

# **Expansión Estratégica de Biogenesys con Python**

**Nombre del autor: Zambón Enzo Agustín**

**Email: eazambon97@gmail.com**

**Cohorte: FT08**

**Fecha de entrega: 18/10/2024**

**Módulo 4**

**Institución:**

BIOGENESYS es una empresa farmacéutica líder en el desarrollo, fabricación y distribución de productos de salud innovadores, con un enfoque en mejorar la calidad de vida de las personas a través de soluciones médicas avanzadas. Comprometida con la investigación y la innovación, BIOGENESYS se especializa en la creación de tratamientos y terapias para una variedad de enfermedades, con un énfasis particular en enfermedades infecciosas, inmunológicas y crónicas.

Con un fuerte enfoque en la sostenibilidad y el bienestar global, BIOGENESYS trabaja de manera continua para expandir su presencia en mercados clave, especialmente en regiones donde la necesidad de acceso a la salud y la mejora de infraestructuras sanitarias es urgente. La compañía también se dedica a la lucha contra pandemias globales, como lo demuestra su enfoque en el desarrollo y distribución de vacunas durante la crisis del COVID-19.

BIOGENESYS se distingue por su capacidad para utilizar tecnologías de vanguardia y análisis de datos en la toma de decisiones estratégicas, buscando siempre optimizar sus operaciones y adaptarse a las necesidades cambiantes del mercado global.



## **Introducción**

Este estudio está orientado a identificar las ubicaciones óptimas para la expansión de laboratorios y centros de vacunación, apoyando decisiones estratégicas basadas en análisis de datos.

### **Objetivos del Proyecto**

**1. Análisis Exploratorio de Datos**

Identificar patrones y tendencias relacionadas con la incidencia de COVID-19 en la región, mediante el uso de estadísticas y visualizaciones interactivas.

**2. Limpieza y Calidad de Datos**

Aplicar procesos de depuración para garantizar la integridad de la información, facilitando decisiones fundamentadas.

**3. Optimización de Procesos ETL**

Implementar operaciones eficientes de extracción, transformación y carga, asegurando un flujo ágil y preciso de datos.

**4. Desarrollo de Dashboards Interactivos**

Diseñar herramientas visuales dinámicas que permitan explorar los datos desde perspectivas clave, apoyando el análisis en tiempo real.

### **Conclusiones del Análisis:**

Se decidió priorizar la inversión en Brasil, México y Colombia, considerando las siguientes razones estratégicas: Población y Tamaño del Mercado, Temperatura y Condiciones Climáticas, Esfuerzos de Vacunación y Cobertura y, Crecimiento Económico y Potencial de Mercado

## Desarrollo del proyecto:

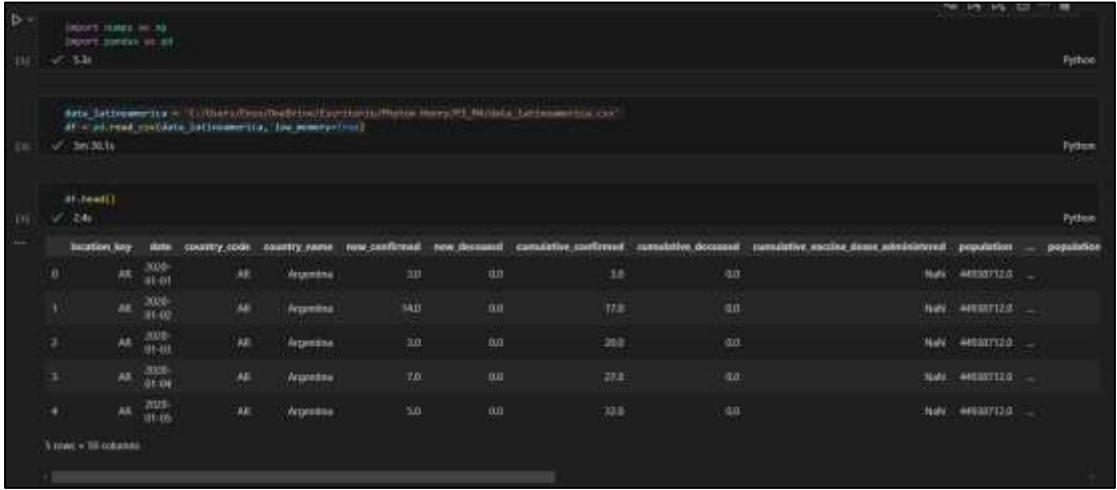
### Avance 1: Carga y transformación de los datos

El objetivo de este avance fue conectar y preparar la base de datos en Visual Studio Code para garantizar datos limpios y listos para análisis.

El primer paso consistió en importar las librerías a utilizar, en este caso numpy (as np) y pandas (as pd). Posteriormente se procedió a importar el Dataset, que se encontraba en formato csv, en una primer variable llamada 'df', a través de la función `pd.read_csv`. En el mismo código se empleó la sentencia 'Low\_memory' para facilitar al sistema operativo la lectura del documento.

Luego, se utilizó el método `.head()`, para obtener una visión rápida y accesible de sus primeros elementos, facilitando la exploración y el análisis de los datos. Se puede observar a simple vista que en nuestro Dataset existen 50 columnas, de cuales algunas cuentan con filas vacías o 'nulas' (NaN), las que resolver su situación mediante la limpieza de datos.

*Ilustración 1: Carga de datos, librerías y primera panorámica del Dataset con .head()*



```
import numpy as np
import pandas as pd

data_latinoamerica = 'C:/Users/Procto/Desktop/ProctoHenry/01_Módulo_4/latinoamerica.csv'
df = pd.read_csv(data_latinoamerica, low_memory=True)

df.head()
```

	location_key	date	country_code	country_name	new_confirmed	new_deceased	cumulative_confirmed	cumulative_deceased	cumulative_vaccine_doses_administered	population	population
0	AR	2020-01-01	AR	Argentina	3.0	0.0	3.0	0.0	NaN	44838712.0	-
1	AR	2020-01-02	AR	Argentina	94.0	0.0	77.0	0.0	NaN	44838712.0	-
2	AR	2020-01-03	AR	Argentina	3.0	0.0	30.0	0.0	NaN	44838712.0	-
3	AR	2020-01-04	AR	Argentina	7.0	0.0	27.0	0.0	NaN	44838712.0	-
4	AR	2020-01-05	AR	Argentina	5.0	0.0	20.0	0.0	NaN	44838712.0	-

5 rows × 12 columns

También se aplicó la sentencia `.dtypes` para conocer los tipos de datos por columna con los que se va a trabajar. La mayoría, son datos del tipo numérico con decimales (float64), aunque también se encuentran datos de tipo texto (object): `location_key`, `country_code` y `country_name`; y `date`, que es del tipo fecha, de acuerdo a como se definió anteriormente.

### Ilustración 2: Conociendo los tipos de datos del Dataset con .dtypes

```

dtypes
Out:
location_key      object
date             datetime64[ns]
country_code      object
country_name      object
new_confirmed     float64
new_deceased      float64
cumulative_confirmed float64
cumulative_deceased float64
cumulative_vaccine_doses_administered float64
population        float64
population_male    float64
population_female  float64
population_rural   float64
population_urban   float64
population_density float64
human_development_index float64
population_age_00_09 float64
population_age_10_19 float64
population_age_20_29 float64
population_age_30_39 float64
population_age_40_49 float64
population_age_50_59 float64
population_age_60_69 float64
population_age_70_79 float64
population_age_80_and_older float64
...
population_mortality_rate float64
comorbidity_mortality_rate float64
new_recovered            float64
cumulative_recovered      float64
dtype: object
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```

El siguiente paso fue asegurarse de que el Dataset contenga la cantidad de registros y columnas especificadas. Para ello se utilizó la función `.shape`, que realiza un conteo de las filas y columnas existentes y asigna cada valor a una variable. La verificación se realizó a través de una igualdad entre la cantidad de filas y columnas efectivas con las esperadas, de donde se determinó que efectivamente el DataSet tenía las dimensiones especificadas: un total de 12.216.057 de filas con 50 columnas.

### Ilustración 3: Identificando el número de filas y columnas del Dataset mediante .shape

```

# Filas, columnas = df.shape
filas_esperadas = 12216057
columnas_esperadas = 50

print('Filas = (filas), columnas = (columnas)')

if (filas,columnas) == (filas_esperadas,columnas_esperadas):
    print('El Dataset tiene las dimensiones indicadas')
else: print('El Dataset no tiene las dimensiones indicadas')

Out:
Filas = 12216057, Columnas = 50
El Dataset tiene las dimensiones indicadas

```

El primer filtro de datos a aplicar fue el de los países en los cuales se van a expandir. En este sentido, lo primero fue crear una lista de los países solicitados a partir de su código de ubicación (`location_key`):

```
países = ['AR', 'CL', 'CO', 'MX', 'BR', 'PE']
```

Posteriormente, se utilizó la funcionalidad `.loc[]` para aplicar el filtro, cuya “`location_key`” (columna de identificación seleccionada) se encontrara dentro de la lista de países previamente creada. Esta verificación se realizó mediante la sentencia `.isin(países)`.

## Módulo 4

El filtro aplicado redujo la cantidad de filas a 5.946, o sea aproximadamente 12.000.000 filas menos.

*Ilustración 4: Filtro de datos aplicado por países con .isin*

```

países = ['AR', 'CL', 'CO', 'CU', 'PE'] # creamos la variable países que sirve como referencia [location_key]
df_filtrado_country = df[df['location_key'].isin(países)] # creamos la variable df_filtrado_country que sirve como referencia [location_key]
df_filtrado_country

```

	location_key	date	country_code	country_name	new_confirmed	new_deceased	cumulative_confirmed	cumulative_deceased	cumulative_vaccine_doses_administered	population
0	AR	2020-01-01	AR	Argentina	3.0	0.0	3.0	0.0	NaN	44000712.0
1	AR	2020-01-02	AR	Argentina	14.0	0.0	17.0	0.0	NaN	44000712.0
2	AR	2020-01-03	AR	Argentina	3.0	0.0	20.0	0.0	NaN	44000712.0
3	AR	2020-01-04	AR	Argentina	7.0	0.0	27.0	0.0	NaN	44000712.0
4	AR	2020-01-05	AR	Argentina	3.0	0.0	30.0	0.0	NaN	44000712.0
...	...	...	...	...	...	...	...	...	...	...
10253872	PE	2022-09-13	PE	Peru	1150.0	22.0	4127612.0	216173.0	NaN	29301884.0
10253873	PE	2022-09-14	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253874	PE	2022-09-15	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253875	PE	2022-09-16	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253876	PE	2022-09-17	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0

5946 rows x 10 columns

Además, se verifico que las únicas “location\_key” existentes sean las especificadas:

*Ilustración 5: Verificación de filtro: 'location\_key'*

```

países_unicos = df_filtrado_country['location_key'].unique()
print(países_unicos)

```

['AR' 'CO' 'CL' 'CU' 'PE']

El segundo filtro aplicado es un filtro de fechas: se requirieron aquellas mayores al 01/01/2021. Este filtro redujo la cantidad de filas a 3.744.

*Ilustración 6: Filtro de datos por fechas mayores a 01-01-2021*

```

df_filtrado_fecha = df_filtrado_country[df_filtrado_country['date'] > '2021-01-01'] # creamos una variable para filtrar por fecha mayor a 2021-01-01
df_filtrado_fecha

```

	location_key	date	country_code	country_name	new_confirmed	new_deceased	cumulative_confirmed	cumulative_deceased	cumulative_vaccine_doses_administered	population
367	AR	2021-01-02	AR	Argentina	7767.0	196.0	1679634.0	46880.0	20334.0	44000712.0
368	AR	2021-01-03	AR	Argentina	4004.0	157.0	167568.0	46840.0	20529.0	44000712.0
369	AR	2021-01-04	AR	Argentina	13953.0	753.0	1689521.0	47000.0	25764.0	44000712.0
370	AR	2021-01-05	AR	Argentina	14035.0	160.0	170406.0	47160.0	29853.0	44000712.0
371	AR	2021-01-06	AR	Argentina	14496.0	131.0	1718102.0	47293.0	34762.0	44000712.0
...	...	...	...	...	...	...	...	...	...	...
10253872	PE	2022-09-13	PE	Peru	1150.0	22.0	4127612.0	216173.0	NaN	29301884.0
10253873	PE	2022-09-14	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253874	PE	2022-09-15	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253875	PE	2022-09-16	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0
10253876	PE	2022-09-17	PE	Peru	NaN	NaN	NaN	NaN	NaN	29301884.0

3744 rows x 10 columns

## Módulo 4

Una vez aplicados los filtros, se procedió a la limpieza de nuestra base. Primero se identificó la cantidad total de registros nulos por columnas, mediante la sentencia compuesta o integrada: `isnull().sum()`. Aquí se observa que, solo en apenas siete columnas existen datos nulos, donde en la mayoría no superan el 50% del total de elementos existentes.

*Ilustración 7: Total de elementos nulos por columna con `isnull().sum()`*

```

df_filtredo_fecha.isnull().sum()

```

Columna	Count
location_key	0
date	0
country_code	0
country_name	0
new_confirmed	21
new_deceased	21
cumulative_confirmed	21
cumulative_deceased	21
cumulative_vaccine_doses_administered	584
population	0
population_male	0
population_female	0
population_rural	0
population_urban	0
population_fertility	0
human_development_index	0
population_age_00_09	0
population_age_10_19	0
population_age_20_29	0
population_age_30_39	0
population_age_40_49	0
population_age_50_59	0
population_age_60_69	0
population_age_70_79	0
population_age_80_and_older	0
population_mortality_rate	0
contaminability_mortality_rate	0
new_recovered	2116
cumulative_recovered	2736
dtypes	int64

El siguiente paso consistió en rellenar los valores faltantes con valores representativos. Para ello, se utilizó el promedio de cada columna diferenciado por país.

*Ilustración 8: Reemplazo de valores nulos por valores medios*

```

df_limpio = df_filtredo_fecha.copy()
columnas_numericas = df_limpio.select_dtypes(include=["float64", "int64"]).columns

for column in columnas_numericas:
    df_limpio[column] = df_limpio.groupby("country_name")[column].transform(lambda x: x.fillna(x.mean()))

df_limpio

```

location_key	date	country_code	country_name	new_confirmed	new_deceased	cumulative_confirmed	cumulative_deceased	cumulative_vaccine_doses_administered	population
357	2021-01-02	AR	Argentina	770700000	16600000	1.470634e+06	4028100000	2.034400e+08	44938712.0
360	2021-01-03	AR	Argentina	403400000	15700000	1.475568e+06	4044500000	2.057900e+08	44938712.0
369	2021-01-04	AR	Argentina	1390300000	15700000	1.4889521e+06	4700200000	2.516400e+08	44938712.0
370	2021-01-06	AR	Argentina	1420500000	16000000	1.705600e+06	4710200000	2.980200e+08	44938712.0
371	2021-01-06	AR	Argentina	1440600000	13100000	1.718102e+06	4729300000	3.416200e+08	44938712.0
10251872	2022-09-12	PE	Peru	115000000	22000000	4.170612e+06	25617300000	4.007186e+07	29381884.0
10251873	2022-09-14	PE	Peru	5032183871	187233871	2.576900e+06	18824386129	4.007186e+07	29381884.0
10251874	2022-09-15	PE	Peru	5032183871	197233871	2.576900e+06	18824386129	4.007186e+07	29381884.0
10251875	2022-09-16	PE	Peru	5032183871	197233871	2.576900e+06	18824386129	4.007186e+07	29381884.0
10251876	2022-09-17	PE	Peru	5032183871	187233871	2.576900e+06	18824386129	4.007186e+07	29381884.0

## Módulo 4

De esta manera, como se puede observar a continuación, solo quedan dos columnas con valores nulos: new\_recovered y cumulative\_recovered, las cuales fueron eliminadas con la sentencia `df_limpiar = df_filldate.drop(columns=['new_recovered', 'cumulative_recovered'])`

*Ilustración 9: Total de elementos nulos por columnas con .isnull()*

```

df_limpiar.isnull().sum()
new_recovered      1872
cumulative_recovered 2406
dtype: int64

```

A continuación, se creó el nuevo dataframe solicitado en formato csv: `DatosFinalesFiltrado = df_limpiar.to_csv('DatosFinalesFiltrado', index=False)`

*Ilustración 10: Creación de DataFrame “DatosFinalesFiltrado”*

```

DatosFinalesFiltrado = df_limpiar.to_csv('DatosFinalesFiltrado', index=False)

```

Posteriormente, se aplicaron bucles for y/o while para el cálculo de estadísticas descriptivas y otras métricas importantes que ofrece pandas por default:



*Ilustración 11: Utilización bucle "for" para obtener estadísticas descriptivas*

```
for i in df_limpio.columns:
    print(i)
    print(df_limpio[i].describe())
    print('-----')
```

location\_key  
count 3744  
unique 6  
top 88  
freq 624  
Name: location\_key, dtype: object

date  
count 3744  
unique 624  
top 2023-01-01 00:00:00  
freq 1  
first 2023-01-01 00:00:00  
last 2023-01-17 00:00:00  
Name: date, dtype: object

country\_code  
count 3744  
unique 6  
top 88  
freq 624  
Name: country\_code, dtype: object

country\_name  
count 3744  
unique 6  
top Argentina  
freq 624  
Name: country\_name, dtype: object

*Ilustración 12: Utilización de bucle "while" para la obtención de estadísticas descriptivas*

```
i = 0
while i < len(df_limpio.columns):
    columna = df_limpio.columns[i]
    print(columna)
    print(df_limpio[columna].describe())
    print('-----')
    i += 1
```

location\_key  
count 3744  
unique 6  
top 88  
freq 624  
Name: location\_key, dtype: object

date  
count 3744  
unique 624  
top 2023-01-01 00:00:00  
freq 1  
first 2023-01-01 00:00:00  
last 2023-01-17 00:00:00  
Name: date, dtype: object

country\_code  
count 3744  
unique 6  
top 88  
freq 624  
Name: country\_code, dtype: object

country\_name  
count 3744  
unique 6  
top Argentina  
freq 624  
Name: country\_name, dtype: object

También, se aplicó la función `.describe()`, para obtener los estadísticos generales de todo el conjunto:

Ilustración 13: Obtención de estadísticos con .describe()

	count	sum	mean	std	min	25%	50%	75%	max	population	population std	population female	population total
count	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000	3744.000000
sum	13811.908931	27623.817862	7.377108e+06	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07
std	24219.561278	367.240038	7.000000e+06	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07
min	-575.000000	0.000000	9.710000e+02	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	1535.750000	26.000000	2.132758e+06	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07
50%	5214.000000	180.000000	3.632144e+06	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07
75%	14834.000000	152.000000	6.234974e+06	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07
max	28848.000000	11447.000000	3.454883e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07	1.629579e+07

## Preguntas:

## 1. ¿Qué implican estas métricas y cómo pueden ayudar en el análisis de datos?

Explicación de las métricas calculadas:

- **Media (mean):** Representa el promedio de los valores de la columna. Es útil para conocer el valor central y comparar el comportamiento de distintas variables.
- **Mediana (median):** Es el valor central de los datos ordenados y es menos sensible a valores extremos que la media, proporcionando un valor central más representativo en distribuciones asimétricas.
- **Desviación estándar (std):** Indica la dispersión de los datos respecto a la media. Un valor alto significa que los datos están más dispersos, mientras que un valor bajo sugiere que están más concentrados alrededor de la media.
- **Mínimo (min) y Máximo (max):** Son los valores extremos de la columna y permiten identificar posibles outliers (valores atípicos) o errores de captura de datos.

Estas métricas ayudan a comprender la distribución y dispersión de los datos en cada columna:

- Identificar la centralización de datos en columnas numéricas.
- Observar la variabilidad de los datos, lo que puede indicar si hay tendencias o inconsistencias significativas.
- Detectar valores extremos que podrían requerir tratamiento adicional, como la imputación de datos o la eliminación de outliers.

## 2. ¿Se muestran todas las estadísticas en todas las columnas durante el análisis?

**Módulo 4**

No, pandas muestra solo estadísticas numéricas para columnas que contienen datos numéricos, y otras estadísticas como frecuencia o valores únicos para columnas categóricas.

### 3. ¿Cuál es la razón de la respuesta anterior y cómo podría afectar la interpretación de los resultados obtenidos?

Las métricas como la media y la desviación estándar no tienen sentido para datos categóricos porque no describen la frecuencia o dispersión de texto o categorías. Del mismo modo, métricas categóricas (frecuencias y valores únicos) no aplican en datos numéricos. Esto afecta la interpretación al hacer necesario revisar cada tipo de dato con las métricas adecuadas; si se analizan incorrectamente, puede llevar a conclusiones equivocadas sobre la variabilidad, tendencia central y comportamiento general de los datos.

Por último, se creó una función para obtener la mediana, la varianza y el rango de cada columna:

*Ilustración 14: Mediana, Varianza y Rango por columna*

```
>>> def calcular_estadisticas(df_limpio):
    estadisticas = {}

    for columna in df_limpio.select_dtypes(include='number').columns:
        mediana = df_limpio[columna].median()
        varianza = df_limpio[columna].var()
        rango = df_limpio[columna].max() - df_limpio[columna].min()

        estadisticas[columna] = {
            'mediana': mediana,
            'varianza': varianza,
            'rango': rango
        }

    return estadisticas

calcular_estadisticas(df_limpio)

[{'mes_confirmado': {'mes': '11-05-2020/11-06-2020',
                    'mediana': 0.0,
                    'std': 274.771847586922,
                    'min': -23997.8,
                    'max': 28688.0},
 'mes_descartado': {'mes': '0-7-2014/19-06-2017',
                    'mediana': 0.0,
                    'std': 11.0926785117749,
                    'min': -4289.0,
                    'max': 11847.0},
 'cumulative_confirmado': {'mes': '12-04-2020/05-05-2021',
                           'mediana': 0.0,
                           'std': 286649.4888004083,
                           'min': 0.0,
                           'max': 34568833.0},
 'cumulative_descartado': {'mes': '21-05-2020/24-06-2021',
                           'mediana': 15.0,
                           'std': 1887.97438970171,
                           'min': 0.0,
                           'max': 681203.0},
 'cumulative_vacuna_administrada': {'mes': '11-04-2021-04-08-2023',
                                     'mediana': 15682.0,
                                     'std': 15682.0,
                                     'min': 0.0,
                                     'max': 15682.0}]
```

**Preguntas:**

#### 1. ¿Qué representa la mediana?

**Mediana:** es el valor central de un conjunto de datos ordenado y representa el "punto medio" en el cual el 50% de los datos están por debajo y el otro 50% por encima. Es una medida de tendencia central que no se ve afectada por valores extremos, por lo cual puede ofrecer una representación más robusta del centro de los datos cuando existen outliers.

**2. ¿Cómo varía la dispersión de los datos en el conjunto de datos analizado, en términos de la varianza y el rango?**

**Varianza:** mide la dispersión o variabilidad de los datos con respecto a la media. Valores altos de varianza indican que los datos están más alejados de la media, mientras que valores bajos sugieren una mayor consistencia en torno a la media.

**Rango:** mide la diferencia entre el valor máximo y el mínimo, proporcionando una visión de la amplitud general de los datos. Si el rango es amplio, hay una diferencia significativa entre los valores extremos, lo que puede indicar una alta dispersión

**3. ¿Qué nos puede indicar esto sobre la consistencia o la variabilidad de los datos en relación con la mediana?**

Comparar la varianza y el rango en relación con la mediana puede ayudar a identificar patrones y consistencias en el conjunto de datos:

- Si la varianza es baja y el rango también es reducido, los datos están muy concentrados y no hay mucha variabilidad en torno a la mediana. Esto sugiere consistencia en los valores.
- Si la varianza es alta y el rango también es amplio, significa que los datos están dispersos y hay una mayor variabilidad alrededor de la mediana. Esto indica que los valores son inconsistentes, lo cual puede ser relevante para analizar tendencias o detectar posibles outliers.
- La mediana puede ofrecer una visión más representativa del centro de los datos cuando la varianza y el rango son altos, ya que no es afectada por valores extremos.

Estos análisis permiten ajustar modelos y tomar decisiones informadas al interpretar la consistencia y variabilidad del conjunto de datos.

## Avance 2: Análisis Exploratorio – Visualización

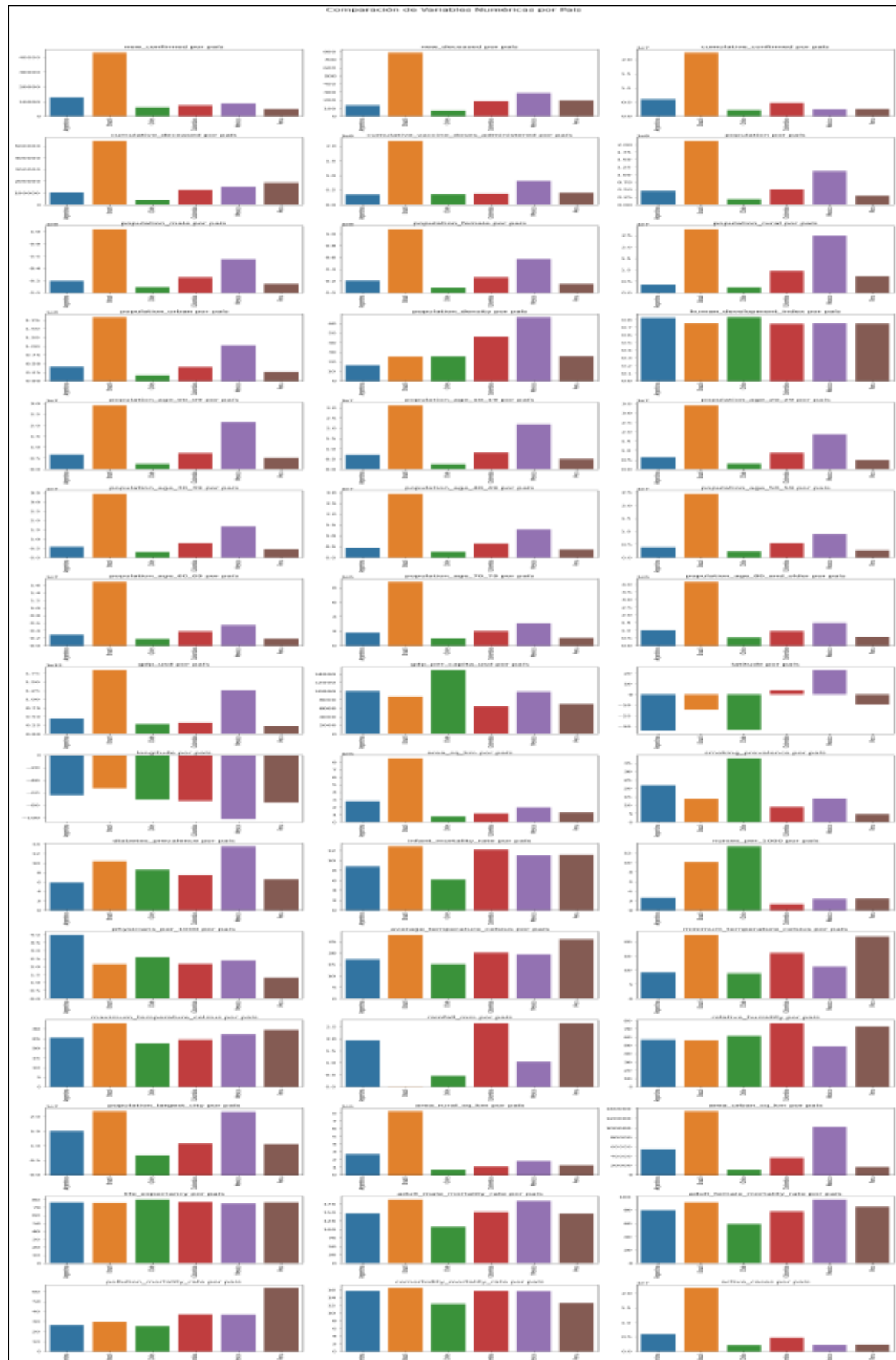
El objetivo de este análisis fue realizar visualizaciones claras y detalladas, que nos permitan descubrir patrones, tendencias y anomalías en nuestros datos para extraer insights valiosos que nos orienten en la planificación estratégica de la expansión de laboratorios farmacéuticos.

### 1. Graficos Generales:

Se comenzó por importar las librerías a utilizar en esta segunda etapa: matplotlib y seaborn, para luego, comparar variables de países a través de gráficos de barras.

*Ilustración 15: Sintaxis para la elaboración de gráficos de barra*

```
In =  
import matplotlib.pyplot as plt  
from matplotlib.ticker import FuncFormatter  
import seaborn as sns  
(Out)  
Python  
In =  
# 1. Definición de listas de países a través de listas de variables  
columnas_numericas = [col for col in df_limpio.select_dtypes(include="number").columns if df_limpio[col].isnull().sum() == 0]  
num_cols = 3  
num_rows = (len(columnas_numericas) // num_cols) + 1  
fig, axes = plt.subplots(num_rows, num_cols, figsize=(18, num_rows * 5))  
fig.suptitle('Comparación de Variables Numéricas por País', fontsize=16)  
axes = axes.flatten()  
  
orden_países = df_limpio['country_name'].unique()  
palette = sns.color_palette("m2", len(orden_países))  
color_dict = dict(zip(orden_países, palette))  
  
for i, col in enumerate(columnas_numericas):  
    df_ordenado = df_limpio.sort_values('country_name')  
    sns.barplot(data=df_ordenado, x='country_name', y=col, ax=axes[i], color=color_dict, palette="m2")  
    axes[i].set_title(f'{col} por país', fontsize=12)  
    axes[i].set_xlabel('')  
    axes[i].set_ylabel('')  
    axes[i].tick_params(axis='x', rotation=45)  
  
for j in range(1 + 1, len(axes)):  
    axes[j].set_visible(False)  
  
plt.tight_layout(rect=[0, 0.1, 0.96])  
plt.show()
```

*Ilustración 16: Comparación de variables numéricas por País*

## Módulo 4

También se creó una matriz de correlación entre todas las variables de nuestro dataframe, a partir de una máscara y con un filtro de 0.5. Una matriz de correlación es una tabla que muestra la relación entre varias variables. En otras palabras, indica el grado en que dos variables están relacionadas entre sí.

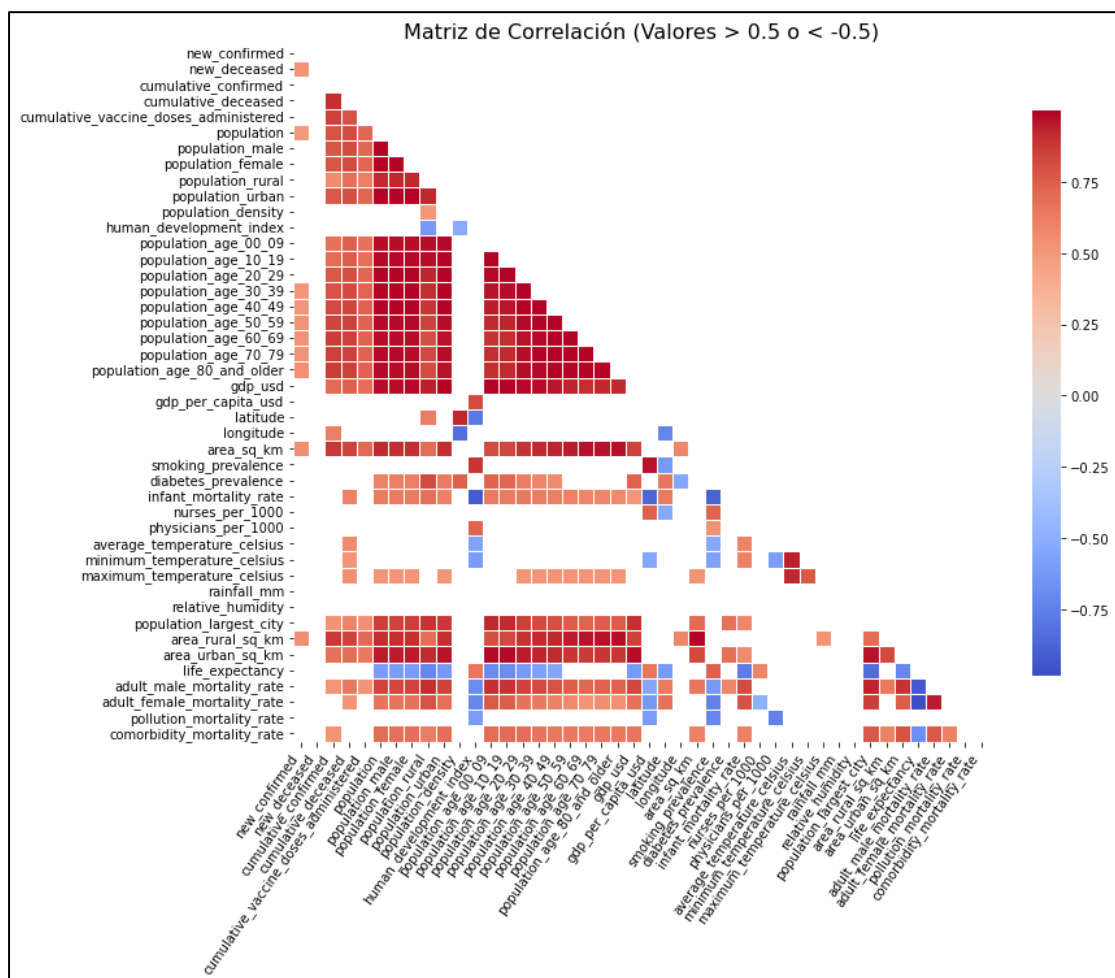
Ilustración 17: Código de creación de la Matriz de Correlación

```
# MATRIZ DE CORRELACIÓN CON MÁSCARA Y FILTRO = 0.5 ( > 0.5 )
corr_matrix = df[['new_confirmed',
                  'new_deceased',
                  'cumulative_confirmed',
                  'cumulative_deceased',
                  'cumulative_vaccine_doses_administered',
                  'population',
                  'population_male',
                  'population_female',
                  'population_rural',
                  'population_urban',
                  'population_density',
                  'human_development_index',
                  'population_age_00_09',
                  'population_age_10_19',
                  'population_age_20_29',
                  'population_age_30_39',
                  'population_age_40_49',
                  'population_age_50_59',
                  'population_age_60_69',
                  'population_age_70_79',
                  'population_age_80_and_older',
                  'gdp_usd',
                  'gdp_per_capita_usd',
                  'latitude',
                  'longitude',
                  'area_sq_km',
                  'smoking_prevalence',
                  'diabetes_prevalence',
                  'infant_mortality_rate',
                  'nurses_per_1000',
                  'physicians_per_1000',
                  'average_temperature_celsius',
                  'minimum_temperature_celsius',
                  'maximum_temperature_celsius',
                  'rainfall_mm',
                  'relative_humidity',
                  'population_largest_city',
                  'area_rural_sq_km',
                  'area_urban_sq_km',
                  'life_expectancy',
                  'adult_male_mortality_rate',
                  'adult_female_mortality_rate',
                  'pollution_mortality_rate',
                  'comorbidity_mortality_rate']]

# Filtrado de la matriz de correlación
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))
corr_matrix = corr_matrix[mask[mask == 0.5]]

plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix, mask=mask, cmap='magma', annot=True, center=0,
            square=True, linewidth=0.5, cbar_kws={'label': 'Correlation'})
plt.title('Matriz de Correlación (Valores > 0.5 o < -0.5)')
plt.xticks(rotation=45)
plt.yticks(rotation=45)
plt.show()
```

Ilustración 18: Matriz de Correlación (Valores > 0.5 o < -0.5)



La correlación puede ser:

- Positiva: Cuando una variable aumenta, la otra también tiende a aumentar.
- Correlación negativa: Cuando una variable aumenta, la otra tiende a disminuir.

**Módulo 4**

- Correlación nula o baja: Si las variables no tienen una relación clara, la correlación estará cerca de 0 (por ejemplo, 0.1). Esto indica que no hay una relación lineal evidente entre las dos variables.

Luego, se graficaron Histogramas de aquellas variables con cambios en sus valores. Los histogramas son una herramienta gráfica utilizada para representar la distribución de un conjunto de datos.

Son útiles para visualizar cómo se distribuyen los valores de una variable, ayudando a identificar patrones, tendencias y características clave de los datos, como la forma de la distribución, su simetría o su asimetría, la presencia de valores atípicos y la concentración de datos en ciertos rangos.

En función de observar la matriz se puede observar a simple vista que no existen variables que tengan influencia significativa, ya sea positiva o negativa, sobre las variables relevantes del caso de estudio: new\_confirmed y new\_deseased, a excepción de las variables de densidad poblacionales y territoriales, como, por ejemplo, population y área\_sq\_km. Por esta razón se debe optar por elegir variables que creamos conveniente en función de algún otro tipo de análisis.

*Ilustración 19: Sintaxis para la creación de Histogramas*

```
#!/usr/bin/env python3
# Histogramas de variables que si tienen cambios de valores HISTOGRAMAS DE VARIABLES QUE SI TIENEN CAMBIO DE VALORES

columnas_numericas = [col for col in df.columns.select_dtypes(include='number'), columnas if df[col].notnull().count() > 1]

num_cols = 3
num_rows = (len(columnas_numericas) + num_cols - 1) // num_cols # Calcula las filas necesarias

fig, axes = plt.subplots(num_rows, num_cols, figsize=(18, num_rows * 4))
fig.suptitle('Histogramas de Variables Numericas', fontsize=16)

axes = axes.flatten()

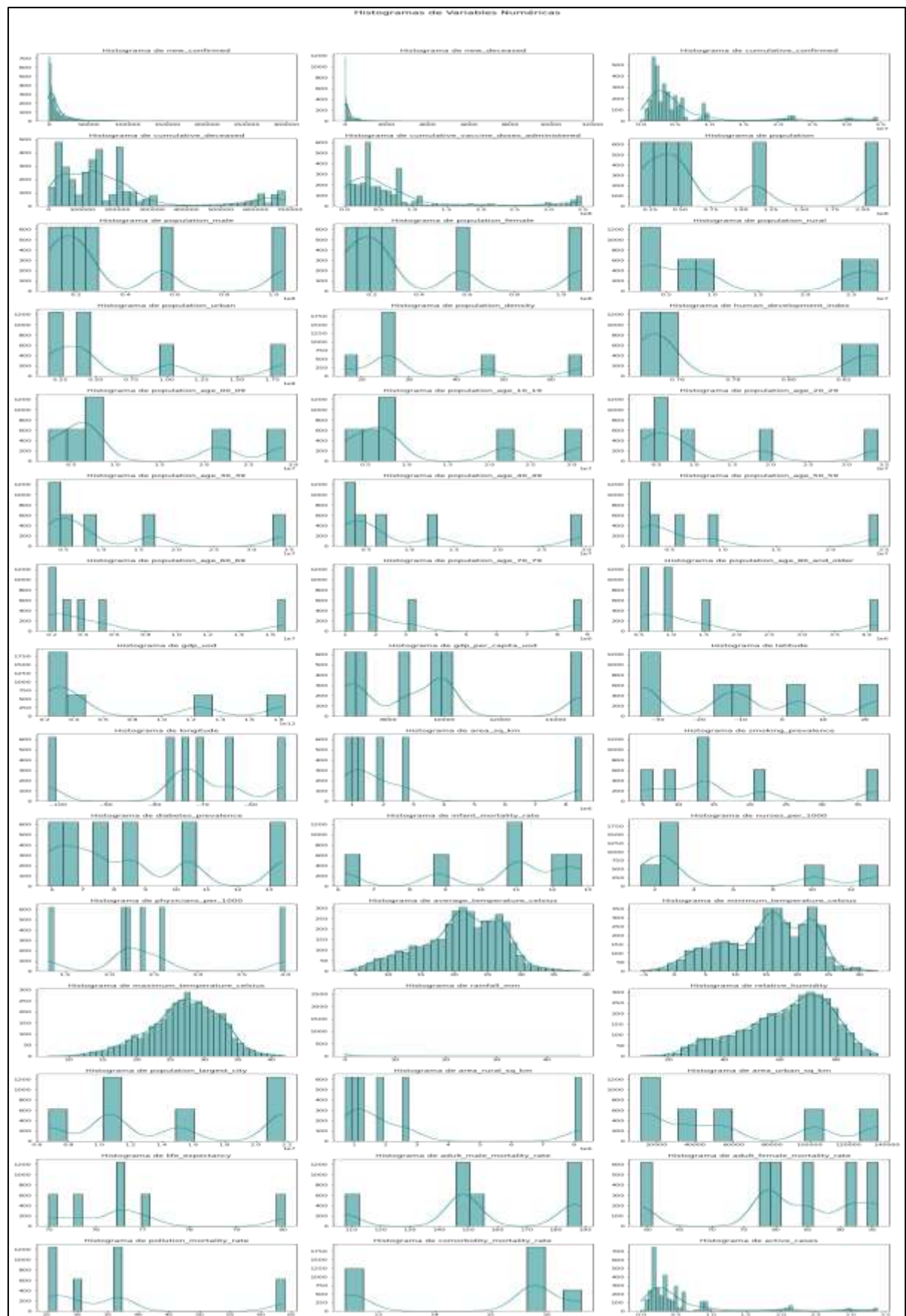
for i, col in enumerate(columnas_numericas):
    ax = axes.flat[0] # Obtén el primer eje de la fila
    axes[i].set_title(f'Histograma de {col}', fontsize=12)
    axes[i].set_xlabel(f'{col}') # Etiqueta el eje x con el nombre de la columna
    axes[i].set_ylabel('')

for j in range(1, num_rows):
    axes[j].set_visible(False)

plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()
```



## Ilustración 20: Histogramas de Variables Numéricas



**Módulo 4**

## 2. Descripción Poblacional:

A fin de tener un panorama más preciso de las características de la población de estudio de los diferentes países se realizaron distintos gráficos de barras. Para empezar, se realizó un gráfico que muestre comparativamente, la población total que tiene cada uno, donde se detecta que Brasil, claramente, es el país con mayor volumen de gente, superando los 200 millones de habitantes. Muy por detrás, se encuentra México, con 110 millones, Colombia con 50 millones y le siguen el resto de países.

*Ilustración 21: Sintaxis para la elaboración de un Gráfico Poblacional*

```
# df_ordenado = df_datos[['country_name', 'population']].sort_values(by='population', ascending=False)

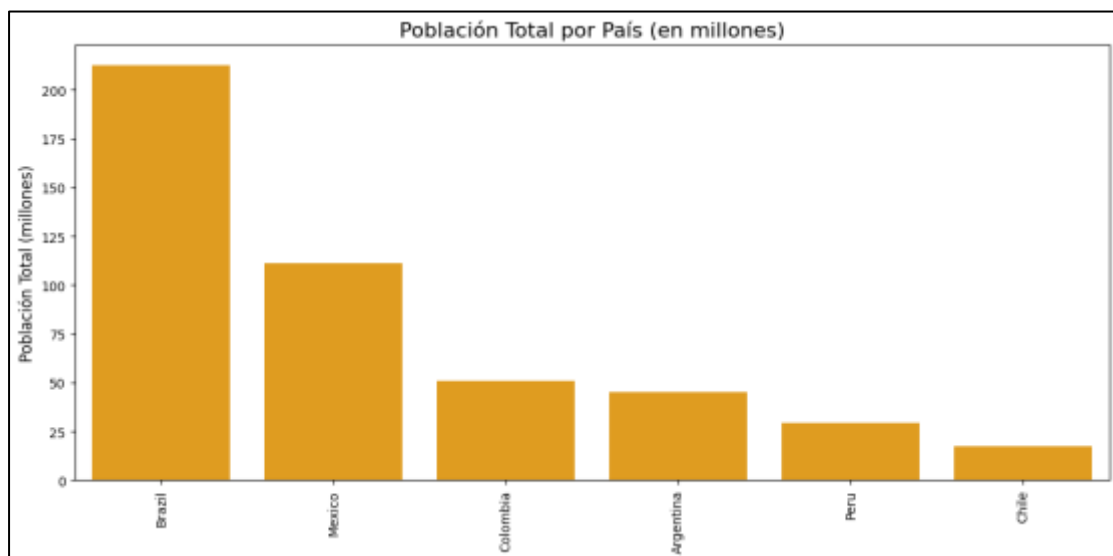
# Convertir población a millones
df_ordenado['population_millions'] = df_ordenado['population'] / 1,000,000

# Configuración del gráfico
plt.figure(figsize=(12, 6))

# Gráfico de barras con labels
plt.barplot(df_ordenado, x='country_name', y='population_millions', color='teal')

# Añadir anotaciones para mejorar la visualización
plt.title('Población Total por País (en millones)', fontsize=16)
plt.xlabel('País', fontsize=12)
plt.ylabel('Población Total (millones)', fontsize=12)
plt.xticks(rotation=90) # Rotar los nombres de los países para mejor legibilidad
plt.tight_layout()

# Mostrar el gráfico
plt.show()
```

*Ilustración 22: Población Total por País (en millones)*

También se observó el PBI per cápita en USD por país, para tener un referencia del nivel de vida de población, donde se determinó que Chile se encuentra en primer lugar, con mas de 14 mil dólares por persona, seguido por Argentina, con 10 mil, mientras que en tercer lugar se encuentra México, casi al mismo nivel de Argentina.

*Ilustración 23: Sintaxis para la elaboración de un gráfico de barras por PBI per cápita*

```
df_pbi_mil = df_pbi[['country_name', 'gdp_per_capita_mil']].reset_index().sort_values('gdp_per_capita_mil', ascending=False)

# convertir el gdp per capita a miles de $
df_pbi_mil['gdp_per_capita_mil'] = df_pbi_mil['gdp_per_capita_mil'] / 1,000

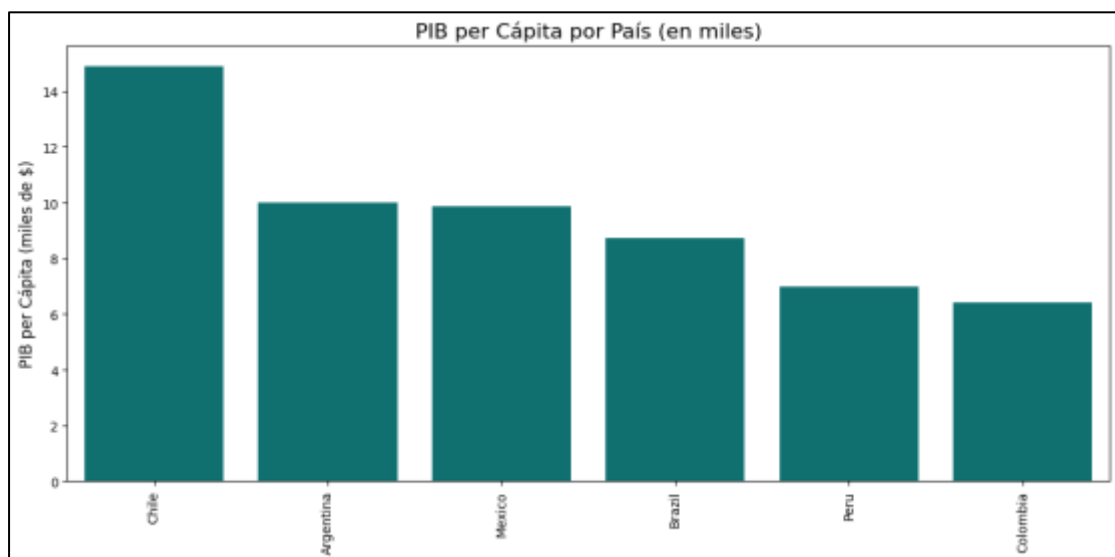
# configuración del gráfico
plt.figure(figsize=(10, 6))

# ordenar de mayor a menor
plt.xticks(rotation=90, labels=df_pbi_mil['country_name'], fontsize=10)

# mostrar el gráfico por país
plt.bar(df_pbi_mil['country_name'], df_pbi_mil['gdp_per_capita_mil'], color='teal')

# mostrar el gráfico
plt.show()
```

*Ilustración 24: PBI per Cápita por País (en Miles de USD)*



Posteriormente, se mostró la distribución de la población de diferentes países en varios grupos de edad. Cada barra representa un grupo de edad específico, y cada color en la barra corresponde a un país diferente, permitiendo comparar visualmente la distribución por grupos de edad entre países.

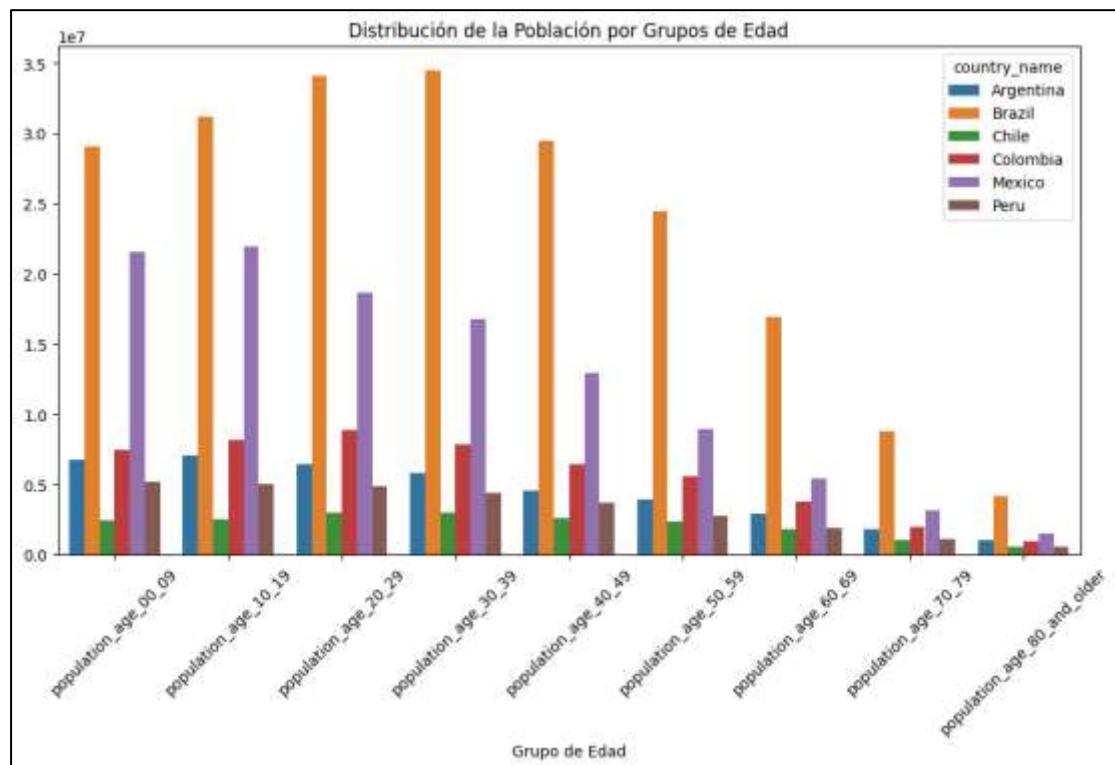
*Ilustración 25: Sintaxis para la elaboración de un Gráfico de Barras por Grupos de Edad*

```
data_cada = [col for col in df_limpio.columns if 'population_age' in col]
df_cada = df_limpio[data_cada].reset_index().sort_values('country_name', var_name='Grupos de Edad', values_name='Población')

plt.figure(figsize=(10, 6))
plt.xticks(rotation=90, labels=df_cada['country_name'], fontsize=10)
plt.title('Distribución de la Población por Grupos de Edad')
plt.bar(df_cada['country_name'], df_cada['Población'], color='teal')
plt.show()
```

## Módulo 4

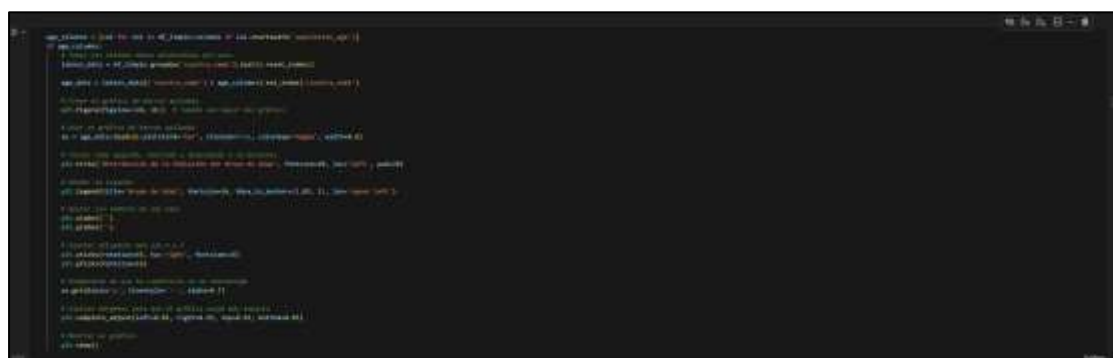
Ilustración 26: Distribución de la Población por Grupos de Edad



El siguiente gráfico, muestra un contenido similar, pero con barras apiladas que detallan la distribución de la población, desglosada por grupos de edad.

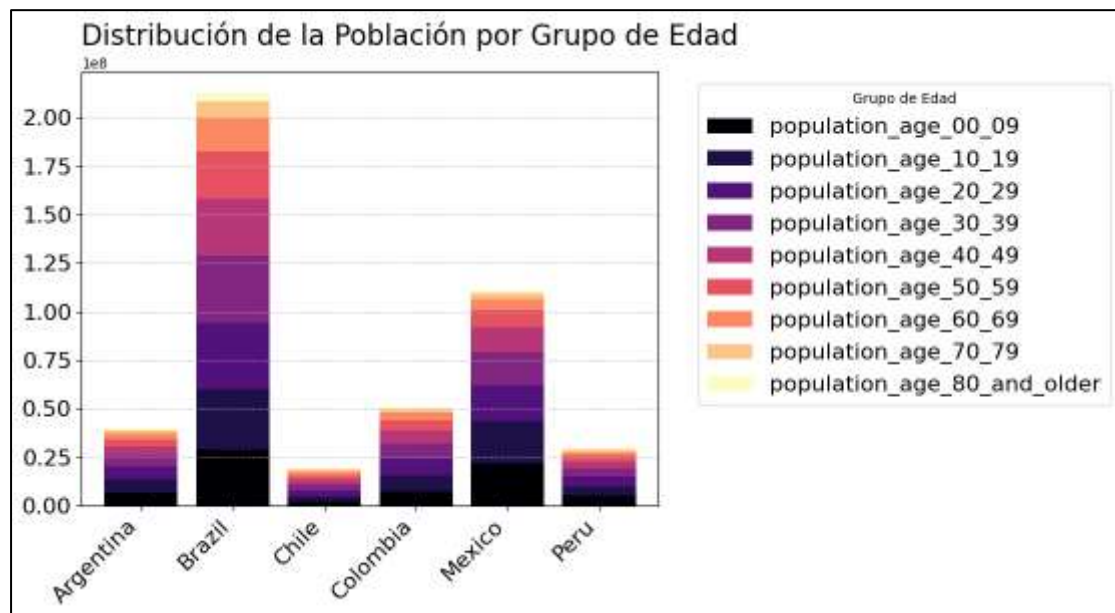
Cada barra representa a un país, y cada color dentro de la barra corresponde a un grupo de edad específico. La suma de las diferentes secciones de cada barra indica la distribución total de la población de ese país por edad.

Ilustración 27: Sintaxis para la elaboración de un Grafico de Barras Apiladas, con la participación de Grupos por Edad



## Módulo 4

Ilustración 28: Distribución de Población por Grupos de Edad



También se compararon las tasas de mortalidad masculina y femenina por país en el siguiente gráfico. Cada barra representa un país, y dentro de cada barra, la sección azul indica la tasa de mortalidad masculina y la sección rosa muestra la tasa de mortalidad femenina.

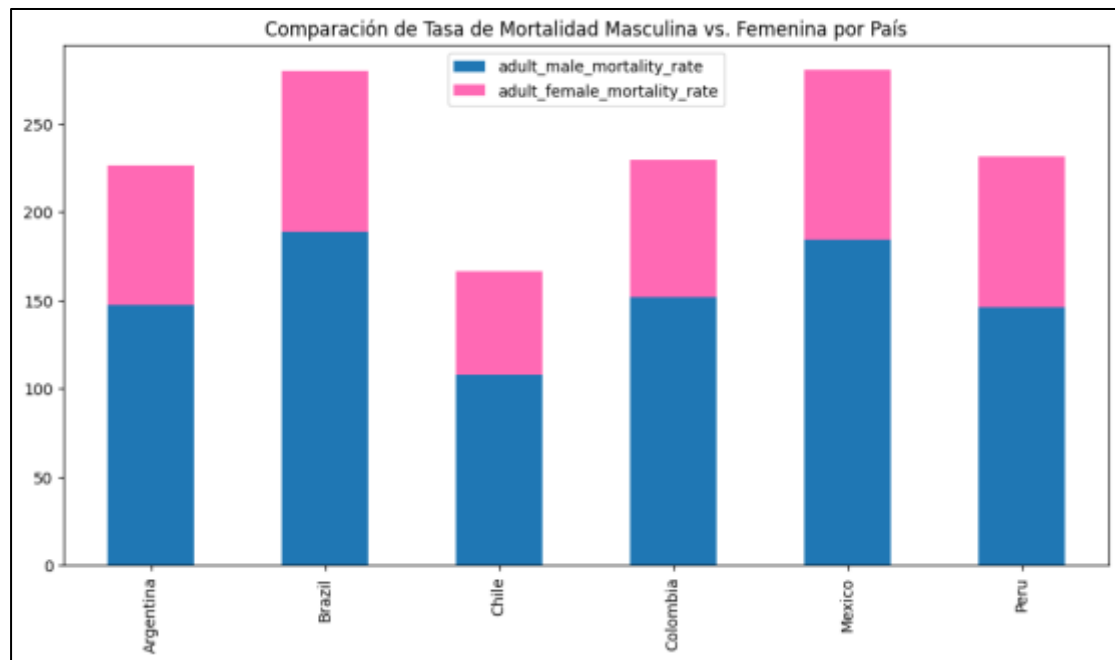
Al ser apiladas, las barras permiten ver la relación entre ambas tasas en cada país y también cómo se comparan en términos absolutos (aunque no es un gráfico de comparación directa en términos de magnitudes absolutas entre países, dado que las barras se apilan).

Ilustración 29: Sintaxis para la Creación de un Grafico de Columnas Apiladas con Tasas de Mortalidad por Género



## Módulo 4

*Ilustración 30: Comparación de Tasa de Mortalidad Masculina vs Femenina por País*



El último de los gráficos consiste en dos subgráficos que comparan la prevalencia de dos condiciones preexistentes (tabaquismo y diabetes) en países con altas y bajas tasas de mortalidad por COVID-19. Los países están divididos en dos grupos según si su tasa de mortalidad es mayor o menor que la mediana global.

*Ilustración 31: Sintaxis para la creación de Subgráficos de Barras Apiladas con Tasas Condiciones Preexistentes y Tasas de Mortalidad*

```
df_clean['mortality_rate'] = df_clean['cumulative_deceased'] / df_clean['cumulative_confirmed'] * 100

country_data = df_clean.groupby('country_name').last().reset_index()
high_mortality_countries = country_data[country_data['mortality_rate'] > country_data['mortality_rate'].median()]
low_mortality_countries = country_data[country_data['mortality_rate'] <= country_data['mortality_rate'].median()]
conditions = ['smoking_prevalence', 'diabetes_prevalence']

fig, ax = plt.subplots(2, 2, figsize=(12, 8), sharey=True)

ax[0].barplot(data=high_mortality_countries, x='country_name', y='smoking_prevalence', color='purple', ax=ax[0], label='Tabaquismo')
ax[0].barplot(data=high_mortality_countries, x='country_name', y='diabetes_prevalence', color='violet', ax=ax[0], label='Diabetes', alpha=0.7)
ax[0].set_title('Alta mortalidad')
ax[0].set_xlabel('País')
ax[0].set_ylabel('Prevalencia (%)')
ax[0].legend(title='Condiciones')
ax[0].tick_params(axis='x', rotation=45)

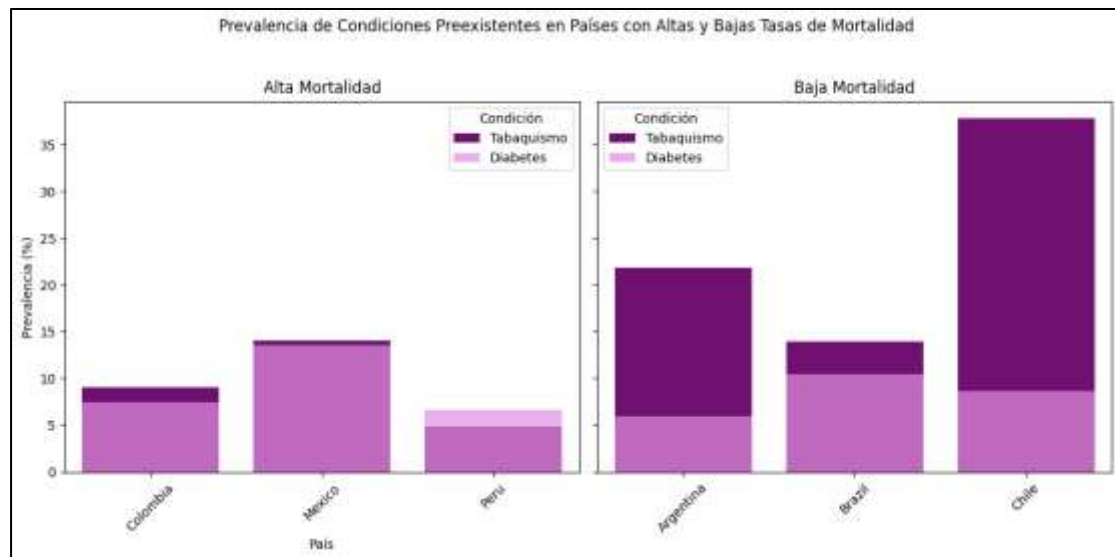
ax[1].barplot(data=low_mortality_countries, x='country_name', y='smoking_prevalence', color='purple', ax=ax[1], label='Tabaquismo')
ax[1].barplot(data=low_mortality_countries, x='country_name', y='diabetes_prevalence', color='violet', ax=ax[1], label='Diabetes', alpha=0.7)
ax[1].set_title('Baja mortalidad')
ax[1].set_xlabel('País')
ax[1].legend(title='Condiciones')
ax[1].tick_params(axis='x', rotation=45)

plt.suptitle('Prevalencia de Condiciones Preexistentes en Países con Altas y Bajas Tasas de Mortalidad')
plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()
```



**Módulo 4**

*Ilustración 32: Prevalencia de Condiciones Preexistentes en Países, con Altas Tasas de Mortalidad y Bajas Tasas de Mortalidad*



### 3. Descripción de condiciones climáticas:

Por otro lado, se realizó también un análisis de las condiciones ambientales de los diferentes países, para conocer y detectar parámetros que puedan indicar o establecer relaciones con el COVID-19.

El objetivo del primer gráfico es mostrar la distribución de las temperaturas medias de los países, ayudando a identificar valores extremos (como países con temperaturas anormalmente altas o bajas), la variabilidad dentro de los países y cómo se comparan entre sí. Además, el uso de un boxplot permite observar la simetría o asimetría de la distribución de las temperaturas en cada país.

En este boxplot se puede apreciar que los países con una mayor temperatura media son Brasil, Perú, Colombia y México, con 28°, 26.5°, 20.2° y 19.46° grados respectivamente, aunque Argentina también muestra una amplia gama de temperaturas, que van de los 33 a los 0 grados.

En cuanto a los outliers o “casos extremos” que, por ejemplo, registra Brasil, si bien en este caso no se modificaron, si se debe tener en cuenta existen y que es útil identificarlos y definir cómo tratarlos, ya que pueden distorsionar el análisis o los modelos predictivos.

En este sentido, existen varias formas de trabajar con ellos:

#### 1. Identificación de Outliers

- Método del Rango Inter cuartil (IQR): Calcula el rango intercuartil:  $IQR = Q3 - Q1$  (donde  $Q1$  y  $Q3$  son el primer y tercer cuartil, respectivamente). Define como outliers aquellos valores menores a  $Q1 - 1.5 * IQR$  o mayores a  $Q3 + 1.5 * IQR$ .

**Módulo 4**

- Desviación Estándar: Si los datos siguen una distribución normal, los valores fuera de 2 o 3 desviaciones estándar de la media pueden considerarse outliers.
- Visualización: Usa gráficos como boxplots (diagramas de caja) o scatter plots para visualizar y detectar outliers de forma rápida.

**2. Tratamiento de Outliers**

Una vez identificados, decide cómo manejar los outliers. Algunas opciones son:

- Eliminación: Puedes excluir los outliers si están relacionados con errores de medición o si su exclusión no afecta el análisis general. Sin embargo, eliminar outliers es riesgoso si representan datos válidos pero extremos.
- Transformación: Aplicar transformaciones como el logaritmo, raíz cuadrada o Box-Cox puede reducir el efecto de los outliers al disminuir la asimetría en la distribución.
- Sustitución o Imputación: Reemplaza los outliers con valores más representativos, como la media o mediana del conjunto de datos, o utiliza técnicas de imputación más avanzadas.
- Asignación de Límites: Puedes limitar los valores atípicos extremos a un valor máximo o mínimo. Por ejemplo, si un valor excede  $Q3 + 1.5 * IQR$ , puedes asignarle el valor límite  $Q3 + 1.5 * IQR$ .
- Uso de Modelos Robustecidos: Algunos modelos, como la regresión robusta o el uso de árboles de decisión, son menos sensibles a los outliers y permiten incluirlos en el análisis sin distorsionar los resultados.

*Ilustración 33: Sintaxis para la elaboración de un Boxplot con Temperaturas Medias por Países*

```
ejemplo33 de temperatura media por país

orden_paises = sorted(df['country_name'].unique())

plt.figure(figsize=(12, 6))

ax = boxplot(data=df['temp'], p=country_name, y='average_temperature_celsius', palette='magma', order=orden_paises)

plt.xticks(rotation=90)

plt.title('Distribución de la Temperatura Media por País')

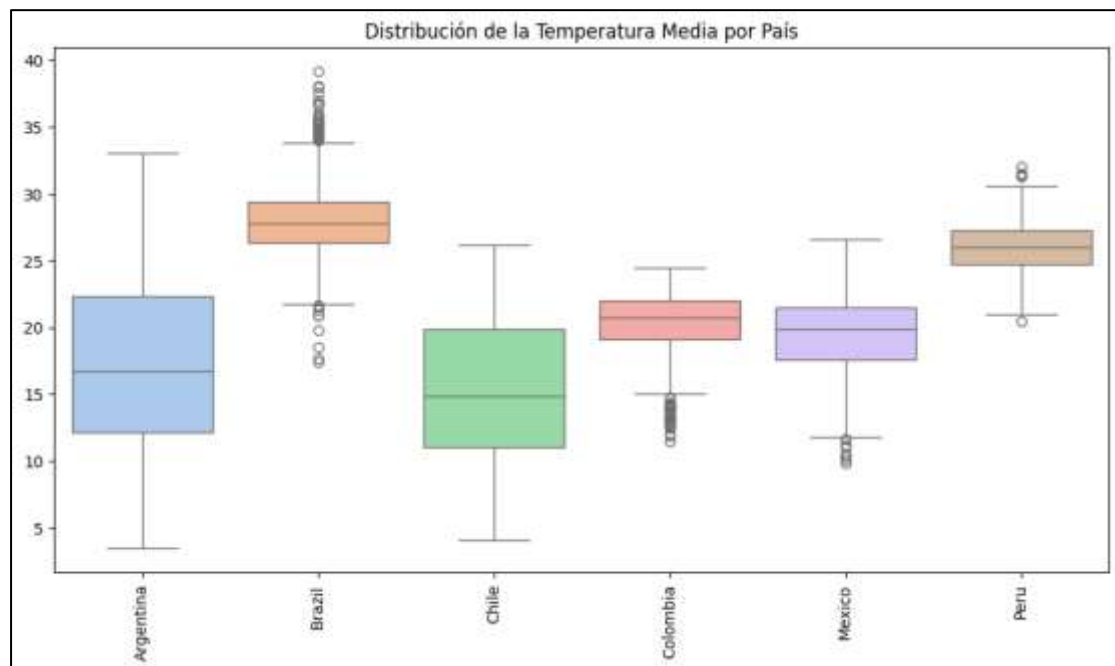
plt.xlabel('')

plt.ylabel('')

plt.show()
```



Ilustración 34: Distribución de Temperatura Media por País



Por otro lado, también, se realizaron diagramas de dispersión para comparar la temperatura media de cada país con los casos confirmados y muertes confirmadas. En estos gráficos se puede observar una posible relación positiva entre el aumento de la temperatura y los aumentos de casos y muertes por COVID-19. Esta situación se observa puntualmente en Brasil y Perú y, en algunos casos particulares, de Argentina.

Ilustración 35: Sintaxis para la elaboración de Scatterplots con Temperatura Media vs Casos Confirmados y Muertes Confirmadas

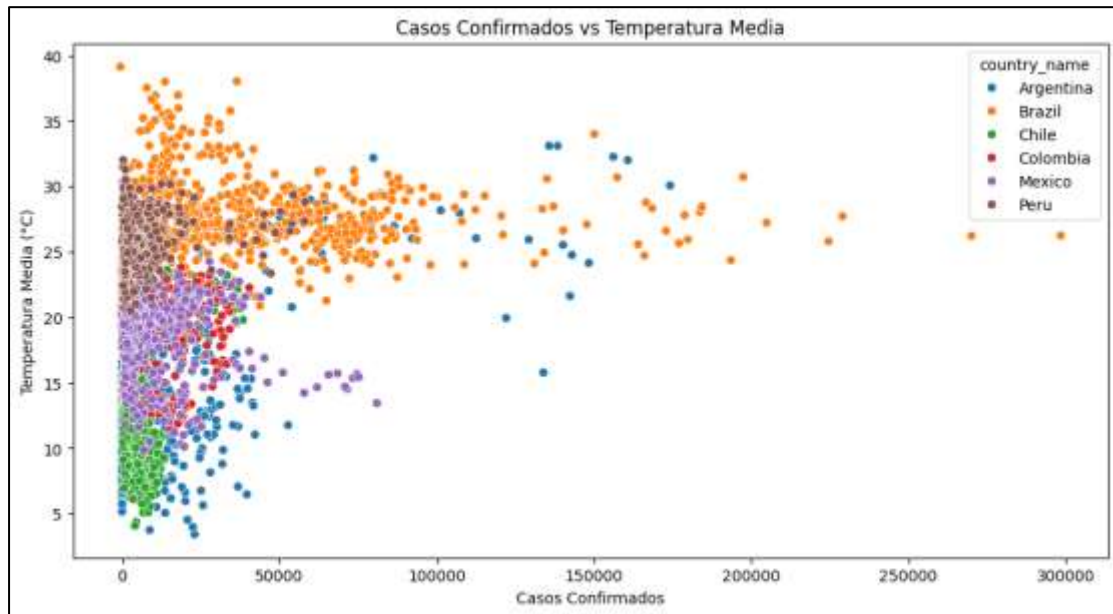
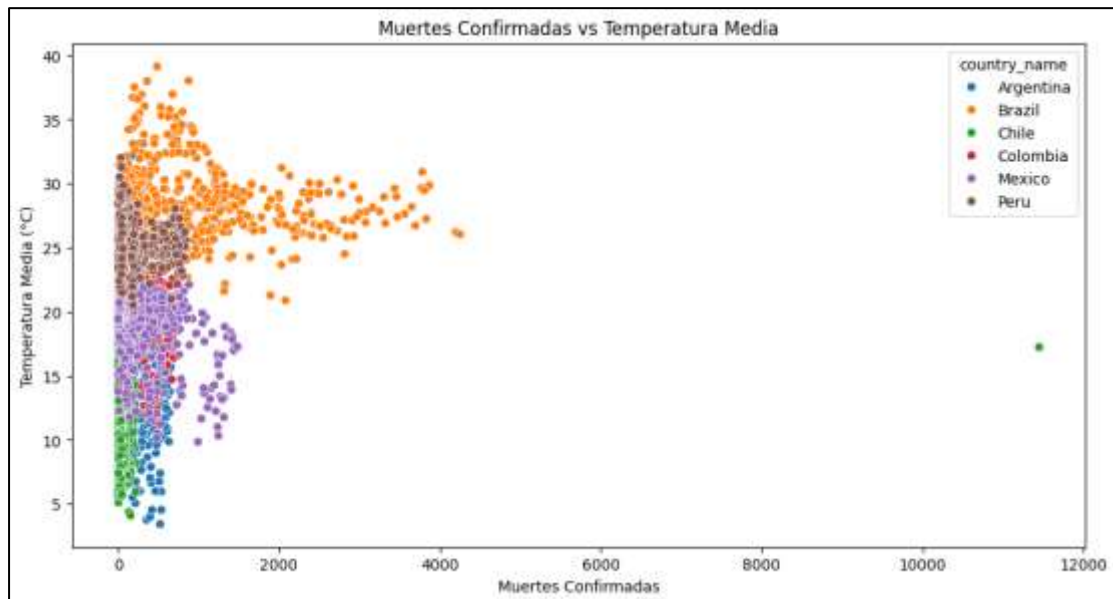
```

# CODIGOS DE DISTRIBUCIÓN: TEMPERATURA MEDIA VS CASOS CONFIRMADOS Y TEMPERATURA MEDIA VS MUERTES CONFIRMADAS
orden_paises = sorted(df_limpio['country_name'], ascending=True)

# Gráfico de Casos Confirmados vs Temperatura Media
plt.figure(figsize=(12, 8))
sns.scatterplot(data=df_limpio, x="avg_confirmed", y="average_temperature_paises", hue="country_name", hue_order=orden_paises)
plt.title("Casos Confirmados vs Temperatura Media")
plt.xlabel("Casos Confirmados")
plt.ylabel("Temperatura Media (°C)")
plt.show()

# Gráfico de Muertes Confirmadas vs Temperatura Media
plt.figure(figsize=(12, 8))
sns.scatterplot(data=df_limpio, x="avg_deaths", y="average_temperature_paises", hue="country_name", hue_order=orden_paises)
plt.title("Muertes Confirmadas vs Temperatura Media")
plt.xlabel("Muertes Confirmadas")
plt.ylabel("Temperatura Media (°C)")
plt.show()

```

**Módulo 4***Ilustración 36: Casos Confirmados vs Temperatura Media por País**Ilustración 37: Temperatura Media vs Muertes Confirmadas por País*

A continuación, se realizaron gráficos de Violinplots, útiles para identificar la distribución de los datos, las asimetrías y los valores atípicos en cada país, ofreciendo una visión más profunda que un simple gráfico de barras o de cajas.

Ilustración 38: Sintaxis para la Creación de Violinplots

```
#VIOLINPLOT DE VARIABLES POR PAIS

columnas_violin = ['new_confirmed', 'new_deceased', 'relative_humidity', 'average_temperature_celsius']
orden_paises = sorted(df_limpio['country_name'].unique())

num_cols = 2
num_rows = (len(columnas_violin) + num_cols - 1) // num_cols

fig, axes = plt.subplots(num_rows, num_cols, figsize=(12, num_rows * 5))
fig.suptitle("Distribución de Variables por País", fontsize=16)

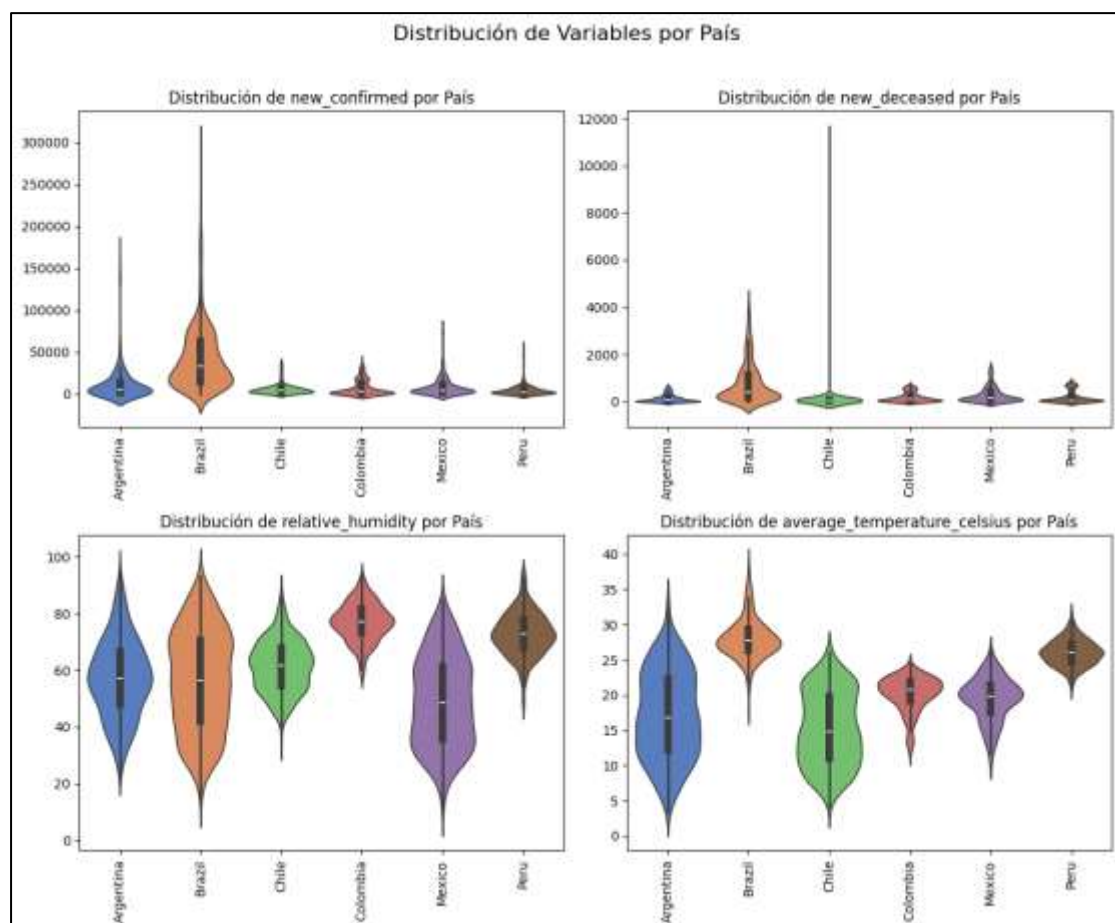
axes = axes.flatten()

for i, col in enumerate(columnas_violin):
    sns.violinplot(data=df_limpio, x='country_name', y=col, palette='muted', ax=axes[i], order=orden_paises)
    axes[i].set_title(f"Distribución de {col} por País", fontsize=12)
    axes[i].tick_params(axis='x', rotation=90)
    axes[i].set_xlabel('')
    axes[i].set_ylabel('')

for j in range(i + 1, len(axes)):
    axes[j].set_visible(False)

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

Ilustración 39: Distribución de Variables por País



### Avance 3: EDA con Numpy y Pandas

En este penúltimo avance del proyecto, se ha realizado un análisis detallado y preparación de datos para la visualización avanzada de la incidencia del COVID-19.

Se comenzó por realizar análisis de series de tiempo. Los primeros gráficos presentados muestran el comportamiento de los nuevos casos confirmados a lo largo del tiempo, con un enfoque en los promedios mensuales.

Ambos gráficos tienen el propósito de visualizar tendencias temporales en la incidencia del COVID-19, pero con una diferencia clave: uno muestra el comportamiento global, mientras que el otro permite hacer comparaciones entre países.

Estos gráficos nos permiten identificar que hubo tres grandes periodos de tiempo donde los casos se incrementaron significativamente. El primero se dio durante los primeros meses del año 2021, alcanzando un promedio global de mas de 20 mil casos confirmados en el mes de junio. El segundo aumento de casos se dio entre los meses de enero y febrero del 2022 (durante el verano sudamericano), donde se hizo presente el pico máximo de casos promedio por mes: 50.000. La última vez que se experimento un incremento en la cantidad de casos fue en junio de ese mismo año, pero alcanzando apenas los 15.000 casos promedio.

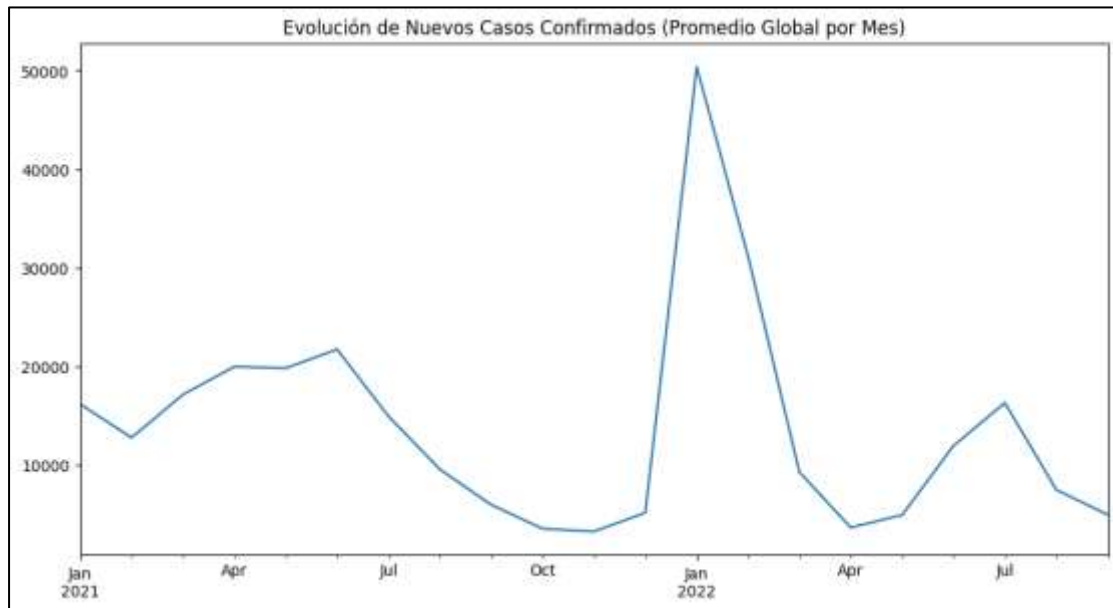
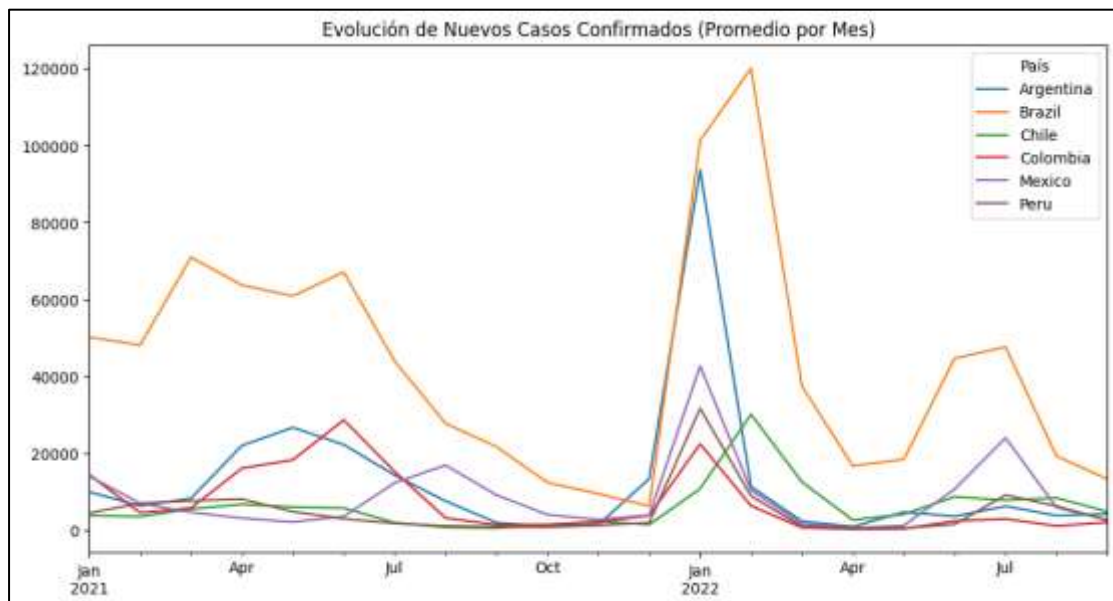
*Ilustración 40: Sintaxis para la Elaboración de Gráficos Lineales por Mes*

```
#EVOLUCIÓN DE VARIABLES A TRAVÉS DEL TIEMPO (MES)
df_limpio['date'] = pd.to_datetime(df_limpio['date'])
df_limpio['month'] = df_limpio['date'].dt.to_period('M')

casos_mensual_global = df_limpio.groupby('month')['new_confirmed'].mean()
casos_mensual_global.plot(figsize=(12, 6))
plt.title("Evolución de Nuevos Casos Confirmados (Promedio Global por Mes)")
plt.xlabel('')
plt.ylabel('')
plt.show()

casos_mensual = df_limpio.groupby(['country_name', 'month'])['new_confirmed'].mean().unstack()
casos_mensual.T.plot(figsize=(12, 6))
plt.title("Evolución de Nuevos Casos Confirmados (Promedio por Mes)")
plt.xlabel('')
plt.ylabel('')
plt.legend(title="País")
plt.show()
```

[74] ✓ 0.3s

**Módulo 4***Ilustración 41: Evolución de Nuevos Casos Confirmados (Promedio Global por Mes)**Ilustración 42: Evolución de Nuevos Casos Confirmados (Promedio Mensual por País)*

Luego se procedió con la evolución de los casos acumulados confirmados de COVID-19, pero con un enfoque en los valores en millones y desglosados tanto a nivel global como por país.

A partir de aquí se observó que la sumatoria de casos registrados todos estos países durante este período fueron de 65 millones.

Ilustración 43: Sintaxis para la Elaboración de Gráficos Lineales

```

df_limpio['date'] = pd.to_datetime(df_limpio['date'])
df_limpio['month'] = df_limpio['date'].dt.to_period('M')

df_limpio['cumulative_confirmed_millones'] = df_limpio['cumulative_confirmed'] / 1e6

casos_mensual_millones = df_limpio.groupby('month')['cumulative_confirmed_millones'].max()

ax = casos_mensual_millones.plot(figsize=(12, 8))
plt.title("Evolución de Casos Acumulados Confirmados (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.show()

casos_mensual_país = df_limpio.groupby(['country_name', 'month'])['cumulative_confirmed_millones'].max().unstack()
casos_mensual_país.T.plot(figsize=(12, 8))
plt.title("Evolución de Casos Acumulados por Mes (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.legend(['País'])
plt.show()

```

Ilustración 44: Evolución de Casos Acumulados Confirmados (en Millones)

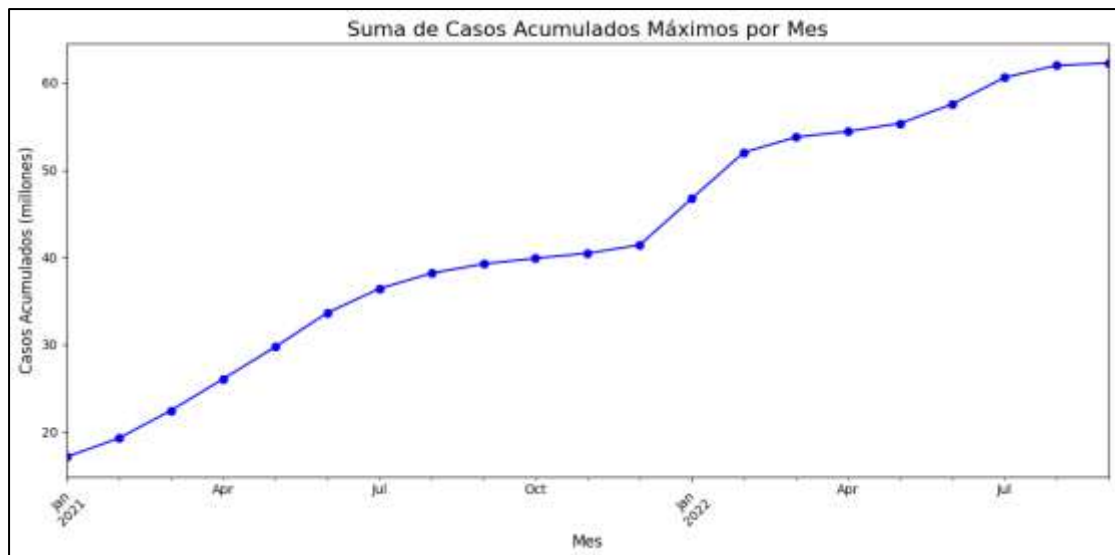
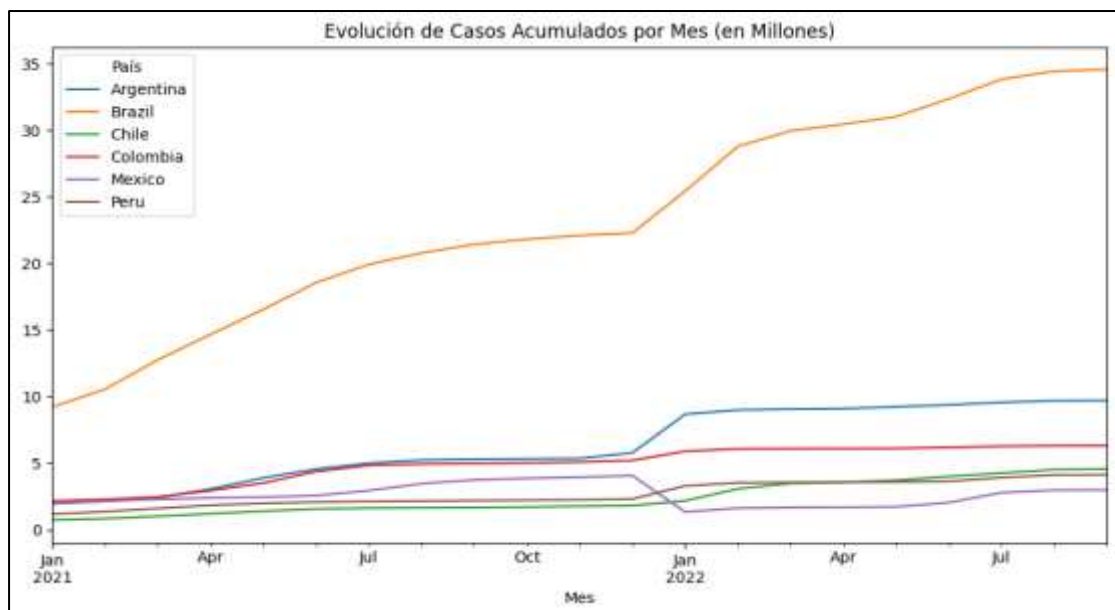


Ilustración 45: Evolución de Casos Acumulados por Mes por País en Millones





**Módulo 4**

También se realizó un análisis de la evolución mensual de las dosis de vacuna administradas en términos de millones. Los análisis se centran en el total de dosis suministradas a lo largo del tiempo y también ofrecen un desglose por país.

Aquí se pudo determinar que el total de dosis suministradas fue de 900 millones, siendo Brasil el que mas vacunas aplico (aproximadamente 350 millones), seguido por México (200 millones) y Argentina (100 millones).

*Ilustración 46: Sintaxis para la elaboración de Grafico Lineal de Vacunación Mensual*

```
df_lineas['cumulative_vaccine_doses_administered_millions'] = df_lineas['cumulative_vaccine_doses_administered'] / 1e6
mas_dosis_mensual_por_pais = df_lineas.groupby(['country_name', 'month'])['cumulative_vaccine_doses_administered_millions'].max().unstack()

mas_dosis_mensual_por_pais.reset_index(inplace=True)

mas_dosis_mensual_por_pais.plot(figsize=(11, 8))
plt.title("Total de Dosis Suministradas por Mes (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.legend('')
plt.show()

mas_dosis_mensual_por_pais.plot(figsize=(12, 8))
plt.title("Total de Dosis Suministradas por Mes (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.legend(title="País")
plt.show()
```

*Ilustración 47: Total de Dosis Acumuladas (en Millones)*

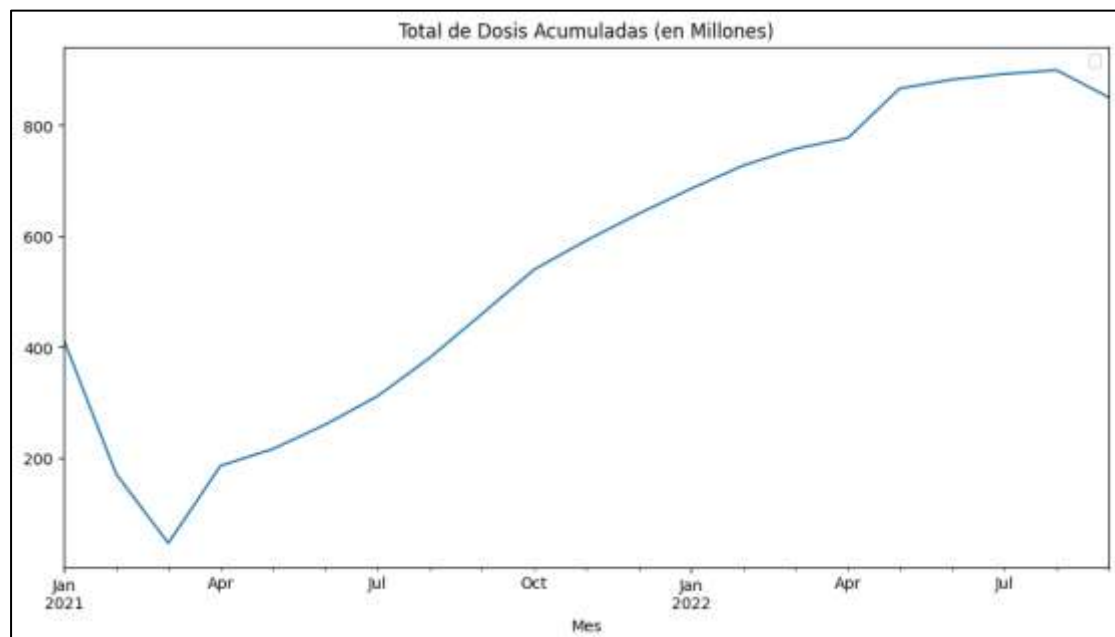
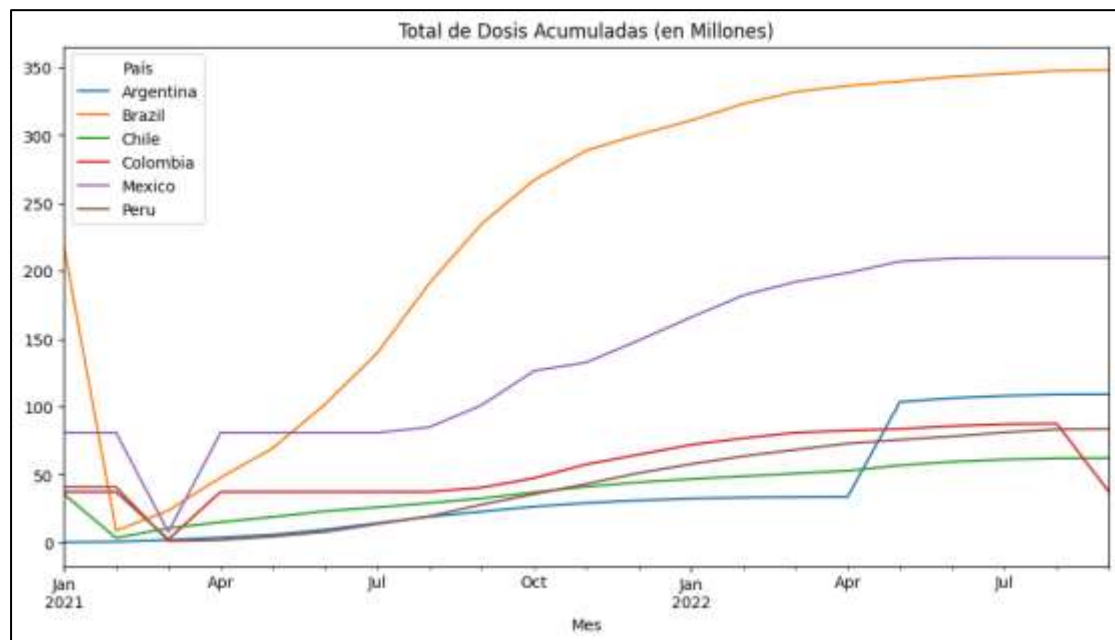


Ilustración 48: Total de Dosis Acumuladas Por País (en Millones)



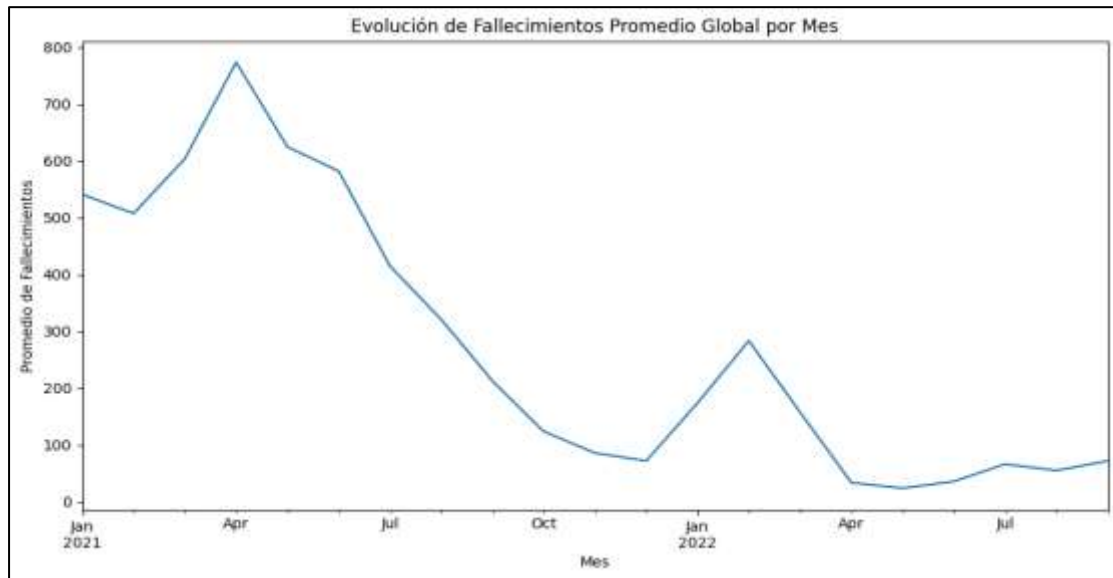
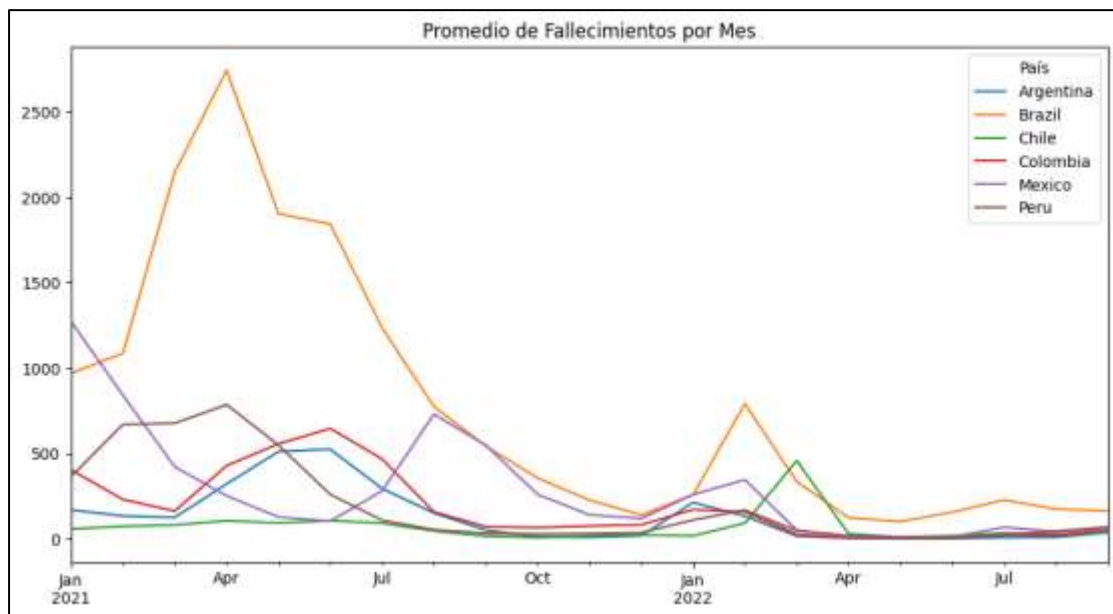
Otros dos gráficos que muestran la evolución de los fallecimientos relacionados con el COVID-19, tanto a nivel global como por país, durante los diferentes meses de la pandemia.

A partir de estos gráficos se detecta que el pico de decesos por COVID-19 se dio en abril de 2021, alcanzando un valor de casi 8.000 muertes promedio mensuales y, desde entonces, descendió significativamente (probablemente por el aumento de las dosis suministradas), hasta el mes de enero del año 2022, donde se registró un segundo salto de muertes por COVID (casi 3.000)

Ilustración 49: Sintaxis para la elaboración de un Grafico Lineal de Fallecimientos Mensuales





**Módulo 4***Ilustración 50: Evolución de Decesos Promedio Global por Mes**Ilustración 51: Promedio de Decesos por País por Mes*

Finalmente, los últimos gráficos detallan la evolución de los fallecimientos acumulados a nivel global y por país de COVID-19. Aquí se determinó que el total de fallecimientos fue de 1.550.000 aproximadamente.

## Módulo 4

*Ilustración 52: Sintaxis para la elaboración de un Grafico Lineal de Fallecimientos Acumulados*

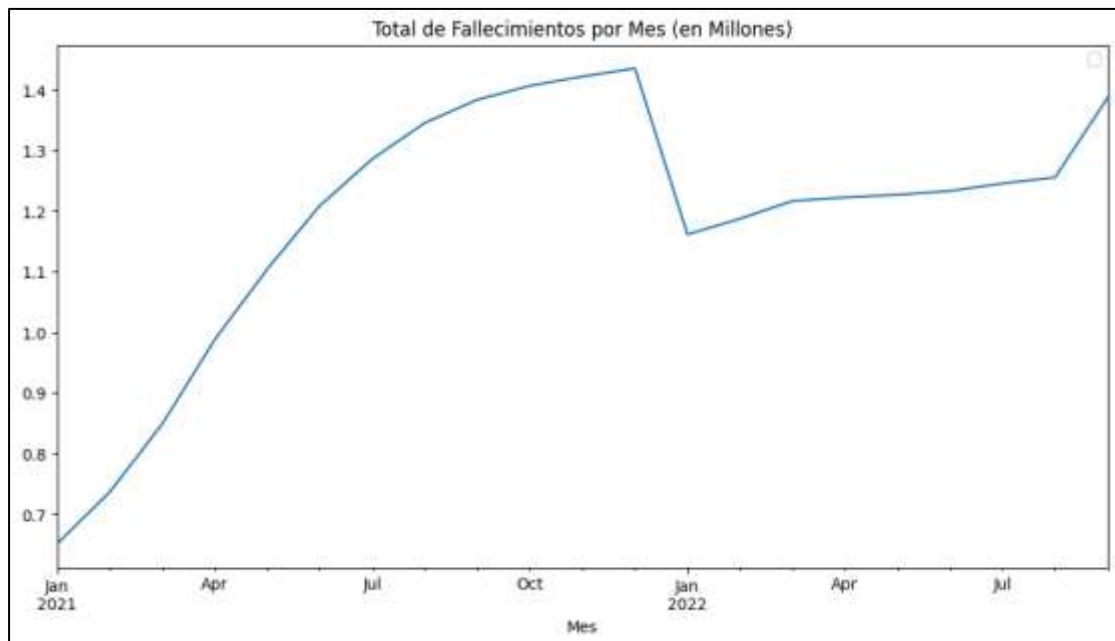
```
df_lineas['cumulative_deceased_millions'] = df_lineas['cumulative_deceased'] / 1e6
max_deceased_anual_por_pais = df_lineas.groupby(['country_name', 'month'])['cumulative_deceased_millions'].max().unstack()

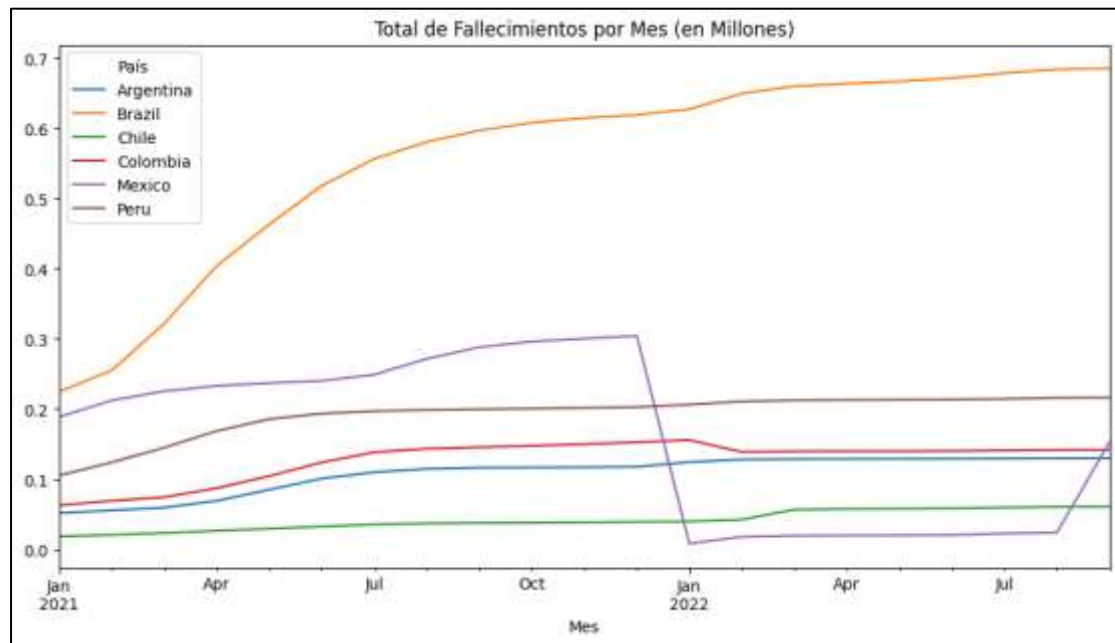
max_max_deceased = max_deceased_anual_por_pais.max()

max_max_deceased.T.plot(figsize=(12, 8))
plt.title("Total de Muertes Fallecimientos por Mes (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.legend('')
plt.show()

max_deceased_anual_por_pais.T.plot(figsize=(12, 8))
plt.title("Total de Fallecimientos por Mes (en Millones)")
plt.xlabel('Mes')
plt.ylabel('')
plt.legend('')
plt.show()
```

*Ilustración 53: Total de Fallecimientos por Mes (en Millones)*



**Módulo 4**

Otros gráficos de temporalidad:

- a. Análisis de autocorrelación: Estos gráficos utilizan la función de autocorrelación (ACF) para analizar la relación temporal en los datos mensuales de nuevos casos y decesos de COVID-19, permitiendo ver cómo cada valor se correlaciona con sus valores anteriores en diferentes lags (retardos o desfases).

Patrones de Tendencia: Si la autocorrelación es alta en varios lags consecutivos, esto puede indicar una tendencia en los datos (por ejemplo, si los casos o decesos aumentan o disminuyen de forma continua).

Estacionalidad: Si ves picos de autocorrelación a intervalos específicos (por ejemplo, cada 12 meses), esto podría indicar estacionalidad, es decir, patrones que se repiten anualmente.

- Gráfico de Autocorrelación de Nuevos Casos por Mes: Si la autocorrelación es alta para ciertos lags, significa que los valores de nuevos casos de meses anteriores tienen una fuerte relación con los valores actuales. Esto puede indicar una tendencia o patrones de estacionalidad (repeticiones periódicas en los datos).
  - Gráfico de Autocorrelación de Nuevos Decesos por Mes: Un coeficiente alto de autocorrelación en ciertos lags indicaría que el número de decesos en un mes podría estar relacionado con los valores de decesos en meses previos, sugiriendo patrones de tendencia o estacionalidad en la serie de decesos.
- b. Evolución mensual de los nuevos casos confirmados y la temperatura promedio: Este tipo de gráfico es útil para observar posibles correlaciones visuales entre la temperatura y los nuevos casos

**Módulo 4**

confirmados. Por ejemplo, si aumentan los casos durante ciertos rangos de temperatura, podrías explorar si la temperatura influye en la propagación del virus. La combinación de ambos ejes ayuda a notar simultáneamente patrones o coincidencias estacionales entre las dos variables.

c. Evolución de la tasa de mortalidad de COVID-19. Ambos gráficos ayudan a identificar tendencias y variaciones en la mortalidad a nivel global y por país:

- Tasa de Mortalidad Global por Mes: Muestra cómo ha cambiado la mortalidad global (porcentaje de decesos respecto a casos confirmados) a lo largo del tiempo.
- Tasa de Mortalidad por Mes y País: Compara la tasa de mortalidad mensual entre países, permitiendo observar diferencias regionales en la gravedad de la pandemia.

d. Evolución semanal

- Promedio Global por Semana: Muestra la tendencia global semanal de nuevos casos, permitiendo ver el aumento o disminución promedio de contagios en el tiempo.
- Nuevos Casos por Semana y País: Compara la evolución semanal de casos entre países, mostrando las diferencias en la propagación del virus por región.
- Evolución semanal de nuevos casos confirmados y nuevas muertes de COVID-19: Este gráfico es útil para analizar la relación entre el número de nuevos casos y las muertes en un período de tiempo semanal.

e. Casos confirmados de COVID-19 en función del día de la semana

- Casos Confirmados por Día de la Semana (Global): Permite identificar si existe un patrón en los reportes de casos por día, como aumentos los lunes tras el fin de semana, o días de menos reportes (posiblemente por reducción de pruebas en ciertos días).
- Casos Confirmados por Día de la Semana (por País): Este gráfico permite comparar el patrón semanal de casos entre distintos países, mostrando si algunos países tienen días específicos de reportes más altos o bajos, posiblemente debido a diferencias en los sistemas de reportes o prácticas de pruebas.

*Ilustración 54: Sintaxis para la elaboración de Gráficos de Autocorrelación*

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
tasa_confirmados_global = df_limpio.groupby("month")["nuevos_confirmados"].mean()

plt.figure(figsize=(12, 6))
plot_acf(tasa_confirmados_global, lags=12)
plt.title('Autocorrelación de Nuevos Casos por Mes')
plt.xlabel('lags')
plt.ylabel('Autocorrelación')
plt.show()

tasa_confirmados_global = df_limpio.groupby("month")["nuevos_decesos"].mean()

plt.figure(figsize=(12, 6))
plot_acf(tasa_confirmados_global, lags=12)
plt.title('Autocorrelación de Nuevos Decesos por Mes')
plt.xlabel('lags')
plt.ylabel('Autocorrelación')
plt.show()
```

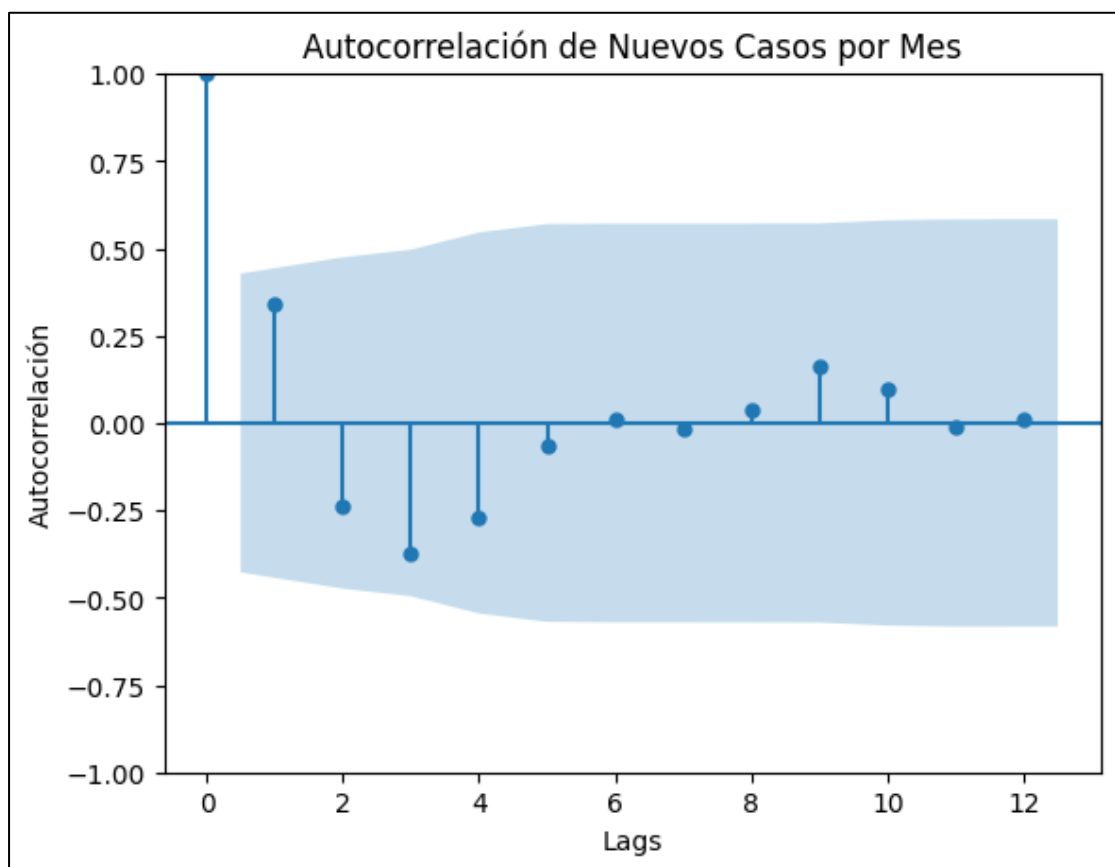
*Ilustración 55: Autocorrelación de Nuevos Casos por Mes*

Ilustración 56: Autocorrelación de Nuevos Decesos por Mes

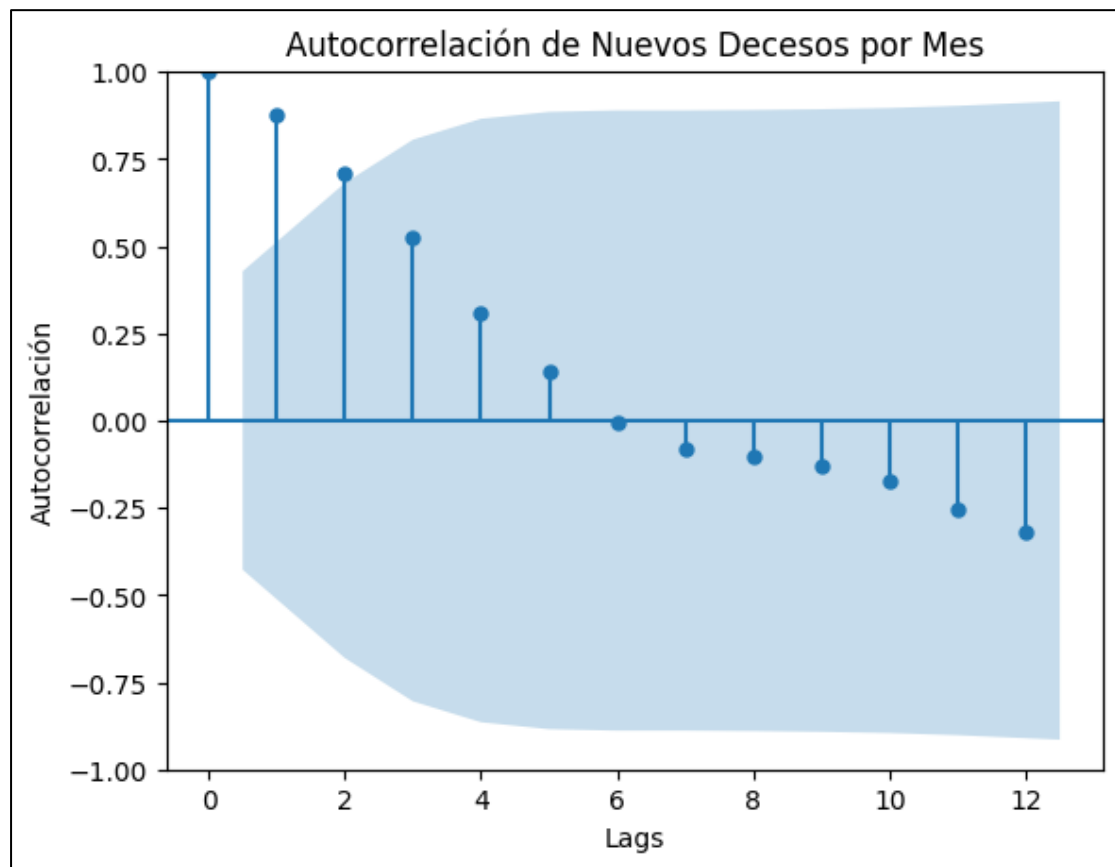


Ilustración 57: Sintaxis para la elaboración de un Grafico Lineal de Nuevos Casos Confirmados Vs Temperatura Media

```

mmsaal = df.groupby('month')[['new_confirmed', 'average_temperature_celsius']].mean()

fig, ax1 = plt.subplots(figsize=(12, 8))
mmsaal['new_confirmed'].plot(ax=ax1, color='blue', label='Nuevos Casos Confirmados', linestyle='-',)
ax1 = ax1.twinx()
mmsaal['average_temperature_celsius'].plot(ax=ax1, color='red', label='Temperatura Promedio (°C)', linestyle='--')

plt.title('Evolución Global de Casos Confirmados y Temperatura Promedio por Mes', fontsize=14)
ax1.set_xlabel('')
ax1.set_ylabel('Nuevos Casos Confirmados', color='blue', fontstyle='italic', fontweight='bold', fontsize=12)
ax2.set_ylabel('Temperatura Promedio (°C)', color='red', fontweight='bold', fontsize=12)

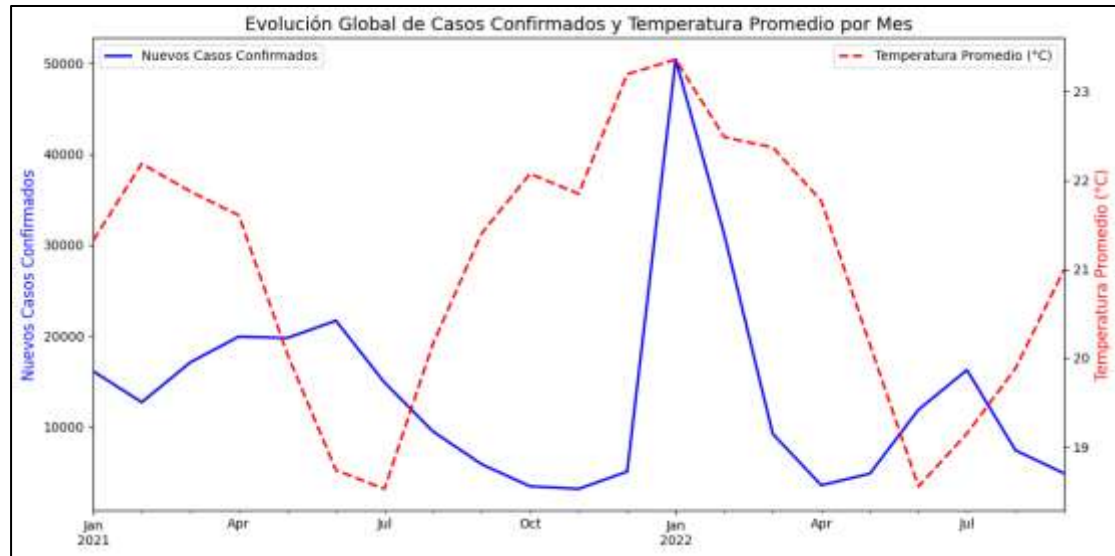
ax1.legend(loc='upper left')
ax2.legend(loc='upper right')

plt.tight_layout()
plt.show()

```

## Módulo 4

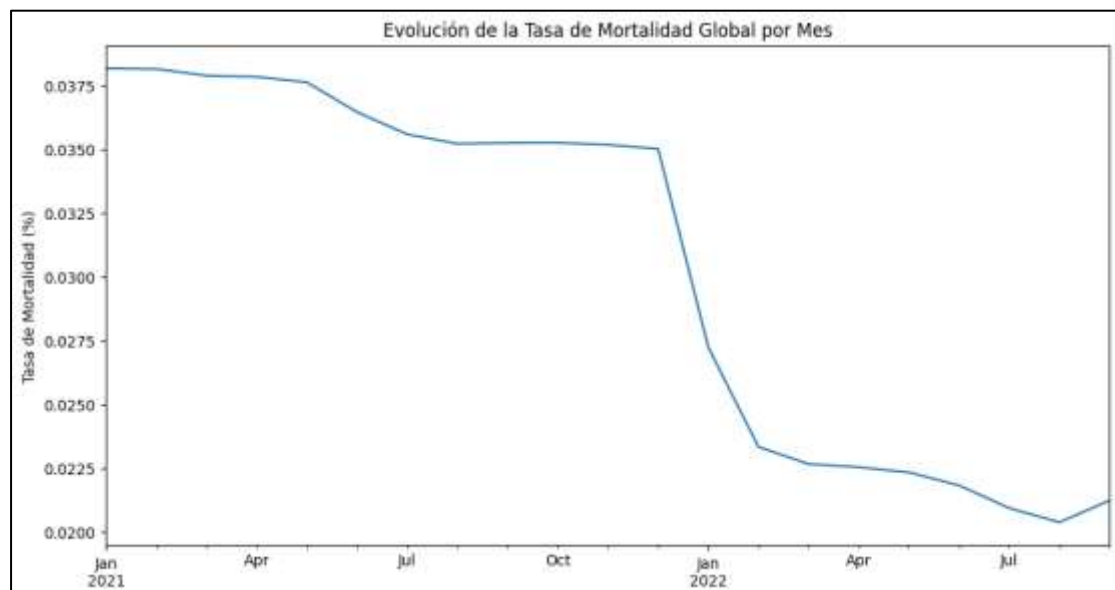
*Ilustración 58: Evolución Global de Casos Confirmados y Temperatura Promedio por Mes*



*Ilustración 59: Sintaxis de elaboración de Grafico Lineales de Tasa de Mortalidad por Mes*

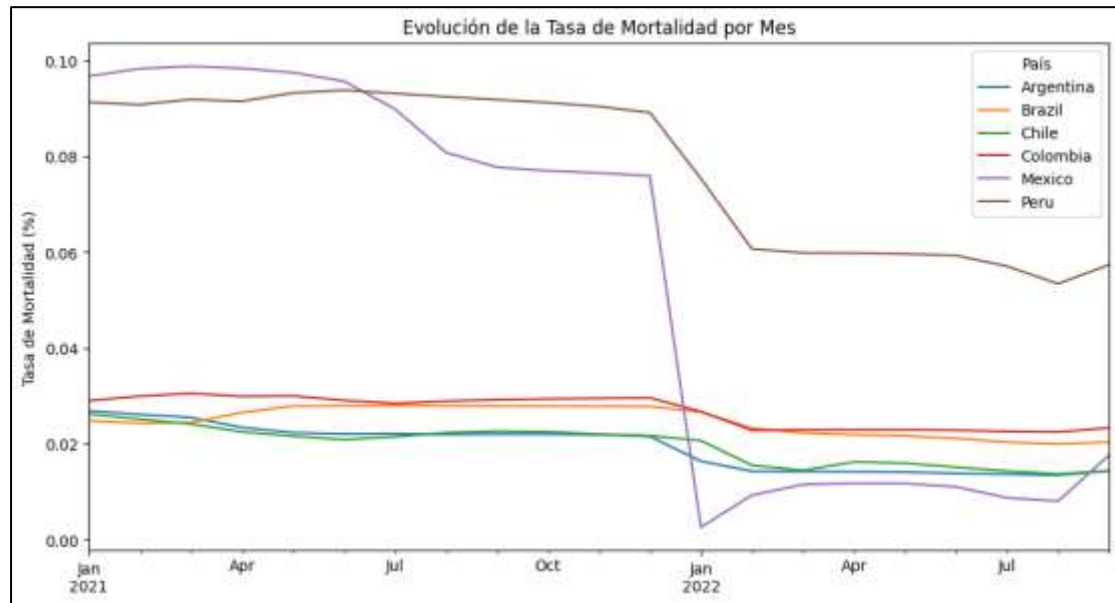


*Ilustración 60: Evolución de la Tasa de Mortalidad Global por Mes*



## Módulo 4

*Ilustración 61: Evolución de Tasa de Mortalidad por Mes por País*



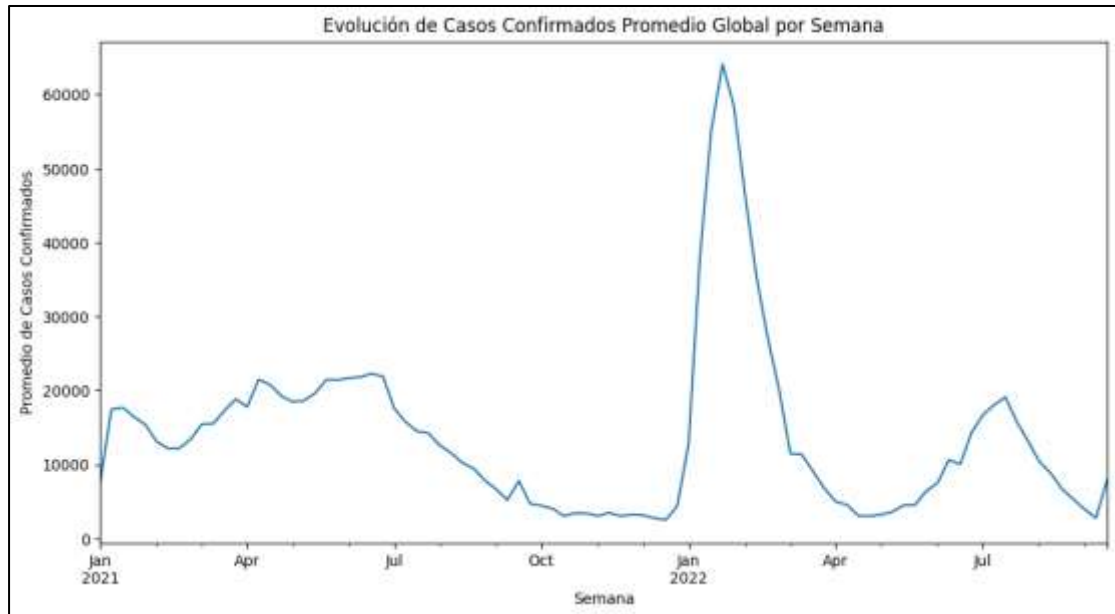
*Ilustración 62: Sintaxis para la elaboración de Gráficos Semanales*

[illegible]

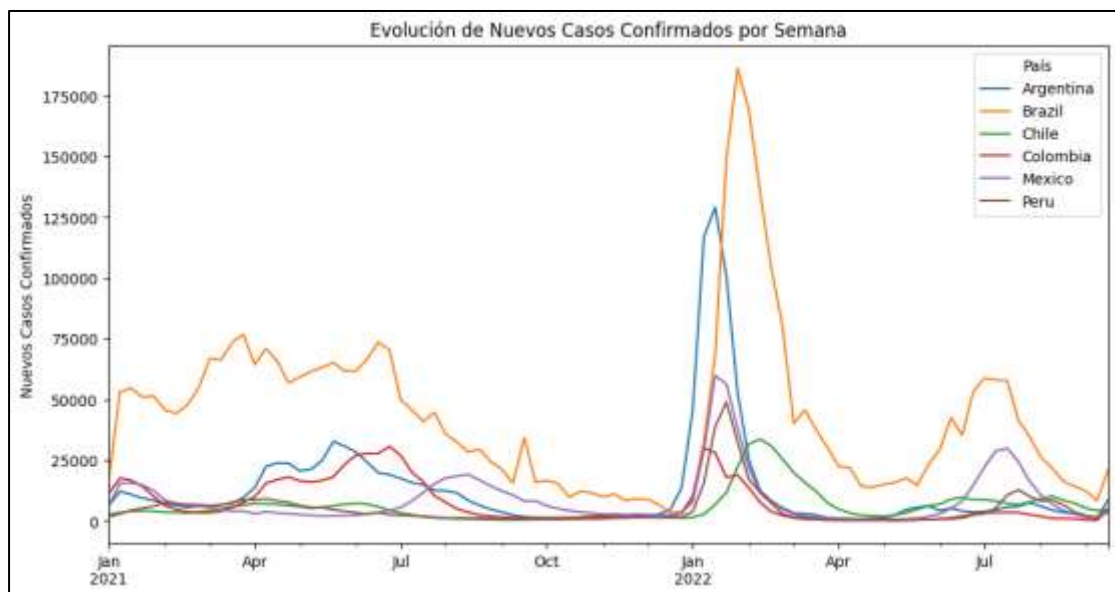


Módulo 4

*Ilustración 63: Evolución de Casos Confirmados, Promedio Global por Semana*



*Ilustración 64: Evolución de Nuevos Casos Confirmados por Semana*



## Módulo 4

Ilustración 65: Nuevos Casos Confirmados y Nuevas Muertes por Semana

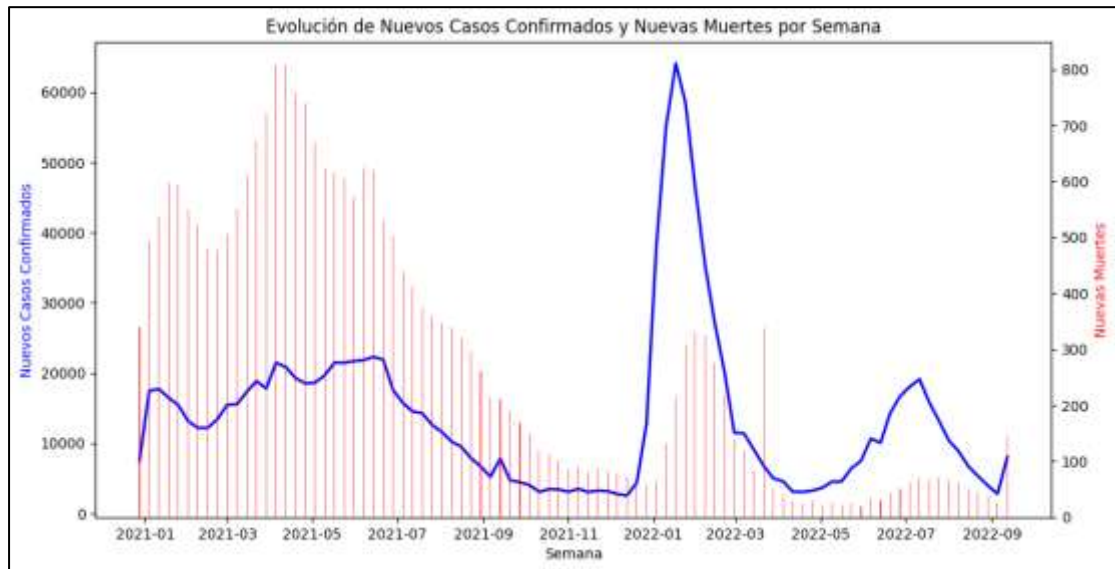


Ilustración 66: Sintaxis para la elaboración de Gráficos Diarios

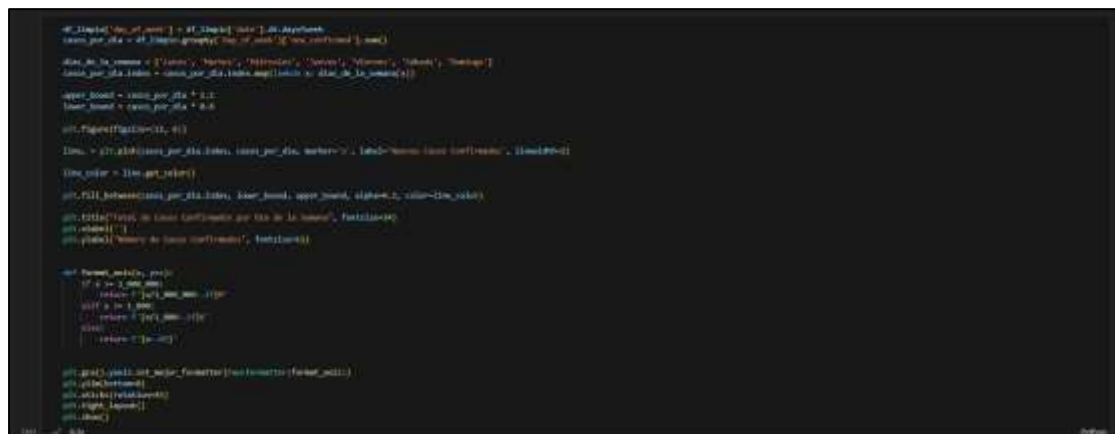
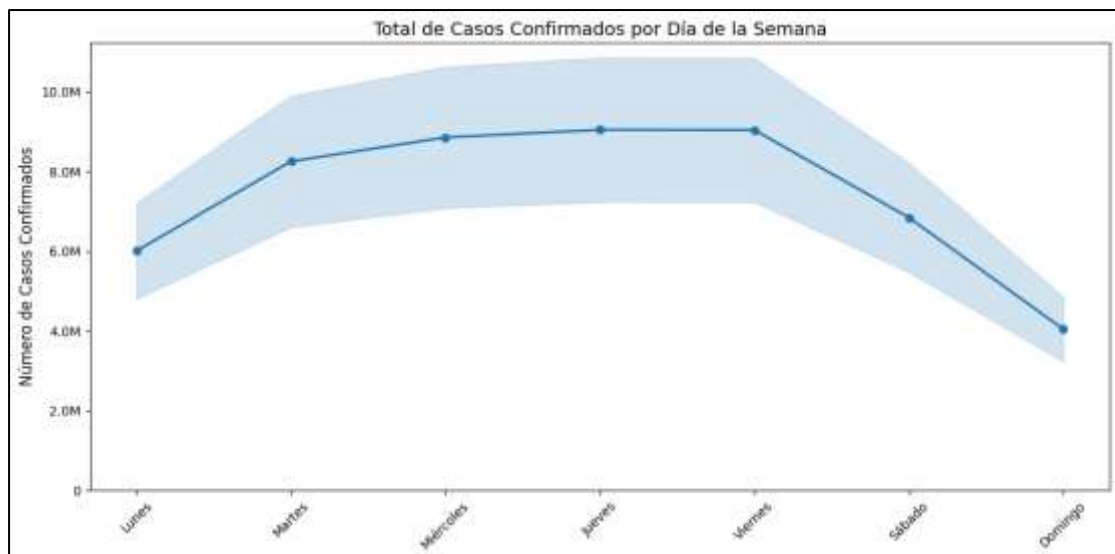


Ilustración 67: Total de Casos Confirmados por Día de la Semana



## Módulo 4

*Ilustración 68: Sintaxis para la elaboración de un Grafico Diario de Casos por País*

```

casos_por_dia_pais = df_casos.groupby(['country_name', 'date_of_report']).sum().reset_index()

dias_de_la_semana = ['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado', 'Domingo']
casos_por_dia_pais.columns = [casos_por_dia_pais.columns[0], casos_por_dia_pais.columns[1]]

plt.figure(figsize=(12, 8))

for pais in casos_por_dia_pais.country_name:
    y = casos_por_dia_pais[pais]

    upper_bound = y * 1.1
    lower_bound = y * 0.9

    lines = [y, upper_bound, lower_bound]
    lines_color = ['black', 'red', 'blue']
    lines_label = ['Caso', 'Upper Bound', 'Lower Bound']

    plt.plot(dias_de_la_semana, y, label=pais, linestyle='solid')
    plt.plot(dias_de_la_semana, upper_bound, label=upper_bound, linestyle='dashed')
    plt.plot(dias_de_la_semana, lower_bound, label=lower_bound, linestyle='dashed')

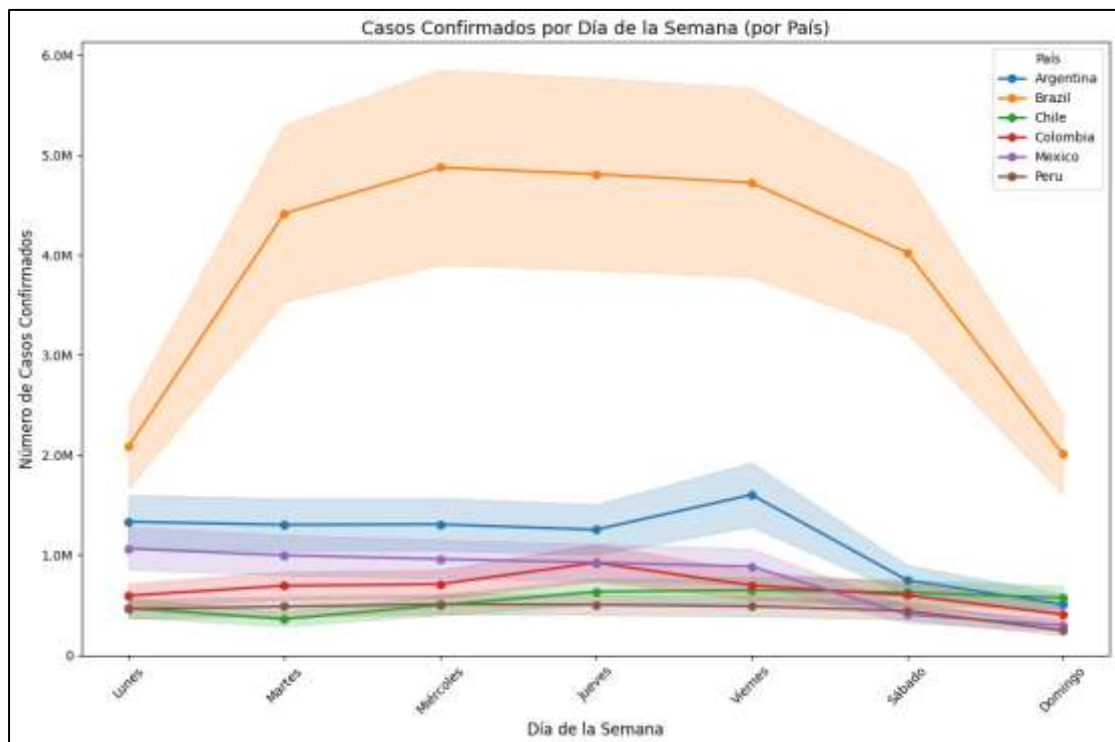
    plt.legend(['Caso', 'Upper Bound', 'Lower Bound'])

plt.title('Casos Confirmados por Día de la Semana (por País)')
plt.xlabel('Día de la Semana')
plt.ylabel('Número de Casos Confirmados')

plt.grid(True)
plt.show()

```

*Ilustración 69: Casos Confirmados por Día de la Semana (por País)*



Análisis de estrategias de vacunación:

Los siguientes tres gráficos de barra permitieron identificar el comportamiento asumido por los diferentes países al momento de enfrentar la enfermedad.

En el primero, se analizó el comportamiento de las dosis administradas por países mediante el promedio de las mismas. Aquí, de la totalidad de vacunas, Brasil representa 200 millones, mientras que México abarca 70 millones de vacunas, seguidos por los demás países.

## Módulo 4

*Ilustración 70: Sintaxis para la elaboración de un Gráfico de barras de vacunas promedio por país*

```

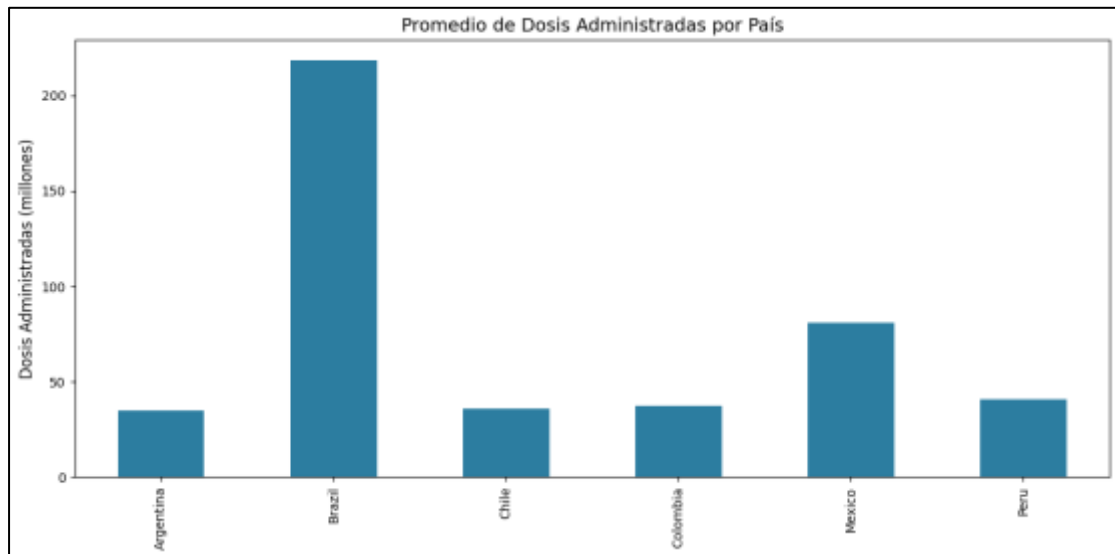
# Importación de las librerías de análisis de datos (de Jupyter) (para Jupyter)
from matplotlib.pyplot import figure, show

# Datos de vacunas promedio por país
data = {'country': 'Argentina', 'doses': 35}, {'country': 'Brazil', 'doses': 220}, {'country': 'Chile', 'doses': 35}, {'country': 'Colombia', 'doses': 35}, {'country': 'Mexico', 'doses': 80}, {'country': 'Peru', 'doses': 40}

# Creación del gráfico de barras
fig = figure(figsize=(12, 6))
plt.bar(data['country'], data['doses'])
plt.title('Promedio de Dosis Administradas por País')
plt.xlabel('País')
plt.ylabel('Dosis Administradas (millones)')
plt.grid(True)
plt.show()

```

*Ilustración 71: Promedio de Dosis Administradas por País*



Sin embargo, el segundo grafico muestra cómo las estrategias de vacunación varían entre diferentes países, con un enfoque en la cantidad de dosis administradas por cada 100 personas de la población. En este caso, se muestra que Chile ha logrado la mejor performance, poniendo a disposición tres vacunas y media cada 100 personas, seguido por Perú con casi tres dosis, y en tercer lugar se encuentra Argentina con casi dos dosis y media.

Al ordenar los países de mayor a menor, se ofrece una comparación visual que facilita identificar qué países han avanzado más en la vacunación frente a la pandemia.

*Ilustración 72: Sintaxis para la elaboración de un gráfico de Barras de Vacunación cada 100 personas*

```

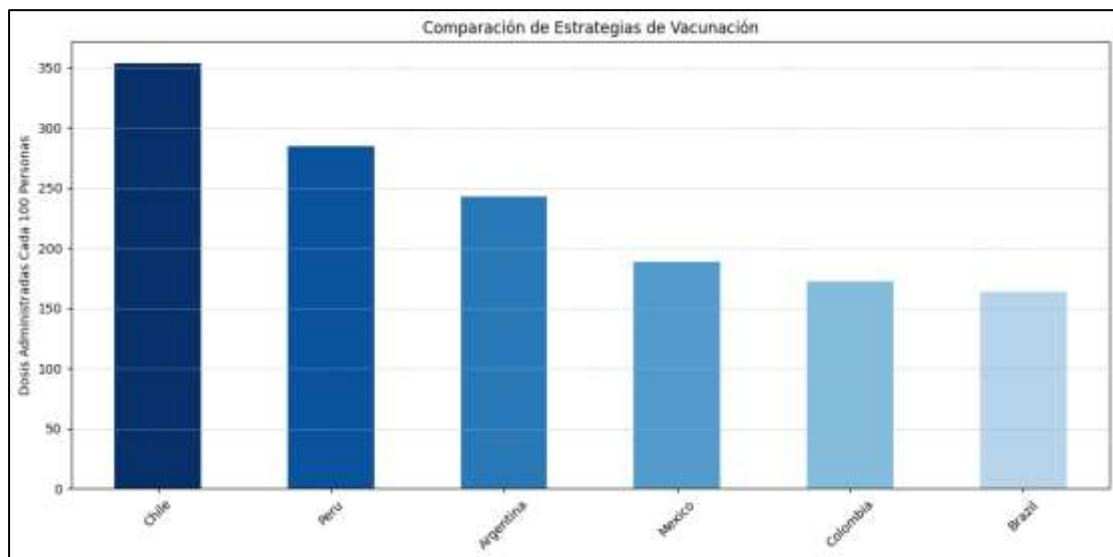
# Importación de las librerías de análisis de datos (de Jupyter) (para Jupyter)
from matplotlib.pyplot import figure, show

# Datos de vacunas por cada 100 personas
data = {'country': 'Argentina', 'doses_per_100': 2.5}, {'country': 'Brazil', 'doses_per_100': 2.2}, {'country': 'Chile', 'doses_per_100': 3.5}, {'country': 'Colombia', 'doses_per_100': 3.5}, {'country': 'Mexico', 'doses_per_100': 0.8}, {'country': 'Peru', 'doses_per_100': 0.4}

# Creación del gráfico de barras
fig = figure(figsize=(12, 6))
plt.bar(data['country'], data['doses_per_100'])
plt.title('Vacunación cada 100 personas')
plt.xlabel('País')
plt.ylabel('Vacunación (por cada 100 personas)')
plt.grid(True)
plt.show()

```

*Ilustración 73: Comparación de Estrategias de Vacunación*



El tercer gráfico, compara la cobertura de vacunación en países de América Latina, mostrando el porcentaje de la población que ha recibido al menos una dosis de la vacuna. Utilizando una paleta de colores que varía con el porcentaje, se facilita la identificación de los países con una mayor o menor tasa de vacunación, lo que permite evaluar la efectividad de las estrategias de vacunación en la región.

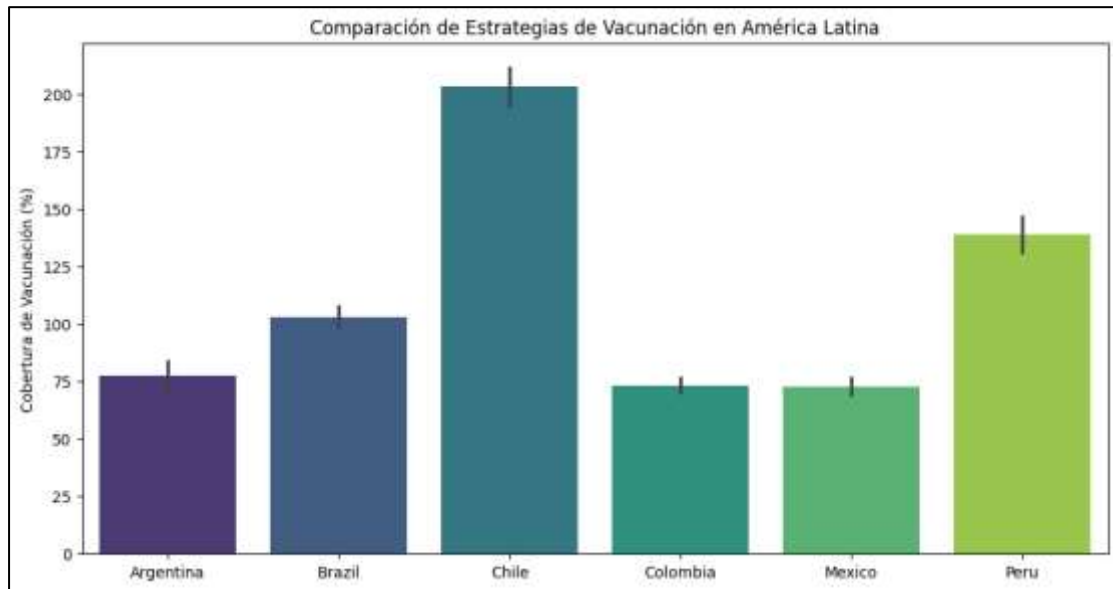
Aquí vemos también que, Chile ha tenido una excelente labor, vacunando al menos dos veces a casi la totalidad de la población. En segundo lugar, se encuentra Perú, donde la cobertura de vacunación ha sido casi del 150%. El tercer puesto es para Brasil, donde la población ha sido vacunada al menos con una dosis.

*Ilustración 74: Sintaxis para la elaboración de un Grafico de Cobertura de Vacunación por País*

[illegible]

## Módulo 4

Ilustración 75: Comparación de Estrategias de Vacunación en América Latina



Por último, se comparó la tasa de mortalidad entre países, identificando que Perú fue el país con mayor un mayor porcentaje de fallecidos, seguido por México, Colombia y Brasil.

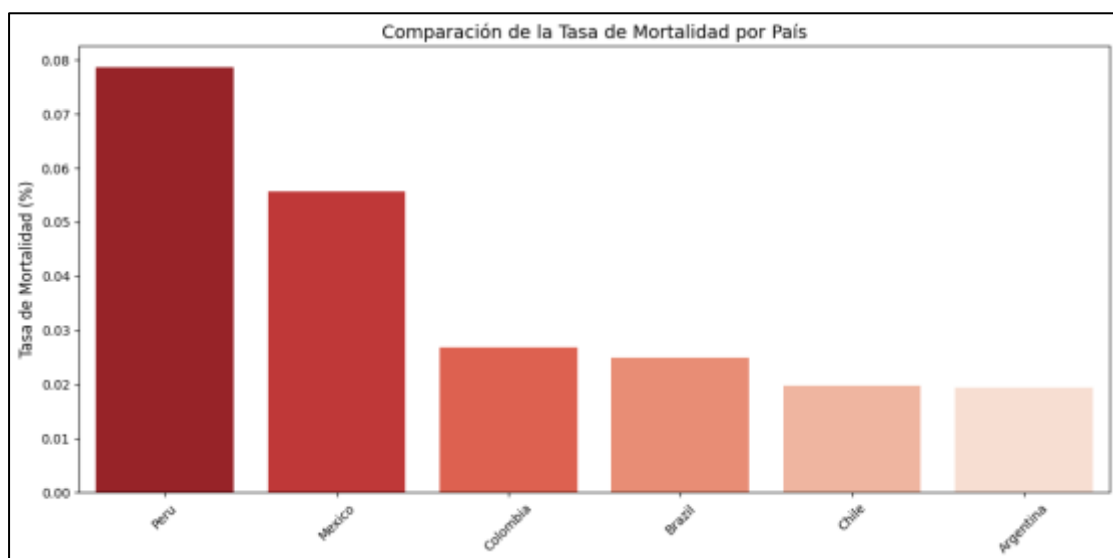
Ilustración 76: Sintaxis para la elaboración de un Grafico de Barras de tasas de mortalidad por país

```

mortality_rate_per_pais = df[['pais', 'mortality_rate']].groupby('pais').mean()
mortality_rate_per_pais = mortality_rate_per_pais.reset_index()
# Crear el gráfico de barras
plt.figure(figsize=(12, 8))
plt.bar(mortality_rate_per_pais['pais'], mortality_rate_per_pais['mortality_rate'])
# Configuración del gráfico
plt.title('Comparación de la Tasa de Mortalidad por País')
plt.xlabel('País')
plt.ylabel('Tasa de Mortalidad (%)')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()

```

Ilustración 77: Comparación de Tasas de Mortalidad por País



**Módulo 4**

Finalmente, se agregó también como extracredit, un mapa de calor interactivo que muestra tres capas de información relacionada con la pandemia de COVID-19: la distribución geográfica de casos confirmados, fallecimientos y dosis de vacuna administradas. Utilizando colores cálidos, el mapa permite visualizar las áreas con mayores concentraciones de estos datos. Es una herramienta útil para explorar las tendencias espaciales de la pandemia y la respuesta de vacunación.

*Ilustración 78: Sintaxis para la elaboración de un Mapa de Calor*

```
import folium
from folium.plugins import HeatMap
from folium import LayerControl

columns=['latitude', 'longitude', 'cumulative_confirmed', 'cumulative_deceased', 'cumulative_vaccine_doses_administered']
data_map= df_linpin[columns].dropna()

map= folium.Map(location=[-14.23584,-51.92527], zoom_start=3)

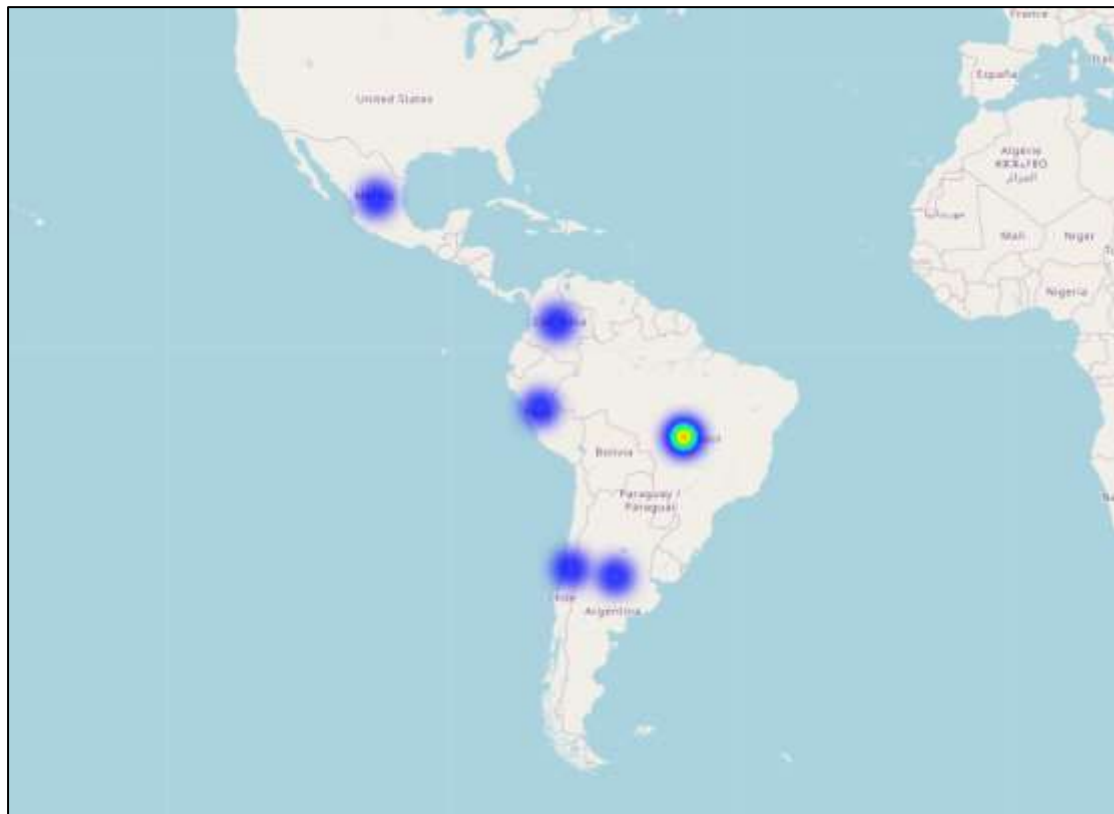
heat_map_confirmed= HeatMap(data=data_map[['latitude', 'longitude', 'cumulative_confirmed']], radius=15)
map.add_child(heat_map_confirmed)

heat_map_deceased= HeatMap(data=data_map[['latitude', 'longitude', 'cumulative_deceased']], radius=15)
map.add_child(heat_map_deceased)

heat_map_vaccine= HeatMap(data=data_map[['latitude', 'longitude', 'cumulative_vaccine_doses_administered']], radius=15)
map.add_child(heat_map_vaccine)

map.save('mapa.html')
map
```

*Ilustración 79: Mapa de Calor, Casos Confirmados por País*





**Módulo 4**

**EDA e insights**

El análisis exploratorio de datos (EDA) tuvo como objetivo descubrir patrones y tendencias para informar decisiones sobre la expansión de laboratorios farmacéuticos en Latinoamérica, considerando el impacto de COVID-19, las condiciones climáticas, y las estrategias de vacunación.

**1. Visualizaciones Generales y Correlación**

- Se generaron gráficos de barras comparativos entre países y una matriz de correlación, para examinar la relación entre variables.
- La matriz de correlación (con un filtro de 0.5) reveló pocas relaciones significativas con las variables clave (new\_confirmed y new\_deceased), salvo con variables poblacionales como population y area\_sq\_km. Este resultado sugiere que factores territoriales pueden influir en el impacto de COVID-19, pero otras variables de contexto deben considerarse para la expansión de infraestructura.

**2. Descripción Poblacional**

- Demografía y PBI per cápita: Gráficos de barras y apilados muestran a Brasil como el país más poblado (200M), seguido por México (110M). Chile lidera en PBI per cápita, lo que refleja una mayor capacidad económica para soportar una expansión.
- Distribución por edad y mortalidad: Se detallaron distribuciones etarias y tasas de mortalidad masculinas y femeninas. Esto es útil para entender la vulnerabilidad de las poblaciones y el posible consumo de servicios de salud por país.

**3. Condiciones Climáticas y COVID-19**

- Temperaturas y COVID-19: Análisis de temperaturas medias (mediante boxplots y scatter plots) sugieren una relación entre el clima y los casos de COVID-19 en países como Brasil y Perú, con temperaturas altas y casos elevados.
- Tratamiento de Outliers: Se recomendó identificar y manejar outliers de temperatura (ej. usando IQR o transformación logarítmica), ya que pueden sesgar el análisis.

**4. Tendencias Temporales y COVID-19**

- Casos y decesos a lo largo del tiempo: Se analizaron tendencias de casos y decesos por mes y por país. Tres picos importantes se observaron: junio 2021, enero 2022, y junio 2022, indicando los periodos críticos de la pandemia.
- Vacunación y su efecto: Un aumento de vacunas suministradas (900 millones de dosis totales) coincide con una reducción en decesos, mostrando que las estrategias de vacunación han sido efectivas.
- Autocorrelación y estacionalidad: Gráficos de autocorrelación sugieren patrones de estacionalidad y tendencia en los casos y decesos de

**Módulo 4**

COVID-19. Esta información puede ayudar a prever futuros brotes y gestionar la capacidad de los laboratorios.

**5. Estrategias de Vacunación**

- Comparación de dosis administradas: Brasil lidera en dosis totales (200 millones), mientras Chile presenta la mayor cobertura relativa (3.5 dosis por cada 100 personas). Esta variabilidad indica diferencias en la efectividad y velocidad de vacunación en cada país.
- Cobertura de vacunación: Chile y Perú tienen la mayor cobertura, habiendo vacunado a casi toda su población al menos una vez. Esto sugiere una infraestructura de vacunación fuerte, que puede influir en las decisiones de expansión.

**6. Tasa de Mortalidad por COVID-19**

- Perú reporta la tasa de mortalidad más alta, seguido por México, Colombia, y Brasil. Esto indica una vulnerabilidad mayor que podría justificar la necesidad de apoyo en salud en estos países.

**Insights Finales para Expansión de Laboratorios:**

- Países con altas poblaciones y baja relación de vacunación (ej. Brasil) presentan una oportunidad para la expansión de infraestructura de vacunación.
- Los análisis de temperatura sugieren que en climas más cálidos la propagación puede ser mayor, indicando una mayor demanda de servicios en estos países.
- Estrategias de vacunación variadas y una alta mortalidad en países como Perú y México resaltan la necesidad de reforzar servicios en zonas con baja capacidad de respuesta a pandemias futuras.

## Análisis del dashboard

El dashboard cuenta con una portada inicial con una imagen representativa de la institución y el logo de la misma. A su vez, existen dos botones de enlaces para desplazarse por el resto de las pestañas.



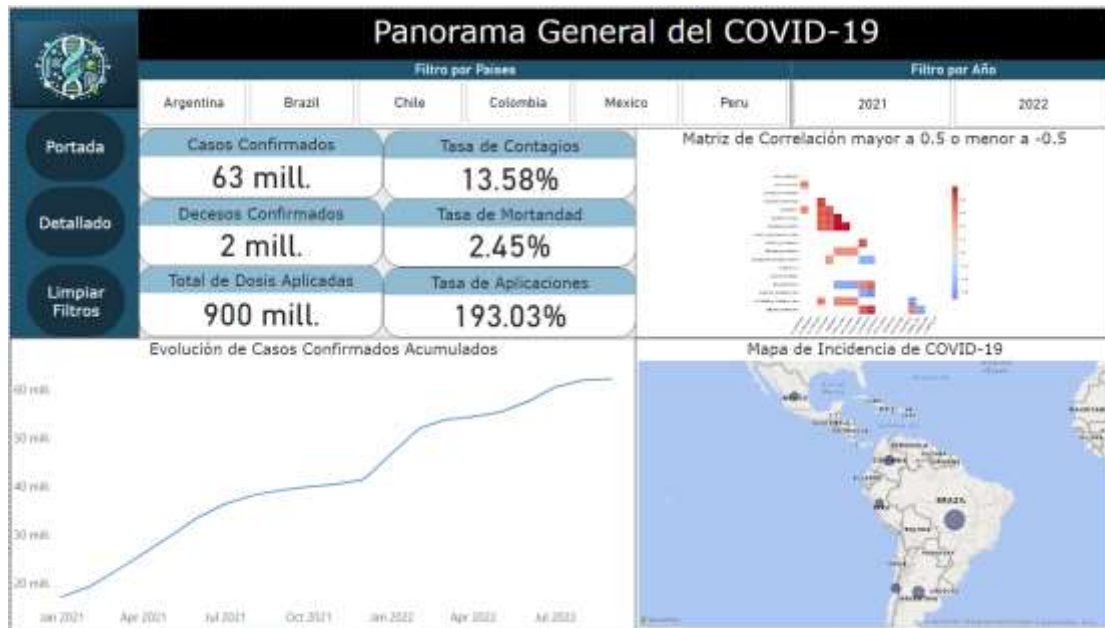
El segundo lienzo se corresponde con un dashboard con título, logo de la empresa, filtros por país y años y tres botones, dos útiles para desplazarse por el archivo y uno para limpiar los filtros aplicados.

Contiene insight generales obtenidos del dataframe: Casos Confirmados, Decesos Confirmados y Dosis Aplicadas, así como también métricas obtenidas a partir de código DAX: Tasa de Contagios, Tasa de Mortalidad y Tasa de Aplicaciones. También se encuentra la Matriz de Correlación: en este caso, con una correlación mayor a 0.5, la cual fue insertada con código de python.

También un Grafico Lineal de temporalidad que muestra los casos de Covid confirmados acumulados, y que se pueden desglosar por país.

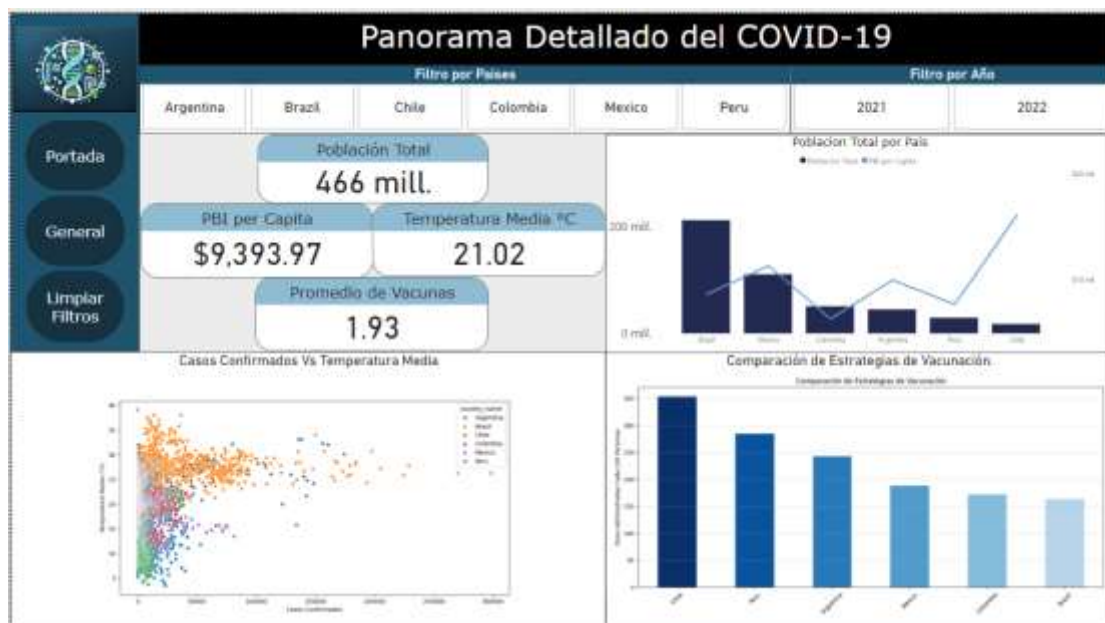
Finalmente, también un mapa de calor que muestra la representatividad de los casos (de acuerdo con el tamaño de la burbuja) en cada país.

## Módulo 4



Por último, así como el caso anterior, cuenta con un título, logo de la empresa, filtros y botones, pero también cuenta con insight específicos de cada país: Población total, PBI per cápita, Temperatura Media (grados C) y el promedio de vacunas distribuidas por habitantes. Así mismo, tiene un gráfico de barras que muestra la temperatura media (en grados) por país y el pbi per capita, graficado a partir de una línea.

Cuenta además con dos gráficos incorporados mediante código de python: el primero es un scatterplot que muestra la relación entre temperatura y casos confirmados y el segundo un gráfico de barras que comprara la estrategia de vacunación por países.



## **Conclusiones y Recomendaciones**

Como conclusiones, se decidió invertir en Brasil, México y Colombia, por varias razones:

1. Población y tamaño del mercado:
  - Brasil es el país más poblado de la región, con más de 200 millones de habitantes, lo que representa una gran oportunidad de mercado. Además, Brasil es crucial en Sudamérica.
  - México tiene alrededor de 110 millones de habitantes y es el segundo mercado más grande de la región, además de estar estratégicamente conectado con América del Norte, lo que le otorga una relevancia adicional en la cadena de suministro y expansión.
  - Colombia, con 50 millones de habitantes, es otro mercado importante en la región y ofrece una puerta de entrada hacia otros países de América Central y del Sur, consolidando su importancia estratégica.
2. Temperatura:

Como se mencionó se ha detectado una leve relación entre una temperatura elevada y el aumento de casos confirmados. En este sentido, las condiciones tropicales y templadas que presentan estos países en algunas regiones pueden influir en la incidencia de enfermedades.
3. Esfuerzos de vacunación y cobertura:
  - Estos tres países han enfrentado altos niveles de casos confirmados y mortalidad por COVID-19, lo cual genera una demanda sostenida de infraestructura y servicios de salud.
  - En términos de administración de vacunas, si bien Brasil lidera con el mayor número de dosis administradas promedio, su performance por persona fue de la peor de la región. Le siguieron Colombia y México, con lo cual se detecta una necesidad de mejorar la infraestructura de salud que permita ampliar el marco de cobertura.
4. Crecimiento económico y potencial de mercado:

En cuanto a PBI per Cápita se destacan México y Brasil, con 11 mil y 8 mil dólares respectivamente. Si bien no son los mejores en este sentido, son cifras relativamente buenas. Estos países son atractivos para inversiones en infraestructura de salud, ya que sus economías más fuertes pueden apoyar y sostener operaciones de laboratorio y centros de vacunación.

## **Reflexión personal**

Me gusto el proyecto, la cantidad de datos disponibles en el Dataset me permitió explorar y analizar un montón de información. Todo el proceso fue entretenido, desde la carga y puesta a punto de la información, pasando por el creación e interpretación de diversos gráficos, hasta el armado del dashboard y la elaboración de métricas para arribar a conclusiones útiles. Se le complico un poco a mi computadora trabajar con este volumen de datos pero por suerte no fue nada grave.