Setup the Athena Database

In [ ]:
```
import boto3
import sagemaker

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

In [ ]:
```
# create bool statements to hold inside store
ingest_create_athena_db_passed = False
```

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

In [ ]:
```
try:
    data_path
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(data_path)
```

```
/root/AAI-540/Module2/csv
```

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 [ ]:
```
database_name = "mod2_db"
```

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

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

In [ ]:
```
statement = "CREATE DATABASE IF NOT EXISTS {}".format(database_name)
print(statement)
```

```
CREATE DATABASE IF NOT EXISTS mod2_db
```

In [ ]:
```
import pandas as pd

pd.read_sql(statement, conn)
```

```
/tmp/ipykernel_1819/3803073958.py:3: 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.
  pd.read_sql(statement, conn)
```

Out[ ]: —

Verify that the databases are created successfully

In [ ]:
```
statement = "SHOW DATABASES"

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

```
/tmp/ipykernel_1819/3999478089.py:3: 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_show = pd.read_sql(statement, conn)
```

Out[ ]:

	database_name
0	default
1	dsoaws
2	mod2_db

In [ ]:
```
if database_name in df_show.values:
    ingest_create_athena_db_mod2_passed = True
```

Store variables and release notebook

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

```
Stored 'ingest_create_athena_db_mod2_passed' (bool)
```

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 [ ]: