# Lab5-Task1: Guided Solution for Ingestion of Google Play Store Data:

## 1. Initialization & Configuration:

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

# Create a SparkSession, setting up configurations as needed.
spark = SparkSession.builder.master("local").appName('ex5_google_apps').getOrCreate()
```

Here, a SparkSession is initiated with the name 'ex6_google_apps'. The master node is set to 'local', which means Spark runs on a single machine.

## 2. Setting Up Age Limit Mapping:

```
#Define an array of Row objects that map age limits to content ratings.
age_limit_arr = [Row(age_limit=18, Content_Rating='Adults only 18+'),
                 Row(age_limit=17, Content_Rating='Mature 17+'),
                 Row(age_limit=12, Content_Rating='Teen'),
                 Row(age_limit=10, Content_Rating='Everyone 10+'),
                 Row(age_limit=0, Content_Rating='Everyone')]
```

This is essentially a key-value mapping to transform content ratings into numeric age limits. If you print it, it will look like list below:

```
print(age_limit_arr)

[Row(Content_Rating='Adults only 18+', age_limit=18), Row(Content_Rating='Mature 17+',
age_limit=17), Row(Content_Rating='Teen', age_limit=12), Row(Content_Rating='Everyone 10+',
age_limit=10), Row(Content_Rating='Everyone', age_limit=0)]
```

**Data Reading**:

```
#Read the Google Play Store data CSV into a DataFrame.
google_apps_df = spark.read.csv('s3a://spark/data/raw/google_apps/', header=True)
google_apps_df.printSchema()
```

This loads the CSV file into a DataFrame and displays its schema.

**Creating Age Limit DataFrame**:

```
#Convert the age limit mapping array to a DataFrame.
age_limit_df = spark.createDataFrame(age_limit_arr).withColumnRenamed('Content_Rating', 'Content Rating')
age_limit_df.show()
```

The **age_limit_arr** list is turned into a DataFrame. It also renames the 'Content_Rating' column for ease of joining.

**Joining DataFrames**:

```
# Join the age_limit_df with the main DataFrame based on the 'Content Rating' column.
joined_df = google_apps_df.join(F.broadcast(age_limit_df), ['Content Rating'])
```

The main data is joined with the age limit mapping DataFrame. We use broadcast join here, as the age_limit_df is small and can be replicated across all nodes to speed up the join process.

**Data Transformation & Cleaning**:

```
selected_df = joined_df\
.    select(F.col('App').alias('application_name,('
            F.col('Category').alias('category'),
            F.col('Rating').alias('rating'),
            F.col('Reviews').cast(T.FloatType()).alias('reviews'),
            F.col('Size').alias('size'),
            F.regexp_replace(F.col('Installs'), '[^0-9]', '').cast(T.DoubleType()).alias('num_of_installs'),
            F.col('Price').cast(T.DoubleType()).alias('price'),
            F.col('age_limit'),
            F.col('Genres').alias('genres'),
            F.col('Current Ver').alias('version')\ (
   .fillna(-1, 'Rating')
```

The data undergoes multiple transformations:

- Relevant columns are selected and renamed.
- Non-numeric characters are removed from the 'Installs' column.
- The 'Price' column is cast to DoubleType.
- Missing ratings are replaced with -1.

**Displaying & Writing Data**:

```
#Show a snippet of the transformed data.
selected_df.show(6)
selected_df.printSchema()

#Write the final DataFrame to a Parquet file.
selected_df.write.parquet('s3a://spark/data/source/google_apps', mode='overwrite')
spark.stop()
```

This portion showcases a sample of the transformed data, its schema, and saves the DataFrame as a Parquet file, overwriting if the file already exists.

Finally, the SparkSession is gracefully terminated.

In this solution, the Google Play Store data is transformed into a structured format and saved to Parquet, enabling optimized querying and data analysis.

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_apps').getOrCreate()

age_limit_arr = [Row(age_limit=18, Content_Rating='Adults only 18+'),
            Row(age_limit=17, Content_Rating='Mature 17+'),
            Row(age_limit=12, Content_Rating='Teen'),
            Row(age_limit=10, Content_Rating='Everyone 10+'),
            Row(age_limit=0, Content_Rating='Everyone')]
print(age_limit_arr)
google_apps_df = spark.read.csv('s3a://spark/data/raw/google_apps/', header=True)

google_apps_df.printSchema()
age_limit_df = spark.createDataFrame(age_limit_arr).withColumnRenamed('Content_Rating', 'Content Rating')
age_limit_df.show()
joined_df = google_apps_df.join(F.broadcast(age_limit_df), ['Content Rating'])

selected_df = joined_df \
   .select(F.col('App').alias('application_name'),
           F.col('Category').alias('category'),
           F.col('Rating').alias('rating'),
           F.col('Reviews').cast(T.FloatType()).alias('reviews'),
           F.col('Size').alias('size'),
           F.regexp_replace(F.col('Installs'), '[^0-9]', '').cast(T.DoubleType()).alias('num_of_installs'),
           F.col('Price').cast(T.DoubleType()).alias('price'),
           F.col('age_limit'),
           F.col('Genres').alias('genres'),
           F.col('Current Ver').alias('version')) \
   .fillna(-1, 'Rating')

selected_df.show(6)
selected_df.printSchema()
selected_df.write.parquet('s3a://spark/data/source/google_apps', mode='overwrite')

spark.stop()
```