Lab5-Task2: Guided Solution for Ingestion of Google Reviews Data

Import Modules & Set Environment Variables:

```
from pyspark.sql import SparkSession
from pyspark.sql import functions as F
from pyspark.sql import Row
from pyspark.sql import types as T
```

This section is about importing the necessary modules and setting the correct environment paths for PySpark, Hadoop, and Python.

Initialize SparkSession:

```
spark = SparkSession.builder.master("local").appName('ex5_google_reviews').getOrCreate()
```

A new SparkSession is created with the name 'ex5_google_reviews', using a single local node.

Setting Up Sentiment Mapping:

```
sentiment_arr = [Row(Sentiment='Positive', sentiment_rank=1),
                 Row(Sentiment='Neutral', sentiment_rank=0),
                 Row(Sentiment='Negative', sentiment_rank=-1)]
print(sentiment_arr)
```

This sets up an array of Row objects to map the sentiment to respective ranks.

Data Loading

```
# Load the Google Reviews CSV data into a DataFrame.
google_reviews_df = spark.read.csv('s3a://spark/data/raw/google_reviews/', header=True)
google_reviews_df.show(6)
```

Here, the CSV file containing the Google reviews is loaded into a DataFrame, and a sample of six records is displayed.

Creating Sentiment DataFrame:

```
##Convert the sentiment array to a DataFrame.
sentiments_df = spark.createDataFrame(sentiment_arr)
sentiments_df.show()
```

This section turns the sentiment mapping array into a DataFrame and displays its content.

Joining DataFrames:

```
# Join the sentiments_df with the main reviews DataFrame based on the 'Sentiment' column.
joined_df = google_reviews_df.join(F.broadcast(sentiments_df), ['Sentiment'])
```

The Google reviews data is joined with the sentiment mapping DataFrame. A broadcast join is utilized to replicate the smaller DataFrame (sentiments_df) across all nodes to speed up the join.

Data Transformation & Cleaning:

```
selected_df = joined_df\
    .select(F.col('App').alias('application_name'),
            F.col('Translated_Review').alias('translated_review'),
            F.col('sentiment_rank'),
            F.col('Sentiment_Polarity').cast(T.FloatType()).alias('sentiment_polarity'),
            F.col('Sentiment_Subjectivity').cast(T.FloatType()).alias('sentiment_subjectivity'))
```

Data is transformed to:

- Extract and rename relevant columns.
- Cast 'Sentiment_Polarity' and 'Sentiment_Subjectivity' columns to float type.

Display & Save Data:

```
# Display the transformed data and its schema.
selected_df.show()
selected_df.printSchema()

# Save the processed data into a Parquet file.
selected_df.write.parquet('s3a://spark/data/source/google_reviews', mode='overwrite')
spark.stop()
```

This portion of the code presents the transformed data, its schema, and then writes the results to a Parquet file. If the destination file already exists, it will be overwritten.

The SparkSession is closed to free up resources.

Summery:

This solution processes Google Reviews data, extracting sentiment analysis metrics, and then saves the cleaned and transformed data into an optimized Parquet format.

Full code solution

```
from pyspark.sql import SparkSession
from pyspark.sql import functions as F
from pyspark.sql import Row
from pyspark.sql import types as T

spark = SparkSession.builder.master("local").appName('ex5_google_reviews').getOrCreate()

sentiment_arr = [Row(Sentiment='Positive', sentiment_rank=1),
            Row(Sentiment='Neutral', sentiment_rank=0),
            Row(Sentiment='Negative', sentiment_rank=-1)]
print(sentiment_arr)

google_reviews_df = spark.read.csv('s3a://spark/data/raw/google_reviews/', header=True)
google_reviews_df.show(6)

sentiments_df = spark.createDataFrame(sentiment_arr)
sentiments_df.show()

joined_df = google_reviews_df.join(F.broadcast(sentiments_df), ['Sentiment'])

selected_df = joined_df \
   .select(F.col('App').alias('application_name'),
         F.col('Translated_Review').alias('translated_review'),
         F.col('sentiment_rank'),
         F.col('Sentiment_Polarity').cast(T.FloatType()).alias('sentiment_polarity'),
         F.col('Sentiment_Subjectivity').cast(T.FloatType()).alias('sentiment_subjectivity'))
selected_df.show()
selected_df.printSchema()
selected_df.write.parquet('s3a://spark/data/source/google_reviews', mode='overwrite')

spark.stop()
```