Project: Data Lake with EMR (Spark & Co. @ AWS)

Creating EMR Cluster (`"emr-5.31.0" - Spark 2.4.6` , `Zeppelin 0.8.2` , `Hadoop 2.10.0`) of AWS
This last EMR Version does have the correct setup up with `JupyterEnterpriseGateway` !

Overview and Steps

- Step 0: Create Base Config (Setup EMR Cluster (with Spark), Create Notebook, Attach Notebook to Cluster)
- Step 1: Create Spark Session
- Step 2: Define Data Paths and Access Data
- Step 3: Process Song Data (read out, define schemas, write as parquet)
- Step 4: Process Log Data (read out, define schemas, write as parquet)
- Step 5: Run [`etl.py`] script with clean code
- Step 6: Clean up Resources (EMR, Notbook, S3)

*Version 3 **(AWS, EMR)** - Revision 09 - 2023/04/04 Mr Morphy - GitHub Profile (https://github.com/MrMorphy)*
GitHub Project - udacity-course-proj-data-lake (https://github.com/MrMorphy/udacity-course-proj-data-lake)

Step 0: Create Base Config

- Setup EMR Cluster (with Spark) at AWS UI
- Create Notebook at AWS UI
- Attach Notebook to 'Waiting' EMR Cluster

Load libraries to execute code

In [1]:
```
#import configparser
from datetime import datetime
import os
```

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	
0	application_1680641614700_0001	pyspark	idle	Link (http://ip-172-31-5-17.us-west-2.compute.internal:20888/proxy/application_1680641614700_0001/)	2.compute.inte

```
SparkSession available as 'spark'.
```

In [2]:
```
from pyspark.sql import SparkSession
```

In [3]:
```
from pyspark.sql.functions import udf, col, monotonically_increasing_id
from pyspark.sql.functions import year, month, dayofmonth, hour, weekofyear, date_format
```

In [4]:
```
from pyspark.sql.types import StructType as R,    StructField as Fld, \
                              DoubleType as Dbl,  StringType as Str, \
                              IntegerType as Int, DateType as Date, TimestampType
```

Step 1: Create Spark Session

In [6]:
```
# DEBUG
spark
```

```
<pyspark.sql.session.SparkSession object at 0x7fc0cb14bdd0>
```

Step 2: Define Data Paths and Access Data

In [7]:
```
# AWS: Online @ AWS - S3:
input_data  = "s3a://udacity-dend/"
output_data = "s3a://data-lake-project-out/"
```

Step 3: Process Song Data

(read out, define schemas, write as parquet)

In [8]:
```
# AWS:
#def process_song_data(spark, input_data, output_data):

# get filepath to FULL SET of song data file [song_data/A/B/C/TRABCEI128F424C983.json]
song_data = input_data + 'song_data/*/*/*/*.json'
```

In [9]:
```
# define the song schema (like [staging_songs] table)
songSchema = R([
   Fld("artist_id",        Str()),
   Fld("artist_latitude",  Dbl()),
   Fld("artist_location",  Str()),
   Fld("artist_longitude", Dbl()),
   Fld("artist_name",      Str()),
   Fld("duration",         Dbl()),
   Fld("num_songs",        Int()),
   Fld("song_id",          Str()),
   Fld("title",            Str()),
   Fld("year",             Int()),
])
```

In [10]:
```
# read song data JSON file into data frame
df = spark.read.json(song_data, schema=songSchema)
```

▸ Spark Job Progress

In [11]:
```
# DEBUG
# songSchema given
df.printSchema()
```

```
root
 |-- artist_id: string (nullable = true)
 |-- artist_latitude: double (nullable = true)
 |-- artist_location: string (nullable = true)
 |-- artist_longitude: double (nullable = true)
 |-- artist_name: string (nullable = true)
 |-- duration: double (nullable = true)
 |-- num_songs: integer (nullable = true)
 |-- song_id: string (nullable = true)
 |-- title: string (nullable = true)
 |-- year: integer (nullable = true)
```

In [13]:
```
# df.count()
print('>> [' + str(df.count()) + '] songs from song_data read out in JSON-format')
```

▸ Spark Job Progress

```
>> [14896] songs from song_data read out in JSON-format
```

In [12]:
```
# DEBUG
df.show(5)
```

▸ Spark Job Progress

```
+------------------+---------------+--------------------+----------------+--------------------+----
-----+---------+------------------+--------------------+----+
|         artist_id|artist_latitude|     artist_location|artist_longitude|         artist_name| dur
ation|num_songs|           song_id|               title|year|
+------------------+---------------+--------------------+----------------+--------------------+----
-----+---------+------------------+--------------------+----+
|AR4T2IF1187B9ADBB7|       63.96027|<a href="http://b...|        10.22442|          Billy Idol|233.
22077|        1|SOVIYJY12AF72A4B00|The Dead Next Doo...|1983|
|AR4T2IF1187B9ADBB7|       63.96027|<a href="http://b...|        10.22442|          Billy Idol|287.
92118|        1|SOVYXYL12AF72A3373|Rebel Yell (1999 ...|1983|
|ARQ846I1187B9A7083|           null|                    |            null|Yvonne S. Moriart...|196.
04853|        1|SOEPTVC12A67ADD0DA|To Zucchabar ["Gl...|   0|
|AR4T2IF1187B9ADBB7|       63.96027|<a href="http://b...|        10.22442|          Billy Idol|247.
53587|        1|SOLQYSZ12AB0181F97|    Mony Mony (Live)|1987|
|AR3TZ691187FB3DBB1|           null|                    |            null|Russell Watson / ...|273.
44934|        1|SOVPFJK12A6701CB16|Barcelona - (Frie...|2000|
+------------------+---------------+--------------------+----------------+--------------------+----
-----+---------+------------------+--------------------+----+
only showing top 5 rows
```

In [14]:
```
# DEBUG
df1 = df.filter(df.title == 'Young Boy Blues')
df1.show(5)
```

▸ Spark Job Progress

```
+---------+---------------+---------------+----------------+-----------+--------+---------+-------+
-----+----+
|artist_id|artist_latitude|artist_location|artist_longitude|artist_name|duration|num_songs|song_id|
title|year|
+---------+---------------+---------------+----------------+-----------+--------+---------+-------+
-----+----+
+---------+---------------+---------------+----------------+-----------+--------+---------+-------+
-----+----+
```

Working with only a subset of the files, to save time!

In [23]:
```
# get filepath to song data file [song_data/A/B/C/TRABCEI128F424C983.json]
#song_data = input_data + 'song_data/*/*/*/*.json'
# Experiement with a subset of data:
song_data = input_data + 'song_data/A/A/A/*.json'
```

In [24]:
```
# read song data JSON file into data frame
df = spark.read.json(song_data, schema=songSchema)
```

In [25]:
```
# df.count()
print('>> [' + str(df.count()) + '] songs from song_data read out in JSON-format')
```

▸ Spark Job Progress

```
>> [24] songs from song_data read out in JSON-format
```

In [26]:
```
# extract columns to create songs table
song_columns = ["song_id", "title", "artist_id", "year", "duration"]
songs_table = df.select(song_columns) \
                .dropDuplicates()
```

In [16]:
```
# DEBUG
songs_table.printSchema()
```

```
root
 |-- song_id: string (nullable = true)
 |-- title: string (nullable = true)
 |-- artist_id: string (nullable = true)
 |-- year: integer (nullable = true)
 |-- duration: double (nullable = true)
```

In [17]:
```
# DEBUG
songs_table.show(5)
```

▸ Spark Job Progress

```
+------------------+--------------------+------------------+----+---------+
|           song_id|               title|         artist_id|year| duration|
+------------------+--------------------+------------------+----+---------+
|SOIIRYF12AAF3B4858|Pitbulls in the P...|ARZNUT51187FB44098|   0|230.29506|
|SOGUKZL12A6D4FDF28|The Apples Are Ju...|ARY1NR31187FB45277|   0|156.39465|
|SORXLTC12A81C20B18|La Chiave (feat.S...|ARLN9UN1187FB481FD|   0|198.00771|
|SOAIWGK12AB018325D|Bach Prelude In G...|ARTPL6J1187FB4D6FB|1992|136.33261|
|SOFZFSD12A8C13DEF4|Sweet Sophia (Alb...|ARYPW621187FB5B49E|2007|191.76444|
+------------------+--------------------+------------------+----+---------+
only showing top 5 rows
```

In [28]:
```
# [X] VERIFICAR @ AWS...
# write songs table to parquet files partitioned by year and artist
songs_table.repartition('year', 'artist_id') \
           .write.partitionBy('year', 'artist_id') \
           .parquet(output_data + 'songs_data/songs_table.parquet', 'overwrite')
```

▸ Spark Job Progress

In [ ]:

In [29]:
```
# extract columns to create artists table
artists_column = ["artist_id", "artist_name as name", "artist_location as location", \
                  "artist_latitude as latitude", "artist_longitude as longitude"]
artists_table = df.selectExpr(artists_column).dropDuplicates()
```

In [30]:
```
# DEBUG
artists_table.printSchema()
```

```
root
 |-- artist_id: string (nullable = true)
 |-- name: string (nullable = true)
 |-- location: string (nullable = true)
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
```

In [31]:
```
# DEBUG
artists_table.show(5)
```

▸ Spark Job Progress

```
+------------------+----------------+--------------------+--------+----------+
|         artist_id|            name|            location|latitude| longitude|
+------------------+----------------+--------------------+--------+----------+
|ARZKCQM1257509D107|     Dataphiles|                    |    null|      null|
|ARKYKXP11F50C47A6A|The Supersuckers|                    |    null|      null|
|ARGE7G11187FB37E05|    Cyndi Lauper|        Brooklyn, NY|    null|      null|
|ARY589G1187B9A9F4E|     Talkdemonic|        Portland, OR|45.51179|-122.67563|
|ARA23XO1187B9AF18F| The Smithereens|Carteret, New Jersey|40.57885| -74.21956|
+------------------+----------------+--------------------+--------+----------+
only showing top 5 rows
```

In [32]:
```
# write artists table to parquet files
artists_table.write.parquet(output_data + 'artists_data/artists_table.parquet', 'overwrite')
```

▸ Spark Job Progress

Step 4: Process Log Data

(read out, define schemas, write as parquet)

In [33]:
```
# AWS:
#def process_log_data(spark, input_data, output_data):

# get filepath to FULL Data Set log data file [log_data/2018/11/2018-11-17-events.json]
log_data = input_data + 'log_data/*/*/*.json'
```

In [34]:
```
# read log data file into data frame
df = spark.read.json(log_data)
```

▸ Spark Job Progress

In [35]:
```
# number of lines
# df.count()
print('>> [' + str(df.count()) + '] logs entries read IN, of JSON logs_data')
```

▸ Spark Job Progress

```
>> [8056] logs entries read IN, of JSON logs_data
```

In [36]:
```
# DEBUG
df.printSchema()
```

```
root
 |-- artist: string (nullable = true)
 |-- auth: string (nullable = true)
 |-- firstName: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- itemInSession: long (nullable = true)
 |-- lastName: string (nullable = true)
 |-- length: double (nullable = true)
 |-- level: string (nullable = true)
 |-- location: string (nullable = true)
 |-- method: string (nullable = true)
 |-- page: string (nullable = true)
 |-- registration: double (nullable = true)
 |-- sessionId: long (nullable = true)
 |-- song: string (nullable = true)
 |-- status: long (nullable = true)
 |-- ts: long (nullable = true)
 |-- userAgent: string (nullable = true)
 |-- userId: string (nullable = true)
```

Working with only a subset of the files, to save time!

In [37]:
```
# 1) experiment with a subset of the files,
# 2) and with local data (extract before from /data/log-data.zip)
log_data = input_data + 'log_data/2018/11/*.json'
```

In [38]:
```
# read log data file into data frame
df = spark.read.json(log_data)
```

▸ Spark Job Progress

In [39]:
```
# number of lines
# df.count()
print('>> [' + str(df.count()) + '] logs entries read IN, of JSON logs_data')
```

▸ Spark Job Progress

```
>> [8056] logs entries read IN, of JSON logs_data
```

In [40]:
```
# DEBUG
df.show(5)
```

▸ Spark Job Progress

```
+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+----------------+---------+--------------+------+-------------+-----------------
---+------+
|     artist|     auth|firstName|gender|itemInSession|lastName|   length|level|            location
|method|    page|     registration|sessionId|           song|status|           ts|            userAg
ent|userId|
+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+----------------+---------+--------------+------+-------------+-----------------
---+------+
|   Harmonia|Logged In|     Ryan|     M|            0|   Smith|655.77751| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|  Sehr kosmisch|   200|1542241826796|"Mozilla/5.0 (X1
1...|    26|
|The Prodigy|Logged In|     Ryan|     M|            1|   Smith|260.07465| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|The Big Gundown|   200|1542242481796|"Mozilla/5.0 (X1
1...|    26|
|      Train|Logged In|     Ryan|     M|            2|   Smith|205.45261| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|       Marry Me|   200|1542242741796|"Mozilla/5.0 (X1
1...|    26|
|       null|Logged In|    Wyatt|     M|            0|   Scott|     null| free|Eureka-Arcata-For...
|   GET|    Home|1.540872073796E12|      563|           null|   200|1542247071796|Mozilla/5.0 (Win
d...|     9|
|       null|Logged In|   Austin|     M|            0| Rosales|     null| free|New York-Newark-J...
|   GET|    Home|1.541059521796E12|      521|           null|   200|1542252577796|Mozilla/5.0 (Win
d...|    12|
+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+----------------+---------+--------------+------+-------------+-----------------
---+------+
only showing top 5 rows
```

In [41]:
```
# filter by actions for song plays
df = df.filter(df.page == 'NextSong')
```

In [42]:
```
# number of reduce amount of lines
# df.count()
print('>> [' + str(df.count()) + '] songs filtered "NextSong" of logs_data')
```

▸ Spark Job Progress

```
>> [6820] songs filtered "NextSong" of logs_data
```

In [43]:
```
# extract columns for USERS table
# artists_table =  # TYPO at resources (*.zip)!
users_columns = ["userId as user_id", "firstName as first_name", \
                 "lastName as last_name", "gender", "level"]
users_table = df.selectExpr(users_columns).dropDuplicates()
```

In [44]:
```
# DEBUG
users_table.printSchema()
```

```
root
 |-- user_id: string (nullable = true)
 |-- first_name: string (nullable = true)
 |-- last_name: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- level: string (nullable = true)
```

In [45]:
```
# DEBUG
users_table.show(5)
```

▸ Spark Job Progress

```
+-------+----------+---------+------+-----+
|user_id|first_name|last_name|gender|level|
+-------+----------+---------+------+-----+
|     81|    Sienna|    Colon|     F| free|
|     75|    Joseph|Gutierrez|     M| free|
|     47|    Kimber|   Norris|     F| free|
|     20|     Aiden|  Ramirez|     M| paid|
|     90|    Andrea|   Butler|     F| free|
+-------+----------+---------+------+-----+
only showing top 5 rows
```

In [46]:
```
# write USERS table to parquet files
users_table.write.parquet(output_data + 'users_data/users_table.parquet', 'overwrite')
```

▸ Spark Job Progress

In [47]:
```
# create "timestamp" column from original timestamp ("ts") column
get_datetime = udf(lambda x: str( datetime.fromtimestamp( int(x)/1000.0 )) )
dfTimestamp = df.withColumn("start_time", get_datetime(df.ts))
```

In [48]:
```
# extract columns to create time table
time_table = dfTimestamp.select("start_time").dropDuplicates() \
                        .withColumn("hour",    hour(col("start_time"))) \
                        .withColumn("day",     dayofmonth(col("start_time"))) \
                        .withColumn("week",    weekofyear(col("start_time"))) \
                        .withColumn("month",   month(col("start_time"))) \
                        .withColumn("year",    year(col("start_time"))) \
                        .withColumn("weekday", date_format(col("start_time"), 'E'))
```

In [50]:
```
# DEBUG
time_table.printSchema()
time_table.show(5)
```

▸ Spark Job Progress

```
root
 |-- start_time: string (nullable = true)
 |-- hour: integer (nullable = true)
 |-- day: integer (nullable = true)
 |-- week: integer (nullable = true)
 |-- month: integer (nullable = true)
 |-- year: integer (nullable = true)
 |-- weekday: string (nullable = true)

+--------------------+----+---+----+-----+----+-------+
|          start_time|hour|day|week|month|year|weekday|
+--------------------+----+---+----+-----+----+-------+
|2018-11-15 07:51:...|   7| 15|  46|   11|2018|    Thu|
|2018-11-15 10:53:...|  10| 15|  46|   11|2018|    Thu|
|2018-11-21 09:13:...|   9| 21|  47|   11|2018|    Wed|
|2018-11-15 17:10:...|  17| 15|  46|   11|2018|    Thu|
|2018-11-15 14:34:...|  14| 15|  46|   11|2018|    Thu|
+--------------------+----+---+----+-----+----+-------+
only showing top 5 rows
```

In [49]:
```
# write time table to parquet files partitioned by year and month
time_table.repartition('year', 'month') \
          .write.partitionBy("year", "month") \
          .parquet(output_data + 'time_data/time_table.parquet', 'overwrite')
```

▸ Spark Job Progress

Preparation and Code for `songplays_table`

In [52]:
```
get_datetime = udf(lambda x: str( datetime.fromtimestamp( int(x)/1000.0 )) )
df = df.withColumn("start_time", get_datetime(df.ts))
df = df.withColumn("year",       year(col("start_time")))
df = df.withColumn("month",      month(col("start_time")))
```

In [55]:
```
# DEBUG
df.printSchema()
df.show(5)
```

▸ Spark Job Progress

```
root
 |-- artist: string (nullable = true)
 |-- auth: string (nullable = true)
 |-- firstName: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- itemInSession: long (nullable = true)
 |-- lastName: string (nullable = true)
 |-- length: double (nullable = true)
 |-- level: string (nullable = true)
 |-- location: string (nullable = true)
 |-- method: string (nullable = true)
 |-- page: string (nullable = true)
 |-- registration: double (nullable = true)
 |-- sessionId: long (nullable = true)
 |-- song: string (nullable = true)
 |-- status: long (nullable = true)
 |-- ts: long (nullable = true)
 |-- userAgent: string (nullable = true)
 |-- userId: string (nullable = true)
 |-- start_time: string (nullable = true)
 |-- year: integer (nullable = true)
 |-- month: integer (nullable = true)

+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+-----------------+---------+--------------------+------+-------------+------------
--------+------+--------------------+----+-----+
|     artist|     auth|firstName|gender|itemInSession|lastName|   length|level|            location
|method|    page|     registration|sessionId|                song|status|           ts|           u
serAgent|userId|          start_time|year|month|
+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+-----------------+---------+--------------------+------+-------------+------------
--------+------+--------------------+----+-----+
|   Harmonia|Logged In|     Ryan|     M|            0|   Smith|655.77751| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|       Sehr kosmisch|   200|1542241826796|"Mozilla/5.0
 (X11...|    26|2018-11-15 00:30:...|2018|   11|
|The Prodigy|Logged In|     Ryan|     M|            1|   Smith|260.07465| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|     The Big Gundown|   200|1542242481796|"Mozilla/5.0
 (X11...|    26|2018-11-15 00:41:...|2018|   11|
|      Train|Logged In|     Ryan|     M|            2|   Smith|205.45261| free|San Jose-Sunnyval...
|   PUT|NextSong|1.541016707796E12|      583|            Marry Me|   200|1542242741796|"Mozilla/5.0
 (X11...|    26|2018-11-15 00:45:...|2018|   11|
|Sony Wonder|Logged In|   Samuel|     M|            0|Gonzalez|218.06975| free|Houston-The Woodl...
|   PUT|NextSong|1.540492941796E12|      597|           Blackbird|   200|1542253449796|"Mozilla/5.0
 (Mac...|    61|2018-11-15 03:44:...|2018|   11|
|  Van Halen|Logged In|    Tegan|     F|            2|  Levine|289.38404| paid|Portland-South Po...
|   PUT|NextSong|1.540794356796E12|      602|Best Of Both Worl...|   200|1542260935796|"Mozilla/5.0
 (Mac...|    80|2018-11-15 05:48:...|2018|   11|
+-----------+---------+---------+------+-------------+--------+---------+-----+--------------------
+------+--------+-----------------+---------+--------------------+------+-------------+------------
--------+------+--------------------+----+-----+
only showing top 5 rows
```

In [56]:
```
# read in song data to use for songplays table
song_df = spark.read.option("mergeSchema", "true") \
               .parquet(output_data + "songs_data/songs_table.parquet")
```

▸ Spark Job Progress

In [57]:
```
# DEBUG
print('>> [' + str(song_df.count()) + '] songs readout from songs_table.PARQUET')
```

▸ Spark Job Progress

```
>> [24] songs readout from songs_table.PARQUET
```

In [58]:
```
# DEBUG
song_df.printSchema()
song_df.show(5)
```

▸ Spark Job Progress

```
root
 |-- song_id: string (nullable = true)
 |-- title: string (nullable = true)
 |-- duration: double (nullable = true)
 |-- year: integer (nullable = true)
 |-- artist_id: string (nullable = true)

+------------------+--------------------+---------+----+------------------+
|           song_id|               title| duration|year|         artist_id|
+------------------+--------------------+---------+----+------------------+
|SOKTJDS12AF72A25E5|Drown In My Own T...|  192.522|   0|ARA23XO1187B9AF18F|
|SOEKAZG12AB018837E|I'll Slap Your Fa...|129.85424|2001|ARSVTNL1187B992A91|
|SOAFBCP12A8C13CC7D|King Of Scurf (20...|301.40036|1972|ARTC1LV1187B9A4858|
|SORRNOC12AB017F52B|The Last Beat Of ...|337.81506|2004|ARSZ7L31187FB4E610|
|SOQPWCR12A6D4FB2A3|A Poor Recipe For...|118.07302|2005|AR73AIO1187B9AD57B|
+------------------+--------------------+---------+----+------------------+
only showing top 5 rows
```

In [64]:
```
# DEBUG - JOIN - Testing (1/3)
df1 = song_df.filter(song_df.title == 'Scream')
df1.show(5)
```

▸ Spark Job Progress

```
+------------------+------+--------+----+------------------+
|           song_id| title|duration|year|         artist_id|
+------------------+------+--------+----+------------------+
|SOBLFFE12AF72AA5BA|Scream|213.9424|2009|ARJNIUY12298900C91|
+------------------+------+--------+----+------------------+
```

In [65]:
```
# DEBUG - JOIN - Testing (2/3)
df2 = df.filter(df.song == 'Scream') # Riverside / Young Boy Blues / Setanta matins
df2.show(5)
```

▸ Spark Job Progress

```
+---------------+---------+---------+------+-------------+--------+-------+-----+------------------
--+------+--------+-----------------+---------+------+------+-------------+--------------------+---
---+--------------------+----+-----+
|         artist|     auth|firstName|gender|itemInSession|lastName| length|level|             locati
on|method|    page|     registration|sessionId|  song|status|           ts|           userAgent|use
rId|          start_time|year|month|
+---------------+---------+---------+------+-------------+--------+-------+-----+------------------
--+------+--------+-----------------+---------+------+------+-------------+--------------------+---
---+--------------------+----+-----+
|Michael Jackson|Logged In|     Kate|     F|           32| Harrell|278.282| paid|Lansing-East Lan
s...|   PUT|NextSong|1.540472624796E12|      605|Scream|   200|1542298745796|"Mozilla/5.0 (X11...|
97|2018-11-15 16:19:...|2018|   11|
+---------------+---------+---------+------+-------------+--------+-------+-----+------------------
--+------+--------+-----------------+---------+------+------+-------------+--------------------+---
---+--------------------+----+-----+
```

In [63]:
```
# DEBUG - JOIN - Testing (3/3)
df3 = df.join(song_df, song_df.title == df.song ).select(df.song.alias("song (df)"), song_df.title.a
df3.count()
df3.show(5)
```

▸ Spark Job Progress

```
+---------+---------------+
|song (df)|title (song_df)|
+---------+---------------+
|   Scream|         Scream|
+---------+---------------+
```

In [66]:
```
# extract columns from joined song and log datasets to create songplays table
songplays_table = song_df.join(df, (song_df.title == df.song) ) \
                        .select('start_time', \
                                df.year, \
                                df.month, \
                                col('userId').alias("user_id"), \
                                df.level, \
                                song_df.song_id, \
                                song_df.artist_id, \
                                col('sessionId').alias("session_id"), \
                                df.location, \
                                col('userAgent').alias("user_agent") \
                                )
```

In [67]:
```
songplays_table = songplays_table.withColumn("songplay_id", monotonically_increasing_id())
```

In [68]:
```
# DEBUG
print('>> [' + str(songplays_table.count()) + '] songs found, on JOIN matching for songsplays_table'
```

▸ Spark Job Progress

```
>> [1] songs found, on JOIN matching for songsplays_table
```

In [69]:
```
# DEBUG
songplays_table.printSchema()
songplays_table.show(3)
```

▸ Spark Job Progress

```
An error was encountered:
Error sending http request and maximum retry encountered.
```

In [ ]:
```
# write songplays table to parquet files partitioned by year and month
songplays_table.repartition('year', 'month') \
               .write.partitionBy("year", "month") \
               .parquet(output_data + 'songplays_data/songplays_table.parquet', 'overwrite')
```

In [ ]:
```
print('>> END! (main)')
```

Step 5: Run [etl.py] script

... with working clean code

In [ ]:
```
!python 'etl.py' # to be executed at the Jupyter Notebook on AWS!
```

Step 6: Clean up Resources

(EMR, Notbook, S3)

- Export Jupyter Notebook
- Terminate EMR Cluster

- Delete S3 Bucket

============ [EOF] ============