DATOS MUNDIALES

SQL + Python

Mariana Carmona

Tabla de contenido

Proyecto: Datos Mundiales

Descripción del proyecto

El proyecto "Datos Mundiales" se desarrollará en tres partes principales. La primera parte se enfocará en la creación y carga de la base de datos. La base de datos, denominada `world`, consta de tres tablas: `city`, `country` y `countrylanguage`. La creación de la base de datos incluirá la definición de los esquemas y las relaciones entre las tablas. Se implementará en dos entornos distintos: en la nube utilizando Azure SQL Server y de manera local utilizando MySQL Workbench. Una vez creadas las tablas, se procederá a importar los datos correspondientes.

En la segunda parte del proyecto, se establecerá una conexión con la base de datos utilizando Python. Para conectarse con Azure SQL Server, se empleará la biblioteca `sqlalchemy`, mientras que para MySQL Workbench se utilizará `mysql-connector-python`. Desde Python, se realizarán consultas SQL para responder a preguntas específicas sobre los datos, también se contestarán las mismas preguntas utilizando DataFrames de Pandas. Entre las consultas posibles se incluyen los nombres y las áreas de superficie de los cinco países más grandes del mundo, la población total de todos los países de cada continente, el nombre de las ciudades y la población de todos los países de Europa, etc.

La tercera parte del proyecto se centrará en la visualización de los datos. Se crearán visualizaciones en Python, se utilizarán bibliotecas como `matplotlib` y `seaborn`, creando gráficos que representen los datos obtenidos de las consultas. Por ejemplo, se podrán generar gráficos de barras para mostrar la población de cada continente o gráficos de pastel para representar el número de países por continente

El proyecto "Datos Mundiales" abarcará desde la creación y carga de una base de datos hasta la realización de consultas y la visualización de los resultados mediante diversas herramientas.

→ Ver video del proyecto

https://youtu.be/alZgVbFXyMQ

→ Ver repositorio del proyecto:

https://github.com/MarianaDwarka/Datos_Mundiales

Primera parte: Creación y carga de la base de datos

Se trabajará con una base de datos llamada world que tiene tres tablas: city, country y countrylanguage, con las siguientes características:

Columna	Tipo	Descripción
ID	int	Identificador único de la ciudad.
Name	char(35)	Nombre de la ciudad.
CountryCode	char(3)	Código del país (de la tabla `country`).
District	char(20)	Distrito o región a la que pertenece la ciudad.
Population	int	Población de la ciudad.

Columna	Tipo	Descripción
Code	char(3)	Código único del país.
Name	char(52)	Nombre del país.
Continent	enum	Continente al que pertenece el país (Asia, Europe, North America, etc.).
Region	char(26)	Región dentro del continente.
SurfaceArea	decimal(10,2)	Área de superficie del país.
IndepYear	smallint	Año de independencia.
Population	int	Población del país.
LifeExpectancy	decimal(3,1)	Esperanza de vida.
GNP	decimal(10,2)	Producto Nacional Bruto.
GNPOld	decimal(10,2)	Producto Nacional Bruto anterior.
LocalName	char(45)	Nombre local del país.
GovernmentForm	char(45)	Forma de gobierno.
HeadOfState	char(60)	Jefe de estado.
Capital	int	Identificador de la capital (relación con la tabla `city`).
Code2	char(2)	Código adicional del país.

Columna	Tipo	Descripción
CountryCode	char(3)	Código del país (relación con la tabla `country`).
Language	char(30)	Idioma hablado en el país.
IsOfficial	enum	Indica si es un idioma oficial en el país ('T' para true, 'F' para false).
Percentage	decimal(4,1)	Porcentaje de personas que hablan el idioma en el país.

Creación de la Base de datos en SQL Server

Paso 1: Creación de la base de datos (Ver ConexionAzure.ipynb)

La creación de la base de datos en Azure SQL Server inicia con el inicio de sesión en Azure, luego, se crea un grupo de recursos, se configura un servidor SQL y también una base de datos llamada “world”, también se establece una regla de firewall para permitir conexiones al servidor.

El código comienza con la autenticación en Azure mediante el comando `az login`. Una vez autenticado, se muestran los detalles de la cuenta de Azure activa en formato de tabla utilizando el comando `az account show --output table`. Posteriormente, se lista todos los grupos de recursos existentes en la cuenta con `az group list`.

Luego, se crea un nuevo grupo de recursos llamado `mariana_dwarka` en la región Este de Estados Unidos (`eastus`) mediante el comando `az group create --location eastus --name mariana_dwarka`. Para verificar que el grupo de recursos se haya creado correctamente, se vuelve a listar todos los grupos de recursos con `az group list`.

El siguiente paso es crear un servidor de SQL en Azure llamado `ServProyectoFinal`, ubicado en la región Centro de México (`mexicocentral`). Este servidor se crea dentro del grupo de recursos `mariana_dwarka` y utiliza `mainexam` como nombre de usuario del administrador y `Abcd1234` como contraseña. Esto se logra con el comando `az sql server create --resource-group mariana_dwarka --name ServProyectoFinal --location mexicocentral --admin-user mainexam --admin-password Abcd1234`.

Para permitir conexiones al servidor SQL desde cualquier dirección IP, se establece una regla de firewall llamada `AllowAll` que abarca desde la dirección IP `0.0.0.0` hasta `255.255.255.255`. Este paso se realiza con el comando `az sql server firewall-rule create --resource-group mariana_dwarka --server ServProyectoFinal --name AllowAll --start-ip-address 0.0.0.0 --end-ip-address 255.255.255.255`.

Finalmente, se crea una base de datos llamada `world` en el servidor SQL `ServProyectoFinal`, dentro del grupo de recursos `mariana_dwarka`. Esta base de datos no es redundante por zonas, lo que significa que no se replica automáticamente en múltiples zonas de disponibilidad. El comando utilizado

para este paso es `az sql db create --resource-group mariana_dwarka --server ServProyectoFinal --name world --zone-redundant false`.

Paso 2: Carga de datos a la base de datos (Ver SQLServerAzure_CargaDatos.ipynb)

Este proceso permite preparar el entorno, definir y crear tablas en una base de datos de Azure SQL Server, y cargar datos desde archivos CSV en las tablas creadas. Así que, comenzamos con la preparación del entorno, esto se hace porque se utiliza un Notebook de Google Colab y requerimos conectarnos con SQL Server.

Primero, es necesario instalar las librerías `sqlalchemy` y `pyodbc` usando `pip`. Esto se hace con el comando `!pip install sqlalchemy pyodbc`.

Luego, en un sistema operativo Ubuntu, se procede a actualizar la lista de paquetes e instalar los prerrequisitos necesarios, incluyendo `curl` y `apt-transport-https`. Después, se descargan las claves GPG del repositorio de Microsoft y se registra este repositorio para la distribución específica de Ubuntu. Esto permite instalar el controlador ODBC de Microsoft (`msodbcsql17`), que es esencial para conectar con SQL Server. Opcionalmente, se instalan los encabezados de desarrollo de `unixODBC`.

Una vez que el entorno está preparado, se configura la conexión a la base de datos de Azure SQL Server usando SQLAlchemy. Se define una cadena de conexión que incluye las credenciales del usuario administrador y la URL del servidor SQL.

Con SQLAlchemy, se crea un objeto `MetaData` que almacenará la información sobre las tablas. Se definen tres tablas: `country`, `city`, y `countrylanguage`, especificando los nombres de las columnas, sus tipos de datos, y las restricciones (como claves primarias y claves foráneas).

Después de definir las tablas, se ejecuta `metadata.create_all(engine)` para crear las tablas en la base de datos.

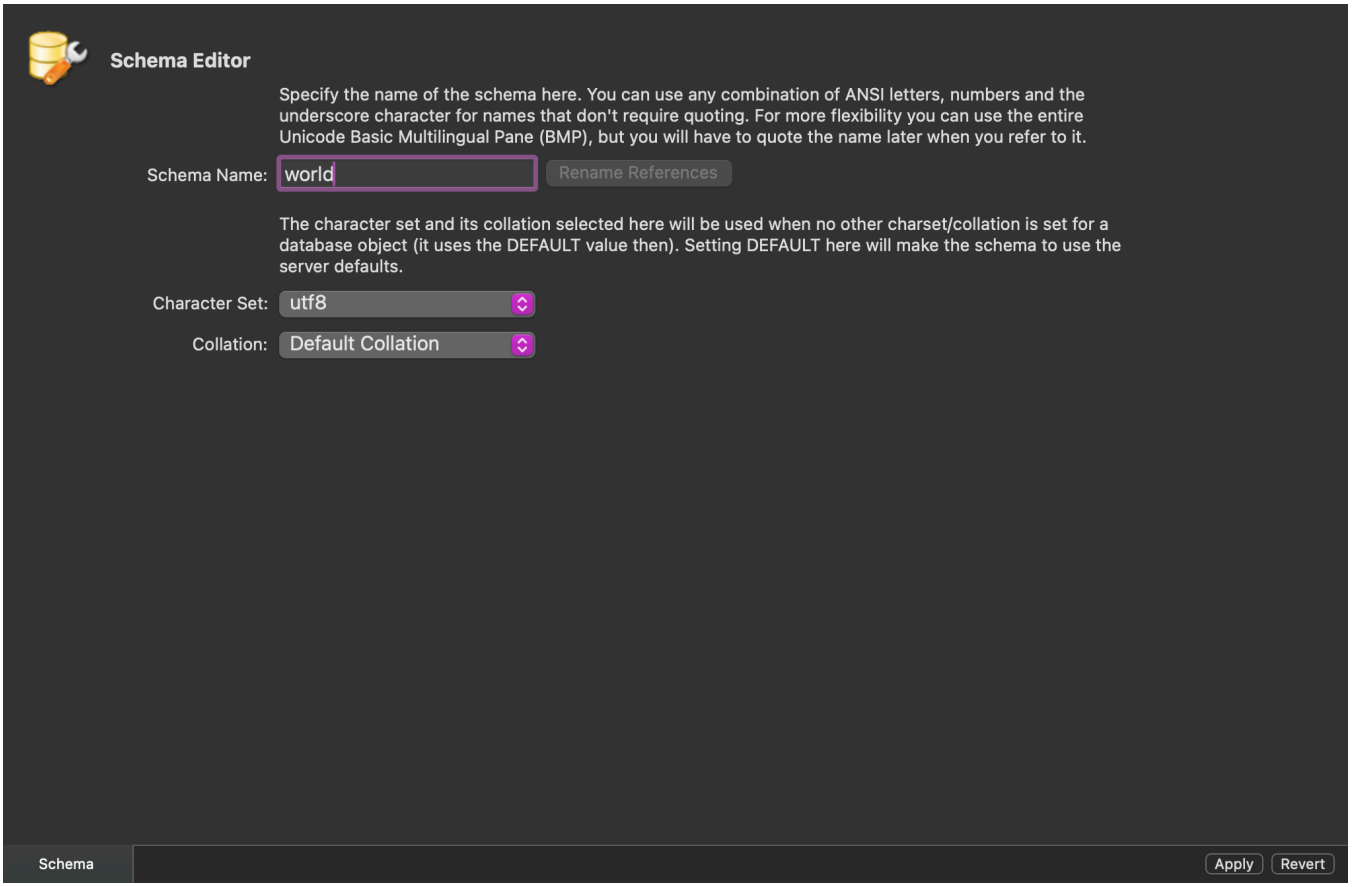
Para trabajar con los datos, se importa la librería `pandas` y se define una función `fetch_data` que permite ejecutar consultas SQL y convertir los resultados en DataFrames de pandas. Se verifica que las tablas se han creado correctamente ejecutando una consulta a `information_schema.tables`.

Finalmente, se cargan los datos en las tablas desde archivos CSV. Se monta Google Drive para acceder a los archivos CSV almacenados allí. Se crea una sesión con SQLAlchemy para manejar las transacciones. Se define una función `insert_data_from_csv` que lee los datos de un archivo CSV y los inserta en la tabla correspondiente, manejando posibles valores `NULL`.

Se intenta insertar los datos en cada tabla (`country`, `city`, `countrylanguage`) dentro de una transacción. Si ocurre algún error, la transacción se revierte para mantener la integridad de los datos. Al finalizar, se cierra la sesión.

Creación de la base de datos en MySQL

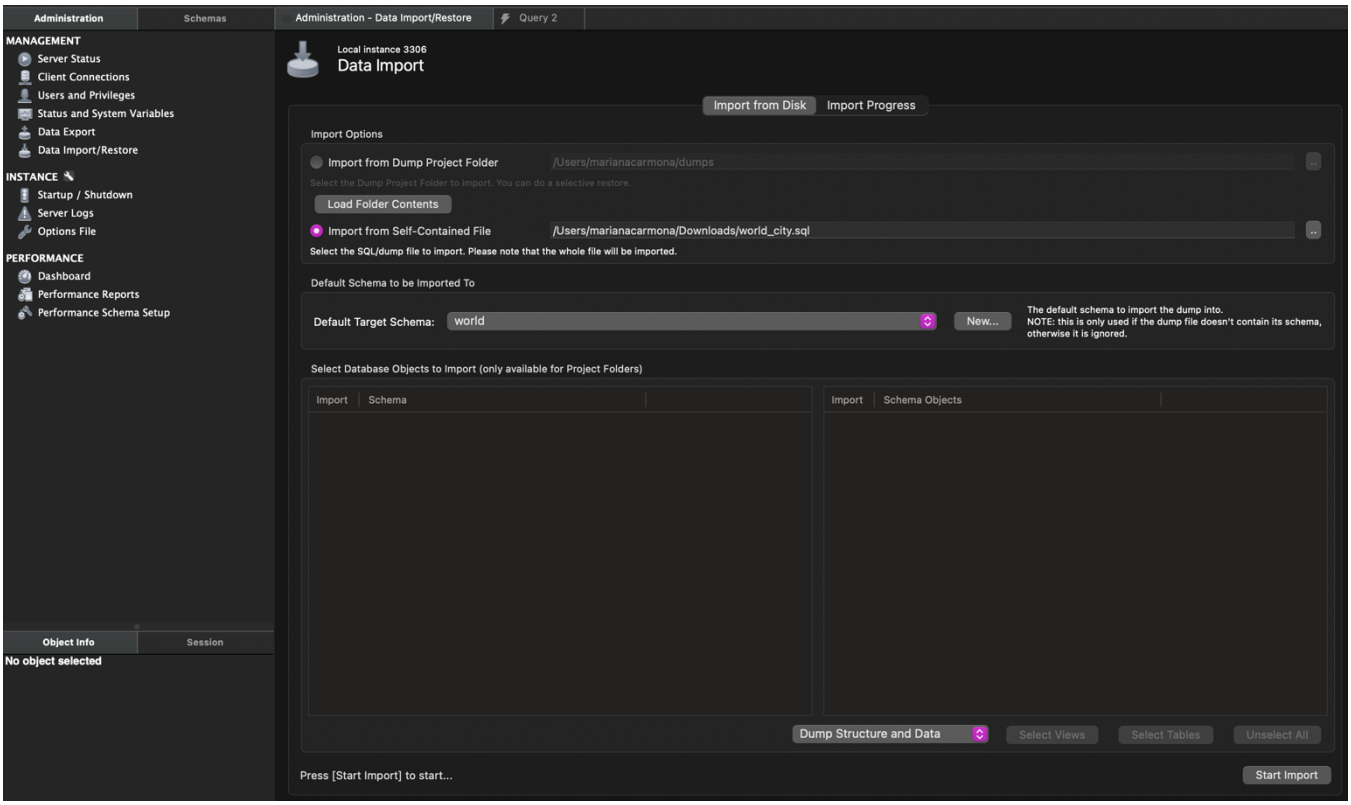
Para el manejo de MySQL se utilizará Workbench, el primer paso es la creación del esquema llamado world.



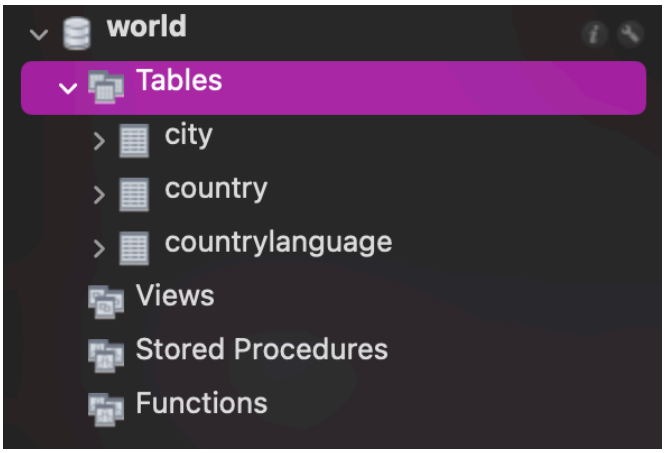
Para la creación de las tablas y la importación de los datos por tabla, se utilizarán los scripts proporcionados:

- world_city.sql
- world_country.sql
- world_countrylanguage.sql

El proceso será a través del asistente de importación de Worlbench.



Este proceso de importación se hará para cada una de las tres tablas, quedando este resultado:



Segunda parte: Consulta en SQL para responder preguntas

Conexión a la base de datos “world”

Se establecerá una conexión con la base de datos utilizando Python. Para conectarse con Azure SQL Server, se empleará la biblioteca `sqlalchemy`, mientras que para MySQL se utilizará `mysql-connector-python`.

Ambos procesos consisten en importar las librerías necesarias, configurar la conexión a la base de datos con las credenciales adecuadas, ejecutar una consulta SQL para obtener datos específicos, cargar los resultados en un DataFrame de pandas y mostrar el DataFrame. La principal diferencia radica en la forma en que se establece la conexión y se ejecutan las consultas debido a las distintas librerías utilizadas para SQL Server y MySQL.

Conexión a la base de datos en Azure SQL Server (sqlalchemy)

Para la conexión a la base de datos de Azure SQL Server, primero se importan las librerías necesarias: `create_engine` de `sqlalchemy`, así como `pandas` bajo el alias `pd`. Se configura la conexión utilizando la función `create_engine` de SQLAlchemy, donde se especifica la cadena de conexión que incluye el controlador ODBC, el nombre de usuario, la contraseña, y la dirección del servidor SQL de Azure. Una vez establecida la conexión, se define una consulta SQL. Esta consulta se ejecuta y los resultados se cargan en un DataFrame de pandas mediante la función `read_sql`. Finalmente, el DataFrame se muestra, presentando los datos obtenidos de la consulta.

```
[ ]  from sqlalchemy import create_engine, text
     import pandas as pd
```

```
[ ]  # Configura la conexión con SQLAlchemy
     engine = create_engine('mssql+pyodbc://mainexam:Abcd1234@servproyectofinal.database.windows.net/world?driver=ODBC+Driver+17+for+SQL+Server')
```

```
[ ]  # Ejercicio 1: Consulta para mostrar el nombre y la población de todos los países del continente europeo
     query = 'SELECT Name, Population FROM country WHERE Continent = \'Europe\''

     # Ejecutar la consulta y cargar los resultados en un DataFrame de pandas
     df1 = pd.read_sql(query, engine)

     # Mostrar el DataFrame
     df1
```

Conexión a la base de datos en MySQL (mysql-connector-python)

En el caso de la conexión a la base de datos MySQL, se importan las librerías `mysql.connector` y `pandas`. Se configura la conexión creando un diccionario `config` que contiene las credenciales de la base de datos: el usuario, la contraseña, el host y el nombre de la base de datos. Utilizando `mysql.connector.connect`, se establece la conexión con los parámetros especificados en `config` y se crea un cursor para ejecutar consultas. Se define una consulta SQL similar a la utilizada para SQL Server. La consulta se ejecuta con el cursor y los resultados se almacenan en una variable. Se obtienen los nombres de las columnas de los resultados y se crea un DataFrame de pandas con los datos obtenidos. Finalmente, el DataFrame se muestra para presentar los resultados de la consulta.

```
import mysql.connector
import pandas as pd

# Configura la conexión a tu base de datos
config = {
    'user': 'root',
    'password': '12345abcd',
    'host': 'localhost',
    'database': 'world'
}

# Establece la conexión a la base de datos
conn = mysql.connector.connect(**config)
cursor = conn.cursor()
```

```
query1 = """
SELECT Name, Population
FROM country
WHERE Continent = 'Europe';
"""

cursor.execute(query1)
resultados1 = cursor.fetchall()

# Obtén los nombres de las columnas
column_names = [desc[0] for desc in cursor.description]

# Crea el DataFrame
df1 = pd.DataFrame(resultados1, columns=column_names)

# Muestra el DataFrame
df1
```

Solución a las preguntas (Ver SQL Server_Preguntas.ipynb, MySQL_Preguntas.ipynb, MySQL_Preguntas_Pandas.ipynb)

Como se mencionó anteriormente, la principal diferencia entre usar SQL Server y MySQL radica en la forma en que se establece la conexión y se ejecutan las consultas debido a las distintas librerías utilizadas para SQL Server y MySQL. Pero como se puede ver, la query es la misma salvo pequeñas variaciones, así que en lo que resta de esta sección solo se estará escribiendo el código en SQL y también se mencionará su equivalencia en caso de quererlo hacer directamente con un DataFrame de Pandas. La solución se puede ver desglosada en 4 notebooks.

- SQLServer_Preguntas.ipynb – Se contestan las preguntas mediante queries SQL
- MySQL_Preguntas.ipynb – Se contestan las preguntas mediante queries SQL
- MySQL_Preguntas_Pandas.ipynb – Se contestan las preguntas mediante manipulación de DataFrames de Pandas

Preguntas:

Ejercicio 1: Mostrar el nombre y la población de todos los países del continente europeo.

SQL:

```
SELECT Name, Population
FROM country
WHERE Continent = 'Europe';
```

Pandas:

```
ejercicio_1 = country[country['Continent'] == 'Europe'][['Name', 'Population']]
```

	Name	Population
4	Albania	3401200
5	Andorra	78000
15	Austria	8091800
18	Belgium	10239000
22	Bulgaria	8190900
25	Bosnia and Herzegovina	3972000
26	Belarus	10236000
39	Switzerland	7160400
55	Czech Republic	10278100
56	Germany	82164700
59	Denmark	5330000

Ejercicio 2: Mostrar los nombres y las áreas de superficie de los cinco países más grandes del mundo (en términos de área de superficie).

SQL:

```
SELECT Name, SurfaceArea
FROM country
ORDER BY SurfaceArea DESC
LIMIT 5;
```

Pandas:

```
top5_countries = country[['Name', 'SurfaceArea']].sort_values(by='SurfaceArea',
ascending=False).head(5)
```

	Name	SurfaceArea
0	Russian Federation	17075400.00
1	Antarctica	13120000.00
2	Canada	9970610.00
3	China	9572900.00
4	United States	9363520.00

Ejercicio 3: Calcular la población total de todos los países de cada continente y mostrar el resultado junto con el nombre del continente.

SQL:

```
SELECT Continent, SUM(Population) as TotalPopulation
FROM country
GROUP BY Continent;
```

	Continent	TotalPopulation
0	North America	482993000
1	Asia	3705025700
2	Africa	784475000
3	Europe	730074600
4	South America	345780000
5	Oceania	30401150
6	Antarctica	0

Pandas:

continent_population = country.groupby('Continent')['Population'].sum().reset_index()

Ejercicio 4: Mostrar el nombre de las ciudades y la población de todos los países de Europa, ordenados por población de la ciudad de manera descendente.

SQL:

SELECT city.Name as CityName, city.Population as CityPopulation
FROM city
JOIN country ON city.CountryCode = country.Code
WHERE country.Continent = 'Europe'
ORDER BY city.Population DESC;

	CityName	CityPopulation
0	Moscow	8389200
1	London	7285000
2	St Petersburg	4694000
3	Berlin	3386667
4	Madrid	2879052
...	...	...
836	Serravalle	4802
837	San Marino	2294
838	Longyearbyen	1438
839	Monaco-Ville	1234
840	Città del Vaticano	455

841 rows × 2 columns

Pandas:

# Realizar el join de los DataFrames 'city' y 'country' en la columna 'CountryCode' y 'Code' respectivamente
merged_df = pd.merge(city, country, left_on='CountryCode', right_on='Code').rename(columns={'Name_x': 'Name', 'Population_x': 'Population'})

# Filtrar el DataFrame para obtener solo las ciudades de Europa
european_cities = merged_df[merged_df['Continent'] == 'Europe']

# Seleccionar las columnas necesarias y ordenar por población en orden descendente
result_df = european_cities[['Name', 'Population']].rename(columns={'Name': 'CityName', 'Population': 'CityPopulation'}).sort_values(by='CityPopulation', ascending=False)

Ejercicio 5: Actualiza la población de China (código de país 'CHN') a 1500000000 (1.5 mil millones).

SQL:

	Name	Population
0	China	1500000000

UPDATE country
SET Population = 1500000000
WHERE Code = 'CHN';

Pandas:

country.loc[country['Code'] == 'CHN', 'Population'] = 1500000000

Preguntas extras:

Pregunta 1: ¿Cuáles son los países con más ciudades en la base de datos?

SQL:

```
SELECT country.Name AS CountryName, COUNT(city.ID) AS CityCount
FROM country
JOIN city ON country.Code = city.CountryCode
GROUP BY country.Name
ORDER BY CityCount DESC;
```

	CountryName	CityCount
0	China	363
1	India	341
2	United States	274
3	Brazil	250
4	Japan	248
...	...	...
227	Virgin Islands, British	1
228	Virgin Islands, U.S.	1
229	Vanuatu	1
230	Wallis and Futuna	1
231	Samoa	1

232 rows × 2 columns

Pandas:

```
city_count = city.groupby('CountryCode').size().reset_index(name='CityCount')
city_count = city_count.merge(country[['Code', 'Name']], left_on='CountryCode',
right_on='Code').drop(columns=['Code'])
top10_countries = city_count.sort_values(by='CityCount', ascending=False).head(10)
```

Pregunta 2: ¿Qué país tiene la mayor diversidad lingüística en términos de cantidad de idiomas hablados?

SQL:

```
SELECT country.Name AS CountryName, COUNT(countrylanguage.Language) AS LanguageCount
FROM country
JOIN countrylanguage ON country.Code = countrylanguage.CountryCode
GROUP BY country.Name
ORDER BY LanguageCount DESC;
```

	CountryName	LanguageCount
0	Canada	12
1	China	12
2	India	12
3	Russian Federation	12
4	United States	12
...	...	...
228	Turks and Caicos Islands	1
229	United States Minor Outlying Islands	1
230	Uruguay	1
231	Holy See (Vatican City State)	1
232	Virgin Islands, British	1

233 rows × 2 columns

Pandas:

```
# Realizar el join de los DataFrames 'country' y 'countrylanguage' en las columnas 'Code' y 'CountryCode' respectivamente
merged_df = pd.merge(country, countrylanguage, left_on='Code', right_on='CountryCode')

# Agrupar por el nombre del país y contar los idiomas
language_count_df = merged_df.groupby('Name').size().reset_index(name='LanguageCount')

# Ordenar por el conteo de idiomas en orden descendente
result_df = language_count_df.sort_values(by='LanguageCount', ascending=False).rename(columns={'Name': 'CountryName'})
```

Pregunta 3: ¿Cuáles son las 10 ciudades más pobladas del mundo?

SQL:

```
SELECT city.Name AS CityName, city.Population
FROM city
ORDER BY city.Population DESC
LIMIT 10;
```

	CityName	Population
0	Mumbai (Bombay)	10500000
1	Seoul	9981619
2	São Paulo	9968485
3	Shanghai	9696300
4	Jakarta	9604900
5	Karachi	9269265
6	Istanbul	8787958
7	Ciudad de México	8591309
8	Moscow	8389200
9	New York	8008278

Pandas:

```
top10_cities = city[['Name', 'Population']].sort_values(by='Population', ascending=False).head(10)
```

Pregunta 4: ¿Qué países tienen una expectativa de vida mayor a 80 años?

SQL:

```
SELECT Name, LifeExpectancy
FROM country
WHERE LifeExpectancy > 80;
```

	Name	LifeExpectancy
0	Andorra	83.5
1	Japan	80.7
2	Macao	81.6
3	Singapore	80.1
4	San Marino	81.1

Pandas:

```
high_life_expectancy = country[country['LifeExpectancy'] > 80][['Name', 'LifeExpectancy']]
```

Pregunta 5: ¿Cuáles son los continentes con la mayor cantidad de países?

SQL:

```
SELECT Continent, COUNT(Name) AS CountryCount
FROM country
GROUP BY Continent
ORDER BY CountryCount DESC;
```

	Continent	CountryCount
0	Africa	58
1	Asia	51
2	Europe	46
3	North America	37
4	Oceania	28
5	South America	14
6	Antarctica	5

Pandas:

```
continent_country_count = country.groupby('Continent').size().reset_index(name='CountryCount')
```

Pregunta 6: ¿Qué continentes tienen la mayor y menor densidad poblacional, y cuál es esa densidad?

SQL:

```
SELECT Continent, SUM(Population) / SUM(SurfaceArea) AS PopulationDensity
FROM country
GROUP BY Continent
ORDER BY PopulationDensity DESC;
```

	Continent	PopulationDensity
0	Asia	123.1915
1	Europe	31.6747
2	Africa	25.9327
3	North America	19.9465
4	South America	19.3552
5	Oceania	3.5498
6	Antarctica	0.0000

Pandas:

```
continent_density = country.groupby('Continent').apply(lambda x: round(x['Population'].sum() / x['SurfaceArea'].sum(),4)).reset_index(name='PopulationDensity')
```

Pregunta 7: ¿Cuál son los 3 idiomás más hablados en el mundo (considerando solo los datos disponibles)?

SQL:

```
SELECT Language, SUM(country.Population * countrylanguage.Percentage / 100) AS TotalSpeakers
FROM country
JOIN countrylanguage ON country.Code = countrylanguage.CountryCode
GROUP BY Language
ORDER BY TotalSpeakers DESC
LIMIT 3;
```

	Language	TotalSpeakers
0	Chinese	1396490179.00000
1	Hindi	405633070.00000
2	Spanish	355029462.00000

Pandas:

```
language_speakers = country.merge(countrylanguage, left_on='Code', right_on='CountryCode')
language_speakers['TotalSpeakers'] = language_speakers['Population'] *
language_speakers['Percentage'] / 100
language_speakers = language_speakers.groupby('Language')['TotalSpeakers'].sum().reset_index()
top3_languages = language_speakers.sort_values(by='TotalSpeakers', ascending=False).head(3)
```

Pregunta 8: ¿Cuáles son los 3 países que tienen el mayor Producto Interno Bruto (PIB)?

SQL:

```
SELECT Name, GNP
FROM country
ORDER BY GNP DESC
LIMIT 3;
```

	Name	GNP
0	United States	8510700.00
1	Japan	3787042.00
2	Germany	2133367.00

Pandas:

```
top3_gnp = country[['Name', 'GNP']].sort_values(by='GNP', ascending=False).head(3)
```

Pregunta 9: ¿Cuál es la proporción de habitantes que viven en la capital de cada país respecto a su población total?

SQL:

```
SELECT country.Name AS CountryName,
    city.Name AS CapitalName,
    city.Population AS CapitalPopulation,
    country.Population AS CountryPopulation,
    (city.Population / country.Population) * 100 AS CapitalPopulationPercentage
FROM country
JOIN city ON country.Capital = city.ID
ORDER BY CapitalPopulationPercentage DESC;
```

	CountryName	CapitalName	CapitalPopulation	CountryPopulation	CapitalPopulationPercentage
0	Singapore	Singapore	4017733	3567000	112.6362
1	Gibraltar	Gibraltar	27025	25000	108.1000
2	Macao	Macao	437500	473000	92.4947
3	Pitcairn	Adamstown	42	50	84.0000
4	Saint Pierre and Miquelon	Saint-Pierre	5808	7000	82.9714
...	...	...	...	...	...
227	China	Peking	7472000	1500000000	0.4981
228	Pakistan	Islamabad	524500	156483000	0.3352
229	Nigeria	Abuja	350100	111506000	0.3140
230	United States	Washington	572059	278357000	0.2055
231	India	New Delhi	301297	1013662000	0.0297

232 rows × 5 columns

Pandas:

```
merged_data = country.merge(city, left_on='Capital', right_on='ID', suffixes=('_country', '_city'))
merged_data['CapitalPopulationPercentage'] = (merged_data['Population_city'] /
merged_data['Population_country']) * 100

# Ordenar los datos por CapitalPopulationPercentage y seleccionar los 10 primeros
top10_data = merged_data.sort_values(by='CapitalPopulationPercentage',
ascending=False)[['Name_country', 'Name_city','Population_country', 'Population_city',
'CapitalPopulationPercentage']]
```

Pregunta 10: ¿Cuáles son los países con la mayor superficie terrestre en cada continente?

SQL:

```
SELECT country.Name AS CountryName,
    city.Name AS CapitalName,
    city.Population AS CapitalPopulation,
    country.Population AS CountryPopulation,
    (city.Population / country.Population) * 100 AS CapitalPopulationPercentage
FROM country
JOIN city ON country.Capital = city.ID
ORDER BY CapitalPopulationPercentage DESC;
```

	Continent	CountryName	SurfaceArea
0	Europe	Russian Federation	17075400.00
1	Antarctica	Antarctica	13120000.00
2	North America	Canada	9970610.00
3	Asia	China	9572900.00
4	South America	Brazil	8547403.00
5	Oceania	Australia	7741220.00
6	Africa	Sudan	2505813.00

Pandas:

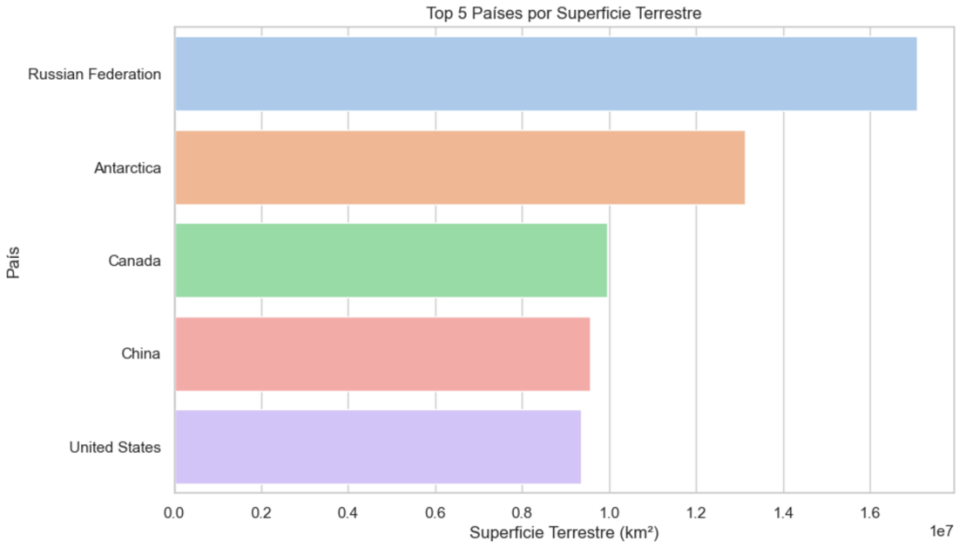
```
largest_countries =
country.loc[country.groupby('Continent')['SurfaceArea'].idxmax()].reset_index(drop=True)[['Continent',
'Name', 'SurfaceArea']]
```

Tercera parte: Visualizaciones en Python

Visualizaciones Seaborn y Matplotlib (Ver MySQL_Visualizaciones.ipynb)

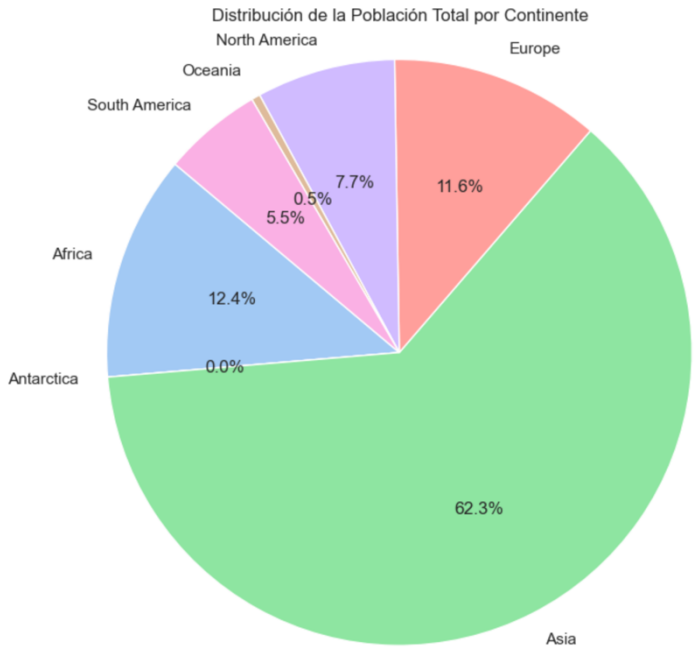
Preguntas:

Ejercicio 2: Mostrar los nombres y las áreas de superficie de los cinco países más grandes del mundo (en términos de área de superficie).



	Name	SurfaceArea
0	Russian Federation	17075400.00
1	Antarctica	13120000.00
2	Canada	9970610.00
3	China	9572900.00
4	United States	9363520.00

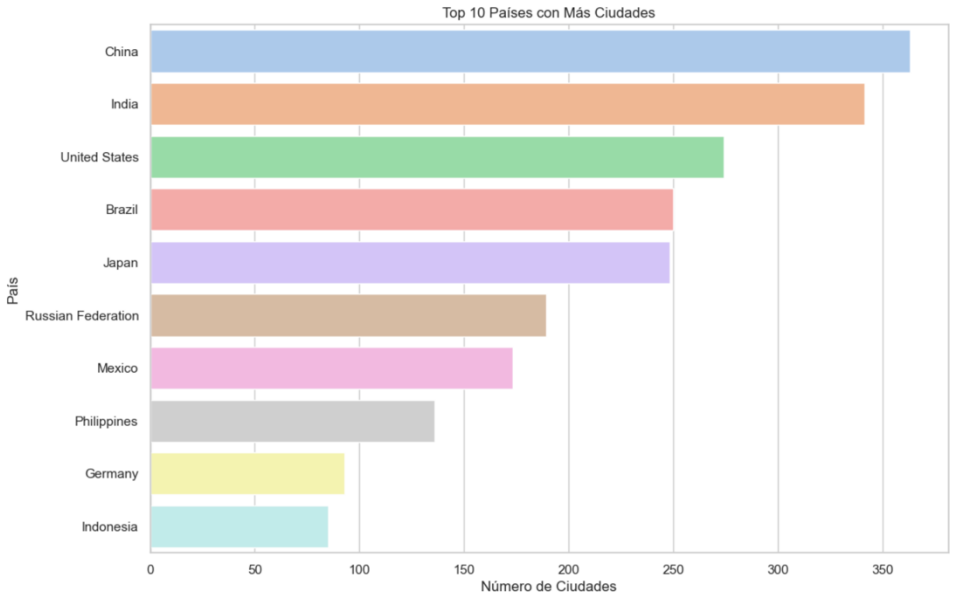
Ejercicio 3: Calcular la población total de todos los países de cada continente y mostrar el resultado junto con el nombre del continente.



	Continent	TotalPopulation
0	North America	482993000
1	Asia	3705025700
2	Africa	784475000
3	Europe	730074600
4	South America	345780000
5	Oceania	30401150
6	Antarctica	0

Preguntas extras:

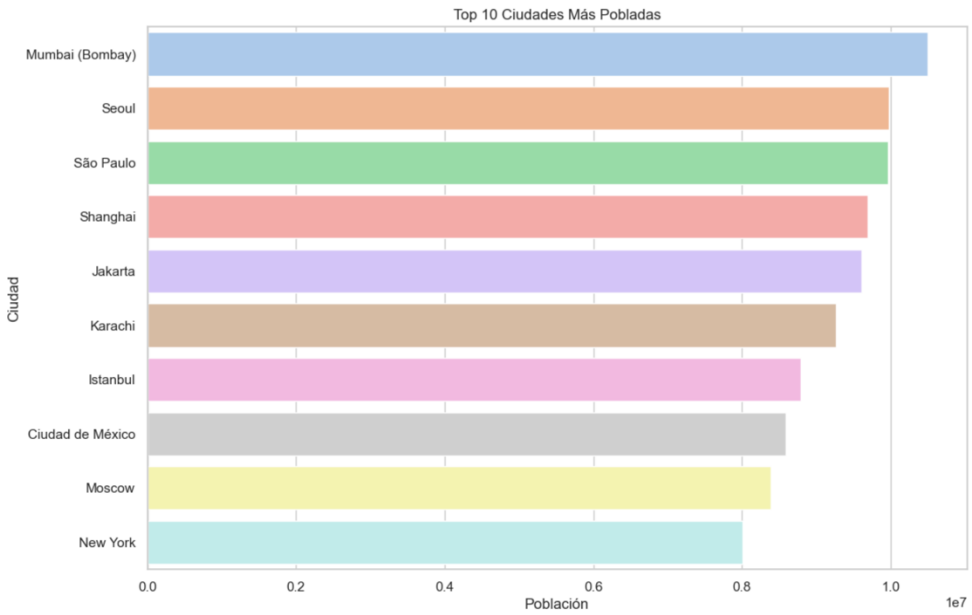
Pregunta 1: ¿Cuáles son los países con más ciudades en la base de datos?



	CountryName	CityCount
0	China	363
1	India	341
2	United States	274
3	Brazil	250
4	Japan	248
...	...	...
227	Virgin Islands, British	1
228	Virgin Islands, U.S.	1
229	Vanuatu	1
230	Wallis and Futuna	1
231	Samoa	1

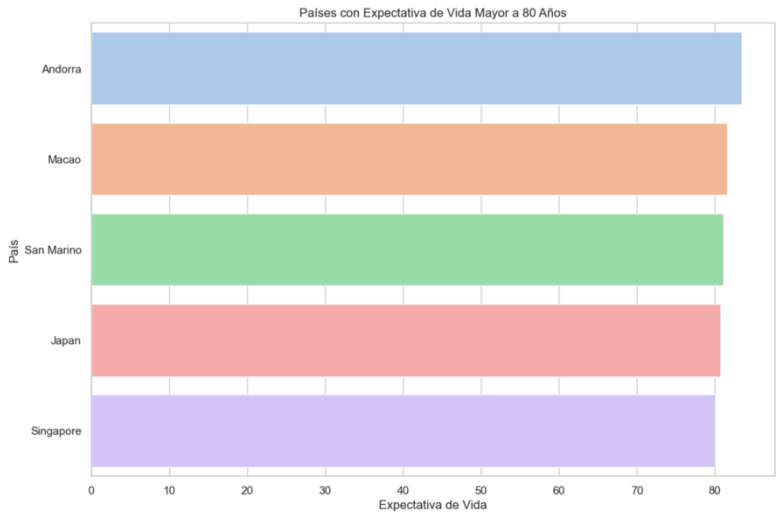
232 rows × 2 columns

Pregunta 3: ¿Cuáles son las 10 ciudades más pobladas del mundo?



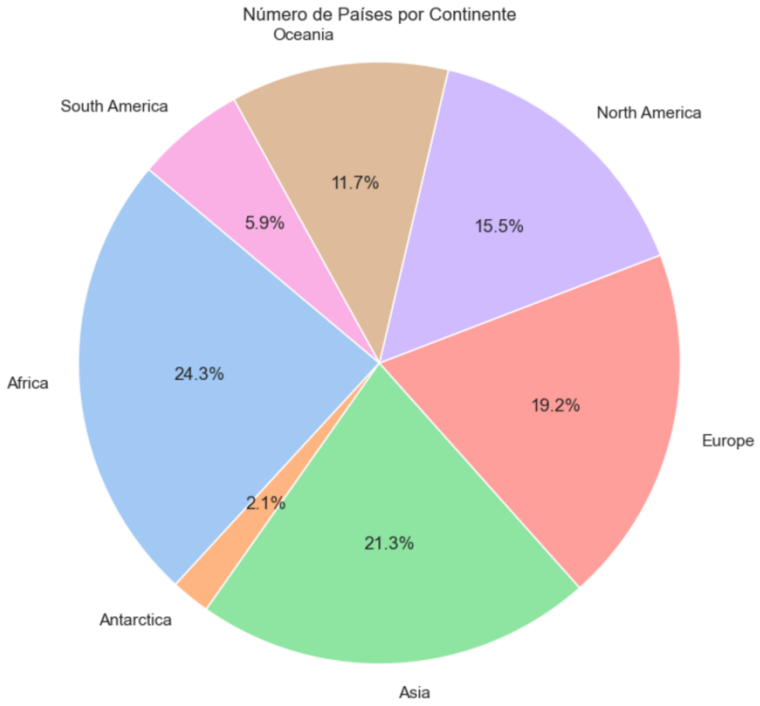
	CityName	Population
0	Mumbai (Bombay)	10500000
1	Seoul	9981619
2	São Paulo	9968485
3	Shanghai	9696300
4	Jakarta	9604900
5	Karachi	9269265
6	Istanbul	8787958
7	Ciudad de México	8591309
8	Moscow	8389200
9	New York	8008278

Pregunta 4: ¿Qué países tienen una expectativa de vida mayor a 80 años?



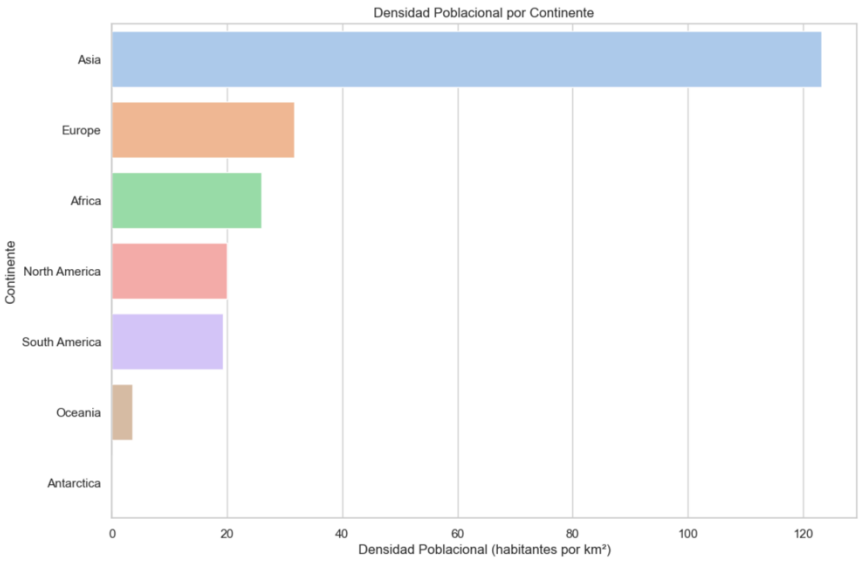
	Name	LifeExpectancy
0	Andorra	83.5
1	Japan	80.7
2	Macao	81.6
3	Singapore	80.1
4	San Marino	81.1

Pregunta 5: ¿Cuáles son los continentes con la mayor cantidad de países?



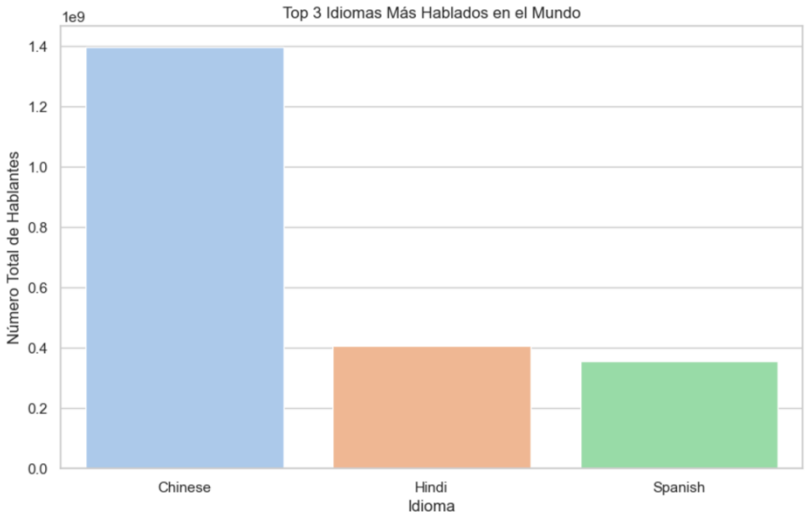
	Continent	CountryCount
0	Africa	58
1	Asia	51
2	Europe	46
3	North America	37
4	Oceania	28
5	South America	14
6	Antarctica	5

Pregunta 6: ¿Qué continentes tienen la mayor y menor densidad poblacional, y cuál es esa densidad?



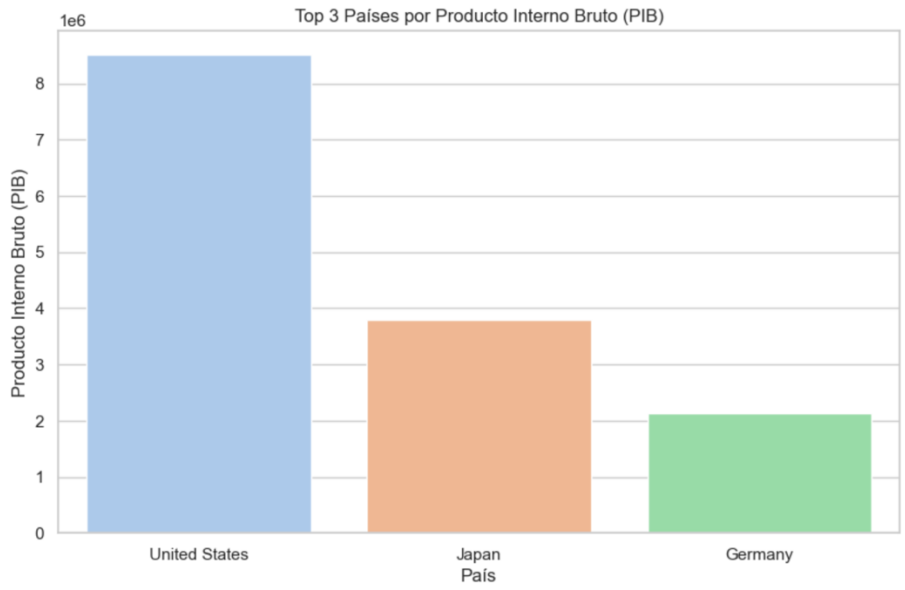
	Continent	PopulationDensity
0	Asia	123.1915
1	Europe	31.6747
2	Africa	25.9327
3	North America	19.9465
4	South America	19.3552
5	Oceania	3.5498
6	Antarctica	0.0000

Pregunta 7: ¿Cuál son los 3 idiomás más hablados en el mundo (considerando solo los datos disponibles)?



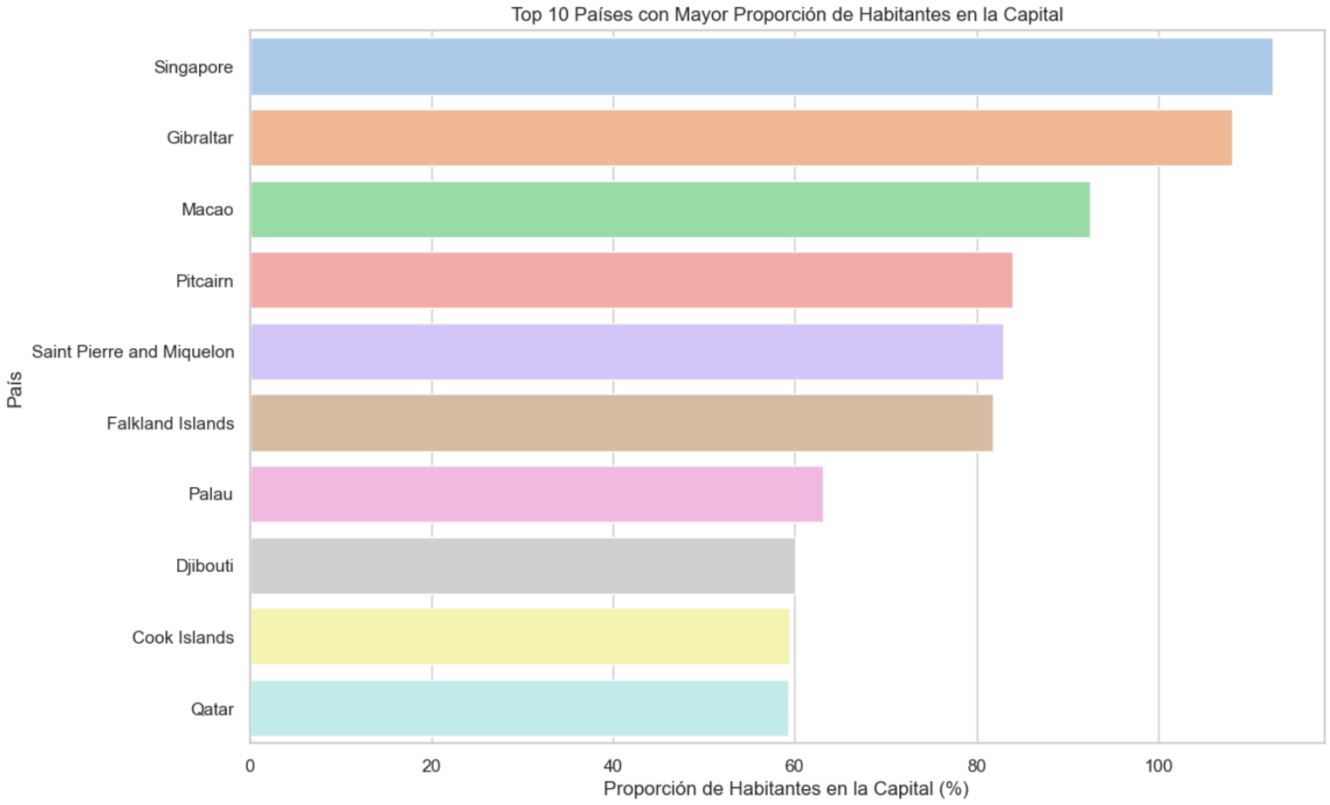
	Language	TotalSpeakers
0	Chinese	1396490179.00000
1	Hindi	405633070.00000
2	Spanish	355029462.00000

Pregunta 8: ¿Cuáles son los 3 países que tienen el mayor Producto Interno Bruto (PIB)?



	Name	GNP
0	United States	8510700.00
1	Japan	3787042.00
2	Germany	2133367.00

Pregunta 9: ¿Cuál es la proporción de habitantes que viven en la capital de cada país respecto a su población total?

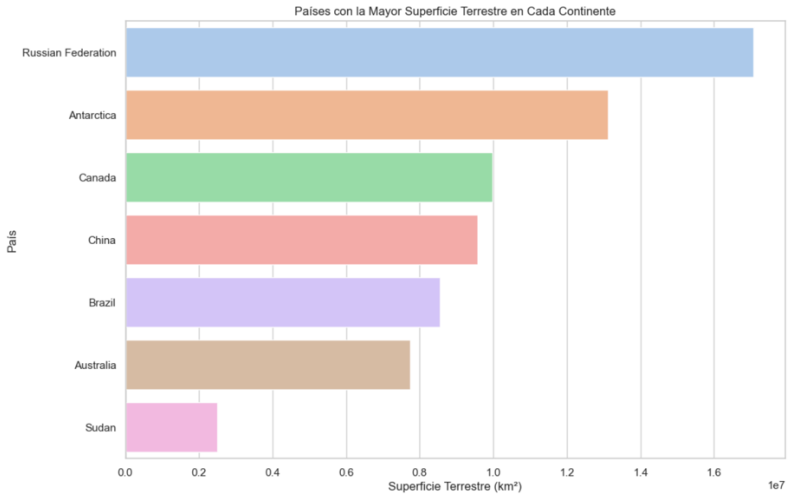


pa

	CountryName	CapitalName	CapitalPopulation	CountryPopulation	CapitalPopulationPercentage
0	Singapore	Singapore	4017733	3567000	112.6362
1	Gibraltar	Gibraltar	27025	25000	108.1000
2	Macao	Macao	437500	473000	92.4947
3	Pitcairn	Adamstown	42	50	84.0000
4	Saint Pierre and Miquelon	Saint-Pierre	5808	7000	82.9714
...	...	...	...	...	...
227	China	Peking	7472000	1500000000	0.4981
228	Pakistan	Islamabad	524500	156483000	0.3352
229	Nigeria	Abuja	350100	111506000	0.3140
230	United States	Washington	572059	278357000	0.2055
231	India	New Delhi	301297	1013662000	0.0297

232 rows × 5 columns

Pregunta 10: ¿Cuáles son los países con la mayor superficie terrestre en cada continente?



	Continent	CountryName	SurfaceArea
0	Europe	Russian Federation	17075400.00
1	Antarctica	Antarctica	13120000.00
2	North America	Canada	9970610.00
3	Asia	China	9572900.00
4	South America	Brazil	8547403.00
5	Oceania	Australia	7741220.00
6	Africa	Sudan	2505813.00

Conclusiones

Después de realizar las consultas y visualizaciones en Python utilizando SQL Server y MySQL, se pueden extraer varias conclusiones importantes sobre los datos analizados, por ejemplo:

La distribución de la población por continente mostró que Asia es el continente con la mayor población, seguido por África y Europa, lo que refleja la gran densidad de población en regiones como Asia que alberga a países como China e India

Al listar las ciudades europeas por su población en orden descendente, se pudo observar que las ciudades más grandes, Moscow, London, St. Petersburg, Berlin, Madrid, son las que concentran la mayor cantidad de habitantes, lo que es crucial para entender la distribución urbana en Europa

Se actualizó la población de China a 1.5 mil millones, reflejando datos más actuales, y este tipo de actualizaciones es vital para mantener la precisión de los análisis y las decisiones basadas en estos datos

Los países con más ciudades en la base de datos fueron identificados, siendo China, India, Estados Unidos los más destacados, lo cual puede ser indicativo de la extensa urbanización y el desarrollo de infraestructuras en estos países

Canadá, China, India, la Federación Rusa y Estados Unidos resultaron ser los países con la mayor diversidad lingüística, contando con la mayor cantidad de idiomas hablados, lo que es un reflejo de la rica herencia cultural y lingüística de cada nación

Se identificaron las diez ciudades más pobladas del mundo, con Mumbai, Seoul y Sao Paulo encabezando la lista, lo que representa importantes centros económicos y culturales a nivel global

Los países con una expectativa de vida mayor a 80 años, como Andorra y Japón, fueron destacados, y esto puede estar relacionado con altos estándares de vida, buenos sistemas de salud y calidad de vida en general

Se determinó que Asia tiene la mayor densidad poblacional, mientras que Antartida/Oceanía tiene la menor, y esta información es crucial para estudios sobre recursos, urbanización y planificación regional

Los tres idiomas más hablados en el mundo se identificaron como el mandarín, el español y el inglés, dominando debido a la gran cantidad de hablantes nativos y su uso global

Los tres países con el mayor PIB son Estados Unidos, Japón y Alemania, lo que es significativo para análisis económicos y estrategias de mercado

Se calculó la proporción de habitantes que viven en las capitales respecto a la población total del país, lo cual es útil para entender la concentración urbana y la importancia de las capitales en cada nación

Se identificaron los países con la mayor superficie terrestre en cada continente, destacando a Rusia en Europa, China en Asia, Canadá en América del Norte y Brasil en América del Sur, lo que proporciona una perspectiva geográfica importante sobre la extensión de los territorios