**Exploratory data analysis on Azure Databricks**

**Name: PRIYESHWAR**

**Mail:** [**priyesh2664@gmail.com**](mailto:priyesh2664@gmail.com)

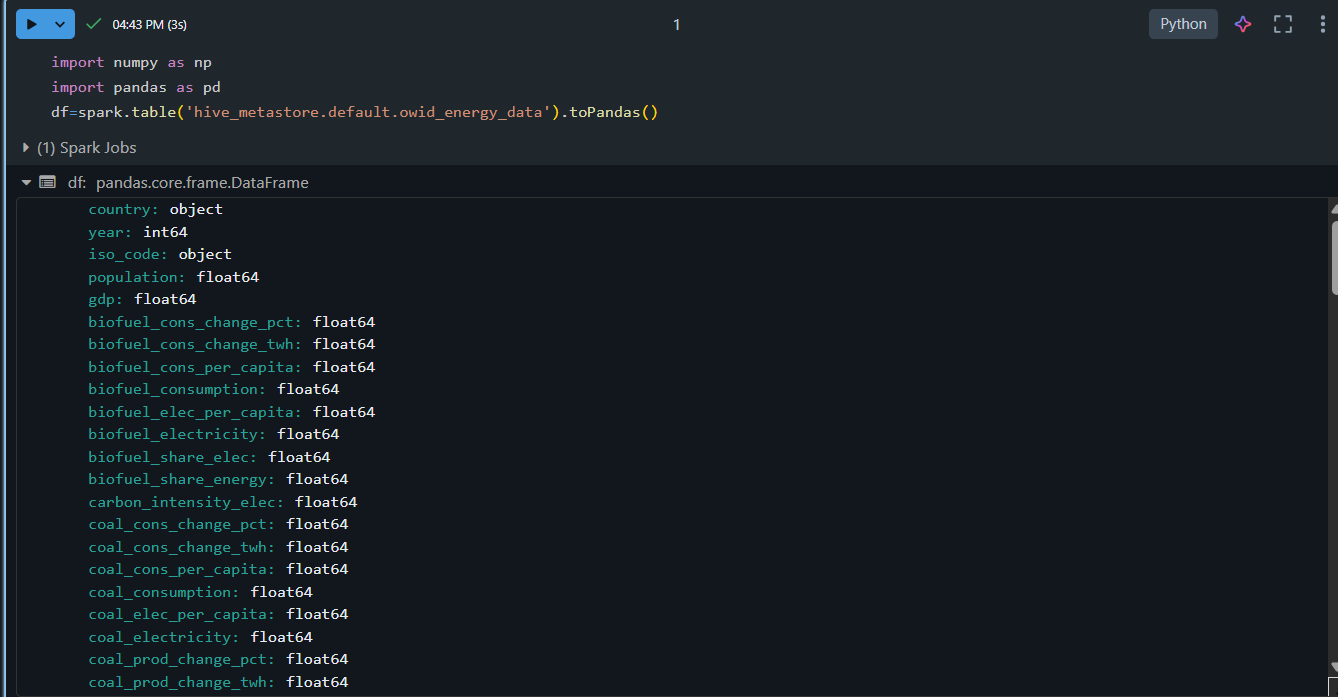
**Load CSV file**

Create a pandas DataFrame from the dataset for easier processing and visualization. Replace the file path below with the one you copied earlier.

**import numpy as np**

**import pandas as pd**

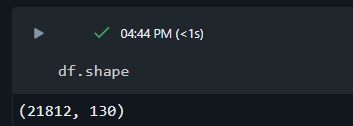
**df=spark.table('hive\_metastore.default.owid\_energy\_data').toPandas()**



**Use pandas for data insights**

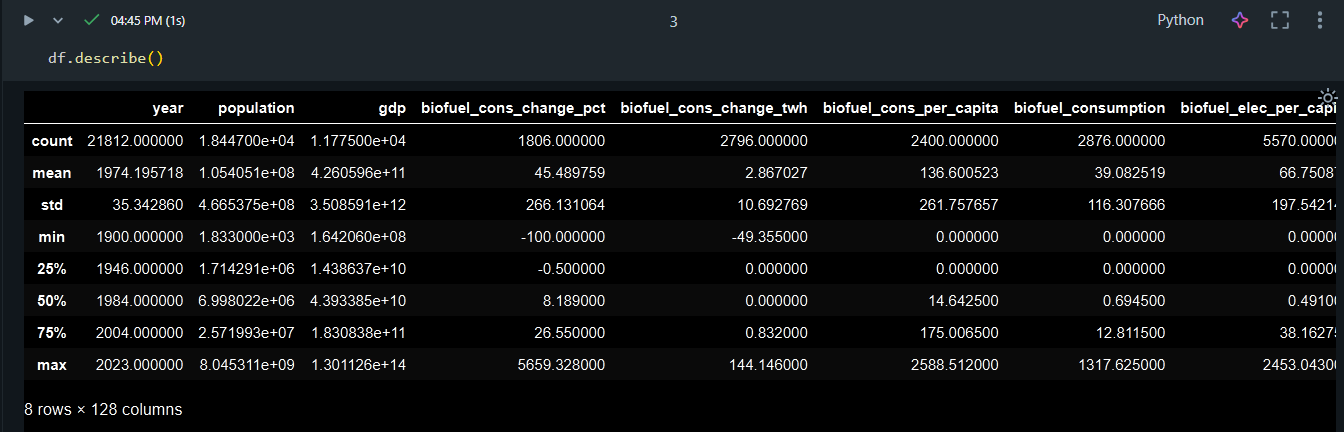
The df.shape command returns the dimensions of the DataFrame, giving you a quick overview of the number of rows and columns.

**df.shape**

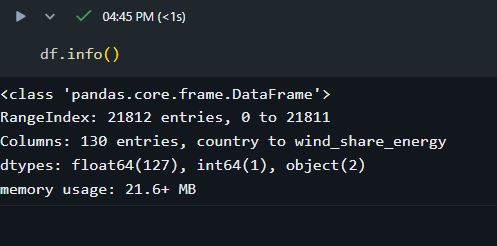


**df.describe()**

The df.describe() command generates descriptive statistics for numerical columns, such as mean, standard deviation, and percentiles, which can help you identify patterns, detect anomalies, and understand the distribution of your data.



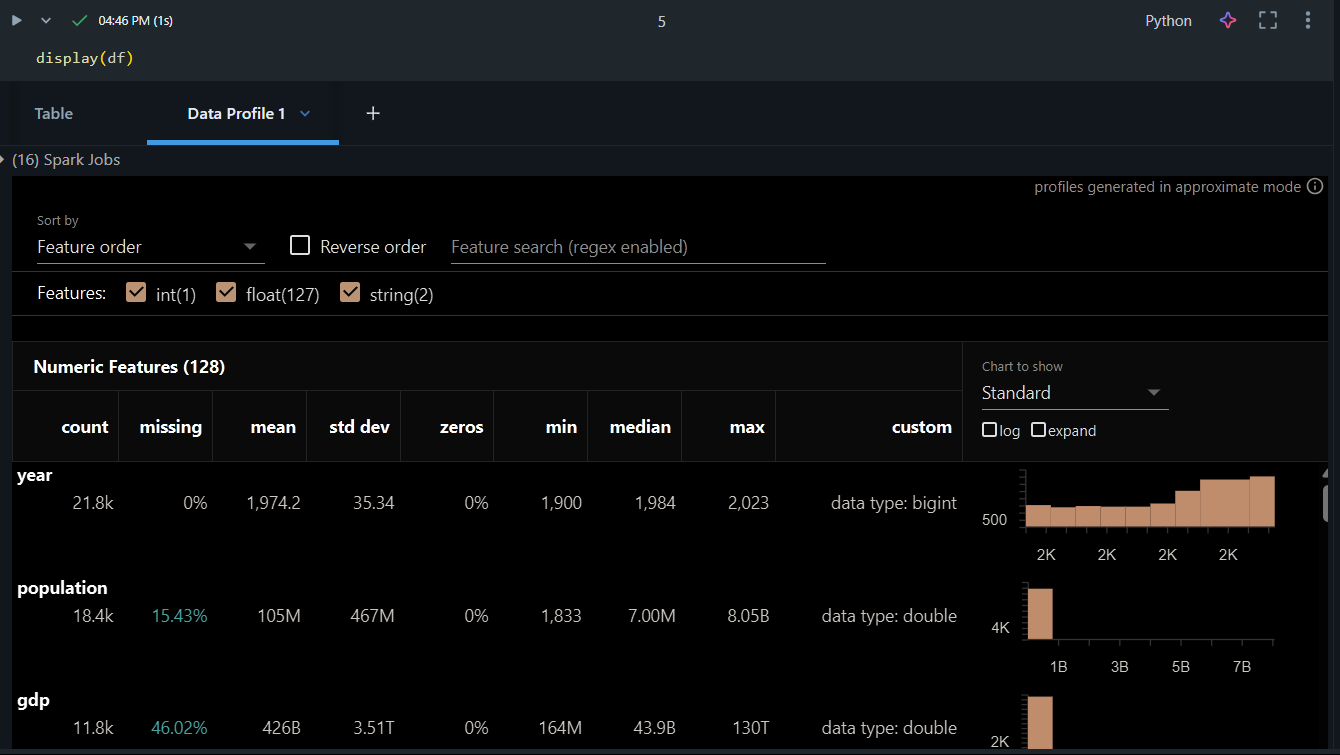
**df.info()**



**Generate a data profile**

Click + > Data Profile next to the Table in the output. This runs a new command that generates a profile of the data in the DataFrame.

**display(df)**



**Clean the data**

**Remove duplicate data**

Check if the data has any duplicate rows or columns. If so, remove them.

**duplicate\_rows = df.duplicated().sum()**

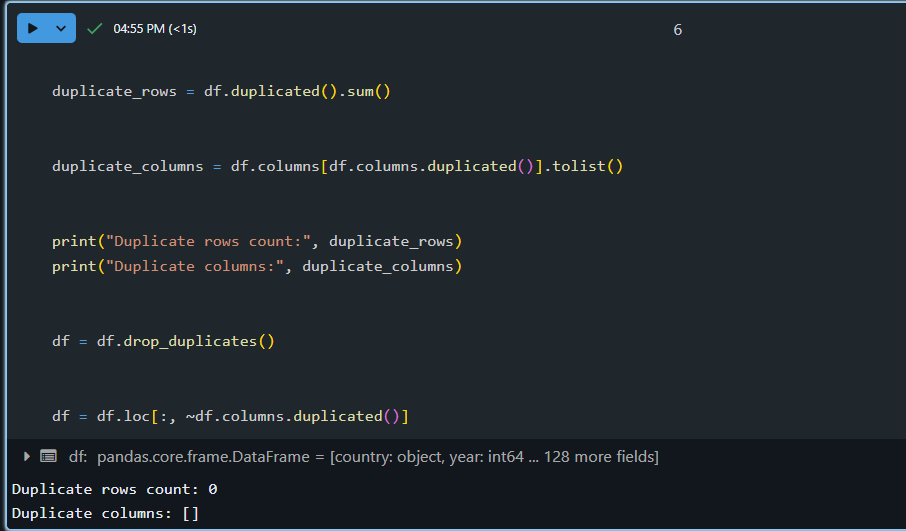
**duplicate\_columns = df.columns[df.columns.duplicated()].tolist()**

**print("Duplicate rows count:", duplicate\_rows)**

**print("Duplicate columns:", duplicate\_columns)**

**df = df.drop\_duplicates()**

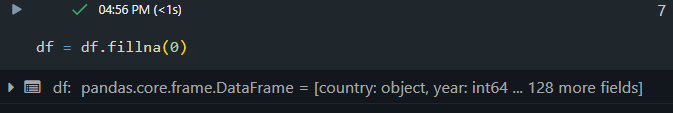
**df = df.loc[:, ~df.columns.duplicated()]**



**Handle null or missing values**

A common way to treat NaN or Null values is to replace them with 0 for easier mathematical processing.

**df = df.fillna(0)**



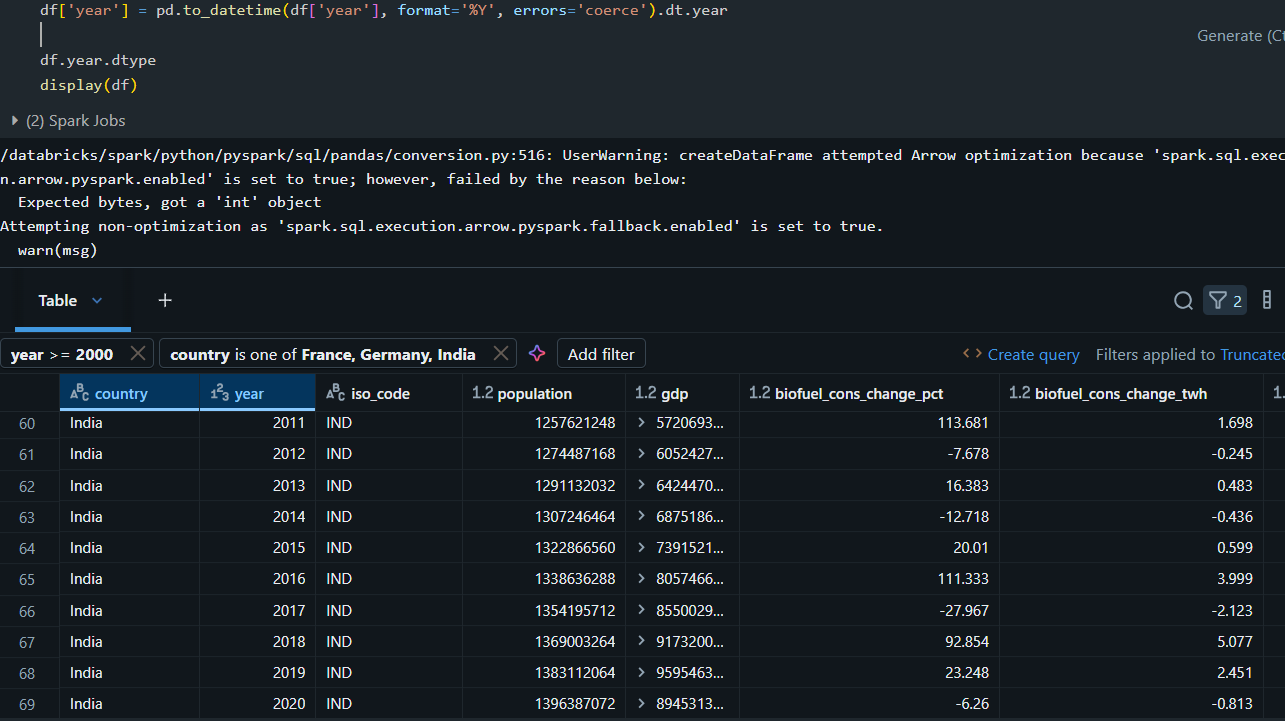
**Reformat dates**

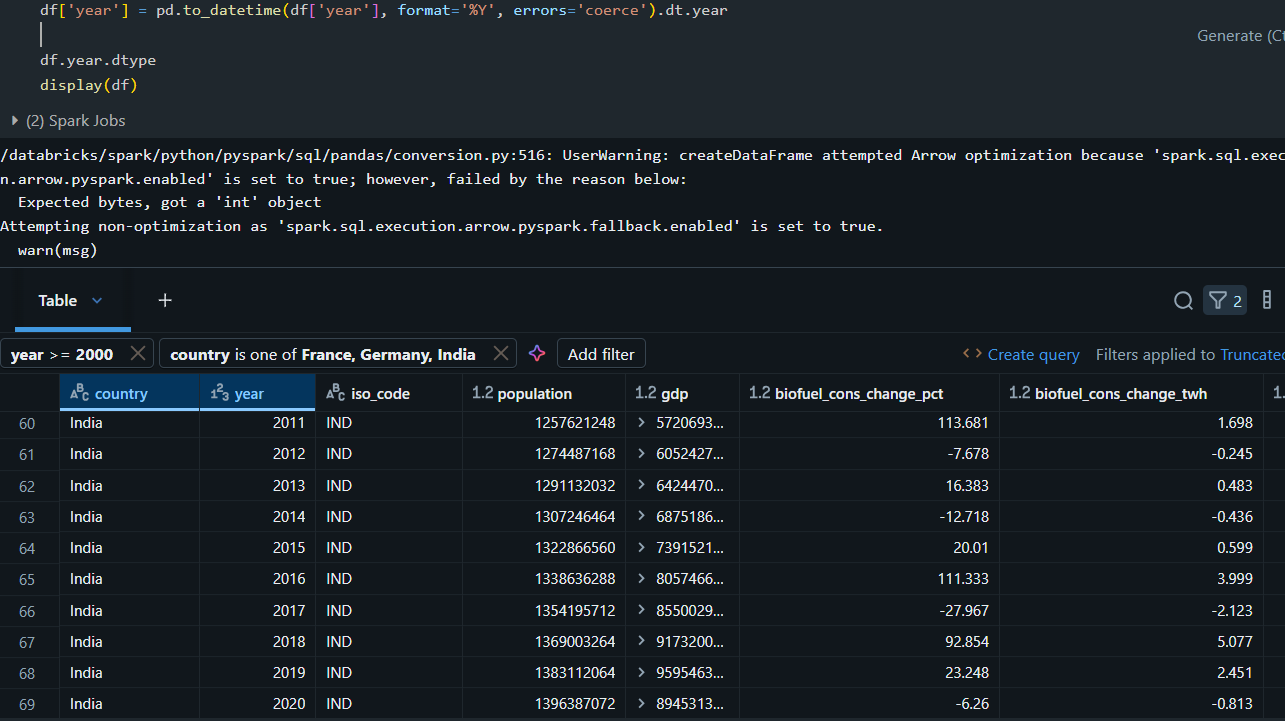
Dates are often formatted in various ways in different datasets. They might be in date format, strings, or integers.

**df['year'] = pd.to\_datetime(df['year'], format='%Y', errors='coerce').dt.year**

**df.year.dtype**

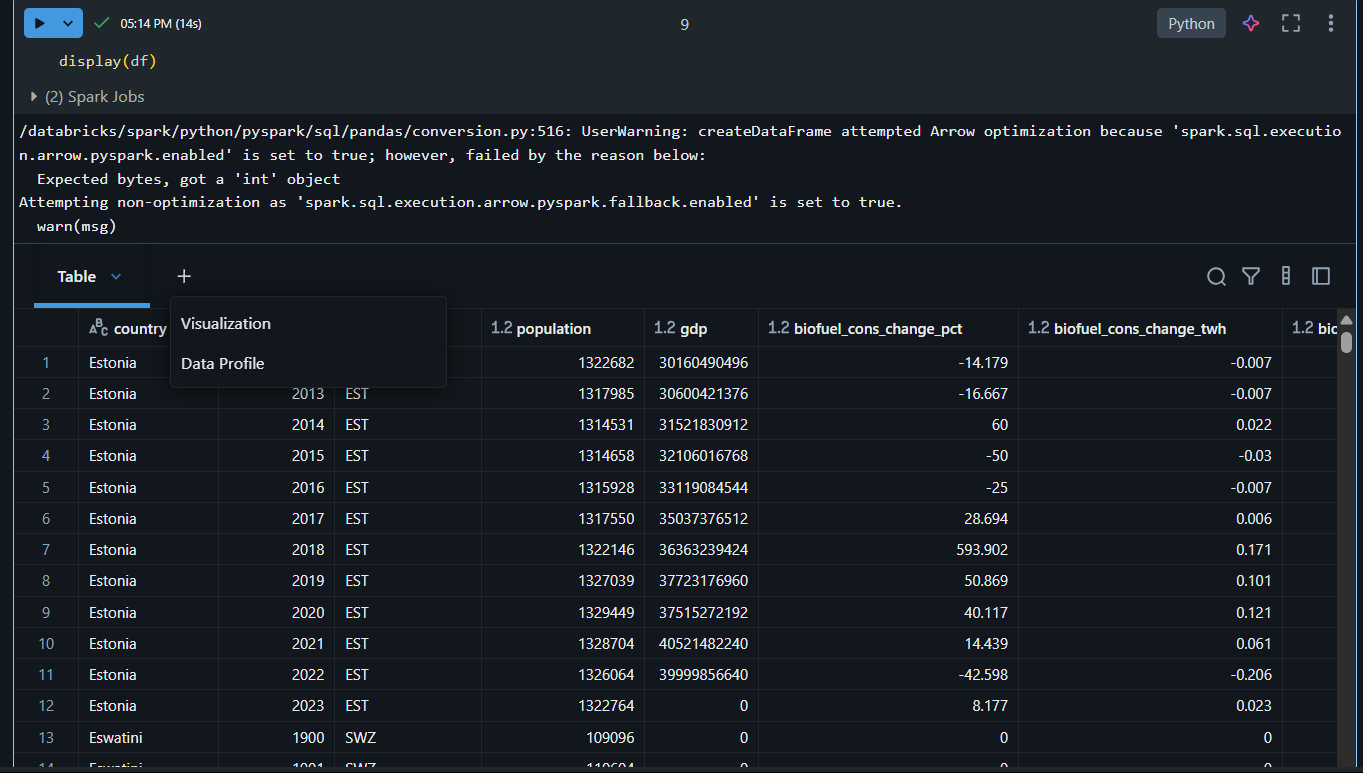
**display(df)**

 **Filter for specific conditions**

You can use built-in table filters to filter your columns for specific conditions. There are several ways to create a filter.

**Create visualizations using the dataset**

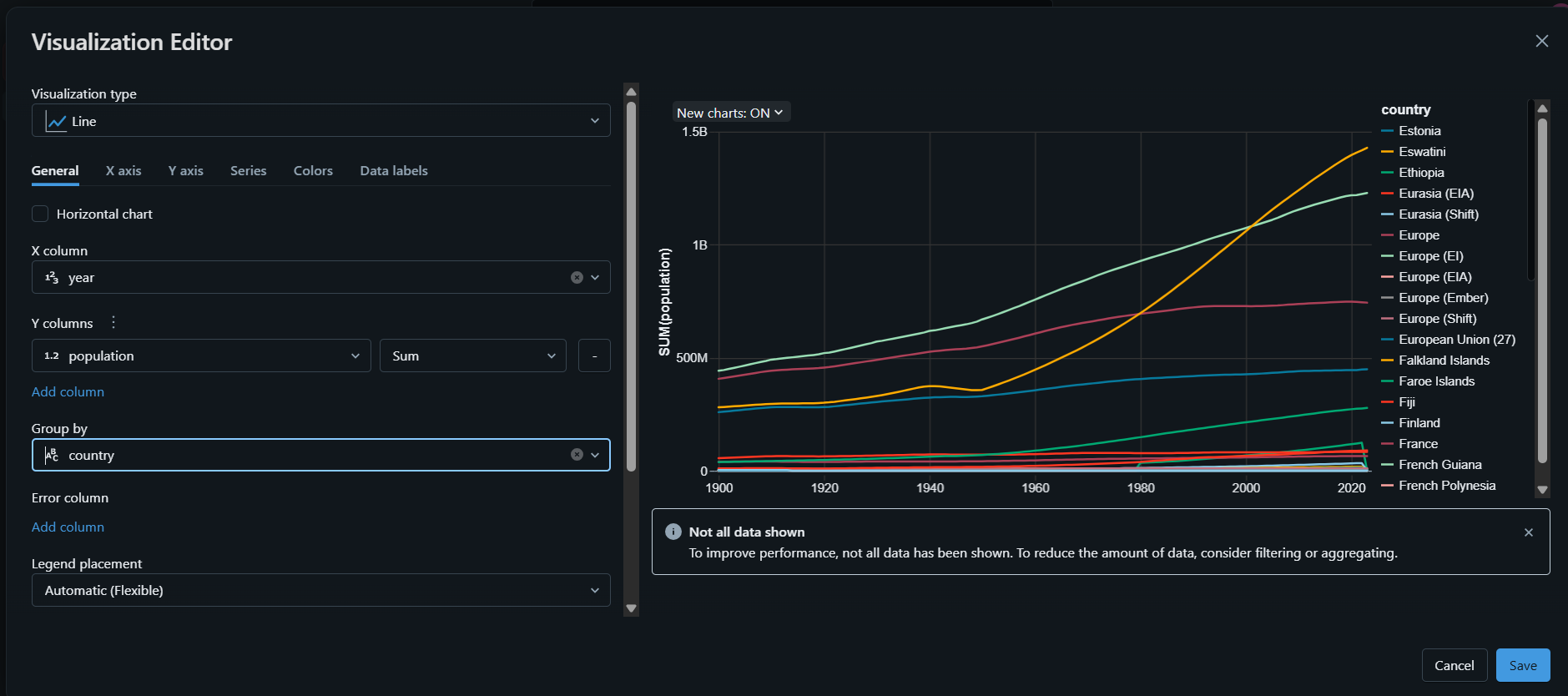
At the top of the output table, click **+** > **Visualization** to open the visualization editor.



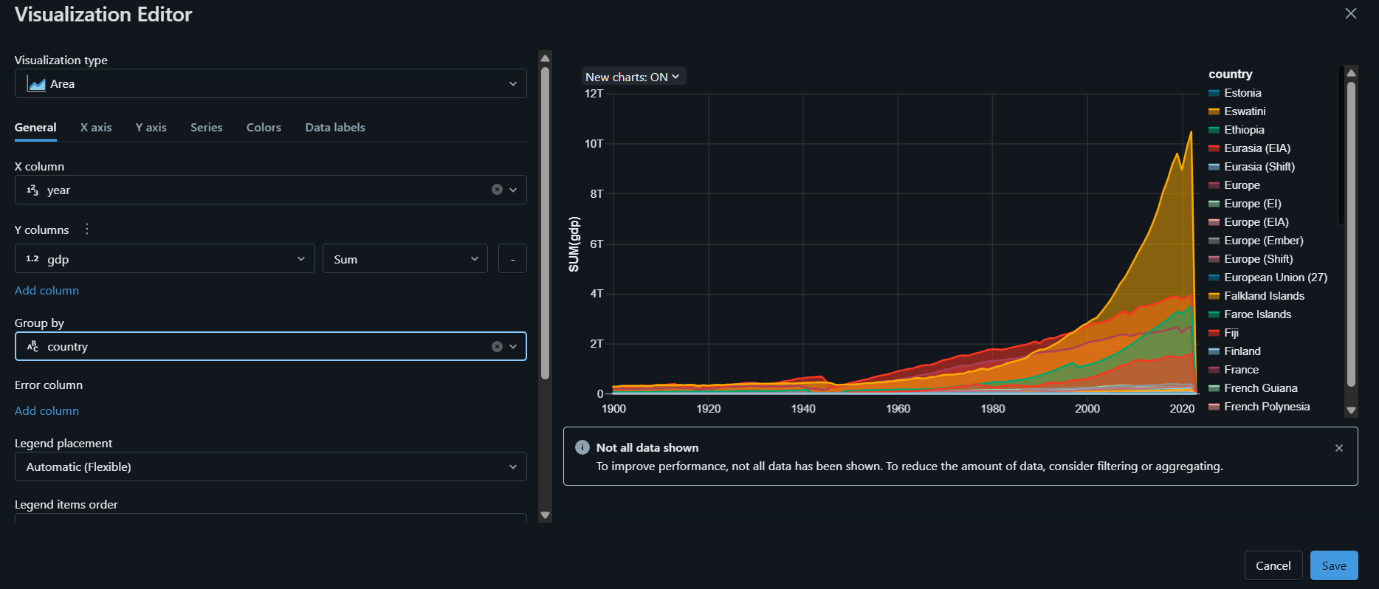
**Visualization**

Select the visualization type and columns you'd like to visualize. The editor displays a preview of the chart based on your configuration

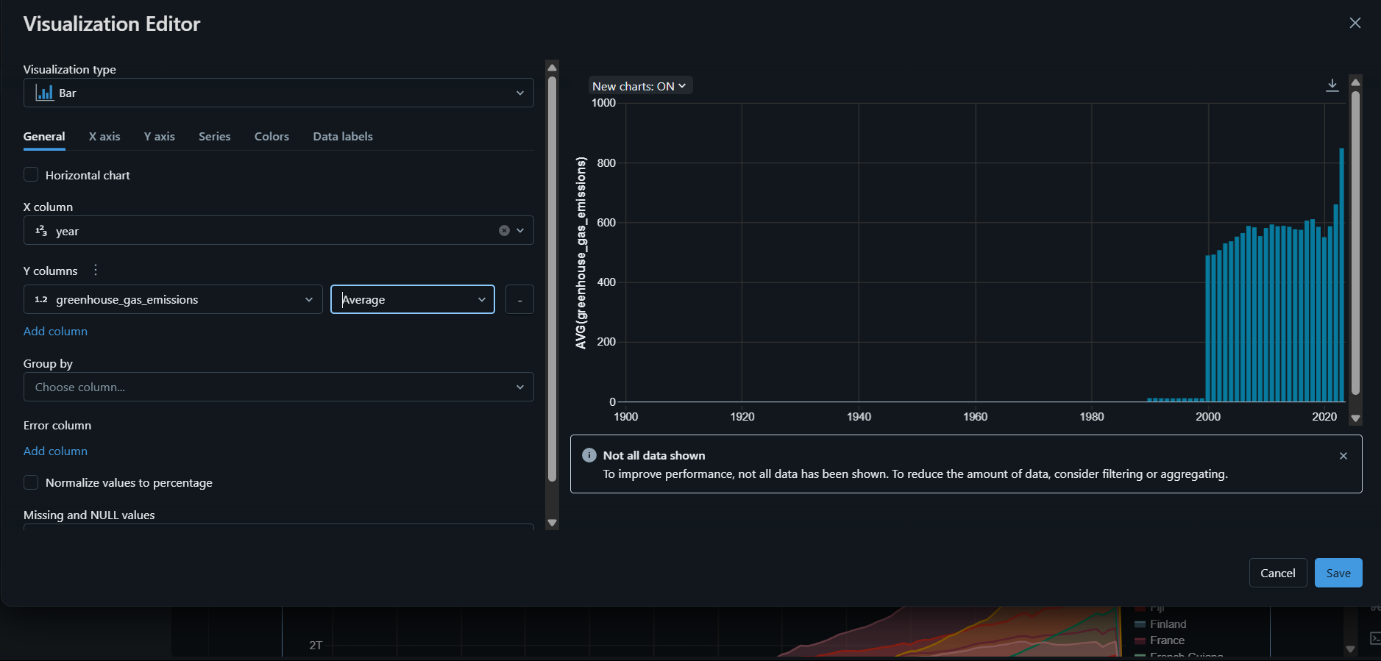
**Line Chart**



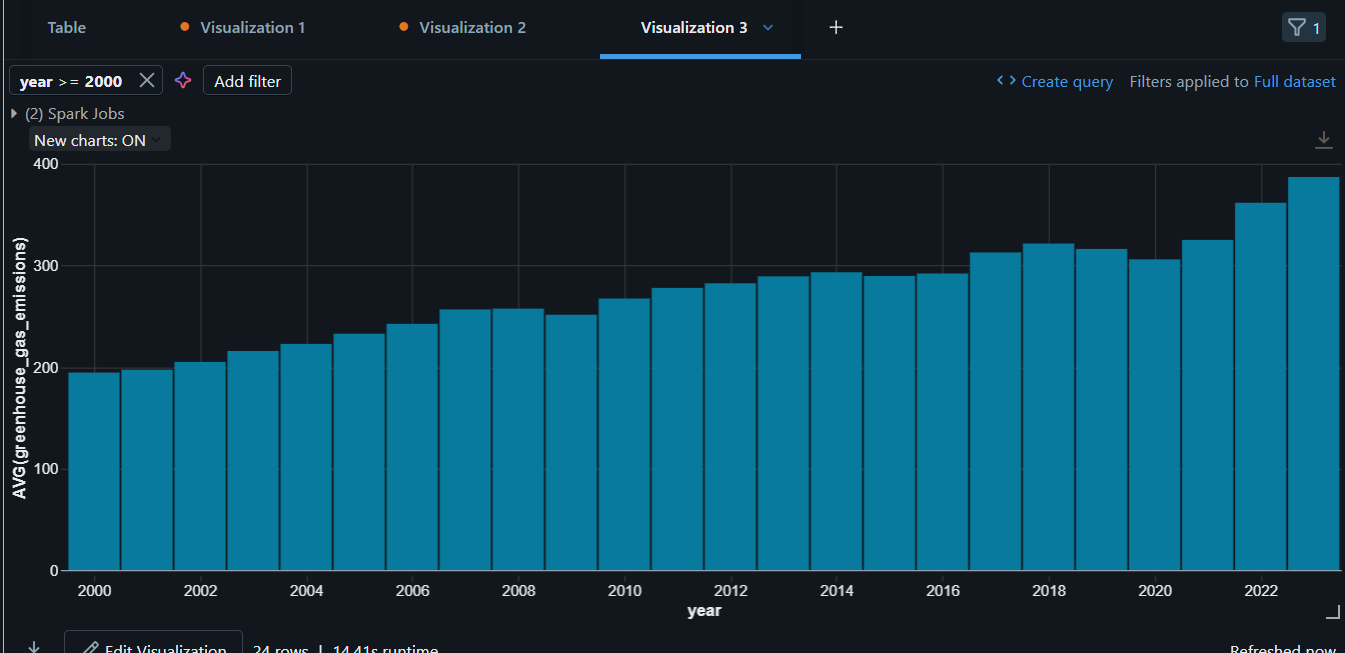
**Area Chart**

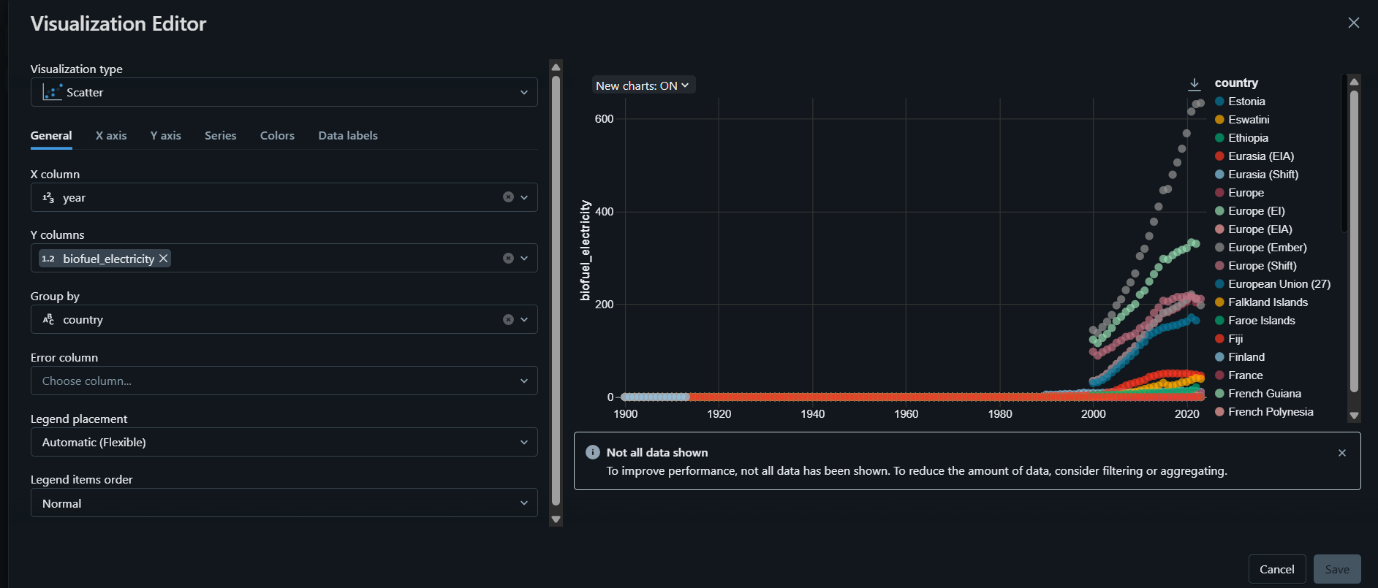


**Bar Chart**

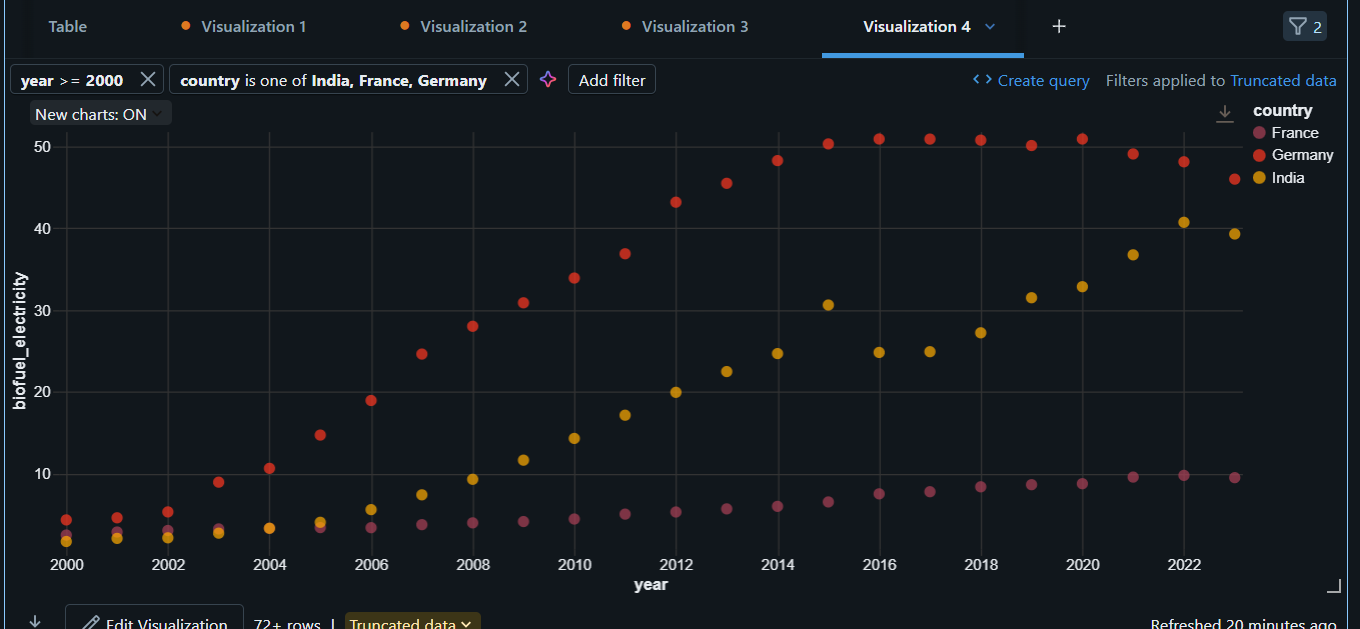


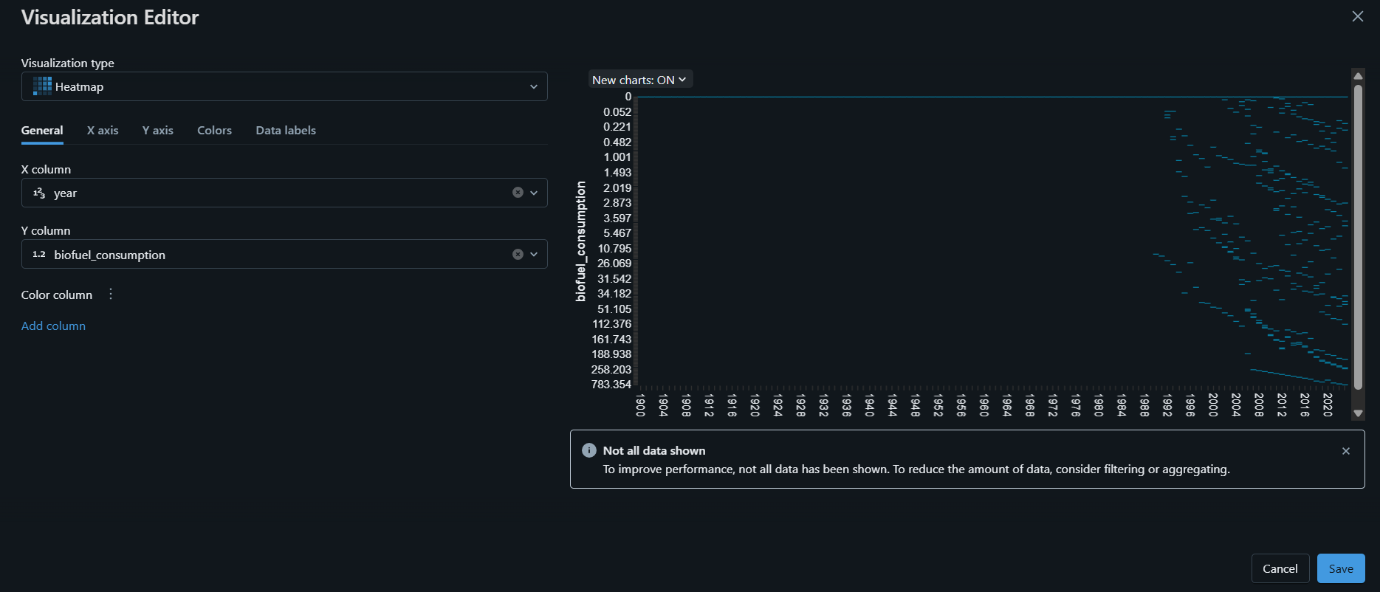
**Bar Chart with Filters**

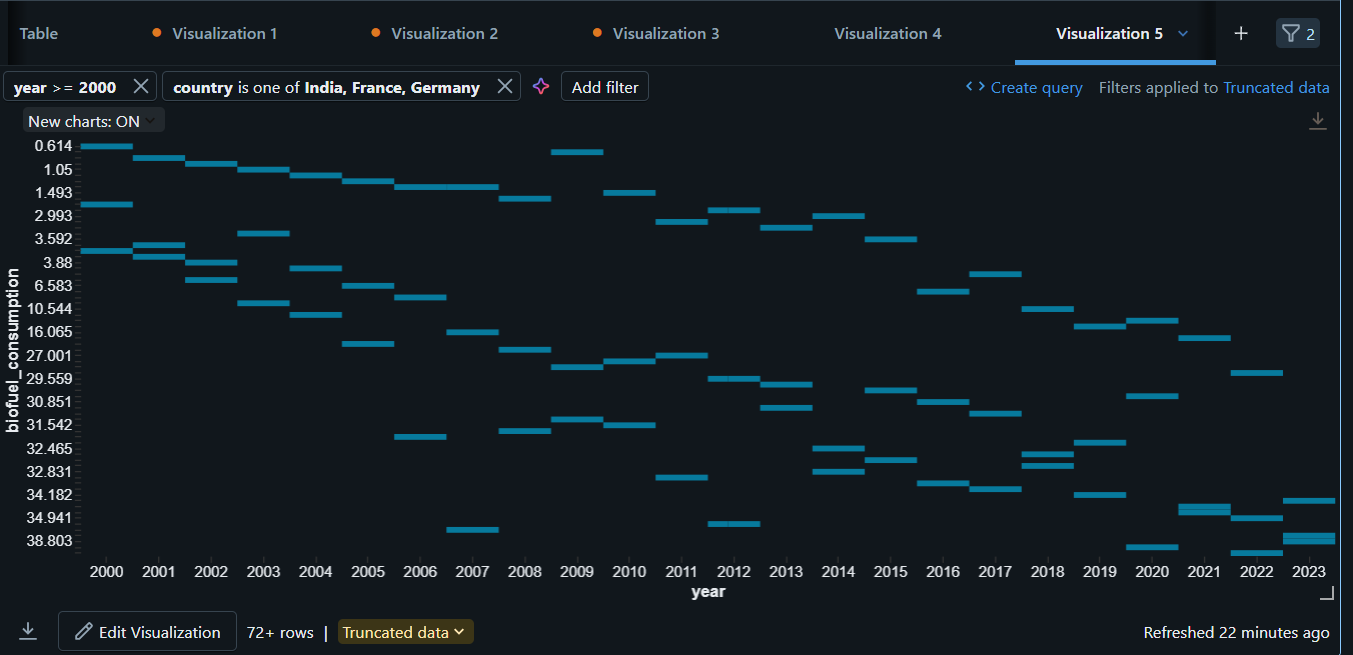


**Scatter Plot**

**Scatter Plot with Filters**



**Heatmap** 

**Heatmap with Filters**

**Combo Chart**

