Project Report

World Energy Consumption

DS8003 | Mgt of Big Data and Tools

Supervisor – Dr. Roy Kucukates

Amarpreet Kaur - 501213603

Kartikey Chauhan - 501259284

Ruchi Parmar – 501034872

Table of Contents

World Energy Consumption

The evolution of energy resources has significantly influenced human history in recent centuries. From the initial discovery of fossil fuels to the subsequent diversification into nuclear, hydropower, and various renewable technologies, there has been a profound impact not only on the sources of energy but also on the scale of production and consumption.

This report concentrates on the magnitude of energy consumption, examining both overall energy and electricity usage. It delves into a per capita analysis across countries and explores the dynamic shifts in energy consumption trends. Additionally, it is noteworthy that nations are increasingly directing their investments towards renewable technologies, as evidenced by the growing shares in this sector. We have utilized Big data management tools like Hadoop MapReduce, Spark, Hive Data Query layer, Apache airflow etc. along with visualization tools like Plotly, to understand the workflow of Data Analysis projects using Big Data Management tools.

1. PROBLEM DEFINITION

The world's energy dynamics have witnessed significant shifts over the past 50 years. With advancements in technology, emergence of new energy sources, and the increasing awareness about climate change, it has become imperative to understand and analyze these patterns.

In this project we analyze energy production and consumption trends for different regions and countries over the past few decades, to understand the world’s energy consumption. We are exploring different factors such as -

1. How has global energy consumption changed over the past three decades, and what insights can be drawn from the significant surge in 2021?
2. What lessons can be learned from the top 15 countries leading the clean energy transition, and how can smaller nations facing energy challenges benefit from these experiences?
3. How has the consumption of various energy sources evolved for the top consumers?
4. Among the top electricity generators, how do their reliance on renewable versus fossil sources vary? Where does the highest electricity generation have the maximum share in terms of primary energy sources?
5. What insights emerge from the intricate relationship between global energy consumption, population size, and per capita energy use?

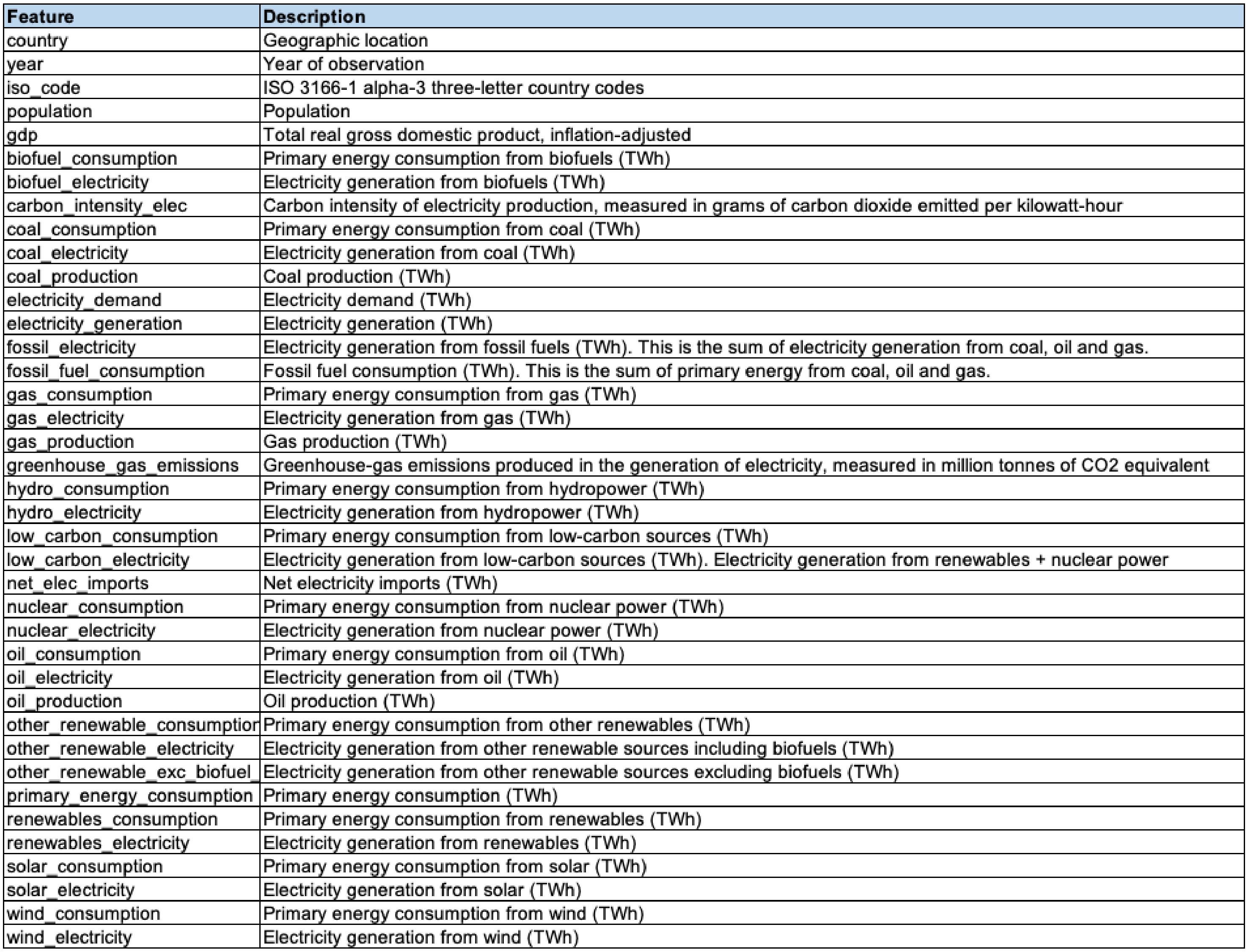
2. DATASET DESCRIPTION

i. Attributes Description

We used a csv dataset sourced from Kaggle - World Energy Consumption. As mentioned by the author, the dataset was originally collected by Hannah Ritchie, Max Roser and Edouard Mathieu. It consists of 17432 rows and 122 features with information on the world's energy consumption on a country level as well as a region level.

These features consists of change and change percentage measured in production and consumption of different energy sources (i.e. Coal, gas, oil, wind, nuclear, etc.) over the years from 1900 till 2022. The dataset also captures crucial data of different countries such as GDP, population, and consumption of sources per capita.

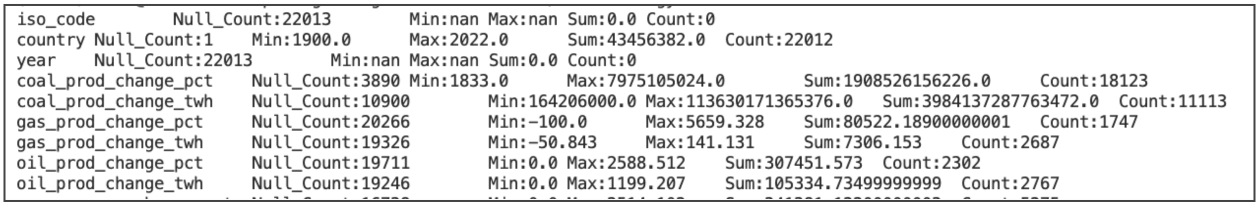
Few of the important features that are used alongside their description: (TWh - *Terawatt-hours*)



Feature	Description
country	Geographic location
year	Year of observation
iso_code	ISO 3166-1 alpha-3 three-letter country codes
population	Population
gdp	Total real gross domestic product, inflation-adjusted
biofuel_consumption	Primary energy consumption from biofuels (TWh)
biofuel_electricity	Electricity generation from biofuels (TWh)
carbon_intensity_elec	Carbon intensity of electricity production, measured in grams of carbon dioxide emitted per kilowatt-hour
coal_consumption	Primary energy consumption from coal (TWh)
coal_electricity	Electricity generation from coal (TWh)
coal_production	Coal production (TWh)
electricity_demand	Electricity demand (TWh)
electricity_generation	Electricity generation (TWh)
fossil_electricity	Electricity generation from fossil fuels (TWh). This is the sum of electricity generation from coal, oil and gas.
fossil_fuel_consumption	Fossil fuel consumption (TWh). This is the sum of primary energy from coal, oil and gas.
gas_consumption	Primary energy consumption from gas (TWh)
gas_electricity	Electricity generation from gas (TWh)
gas_production	Gas production (TWh)
greenhouse_gas_emissions	Greenhouse-gas emissions produced in the generation of electricity, measured in million tonnes of CO2 equivalent
hydro_consumption	Primary energy consumption from hydropower (TWh)
hydro_electricity	Electricity generation from hydropower (TWh)
low_carbon_consumption	Primary energy consumption from low-carbon sources (TWh)
low_carbon_electricity	Electricity generation from low-carbon sources (TWh). Electricity generation from renewables + nuclear power
net_elec_imports	Net electricity imports (TWh)
nuclear_consumption	Primary energy consumption from nuclear power (TWh)
nuclear_electricity	Electricity generation from nuclear power (TWh)
oil_consumption	Primary energy consumption from oil (TWh)
oil_electricity	Electricity generation from oil (TWh)
oil_production	Oil production (TWh)
other_renewable_consumptior	Primary energy consumption from other renewables (TWh)
other_renewable_electricity	Electricity generation from other renewable sources including biofuels (TWh)
other_renewable_exc_biofuel_	Electricity generation from other renewable sources excluding biofuels (TWh)
primary_energy_consumption	Primary energy consumption (TWh)
renewables_consumption	Primary energy consumption from renewables (TWh)
renewables_electricity	Electricity generation from renewables (TWh)
solar_consumption	Primary energy consumption from solar (TWh)
solar_electricity	Electricity generation from solar (TWh)
wind_consumption	Primary energy consumption from wind (TWh)
wind_electricity	Electricity generation from wind (TWh)

ii. Statistics of the data

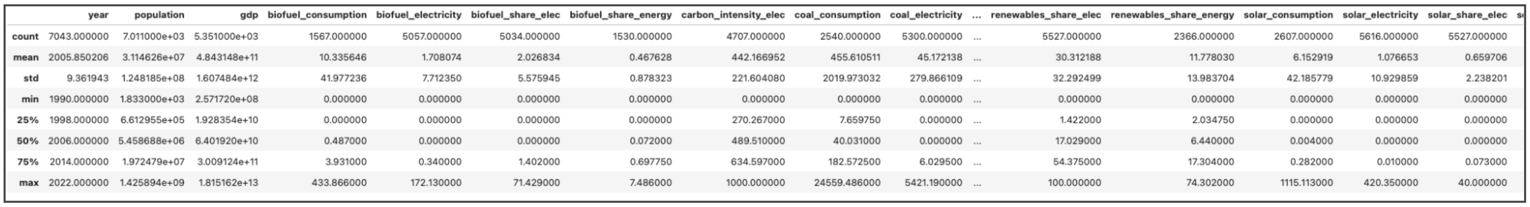
- Basic statistics of the data i.e. min, max, null count etc on original dataset (**using Hadoop Map-Reduce)**



```
iso_code        Null_Count:22013        Min:nan Max:nan Sum:0.0 Count:0
country Null_Count:1    Min:1900.0      Max:2022.0      Sum:43456382.0  Count:22012
year    Null_Count:22013        Min:nan Max:nan Sum:0.0 Count:0
coal_prod_change_pct    Null_Count:3890 Min:1833.0      Max:7975105024.0        Sum:1908526156226.0     Count:18123
coal_prod_change_twh    Null_Count:10900        Min:164206000.0 Max:113630171365376.0   Sum:3984137287763472.0  Count:11113
gas_prod_change_pct     Null_Count:20266        Min:-100.0      Max:5659.328    Sum:80522.18900000001   Count:1747
gas_prod_change_twh     Null_Count:19326        Min:-50.843     Max:141.131     Sum:7306.153    Count:2687
oil_prod_change_pct     Null_Count:19711        Min:0.0 Max:2588.512    Sum:307451.573  Count:2302
oil_prod_change_twh     Null_Count:19246        Min:0.0 Max:1199.207    Sum:105334.73499999999  Count:2767
```

Due to page limit, screenshot of output for only few features are taken for this and following screenshots

- Details data statistics **using spark RDD functions** i.e. describe, count etc on data from 1990 (after excluding nulls)



	year	population	gdp	biofuel_consumption	biofuel_electricity	biofuel_share_elec	biofuel_share_energy	carbon_intensity_elec	coal_consumption	coal_electricity	...	renewables_share_elec	renewables_share_energy	solar_consumption	solar_electricity	solar_share_elec
count	7043.000000	7.011000e+03	5.351000e+03	1567.000000	5057.000000	5034.000000	1530.000000	4707.000000	2540.000000	5300.000000	...	5527.000000	2366.000000	2607.000000	5616.000000	5527.000000
mean	2005.850206	3.114626e+07	4.843148e+11	10.335646	1.708074	2.026834	0.467628	442.166952	455.610511	45.172138	...	30.312188	11.778030	6.152919	1.076653	0.659706
std	9.361943	1.248185e+08	1.607484e+12	41.977236	7.712350	5.575945	0.878323	221.604080	2019.973032	279.866109	...	32.292499	13.983704	42.185779	10.929859	2.238201
min	1990.000000	1.833000e+03	2.571720e+08	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1998.000000	6.612955e+05	1.928354e+10	0.000000	0.000000	0.000000	0.000000	270.267000	7.659750	0.000000	...	1.422000	2.034750	0.000000	0.000000	0.000000
50%	2006.000000	5.458688e+06	6.401920e+10	0.487000	0.000000	0.000000	0.072000	489.510000	40.031000	0.000000	...	17.029000	6.440000	0.004000	0.000000	0.000000
75%	2014.000000	1.972479e+07	3.009124e+11	3.931000	0.340000	1.402000	0.697750	634.597000	182.572500	6.029500	...	54.375000	17.304000	0.282000	0.010000	0.073000
max	2022.000000	1.425894e+09	1.815162e+13	433.866000	172.130000	71.429000	7.486000	1000.000000	24559.486000	5421.190000	...	100.000000	74.302000	1115.113000	420.350000	40.000000

3 . WORK DISTRIBUTION

Project was segmented in four major tasks after proposal - Data Discovery & EDA, Analysis on data and insight generation, Visualization & Documentation. While learning concepts like mapreduce individually and later applying our learnings into the project, we divided responsibilities for other tasks among us to meet a timeline.

Amarpreet Kaur - Data Discovery, EDA, Hive SQL queries, Insight Analysis - 1, 4, Report

Ruchi Parmar - Proposal, Data Statistics & Wrangling, EDA, Insight Analysis - 2, 3, 5, Presentation

Kartikey Chauhan - Proposal, Data Transformation, Docker, Airflow, Visualization

Hamna Ashraf - Project Proposal, Data Discovery

4 . SOLUTIONS DESCRIPTION

i. Tools used

1. Hadoop Distributed FIle System (HDFS)
2. Hadoop MapReduce
3. Spark
4. Hive Data Query Layer
5. Plotly
6. Apache Airflow
7. Docker

ii & iii. How tools are used & Why they were chosen?

1. HDFS - Scalable File Storage:

The original data (csv file) is stored in Hadoop HDFS as it offers scalability, enabling parallel operations. Hadoop HDFS served as a fault-tolerant, distributed data storage layer, crucial for handling large data volumes.

2. Hadoop MapReduce - Parallel Data Discovery:

Employed for basic Exploratory Data Analysis (EDA) during the data discovery stage, including parallel execution of tasks such as null counting and finding min and max values. Although tools like spark offer this in the backend, we utilized this tool to learn & understand how map-reduce works at ground level.

3. Spark - Advanced EDA & Data Transformation:

Utilized Spark RDD transformations and various operations for data transformation according to analysis requirements. Both Spark & Spark SQL were used for batch processing and advanced analytics. Major reason for using spark was the various operations offered by it on dataframes that make transformation easier and one-liner codes - much similar to python. Also, operations like treating null were possible using its window partition functionalities.

4. Hive - Structured Data Query and Analysis:

Hive data queries were instrumental in analyzing hypotheses and gaining insights into global energy consumption. Hive served as a data query layer for structured data querying and analysis. As our data is large scale and can be partitioned by countries and years, we used Hive's partitioning functionalities while querying data. That way query processing time can be improved as only countries whose data is being analyzed will be read instead of the whole table.

5. Plotly - Insightful Visualization:

Insights were translated into Plotly visualizations to enhance user experience and understanding. Plotly is easy to understand and code along with python. Converting insights into visualization will translate data into a story and be easy to understand for stakeholders.

6. Apache Airflow - Workflow Orchestration:

Apache Airflow facilitated the creation, orchestration, and scheduling of diverse data processing workflows. Offering a flexible and extensible platform, it ensured efficient and automated data processing. We used Apache Airflow to understand how a cycle of data analysis can be automated and if it has any future applications in projects with a more dynamic approach.

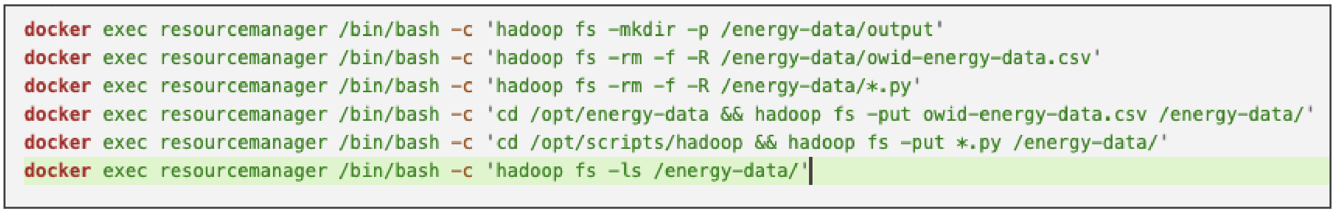
7. Docker - Streamlined Deployment:

Docker played a crucial role in achieving streamlined and consistent deployment across diverse environments. Docker was chosen for streamlined deployment, offering portability, efficiency in managing dependencies, and ensuring reproducibility across different environments. Also, it helped all of the team members to work in parallel on the same things without sharing of codes and files back and forth.

iv. Code snippets and explaining the logic behind the code snippets

1. HDFS - Transferring files to HDFS from local

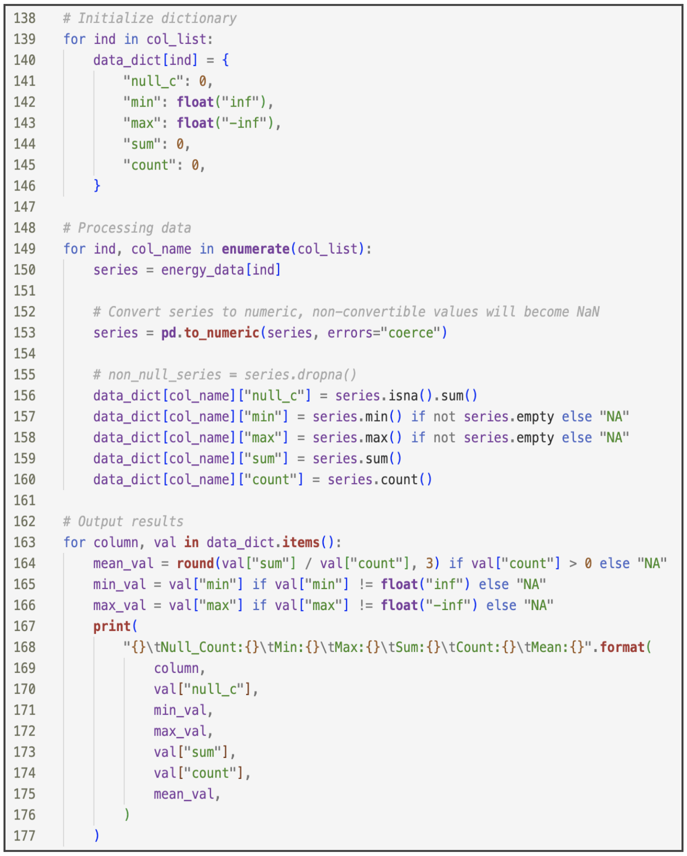
These commands are used to clear existing folders/scripts (if any) and then upload the dataset and mapreduce scripts to HDFS.



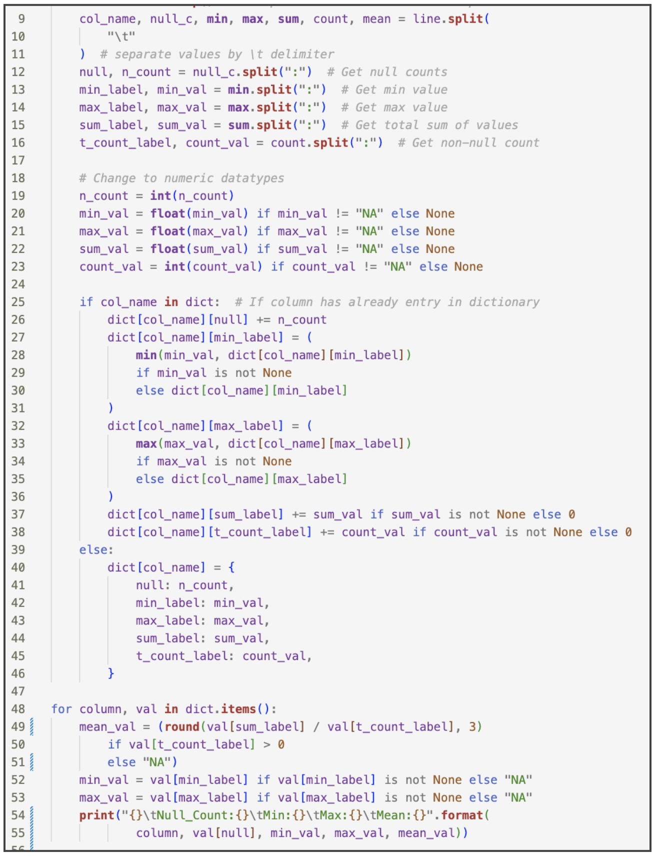
```
docker exec resourcemanager /bin/bash -c 'hadoop fs -mkdir -p /energy-data/output'
docker exec resourcemanager /bin/bash -c 'hadoop fs -rm -f -R /energy-data/owid-energy-data.csv'
docker exec resourcemanager /bin/bash -c 'hadoop fs -rm -f -R /energy-data/*.py'
docker exec resourcemanager /bin/bash -c 'cd /opt/energy-data && hadoop fs -put owid-energy-data.csv /energy-data/'
docker exec resourcemanager /bin/bash -c 'cd /opt/scripts/hadoop && hadoop fs -put *.py /energy-data/'
docker exec resourcemanager /bin/bash -c 'hadoop fs -ls /energy-data/'
```

2. Hadoop MapReduce - Basic EDA on original data

Mapper and reducer scripts read the input data, process each column to compute statistical measures (like min, max, sum, count, and mean) and then print these statistics.



```python
# Initialize dictionary
for ind in col_list:
    data_dict[ind] = {
        "null_c": 0,
        "min": float("inf"),
        "max": float("-inf"),
        "sum": 0,
        "count": 0,
    }

# Processing data
for ind, col_name in enumerate(col_list):
    series = energy_data[ind]

    # Convert series to numeric, non-convertible values will become NaN
    series = pd.to_numeric(series, errors="coerce")

    # non_null_series = series.dropna()
    data_dict[col_name]["null_c"] = series.isna().sum()
    data_dict[col_name]["min"] = series.min() if not series.empty else "NA"
    data_dict[col_name]["max"] = series.max() if not series.empty else "NA"
    data_dict[col_name]["sum"] = series.sum()
    data_dict[col_name]["count"] = series.count()

# Output results
for column, val in data_dict.items():
    mean_val = round(val["sum"] / val["count"], 3) if val["count"] > 0 else "NA"
    min_val = val["min"] if val["min"] != float("inf") else "NA"
    max_val = val["max"] if val["max"] != float("-inf") else "NA"
    print(
        "{}\tNull_Count:{}\tMin:{}\tMax:{}\tSum:{}\tCount:{}\tMean:{}".format(
            column,
            val["null_c"],
            min_val,
            max_val,
            val["sum"],
            val["count"],
            mean_val,
        )
    )
```

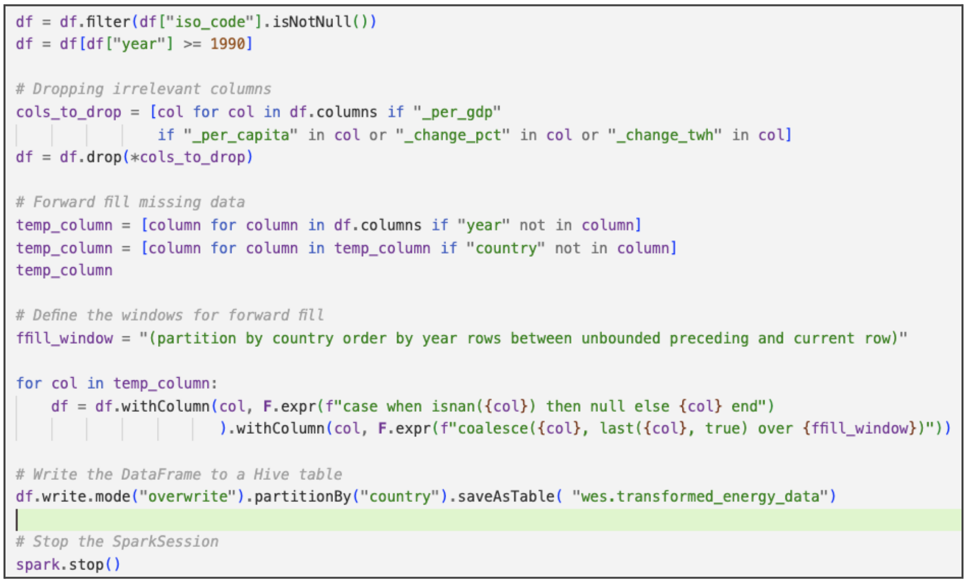


```python
    col_name, null_c, min, max, sum, count, mean = line.split(
        "\t"
    )  # separate values by \t delimiter
    null, n_count = null_c.split(":")  # Get null counts
    min_label, min_val = min.split(":")  # Get min value
    max_label, max_val = max.split(":")  # Get max value
    sum_label, sum_val = sum.split(":")  # Get total sum of values
    t_count_label, count_val = count.split(":")  # Get non-null count

    # Change to numeric datatypes
    n_count = int(n_count)
    min_val = float(min_val) if min_val != "NA" else None
    max_val = float(max_val) if max_val != "NA" else None
    sum_val = float(sum_val) if sum_val != "NA" else None
    count_val = int(count_val) if count_val != "NA" else None

    if col_name in dict:  # If column has already entry in dictionary
        dict[col_name][null] += n_count
        dict[col_name][min_label] = (
            min(min_val, dict[col_name][min_label])
            if min_val is not None
            else dict[col_name][min_label]
        )
        dict[col_name][max_label] = (
            max(max_val, dict[col_name][max_label])
            if max_val is not None
            else dict[col_name][max_label]
        )
        dict[col_name][sum_label] += sum_val if sum_val is not None else 0
        dict[col_name][t_count_label] += count_val if count_val is not None else 0
    else:
        dict[col_name] = {
            null: n_count,
            min_label: min_val,
            max_label: max_val,
            sum_label: sum_val,
            t_count_label: count_val,
        }

for column, val in dict.items():
    mean_val = (round(val[sum_label] / val[t_count_label], 3)
        if val[t_count_label] > 0
        else "NA")
    min_val = val[min_label] if val[min_label] is not None else "NA"
    max_val = val[max_label] if val[max_label] is not None else "NA"
    print("{}\tNull_Count:{}\tMin:{}\tMax:{}\tMean:{}".format(
            column, val[null], min_val, max_val, mean_val))
```

3. Spark - Advanced EDA & Data transformation

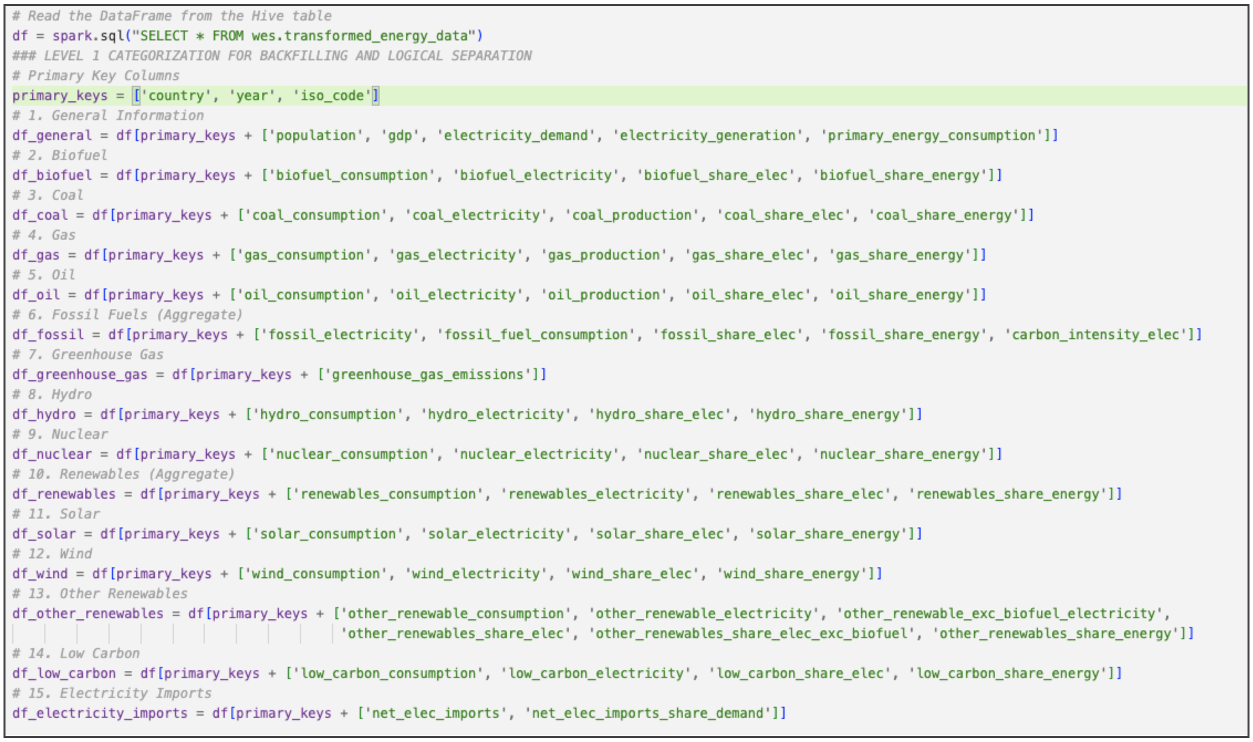
The Data transformation PySpark script does the following EDA and transformation operations:

- Separating regions and countries (filtering by isoCode != null as our scope of analysis)
- Null treatment (dropping data before 1990 as the null percentage is high before).
- Dropping redundant columns (change_pct, per_capita, change_twh, per_gdp).
- Front-filling using spark window function (backfilling was not done as we can not accurately represent past energy usage) and save this into a temporary hive table for categorization.



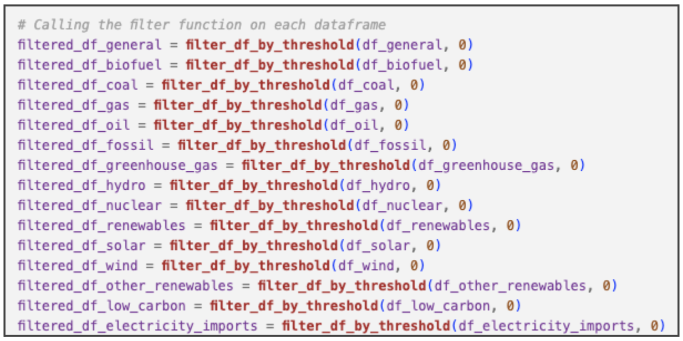
```
df = df.filter(df["iso_code"].isNotNull())
df = df[df["year"] >= 1990]

# Dropping irrelevant columns
cols_to_drop = [col for col in df.columns if "_per_gdp"
                if "_per_capita" in col or "_change_pct" in col or "_change_twh" in col]
df = df.drop(*cols_to_drop)

# Forward fill missing data
temp_column = [column for column in df.columns if "year" not in column]
temp_column = [column for column in temp_column if "country" not in column]
temp_column

# Define the windows for forward fill
ffill_window = "(partition by country order by year rows between unbounded preceding and current row)"

for col in temp_column:
    df = df.withColumn(col, F.expr(f"case when isnan({col}) then null else {col} end")
                      ).withColumn(col, F.expr(f"coalesce({col}, last({col}, true) over {ffill_window})"))

# Write the DataFrame to a Hive table
df.write.mode("overwrite").partitionBy("country").saveAsTable( "wes.transformed_energy_data")

# Stop the SparkSession
spark.stop()
```

The Data Categorization script categorized the >100 columns into 15 tables based on energy sources and partitioned by year/country.

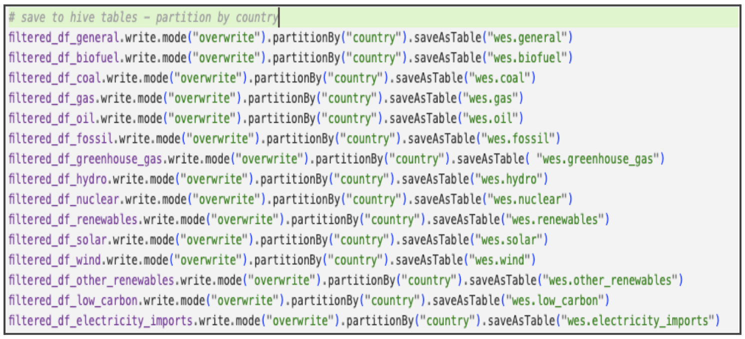


```
# Read the DataFrame from the Hive table
df = spark.sql("SELECT * FROM wes.transformed_energy_data")
### LEVEL 1 CATEGORIZATION FOR BACKFILLING AND LOGICAL SEPARATION
# Primary Key Columns
primary_keys = ['country', 'year', 'iso_code']
# 1. General Information
df_general = df[primary_keys + ['population', 'gdp', 'electricity_demand', 'electricity_generation', 'primary_energy_consumption']]
# 2. Biofuel
df_biofuel = df[primary_keys + ['biofuel_consumption', 'biofuel_electricity', 'biofuel_share_elec', 'biofuel_share_energy']]
# 3. Coal
df_coal = df[primary_keys + ['coal_consumption', 'coal_electricity', 'coal_production', 'coal_share_elec', 'coal_share_energy']]
# 4. Gas
df_gas = df[primary_keys + ['gas_consumption', 'gas_electricity', 'gas_production', 'gas_share_elec', 'gas_share_energy']]
# 5. Oil
df_oil = df[primary_keys + ['oil_consumption', 'oil_electricity', 'oil_production', 'oil_share_elec', 'oil_share_energy']]
# 6. Fossil Fuels (Aggregate)
df_fossil = df[primary_keys + ['fossil_electricity', 'fossil_fuel_consumption', 'fossil_share_elec', 'fossil_share_energy', 'carbon_intensity_elec']]
# 7. Greenhouse Gas
df_greenhouse_gas = df[primary_keys + ['greenhouse_gas_emissions']]
# 8. Hydro
df_hydro = df[primary_keys + ['hydro_consumption', 'hydro_electricity', 'hydro_share_elec', 'hydro_share_energy']]
# 9. Nuclear
df_nuclear = df[primary_keys + ['nuclear_consumption', 'nuclear_electricity', 'nuclear_share_elec', 'nuclear_share_energy']]
# 10. Renewables (Aggregate)
df_renewables = df[primary_keys + ['renewables_consumption', 'renewables_electricity', 'renewables_share_elec', 'renewables_share_energy']]
# 11. Solar
df_solar = df[primary_keys + ['solar_consumption', 'solar_electricity', 'solar_share_elec', 'solar_share_energy']]
# 12. Wind
df_wind = df[primary_keys + ['wind_consumption', 'wind_electricity', 'wind_share_elec', 'wind_share_energy']]
# 13. Other Renewables
df_other_renewables = df[primary_keys + ['other_renewable_consumption', 'other_renewable_electricity', 'other_renewable_exc_biofuel_electricity',
                                          'other_renewables_share_elec', 'other_renewables_share_elec_exc_biofuel', 'other_renewables_share_energy']]
# 14. Low Carbon
df_low_carbon = df[primary_keys + ['low_carbon_consumption', 'low_carbon_electricity', 'low_carbon_share_elec', 'low_carbon_share_energy']]
# 15. Electricity Imports
df_electricity_imports = df[primary_keys + ['net_elec_imports', 'net_elec_imports_share_demand']]
```

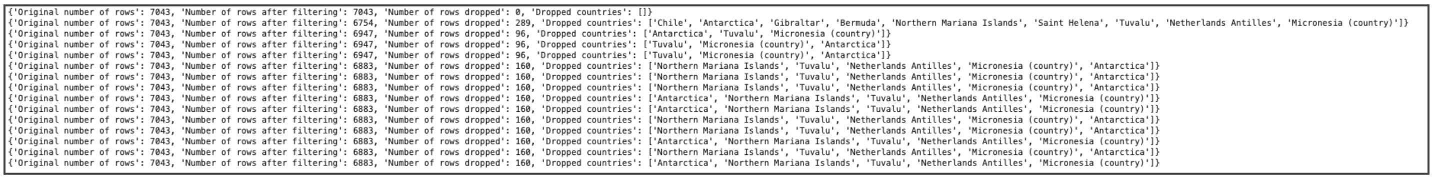
We're using a filter_df_by_threshold function to drop countries that do not have even 1 column of data. Final dataframe RDD are saved as individual tables in hive and partitioned by country/year to run our hive insight analysis efficiently.



```
# Calling the filter function on each dataframe
filtered_df_general = filter_df_by_threshold(df_general, 0)
filtered_df_biofuel = filter_df_by_threshold(df_biofuel, 0)
filtered_df_coal = filter_df_by_threshold(df_coal, 0)
filtered_df_gas = filter_df_by_threshold(df_gas, 0)
filtered_df_oil = filter_df_by_threshold(df_oil, 0)
filtered_df_fossil = filter_df_by_threshold(df_fossil, 0)
filtered_df_greenhouse_gas = filter_df_by_threshold(df_greenhouse_gas, 0)
filtered_df_hydro = filter_df_by_threshold(df_hydro, 0)
filtered_df_nuclear = filter_df_by_threshold(df_nuclear, 0)
filtered_df_renewables = filter_df_by_threshold(df_renewables, 0)
filtered_df_solar = filter_df_by_threshold(df_solar, 0)
filtered_df_wind = filter_df_by_threshold(df_wind, 0)
filtered_df_other_renewables = filter_df_by_threshold(df_other_renewables, 0)
filtered_df_low_carbon = filter_df_by_threshold(df_low_carbon, 0)
filtered_df_electricity_imports = filter_df_by_threshold(df_electricity_imports, 0)
```



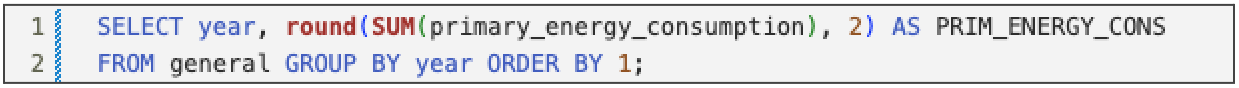
```
# save to hive tables - partition by country
filtered_df_general.write.mode("overwrite").partitionBy("country").saveAsTable("wes.general")
filtered_df_biofuel.write.mode("overwrite").partitionBy("country").saveAsTable("wes.biofuel")
filtered_df_coal.write.mode("overwrite").partitionBy("country").saveAsTable("wes.coal")
filtered_df_gas.write.mode("overwrite").partitionBy("country").saveAsTable("wes.gas")
filtered_df_oil.write.mode("overwrite").partitionBy("country").saveAsTable("wes.oil")
filtered_df_fossil.write.mode("overwrite").partitionBy("country").saveAsTable("wes.fossil")
filtered_df_greenhouse_gas.write.mode("overwrite").partitionBy("country").saveAsTable( "wes.greenhouse_gas")
filtered_df_hydro.write.mode("overwrite").partitionBy("country").saveAsTable("wes.hydro")
filtered_df_nuclear.write.mode("overwrite").partitionBy("country").saveAsTable("wes.nuclear")
filtered_df_renewables.write.mode("overwrite").partitionBy("country").saveAsTable("wes.renewables")
filtered_df_solar.write.mode("overwrite").partitionBy("country").saveAsTable("wes.solar")
filtered_df_wind.write.mode("overwrite").partitionBy("country").saveAsTable("wes.wind")
filtered_df_other_renewables.write.mode("overwrite").partitionBy("country").saveAsTable("wes.other_renewables")
filtered_df_low_carbon.write.mode("overwrite").partitionBy("country").saveAsTable("wes.low_carbon")
filtered_df_electricity_imports.write.mode("overwrite").partitionBy("country").saveAsTable("wes.electricity_imports")
```



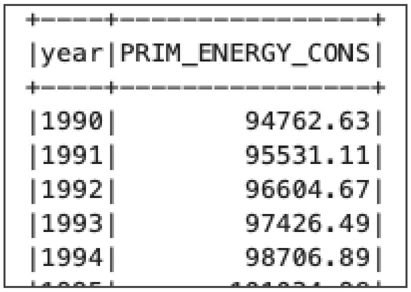
```
{'Original number of rows': 7043, 'Number of rows after filtering': 7043, 'Number of rows dropped': 0, 'Dropped countries': []}
{'Original number of rows': 7043, 'Number of rows after filtering': 6754, 'Number of rows dropped': 289, 'Dropped countries': ['Chile', 'Antarctica', 'Gibraltar', 'Bermuda', 'Northern Mariana Islands', 'Saint Helena', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6947, 'Number of rows dropped': 96, 'Dropped countries': ['Antarctica', 'Tuvalu', 'Micronesia (country)']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6947, 'Number of rows dropped': 96, 'Dropped countries': ['Tuvalu', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6947, 'Number of rows dropped': 96, 'Dropped countries': ['Tuvalu', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Antarctica', 'Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Antarctica', 'Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Antarctica', 'Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)', 'Antarctica']}
{'Original number of rows': 7043, 'Number of rows after filtering': 6883, 'Number of rows dropped': 160, 'Dropped countries': ['Antarctica', 'Northern Mariana Islands', 'Tuvalu', 'Netherlands Antilles', 'Micronesia (country)']}
```

4. Hive - Insight Analysis

Insight 1- This hive-SQL logic retrieves the sum of primary energy consumption for each year from the "general" table, rounding the result to two decimal places, and organizes the output in ascending order based on the year.

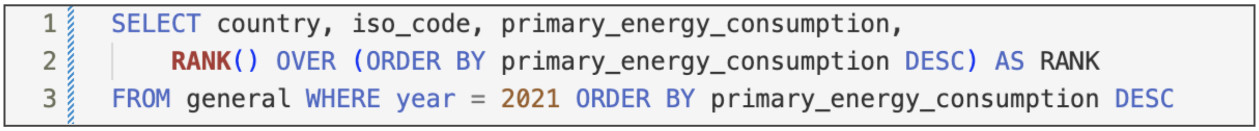


```
SELECT year, round(SUM(primary_energy_consumption), 2) AS PRIM_ENERGY_CONS
FROM general GROUP BY year ORDER BY 1;
```

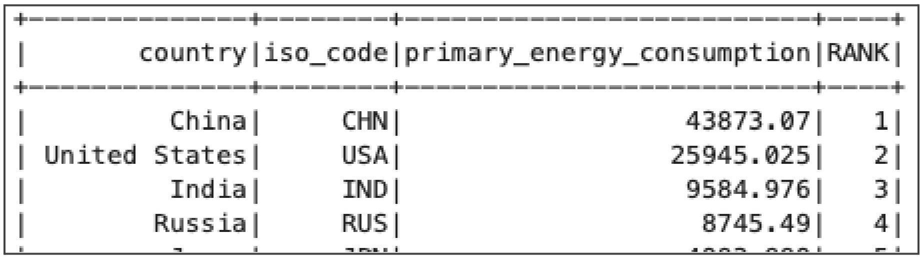


```
+----+----------------+
|year|PRIM_ENERGY_CONS|
+----+----------------+
|1990|        94762.63|
|1991|        95531.11|
|1992|        96604.67|
|1993|        97426.49|
|1994|        98706.89|
```

Insight 2- This hive-SQL query retrieves the country, ISO code, and primary energy consumption for the year 2021, assigning a rank to each country based on their descending primary energy consumption. The results are ordered by primary energy consumption in descending order.

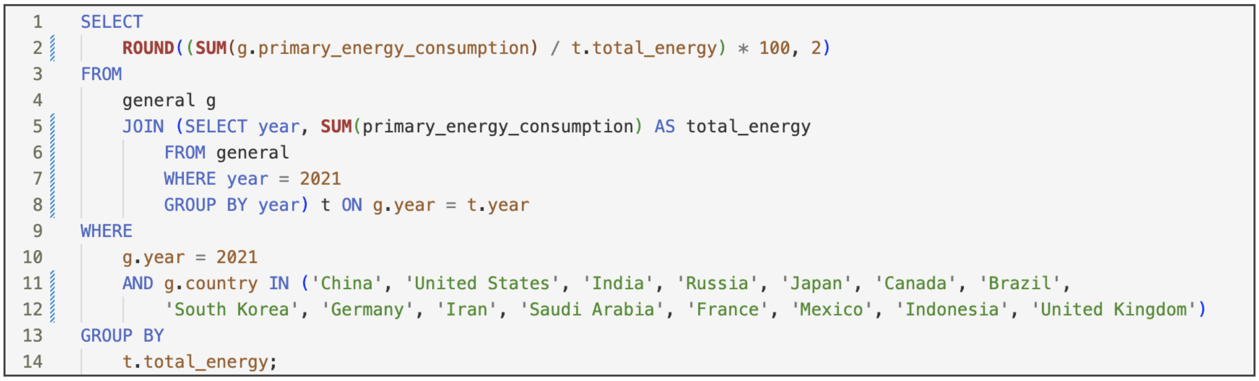


```
SELECT country, iso_code, primary_energy_consumption,
    RANK() OVER (ORDER BY primary_energy_consumption DESC) AS RANK
FROM general WHERE year = 2021 ORDER BY primary_energy_consumption DESC
```

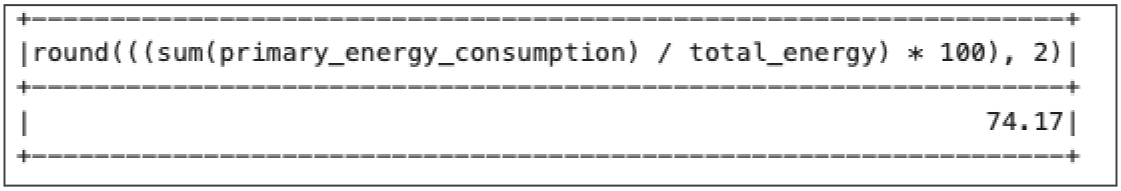


```
+--------------+--------+--------------------------+----+
|       country|iso_code|primary_energy_consumption|RANK|
+--------------+--------+--------------------------+----+
|         China|     CHN|                  43873.07|   1|
| United States|     USA|                 25945.025|   2|
|         India|     IND|                  9584.976|   3|
|        Russia|     RUS|                   8745.49|   4|
```

2b. This hive-SQL logic calculates the percentage contribution of primary energy consumption for selected countries in the year 2021 to the total global energy consumption for that year, rounded to two decimal places.

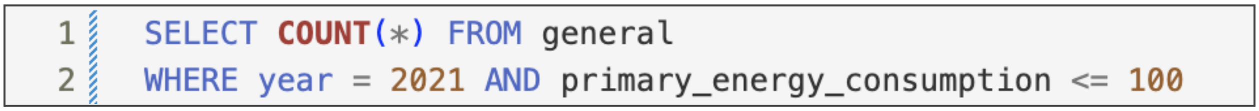


```sql
SELECT
    ROUND((SUM(g.primary_energy_consumption) / t.total_energy) * 100, 2)
FROM
    general g
    JOIN (SELECT year, SUM(primary_energy_consumption) AS total_energy
        FROM general
        WHERE year = 2021
        GROUP BY year) t ON g.year = t.year
WHERE
    g.year = 2021
    AND g.country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil',
        'South Korea', 'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia', 'United Kingdom')
GROUP BY
    t.total_energy;
```

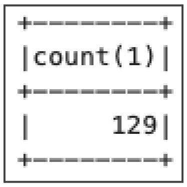


```
+------------------------------------------------------------------+
|round(((sum(primary_energy_consumption) / total_energy) * 100), 2)|
+------------------------------------------------------------------+
|                                                             74.17|
+------------------------------------------------------------------+
```

2c. This hive-SQL query retrieves the number of countries with less than 100 twH energy consumption in the year 2021.

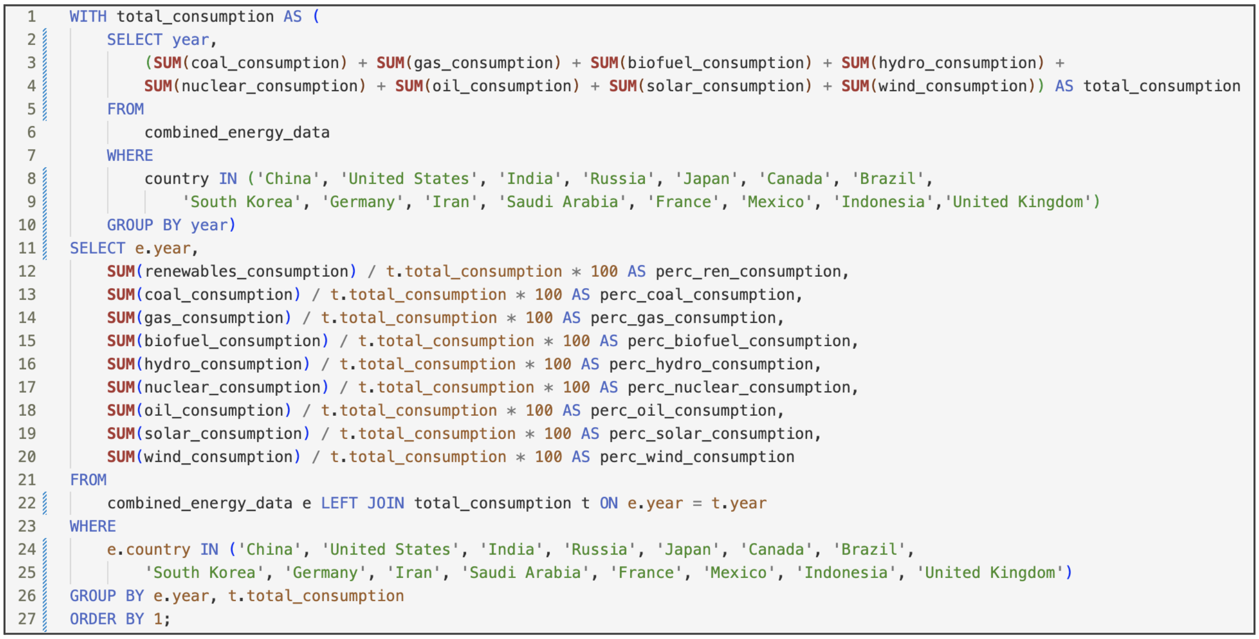


```sql
SELECT COUNT(*) FROM general
WHERE year = 2021 AND primary_energy_consumption <= 100
```

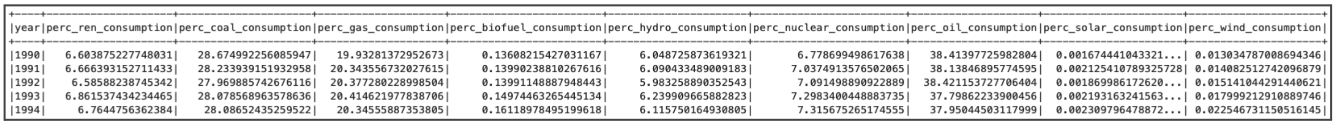


```
+--------+
|count(1)|
+--------+
|     129|
+--------+
```

Insight 3- This hive-SQL query computes the percentage contribution of various energy sources (coal, gas, biofuel, hydro, nuclear, oil, solar, wind) to the total energy consumption for specific countries in each year. It utilizes a Common Table Expression (CTE) for total consumption and calculates the percentages based on the total energy consumed, grouping the results by year.

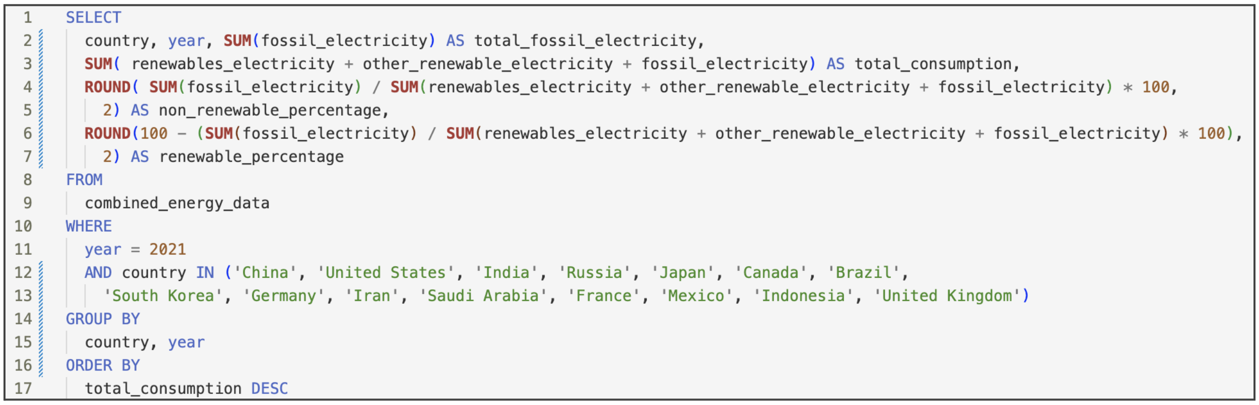


```sql
WITH total_consumption AS (
    SELECT year,
        (SUM(coal_consumption) + SUM(gas_consumption) + SUM(biofuel_consumption) + SUM(hydro_consumption) +
        SUM(nuclear_consumption) + SUM(oil_consumption) + SUM(solar_consumption) + SUM(wind_consumption)) AS total_consumption
    FROM
        combined_energy_data
    WHERE
        country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil',
            'South Korea', 'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia','United Kingdom')
    GROUP BY year)
SELECT e.year,
    SUM(renewables_consumption) / t.total_consumption * 100 AS perc_ren_consumption,
    SUM(coal_consumption) / t.total_consumption * 100 AS perc_coal_consumption,
    SUM(gas_consumption) / t.total_consumption * 100 AS perc_gas_consumption,
    SUM(biofuel_consumption) / t.total_consumption * 100 AS perc_biofuel_consumption,
    SUM(hydro_consumption) / t.total_consumption * 100 AS perc_hydro_consumption,
    SUM(nuclear_consumption) / t.total_consumption * 100 AS perc_nuclear_consumption,
    SUM(oil_consumption) / t.total_consumption * 100 AS perc_oil_consumption,
    SUM(solar_consumption) / t.total_consumption * 100 AS perc_solar_consumption,
    SUM(wind_consumption) / t.total_consumption * 100 AS perc_wind_consumption
FROM
    combined_energy_data e LEFT JOIN total_consumption t ON e.year = t.year
WHERE
    e.country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil',
        'South Korea', 'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia', 'United Kingdom')
GROUP BY e.year, t.total_consumption
ORDER BY 1;
```

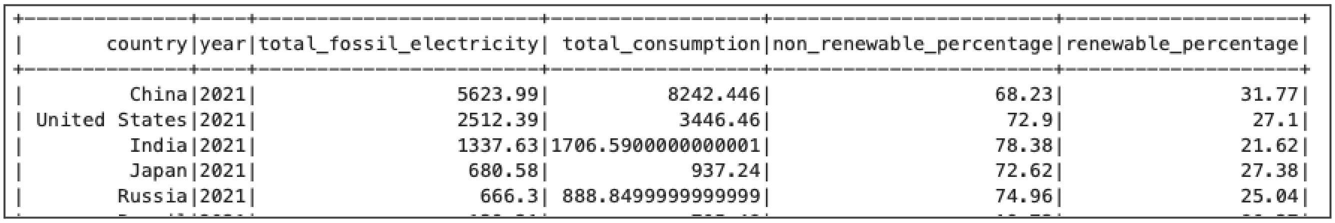


```
+----+--------------------+---------------------+--------------------+------------------------+----------------------+------------------------+--------------------+----------------------+---------------------+
|year|perc_ren_consumption|perc_coal_consumption|perc_gas_consumption|perc_biofuel_consumption|perc_hydro_consumption|perc_nuclear_consumption|perc_oil_consumption|perc_solar_consumption|perc_wind_consumption|
+----+--------------------+---------------------+--------------------+------------------------+----------------------+------------------------+--------------------+----------------------+---------------------+
|1990|   6.603875227748031|   28.674992256085947|   19.93281372952673|     0.13608215427031167|     6.048725873619321|       6.778699498617638|   38.41397725982804|  0.001674441043321...| 0.013034787008694346|
|1991|   6.666393152711433|   28.233939151932958|  20.343556732027615|     0.13990238810267616|     6.090433489009183|      7.0374913576502065|   38.13846895774595|  0.002125410789325728| 0.014082512742096879|
|1992|    6.58588238745342|   27.969885742676116|  20.377280228998504|     0.13991148887948443|     5.983258890352543|       7.091498890922889|  38.421153727706404|  0.001869986172620...| 0.015141044291440621|
|1993|   6.861537434234465|   28.078568963578636|  20.414621977838706|     0.14974463265445134|     6.239909665882823|      7.2983400448883735|   37.79862233900456|  0.002193163241563...| 0.017999212910889746|
|1994|     6.7644756362384|    28.08652435259522|   20.34555887353805|     0.16118978495199618|     6.115750164930805|       7.315675265174555|   37.95044503117999|  0.002309796478872...| 0.022546731150516145|
```

Insight 4a- This hive-SQL query calculates the percentage of non-renewable and renewable energy source consumption for electricity generation for specific countries in 2021. It aggregates data by country and year, presenting the results ordered by total electricity consumption in descending order.

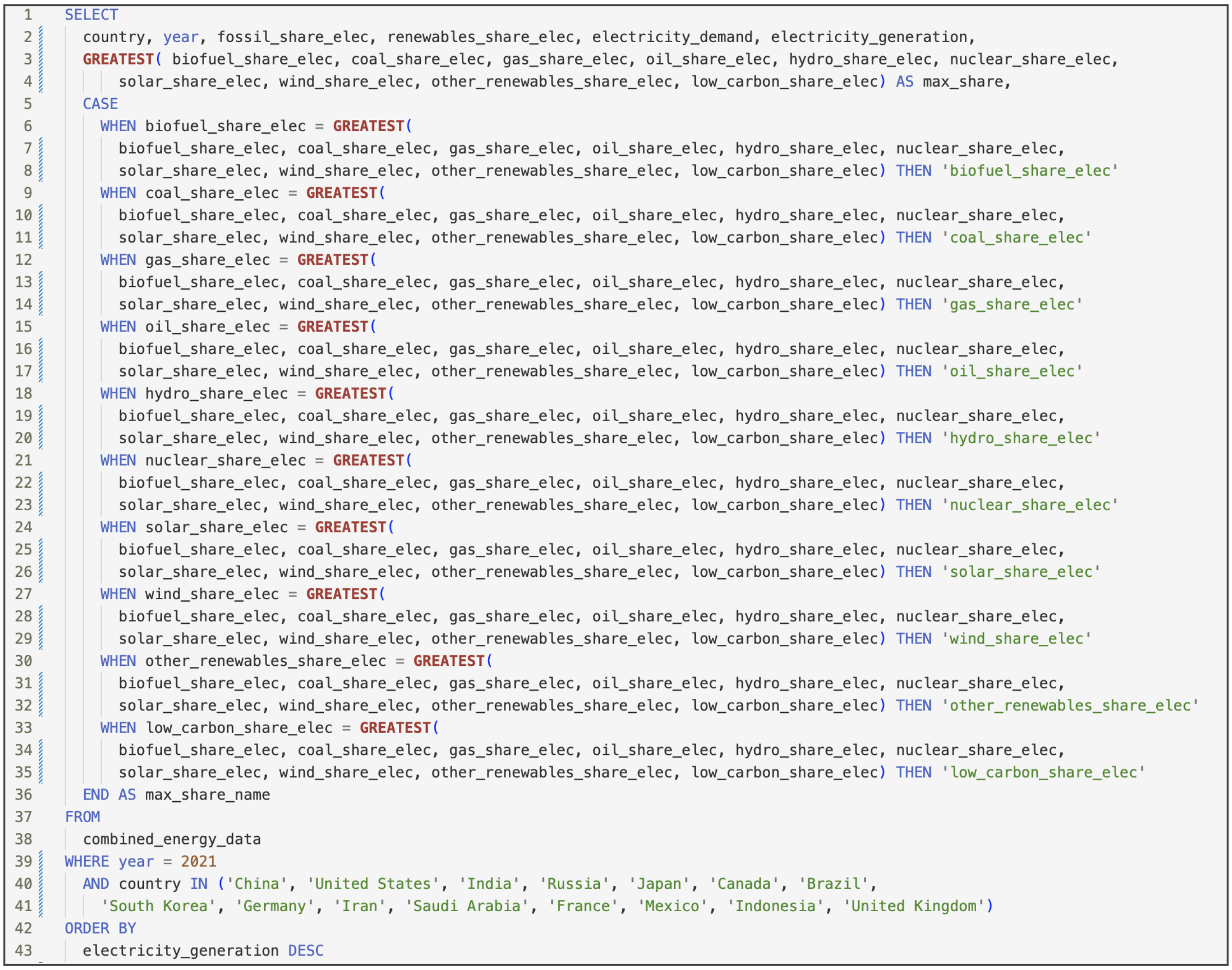


```sql
SELECT
  country, year, SUM(fossil_electricity) AS total_fossil_electricity,
  SUM( renewables_electricity + other_renewable_electricity + fossil_electricity) AS total_consumption,
  ROUND( SUM(fossil_electricity) / SUM(renewables_electricity + other_renewable_electricity + fossil_electricity) * 100,
    2) AS non_renewable_percentage,
  ROUND(100 - (SUM(fossil_electricity) / SUM(renewables_electricity + other_renewable_electricity + fossil_electricity) * 100),
    2) AS renewable_percentage
FROM
  combined_energy_data
WHERE
  year = 2021
  AND country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil',
    'South Korea', 'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia', 'United Kingdom')
GROUP BY
  country, year
ORDER BY
  total_consumption DESC
```

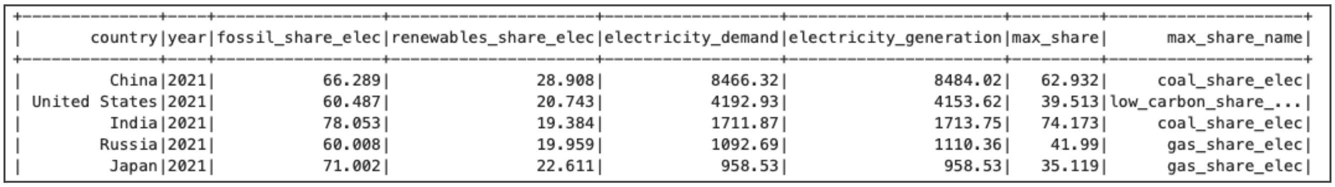


```
+--------------+----+------------------------+------------------+------------------------+--------------------+
|       country|year|total_fossil_electricity| total_consumption|non_renewable_percentage|renewable_percentage|
+--------------+----+------------------------+------------------+------------------------+--------------------+
|         China|2021|                 5623.99|          8242.446|                   68.23|               31.77|
| United States|2021|                 2512.39|           3446.46|                    72.9|                27.1|
|         India|2021|                 1337.63|1706.5900000000001|                   78.38|               21.62|
|         Japan|2021|                  680.58|            937.24|                   72.62|               27.38|
|        Russia|2021|                   666.3| 888.8499999999999|                   74.96|               25.04|
```

4b. This hive-SQL query retrieves key metrics related to electricity generation for specific countries in 2021, including fossil and renewables shares. It identifies the maximum share of electricity generation and specifies the corresponding type, presenting results ordered by electricity generation in descending order.

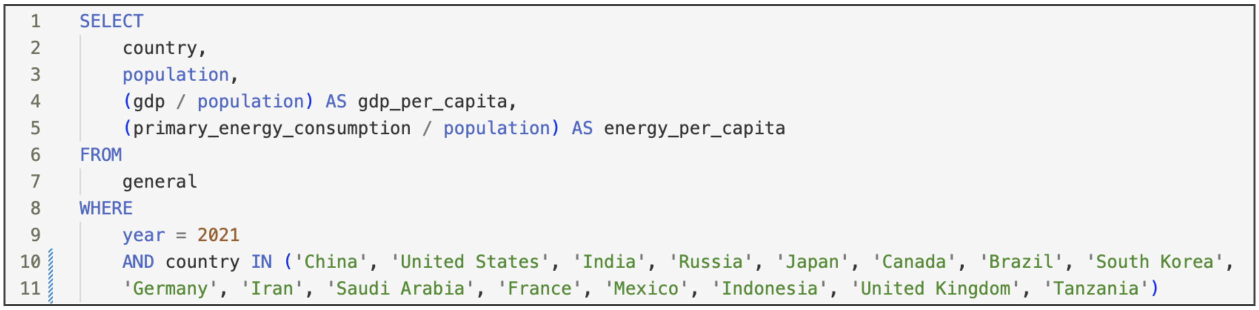


```sql
SELECT
  country, year, fossil_share_elec, renewables_share_elec, electricity_demand, electricity_generation,
  GREATEST( biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
    solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) AS max_share,
  CASE
    WHEN biofuel_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'biofuel_share_elec'
    WHEN coal_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'coal_share_elec'
    WHEN gas_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'gas_share_elec'
    WHEN oil_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'oil_share_elec'
    WHEN hydro_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'hydro_share_elec'
    WHEN nuclear_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'nuclear_share_elec'
    WHEN solar_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'solar_share_elec'
    WHEN wind_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'wind_share_elec'
    WHEN other_renewables_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'other_renewables_share_elec'
    WHEN low_carbon_share_elec = GREATEST(
      biofuel_share_elec, coal_share_elec, gas_share_elec, oil_share_elec, hydro_share_elec, nuclear_share_elec,
      solar_share_elec, wind_share_elec, other_renewables_share_elec, low_carbon_share_elec) THEN 'low_carbon_share_elec'
  END AS max_share_name
FROM
  combined_energy_data
WHERE year = 2021
  AND country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil',
    'South Korea', 'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia', 'United Kingdom')
ORDER BY
  electricity_generation DESC
```

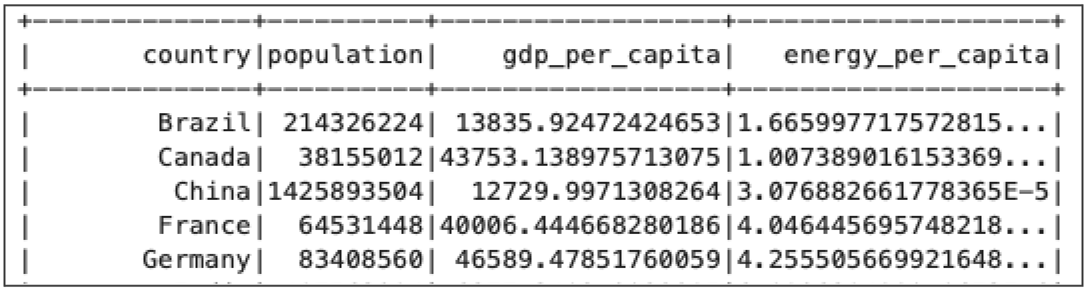


```
+-------------+----+-----------------+---------------------+------------------+----------------------+---------+--------------------+
|      country|year|fossil_share_elec|renewables_share_elec|electricity_demand|electricity_generation|max_share|       max_share_name|
+-------------+----+-----------------+---------------------+------------------+----------------------+---------+--------------------+
|        China|2021|           66.289|               28.908|           8466.32|               8484.02|   62.932|     coal_share_elec|
|United States|2021|           60.487|               20.743|           4192.93|               4153.62|   39.513|low_carbon_share_...|
|        India|2021|           78.053|               19.384|           1711.87|               1713.75|   74.173|     coal_share_elec|
|       Russia|2021|           60.008|               19.959|           1092.69|               1110.36|    41.99|      gas_share_elec|
|        Japan|2021|           71.002|               22.611|            958.53|                958.53|   35.119|      gas_share_elec|
```

Insight 5- This hive-SQL query gets population, GDP per capita and energy consumption per capita for top 15 countries to compare and find any correlation between these three if exists.



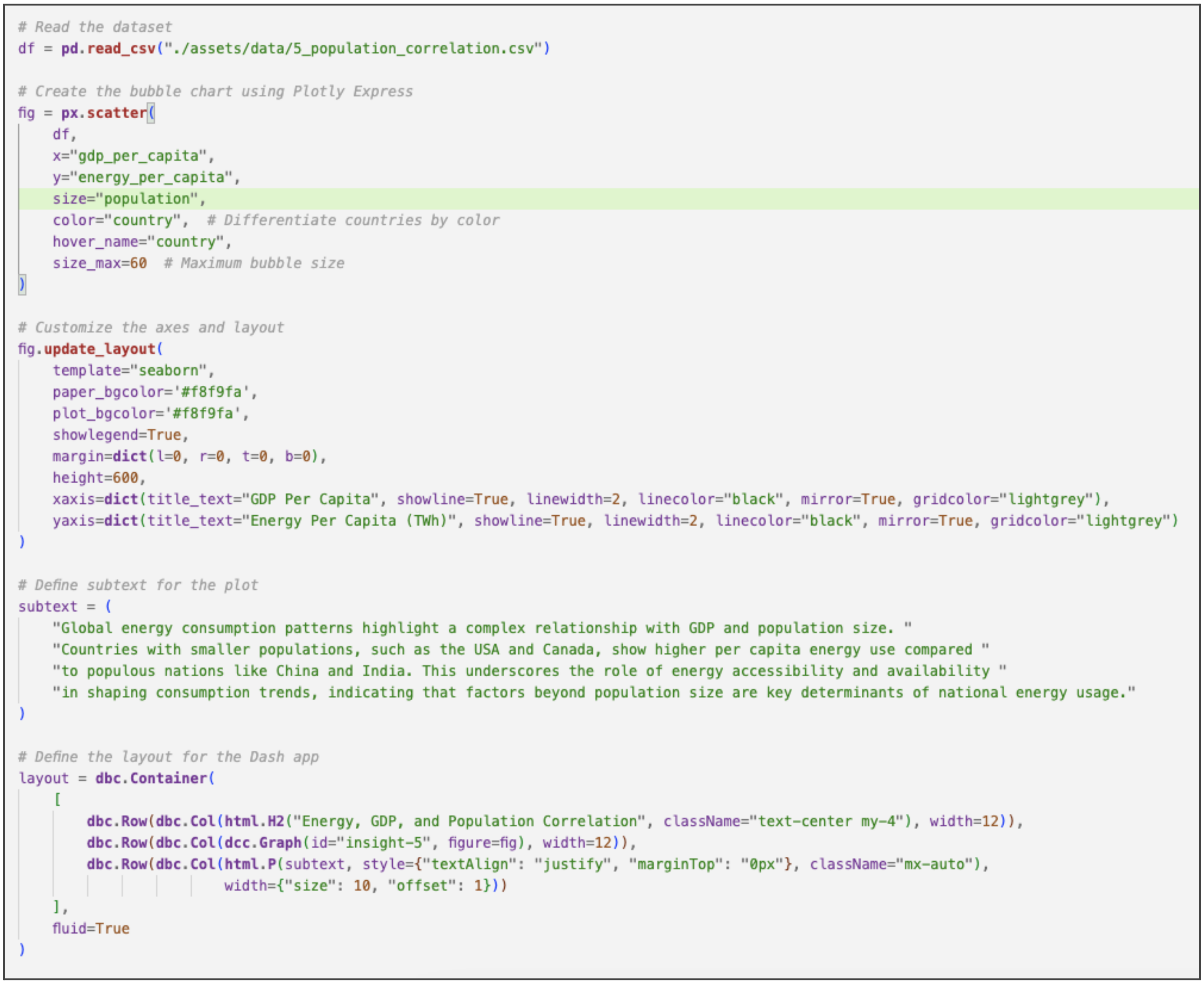
```sql
SELECT
    country,
    population,
    (gdp / population) AS gdp_per_capita,
    (primary_energy_consumption / population) AS energy_per_capita
FROM
    general
WHERE
    year = 2021
    AND country IN ('China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil', 'South Korea',
    'Germany', 'Iran', 'Saudi Arabia', 'France', 'Mexico', 'Indonesia', 'United Kingdom', 'Tanzania')
```



```
+--------------+----------+------------------+--------------------+
|       country|population|    gdp_per_capita|   energy_per_capita|
+--------------+----------+------------------+--------------------+
|        Brazil| 214326224| 13835.92472424653|1.665997717572815...|
|        Canada|  38155012|43753.138975713075|1.007389016153369...|
|         China|1425893504|  12729.9971308264|3.076882661778365E-5|
|        France|  64531448|40006.444668280186|4.046445695748218...|
|       Germany|  83408560| 46589.47851760059|4.255505669921648...|
```

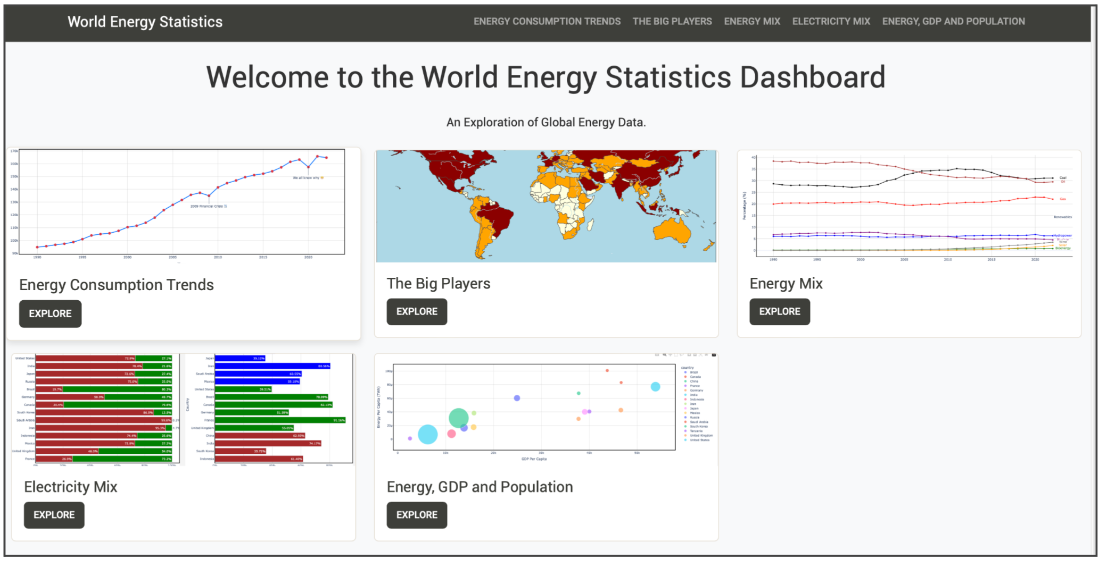
5. Plotly - Visualization of insights

We used plotly for visualization of all our insights. The below code for one of the insights (#5), is used for creating a bubble plot, adding all required annotations and labels and giving a short description of the insight. These components are then rendered into the main Plotly Dash application.



```python
# Read the dataset
df = pd.read_csv("./assets/data/5_population_correlation.csv")

# Create the bubble chart using Plotly Express
fig = px.scatter(
    df,
    x="gdp_per_capita",
    y="energy_per_capita",
    size="population",
    color="country",  # Differentiate countries by color
    hover_name="country",
    size_max=60  # Maximum bubble size
)

# Customize the axes and layout
fig.update_layout(
    template="seaborn",
    paper_bgcolor='#f8f9fa',
    plot_bgcolor='#f8f9fa',
    showlegend=True,
    margin=dict(l=0, r=0, t=0, b=0),
    height=600,
    xaxis=dict(title_text="GDP Per Capita", showline=True, linewidth=2, linecolor="black", mirror=True, gridcolor="lightgrey"),
    yaxis=dict(title_text="Energy Per Capita (TWh)", showline=True, linewidth=2, linecolor="black", mirror=True, gridcolor="lightgrey")
)

# Define subtext for the plot
subtext = (
    "Global energy consumption patterns highlight a complex relationship with GDP and population size. "
    "Countries with smaller populations, such as the USA and Canada, show higher per capita energy use compared "
    "to populous nations like China and India. This underscores the role of energy accessibility and availability "
    "in shaping consumption trends, indicating that factors beyond population size are key determinants of national energy usage."
)

# Define the layout for the Dash app
layout = dbc.Container(
    [
        dbc.Row(dbc.Col(html.H2("Energy, GDP, and Population Correlation", className="text-center my-4"), width=12)),
        dbc.Row(dbc.Col(dcc.Graph(id="insight-5", figure=fig), width=12)),
        dbc.Row(dbc.Col(html.P(subtext, style={"textAlign": "justify", "marginTop": "0px"}, className="mx-auto"),
                        width={"size": 10, "offset": 1}))
    ],
    fluid=True
)
```

This is how the main dashboard looks like (Plotly dashboard link).

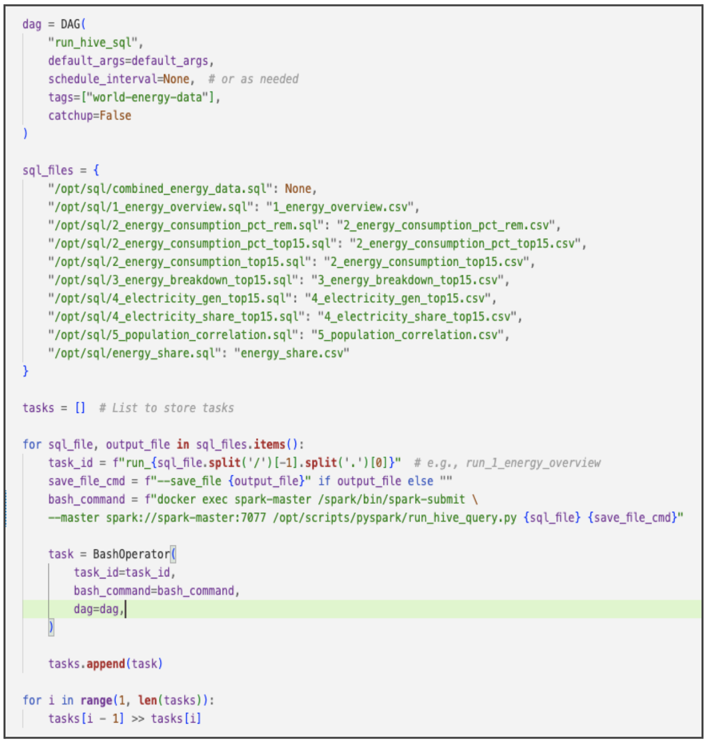


Welcome to the World Energy Statistics Dashboard

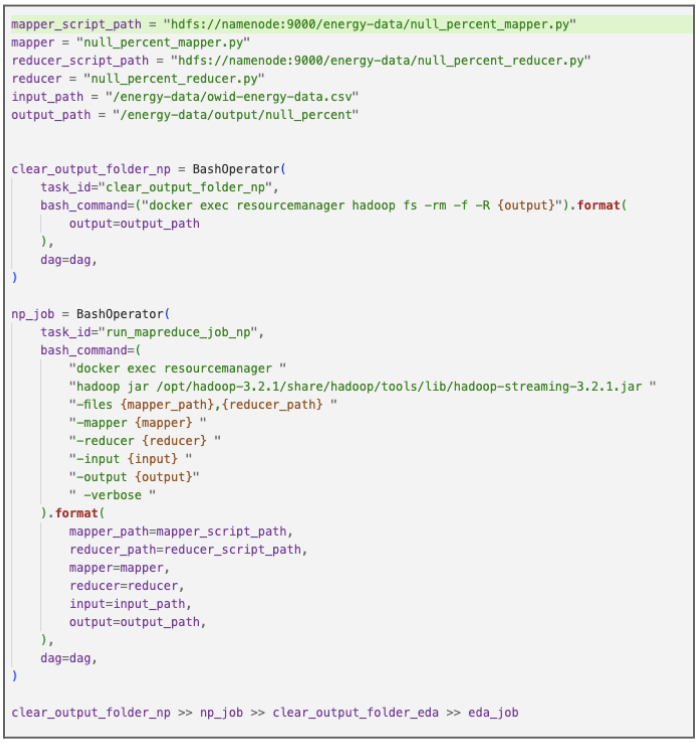
An Exploration of Global Energy Data.

6. Apache Airflow - Binding everything in one place.

   Dag scripts are created to run individual data processes in Airflow. For example, the first dag below runs all the insight queries and exports them to csv files. The second dag runs the mapreduce jobs required for EDA and data statistics using BashOperator.

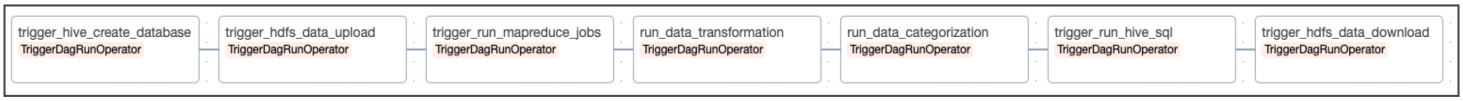


```python
dag = DAG(
    "run_hive_sql",
    default_args=default_args,
    schedule_interval=None,  # or as needed
    tags=["world-energy-data"],
    catchup=False
)

sql_files = {
    "/opt/sql/combined_energy_data.sql": None,
    "/opt/sql/1_energy_overview.sql": "1_energy_overview.csv",
    "/opt/sql/2_energy_consumption_pct_rem.sql": "2_energy_consumption_pct_rem.csv",
    "/opt/sql/2_energy_consumption_pct_top15.sql": "2_energy_consumption_pct_top15.csv",
    "/opt/sql/2_energy_consumption_top15.sql": "2_energy_consumption_top15.csv",
    "/opt/sql/3_energy_breakdown_top15.sql": "3_energy_breakdown_top15.csv",
    "/opt/sql/4_electricity_gen_top15.sql": "4_electricity_gen_top15.csv",
    "/opt/sql/4_electricity_share_top15.sql": "4_electricity_share_top15.csv",
    "/opt/sql/5_population_correlation.sql": "5_population_correlation.csv",
    "/opt/sql/energy_share.sql": "energy_share.csv"
}

tasks = []  # List to store tasks

for sql_file, output_file in sql_files.items():
    task_id = f"run_{sql_file.split('/')[-1].split('.')[0]}"  # e.g., run_1_energy_overview
    save_file_cmd = f"--save_file {output_file}" if output_file else ""
    bash_command = f"docker exec spark-master /spark/bin/spark-submit \
    --master spark://spark-master:7077 /opt/scripts/pyspark/run_hive_query.py {sql_file} {save_file_cmd}"

    task = BashOperator(
        task_id=task_id,
        bash_command=bash_command,
        dag=dag,
    )

    tasks.append(task)

for i in range(1, len(tasks)):
    tasks[i - 1] >> tasks[i]
```



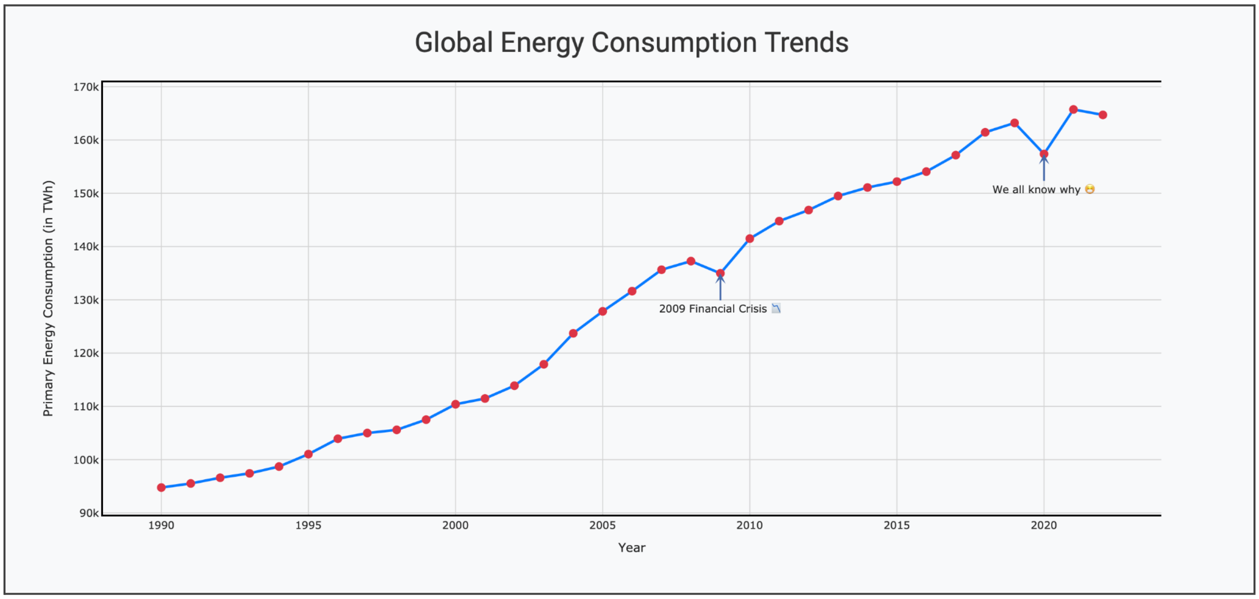
```python
mapper_script_path = "hdfs://namenode:9000/energy-data/null_percent_mapper.py"
mapper = "null_percent_mapper.py"
reducer_script_path = "hdfs://namenode:9000/energy-data/null_percent_reducer.py"
reducer = "null_percent_reducer.py"
input_path = "/energy-data/owid-energy-data.csv"
output_path = "/energy-data/output/null_percent"


clear_output_folder_np = BashOperator(
    task_id="clear_output_folder_np",
    bash_command=("docker exec resourcemanager hadoop fs -rm -f -R {output}").format(
        output=output_path
    ),
    dag=dag,
)

np_job = BashOperator(
    task_id="run_mapreduce_job_np",
    bash_command=(
        "docker exec resourcemanager "
        "hadoop jar /opt/hadoop-3.2.1/share/hadoop/tools/lib/hadoop-streaming-3.2.1.jar "
        "-files {mapper_path},{reducer_path} "
        "-mapper {mapper} "
        "-reducer {reducer} "
        "-input {input} "
        "-output {output}"
        " -verbose "
    ).format(
        mapper_path=mapper_script_path,
        reducer_path=reducer_script_path,
        mapper=mapper,
        reducer=reducer,
        input=input_path,
        output=output_path,
    ),
    dag=dag,
)

clear_output_folder_np >> np_job >> clear_output_folder_eda >> eda_job
```

There is also a main dag to trigger the entire data pipeline from start to finish.

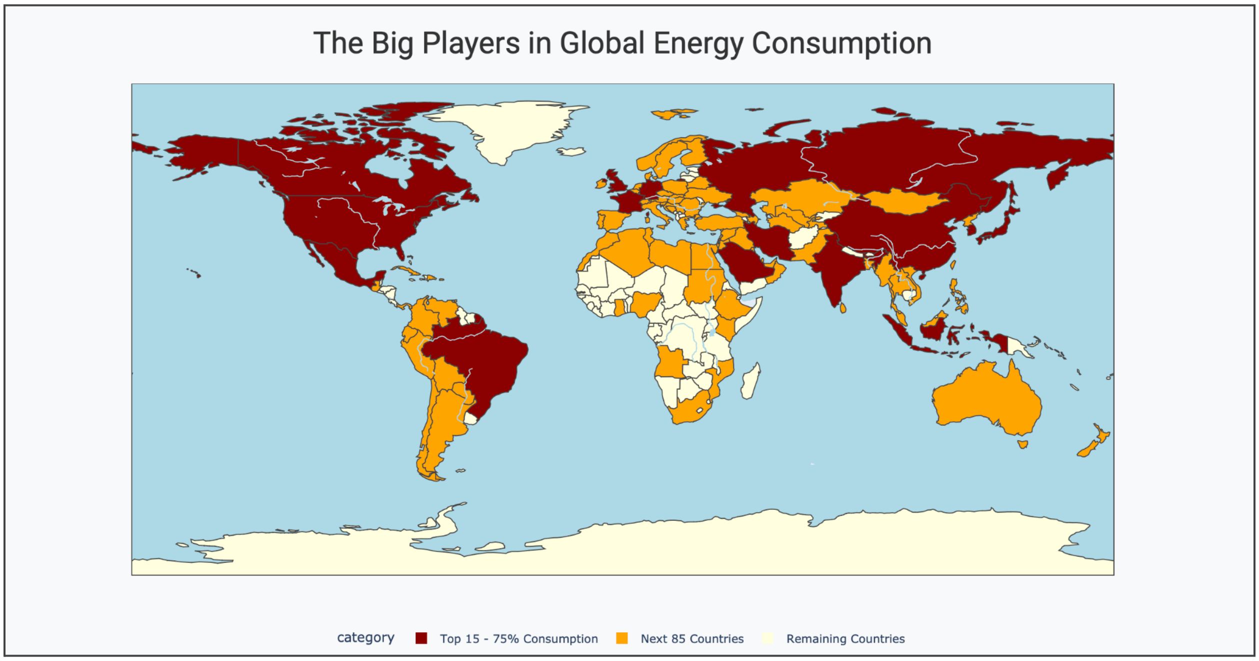


5. INSIGHTS (Plotly dashboard link)

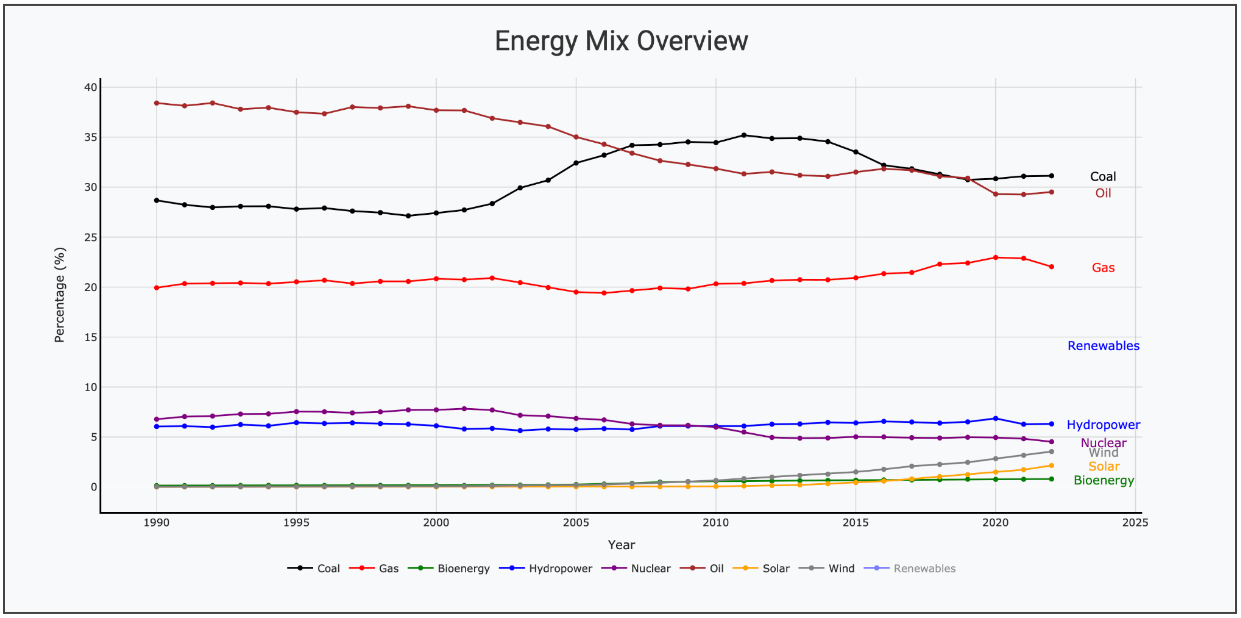
1. Energy Consumption Trends - The world's energy consumption has surged by over 70 percent in the past three decades. A snapshot of global energy utilization in the year 2021 reveals a total consumption of 164,710.98 terawatt-hours (TWH) of primary energy. We see that global energy consumption has increased nearly every year for more than 3 decades. The exceptions to this are in early 2009 (due to financial crisis) and 2020 (year of covid pandemic).



2. The Big Players - Top 15 countries - 'China', 'United States', 'India', 'Russia', 'Japan', 'Canada', 'Brazil', 'South Korea', 'Germany', 'Iran', ’Saudi Arabia’, ’France’, ’Mexico’, ’Indonesia’, ’United Kingdom’, accounting for 75% of global energy consumption, highlight the importance of major economies leading the clean energy transition. Over 130 countries with consumption below 100 tWH face the dual challenge of energy poverty and sustainable development. These nations have a unique opportunity to learn from the larger economies' experiences, adopting best practices and leapfrogging to cleaner, more sustainable energy solutions.

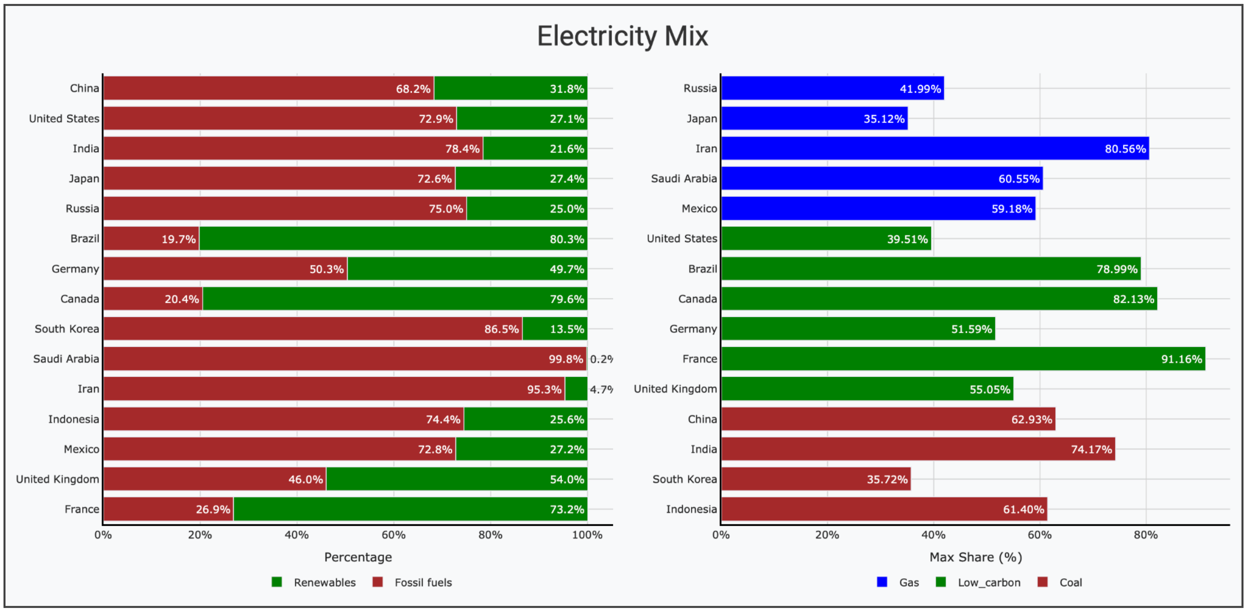

The Big Players in Global Energy Consumption
category
🟥 Top 15 - 75% Consumption
🟧 Next 85 Countries
Remaining Countries

3. Energy Mix - Coal consumption increased significantly with nearly 90% hike between 1990 and 2010, then stabilized, while the change in other fossil energy source consumption has remained nearly constant. There has been a marked increase in the use of Solar and Wind energy sources in the past decade with ~27% and ~50% change. Nuclear energy usage has remained constant over the past three, possibly due to safety concerns, despite its potential as a fossil fuel alternative. Biofuel is another renewable source which is increasing slowly over past decades and might become more familiar in the future.

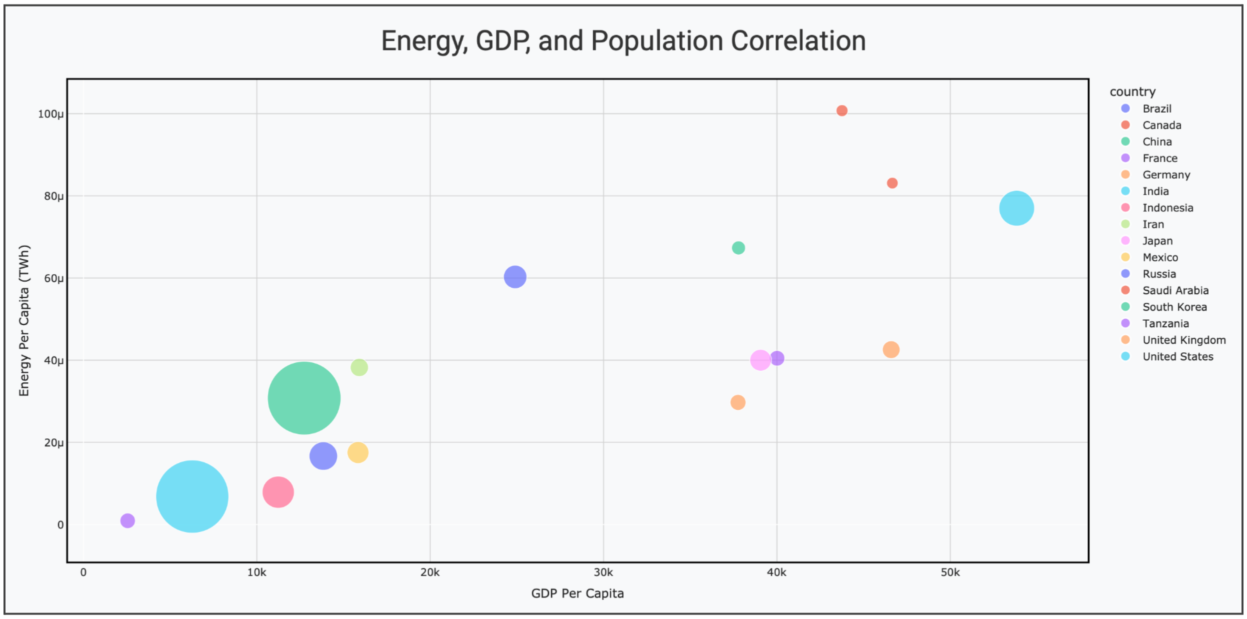


4. Electricity Generation from fossil fuels and renewables- Countries like Brazil, Canada, and France, among the top electricity-generating countries, distinguish themselves by prioritizing renewable energy. In contrast, major

contributors like China, the United States, India, Russia, and Japan exhibit a higher reliance on fossil fuels in their electricity generation. This disparity underscores the global challenge of transitioning towards cleaner energy sources and emphasizes the need for concerted efforts to promote sustainability.The 'Max Share' % reveals key insights into each country's primary electricity generation sources in 2021. For instance, China and India rely heavily on coal , while the United Kingdom emphasizes low-carbon sources. Brazil and Canada showcase a commitment to low-carbon energy, with shares of 79% and 82%, respectively. France stands out with a remarkable 91% share from low-carbon sources, primarily nuclear and renewables.



5. Energy, GDP and Population - Global energy consumption patterns highlight a complex relationship that extends beyond population size. While the USA and Canada, with smaller populations, exhibit significantly higher per capita energy use compared to populous nations like China and India, the disparity in consumption is even more pronounced when comparing countries like France and Tanzania, which have similar population sizes.This divergence underscores the critical role of energy accessibility and availability in shaping consumption trends.Enhancing energy access in countries with lower consumption rates could markedly improve living standards, suggesting that factors other than population are key determinants of national energy usage.



5. FUTURE WORK

Our project can contribute to a more dynamic and responsive approach to global energy challenges, fostering sustainable development and improving energy accessibility worldwide, by addressing these future considerations:

1. Explore opportunities for nations to strategically align with the clean energy practices observed in leading countries, fostering sustainable development and addressing energy challenges. Investigate potential collaborations and knowledge-sharing mechanisms to accelerate the adoption of cleaner energy sources globally.
2. Develop comprehensive policy measures for countries aspiring to transition towards cleaner energy sources, drawing insights from the consumption patterns of top electricity generators. Propose tailored strategies that consider the unique socio-economic and infrastructural contexts of individual nations to ensure effective policy implementation.
3. Apply the lessons learned from global energy consumption trends to formulate actionable plans for enhancing energy accessibility and living standards, particularly in countries with lower consumption rates. Design interventions that prioritize inclusivity, affordability, and sustainability, leveraging innovative technologies and cross-sectoral collaborations.
4. Explore advancements in Big data storage and processing technologies beyond Hadoop HDFS, MapReduce, Spark, Hive, and Docker. Investigate emerging solutions that enhance scalability, fault tolerance, and efficiency to keep pace with evolving data processing demands in the energy sector.

5. Evolve visualization techniques beyond Plotly to offer more sophisticated and interactive representations of energy consumption trends. Incorporate emerging visualization tools and technologies to provide stakeholders with deeper insights and facilitate better-informed decision-making.
6. Further optimize data processing workflows using Apache Airflow, ensuring seamless integration with evolving technologies and platforms. Explore automation enhancements and new features within Apache Airflow to streamline and enhance the efficiency of dynamic data processing tasks when a new batch of data is available.
7. Implement a robust monitoring and evaluation framework to continuously assess the impact of deployed technologies and methodologies. Regularly update the project infrastructure to align with the latest advancements in data science, energy technologies, and sustainability practices.

6. REFERENCES

1. Hannah Ritchie, Max Roser and Pablo Rosado (2023) - “Energy” Published online at OurWorldInData.org. Retrieved from: 'https://ourworldindata.org/energy' [Online Resource]
2. Oguz Ozan Yolcan, World energy outlook and state of renewable energy: 10-Year evaluation, Innovation and Green Development, Volume 2, Issue 4,2023,100070,ISSN 2949-7531,https://doi.org/10.1016/j.igd.2023.100070.
3. (https://www.sciencedirect.com/science/article/pii/S2949753123000383)
4. https://en.wikipedia.org/wiki/Renewable_energy
5. https://www.kaggle.com/datasets/pralabhpoudel/world-energy-consumption/data
6. Almozaini, M. S. (2019). The Causality Relationship between Economic Growth and Energy Consumption in The World’s Top Energy Consumers. *International Journal of Energy Economics and Policy, 9*(4), 40-53. http://ezproxy.lib.torontomu.ca/login?url=https://www.proquest.com/scholarly-journals/causality-relationship-between-economic-growth/docview/2256123472/se-2
7. T. Kober, H.-W. Schiffer, M. Densing, E. Panos,Global energy perspectives to 2060 – WEC's World Energy Scenarios 2019,Energy Strategy Reviews,Volume 31,2020,100523,ISSN 2211-467X,https://doi.org/10.1016/j.esr.2020.100523. (https://www.sciencedirect.com/science/article/pii/S2211467X20300766)
8. Samuel Adams, Edem Kwame Mensah Klobodu, Alfred Apio,Renewable and non-renewable energy, regime type and economic growth,Renewable Energy,Volume 125,2018,Pages 755-767,ISSN 0960-1481,https://doi.org/10.1016/j.renene.2018.02.135(https://www.sciencedirect.com/science/article/pii/S0960148118302878)
9. https://www.kaggle.com/code/gianlab/consumption-of-renewable-non-renew-electricity