# DATA WAREHOUSING AND DATA MINING PROJECT (LOW LEVEL IMPLEMENTATION THROUGH IPYNB NOTEBOOK)


# CONTRIBUTORS

## SACHIN SINGH - 2023BCS0064

## ANSH RASTOGI - 2023BCS0151

## ABHINAV VS - 2023BCS0181

## TANISH BABU - 2023BCS0067


# COVID-19 Data Warehousing and Data Mining Analysis

## Objective

The objective of this project is to design a data warehousing pipeline and apply data mining techniques to analyze global COVID-19 trends. The project integrates ETL processes, OLAP operations, clustering analysis, outlier detection, association rule mining, and business intelligence visualizations to extract meaningful knowledge from pandemic data.

## Dataset

Source: Our World in Data (OWID) COVID-19 Dataset

The dataset contains country-level pandemic indicators including cases, deaths, vaccinations, demographic and economic attributes.

## Importing all the necessary modules for implementation

# 1. Importing Required Libraries

Libraries for:

- data manipulation (Pandas, NumPy)
- visualization (Matplotlib)
- clustering and mining (Scikit-learn)
- association rule mining

In [ ]:

```
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import linkage, dendrogram
from scipy.stats import zscore

from mlxtend.frequent_patterns import apriori, association_rules
```

# 2. Data Loading and Understanding

The dataset is loaded into a dataframe to understand structure, attributes, and completeness before preprocessing.

In [ ]:

```
import pandas as pd

df = pd.read_csv("owid-covid-data.csv")
df.head()
```

Out[ ]:

| | iso_code | continent | location | date | total_cases | new_cases | new_cas |
|---|---|---|---|---|---|---|---|
| **0** | AFG | Asia | Afghanistan | 2020-01-05 | 0.0 | 0.0 | |
| **1** | AFG | Asia | Afghanistan | 2020-01-06 | 0.0 | 0.0 | |
| **2** | AFG | Asia | Afghanistan | 2020-01-07 | 0.0 | 0.0 | |
| **3** | AFG | Asia | Afghanistan | 2020-01-08 | 0.0 | 0.0 | |
| **4** | AFG | Asia | Afghanistan | 2020-01-09 | 0.0 | 0.0 | |

5 rows × 67 columns

## Dataset Overview

We inspect attribute types and missing values to prepare for cleaning and transformation.

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 429435 entries, 0 to 429434
Data columns (total 67 columns):
 #   Column                                      Non-Null Count   Dtype  
---  ------                                      --------------   -----  
 0   iso_code                                    429435 non-null  object 
 1   continent                                   402910 non-null  object 
 2   location                                    429435 non-null  object 
 3   date                                        429435 non-null  object 
 4   total_cases                                 411804 non-null  float64
 5   new_cases                                   410159 non-null  float64
 6   new_cases_smoothed                          408929 non-null  float64
 7   total_deaths                                411804 non-null  float64
 8   new_deaths                                  410608 non-null  float64
 9   new_deaths_smoothed                         409378 non-null  float64
 10  total_cases_per_million                     411804 non-null  float64
 11  new_cases_per_million                       410159 non-null  float64
 12  new_cases_smoothed_per_million              408929 non-null  float64
 13  total_deaths_per_million                    411804 non-null  float64
 14  new_deaths_per_million                      410608 non-null  float64
 15  new_deaths_smoothed_per_million             409378 non-null  float64
 16  reproduction_rate                           184817 non-null  float64
 17  icu_patients                                39116 non-null   float64
 18  icu_patients_per_million                    39116 non-null   float64
 19  hosp_patients                               40656 non-null   float64
 20  hosp_patients_per_million                   40656 non-null   float64
 21  weekly_icu_admissions                       10993 non-null   float64
 22  weekly_icu_admissions_per_million           10993 non-null   float64
 23  weekly_hosp_admissions                      24497 non-null   float64
 24  weekly_hosp_admissions_per_million          24497 non-null   float64
 25  total_tests                                 79387 non-null   float64
 26  new_tests                                   75403 non-null   float64
 27  total_tests_per_thousand                    79387 non-null   float64
 28  new_tests_per_thousand                      75403 non-null   float64
 29  new_tests_smoothed                          103965 non-null  float64
 30  new_tests_smoothed_per_thousand             103965 non-null  float64
 31  positive_rate                               95927 non-null   float64
 32  tests_per_case                              94348 non-null   float64
 33  tests_units                                 106788 non-null  object 
 34  total_vaccinations                          85417 non-null   float64
 35  people_vaccinated                           81132 non-null   float64
 36  people_fully_vaccinated                     78061 non-null   float64
 37  total_boosters                              53600 non-null   float64
 38  new_vaccinations                            70971 non-null   float64
 39  new_vaccinations_smoothed                   195029 non-null  float64
 40  total_vaccinations_per_hundred              85417 non-null   float64
 41  people_vaccinated_per_hundred               81132 non-null   float64
 42  people_fully_vaccinated_per_hundred         78061 non-null   float64
 43  total_boosters_per_hundred                  53600 non-null   float64
 44  new_vaccinations_smoothed_per_million       195029 non-null  float64
 45  new_people_vaccinated_smoothed              192177 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  192177 non-null  float64
 47  stringency_index                            196190 non-null  float64
 48  population_density                          360492 non-null  float64
```

```
 49  median_age                                  334663 non-null  float64
 50  aged_65_older                               323270 non-null  float64
 51  aged_70_older                               331315 non-null  float64
 52  gdp_per_capita                              328292 non-null  float64
 53  extreme_poverty                             211996 non-null  float64
 54  cardiovasc_death_rate                       328865 non-null  float64
 55  diabetes_prevalence                         345911 non-null  float64
 56  female_smokers                              247165 non-null  float64
 57  male_smokers                                243817 non-null  float64
 58  handwashing_facilities                      161741 non-null  float64
 59  hospital_beds_per_thousand                  290689 non-null  float64
 60  life_expectancy                             390299 non-null  float64
 61  human_development_index                     319127 non-null  float64
 62  population                                  429435 non-null  int64
 63  excess_mortality_cumulative_absolute        13411 non-null   float64
 64  excess_mortality_cumulative                 13411 non-null   float64
 65  excess_mortality                            13411 non-null   float64
 66  excess_mortality_cumulative_per_million     13411 non-null   float64
dtypes: float64(61), int64(1), object(5)
memory usage: 219.5+ MB
```

# 3. Attribute Selection

Relevant attributes were selected based on pandemic impact and response indicators.
This reduces dimensionality and improves mining efficiency.

In [ ]:
```
cols = [
    'location',
    'continent',
    'date',
    'total_cases',
    'total_deaths',
    'people_vaccinated',
    'population',
    'gdp_per_capita'
]

df = df[cols]
```

# 4. Data Cleaning

Cleaning removes inconsistencies:

- rows without continent information removed
- missing numerical values replaced
- date converted to datetime format

In [ ]:
```
df = df.dropna(subset=['continent'])
```

In [ ]:
```
df[['total_cases','total_deaths','people_vaccinated']] = \
df[['total_cases','total_deaths','people_vaccinated']].fillna(0)
```

In [ ]:
```
df['date'] = pd.to_datetime(df['date'])
```

# 5. Data Transformation (Feature Engineering)

New analytical indicators are created:

- Death Rate → severity indicator
- Vaccination Rate → healthcare response effectiveness

In [ ]:
```
df['death_rate'] = (
    df['total_deaths'] / df['total_cases']
).replace([np.inf, -np.inf],0).fillna(0)

df['vaccination_rate'] = (
    df['people_vaccinated'] / df['population']
).replace([np.inf, -np.inf],0).fillna(0)
```

In [ ]:
```
df[['death_rate','vaccination_rate']].head()
```

Out[ ]:

| | death_rate | vaccination_rate |
|---|---|---|
| **0** | 0.0 | 0.0 |
| **1** | 0.0 | 0.0 |
| **2** | 0.0 | 0.0 |
| **3** | 0.0 | 0.0 |
| **4** | 0.0 | 0.0 |

# 6. Data Discretization and Concept Hierarchy

Population values are categorized into levels to support multidimensional OLAP analysis.

In [ ]:
```
def pop_category(x):
    if x < 1e7:
        return "Low"
    elif x < 1e8:
        return "Medium"
    else:
        return "High"
```

```
df['pop_category'] = df['population'].apply(pop_category)
```

In [ ]:
```
reduced_df = df.copy()
```

# 7. Data Warehouse Construction

A Star Schema is implemented consisting of:

- Country Dimension
- Date Dimension
- Fact Table containing analytical measures.

In [ ]:
```
dim_country = reduced_df[
['location','continent','population','gdp_per_capita']
].drop_duplicates()

dim_country['country_id'] = range(len(dim_country))
```

In [ ]:
```
dim_date = reduced_df[['date']].drop_duplicates()

dim_date['year'] = dim_date['date'].dt.year
dim_date['month'] = dim_date['date'].dt.month
dim_date['quarter'] = dim_date['date'].dt.quarter
```

In [ ]:
```
fact_table = reduced_df.merge(
    dim_country[['location','country_id']],
    on='location'
)
```

In [ ]:
```
fact_table.columns
```

Out[ ]:
```
Index(['location', 'continent', 'date', 'total_cases', 'total_deaths',
       'people_vaccinated', 'population', 'gdp_per_capita', 'death_rate',
       'vaccination_rate', 'pop_category', 'country_id'],
      dtype='object')
```

# 8. OLAP Operations

Multidimensional analysis performed using:

- Roll-Up
- Drill-Down
- Slice
- Dice

In [ ]:
```
monthly_cases = fact_table.groupby(
```

```
['location', fact_table['date'].dt.to_period('M')]
)['total_cases'].sum().reset_index()
```

In [ ]:

```
continent_country = fact_table.groupby(
['continent','location']
)['total_cases'].sum()
```

In [ ]:

```
dice_data = fact_table[
(fact_table['continent'].isin(['Asia','Europe'])) &
(fact_table['date'].dt.year == 2021)
]
```

# 9. Data Cube Representation

A cube is created using Continent and Year dimensions with Total Cases as measure.

In [ ]:

```
cube = fact_table.pivot_table(
values='total_cases',
index='continent',
columns=fact_table['date'].dt.year,
aggfunc='sum'
)

cube
```

Out[ ]:

| date<br>continent | 2020 | 2021 | 2022 | 2023 | 202 |
|---|---|---|---|---|---|
| **Africa** | 2.850575e+08 | 2.222352e+09 | 4.480156e+09 | 4.777568e+09 | 2.851753e+ |
| **Asia** | 2.032098e+09 | 1.945136e+10 | 5.678968e+10 | 1.084923e+11 | 6.540192e+ |
| **Europe** | 1.625730e+09 | 1.815965e+10 | 7.157639e+10 | 9.061265e+10 | 5.478227e+ |
| **North America** | 1.952655e+09 | 1.545812e+10 | 3.751653e+10 | 4.513684e+10 | 2.700953e+ |
| **Oceania** | 6.973459e+06 | 5.865609e+07 | 3.132067e+09 | 5.173457e+09 | 3.226900e+ |
| **South America** | 1.456051e+09 | 1.077255e+10 | 2.139144e+10 | 2.494248e+10 | 1.492205e+ |

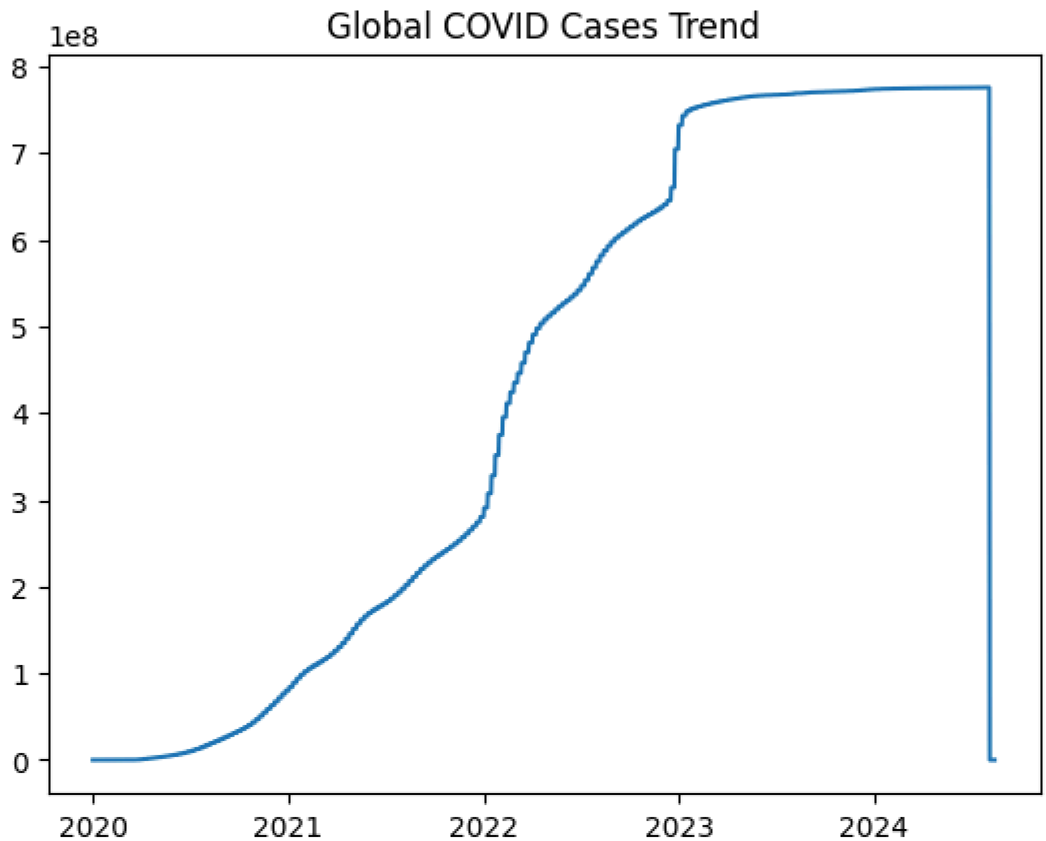
# 10. Global Pandemic Trend Analysis

This visualization shows how COVID cases evolved over time globally.

### Insight

Multiple peaks indicate pandemic waves and policy response cycles.

In [ ]:
```
global_cases = fact_table.groupby('date')['total_cases'].sum()

plt.figure()
plt.plot(global_cases)
plt.title("Global COVID Cases Trend")
plt.show()
```




## 11. Vaccination Effectiveness Analysis

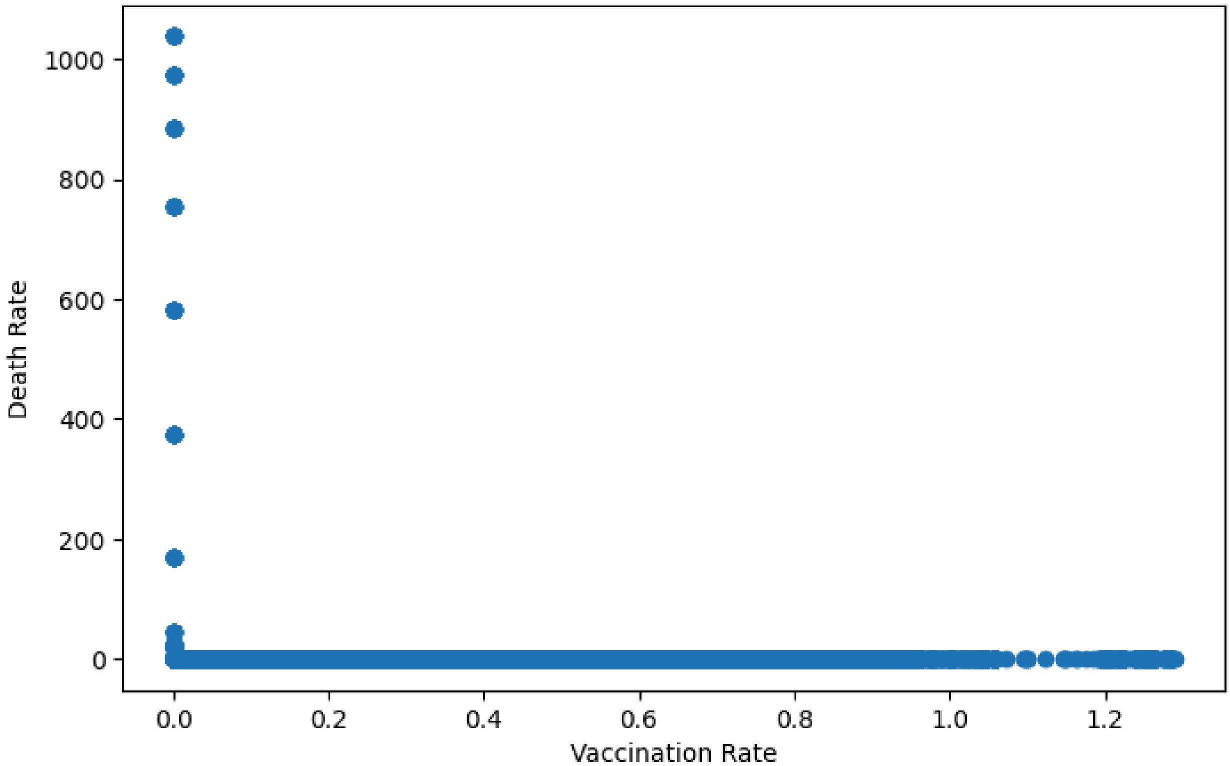
Examines relationship between vaccination coverage and mortality.

### Observation

Higher vaccination rates generally correspond to lower death rates.

In [96]:
```
plt.figure(figsize=(8,5))

plt.scatter(
fact_table['vaccination_rate'],
```

```
fact_table['death_rate']
)

plt.xlabel("Vaccination Rate")
plt.ylabel("Death Rate")
plt.show()
```




# 12. K-Means Clustering (Partitioning Method)

Countries are grouped based on pandemic similarity using:

- total cases
- total deaths
- vaccination rate

In [ ]:
```
cluster_data = fact_table.groupby('location')[[
'total_cases',
'total_deaths',
'vaccination_rate'
]].mean()
```

In [ ]:
```
scaler = StandardScaler()
X = scaler.fit_transform(cluster_data)
```

In [ ]:
```
kmeans = KMeans(n_clusters=3, random_state=0)
cluster_data['cluster'] = kmeans.fit_predict(X)
```

In [ ]:

```
cluster_sample = cluster_data.sort_values(
'total_cases',
ascending=False
).head(30)
```

In [ ]:

```
from scipy.cluster.hierarchy import linkage, dendrogram

Z = linkage(
cluster_sample[['total_cases','total_deaths','vaccination_rate']],
method='ward'
)
```

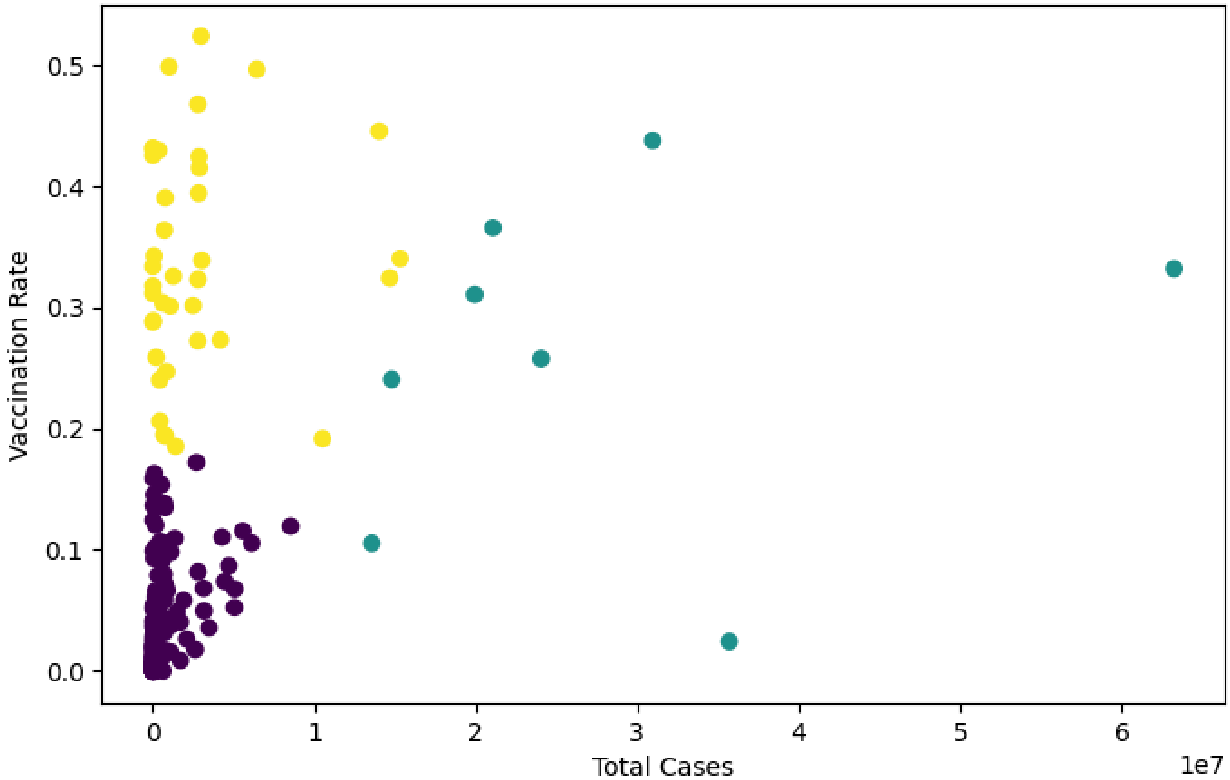
## Cluster Visualization

Each point represents a country. Colors indicate cluster membership.

Clusters reveal groups of countries with similar pandemic impact levels.

In [97]:

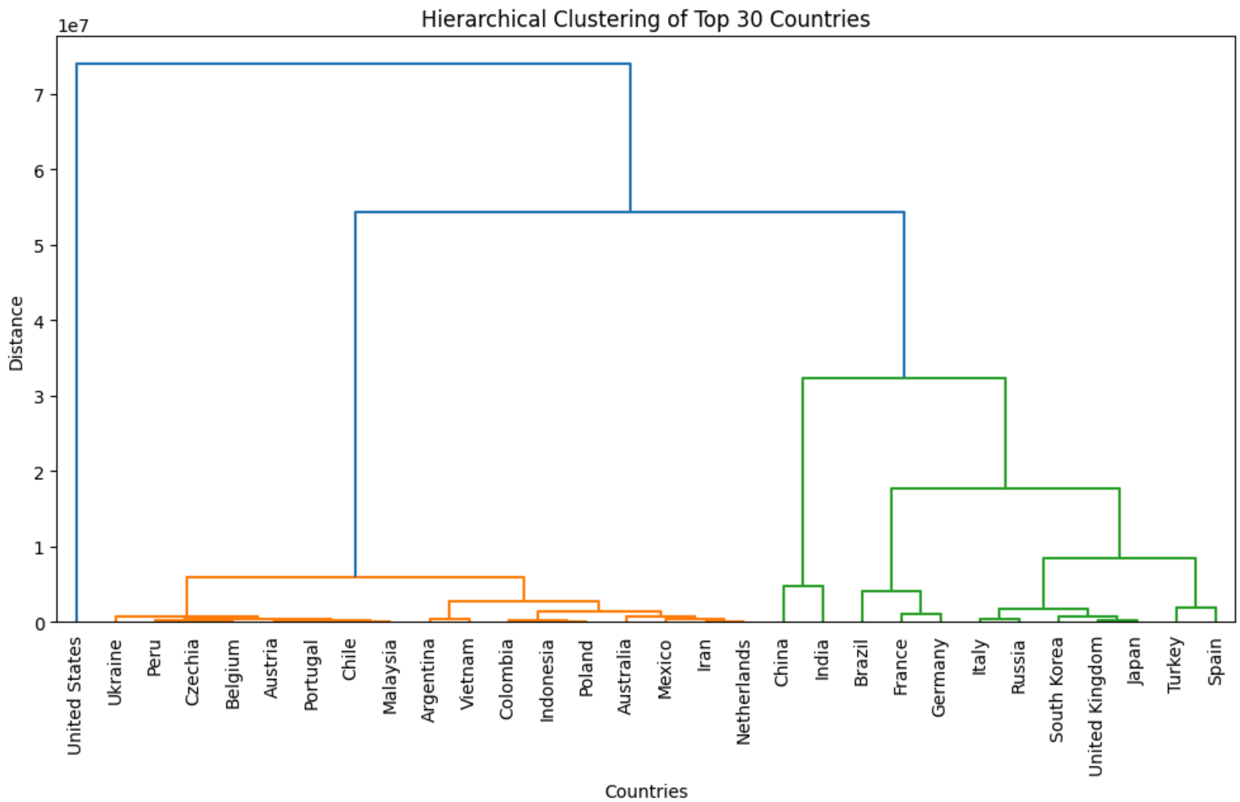
```
plt.figure(figsize=(8,5))

plt.scatter(
cluster_data['total_cases'],
cluster_data['vaccination_rate'],
c=cluster_data['cluster']
)

plt.xlabel("Total Cases")
plt.ylabel("Vaccination Rate")
plt.show()
```




# 13. Hierarchical Clustering

A dendrogram visualizes similarity relationships among countries without predefined cluster numbers.

In [ ]:
```
plt.figure(figsize=(12,6))

dendrogram(
Z,
labels=cluster_sample.index,
leaf_rotation=90
)

plt.title("Hierarchical Clustering of Top 30 Countries")
plt.xlabel("Countries")
plt.ylabel("Distance")

plt.show()
```




# 14. Outlier Detection

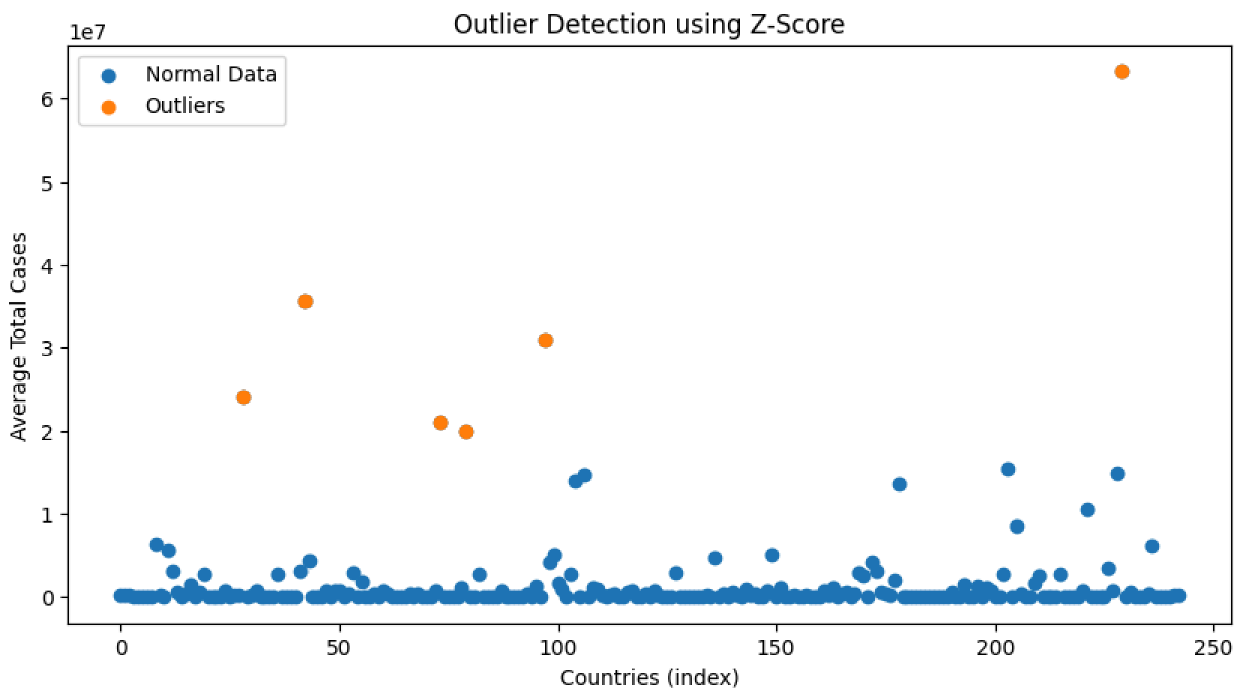
Z-score analysis identifies countries with abnormal pandemic patterns.

In [ ]:
```
cluster_data['zscore_cases'] = zscore(cluster_data['total_cases'])

outliers = cluster_data[
abs(cluster_data['zscore_cases']) > 3
]

outliers
```

Out[ ]:

| location | total_cases | total_deaths | vaccination_rate | cluster | zscore_cases |
|---|---|---|---|---|---|
| **Brazil** | 2.405882e+07 | 505097.671446 | 0.257536 | 1 | 3.759158 |
| **China** | 3.571664e+07 | 47358.838710 | 0.024245 | 1 | 5.724301 |
| **France** | 2.107340e+07 | 117994.496416 | 0.365552 | 1 | 3.255909 |
| **Germany** | 1.994243e+07 | 113734.657706 | 0.310501 | 1 | 3.065263 |
| **India** | 3.096283e+07 | 378511.336504 | 0.437517 | 1 | 4.922958 |
| **United States** | 6.327030e+07 | 777909.996416 | 0.331806 | 1 | 10.368985 |

In [ ]:
```
plt.figure(figsize=(10,5))

plt.scatter(
range(len(cluster_data)),
cluster_data['total_cases'],
label="Normal Data"
)

outliers = cluster_data[abs(cluster_data['zscore_cases']) > 3]

plt.scatter(
outliers.index.map(lambda x: cluster_data.index.get_loc(x)),
outliers['total_cases'],
label="Outliers"
)

plt.title("Outlier Detection using Z-Score")
plt.xlabel("Countries (index)")
plt.ylabel("Average Total Cases")

plt.legend()
plt.show()
```




# Continent-Level Pandemic Impact Analysis (OLAP Roll-Up Visualization)

This visualization presents the total number of COVID-19 cases aggregated at the continent level.
The aggregation represents an **OLAP Roll-Up operation**, where detailed country-

level data is summarized into a higher-level geographical dimension.

## Purpose

- To compare pandemic impact across continents.
- To identify regions contributing most to global case counts.
- To demonstrate multidimensional aggregation using data warehousing concepts.

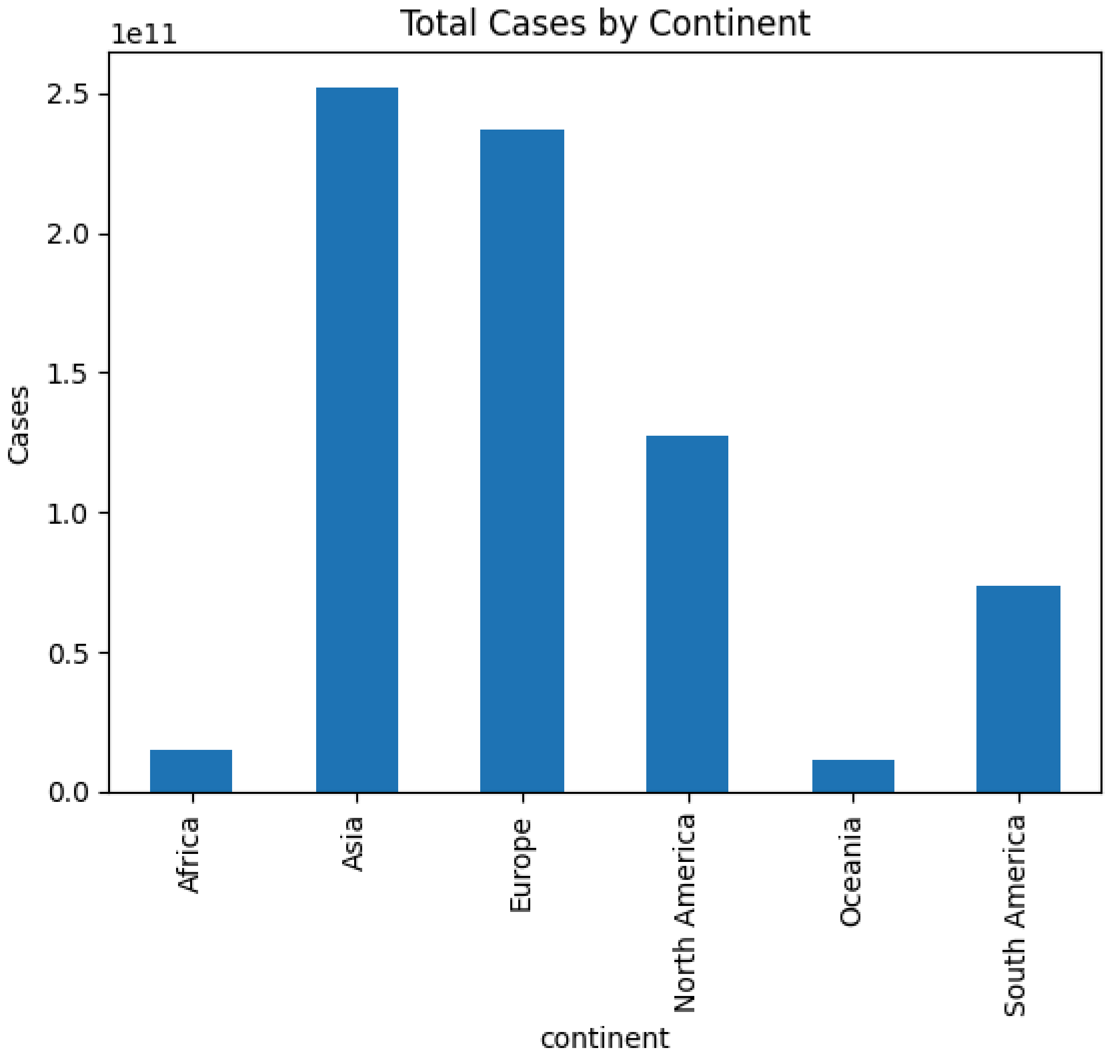
## Analytical Observations

By aggregating cases by continent, large-scale regional trends become visible that are not easily observable at the country level. Differences in total cases may reflect variations in population size, mobility patterns, healthcare infrastructure, and policy responses.

## Insight

This visualization supports Business Intelligence by enabling decision-makers to quickly assess regional pandemic severity and prioritize resource allocation at a macro level.

In [ ]:
```
continent_cases = fact_table.groupby('continent')['total_cases'].sum()

plt.figure()
continent_cases.plot(kind='bar')
plt.title("Total Cases by Continent")
plt.ylabel("Cases")
plt.show()
```




## Top 10 Countries by Total COVID-19 Cases (Business Intelligence Visualization)

This visualization identifies the ten countries with the highest recorded total COVID-19 cases.
The data is aggregated at the country level by selecting the maximum reported case count for each location, representing the peak pandemic impact experienced by each country.

### Purpose

- To highlight the most affected countries globally.
- To support comparative analysis between nations.
- To provide a clear Business Intelligence view of pandemic severity.

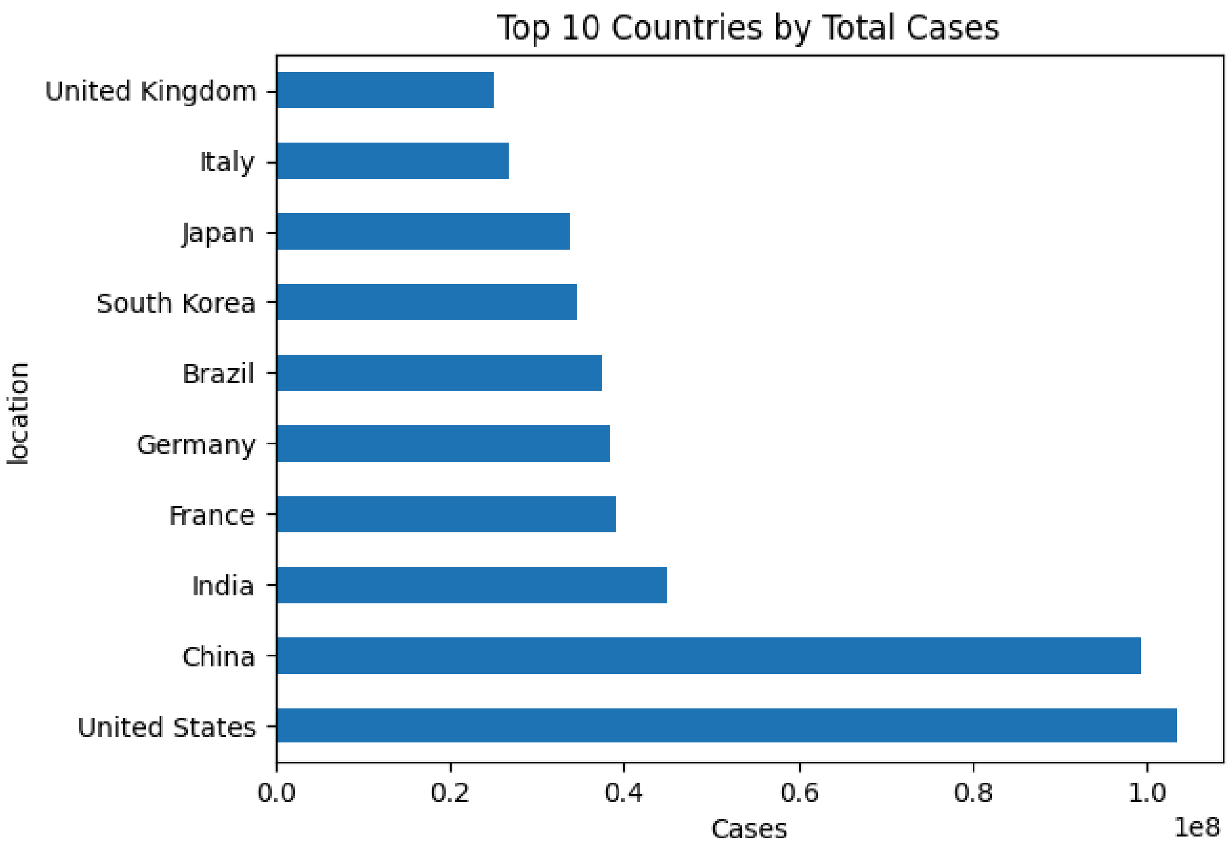
## Analytical Observations

The horizontal bar chart enables easy comparison of countries with extreme case counts. Countries appearing in this ranking typically have large populations, high urban density, or experienced multiple infection waves.

## Insight

This visualization helps decision-makers quickly identify regions requiring significant healthcare resources and policy intervention. It also demonstrates how aggregation transforms raw data into actionable insights for strategic planning.

In [ ]:
```
top10 = fact_table.groupby('location')['total_cases'].max().nlargest(10)

plt.figure()
top10.plot(kind='barh')
plt.title("Top 10 Countries by Total Cases")
plt.xlabel("Cases")
plt.show()
```



# Multidimensional Data Cube Visualization (Continent vs Year)

This visualization represents a **Data Cube** constructed using multidimensional aggregation.
The cube summarizes total COVID-19 cases across two analytical dimensions:

- **Geographical Dimension:** Continent
- **Temporal Dimension:** Year
- **Measure:** Total COVID-19 Cases

The pivot table performs aggregation similar to OLAP cube construction, and the heatmap provides a visual representation of intensity across dimensions.

## Purpose

- To demonstrate multidimensional data representation.
- To analyze how pandemic impact varies across regions and time.
- To support OLAP-based analytical exploration.

## Analytical Observations

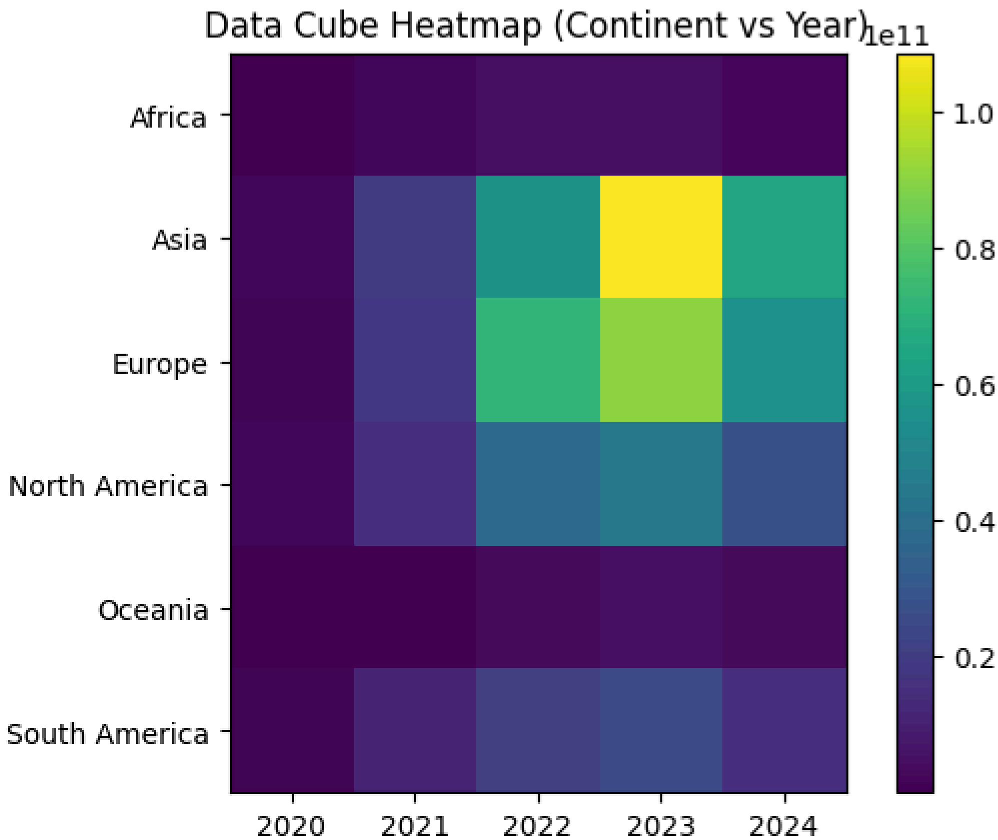
Color intensity reflects the magnitude of total cases. Darker regions indicate periods and continents experiencing higher infection levels. The visualization makes temporal trends and regional disparities immediately visible without inspecting numerical tables.

## Insight

The data cube enables decision-makers to compare pandemic progression across continents over multiple years simultaneously. This illustrates how data warehousing techniques transform large datasets into structured analytical views suitable for Business Intelligence and strategic analysis.

In [ ]:
```
cube = fact_table.pivot_table(
values='total_cases',
index='continent',
columns=fact_table['date'].dt.year,
aggfunc='sum'
)

plt.figure()
plt.imshow(cube)
plt.colorbar()
```

```
plt.xticks(range(len(cube.columns)), cube.columns)
plt.yticks(range(len(cube.index)), cube.index)
plt.title("Data Cube Heatmap (Continent vs Year)")
plt.show()
```




# Vaccination Rate Distribution Analysis

This histogram visualizes the distribution of vaccination rates across countries in the dataset.
The purpose of this analysis is to understand how vaccination coverage is spread globally and to identify common patterns, concentration ranges, and variability among nations.

## Purpose

- To examine the statistical distribution of vaccination rates.
- To understand data characteristics before applying mining techniques.
- To identify whether vaccination adoption is uniform or uneven across countries.

## Analytical Observations

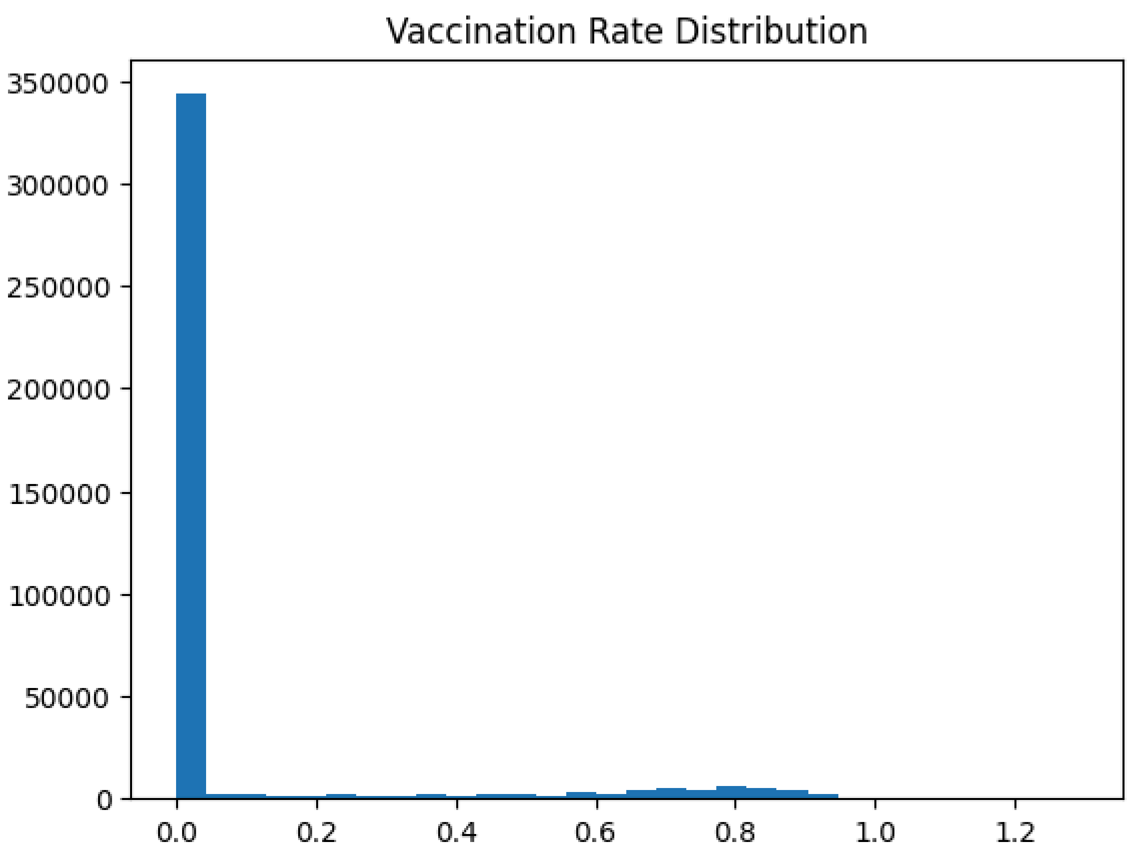
The histogram shows how frequently different vaccination rate ranges occur. Peaks in specific intervals indicate where most countries fall in terms of vaccination coverage, while sparse regions represent countries with unusually low or high vaccination levels.

## Insight

The distribution reveals global inequality in vaccination rollout, where many countries cluster around moderate vaccination levels while fewer achieve extremely high coverage. Understanding this distribution helps interpret clustering results and supports further analysis of vaccination effectiveness.

In [ ]:

```
plt.figure()
plt.hist(fact_table['vaccination_rate'], bins=30)
plt.title("Vaccination Rate Distribution")
plt.show()
```



## 16. Knowledge Extraction and Conclusion

Key findings:

- Vaccination significantly reduces mortality risk.
- Countries form natural clusters based on pandemic severity.
- Outlier countries show abnormal spread characteristics.
- OLAP and data cubes enable multidimensional analysis.

# The project demonstrates how data warehousing combined with data mining converts raw data into actionable intelligence.