

Register the CSV file with Athena

```
In [ ]: import boto3
import sagemaker

sess = sagemaker.Session()
bucket = sess.default_bucket()
role = sagemaker.get_execution_role()
region = boto3.Session().region_name
```

```
In [ ]: ingest_create_athena_table_csv_passed = False
```

```
In [ ]: %store
```

Stored variables and their in-db values:

data_path	-> '/root/AAI-540/Module2/csv'
ingest_create_athena_db_mod2_passed	-> True
ingest_create_athena_table_csv_passed	-> True
s3_private_path_csv	-> 's3://sagemaker-us-east-1-004608622582/module2_dat
setup_dependencies_mod2_passed	-> True
setup_s3_bucket_passed	-> True

```
In [ ]: %store -r ingest_create_athena_db_mod2_passed
```

```
In [ ]: try:
    ingest_create_athena_db_mod2_passed
except NameError:
    print("+++++")
    print("[ERROR] YOU HAVE TO RUN ALL PREVIOUS NOTEBOOKS. You did not create the")
    print("+++++")
```

```
In [ ]: print(ingest_create_athena_db_mod2_passed)
```

True

```
In [ ]: if not ingest_create_athena_db_mod2_passed:
    print("+++++")
    print("[ERROR] YOU HAVE TO RUN ALL PREVIOUS NOTEBOOKS. You did not create the")
    print("+++++")
else:
    print("[OK]")
```

[OK]

```
In [ ]: %store -r s3_private_path_csv
```

```
In [ ]: try:
    s3_private_path_csv
except NameError:
    print("*****")
    print("[ERROR] PLEASE RE-RUN THE PREVIOUS COPY TSV TO S3 NOTEBOOK *****")
```

```
print("[ERROR] THIS NOTEBOOK WILL NOT RUN PROPERLY. *****")
print("*****")
```

```
In [ ]: print(s3_private_path_csv)
```

s3://sagemaker-us-east-1-004608622582/module2_data/csv

Import PyAthena

```
In [ ]: from pyathena import connect
```

```
In [ ]: # Set S3 staging directory -- this is a temporary directory used for Athena queries
s3_staging_dir = "s3://{0}/athena/staging".format(bucket)
```

```
In [ ]: # Set Athena parameters
database_name = "mod2_db"
table_name_csv = "music"
```

```
In [ ]: conn = connect(region_name=region, s3_staging_dir=s3_staging_dir)
```

```
In [ ]: # Create Statement
statement = """CREATE EXTERNAL TABLE IF NOT EXISTS {}.{}(
    track_id STRING,
    artists STRING,
    popularity INT,
    duration_ms INT,
    explicit BOOLEAN,
    danceability FLOAT,
    energy FLOAT,
    key INT,
    loudness FLOAT,
    mode INT,
    speechiness FLOAT,
    acousticness FLOAT,
    instrumentalness FLOAT,
    liveness FLOAT,
    valence FLOAT,
    tempo FLOAT,
    time_signature INT,
    track_genre STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\\n' LOCATION '{
TBLPROPERTIES ('skip.header.line.count'='1')""".format(
    database_name, table_name_csv, s3_private_path_csv
)

print(statement)
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS mod2_db.music(
    track_id STRING,
    artists STRING,
    popularity INT,
    duration_ms INT,
    explicit BOOLEAN,
    danceability FLOAT,
    energy FLOAT,
    key INT,
    loudness FLOAT,
    mode INT,
    speechiness FLOAT,
    acousticness FLOAT,
    instrumentalness FLOAT,
    liveness FLOAT,
    valence FLOAT,
    tempo FLOAT,
    time_signature INT,
    track_genre STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' LOCATION 's3://sagemaker-us-east-1-004608622582/module2_data/csv'
TBLPROPERTIES ('skip.header.line.count'='1')
```

In []: `import pandas as pd`

```
pd.read_sql(statement, conn)
```

/tmp/ipykernel_1867/3803073958.py:3: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.

```
pd.read_sql(statement, conn)
```

Out[]: —

Verify that Table has been created successfully

In []: `statement = "SHOW TABLES in {}".format(database_name)`

```
df_show = pd.read_sql(statement, conn)
df_show.head(5)
```

/tmp/ipykernel_1867/2201015668.py:3: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.

```
df_show = pd.read_sql(statement, conn)
```

Out[]: **tab_name**

```
0      music
```

In []: `if table_name_csv in df_show.values:`
 `ingest_create_athena_table_csv_passed = True`
`print(ingest_create_athena_table_csv_passed)`

True

In []: `%store ingest_create_athena_table_csv_passed`

Stored 'ingest_create_athena_table_csv_passed' (bool)

In []: `%store`

Stored variables and their in-db values:

data_path	-> '/root/AAI-540/Module2/csv'
ingest_create_athena_db_mod2_passed	-> True
ingest_create_athena_table_csv_passed	-> True
s3_private_path_csv	-> 's3://sagemaker-us-east-1-00460
8622582/module2_dat	
setup_dependencies_mod2_passed	-> True
setup_s3_bucket_passed	-> True

Run A Sample Query

In []: `artists = "Jason Mraz"`

```
statement = """SELECT * FROM {}.{}
WHERE artists = '{}' LIMIT 10""".format(
    database_name, table_name_csv, artists
)

print(statement)
```

```
SELECT * FROM mod2_db.music
WHERE artists = 'Jason Mraz' LIMIT 10
```

In []: `df = pd.read_sql(statement, conn)`
`df.head(5)`

/tmp/ipykernel1_1867/2446512133.py:1: UserWarning: pandas only supports SQLAlchemy connectable (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)
```

Out[]:

	track_id	artists	popularity	duration_ms	explicit	danceability	ene
0	1EzrEOxmMH3G43AXT1y7pA	Jason Mraz	80	242946	False	0.703	0.
1	5ivF4eQBqJiVL5IAE9jRyl	Jason Mraz	69	240165	False	0.483	0.
2	3S0OXQeoh0w6AY8WQVckRW	Jason Mraz	75	242946	False	0.703	0.
3	0BUuuEvNa5T4lMaewyiudB	Jason Mraz	0	216386	False	0.572	0.
4	3Hn3LfhrQOaKihdCibJsTs	Jason Mraz	0	231266	False	0.796	0.

```
In [ ]: if not df.empty:
        print("[OK]")
    else:
        print("+++++")
        print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN PREVIOUS")
        print("+++++")
```

[OK]

Review in GLUE Catalog

```
In [ ]: from IPython.core.display import display, HTML

display(
    HTML(
        '<b>Review <a target="top" href="https://console.aws.amazon.com/glue/home?r
        region
    )
)
```

/tmp/ipykernel_1867/4130537117.py:1: DeprecationWarning: Importing display from IPython.core.display is deprecated since IPython 7.14, please import from IPython display

```
from IPython.core.display import display, HTML
```

Review AWS Glue Catalog

```
In [ ]: %%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:shutdown">

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

Shutting down your kernel for this notebook to release resources.

In []: