Universidad Nacional Autónoma de México

Facultad de Ingeniería

PRÁCTICA 2

ANÁLISIS EXPLORATORIO DE DATOS (EDA)

Minería de Datos

Profesor:

Dr. Molero Castillo Guillermo Gilberto

Grupo 1

Alumna:

Monroy Velázquez Alejandra Sarahí

OBJETIVO

Hacer un análisis exploratorio de datos sobre el progreso mundial de vacunación contra COVID-19.

DESARROLLO

El conjunto de datos corresponde a COVID-19 World Vaccination Progress de Kaggle. El diccionario de datos es el siguiente:

Diccionario de datos

1. **País:** Nombre del país.
2. **Código ISO:** Código ISO del país.
3. **Fecha:** Fecha de registro.
4. **Total de vacunaciones:** Número total de vacunaciones en el país.
5. **Total de personas vacunadas:** Una persona, según el esquema de inmunización, recibirá una o más vacunas (normalmente 2).
6. **Total de personas completamente vacunadas:** Número de personas que recibieron el esquema completo de vacunación.
7. **Vacunas diarias (crudos):** Número de vacunaciones para esa fecha/país.
8. **Vacunas diarias:** Número de vacunaciones para esa fecha/país.
9. **Total de vacunaciones por cien:** Relación (en porcentaje) entre el número de vacunaciones y la población total hasta la fecha.
10. **Total de personas vacunadas por cien:** Relación (en porcentaje) entre la población inmunizada y la población total hasta la fecha.
11. **Total de personas totalmente vacunadas por cien:** Relación (en porcentaje) entre la población totalmente inmunizada y la población total hasta la fecha en el país.
12. **Vacunas diarias por millón:** Relación (en ppm) entre el número de vacunaciones y la población total para la fecha actual en el país.
13. **Vacunas utilizadas en el país:** Nombre de las vacunas utilizadas en el país (hasta la fecha).
14. **Nombre de la fuente:** Fuente de la información (autoridad nacional, organización internacional, organización local, entre otros).
15. **Sitio web de origen:** Fuente de información.

4

Primero comenzamos la importación de bibliotecas correspondientes que nos ayudarán para la realización del código, las cuales son *pandas* para la manipulación y análisis de datos, *numpy* para crear vectores y matrices, *matplotlib* para la generación de gráficas, así como *seaborn* para la visualización de datos. También *file* para subir el archivo *csv* que contiene todos los datos.

Una vez importadas, el *dataframe* se lee y se despliega en pantalla, en este caso también se puede utilizar *head()* para visualizar solo las primeras cinco filas del *dataframe*.

```
import pandas as pd          #Para la manipulación y analisis de datos
import numpy as np           #Para crear vectores y matrices n dimensionales
import matplotlib.pyplot as plt #Para la generación de gráficas a partir de los datos
import seaborn as sns        #Para la visualización de datos basados en matplotlib
%matplotlib inline
#Generar imagenes dentro del cuaderno
```

```
from google.colab import files
files.upload()
```

```
Elegir archivos Ningún archivo seleccionado Upload widget is only available when the cell has been executed in the current browser session. Ple
Saving country_vaccinations.csv to country_vaccinations.csv
{'country_vaccinations.csv': b'country,iso_code,date,total_vaccinations,people_vaccinated,people_fully_vaccinated,dai
```

```
DatosVacunacion = pd.read_csv('country_vaccinations.csv')
DatosVacunacion
```

Ahora que el *dataframe* está cargado, comenzamos con los pasos vistos en clase:

1) ***Descripción de la estructura de los datos***

   Lo primero que se hace es ver la forma de la matriz, utilizando el atributo *shape* el cual nos regresa la cantidad de filas y columnas que tiene, en este caso nuestro *dataframe* tiene 15 columnas con 45257 registros.

```
DatosVacunacion.shape
```

```
(45257, 15)
```

   Luego observamos los tipos de datos, utilizando el atributo *dtypes*, en este caso tenemos datos que son objetos y flotantes:

```
DatosVacunacion.dtypes
```

```
country                                object
iso_code                               object
date                                   object
total_vaccinations                    float64
people_vaccinated                     float64
people_fully_vaccinated               float64
daily_vaccinations_raw                float64
daily_vaccinations                    float64
total_vaccinations_per_hundred        float64
people_vaccinated_per_hundred         float64
people_fully_vaccinated_per_hundred   float64
daily_vaccinations_per_million        float64
vaccines                               object
source_name                            object
source_website                         object
dtype: object
```

   Una vez que tenemos una idea general de los datos que conforman nuestro *dataframe* pasamos al siguiente paso.

2) *Identificación de datos faltantes*

En este paso observamos los datos faltantes dentro del *dataframe*, esto nos servirá para tomar decisiones más adelante, ya sea que si no son relevantes los eliminemos o los remplazemos con diferente información, para ello utilizamos la función *isnull().sum()* la cual nos regresa la suma de todos los valores nulos en cada variable.

```
DatosVacunacion.isnull().sum()
```

```
country                                0
iso_code                               0
date                                   0
total_vaccinations                 20550
people_vaccinated                  21676
people_fully_vaccinated            24567
daily_vaccinations_raw             24999
daily_vaccinations                   304
total_vaccinations_per_hundred     20550
people_vaccinated_per_hundred      21676
people_fully_vaccinated_per_hundred 24567
daily_vaccinations_per_million       304
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

Como vemos, en la mayoría de las variables se presentan datos faltantes, a excepción de las que tienen un cero.

Otra opción es usar *info()* la cual es similar:

```
DatosVacunacion.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45257 entries, 0 to 45256
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   country                              45257 non-null  object 
 1   iso_code                             45257 non-null  object 
 2   date                                 45257 non-null  object 
 3   total_vaccinations                   24707 non-null  float64
 4   people_vaccinated                    23581 non-null  float64
 5   people_fully_vaccinated              20690 non-null  float64
 6   daily_vaccinations_raw               20258 non-null  float64
 7   daily_vaccinations                   44953 non-null  float64
 8   total_vaccinations_per_hundred       24707 non-null  float64
 9   people_vaccinated_per_hundred        23581 non-null  float64
 10  people_fully_vaccinated_per_hundred  20690 non-null  float64
 11  daily_vaccinations_per_million       44953 non-null  float64
 12  vaccines                             45257 non-null  object 
 13  source_name                          45257 non-null  object 
 14  source_website                       45257 non-null  object 
dtypes: float64(9), object(6)
memory usage: 5.2+ MB
```

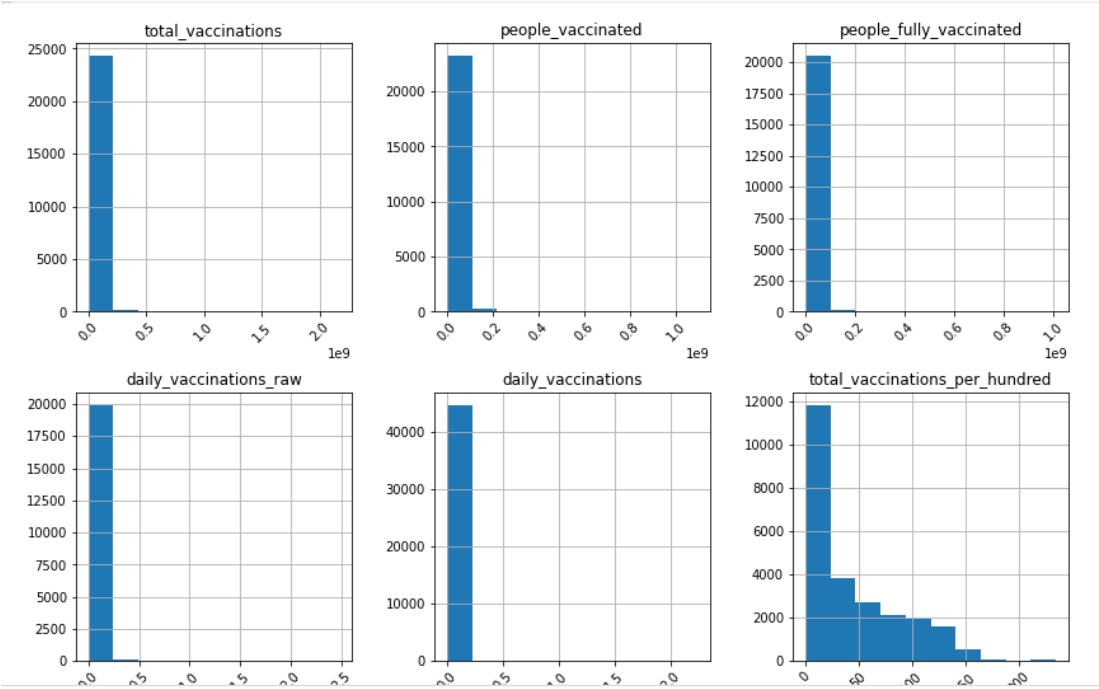
3) *Detección de valores atípicos*

En este paso utilizamos gráficos para visualizar de una forma más facil y rápida los valores atípicos que se encuentran en el *dataframe*. En este paso hacemos cinco pasos intermedios:

a) *Distribución de variables numéricas*

Utilizamos los histogramas para visualizar valores atípicos que podrían ser errores de medición, lo que se buscó fueron limites que no tuvieran sentido, en este caso vemos que *total_vaccinations*, *people_vaccinated*, *people_fully_vaccinated*, *daily_vaccinations_raw*, *daily_vaccinations* y *daily_vaccinations_per_million* tienen valores atípicos de acuerdo con las gráficas:

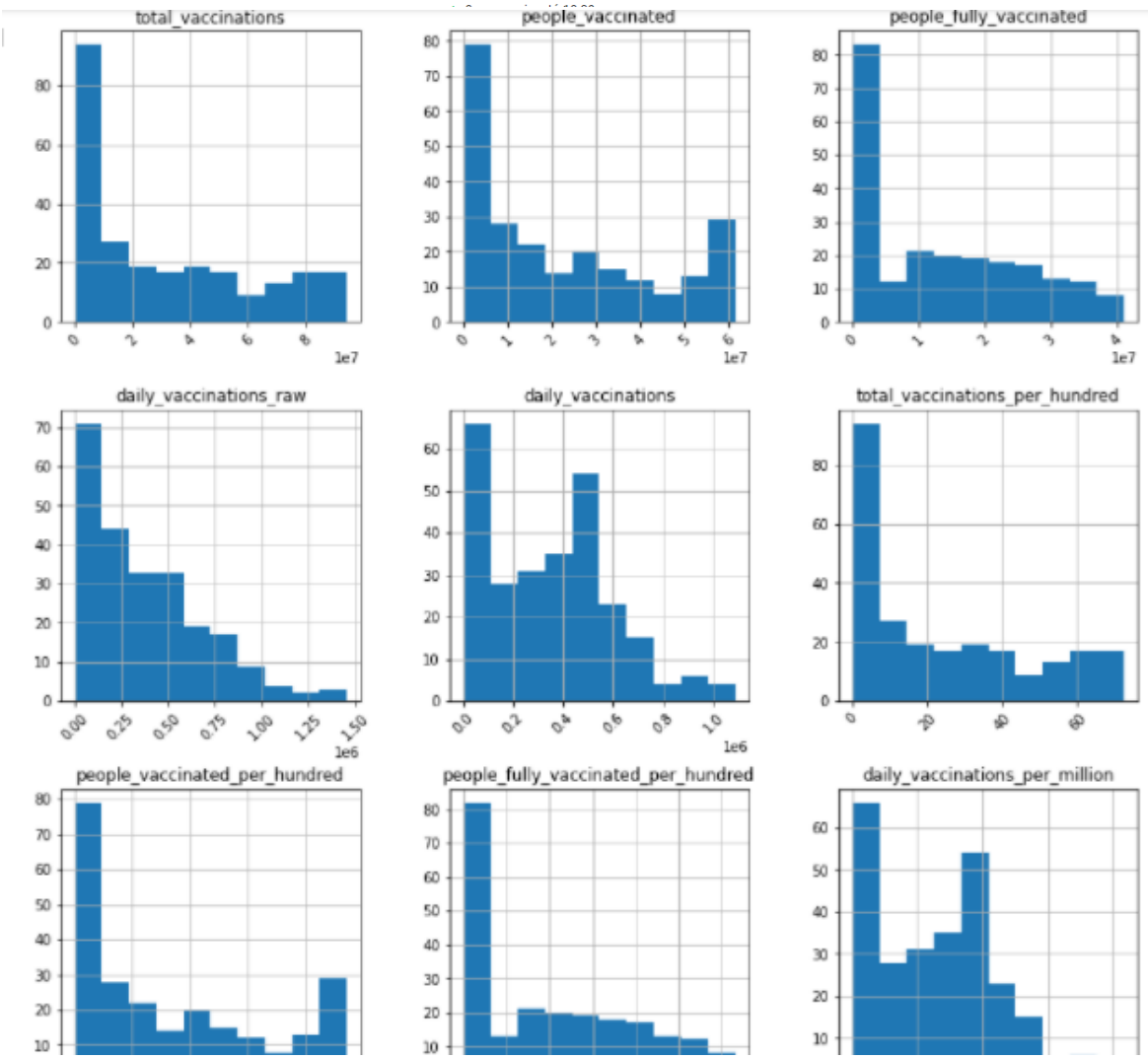
```
DatosVacunacion.hist(figsize=(14,14), xrot=45)
plt.show()
```



Luego, observamos los datos de vacunación, pero ahora los filtramos para solo observar los de México, así como las graficas de los histogramas correspondientes solo a este país:

```
DatosVacunacion[DatosVacunacion.country == 'Mexico']
```

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinati
25968	Mexico	MEX	2020-12-24	2924.0	2924.0	NaN	NaN	NaN	
25969	Mexico	MEX	2020-12-25	NaN	NaN	NaN	NaN	1300.0	
25970	Mexico	MEX	2020-12-26	NaN	NaN	NaN	NaN	1300.0	
25971	Mexico	MEX	2020-12-27	6824.0	6824.0	NaN	NaN	1300.0	
25972	Mexico	MEX	2020-12-28	9579.0	9579.0	NaN	2755.0	1664.0	



Aquí observamos que la distribución de los datos en los histogramas es menos anormal si la comparamos con las anteriores.

b) *Resumen estadístico de variables numéricas*

Después utilizamos *describe()* para observar la matriz que muestra un resumen estadístico de las variables numéricas, con base en ello volvemos a encontrar que podemos identificar valores atípicos.

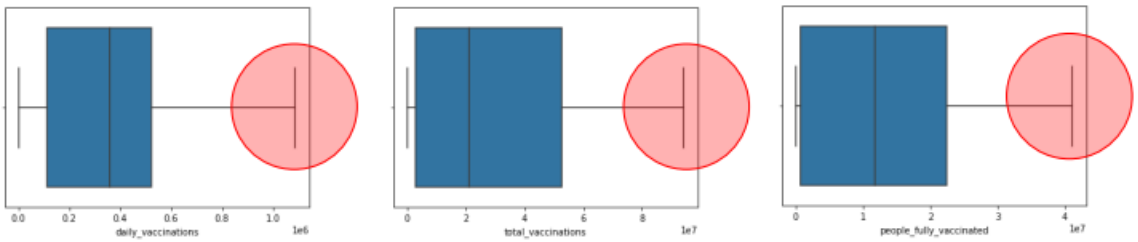
```
DatosVacunacion[DatosVacunacion.country == 'Mexico'].describe()
```

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_v
count	2.490000e+02	2.400000e+02	2.230000e+02	2.350000e+02	2.660000e+02	
mean	3.080362e+07	2.218104e+07	1.341907e+07	3.692836e+05	3.489689e+05	
std	3.012615e+07	2.058768e+07	1.240475e+07	3.192058e+05	2.580214e+05	
min	2.924000e+03	2.924000e+03	1.958000e+03	0.000000e+00	1.300000e+03	
25%	2.676035e+06	2.036169e+06	6.712345e+05	9.965800e+04	1.105195e+05	
50%	2.100862e+07	1.483526e+07	1.179455e+07	3.113180e+05	3.595720e+05	
75%	5.270496e+07	3.782661e+07	2.245211e+07	5.545190e+05	5.221030e+05	
max	9.430053e+07	6.161690e+07	4.111521e+07	1.454578e+06	1.088095e+06	

c) *Diagramas para detectar posibles valores atípicos*

Para una mejor visualización de los datos atípicos se utiliza *seaborn*, lo que nos permite visualizar los diagramas de cajas, con base en ellos podemos observar los valores fuera de rango, lo que una vez más nos confirma que existen valores atípicos en las variables *daily_vaccinations*, *total_vaccinations* y *people_fully_vaccinated*.

```
VariablesValoresAtipicos = ['daily_vaccinations', 'total_vaccinations', 'people_fully_vaccinated']
for col in VariablesValoresAtipicos:
  sns.boxplot(col, data=DatosVacunacion[DatosVacunacion.country == 'Mexico'])
  plt.show()
```



d) *Distribución de variables categóricas*

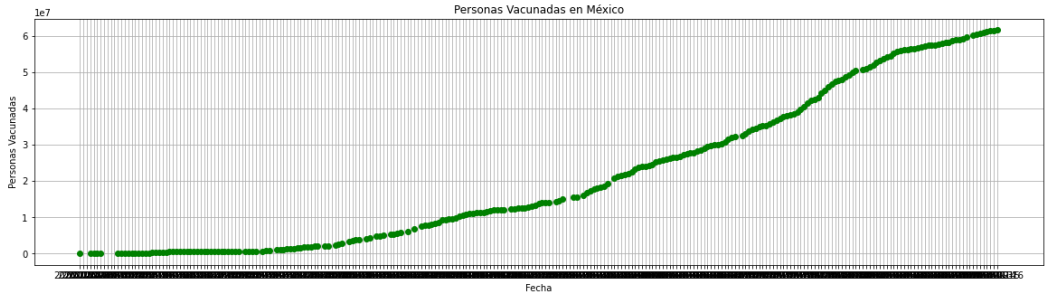
En este paso observamos el recuento de los valores de cada variable, el número de clases únicas, la clase más frecuente y con qué frecuencia ocurre esa clase en el conjunto de datos.

```
DatosVacunacion[DatosVacunacion.country == 'Mexico'].describe(include='object')
```

	country	iso_code	date	vaccines	source_name	source_website
count	267	267	267	267	267	267
unique	1	1	267	1	1	1
top	Mexico	MEX	2021-04-22	CanSino, Johnson&Johnson, Moderna, Oxford/Astr...	Secretary of Health	http://www.gob.mx/cms/uploads/attachment/file/...
freq	267	267	1	267	267	267

Luego, graficamos los datos del total de personas vacunadas y en qué fecha:

```
plt.figure(figsize=(20,5))
plt.plot(DatosVacunacion[DatosVacunacion.country == 'Mexico']['date'], DatosVacunacion[DatosVacunacion.country == 'Mexico']['people_vaccinated'], color='green',
plt.xlabel('Fecha')
plt.ylabel('Personas Vacunadas')
plt.title('Personas Vacunadas en México')
plt.grid(True)
plt.show()
```



4) *Identificación de relaciones entre pares de variables*

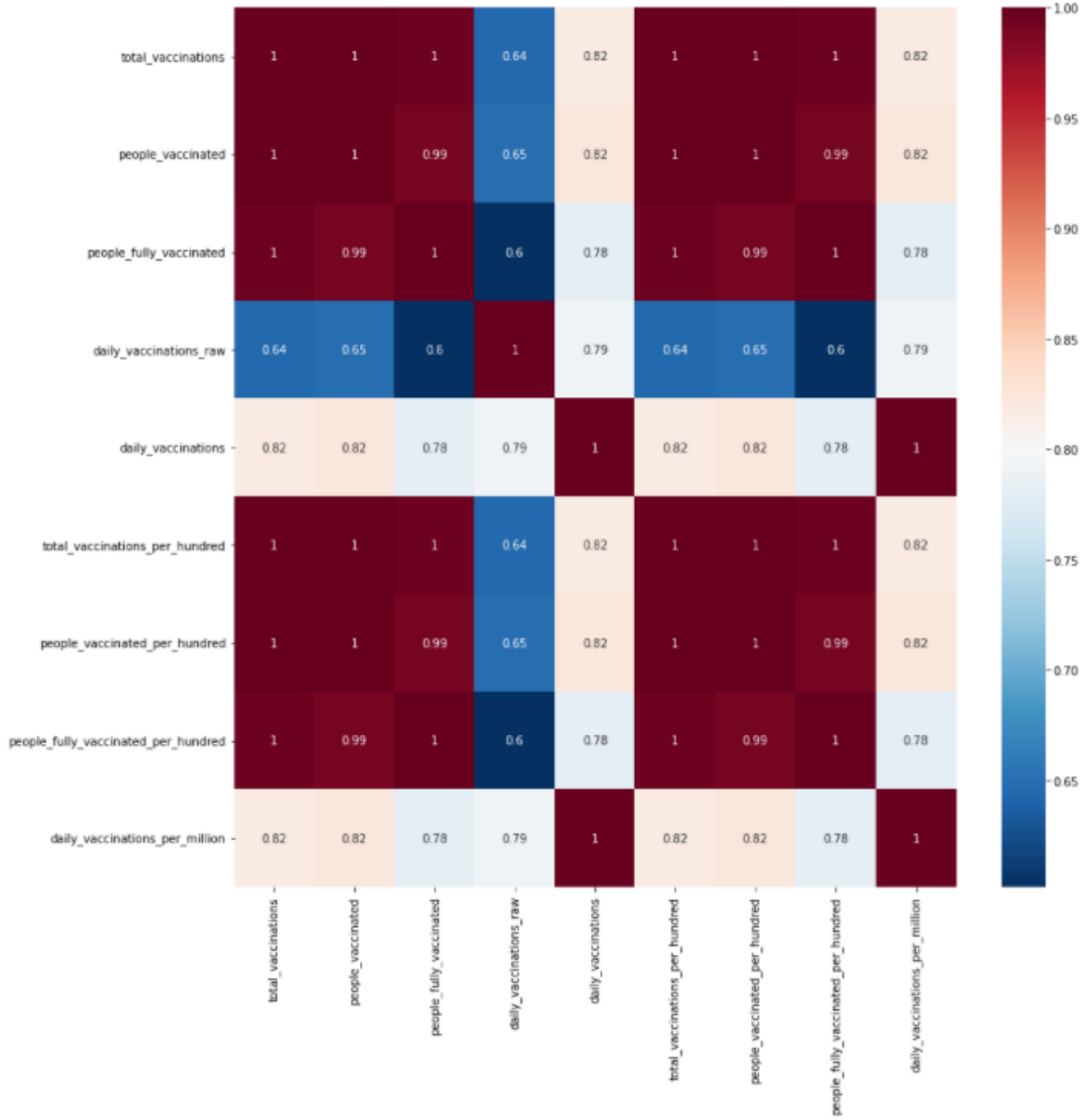
Por último, hacemos uso de la correlación para analizar la relación entre las variables numéricas, utilizando la función *corr()*, esta nos regresara una matriz:

```
DatosVacunacion[DatosVacunacion.country == 'Mexico'].corr()
```

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	t
total_vaccinations	1.000000	0.999259	0.995915	0.643654	0.817368	
people_vaccinated	0.999259	1.000000	0.992050	0.648258	0.823410	
people_fully_vaccinated	0.995915	0.992050	1.000000	0.601532	0.781864	
daily_vaccinations_raw	0.643654	0.648258	0.601532	1.000000	0.792243	
daily_vaccinations	0.817368	0.823410	0.781864	0.792243	1.000000	
total_vaccinations_per_hundred	1.000000	0.999260	0.995915	0.643656	0.817368	
people_vaccinated_per_hundred	0.999259	1.000000	0.992047	0.648273	0.823420	
people_fully_vaccinated_per_hundred	0.995914	0.992049	1.000000	0.601544	0.781851	
daily_vaccinations_per_million	0.817365	0.823407	0.781860	0.792255	1.000000	

Para visualizar mejor esta correlación, utilizamos nuevamente la biblioteca *seaborn* para generar un mapa de calor, entre más rojo es el cuadrado significa que las variables son más similares, lo contrario cuando se acerca al color azul.

```
plt.figure(figsize=(14,14))
sns.heatmap(DatosVacunacion[DatosVacunacion.country == 'Mexico'].corr(), cmap='RdBu_r', annot=True)
plt.show()
```



CONCLUSIÓN

El análisis exploratorio de datos nos ayudo a identificar información relevante en el conjunto de datos de datos sobre el progreso mundial de vacunación contra COVID-19. En esta práctica exploramos de qué forma se conformaba este set de datos, observamos la distribución de las variables mediante histogramas, así como también trabajamos solamente con los datos de México. Vimos los datos atípicos que lo conformaban, así como también los datos estadísticos. Así como en la practica anterior este proceso nos ayudó a tener un panorama general del set de datos.