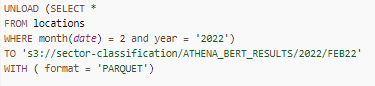
**GDELT Data – Sector Classification**

This document outlines the working of the Python Notebook for classifying the GDELT data on AWS into different sectors. The notebook which currently resides on AWS Sagemaker studio is able to fetch the data from S3 bucket, predict the sector for each row of data and write the result back to the any location on AWS S3. The steps involved are as follows:

1. Generating Parquet Files for Each Month using AWS Athena Query Service:

Due to the large size of the dataset, it is queried one month at a time using the AWS Athena service. Each month of data is partitioned in to 30 Parquet files and stored into user specified S3 Bucket. Partitioning the data into small chunks removes huge RAM requirements when the data is later read into Pandas Dataframes.

Sample SQL Query:



1. Reading parquet files from S3 buckets into Pandas dataframe:

The following functions which are present in the **Sagemaker notebook** are used to read all the parquet files for a month in a loop.

**‘return\_all\_files(bucket, parquet\_folder\_location)’ 🡪** Returns a list of all file keys present in a folder on the S3 bucket.

**‘read\_parquet\_file(bucket, file\_key)’ 🡪** Reads the parquet file identified by the file\_key and returns the data in a dataframe format.

1. Pre-processing the dataframe to get ready for predictions:

The function **‘obtain\_text(row)’** takes into input a row of the Pandas dataframe and return a formatted string to be used as the input for the prediction model.

1. Predictions:

The function **‘make\_predictions(df\_batch, batch\_size)’** takes as the argument the dataframe and batch size. The batch size indicates the batch size or the number of rows to be processed at a time by the BERT model. The function returns a list of integers corresponding to the sector predictions of each row. The list is appended to a master list and thereafter appended to the dataframe.

1. Storing results back into S3 bucket:

The function **‘save\_df\_to\_s3(df, bucket\_save, file\_location, year, month, counter, file\_format)’** takes as arguments the resultant dataframe, location of the S3 bucket in which the file is to be stored, location of the folder where the file is to be stored. The function converts the dataframe back to a parquet format for efficient storage and writes the file back to AWS S3 bucket.

**USER INPUTS:**

For convenience, the user can specify the locations of the folders from where the data is to be read as well as the location of the folder where the data is to be stored in a single cell of the notebook. All the other functions in the notebook use these inputs for identifying the files on which the predictions are to be made.

Sample:

