Instalar kagglehub

In [1]:

```
conda install kagglehub
```

```
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): ...working... done
Solving environment: ...working... done

## Package Plan ##

  environment location: C:\Users\darly\anaconda3\envs\IAexplores

  added / updated specs:
    - kagglehub


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    tqdm-4.67.1                |  py312hfc267ef_0         187 KB
    ------------------------------------------------------------
                                           Total:         187 KB

The following NEW packages will be INSTALLED:

  kagglehub          pkgs/main/win-64::kagglehub-0.2.7-py312haa95532_0
  tqdm               pkgs/main/win-64::tqdm-4.67.1-py312hfc267ef_0



Downloading and Extracting Packages: ...working...
tqdm-4.67.1          | 187 KB    |            |   0%
tqdm-4.67.1          | 187 KB    | 8          |   9%
tqdm-4.67.1          | 187 KB    | ######8    |  68%
tqdm-4.67.1          | 187 KB    | ########## | 100%
tqdm-4.67.1          | 187 KB    | ########## | 100%

 done
Preparing transaction: done
Executing transaction: done

Note: you may need to restart the kernel to use updated packages.
```

importa kaggle, pandas y numpy , y descargar data

In [66]:

```
import kagglehub
import pandas as pd
import numpy as np

#visualizacion
import plotly.express as px
import matplotlib.pyplot as plt
```

In [2]:

```
# Download latest version
path = kagglehub.dataset_download("ruchi798/data-science-job-salaries")

print("Path to dataset files:", path)
```

```
C:\Users\darly\anaconda3\envs\IAexplores\Lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
Warning: Looks like you're using an outdated `kagglehub` version, please consider updating (latest version: 0.3.10)
Downloading from https://www.kaggle.com/api/v1/datasets/download/ruchi798/data-science-job-salaries?dataset_version_number=1...
100%|██████████| 7.37k/7.37k [00:00<?, ?B/s]
Extracting model files...
Path to dataset files: C:\Users\darly\.cache\kagglehub\datasets\ruchi798\data-science-job-salaries\versions\1
```

crear un data frame, una tabla como ejemplo

In [6]:

```
data= pd.DataFrame({
    "nombres": ["ana", "juana", "sara"],
    "edad": [12,23,34]
})
data
```

Out[6]:

	nombres	edad
0	ana	12
1	juana	23
2	sara	34

In [7]:

```
data2= pd.DataFrame({
    "nombres": ["ana", "juana", "sara"],
    "salario": [120,230,340]
})
data2
```

Out[7]:

	nombres	salario
0	ana	120
1	juana	230
2	sara	340

unir data_frame

In [8]:

```
new_df= data.merge(data2)
```

In [9]:

```
new_df
```

Out[9]:

	nombres	edad	salario
0	ana	12	120
1	juana	23	230
2	sara	34	340

leer un archivo csv, ya descargado, e imprimir la cabeza (primero 5 elementos)

In [13]:

```
df = pd.read_csv("C:/Users/darly/.cache/kagglehub/datasets/ruchi798/data-science-jo
df.head()
```

Out[13]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_curr
0	0	2020	MI	FT	Data Scientist	70000	
1	1	2020	SE	FT	Machine Learning Scientist	260000	
2	2	2020	SE	FT	Big Data Engineer	85000	
3	3	2020	MI	FT	Product Data Analyst	20000	
4	4	2020	SE	FT	Machine Learning Engineer	150000	

para completar lineas

In [14]:

```
%config Completer.use_jedi = True
```

mostrar las ultimas 5 lineas

In [15]:

```
df.tail()
```

Out[15]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_cu
602	602	2022	SE	FT	Data Engineer	154000	
603	603	2022	SE	FT	Data Engineer	126000	
604	604	2022	SE	FT	Data Analyst	129000	
605	605	2022	SE	FT	Data Analyst	150000	
606	606	2022	MI	FT	AI Scientist	200000	

para describir la data, muestra un resumen del dataset solo en las variables numericas

In [16]:
```
df.describe()
```

Out[16]:

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio
count	607.000000	607.000000	6.070000e+02	607.000000	607.00000
mean	303.000000	2021.405272	3.240001e+05	112297.869852	70.92257
std	175.370085	0.692133	1.544357e+06	70957.259411	40.70913
min	0.000000	2020.000000	4.000000e+03	2859.000000	0.00000
25%	151.500000	2021.000000	7.000000e+04	62726.000000	50.00000
50%	303.000000	2022.000000	1.150000e+05	101570.000000	100.00000
75%	454.500000	2022.000000	1.650000e+05	150000.000000	100.00000
max	606.000000	2022.000000	3.040000e+07	600000.000000	100.00000

muestra una lista con todas las columnas que tiene el data frame

In [17]:
```
df.columns
```

Out[17]:
```
Index(['Unnamed: 0', 'work_year', 'experience_level', 'employment_type',
       'job_title', 'salary', 'salary_currency', 'salary_in_usd',
       'employee_residence', 'remote_ratio', 'company_location',
       'company_size'],
      dtype='object')
```

esto sirve para hacer consultas especificas del dataframe

In [18]:
```
df[df.salary_in_usd > 250000]
```

Out[18]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_cu
1	1	2020	SE	FT	Machine Learning Scientist	260000	
25	25	2020	EX	FT	Director of Data Science	325000	
33	33	2020	MI	FT	Research Scientist	450000	
63	63	2020	SE	FT	Data Scientist	412000	
78	78	2021	MI	CT	ML Engineer	270000	
93	93	2021	SE	FT	Lead Data Engineer	276000	
97	97	2021	MI	FT	Financial Data Analyst	450000	
157	157	2021	MI	FT	Applied Machine Learning Scientist	423000	
225	225	2021	EX	CT	Principal Data Scientist	416000	
231	231	2021	SE	FT	ML Engineer	256000	
252	252	2021	EX	FT	Principal Data Engineer	600000	
416	416	2022	SE	FT	Data Scientist	260000	
482	482	2022	EX	FT	Data Engineer	324000	
519	519	2022	SE	FT	Applied Data Scientist	380000	
523	523	2022	SE	FT	Data Analytics Lead	405000	

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_cu
534	534	2022	SE	FT	Data Architect	266400	

In [19]:
```
df[df.salary_in_usd > 250000].describe()
```

Out[19]:

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio
count	16.00000	16.000000	16.000000	16.000000	16.000000
mean	233.06250	2021.062500	360837.500000	360837.500000	78.125000
std	197.70364	0.771902	97733.221066	97733.221066	40.697051
min	1.00000	2020.000000	256000.000000	256000.000000	0.000000
25%	74.25000	2020.750000	269100.000000	269100.000000	87.500000
50%	191.00000	2021.000000	352500.000000	352500.000000	100.000000
75%	432.50000	2022.000000	417750.000000	417750.000000	100.000000
max	534.00000	2022.000000	600000.000000	600000.000000	100.000000

realizar consulta para datos cualitativos

In [20]:
```
df.job_title
```

Out[20]:
```
0                    Data Scientist
1         Machine Learning Scientist
2                 Big Data Engineer
3              Product Data Analyst
4          Machine Learning Engineer
                    ...
602                   Data Engineer
603                   Data Engineer
604                    Data Analyst
605                    Data Analyst
606                    AI Scientist
Name: job_title, Length: 607, dtype: object
```

In [21]:
```
df.query("job_title == 'Data Scientist'") #RECUERDE QUE LA CONSULTA QUERY DEBE SER
```

Out[21]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_
0	0	2020	MI	FT	Data Scientist	70000	
7	7	2020	MI	FT	Data Scientist	11000000	
10	10	2020	EN	FT	Data Scientist	45000	
11	11	2020	MI	FT	Data Scientist	3000000	
12	12	2020	EN	FT	Data Scientist	35000	
...	...	...	...	...	...	...	
592	592	2022	SE	FT	Data Scientist	230000	
593	593	2022	SE	FT	Data Scientist	150000	
596	596	2022	SE	FT	Data Scientist	210000	
598	598	2022	MI	FT	Data Scientist	160000	
599	599	2022	MI	FT	Data Scientist	130000	

143 rows × 12 columns

las filas determinadas

In [22]:
```
df.iloc[20:40]
```

Out[22]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_
20	20	2020	MI	FT	Machine Learning Engineer	299000	
21	21	2020	MI	FT	Product Data Analyst	450000	
22	22	2020	SE	FT	Data Engineer	42000	
23	23	2020	MI	FT	BI Data Analyst	98000	
24	24	2020	MI	FT	Lead Data Scientist	115000	
25	25	2020	EX	FT	Director of Data Science	325000	
26	26	2020	EN	FT	Research Scientist	42000	
27	27	2020	SE	FT	Data Engineer	720000	
28	28	2020	EN	CT	Business Data Analyst	100000	
29	29	2020	SE	FT	Machine Learning Manager	157000	
30	30	2020	MI	FT	Data Engineering Manager	51999	
31	31	2020	EN	FT	Big Data Engineer	70000	
32	32	2020	SE	FT	Data Scientist	60000	
33	33	2020	MI	FT	Research Scientist	450000	
34	34	2020	MI	FT	Data Analyst	41000	
35	35	2020	MI	FT	Data Engineer	65000	

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_
36	36	2020	MI	FT	Data Science Consultant	103000	
37	37	2020	EN	FT	Machine Learning Engineer	250000	
38	38	2020	EN	FT	Data Analyst	10000	
39	39	2020	EN	FT	Machine Learning Engineer	138000	

columnas especificas de una dataframe

```
In [23]: df[["job_title", "salary"]]
```

Out[23]:

	job_title	salary
0	Data Scientist	70000
1	Machine Learning Scientist	260000
2	Big Data Engineer	85000
3	Product Data Analyst	20000
4	Machine Learning Engineer	150000
...	...	...
602	Data Engineer	154000
603	Data Engineer	126000
604	Data Analyst	129000
605	Data Analyst	150000
606	AI Scientist	200000

607 rows × 2 columns

otra forma es con la estructura iloc, pero no dando nombres sino posiciones 8recordar que la primera posicion es filas las demas columnas)

```
In [24]: df.iloc[:, [2,4,5]]
```

Out[24]:

	experience_level	job_title	salary
0	MI	Data Scientist	70000
1	SE	Machine Learning Scientist	260000
2	SE	Big Data Engineer	85000
3	MI	Product Data Analyst	20000
4	SE	Machine Learning Engineer	150000
...	...	...	...
602	SE	Data Engineer	154000
603	SE	Data Engineer	126000
604	SE	Data Analyst	129000
605	SE	Data Analyst	150000
606	MI	AI Scientist	200000

607 rows × 3 columns

columnas determinadas y filas determinadas (estas ultimas son las primeras)

In [25]:

```
df.iloc[10:40, [2,4,5]]
```

Out[25]:

	experience_level	job_title	salary
10	EN	Data Scientist	45000
11	MI	Data Scientist	3000000
12	EN	Data Scientist	35000
13	MI	Lead Data Analyst	87000
14	MI	Data Analyst	85000
15	MI	Data Analyst	8000
16	EN	Data Engineer	4450000
17	SE	Big Data Engineer	100000
18	EN	Data Science Consultant	423000
19	MI	Lead Data Engineer	56000
20	MI	Machine Learning Engineer	299000
21	MI	Product Data Analyst	450000
22	SE	Data Engineer	42000
23	MI	BI Data Analyst	98000
24	MI	Lead Data Scientist	115000
25	EX	Director of Data Science	325000
26	EN	Research Scientist	42000
27	SE	Data Engineer	720000
28	EN	Business Data Analyst	100000
29	SE	Machine Learning Manager	157000
30	MI	Data Engineering Manager	51999
31	EN	Big Data Engineer	70000
32	SE	Data Scientist	60000
33	MI	Research Scientist	450000
34	MI	Data Analyst	41000
35	MI	Data Engineer	65000
36	MI	Data Science Consultant	103000
37	EN	Machine Learning Engineer	250000
38	EN	Data Analyst	10000
39	EN	Machine Learning Engineer	138000

las columnas con nombres y no por posicion, desde una a otra

```
In [26]: df.loc[:,"experience_level": "job_title"]
```

Out[26]:

	experience_level	employment_type	job_title
0	MI	FT	Data Scientist
1	SE	FT	Machine Learning Scientist
2	SE	FT	Big Data Engineer
3	MI	FT	Product Data Analyst
4	SE	FT	Machine Learning Engineer
...	...	...	...
602	SE	FT	Data Engineer
603	SE	FT	Data Engineer
604	SE	FT	Data Analyst
605	SE	FT	Data Analyst
606	MI	FT	AI Scientist

607 rows × 3 columns

otra forma de consultar, parecido al query

```
In [27]: df.loc[df["experience_level"]== "MI"]
```

Out[27]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_
0	0	2020	MI	FT	Data Scientist	70000	
3	3	2020	MI	FT	Product Data Analyst	20000	
7	7	2020	MI	FT	Data Scientist	11000000	
8	8	2020	MI	FT	Business Data Analyst	135000	
11	11	2020	MI	FT	Data Scientist	3000000	
...	...	...	...	...	...	...	
567	567	2022	MI	FT	Data Analyst	50000	
586	586	2022	MI	FT	Data Analyst	35000	
598	598	2022	MI	FT	Data Scientist	160000	
599	599	2022	MI	FT	Data Scientist	130000	
606	606	2022	MI	FT	AI Scientist	200000	

213 rows × 12 columns

In [28]:

```
df.loc[df["experience_level"]== "MI", ["job_title",     "salary"]]
```

Out[28]:

	job_title	salary
0	Data Scientist	70000
3	Product Data Analyst	20000
7	Data Scientist	11000000
8	Business Data Analyst	135000
11	Data Scientist	3000000
...	...	...
567	Data Analyst	50000
586	Data Analyst	35000
598	Data Scientist	160000
599	Data Scientist	130000
606	AI Scientist	200000

213 rows × 2 columns

In [29]:

```
df.loc[df["experience_level"]== "MI", ["job_title",     "salary"]].sort_values("sal
```

Out[29]:

	job_title	salary
185	Data Engineer	4000
15	Data Analyst	8000
184	Machine Learning Scientist	12000
192	Big Data Engineer	18000
208	Data Engineer	20000
...	...	...
136	ML Engineer	7000000
137	ML Engineer	8500000
7	Data Scientist	11000000
102	BI Data Analyst	11000000
177	Data Scientist	30400000

213 rows × 2 columns

cambiar el nombre de una columna

In [30]:

```
df.rename(columns= {"salary": "salario"})
```

Out[30]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salario	salary_cu
0	0	2020	MI	FT	Data Scientist	70000	
1	1	2020	SE	FT	Machine Learning Scientist	260000	
2	2	2020	SE	FT	Big Data Engineer	85000	
3	3	2020	MI	FT	Product Data Analyst	20000	
4	4	2020	SE	FT	Machine Learning Engineer	150000	
...	...	...	...	...	...	...	
602	602	2022	SE	FT	Data Engineer	154000	
603	603	2022	SE	FT	Data Engineer	126000	
604	604	2022	SE	FT	Data Analyst	129000	
605	605	2022	SE	FT	Data Analyst	150000	
606	606	2022	MI	FT	AI Scientist	200000	

607 rows × 12 columns

borrar columnas

In [31]:
```
df.drop(columns={"salary"})
```

Out[31]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary_currency
0	0	2020	MI	FT	Data Scientist	EUR
1	1	2020	SE	FT	Machine Learning Scientist	USD
2	2	2020	SE	FT	Big Data Engineer	GBP
3	3	2020	MI	FT	Product Data Analyst	USD
4	4	2020	SE	FT	Machine Learning Engineer	USD
...	...	...	...	...	...	...
602	602	2022	SE	FT	Data Engineer	USD
603	603	2022	SE	FT	Data Engineer	USD
604	604	2022	SE	FT	Data Analyst	USD
605	605	2022	SE	FT	Data Analyst	USD
606	606	2022	MI	FT	AI Scientist	USD

607 rows × 11 columns

agregar una nueva columna o modificarla

In [32]:
```
df["salario en pesos"] = df.salary * 4500
df
```

Out[32]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_cu
0	0	2020	MI	FT	Data Scientist	70000	
1	1	2020	SE	FT	Machine Learning Scientist	260000	
2	2	2020	SE	FT	Big Data Engineer	85000	
3	3	2020	MI	FT	Product Data Analyst	20000	
4	4	2020	SE	FT	Machine Learning Engineer	150000	
...	...	...	...	...	...	...	
602	602	2022	SE	FT	Data Engineer	154000	
603	603	2022	SE	FT	Data Engineer	126000	
604	604	2022	SE	FT	Data Analyst	129000	
605	605	2022	SE	FT	Data Analyst	150000	
606	606	2022	MI	FT	AI Scientist	200000	

607 rows × 13 columns

obtener muestras aleatorias (usos testing)

In [33]:

```
df.sample(frac=0.5) #fragmento deel 50 por ciento de los datos
```

Out[33]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary
465	465	2022	EN	FT	Data Engineer	120000	
435	435	2022	MI	FT	Data Engineer	70000	
232	232	2021	SE	FT	Director of Data Engineering	200000	
259	259	2021	EX	FT	Director of Data Science	120000	
444	444	2022	SE	FT	Data Scientist	215300	
...	...	...	...	...	...	...	
408	408	2022	MI	FT	Data Analyst	40000	
99	99	2021	MI	FT	Computer Vision Software Engineer	81000	
537	537	2022	SE	FT	Data Engineer	155000	
475	475	2022	MI	FT	Data Scientist	70000	
174	174	2021	SE	FT	Research Scientist	51400	

304 rows × 13 columns

In [34]:

```
df.sample(n=100) #numero determinado de muestras
```

Out[34]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	sala
280	280	2021	MI	FT	Data Engineer	112000	
460	460	2022	MI	FT	Machine Learning Infrastructure Engineer	53000	
246	246	2021	EN	FT	Data Scientist	31000	
203	203	2021	SE	FT	Research Scientist	50000	
114	114	2021	MI	FT	Data Engineer	38400	
...	...	...	...	...	...	...	
44	44	2020	MI	FT	Data Engineer	88000	
500	500	2022	SE	FT	Machine Learning Engineer	57000	
273	273	2021	EN	FT	Machine Learning Engineer	85000	
568	568	2022	SE	FT	Data Analyst	80000	
408	408	2022	MI	FT	Data Analyst	40000	

100 rows × 13 columns

agrupar datos determinados y bajo una medida

In [35]:

```
df.groupby("job_title").mean(numeric_only=True)
```

Out[35]:

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio	sala
job_title						
3D Computer Vision Researcher	77.000000	2021.000000	4.000000e+05	5409.000000	50.000000	1.80000
AI Scientist	254.142857	2021.142857	2.905714e+05	66135.571429	78.571429	1.30757
Analytics Engineer	458.250000	2022.000000	1.750000e+05	175000.000000	50.000000	7.87500
Applied Data Scientist	351.600000	2021.600000	1.724000e+05	175655.000000	70.000000	7.75800
Applied Machine Learning Scientist	321.000000	2021.500000	1.413500e+05	142068.750000	87.500000	6.36075
BI Data Analyst	106.333333	2020.833333	1.902045e+06	74755.166667	66.666667	8.55920
Big Data Architect	255.000000	2021.000000	1.250000e+05	99703.000000	50.000000	5.62500
Big Data Engineer	123.125000	2020.625000	4.550000e+05	51974.000000	50.000000	2.04750
Business Data Analyst	256.800000	2021.000000	3.550000e+05	76691.200000	90.000000	1.59750
Cloud Data Engineer	122.000000	2021.000000	1.400000e+05	124647.000000	75.000000	6.30000
Computer Vision Engineer	274.833333	2021.166667	8.350000e+04	44419.333333	58.333333	3.75750
Computer Vision Software Engineer	235.666667	2021.333333	1.003333e+05	105248.666667	100.000000	4.51500
Data Analyst	362.010309	2021.680412	9.660496e+04	92893.061856	75.257732	4.34722
Data Analytics Engineer	216.750000	2021.250000	6.175000e+04	64799.250000	75.000000	2.77875
Data Analytics Lead	523.000000	2022.000000	4.050000e+05	405000.000000	100.000000	1.82250
Data Analytics Manager	366.285714	2021.571429	1.271343e+05	127134.285714	85.714286	5.72104

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio	sala
job_title						
Data Architect	390.636364	2021.727273	1.778739e+05	177873.909091	100.000000	8.00432
Data Engineer	343.537879	2021.590909	1.792106e+05	112725.000000	75.000000	8.06447
Data Engineering Manager	107.200000	2020.600000	1.197998e+05	123227.200000	70.000000	5.39099
Data Science Consultant	138.000000	2020.714286	1.227143e+05	69420.714286	71.428571	5.52214
Data Science Engineer	229.666667	2021.333333	8.450000e+04	75803.333333	83.333333	3.80250
Data Science Manager	274.000000	2021.333333	1.062599e+06	158328.500000	83.333333	4.78169
Data Scientist	314.832168	2021.391608	5.083472e+05	108187.832168	63.986014	2.28756
Data Specialist	165.000000	2021.000000	1.650000e+05	165000.000000	100.000000	7.42500
Director of Data Engineering	171.500000	2021.000000	1.412500e+05	156738.000000	100.000000	6.35625
Director of Data Science	185.857143	2021.000000	1.932857e+05	195074.000000	42.857143	8.69785
ETL Developer	373.500000	2022.000000	5.000000e+04	54957.000000	0.000000	2.25000
Finance Data Analyst	183.000000	2021.000000	4.500000e+04	61896.000000	50.000000	2.02500
Financial Data Analyst	279.000000	2021.500000	2.750000e+05	275000.000000	75.000000	1.23750
Head of Data	302.200000	2021.400000	1.564000e+05	160162.600000	90.000000	7.03800
Head of Data Science	270.250000	2021.500000	1.467188e+05	146718.750000	50.000000	6.60234
Head of Machine Learning	384.000000	2022.000000	6.000000e+06	79039.000000	50.000000	2.70000
Lead Data Analyst	64.333333	2020.666667	5.690000e+05	92203.000000	100.000000	2.56050
Lead Data Engineer	145.500000	2020.833333	1.403333e+05	139724.500000	66.666667	6.31500

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio	sala
job_title						
Lead Data Scientist	53.000000	2020.333333	1.101667e+06	115190.000000	50.000000	4.95750
Lead Machine Learning Engineer	457.000000	2022.000000	8.000000e+04	87932.000000	0.000000	3.60000
ML Engineer	179.333333	2021.000000	2.676667e+06	117504.000000	83.333333	1.20450
Machine Learning Developer	358.000000	2021.666667	1.000000e+05	85860.666667	83.333333	4.50000
Machine Learning Engineer	288.585366	2021.317073	2.727179e+05	104880.146341	67.073171	1.22723
Machine Learning Infrastructure Engineer	234.333333	2021.000000	9.733333e+04	101145.000000	50.000000	4.38000
Machine Learning Manager	29.000000	2020.000000	1.570000e+05	117104.000000	50.000000	7.06500
Machine Learning Scientist	248.000000	2021.250000	1.584125e+05	158412.500000	68.750000	7.12856
Marketing Data Analyst	90.000000	2021.000000	7.500000e+04	88654.000000	100.000000	3.37500
NLP Engineer	455.000000	2022.000000	2.400000e+05	37236.000000	50.000000	1.08000
Principal Data Analyst	370.000000	2021.500000	1.225000e+05	122500.000000	100.000000	5.51250
Principal Data Engineer	196.000000	2021.000000	3.283333e+05	328333.333333	100.000000	1.47750
Principal Data Scientist	205.285714	2021.000000	2.067143e+05	215242.428571	85.714286	9.30214
Product Data Analyst	12.000000	2020.000000	2.350000e+05	13036.000000	50.000000	1.05750
Research Scientist	246.562500	2021.125000	1.104937e+05	109019.500000	53.125000	4.97221
Staff Data Scientist	283.000000	2021.000000	1.050000e+05	105000.000000	100.000000	4.72500

In [36]:

```
df.groupby("job_title").mean(numeric_only=True).count() #cuenta
```

Out[36]:

```
Unnamed: 0          50
work_year           50
salary              50
salary_in_usd       50
remote_ratio        50
salario en pesos    50
dtype: int64
```

In [37]:

```
df.groupby("job_title").agg({
    "salary": ["max", "mean"]
})  #agrupar por una columan y determinadas medidas
```

Out[37]:

	salary	
	max	mean
job_title		
3D Computer Vision Researcher	400000	4.000000e+05
AI Scientist	1335000	2.905714e+05
Analytics Engineer	205300	1.750000e+05
Applied Data Scientist	380000	1.724000e+05
Applied Machine Learning Scientist	423000	1.413500e+05
BI Data Analyst	11000000	1.902045e+06
Big Data Architect	125000	1.250000e+05
Big Data Engineer	1672000	4.550000e+05
Business Data Analyst	1400000	3.550000e+05
Cloud Data Engineer	160000	1.400000e+05
Computer Vision Engineer	180000	8.350000e+04
Computer Vision Software Engineer	150000	1.003333e+05
Data Analyst	450000	9.660496e+04
Data Analytics Engineer	110000	6.175000e+04
Data Analytics Lead	405000	4.050000e+05
Data Analytics Manager	150260	1.271343e+05
Data Architect	266400	1.778739e+05
Data Engineer	4450000	1.792106e+05
Data Engineering Manager	174000	1.197998e+05
Data Science Consultant	423000	1.227143e+05
Data Science Engineer	159500	8.450000e+04
Data Science Manager	7000000	1.062599e+06
Data Scientist	30400000	5.083472e+05
Data Specialist	165000	1.650000e+05
Director of Data Engineering	200000	1.412500e+05
Director of Data Science	325000	1.932857e+05
ETL Developer	50000	5.000000e+04
Finance Data Analyst	45000	4.500000e+04

	salary	
	max	mean
job_title		
Financial Data Analyst	450000	2.750000e+05
Head of Data	235000	1.564000e+05
Head of Data Science	224000	1.467188e+05
Head of Machine Learning	6000000	6.000000e+06
Lead Data Analyst	1450000	5.690000e+05
Lead Data Engineer	276000	1.403333e+05
Lead Data Scientist	3000000	1.101667e+06
Lead Machine Learning Engineer	80000	8.000000e+04
ML Engineer	8500000	2.676667e+06
Machine Learning Developer	100000	1.000000e+05
Machine Learning Engineer	4900000	2.727179e+05
Machine Learning Infrastructure Engineer	195000	9.733333e+04
Machine Learning Manager	157000	1.570000e+05
Machine Learning Scientist	260000	1.584125e+05
Marketing Data Analyst	75000	7.500000e+04
NLP Engineer	240000	2.400000e+05
Principal Data Analyst	170000	1.225000e+05
Principal Data Engineer	600000	3.283333e+05
Principal Data Scientist	416000	2.067143e+05
Product Data Analyst	450000	2.350000e+05
Research Scientist	450000	1.104937e+05
Staff Data Scientist	105000	1.050000e+05

contar elementos de una columnas

In [39]:
```
df.shape #tamaño de data
```

Out[39]:
```
(607, 13)
```

elementos unicos de cada columna

In [40]:
```
df.nunique()
```

```
Out[40]: Unnamed: 0            607
         work_year               3
         experience_level        4
         employment_type         4
         job_title              50
         salary                272
         salary_currency        17
         salary_in_usd         369
         employee_residence     57
         remote_ratio            3
         company_location       50
         company_size            3
         salario en pesos      272
         dtype: int64
```

hacer limpieza de datos

```
In [41]: df.count() #contar datos
```

```
Out[41]: Unnamed: 0            607
         work_year             607
         experience_level      607
         employment_type       607
         job_title             607
         salary                607
         salary_currency       607
         salary_in_usd         607
         employee_residence    607
         remote_ratio          607
         company_location      607
         company_size          607
         salario en pesos      607
         dtype: int64
```

```
In [42]: df.isnull().sum() #que datos son nulos
```

```
Out[42]: Unnamed: 0            0
         work_year             0
         experience_level      0
         employment_type       0
         job_title             0
         salary                0
         salary_currency       0
         salary_in_usd         0
         employee_residence    0
         remote_ratio          0
         company_location      0
         company_size          0
         salario en pesos      0
         dtype: int64
```

Visualizacion de la data

```
In [143…  top10_job_title = df['job_title'].value_counts()
          top10_job_title
```

Out[143…

```
job_title
Data Scientist                              143
Data Engineer                               132
Data Analyst                                 97
Machine Learning Engineer                    41
Research Scientist                           16
Data Science Manager                         12
Data Architect                               11
Machine Learning Scientist                    8
Big Data Engineer                             8
Director of Data Science                      7
AI Scientist                                  7
Principal Data Scientist                      7
Data Science Consultant                       7
Data Analytics Manager                        7
BI Data Analyst                               6
Computer Vision Engineer                      6
ML Engineer                                   6
Lead Data Engineer                            6
Applied Data Scientist                        5
Business Data Analyst                         5
Data Engineering Manager                      5
Head of Data                                  5
Data Analytics Engineer                       4
Head of Data Science                          4
Applied Machine Learning Scientist            4
Analytics Engineer                            4
Machine Learning Developer                    3
Data Science Engineer                         3
Lead Data Analyst                             3
Machine Learning Infrastructure Engineer      3
Lead Data Scientist                           3
Principal Data Engineer                       3
Computer Vision Software Engineer             3
Product Data Analyst                          2
ETL Developer                                 2
Cloud Data Engineer                           2
Financial Data Analyst                        2
Director of Data Engineering                  2
Principal Data Analyst                        2
Machine Learning Manager                      1
Marketing Data Analyst                        1
3D Computer Vision Researcher                 1
Finance Data Analyst                          1
Data Specialist                               1
Staff Data Scientist                          1
Big Data Architect                            1
Head of Machine Learning                      1
NLP Engineer                                  1
Lead Machine Learning Engineer                1
Data Analytics Lead                           1
Name: count, dtype: int64
```

In [144…

```python
top10_job_title = df['job_title'].value_counts()[:10] #las primeras 10 empleos mas
top10_job_title
```

```
Out[144…   job_title
           Data Scientist                143
           Data Engineer                 132
           Data Analyst                   97
           Machine Learning Engineer      41
           Research Scientist             16
           Data Science Manager           12
           Data Architect                 11
           Machine Learning Scientist      8
           Big Data Engineer               8
           Director of Data Science        7
           Name: count, dtype: int64
```

dibujar un diagrama de barras * px.bar(...): Crea un gráfico de barras. * x=top10_job_title.index: Usa los títulos de trabajo (índices de la serie) como el eje X. * y=top10_job_title.values: Usa la cantidad de veces que aparecen los títulos como eje Y. * color=top10_job_title.index: Asigna diferentes colores a cada categoría (título de trabajo). * color_discrete_sequence=px.colors.sequential.PuBuGn: Usa una paleta de colores predefinida (PuBuGn). * text=top10_job_title.values: Muestra los valores sobre las barras. * title='2.1.2. Top 10 Job Titles': Agrega un título al gráfico. * template='plotly_dark': Usa un tema oscuro para el diseño.

```
In [148…   px.bar?  #visualizar opciones de la función
```

```
Signature:
px.bar(
    data_frame=None,
    x=None,
    y=None,
    color=None,
    pattern_shape=None,
    facet_row=None,
    facet_col=None,
    facet_col_wrap=0,
    facet_row_spacing=None,
    facet_col_spacing=None,
    hover_name=None,
    hover_data=None,
    custom_data=None,
    text=None,
    base=None,
    error_x=None,
    error_x_minus=None,
    error_y=None,
    error_y_minus=None,
    animation_frame=None,
    animation_group=None,
    category_orders=None,
    labels=None,
    color_discrete_sequence=None,
    color_discrete_map=None,
    color_continuous_scale=None,
    pattern_shape_sequence=None,
    pattern_shape_map=None,
    range_color=None,
    color_continuous_midpoint=None,
    opacity=None,
    orientation=None,
    barmode='relative',
    log_x=False,
    log_y=False,
    range_x=None,
    range_y=None,
    text_auto=False,
    title=None,
    template=None,
    width=None,
    height=None,
) -> plotly.graph_objs._figure.Figure
Docstring:
    In a bar plot, each row of `data_frame` is represented as a rectangular
    mark.

Parameters
----------
data_frame: DataFrame or array-like or dict
    This argument needs to be passed for column names (and not keyword
    names) to be used. Array-like and dict are transformed internally to a
    pandas DataFrame. Optional: if missing, a DataFrame gets constructed
    under the hood using the other arguments.
```

```
x: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    position marks along the x axis in cartesian coordinates. Either `x` or
    `y` can optionally be a list of column references or array_likes,  in
    which case the data will be treated as if it were 'wide' rather than
    'long'.
y: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    position marks along the y axis in cartesian coordinates. Either `x` or
    `y` can optionally be a list of column references or array_likes,  in
    which case the data will be treated as if it were 'wide' rather than
    'long'.
color: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    assign color to marks.
pattern_shape: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    assign pattern shapes to marks.
facet_row: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    assign marks to facetted subplots in the vertical direction.
facet_col: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like are used to
    assign marks to facetted subplots in the horizontal direction.
facet_col_wrap: int
    Maximum number of facet columns. Wraps the column variable at this
    width, so that the column facets span multiple rows. Ignored if 0, and
    forced to 0 if `facet_row` or a `marginal` is set.
facet_row_spacing: float between 0 and 1
    Spacing between facet rows, in paper units. Default is 0.03 or 0.07
    when facet_col_wrap is used.
facet_col_spacing: float between 0 and 1
    Spacing between facet columns, in paper units Default is 0.02.
hover_name: str or int or Series or array-like
    Either a name of a column in `data_frame`, or a pandas Series or
    array_like object. Values from this column or array_like appear in bold
    in the hover tooltip.
hover_data: str, or list of str or int, or Series or array-like, or dict
    Either a name or list of names of columns in `data_frame`, or pandas
    Series, or array_like objects or a dict with column names as keys, with
    values True (for default formatting) False (in order to remove this
    column from hover information), or a formatting string, for example
    ':.3f' or '|%a' or list-like data to appear in the hover tooltip or
    tuples with a bool or formatting string as first element, and list-like
    data to appear in hover as second element Values from these columns
    appear as extra data in the hover tooltip.
custom_data: str, or list of str or int, or Series or array-like
    Either name or list of names of columns in `data_frame`, or pandas
    Series, or array_like objects Values from these columns are extra data,
    to be used in widgets or Dash callbacks for example. This data is not
```

```
        user-visible but is included in events emitted by the figure (lasso
        selection etc.)
    text: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like appear in the
        figure as text labels.
    base: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        position the base of the bar.
    error_x: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        size x-axis error bars. If `error_x_minus` is `None`, error bars will
        be symmetrical, otherwise `error_x` is used for the positive direction
        only.
    error_x_minus: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        size x-axis error bars in the negative direction. Ignored if `error_x`
        is `None`.
    error_y: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        size y-axis error bars. If `error_y_minus` is `None`, error bars will
        be symmetrical, otherwise `error_y` is used for the positive direction
        only.
    error_y_minus: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        size y-axis error bars in the negative direction. Ignored if `error_y`
        is `None`.
    animation_frame: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        assign marks to animation frames.
    animation_group: str or int or Series or array-like
        Either a name of a column in `data_frame`, or a pandas Series or
        array_like object. Values from this column or array_like are used to
        provide object-constancy across animation frames: rows with matching
        `animation_group`s will be treated as if they describe the same object
        in each frame.
    category_orders: dict with str keys and list of str values (default `{}`)
        By default, in Python 3.6+, the order of categorical values in axes,
        legends and facets depends on the order in which these values are first
        encountered in `data_frame` (and no order is guaranteed by default in
        Python below 3.6). This parameter is used to force a specific ordering
        of values per column. The keys of this dict should correspond to column
        names, and the values should be lists of strings corresponding to the
        specific display order desired.
    labels: dict with str keys and str values (default `{}`)
        By default, column names are used in the figure for axis titles, legend
        entries and hovers. This parameter allows this to be overridden. The
        keys of this dict should correspond to column names, and the values
        should correspond to the desired label to be displayed.
    color_discrete_sequence: list of str
```

```
        Strings should define valid CSS-colors. When `color` is set and the
        values in the corresponding column are not numeric, values in that
        column are assigned colors by cycling through `color_discrete_sequence`
        in the order described in `category_orders`, unless the value of
        `color` is a key in `color_discrete_map`. Various useful color
        sequences are available in the `plotly.express.colors` submodules,
        specifically `plotly.express.colors.qualitative`.
    color_discrete_map: dict with str keys and str values (default `{}`)
        String values should define valid CSS-colors Used to override
        `color_discrete_sequence` to assign a specific colors to marks
        corresponding with specific values. Keys in `color_discrete_map` should
        be values in the column denoted by `color`. Alternatively, if the
        values of `color` are valid colors, the string `'identity'` may be
        passed to cause them to be used directly.
    color_continuous_scale: list of str
        Strings should define valid CSS-colors This list is used to build a
        continuous color scale when the column denoted by `color` contains
        numeric data. Various useful color scales are available in the
        `plotly.express.colors` submodules, specifically
        `plotly.express.colors.sequential`, `plotly.express.colors.diverging`
        and `plotly.express.colors.cyclical`.
    pattern_shape_sequence: list of str
        Strings should define valid plotly.js patterns-shapes. When
        `pattern_shape` is set, values in that column are assigned patterns-
        shapes by cycling through `pattern_shape_sequence` in the order
        described in `category_orders`, unless the value of `pattern_shape` is
        a key in `pattern_shape_map`.
    pattern_shape_map: dict with str keys and str values (default `{}`)
        Strings values define plotly.js patterns-shapes. Used to override
        `pattern_shape_sequences` to assign a specific patterns-shapes to lines
        corresponding with specific values. Keys in `pattern_shape_map` should
        be values in the column denoted by `pattern_shape`. Alternatively, if
        the values of `pattern_shape` are valid patterns-shapes names, the
        string `'identity'` may be passed to cause them to be used directly.
    range_color: list of two numbers
        If provided, overrides auto-scaling on the continuous color scale.
    color_continuous_midpoint: number (default `None`)
        If set, computes the bounds of the continuous color scale to have the
        desired midpoint. Setting this value is recommended when using
        `plotly.express.colors.diverging` color scales as the inputs to
        `color_continuous_scale`.
    opacity: float
        Value between 0 and 1. Sets the opacity for markers.
    orientation: str, one of `'h'` for horizontal or `'v'` for vertical.
        (default `'v'` if `x` and `y` are provided and both continous or both
        categorical,  otherwise `'v'`(`'h'`) if `x`(`y`) is categorical and
        `y`(`x`) is continuous,  otherwise `'v'`(`'h'`) if only `x`(`y`) is
        provided)
    barmode: str (default `'relative'`)
        One of `'group'`, `'overlay'` or `'relative'` In `'relative'` mode,
        bars are stacked above zero for positive values and below zero for
        negative values. In `'overlay'` mode, bars are drawn on top of one
        another. In `'group'` mode, bars are placed beside each other.
    log_x: boolean (default `False`)
        If `True`, the x-axis is log-scaled in cartesian coordinates.
    log_y: boolean (default `False`)
        If `True`, the y-axis is log-scaled in cartesian coordinates.
    range_x: list of two numbers
        If provided, overrides auto-scaling on the x-axis in cartesian
        coordinates.
    range_y: list of two numbers
        If provided, overrides auto-scaling on the y-axis in cartesian
        coordinates.
    text_auto: bool or string (default `False`)
        If `True` or a string, the x or y or z values will be displayed as
        text, depending on the orientation A string like `'.2f'` will be
        interpreted as a `texttemplate` numeric formatting directive.
    title: str
        The figure title.
    template: str or dict or plotly.graph_objects.layout.Template instance
        The figure template name (must be a key in plotly.io.templates) or
        definition.
    width: int (default `None`)
        The figure width in pixels.
    height: int (default `None`)
        The figure height in pixels.

    Returns
    -------
        plotly.graph_objects.Figure
    File:      c:\users\darly\anaconda3\envs\iaexplores\lib\site-packages\plotly\express
    \_chart_types.py
    Type:      function
```

In [147…

```
fig = px.bar(y=top10_job_title.values,
             x=top10_job_title.index,
             color = top10_job_title.index,
             color_discrete_sequence=px.colors.sequential.PuBuGn,
             text=top10_job_title.values,
             title= '2.1.2. Top 10 Job Titles',
             template= 'plotly_dark')
fig.show()
```

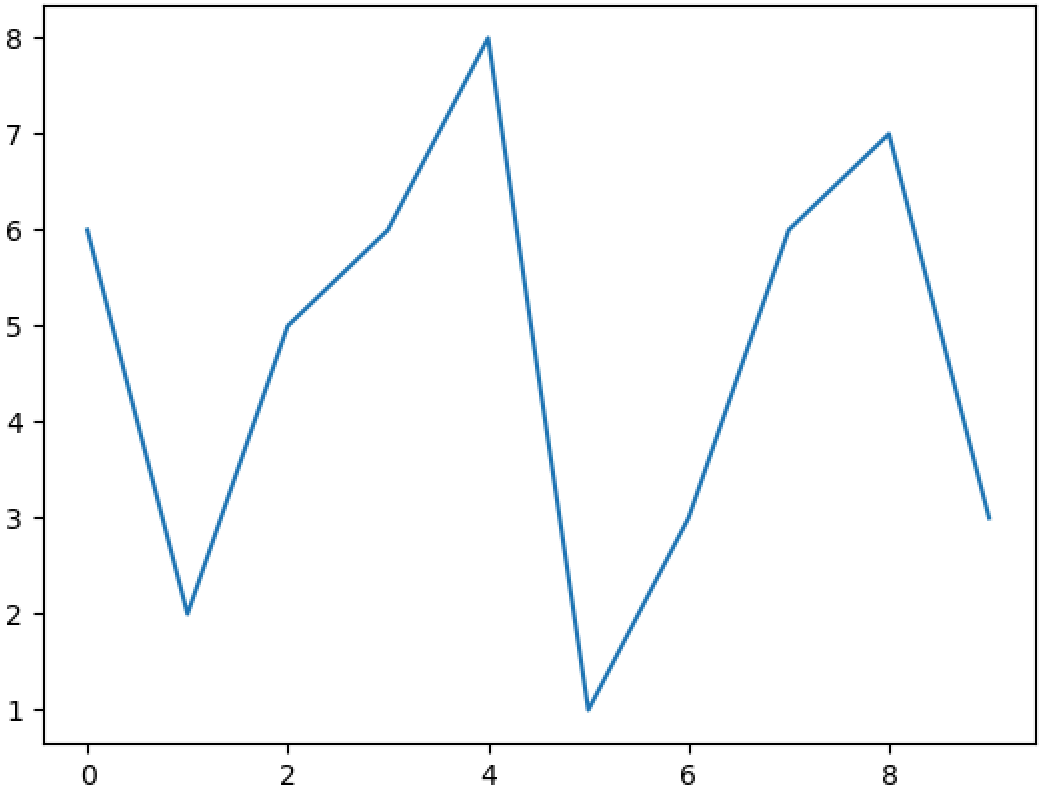
El método update_layout() se usa para modificar el diseño del gráfico. Aquí está lo que hace cada argumento: * xaxis_title="Job Titles" : Cambia el título del eje X a "Job Titles" (Títulos de Trabajo). y Este eje representa las categorías (diferentes títulos de trabajo). *yaxis_title="count" : Cambia el título del eje Y a "count" (Cantidad). Este eje muestra la frecuencia de cada título de trabajo en los datos. * font=dict(size=17, family="Franklin Gothic") Ajusta el tamaño y la fuente del texto en el gráfico. size=17: Aumenta el tamaño del texto a 17 puntos. family="Franklin Gothic": Usa la fuente "Franklin Gothic" para los textos.

In [48]:

```
fig.update_layout(
    xaxis_title="Job Titles",
    yaxis_title="count",
    font = dict(size=17,family="Franklin Gothic"))
fig.show()
```

In [150…

```
lista = [6, 2, 5, 6, 8, 1, 3, 6, 7, 3]
plt.plot(lista)
plt.show()
```



vamos a construir un digrama de lineas por cada variable cuantitativa, sirve para ver el comportramiento de una variable en el tiempo

In [107…

```
df_cuant= df.select_dtypes(include=['int64', 'float64'])
df_cuant
```

Out[107…

	Unnamed: 0	work_year	salary	salary_in_usd	remote_ratio	salario en pesos
0	0	2020	70000	79833	0	315000000
1	1	2020	260000	260000	0	1170000000
2	2	2020	85000	109024	50	382500000
3	3	2020	20000	20000	0	90000000
4	4	2020	150000	150000	50	675000000
...	...	...	...	...	...	...
602	602	2022	154000	154000	100	693000000
603	603	2022	126000	126000	100	567000000
604	604	2022	129000	129000	0	580500000
605	605	2022	150000	150000	100	675000000
606	606	2022	200000	200000	100	900000000

607 rows × 6 columns

In [108…

```
df_cuant= df_cuant.iloc[:, 1:] #eliminar la columna cero a partir del indice
```

In [109…

```
df_cuant
```

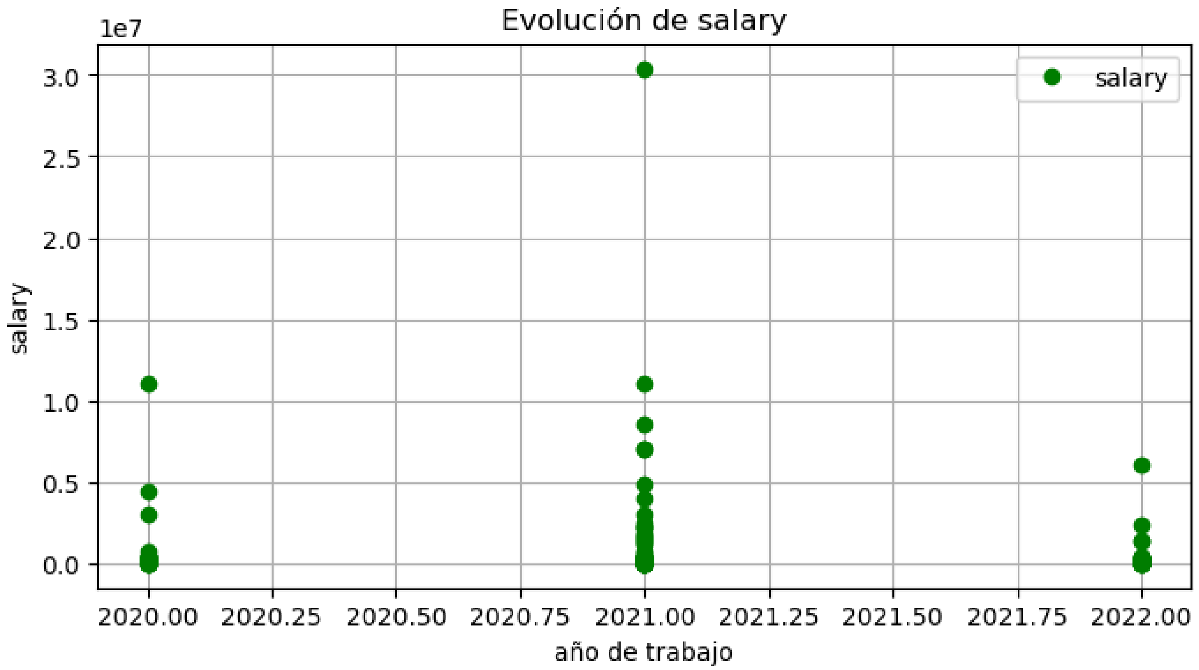
Out[109…

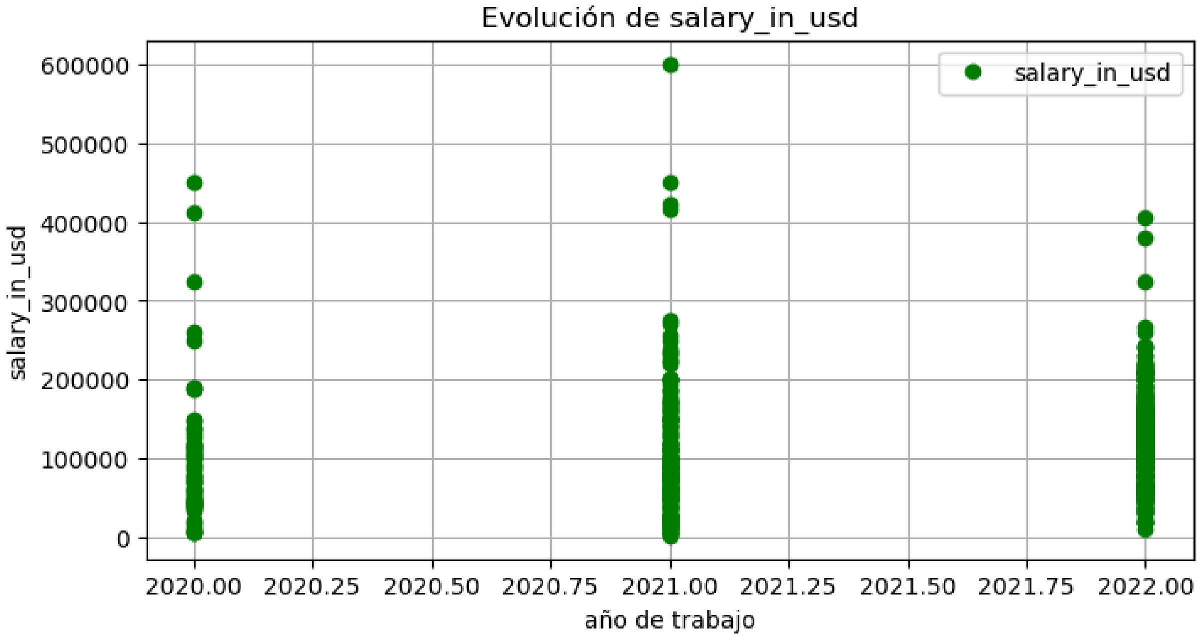
	work_year	salary	salary_in_usd	remote_ratio	salario en pesos
0	2020	70000	79833	0	315000000
1	2020	260000	260000	0	1170000000
2	2020	85000	109024	50	382500000
3	2020	20000	20000	0	90000000
4	2020	150000	150000	50	675000000
...	...	...	...	...	...
602	2022	154000	154000	100	693000000
603	2022	126000	126000	100	567000000
604	2022	129000	129000	0	580500000
605	2022	150000	150000	100	675000000
606	2022	200000	200000	100	900000000

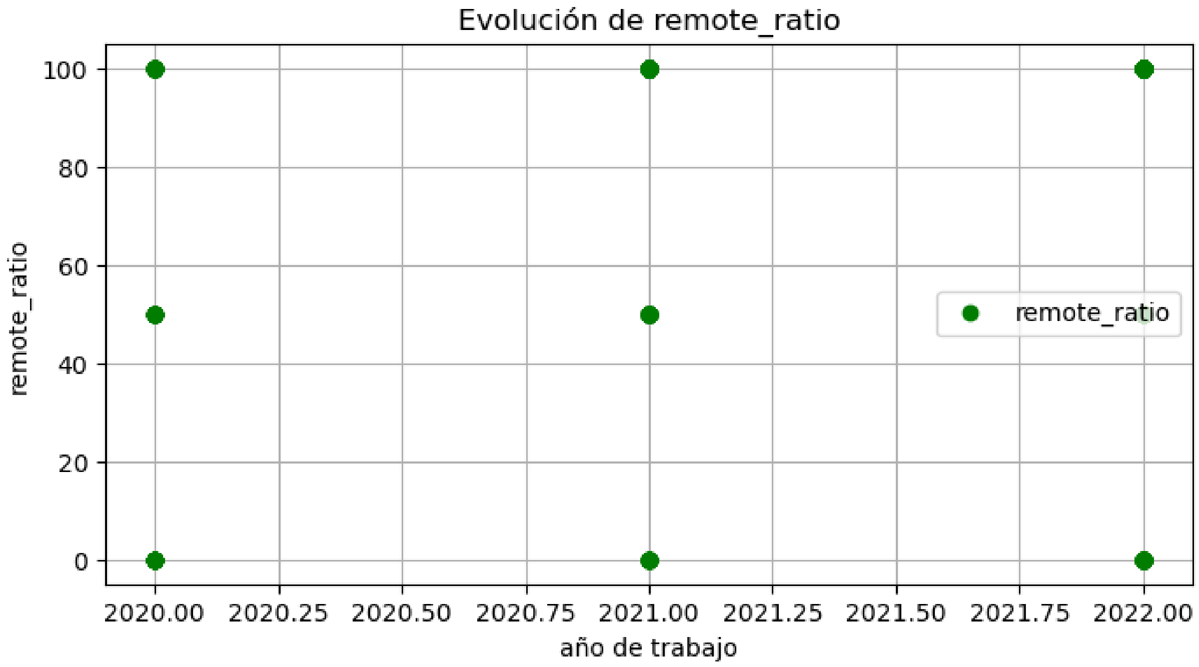
607 rows × 5 columns

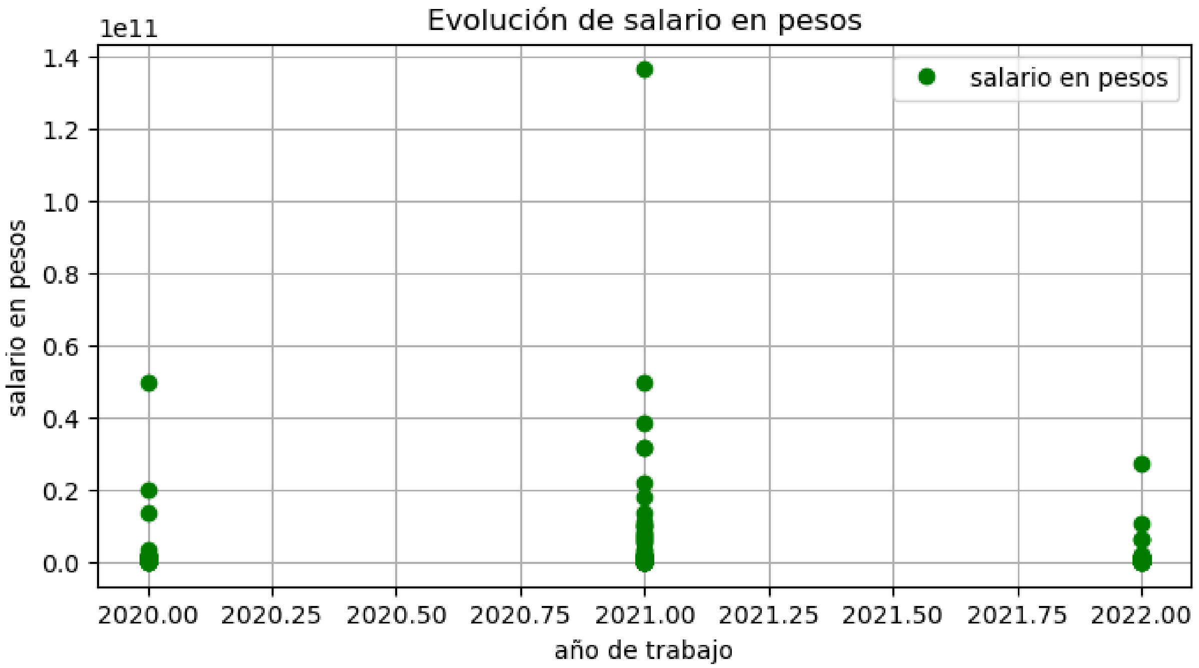
In [156…

```
for i in range(1, df_cuant.shape[1]): #ciclo para iterar sobre cada columna
    plt.figure(figsize=(8, 4))  # Crear una nueva figura para cada gráfico

    plt.plot(df_cuant.work_year, df_cuant.iloc[:, i], marker="o", linestyle="",colo

    # Personalización del gráfico
    plt.xlabel("año de trabajo")
    plt.ylabel(df_cuant.columns[i])
    plt.title(f"Evolución de {df_cuant.columns[i]}")
    plt.legend()
    plt.grid(True)

    plt.show()  # Mostrar cada gráf
```


Evolución de salary
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
salary
● salary
2020.00
2020.25
2020.50
2020.75
2021.00
2021.25
2021.50
2021.75
2022.00
año de trabajo


Evolución de salary_in_usd
600000
500000
400000
300000
200000
100000
0
salary_in_usd
● salary_in_usd
2020.00
2020.25
2020.50
2020.75
2021.00
2021.25
2021.50
2021.75
2022.00
año de trabajo





distribucion normal

```
In [101…  df_cuant= df_cuant.iloc[:,1:]
          df_cuant
```

Out[101…

	salary	salary_in_usd	remote_ratio	salario en pesos
0	70000	79833	0	315000000
1	260000	260000	0	1170000000
2	85000	109024	50	382500000
3	20000	20000	0	90000000
4	150000	150000	50	675000000
...	...	...	...	...
602	154000	154000	100	693000000
603	126000	126000	100	567000000
604	129000	129000	0	580500000
605	150000	150000	100	675000000
606	200000	200000	100	900000000

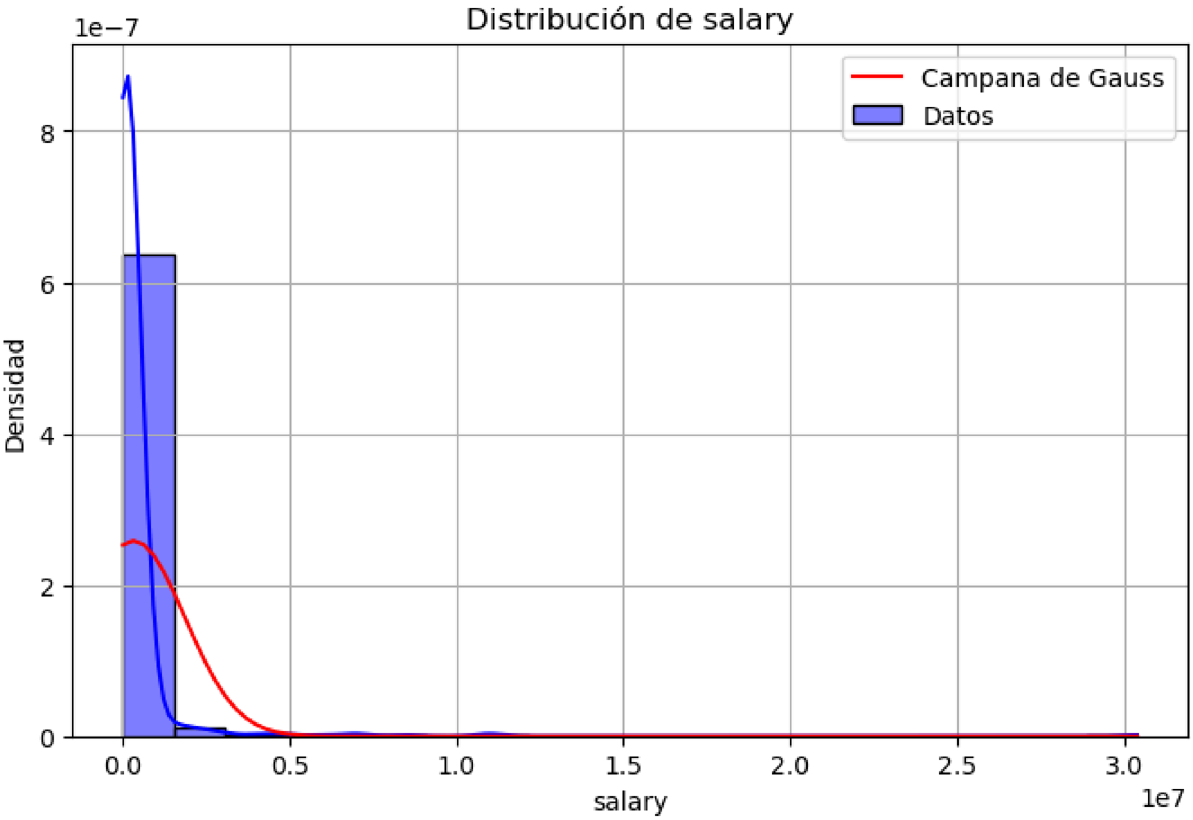
607 rows × 4 columns

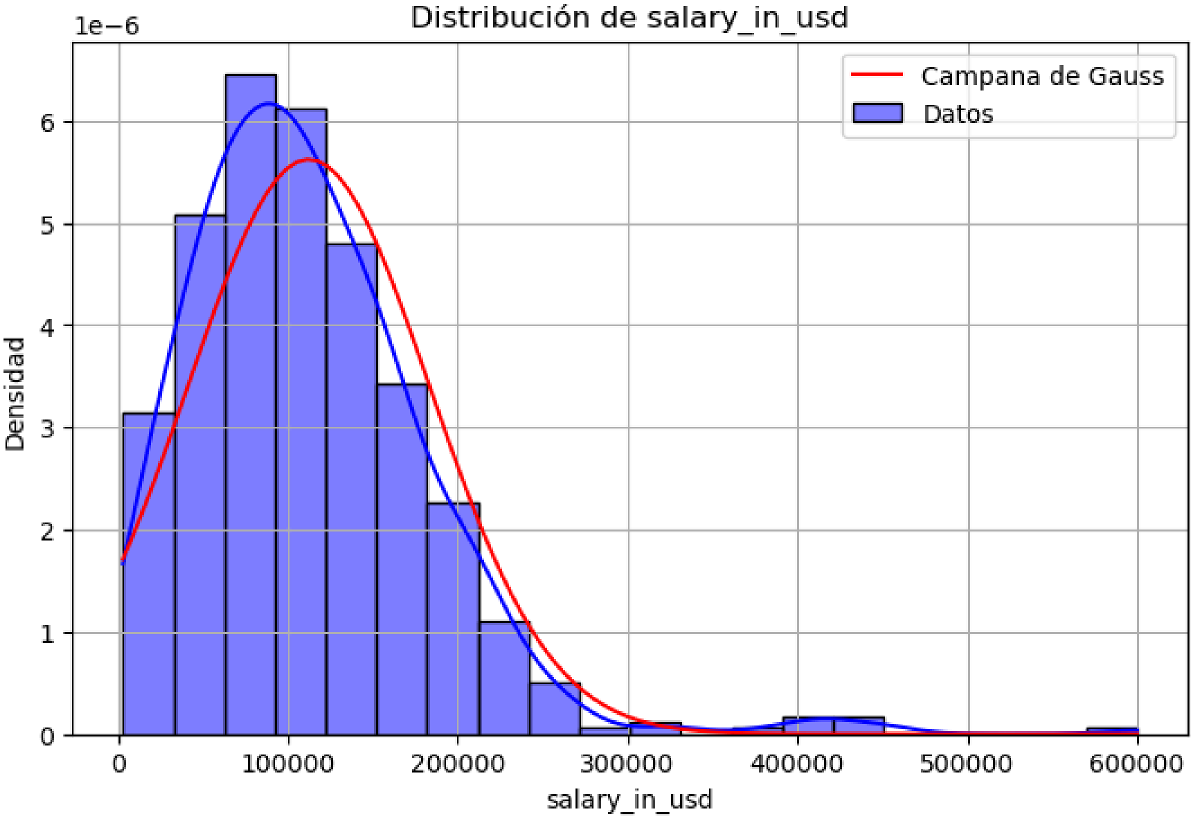
In [ ]:

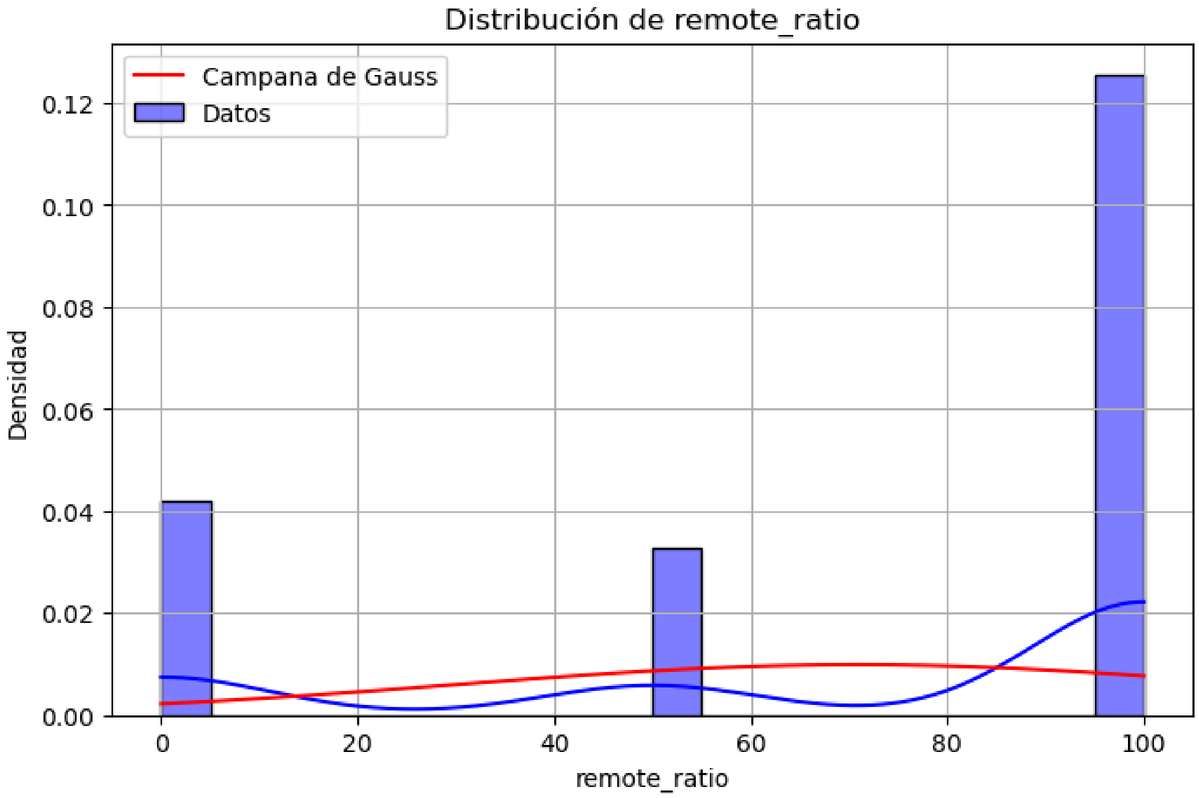
```
distribicón normal de los datos
```

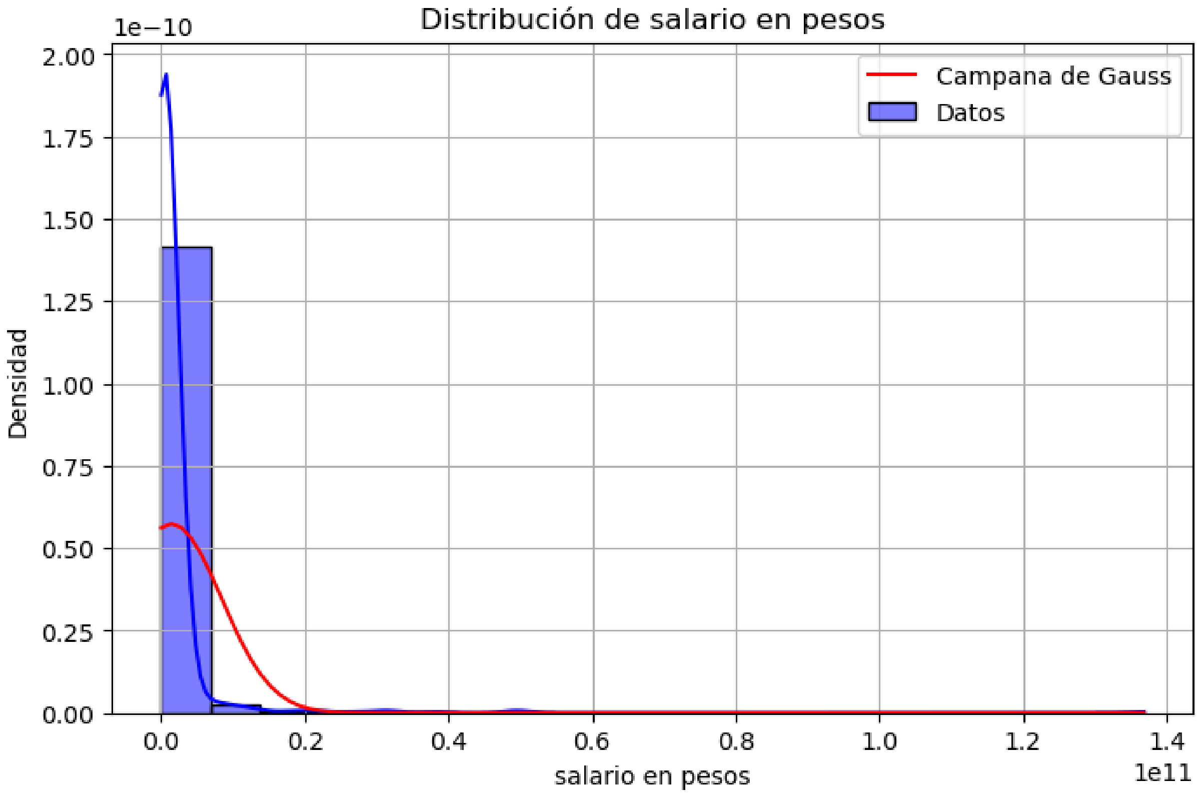
In [102…

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
import plotly.express as px



# Graficar cada variable numérica con su campana de Gauss
for columna in df_cuant.columns:
    plt.figure(figsize=(8, 5))  # Nueva figura para cada variable

    # Histograma con densidad
    sns.histplot(df_cuant[columna], kde=True, bins=20, stat="density", color="blue"

    # Ajuste de la curva normal teórica
    media = df_cuant[columna].mean()
    desviacion = df_cuant[columna].std()
    x = np.linspace(df_cuant[columna].min(), df_cuant[columna].max(), 100) #linea a
    y = norm.pdf(x, media, desviacion)
    plt.plot(x, y, color="red", label="Campana de Gauss")

    # Personalización del gráfico
    plt.title(f"Distribución de {columna}")
    plt.xlabel(columna)
    plt.ylabel("Densidad")
    plt.legend()
    plt.grid(True)
    plt.show()  # Muestra cada gráfico individualmente
```









la correlaccion entre los datos, sirve para revisarvla relacion de los datos

In [103…
```
correlacion = df_cuant.corr()
```

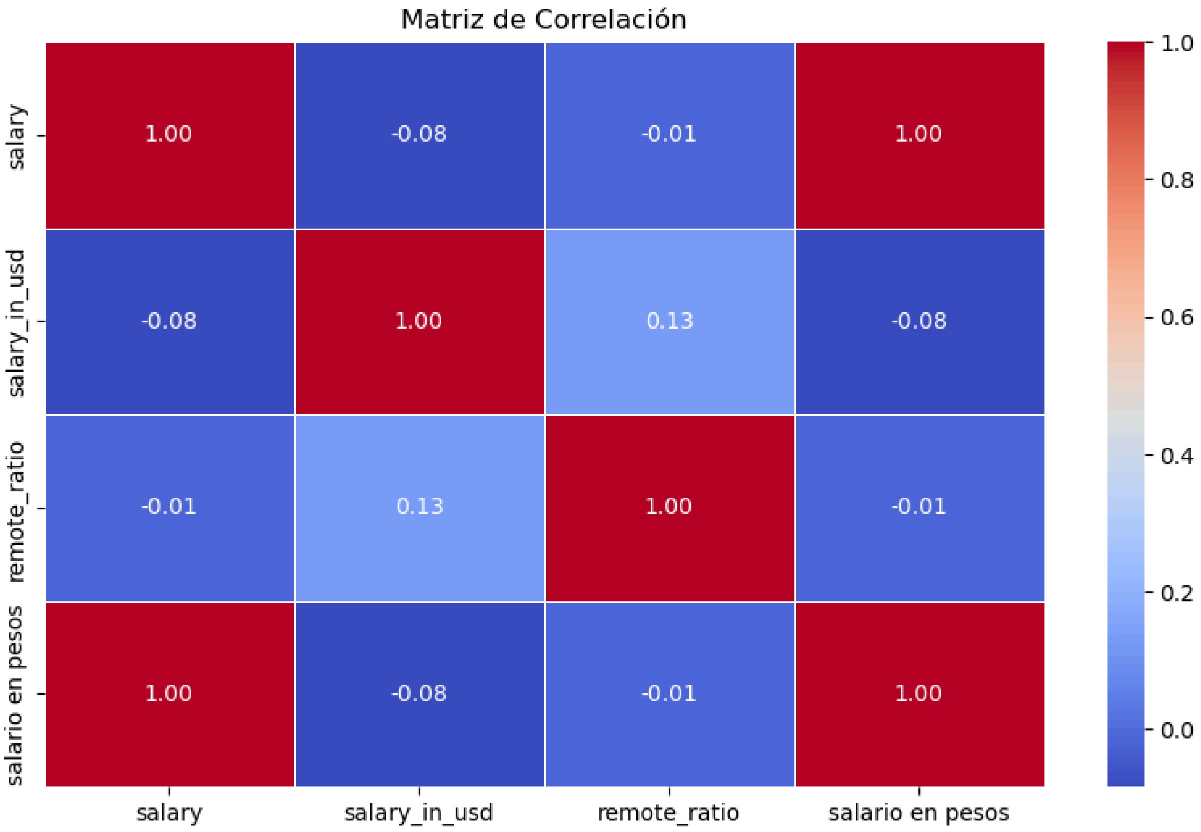
In [104…
```
correlacion
```

Out[104…

	salary	salary_in_usd	remote_ratio	salario en pesos
salary	1.000000	-0.083906	-0.014608	1.000000
salary_in_usd	-0.083906	1.000000	0.132122	-0.083906
remote_ratio	-0.014608	0.132122	1.000000	-0.014608
salario en pesos	1.000000	-0.083906	-0.014608	1.000000

In [141…

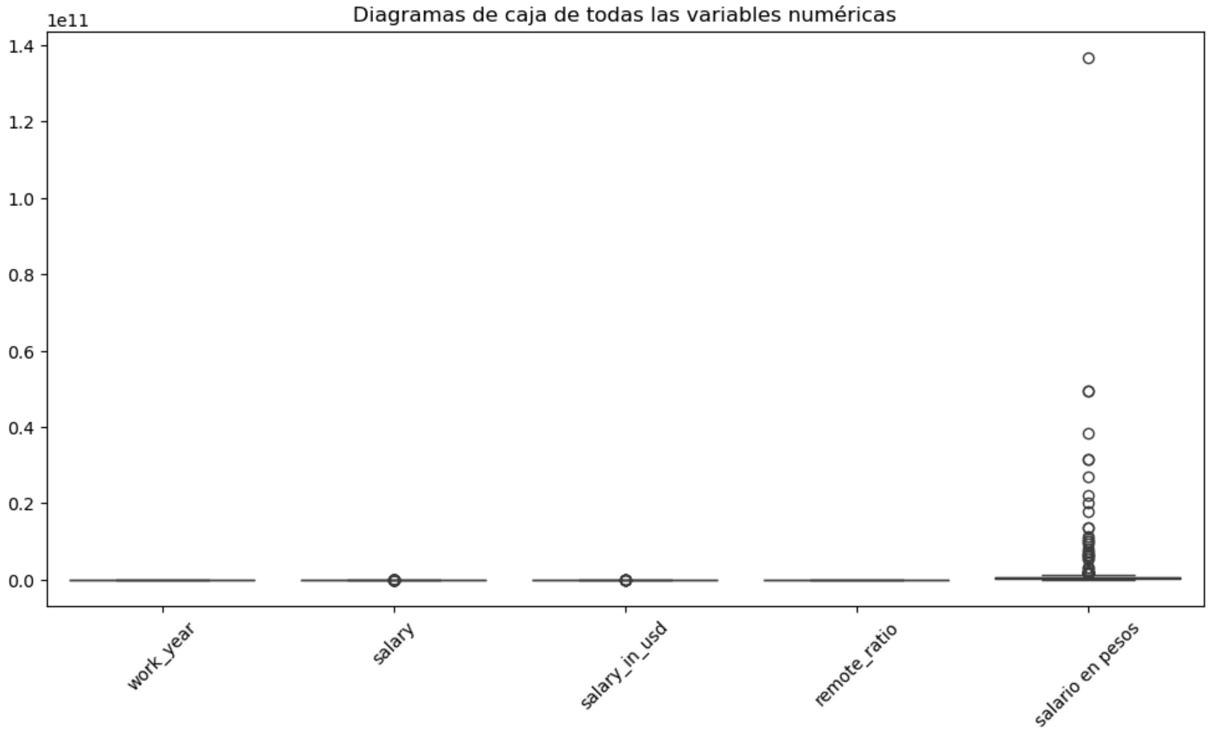
```
# 🔹 Crear el mapa de calor
plt.figure(figsize=(10, 6))  # Ajustar tamaño de la figura
sns.heatmap(correlacion, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)

# 🔹 Título del gráfico
plt.title("Matriz de Correlación")

# 🔹 Mostrar el gráfico
plt.show()
```



In [118…

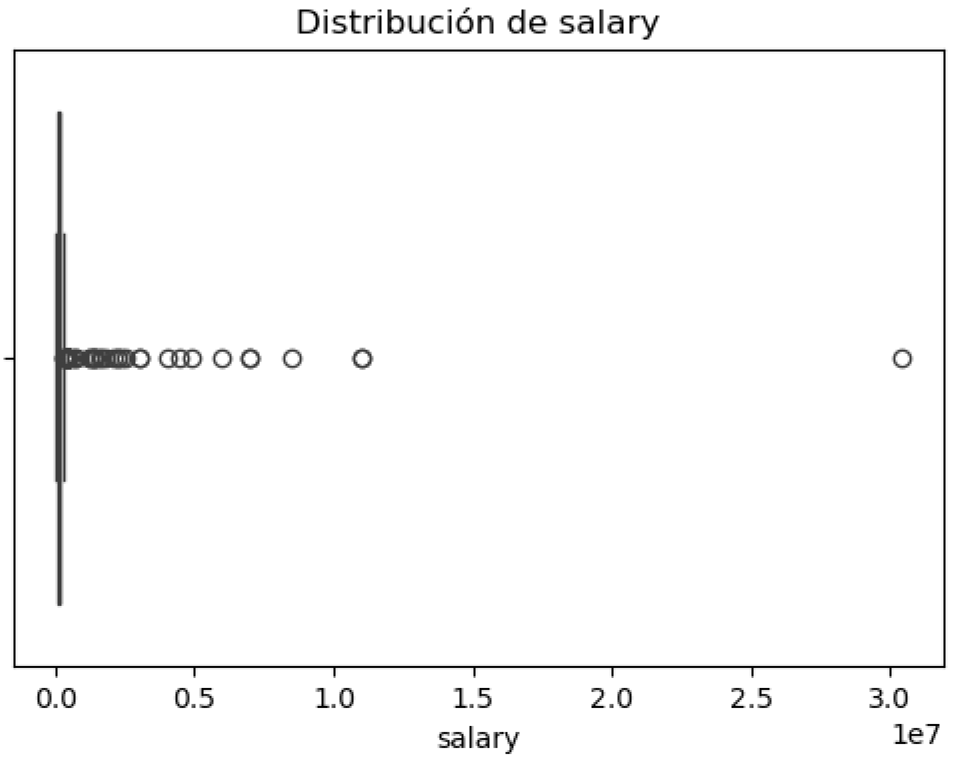
```
import seaborn as sns
import matplotlib.pyplot as plt

# 🔹 Seleccionar solo las columnas numéricas del DataFrame


# 🔹 Crear un boxplot para todas las columnas numéricas
plt.figure(figsize=(12,6))  # Tamaño del gráfico
sns.boxplot(df_cuant)
```

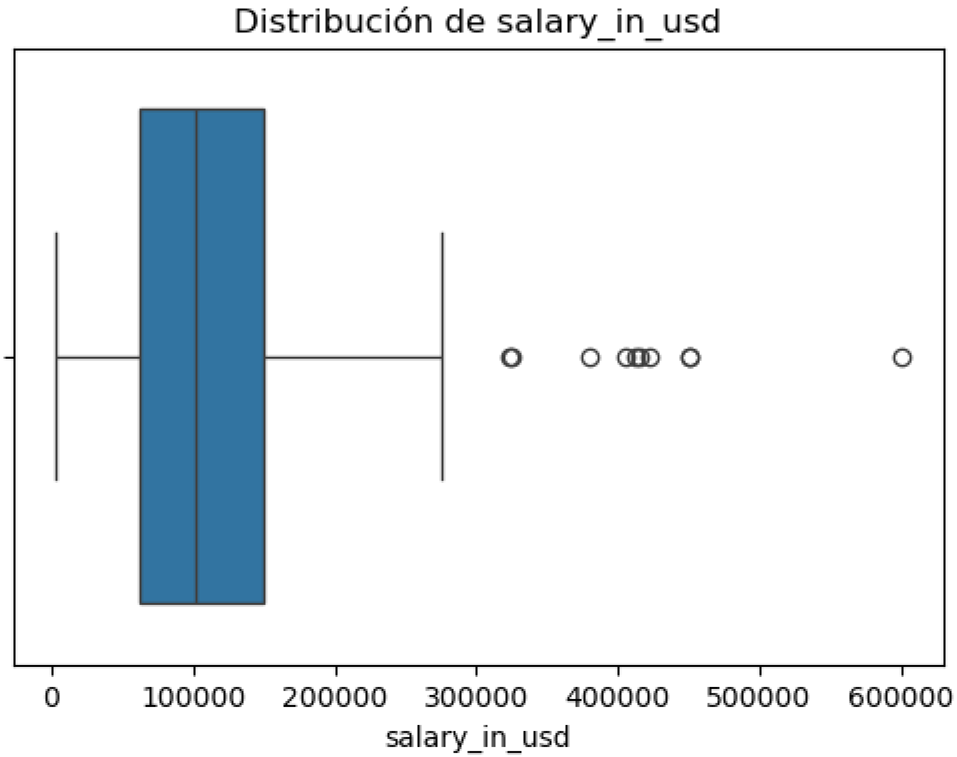
```
# 🔹 Mejorar visualización
plt.xticks(rotation=45)  # Rotar nombres de variables
plt.title("Diagramas de caja de todas las variables numéricas")

# 🔹 Mostrar gráfico
plt.show()
```

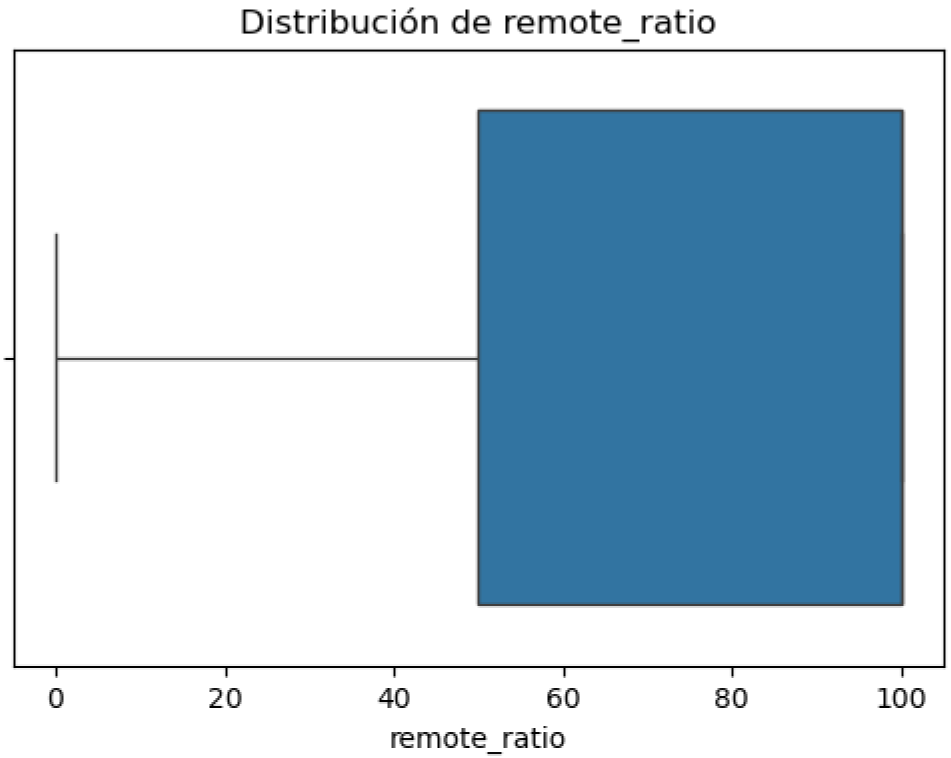


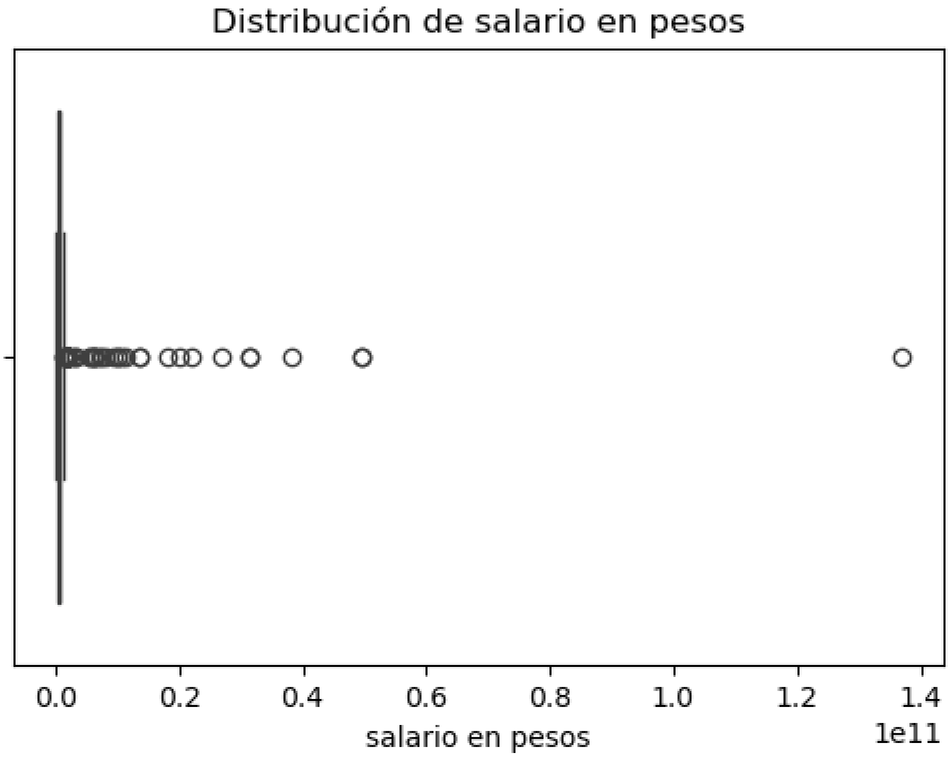
In [130…

```
# 🔹 Recorrer cada columna numérica y hacer un boxplot individual
for i in range(1, df_cuant.shape[1]):
    plt.figure(figsize=(6,4))  # Tamaño de cada gráfico
    sns.boxplot(x=df_cuant.iloc[:, i])
    plt.title(f"Distribución de {df_cuant.columns[i]}")  # Título con el nombre de
    plt.show()
```


Distribución de salary
0.0
0.5
1.0
1.5
2.0
2.5
3.0
1e7
salary


Distribución de salary_in_usd
0
100000
200000
300000
400000
500000
600000
salary_in_usd





```
In [131…  df_cuant.describe()
```

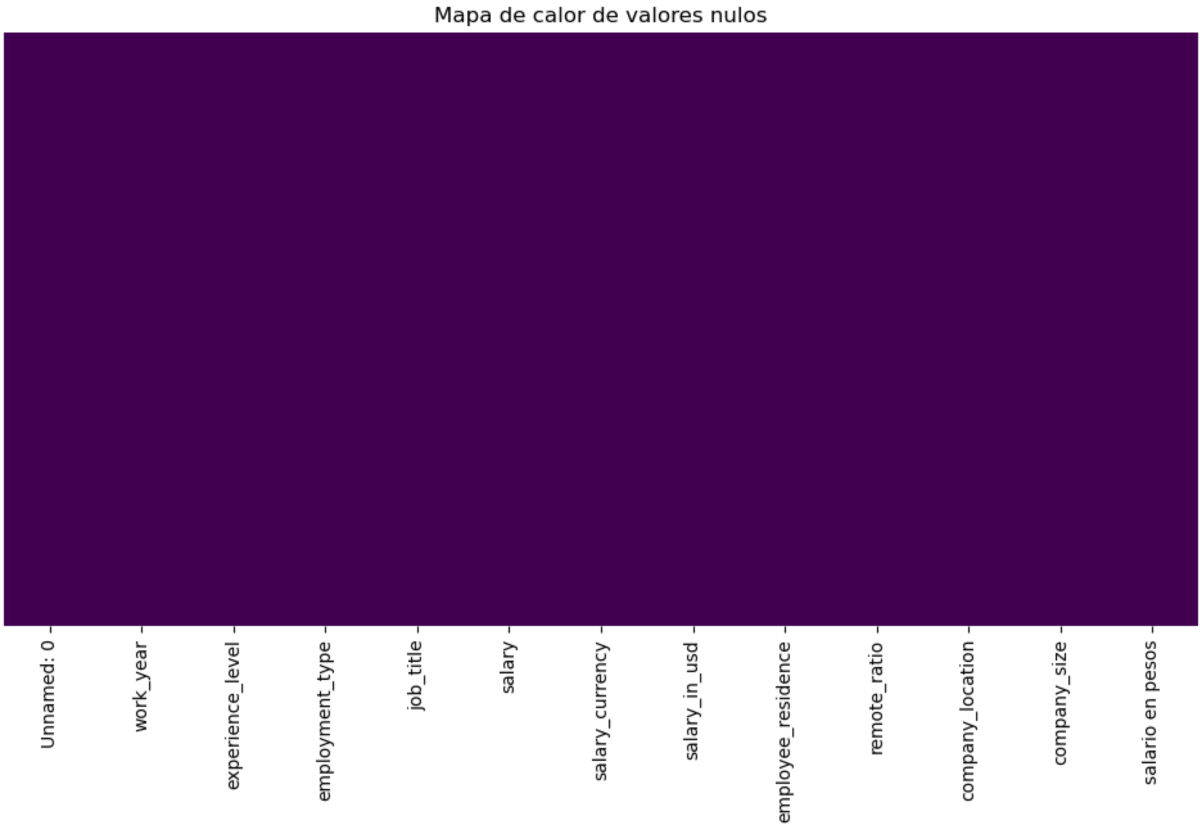
Out[131…

	work_year	salary	salary_in_usd	remote_ratio	salario en pesos
count	607.000000	6.070000e+02	607.000000	607.00000	6.070000e+02
mean	2021.405272	3.240001e+05	112297.869852	70.92257	1.458000e+09
std	0.692133	1.544357e+06	70957.259411	40.70913	6.949609e+09
min	2020.000000	4.000000e+03	2859.000000	0.00000	1.800000e+07
25%	2021.000000	7.000000e+04	62726.000000	50.00000	3.150000e+08
50%	2022.000000	1.150000e+05	101570.000000	100.00000	5.175000e+08
75%	2022.000000	1.650000e+05	150000.000000	100.00000	7.425000e+08
max	2022.000000	3.040000e+07	600000.000000	100.00000	1.368000e+11

vaores nulos en la data

In [133…

```
plt.figure(figsize=(12,6))
sns.heatmap(df.isnull(), cmap="viridis", cbar=False, yticklabels=False)
plt.title("Mapa de calor de valores nulos")
plt.show()
```



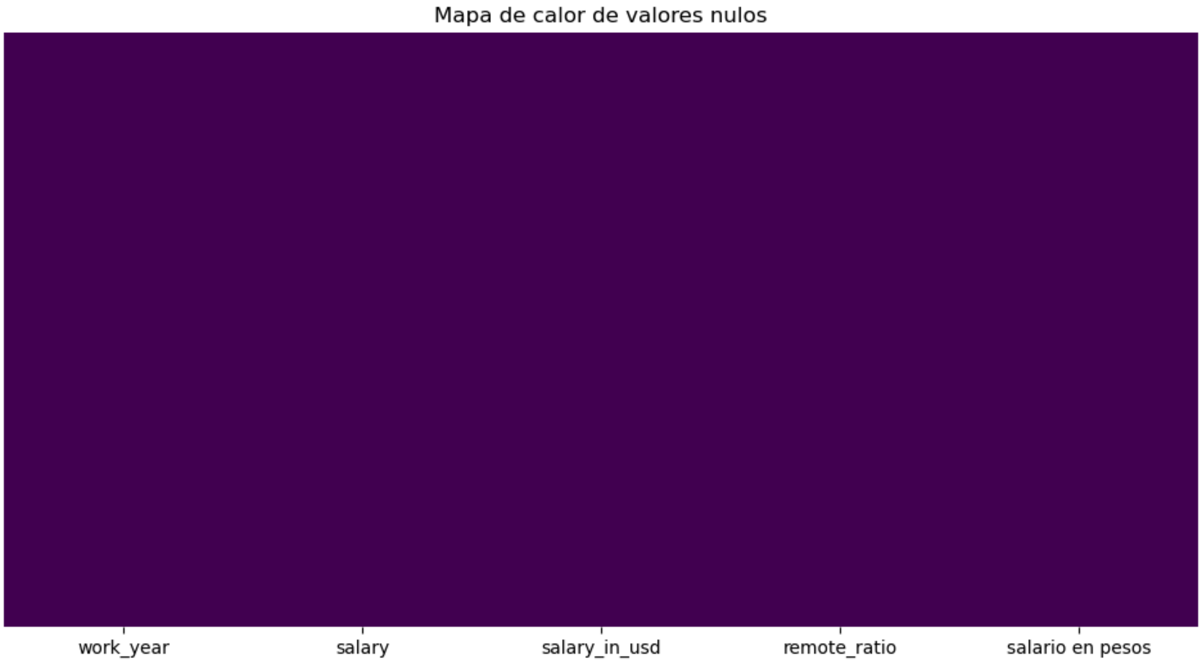
In [ ]:

```
los espacios en blanco son nulos
```

In [134…

```
plt.figure(figsize=(12,6))
sns.heatmap(df_cuant.isnull(), cmap="viridis", cbar=False, yticklabels=False)
```

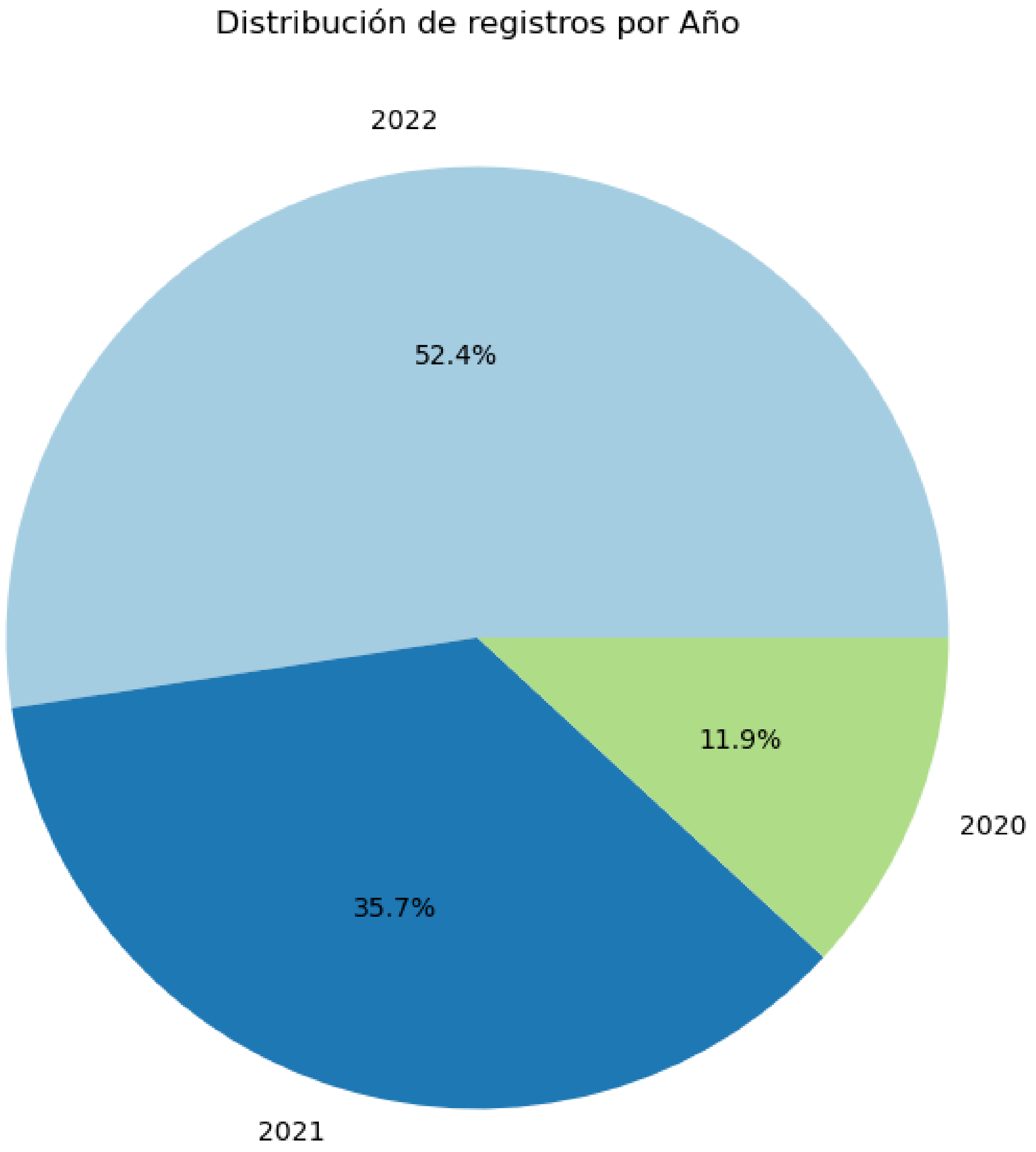
```
plt.title("Mapa de calor de valores nulos")
plt.show()
```



In [140…

```
# Contar cuántos registros hay por año
conteo_años = df["work_year"].value_counts()
print(conteo_años)

# Crear el gráfico de torta
plt.figure(figsize=(8,8))
plt.pie(conteo_años, labels=conteo_años.index, autopct="%1.1f%%", colors=plt.cm.Pai

# Título y mostrar gráfico
plt.title("Distribución de registros por Año")
plt.show()
```

```
work_year
2022    318
2021    217
2020     72
Name: count, dtype: int64
```



guardar fichero

```
In [160…  df_cuant.to_csv("C:/Users/darly/OneDrive/Escritorio/materialClaseIA/datos.csv", ind
```

```
In [158…  data = {
              "fecha": ["2016-04-18 06:00:00", "2016-04-19 06:00:00", "2016-04-20 06:00:00"],
              "valor": ["7,33", "8,21", "6,75"]
          }
```

```
In [159…  data= pd.DataFrame(data)
          data
```

Out[159…

	fecha	valor
0	2016-04-18 06:00:00	7,33
1	2016-04-19 06:00:00	8,21
2	2016-04-20 06:00:00	6,75

In [ ]: