ETL_Project brought to you by the Fearsome Foursome

Lora Brown, Stephen Peters, Nathan Putnam, Eliza De Barros

Identify datasets and perform ETL on the data.

Sources of data:

Source link for coffee data:
https://www.kaggle.com/yamaerenay/ico-coffee-dataset-worldwide?select=domestic-consumption

Source link for alcohol data:
https://www.kaggle.com/marcospessotto/happiness-and-alcohol-consumption

Source link for suicide data:
https://www.kaggle.com/russellyates88/suicide-rates-overview-1985-to-2016

Transformations needed for this data (cleaning, joining, filtering, aggregating, etc).
Cleaning, filtering and joining

1. Dropping unneeded columns
2. Renaming columns
3. Connecting tables with joins
4. Creating tables with joins

The type of final production database to load the data into (relational or non-relational).

- Postgress SQL

The final tables or collections that will be used in the production database.

- coffee_df_2016
- happinessAlcohol_df
- suicide_df_2016
- final_db

Project Report

* **E**xtract: your original data sources and how the data was formatted (CSV, JSON, pgAdmin 4, etc).

Source data files are CSV files in the Resources directory:

- Domestic-consumption.csv
- HappinessAlcoholConsumption.csv
- SuicideRates.csv

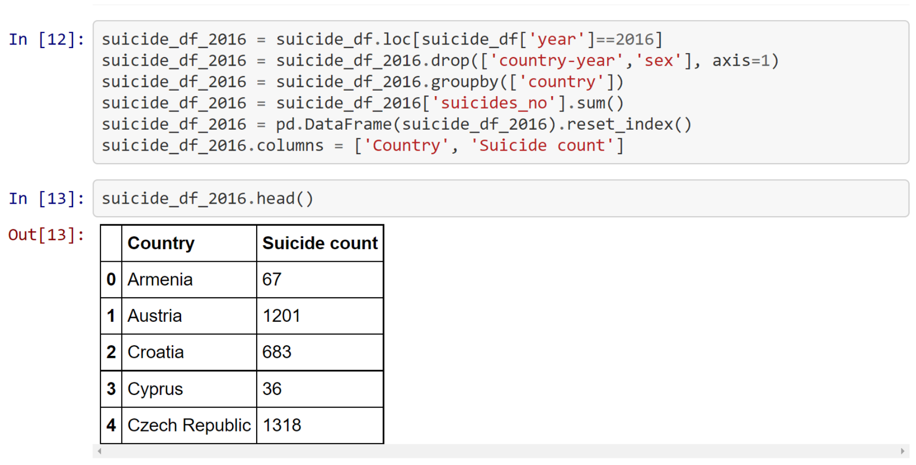
* **T**ransform: what data cleaning or transformation was required.

Transformations done in Python, using the ETL.ipynb jupyter notebook.

The user’s Postgres user password needs to be in config.py for the script to run.

Transform suicide data:

1. Drop all years but 2016
2. Drop unneeded columns
3. Group by country
4. Sum suicide numbers

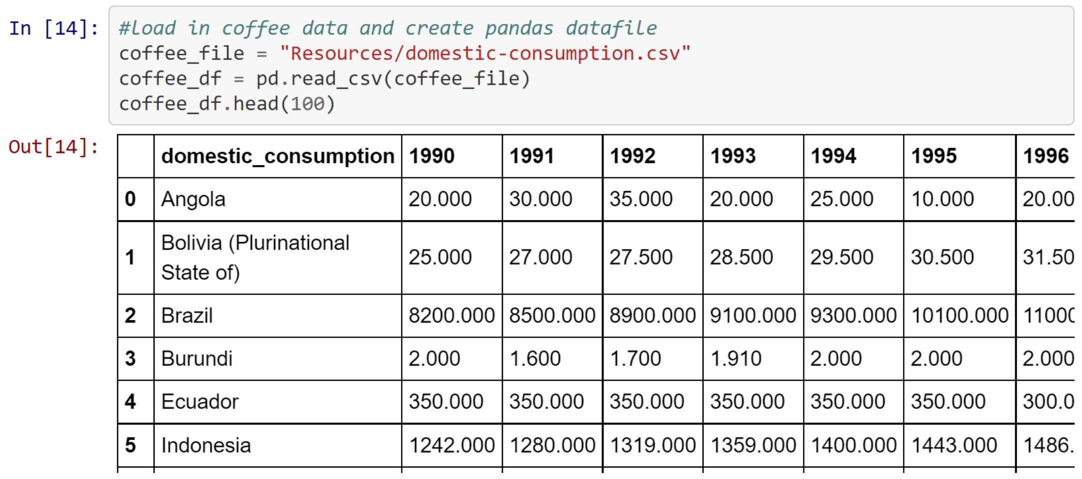


```
In [12]: suicide_df_2016 = suicide_df.loc[suicide_df['year']==2016]
         suicide_df_2016 = suicide_df_2016.drop(['country-year','sex'], axis=1)
         suicide_df_2016 = suicide_df_2016.groupby(['country'])
         suicide_df_2016 = suicide_df_2016['suicides_no'].sum()
         suicide_df_2016 = pd.DataFrame(suicide_df_2016).reset_index()
         suicide_df_2016.columns = ['Country', 'Suicide count']

In [13]: suicide_df_2016.head()

Out[13]:
```

	Country	Suicide count
0	Armenia	67
1	Austria	1201
2	Croatia	683
3	Cyprus	36
4	Czech Republic	1318

Transform Coffee data:

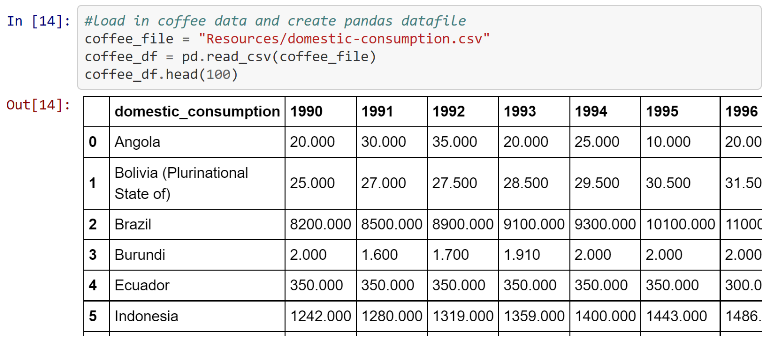
1. Drop all years but 2016



```
In [14]: #load in coffee data and create pandas datafile
         coffee_file = "Resources/domestic-consumption.csv"
         coffee_df = pd.read_csv(coffee_file)
         coffee_df.head(100)
```

Out[14]:

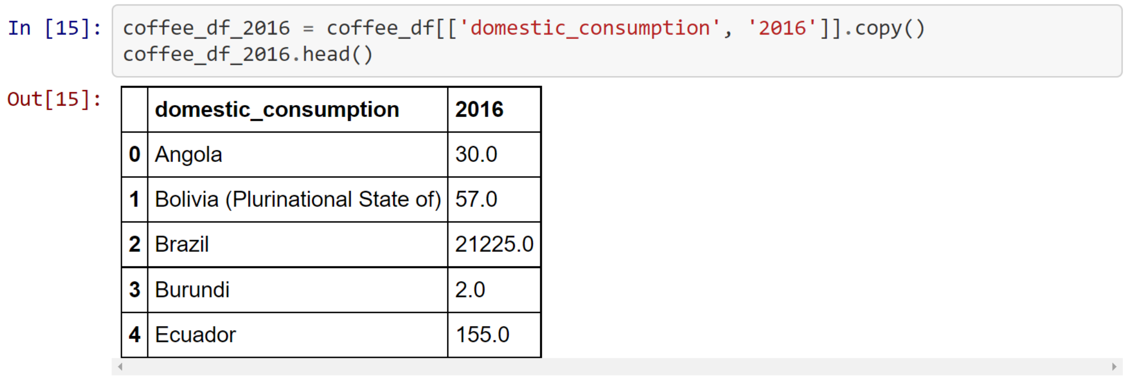
	domestic_consumption	1990	1991	1992	1993	1994	1995	1996
0	Angola	20.000	30.000	35.000	20.000	25.000	10.000	20.00
1	Bolivia (Plurinational State of)	25.000	27.000	27.500	28.500	29.500	30.500	31.50
2	Brazil	8200.000	8500.000	8900.000	9100.000	9300.000	10100.000	11000
3	Burundi	2.000	1.600	1.700	1.910	2.000	2.000	2.000
4	Ecuador	350.000	350.000	350.000	350.000	350.000	350.000	300.0
5	Indonesia	1242.000	1280.000	1319.000	1359.000	1400.000	1443.000	1486.



```
In [14]: #load in coffee data and create pandas datafile
         coffee_file = "Resources/domestic-consumption.csv"
         coffee_df = pd.read_csv(coffee_file)
         coffee_df.head(100)
```

Out[14]:

	domestic_consumption	1990	1991	1992	1993	1994	1995	1996
0	Angola	20.000	30.000	35.000	20.000	25.000	10.000	20.00
1	Bolivia (Plurinational State of)	25.000	27.000	27.500	28.500	29.500	30.500	31.50
2	Brazil	8200.000	8500.000	8900.000	9100.000	9300.000	10100.000	11000
3	Burundi	2.000	1.600	1.700	1.910	2.000	2.000	2.000
4	Ecuador	350.000	350.000	350.000	350.000	350.000	350.000	300.0
5	Indonesia	1242.000	1280.000	1319.000	1359.000	1400.000	1443.000	1486.



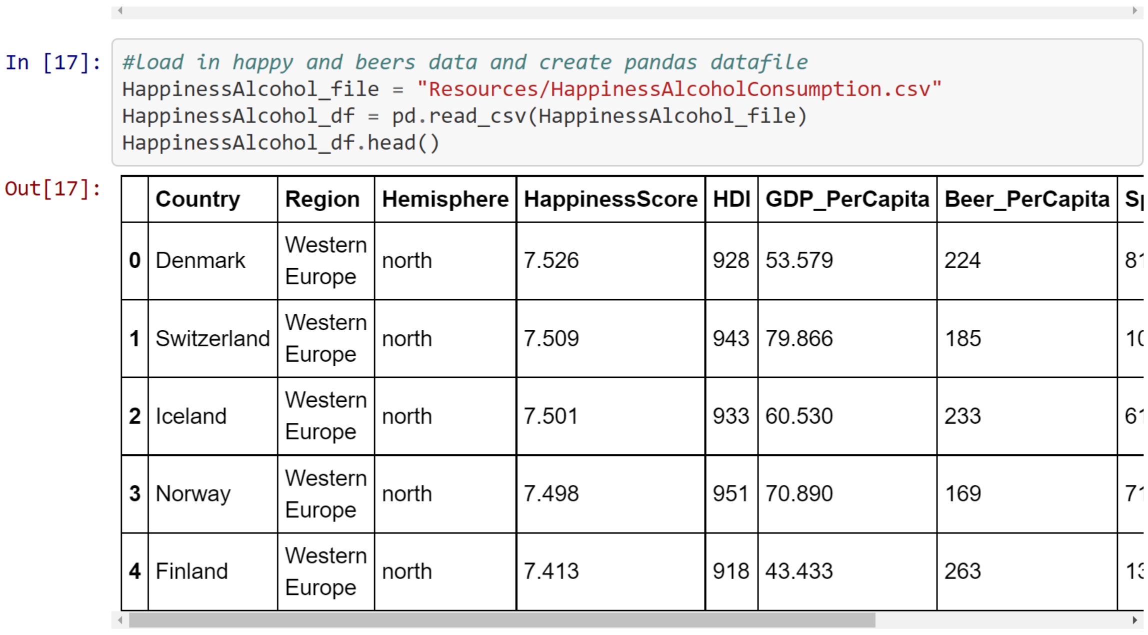
```
In [15]: coffee_df_2016 = coffee_df[['domestic_consumption', '2016']].copy()
         coffee_df_2016.head()
```

Out[15]:

	domestic_consumption	2016
0	Angola	30.0
1	Bolivia (Plurinational State of)	57.0
2	Brazil	21225.0
3	Burundi	2.0
4	Ecuador	155.0

Transform Alcohol consumption data:

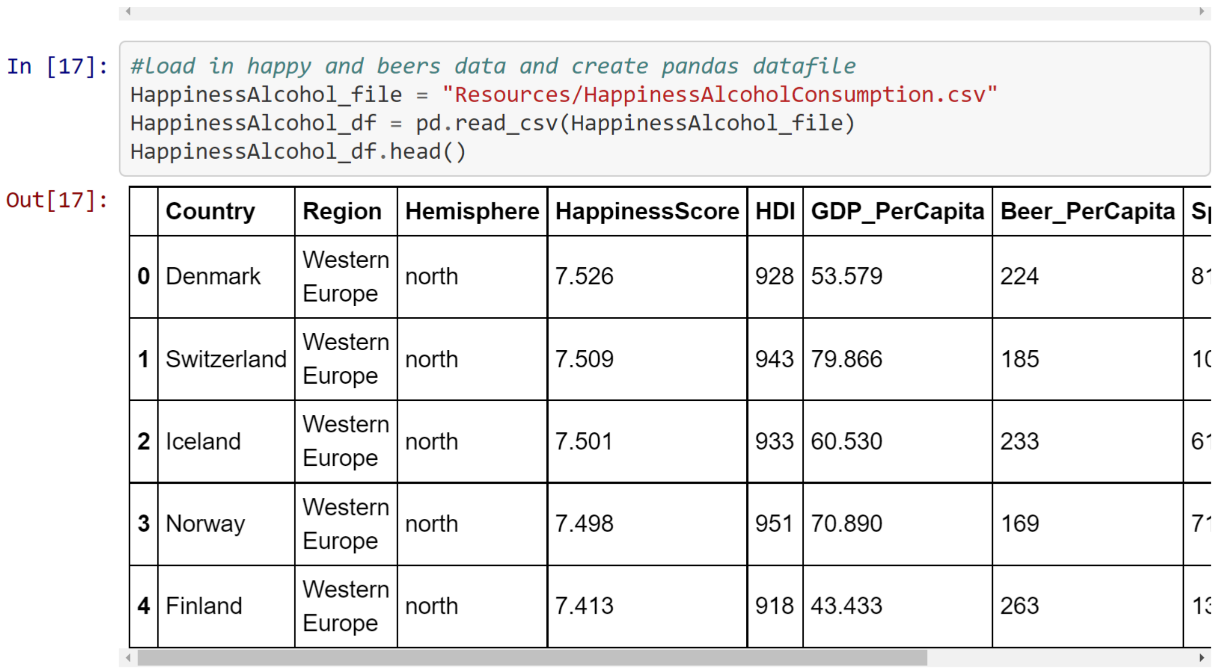
1. Drop region and hemisphere
2. Rename columns



In [17]:
```
#load in happy and beers data and create pandas datafile
HappinessAlcohol_file = "Resources/HappinessAlcoholConsumption.csv"
HappinessAlcohol_df = pd.read_csv(HappinessAlcohol_file)
HappinessAlcohol_df.head()
```

Out[17]:

	Country	Region	Hemisphere	HappinessScore	HDI	GDP_PerCapita	Beer_PerCapita	S
0	Denmark	Western Europe	north	7.526	928	53.579	224	8
1	Switzerland	Western Europe	north	7.509	943	79.866	185	1
2	Iceland	Western Europe	north	7.501	933	60.530	233	6
3	Norway	Western Europe	north	7.498	951	70.890	169	7
4	Finland	Western Europe	north	7.413	918	43.433	263	1



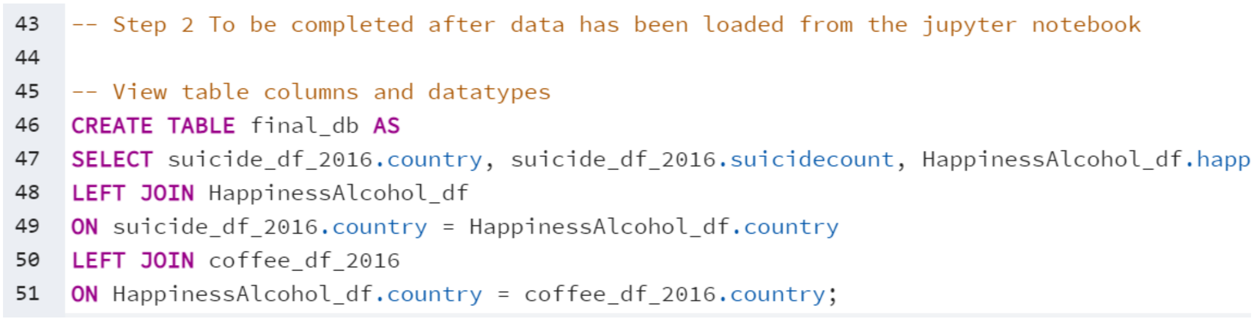
In [17]:
```
#load in happy and beers data and create pandas datafile
HappinessAlcohol_file = "Resources/HappinessAlcoholConsumption.csv"
HappinessAlcohol_df = pd.read_csv(HappinessAlcohol_file)
HappinessAlcohol_df.head()
```

Out[17]:

	Country	Region	Hemisphere	HappinessScore	HDI	GDP_PerCapita	Beer_PerCapita	S
0	Denmark	Western Europe	north	7.526	928	53.579	224	8
1	Switzerland	Western Europe	north	7.509	943	79.866	185	1
2	Iceland	Western Europe	north	7.501	933	60.530	233	6
3	Norway	Western Europe	north	7.498	951	70.890	169	7
4	Finland	Western Europe	north	7.413	918	43.433	263	1

Transform SQL tables with two joins

1. Left joined the suicide_df_2016 with the happinessAlcohol_df on the country names
3. Left joined the new table with the coffee_df_2016 on the country
4. From the joins we created a final table titled final_db

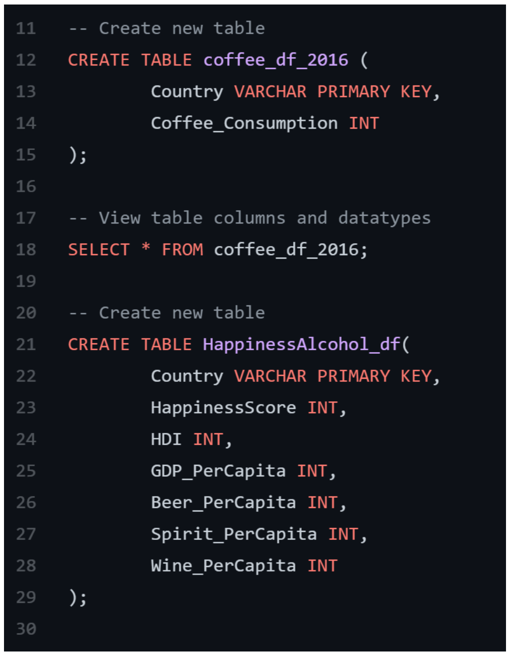


```sql
-- Step 2 To be completed after data has been loaded from the jupyter notebook

-- View table columns and datatypes
CREATE TABLE final_db AS
SELECT suicide_df_2016.country, suicide_df_2016.suicidecount, HappinessAlcohol_df.happ
LEFT JOIN HappinessAlcohol_df
ON suicide_df_2016.country = HappinessAlcohol_df.country
LEFT JOIN coffee_df_2016
ON HappinessAlcohol_df.country = coffee_df_2016.country;
```

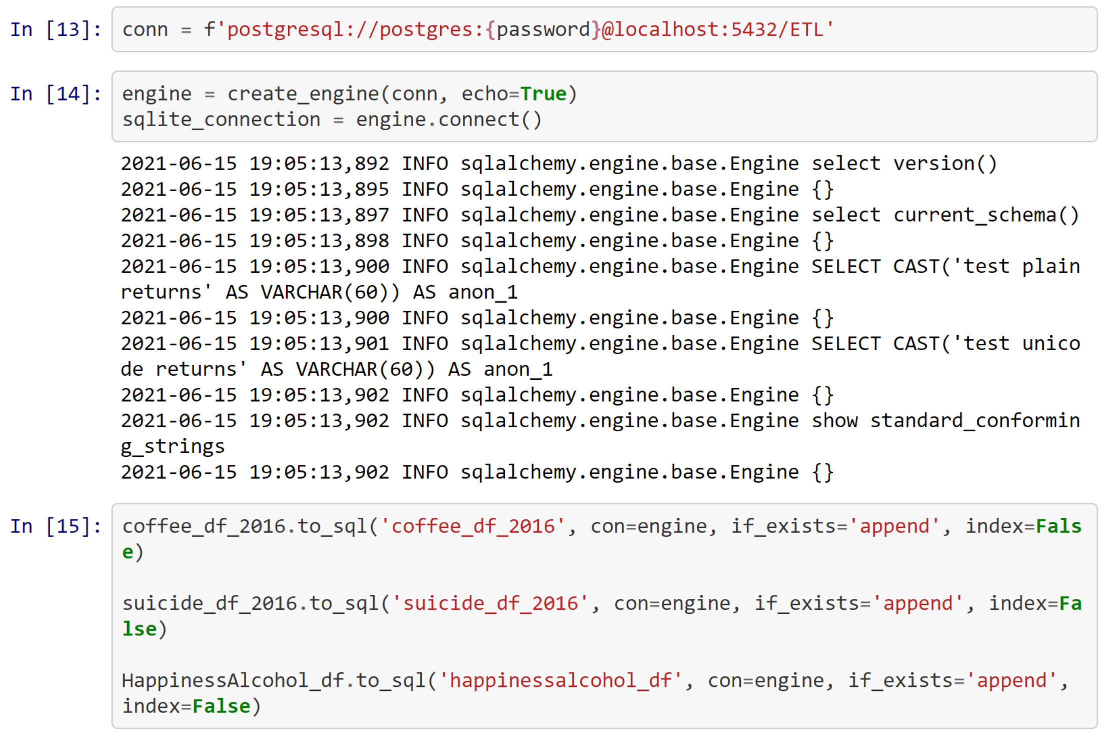
* **L**oad: the final database, tables/collections, and why this was chosen.

Load data into SQL

1. Created tables within SQL
2. In the jupyter notebook we created an engine and loaded the cleaned databases from Pandas to SQL



```sql
-- Create new table
CREATE TABLE coffee_df_2016 (
        Country VARCHAR PRIMARY KEY,
        Coffee_Consumption INT
);

-- View table columns and datatypes
SELECT * FROM coffee_df_2016;

-- Create new table
CREATE TABLE HappinessAlcohol_df(
        Country VARCHAR PRIMARY KEY,
        HappinessScore INT,
        HDI INT,
        GDP_PerCapita INT,
        Beer_PerCapita INT,
        Spirit_PerCapita INT,
        Wine_PerCapita INT
);

```



```
In [13]: conn = f'postgresql://postgres:{password}@localhost:5432/ETL'

In [14]: engine = create_engine(conn, echo=True)
         sqlite_connection = engine.connect()

         2021-06-15 19:05:13,892 INFO sqlalchemy.engine.base.Engine select version()
         2021-06-15 19:05:13,895 INFO sqlalchemy.engine.base.Engine {}
         2021-06-15 19:05:13,897 INFO sqlalchemy.engine.base.Engine select current_schema()
         2021-06-15 19:05:13,898 INFO sqlalchemy.engine.base.Engine {}
         2021-06-15 19:05:13,900 INFO sqlalchemy.engine.base.Engine SELECT CAST('test plain
         returns' AS VARCHAR(60)) AS anon_1
         2021-06-15 19:05:13,900 INFO sqlalchemy.engine.base.Engine {}
         2021-06-15 19:05:13,901 INFO sqlalchemy.engine.base.Engine SELECT CAST('test unico
         de returns' AS VARCHAR(60)) AS anon_1
         2021-06-15 19:05:13,902 INFO sqlalchemy.engine.base.Engine {}
         2021-06-15 19:05:13,902 INFO sqlalchemy.engine.base.Engine show standard_conformin
         g_strings
         2021-06-15 19:05:13,902 INFO sqlalchemy.engine.base.Engine {}

In [15]: coffee_df_2016.to_sql('coffee_df_2016', con=engine, if_exists='append', index=Fals
         e)

         suicide_df_2016.to_sql('suicide_df_2016', con=engine, if_exists='append', index=Fa
         lse)

         HappinessAlcohol_df.to_sql('happinessalcohol_df', con=engine, if_exists='append',
         index=False)
```

Python code expects the user to have an existing Postgres database named “ETL”.

Narrative:

- Only data from 2016
- 4 tables are created:
  - Coffee consumption by country
  - Alcohol consumption by country (includes happiness score)
  - Suicide rates by country
  - Final_db with joined data from all three tables
- Note: our coffee data source did not cover as many countries as the other data sources resulting in many null values in our coffee_consumption column.

The Why:

- Simplify: only one year. 2016 chosen because the alcohol table only had data from that year.
- Simplify: relate by country, country is the primary key for our tables

Please upload the report to Github and submit a link to Bootcampspot.

On Github: https://github.com/nathanPM104/ETL_Project-.git