-rows.csv', format='csv', inferSchema=False, header=False)
start_time = datetime.now()
#print Group and count
df.groupBy('_c1').count().show()
#print avg
df.select(mean('_c2')).show()
#print min
df.select(min('_c2')).show()
#print max
df.select(max('_c2')).show()
#print count of unique
df.select(countDistinct('_c2')).show()
end_time = datetime.now()
print ("Total Duration for the above aggregations on csv file: ",end_time-start_time)
```

```
+----------------+-----+
|              _c1|count|
+----------------+-----+
|     Jesus Ocampo|   30|
|      Jim Mitchum|   60|
|         Joy Bell|   66|
|    Hilary Holden|   66|
|  Patrick O'Brill|   54|
|   Parhena Norris|   42|
|     Ruben Ausman|   48|
| Michelle Ellison|  108|
|     Ben Peterman|   42|
|         Jay Fine|  116|
|  Ted Butterfield|   12|
|    Bill Overfelt|   60|
|        Denny Joy|   42|
|      Joseph Holt|   18|
|     Melanie Page|   72|
|     Xylona Price|   84|
|       Carl Weiss|  126|
|Brooke Gillingham|  108|
|   Darren Koutras|   42|
|     Craig Yedwab|   42|
+----------------+-----+
only showing top 20 rows

+------------------+
|          avg(_c2)|
+------------------+
|30586.989853104984|
+------------------+

+---------+
| min(_c2)|
+---------+
| 10/Pack"|
+---------+

+--------------+
|      max(_c2)|
+--------------+
|Yoseph Carroll|
+--------------+

+-------------------+
|count(DISTINCT _c2)|
+-------------------+
|               5585|
+-------------------+

Total Duration for the above aggregations on csv file:  0:00:03.639811
```

From the above figure, we had written code to perform Aggregation functions like Max, Min, Avg and also finding the unique record on each of the column of CSV file. Thereby, the total runtime it took was nearly 3.63 Sec.

Aggragations on Parquet File

```
df = spark.read.parquet("/content/drive/MyDrive/PBDM_Dataset/parquet_output")
start_time = datetime.now()
#print Group and count
df.groupBy('_c1').count().show()
#print avg
df.select(mean('_c2')).show()
#print min
df.select(min('_c2')).show()
#print max
df.select(max('_c2')).show()
#print count of unique
df.select(countDistinct('_c2')).show()
end_time = datetime.now()
print ("Total Duration for the above aggregations on parquet file: ",end_time-start_time)
```

```
+-----------------+-----+
|              _c1|count|
+-----------------+-----+
|     Jesus Ocampo|   30|
|      Jim Mitchum|   60|
|         Joy Bell|   66|
|    Hilary Holden|   66|
|  Patrick O'Brill|   54|
|   Parhena Norris|   42|
|     Ruben Ausman|   48|
| Michelle Ellison|  108|
|     Ben Peterman|   42|
|         Jay Fine|  116|
|  Ted Butterfield|   12|
|    Bill Overfelt|   60|
|        Denny Joy|   42|
|      Joseph Holt|   18|
|     Melanie Page|   72|
|     Xylona Price|   84|
|       Carl Weiss|  126|
|Brooke Gillingham|  108|
|   Darren Koutras|   42|
|     Craig Yedwab|   42|
+-----------------+-----+
only showing top 20 rows

+------------------+
|          avg(_c2)|
+------------------+
|30586.989853104984|
+------------------+

+---------+
| min(_c2)|
+---------+
| 10/Pack"|
+---------+

+--------------+
|      max(_c2)|
+--------------+
|Yoseph Carroll|
+--------------+

+-------------------+
|count(DISTINCT _c2)|
+-------------------+
|               5585|
+-------------------+

Total Duration for the above aggregations on parquet file:  0:00:03.133509
```

From the above figure, similar to CSV, we had written code to perform Aggregation functions on each of the column of Parquet file. Thereby, the total runtime it took was nearly 3.13 Sec which is less when compared to CSV.

Hence from the above findings, we can conclude that

- Parquet format is better in terms of storage when compared to CSV and JSON.
- It is also far more efficient than CSV and JSON as because conversion to Parquet took very less time due to less size.
- Runtime in executing queries is faster than other formats. Thereby, optimizing the Query performance in Parquet.