Set Up Athena and Data Lake Queries

In [8]:
```
import boto3
import pandas as pd
from pyathena import connect

# Session and S3 setup
sess = boto3.Session()
s3_staging_dir = "s3://sagemaker-us-east-1-152605355048/athena/staging/"
region = sess.region_name

# Athena connection setup
conn = connect(region_name=region, s3_staging_dir=s3_staging_dir)

# Database and table names
database_name = "homework2"
table_name_csv = "datalake2"

# Function to run SQL queries
def run_query(statement):
    try:
        return pd.read_sql(statement, conn)
    except Exception as e:
        print(f"Failed to execute query: {e}")

# Verify if the table is created successfully
statement = "SHOW TABLES in {}".format(database_name)
df_show = run_query(statement)
print(df_show.head())
```

```
/tmp/ipykernel_2888/2226939753.py:20: UserWarning: pandas only supports SQLAlchemy c
onnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection.
Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  return pd.read_sql(statement, conn)
    tab_name
0   datalake
1  datalake2
```

SQL Queries:

1. List artist, track_name, and popularity for songs that have a popularity greater than or equal to 99

In [9]:
```
statement = """
SELECT artists, track_name, popularity
FROM {}.{}
WHERE popularity >= 99
""".format(database_name, table_name_csv)
df_sql = run_query(statement)
print(df_sql)
```

```
/tmp/ipykernel_2888/2226939753.py:20: UserWarning: pandas only supports SQLAlchemy c
onnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection.
Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  return pd.read_sql(statement, conn)
```

```
                artists                  track_name  popularity
0  Sam SmithKim Petras  Unholy (feat. Kim Petras)         100
1  Sam SmithKim Petras  Unholy (feat. Kim Petras)         100
```

2. List artists with an average popularity of 92

In [11]:
```
statement = """
SELECT artists, AVG(popularity) as avg_popularity
FROM {}.{}
GROUP BY artists
HAVING AVG(popularity) = 92
""".format(database_name, table_name_csv)

# Execute the query
try:
    df = pd.read_sql(statement, conn)
    print(df)
except Exception as e:
    print(f"Failed to execute query: {e}")
```

```
/tmp/ipykernel_2888/3528674370.py:10: UserWarning: pandas only supports SQLAlchemy c
onnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection.
Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df = pd.read_sql(statement, conn)
           artists  avg_popularity
0      Harry Styles            92.0
1  RemaSelena Gomez            92.0
```

3. List the Top 10 most energetic genres

In [12]:
```
statement = """
SELECT track_genre, AVG(energy) as avg_energy
FROM {}.{}
GROUP BY track_genre
ORDER BY avg_energy DESC
LIMIT 10
""".format(database_name, table_name_csv)

# Execute the query
try:
    df = pd.read_sql(statement, conn)
    print(df)
except Exception as e:
    print(f"Failed to execute query: {e}")
```

```
/tmp/ipykernel_2888/3955700541.py:11: UserWarning: pandas only supports SQLAlchemy c
onnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection.
Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df = pd.read_sql(statement, conn)
  track_genre  avg_energy
0       0.797   1174026.0
1       0.556    691306.0
2       0.492    542000.0
3        0.45    538160.0
4       0.347    526706.0
5      0.0761    502786.0
6      0.0903    449813.0
7       0.035    440310.0
8       0.483    371160.0
9       0.147    355693.0
```

4. How many tracks is Bad Bunny on?

In [13]:
```
statement = """
SELECT COUNT(*) as track_count
FROM {}.{}
WHERE artists LIKE '%Bad Bunny%'
""".format(database_name, table_name_csv)
# Execute the query
try:
    df = pd.read_sql(statement, conn)
    print(df)
except Exception as e:
    print(f"Failed to execute query: {e}")
```

```
/tmp/ipykernel_2888/1972276672.py:8: UserWarning: pandas only supports SQLAlchemy co
nnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection. O
ther DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df = pd.read_sql(statement, conn)
   track_count
0          416
```

5. Show the top 10 genres in terms of popularity sorted by their most popular track

In [14]:
```
statement = """
SELECT track_genre, MAX(popularity) as max_popularity
FROM {}.{}
GROUP BY track_genre
ORDER BY max_popularity DESC
LIMIT 10
""".format(database_name, table_name_csv)

# Execute the query
try:
    df = pd.read_sql(statement, conn)
    print(df)
except Exception as e:
    print(f"Failed to execute query: {e}")
```

```
/tmp/ipykernel_2888/3815318429.py:11: UserWarning: pandas only supports SQLAlchemy c
onnectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection.
Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df = pd.read_sql(statement, conn)
  track_genre  max_popularity
0         pop             100
1       dance             100
2         edm              98
3      latino              98
4       latin              98
5   reggaeton              98
6      reggae              98
7       piano              96
8        rock              96
9       chill              93
```

Set up AWS Data Wrangler

In [15]:
```
!pip install awswrangler
```

```
Requirement already satisfied: awswrangler in /opt/conda/lib/python3.10/site-package
s (3.9.1)
Requirement already satisfied: boto3<2.0.0,>=1.20.32 in /opt/conda/lib/python3.10/si
te-packages (from awswrangler) (1.34.84)
Requirement already satisfied: botocore<2.0.0,>=1.23.32 in /opt/conda/lib/python3.1
0/site-packages (from awswrangler) (1.34.84)
Requirement already satisfied: numpy<2.0,>=1.18 in /opt/conda/lib/python3.10/site-pa
ckages (from awswrangler) (1.26.4)
Requirement already satisfied: packaging<25.0,>=21.1 in /opt/conda/lib/python3.10/si
te-packages (from awswrangler) (23.2)
Requirement already satisfied: pandas<3.0.0,>=1.2.0 in /opt/conda/lib/python3.10/sit
e-packages (from awswrangler) (2.2.2)
Requirement already satisfied: pyarrow>=8.0.0 in /opt/conda/lib/python3.10/site-pack
ages (from awswrangler) (15.0.2)
Requirement already satisfied: typing-extensions<5.0.0,>=4.4.0 in /opt/conda/lib/pyt
hon3.10/site-packages (from awswrangler) (4.11.0)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /opt/conda/lib/python3.10/s
ite-packages (from boto3<2.0.0,>=1.20.32->awswrangler) (1.0.1)
Requirement already satisfied: s3transfer<0.11.0,>=0.10.0 in /opt/conda/lib/python3.
10/site-packages (from boto3<2.0.0,>=1.20.32->awswrangler) (0.10.1)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/lib/python
3.10/site-packages (from botocore<2.0.0,>=1.23.32->awswrangler) (2.9.0)
Requirement already satisfied: urllib3!=2.2.0,<3,>=1.25.4 in /opt/conda/lib/python3.
10/site-packages (from botocore<2.0.0,>=1.23.32->awswrangler) (2.2.1)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-packag
es (from pandas<3.0.0,>=1.2.0->awswrangler) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.10/site-pack
ages (from pandas<3.0.0,>=1.2.0->awswrangler) (2024.1)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packages
(from python-dateutil<3.0.0,>=2.1->botocore<2.0.0,>=1.23.32->awswrangler) (1.16.0)
WARNING: Running pip as the 'root' user can result in broken permissions and conflic
ting behaviour with the system package manager, possibly rendering your system unusa
ble.It is recommended to use a virtual environment instead: https://pip.pypa.io/warn
ings/venv. Use the --root-user-action option if you know what you are doing and want
to suppress this warning.
```

In [16]:
```
import awswrangler as wr
import pandas as pd

# S3 bucket where your dataset is located
bucket = 'sagemaker-us-east-1-152605355048 '
csv_file_path = f"s3://sagemaker-us-east-1-152605355048/homework2_dataset/csv/"

# Load CSV data into pandas dataframe using AWS Data Wrangler
df = wr.s3.read_csv(path=csv_file_path)

# Preview the first few rows of the dataframe
print(df.head())
```

```
                artists                track_name  popularity  duration_ms  \
0           Gen Hoshino                    Comedy          73       230666
1          Ben Woodward           Ghost - Acoustic          55       149610
2  Ingrid MichaelsonZAYN             To Begin Again          57       210826
3          Kina Grannis  Cant Help Falling In Love          71       201933
4      Chord Overstreet                   Hold On          82       198853

   explicit  danceability  energy  key  loudness  mode  speechiness  \
0     False         0.676  0.4610    1    -6.746     0       0.1430
1     False         0.420  0.1660    1   -17.235     1       0.0763
2     False         0.438  0.3590    0    -9.734     1       0.0557
3     False         0.266  0.0596    0   -18.515     1       0.0363
4     False         0.618  0.4430    2    -9.681     1       0.0526

   acousticness  instrumentalness  liveness  valence    tempo  time_signature  \
0        0.0322          0.000001    0.3580    0.715   87.917               4
1        0.9240          0.000006    0.1010    0.267   77.489               4
2        0.2100          0.000000    0.1170    0.120   76.332               4
3        0.9050          0.000071    0.1320    0.143  181.740               3
4        0.4690          0.000000    0.0829    0.167  119.949               4

  track_genre
0    acoustic
1    acoustic
2    acoustic
3    acoustic
4    acoustic
```

Panda Queries:

1. List artist, track_name, and popularity for songs that have a popularity greater than or equal to 99

In [17]:
```
# Query for songs with popularity >= 99
df_popular = df[df['popularity'] >= 99][['artists', 'track_name', 'popularity']]
print(df_popular)
```

```
                  artists                          track_name  popularity
20001  Sam SmithKim Petras           Unholy (feat. Kim Petras)         100
51664      BizarrapQuevedo  Quevedo: Bzrp Music Sessions, Vol. 52          99
81051  Sam SmithKim Petras           Unholy (feat. Kim Petras)         100
```

2. List artists with an average popularity of 92

In [18]:
```
# Group by artists and calculate the mean popularity
df_avg_pop = df.groupby('artists')['popularity'].mean().reset_index()

# Filter artists with average popularity of 92
df_avg_92 = df_avg_pop[df_avg_pop['popularity'] == 92]
print(df_avg_92)
```

```
               artists  popularity
11487      Harry Styles        92.0
22842  RemaSelena Gomez        92.0
```

3. List the Top 10 most energetic genres

In [19]:
```
# Group by track_genre and calculate the mean energy, then sort by energy and select
df_top_genres_energy = df.groupby('track_genre')['energy'].mean().reset_index()
df_top_genres_energy = df_top_genres_energy.sort_values(by='energy', ascending=False)
print(df_top_genres_energy)
```

```
      track_genre    energy
22    death-metal  0.931470
42      grindcore  0.924201
72      metalcore  0.914485
46          happy  0.910971
49      hardstyle  0.901246
27  drum-and-bass  0.876635
6     black-metal  0.874897
50    heavy-metal  0.874003
78          party  0.871237
61         j-idol  0.868677
```

4. How many tracks is Bad Bunny on?

In [21]:
```
# Fill NaN values with an empty string and then filter for 'Bad Bunny'
df_bad_bunny = df[df['artists'].fillna('').str.contains('Bad Bunny', case=False)]

# Output the number of tracks
print(f"Number of tracks Bad Bunny is on: {df_bad_bunny.shape[0]}")
```

```
Number of tracks Bad Bunny is on: 416
```

5. Show the top 10 genres in terms of popularity sorted by their most popular track

In [22]:
```
# Group by track_genre and calculate the max popularity, then sort by popularity and
df_top_genres_pop = df.groupby('track_genre')['popularity'].max().reset_index()
df_top_genres_pop = df_top_genres_pop.sort_values(by='popularity', ascending=False).h
print(df_top_genres_pop)
```

```
   track_genre  popularity
80         pop         100
20       dance         100
51     hip-hop          99
68      latino          98
89   reggaeton          98
30         edm          98
67       latin          98
88      reggae          98
79       piano          96
90        rock          96
```

In [ ]:
```

```