Project Name

Global Vaccination & Disease Insights

Presented by:Ennawar Balaji

Project Summary

This project focuses on a comprehensive Vaccination Data Analysis and Visualization initiative within the domain of Public Health and Epidemiology. The primary objective is to leverage global vaccination and disease data to gain actionable insights into vaccination strategies and their impact on disease control

Objective

To develop a robust data analysis pipeline and an interactive Power BI dashboard to provide actionable insights into global vaccination coverage, disease incidence, and the effectiveness of immunization programs. By leveraging Python for data cleaning and EDA, SQL for data integrity, and Power BI for dynamic visualization, the project aims to:

Identify gaps in vaccination coverage and regional disparities to inform targeted public health interventions.

Analyze the impact of vaccination campaigns on disease reduction and support evidence-based policy decisions.

Create a user-friendly dashboard that empowers public health organizations, governments, and researchers to monitor progress and allocate resources efficiently

Start coding or generate with AI.

```
import pandas as pd
import numpy as np
import os
import sqlite3

# --- Step 1: Load the datasets into pandas DataFrames ---
# Check if the code is running in a Colab environment or locally.
base_path = './'

try:
    # Adding 'encoding='latin1'' to handle potential UnicodeDecodeError.
    coverage_df = pd.read_csv(f'{base_path}coverage-data.csv', encoding='latin1')
    incidence_df = pd.read_csv(f'{base_path}incidence-rate-data.csv', encoding='latin1')
    cases_df = pd.read_csv(f'{base_path}reported-cases-data.csv', encoding='latin1')
    intro_df = pd.read_csv(f'{base_path}vaccine-introduction-data.csv', encoding='latin1')
    schedule_df = pd.read_csv(f'{base_path}vaccine-schedule-data.csv', encoding='latin1')

    print("Successfully loaded all data files.")

except FileNotFoundError as e:
    print(f"Error: One or more data files not found. Please ensure all files are in the same directory as the script.")
    print(e)
    # Exit if a critical file is missing
    exit()

# --- Step 2: Perform Initial Data Cleaning and EDA on each dataset ---
print("\n--- Initial Cleaning and EDA ---")

# --- Cleaning for 'coverage_df' ---
# Standardize all column names to lowercase for consistency
coverage_df.columns = coverage_df.columns.str.lower()
coverage_df.rename(columns={'group': 'Category', 'name': 'CountryName'}, inplace=True)
coverage_df['coverage'].fillna(0, inplace=True)
coverage_df['year'] = pd.to_numeric(coverage_df['year'], errors='coerce').astype('Int64')
coverage_df['target_number'] = pd.to_numeric(coverage_df['target_number'], errors='coerce')
coverage_df['doses'] = pd.to_numeric(coverage_df['doses'], errors='coerce')
coverage_df['coverage'] = pd.to_numeric(coverage_df['coverage'], errors='coerce')
print("\nCleaned Coverage Data:")
print(coverage_df.info())
print(coverage_df.head())

# --- Cleaning for 'incidence_df' ---
# Standardize all column names to lowercase for consistency
incidence_df.columns = incidence_df.columns.str.lower()
incidence_df.rename(columns={'name': 'CountryName', 'incidence_rate': 'IncidenceRate'}, inplace=True)
incidence_df['IncidenceRate'] = pd.to_numeric(incidence_df['IncidenceRate'], errors='coerce')
incidence_df['year'] = pd.to_numeric(incidence_df['year'], errors='coerce').astype('Int64')
print("\nCleaned Incidence Rate Data:")
print(incidence_df.info())
print(incidence_df.head())

# --- Cleaning for 'cases_df' ---
# Standardize all column names to lowercase for consistency
cases_df.columns = cases_df.columns.str.lower()
cases_df.rename(columns={'name': 'CountryName', 'cases': 'ReportedCases'}, inplace=True)
cases_df['ReportedCases'] = pd.to_numeric(cases_df['ReportedCases'], errors='coerce')
cases_df['year'] = pd.to_numeric(cases_df['year'], errors='coerce').astype('Int64')
print("\nCleaned Reported Cases Data:")
print(cases_df.info())
print(cases_df.head())

# --- Cleaning for 'intro_df' ---
# Standardize all column names to lowercase for consistency
intro_df.columns = intro_df.columns.str.lower()
intro_df.rename(columns={'countryname': 'CountryName'}, inplace=True)
intro_df['year'] = pd.to_numeric(intro_df['year'], errors='coerce').astype('Int64')
print("\nCleaned Vaccine Introduction Data:")
print(intro_df.info())
print(intro_df.head())

# --- Cleaning for 'schedule_df' ---
# Standardize all column names to lowercase for consistency
schedule_df.columns = schedule_df.columns.str.lower()
schedule_df.rename(columns={'countryname': 'CountryName'}, inplace=True)
schedule_df['year'] = pd.to_numeric(schedule_df['year'], errors='coerce').astype('Int64')
print("\nCleaned Vaccine Schedule Data:")
print(schedule_df.info())
print(schedule_df.head())

# --- Step 3: Store cleaned data in a SQL database (SQLite example) ---
print("\n--- Creating SQL Database and Tables ---")
conn = sqlite3.connect('vaccination_data.db')

try:
    coverage_df.to_sql('coverage', conn, if_exists='replace', index=False)
    incidence_df.to_sql('incidence_rate', conn, if_exists='replace', index=False)
    cases_df.to_sql('reported_cases', conn, if_exists='replace', index=False)
    intro_df.to_sql('vaccine_introduction', conn, if_exists='replace', index=False)
    schedule_df.to_sql('vaccine_schedule', conn, if_exists='replace', index=False)
    print("Successfully created tables in vaccination_data.db.")
except Exception as e:
    print(f"Error creating tables: {e}")

# --- Step 4: Perform EDA to answer a key project question ---
print("\n--- Exploratory Data Analysis: Correlation Analysis ---")

# Merge a subset of data for this analysis
# This is a key step that connects different parts of your project data.
eda_df = pd.merge(coverage_df, incidence_df, on=['code', 'year'], suffixes=('_coverage', '_incidence'))
eda_df = pd.merge(eda_df, cases_df, on=['code', 'year'], suffixes=('', '_cases'))

# Calculate correlation between Coverage and Incidence Rate
correlation_coverage_incidence = eda_df['coverage'].corr(eda_df['IncidenceRate'])
print(f"Correlation between Vaccination Coverage and Incidence Rate: {correlation_coverage_incidence:.2f}")

# Calculate correlation between Coverage and Reported Cases
correlation_coverage_cases = eda_df['coverage'].corr(eda_df['ReportedCases'])
print(f"Correlation between Vaccination Coverage and Reported Cases: {correlation_coverage_cases:.2f}")

conn.close()
print("\nEDA complete and SQL connection closed.")
```

```
Successfully loaded all data files.

--- Initial Cleaning and EDA ---

Cleaned Coverage Data:
/tmp/ipython-input-2959885028.py:33: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are settin

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[c


  coverage_df['coverage'].fillna(0, inplace=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 406384 entries, 0 to 406383
Data columns (total 11 columns):
 #   Column                         Non-Null Count   Dtype  
---  ------                         --------------   -----  
 0   Category                       406384 non-null  object 
 1   code                           406383 non-null  object 
 2   CountryName                    405109 non-null  object 
 3   year                           406383 non-null  Int64  
 4   antigen                        406383 non-null  object 
 5   antigen_description            406383 non-null  object 
 6   coverage_category              406383 non-null  object 
 7   coverage_category_description  406383 non-null  object 
 8   target_number                  84557 non-null   float64
 9   doses                          84851 non-null   float64
 10  coverage                       406384 non-null  float64
dtypes: Int64(1), float64(3), object(7)
memory usage: 34.5+ MB
None
    Category code CountryName  year  antigen  \
0  COUNTRIES  ABW       Aruba  2023      BCG   
1  COUNTRIES  ABW       Aruba  2023      BCG   
2  COUNTRIES  ABW       Aruba  2023  DIPHCV4   
3  COUNTRIES  ABW       Aruba  2023  DIPHCV4   
4  COUNTRIES  ABW       Aruba  2023  DIPHCV5   

                                 antigen_description coverage_category  \
0                                                BCG             ADMIN   
1                                                BCG          OFFICIAL   
2  Diphtheria-containing vaccine, 4th dose (1st b...             ADMIN   
3  Diphtheria-containing vaccine, 4th dose (1st b...          OFFICIAL   
4  Diphtheria-containing vaccine, 5th dose (2nd b...             ADMIN   

  coverage_category_description  target_number   doses  coverage  
0       Administrative coverage            NaN     NaN      0.00  
1             Official coverage            NaN     NaN      0.00  
2       Administrative coverage         1044.0   945.0     90.52  
3             Official coverage            NaN     NaN     90.52  
4       Administrative coverage         1219.0  1008.0     82.69  

Cleaned Incidence Rate Data:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 84946 entries, 0 to 84945
Data columns (total 8 columns):
 #   Column               Non-Null Count  Dtype
```