Authentication - you will need to upload the service account key json file when prompted

```
from google.colab import auth
auth.authenticate_service_account()
```

```
Upload the private key for your service account.

See the guide at https://cloud.google.com/iam/docs/creating-managing-service-a
```

Choose File no file selected Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-1-0302f90c5dfb> in <cell line: 2>()
      1 from google.colab import auth
----> 2 auth.authenticate_service_account()
```

2 frames

```
/usr/local/lib/python3.10/dist-packages/google/colab/files.py in
_upload_files(multiple)
    161   files = _collections.defaultdict(bytes)
    162
--> 163   while result['action'] != 'complete':
    164     result = _output.eval_js(
    165         'google.colab._files._uploadFilesContinue("
{output_id}")'.format(

TypeError: 'NoneType' object is not subscriptable
```

```
import pandas as pd
from google.cloud import bigquery

project_name='weatherlink-404323'
client = bigquery.Client()

year_start = 2015
year_stop = 2025

# Initialize a dictionary to store the schema for each table
table_schemas = {}

# Initialize a set to store common columns
common_columns_set = None

# Initialize a dictionary to store unique columns for each table
```

```
unique_columns_dict = {}

# Initialize a list to store DataFrames for each year
dfs = []

for i in range(year_start, year_stop):
    table_name = f"bigquery-public-data.nhtsa_traffic_fatalities. accident_{i}"


    try:
        # Fetch the schema (column information) for each table
        if table_name not in table_schemas:
            table = client.get_table(table_name)
            table_schemas[table_name] = set([field.name for field in table.schema])

        # If this is the first DataFrame, initialize the set with its columns
        if common_columns_set is None:
            common_columns_set = set(table_schemas[table_name])
        else:
            # Update the set to include only columns present in both DataFrames
            common_columns_set.intersection_update(table_schemas[table_name])

        # Update the set to include only columns not present in other DataFrames
        unique_columns_dict[table_name] = table_schemas[table_name].difference(comm

        print(f"Table name {table_name} has {len(table_schemas[table_name])} cols")

        # Fetch the data and add it to a DataFrame
        query = f"SELECT * FROM `{table_name}`"
        df = client.query(query).to_dataframe()

        # Add the 'year' column to the DataFrame
        df['year'] = i

        # Calculate the FIPS code
        df['county'] = df['county'].astype(str)
        df['county'] = df['county'].str.zfill(3)
        df['state_number'] = df['state_number'].astype(str)
        df['state_number'] = df['state_number'].str.zfill(2)

        df['geoid'] = df['state_number'] + df['county']

        # Keep only the common columns
        df = df[list(common_columns_set) + ['year'] + ['geoid']]

        # Append the DataFrame to the list
        dfs.append(df)

    except Exception as e:
```

```
    except Exception as e:
        print(e)
        print(f"Table {table_name} not found")

# Print unique columns for each table
for i in range(year_start, year_stop):
    table_name = f"bigquery-public-data.nhtsa_traffic_fatalities. accident_{i}"
    try:
        print(f"Unique columns in {table_name}: {unique_columns_dict[table_name]}")
    except Exception as e:
        print(e)
        print(f"Table {table_name} not found")

# Concatenate all DataFrames into a master DataFrame
master_df = pd.concat(dfs, ignore_index=True)

# Print the master DataFrame
print("\nMaster DataFrame:")
print(master_df)
```

```
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2015 has
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2016 has
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2017 has
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2018 has
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2019 has
Table name bigquery-public-data.nhtsa_traffic_fatalities. accident_2020 has
404 GET https://bigquery.googleapis.com/bigquery/v2/projects/bigquery-public
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2021 not founc
404 GET https://bigquery.googleapis.com/bigquery/v2/projects/bigquery-public
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2022 not founc
404 GET https://bigquery.googleapis.com/bigquery/v2/projects/bigquery-public
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2023 not founc
404 GET https://bigquery.googleapis.com/bigquery/v2/projects/bigquery-public
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2024 not founc
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
Unique columns in bigquery-public-data.nhtsa_traffic_fatalities. accident_20
'bigquery-public-data.nhtsa_traffic_fatalities. accident_2021'
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2021 not founc
'bigquery-public-data.nhtsa_traffic_fatalities. accident_2022'
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2022 not founc
'bigquery-public-data.nhtsa_traffic_fatalities. accident_2023'
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2023 not founc
'bigquery-public-data.nhtsa_traffic_fatalities. accident_2024'
Table bigquery-public-data.nhtsa_traffic_fatalities. accident_2024 not founc

Master DataFrame:
        number_of_persons_in_motor_vehicles_in_transport_mvit milepoint_name
0                                                       1              489
```

```
1                                                   2           680
2                                                   1           110
3                                                   3           639
4                                                   2           760
...                                               ...           ...
203460                                              1            74
203461                                              1      Not Reportec
203462                                              1            27
203463                                              1           105
203464                                              1      Not Reportec

        number_of_forms_submitted_for_persons_in_motor_vehicles  \
0                                                   1
1                                                   2
2                                                   1
3                                                   3
4                                                   2
...                                               ...
203460                                              1
203461                                              1
203462                                              1
203463                                              1
203464                                              1

              ownership_name  \
0       State Highway Agency
```

Double-click (or enter) to edit

we are going to upload this dataframe to Google Big Query in our 'Data Warehouse', and then select a number of columns to compare for analysis.

```
pip install pandas_gbq

    Requirement already satisfied: pandas_gbq in /usr/local/lib/python3.10/dist-pa
    Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-pa
    Requirement already satisfied: db-dtypes<2.0.0,>=0.3.1 in /usr/local/lib/pytho
    Requirement already satisfied: numpy>=1.16.6 in /usr/local/lib/python3.10/dist
    Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.10/dis
    Requirement already satisfied: pyarrow<10.0dev,>=3.0.0 in /usr/local/lib/pytho
    Requirement already satisfied: pydata-google-auth in /usr/local/lib/python3.10
    Requirement already satisfied: google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,
    Requirement already satisfied: google-auth>=1.25.0 in /usr/local/lib/python3.1
    Requirement already satisfied: google-auth-oauthlib>=0.0.1 in /usr/local/lib/p
    Requirement already satisfied: google-cloud-bigquery!=2.4.*,<4.0.0dev,>=1.27.2
    Requirement already satisfied: google-cloud-bigquery-storage<3.0.0dev,>=1.1.0 
    Requirement already satisfied: packaging>=17.0 in /usr/local/lib/python3.10/di
    Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in /
    Requirement already satisfied: protobuf!=3.20.0,!=3.20.1,!=4.21.0,!=4.21.1,!=4
    Requirement already satisfied: requests<3.0.0.dev0,>=2.18.0 in /usr/local/lib/
    Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python
    Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3
    Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-pa
    Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist
    Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/pyth
    Requirement already satisfied: grpcio<2.0dev,>=1.47.0 in /usr/local/lib/python
    Requirement already satisfied: proto-plus<2.0.0dev,>=1.15.0 in /usr/local/lib/
    Requirement already satisfied: google-cloud-core<3.0.0dev,>=1.6.0 in /usr/loca
    Requirement already satisfied: google-resumable-media<3.0dev,>=0.6.0 in /usr/l
    Requirement already satisfied: python-dateutil<3.0dev,>=2.7.2 in /usr/local/li
    Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: grpcio-status<2.0.dev0,>=1.33.2 in /usr/local/l
    Requirement already satisfied: google-crc32c<2.0dev,>=1.0 in /usr/local/lib/py
    Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in /usr/local/lib/python3.
    Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/pyth
    Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10
    Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/di
```

```
from pandas_gbq import to_gbq
from pandas_gbq.schema import generate_bq_schema
from io import StringIO

client = bigquery.Client()

project_id='weatherlink-404323'
dataset_id = 'weatherlink_master'
table_id = 'accident_master'

# This is a weird work around to get the dataframe acceptable for upload
# temporarily store the dataframe as a csv in a string variable
temp_csv_string = master_df.to_csv(sep=";", index=False)
temp_csv_string_IO = StringIO(temp_csv_string)
# create new dataframe from string variable
new_df = pd.read_csv(temp_csv_string_IO, sep=";")

to_gbq(new_df, f"{dataset_id}.{table_id}", project_id=project_id, if_exists='repl
```

```
    <ipython-input-20-e6e8c136e94d>:16: DtypeWarning: Columns (6,11,57,82,84) hav
      new_df = pd.read_csv(temp_csv_string_IO, sep=";")
    100%|██████████| 1/1 [00:00<00:00, 6765.01it/s]
```

Lets test by querying our new master table

```
sql = f" SELECT COUNT(*) FROM {project_id}.{dataset_id}.{table_id}"

result = client.query(sql)

print(result)
```

```
    QueryJob<project=weatherlink-404323, location=US, id=a28bd2f3-62ce-40c4-9140-
```