In [1]:
```
# Import libraries.
import pandas as pd
import numpy as np
import scipy.stats as stats

import warnings
warnings.filterwarnings('ignore')
```

Load All 5 Datasets

In [2]:
```
# Loads the world dataset.
data1 = pd.read_csv('jobs_in_data.csv')

# Creates a copy of the dataset.
world_df2 = data1.copy()

# Displays the first 5 records.
world_df2.head()
```

Out[2]:

	work_year	job_title	job_category	salary_currency	salary	salary_in_usd	employee_res
0	2023	Data DevOps Engineer	Data Engineering	EUR	88000	95012	G
1	2023	Data Architect	Data Architecture and Modeling	USD	186000	186000	United
2	2023	Data Architect	Data Architecture and Modeling	USD	81800	81800	United
3	2023	Data Scientist	Data Science and Research	USD	212000	212000	United
4	2023	Data Scientist	Data Science and Research	USD	93300	93300	United

In [113]:

```
# Loads the United States only dataset.
data2 = pd.read_csv(
    'Data Analyst Salaries in The USA.csv')

# Creates a copy of the dataset.
usa_df2 = data2.copy()

# Displays the first 5 records.
usa_df2.head()
```

Out[113]:

	Title	Min Salary	Max Salary	Salary Period	Company Name	State	Remote
0	Carbon data analyst	NaN	NaN	NaN	NaN	NaN	No
1	Data Analyst	80783.0	103333.0	Yearly	DC Public Library	NaN	No
2	Data Reporting Analyst III	75300.0	100800.0	Yearly	Horizon Blue Cross Blue Shield of New Jersey	NaN	No
3	Senior Data Analyst	78700.0	163400.0	Yearly	First American Financial Corporation	CA	Yes
4	Product Data Analyst	95200.0	121000.0	Yearly	Concept Art House	NaN	Yes

In [4]:

```
# Loads the United Kingdom only dataset.
data3 = pd.read_csv('deduped-jobs.csv')

# Creates a copy of the dataset.
uk_df2 = data3.copy()

# Displays the first 5 records.
uk_df2.head()
```

Out[4]:

	reference	title	date_posted	date_ending	advertiser	location	city	co
0	41857664	Data Science Manager	2021-01-26	2021-03-09T23:55:00.0000000	Charles Simon Associates Ltd	London	Camden	
1	41924233	Data Science Recruiter	2021-02-03	2021-03-03T23:55:00.0000000	Crone Corkill	South East England	London	
2	41752222	Data Science Lead	2021-01-14	2021-02-25T23:55:00.0000000	Harnham	South East England	London	
3	41642513	Data Science Consultant	2020-12-27	2021-02-07T23:55:00.0000000	QUINTON DAVIES LIMITED	Avon	Bristol	
4	41764338	Data Science Manager	2021-01-15	2021-02-26T23:55:00.0000000	Data Idols	South East England	London	

In [5]:

```
# Loads the international dataset.
data4 = pd.read_csv('ds_salaries.csv')

# Creates a copy of the dataset.
int_df2 = data4.copy()

# Displays the first 5 records.
int_df2.head()
```

Out[5]:

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

In [6]:
```
# Loads the Canada only dataset.
data5 = pd.read_csv('Job_list_Canada.csv')

# Creates a copy of the dataset.
can_df2 = data5.copy()

# Displays the first 5 records.
can_df2.head()
```

Out[6]:

	JobTitle	Company	Location	Salary	PostDate	Summary	
0	Machine Learning Engineer	Boast Capital	Montréal, QC	NaN	3 days ago	Fullstack development of AI features and appli...	https://ca.indeed.c mc
1	Machine Learning Engineer (Canada, Remote)	Diligen	Toronto, ON	$74,062–$188,358 a year	2 days ago	Experience in machine learning techniques for ...	https://ca.indeed.c mc
2	Senior Machine Learning Engineer - InSight - OPS	Iron Mountain	Remote	NaN	13 days ago	The ML engineer works closely with multiple in...	https://ca.indeed.c mc
3	Jr. Machine Learning Engineer	Niricson Software Inc.	Vancouver, BC	$50,000–$60,000 a year	5 days ago	2+ years of industry experience in ML modellin...	https://ca.indeed.com/cor
4	Research Engineer, Vision	Facebook	Montréal, QC	NaN	30+ days ago	Experience developing machine learning algorit...	https://ca.indeed.c mc

Data Exploration

In [7]:
```
# Displays calculations of values.
world_df2.describe()
```

Out[7]:

	work_year	salary	salary_in_usd
count	9355.000000	9355.000000	9355.000000
mean	2022.760449	149927.981293	150299.495564
std	0.519470	63608.835387	63177.372024
min	2020.000000	14000.000000	15000.000000
25%	2023.000000	105200.000000	105700.000000
50%	2023.000000	143860.000000	143000.000000
75%	2023.000000	187000.000000	186723.000000
max	2023.000000	450000.000000	450000.000000

In [8]:
```
# Displays calculations of values.
usa_df2.describe()
```

Out[8]:

	Min Salary	Max Salary
count	738.000000	738.000000
mean	69904.707358	92407.143076
std	27360.306143	38766.509993
min	15.000000	15.000000
25%	59682.500000	76725.000000
50%	70900.000000	92650.000000
75%	85600.000000	111000.000000
max	151800.000000	245700.000000

In [9]:
```
# Displays calculations of values.
uk_df2.describe()
```

Out[9]:

	reference	salary	salary_min	salary_max	salary_currency
count	5.950000e+02	595.000000	595.000000	595.000000	0.0
mean	4.178603e+07	57037.773109	57037.773109	72586.292437	NaN
std	1.298250e+05	21730.281996	21730.281996	37994.233441	NaN
min	4.098746e+07	10000.000000	10000.000000	10000.000000	NaN
25%	4.170866e+07	40000.000000	40000.000000	55000.000000	NaN
50%	4.179374e+07	55000.000000	55000.000000	70000.000000	NaN
75%	4.189485e+07	70000.000000	70000.000000	85000.000000	NaN
max	4.195368e+07	150000.000000	150000.000000	750000.000000	NaN

In [10]:
```
# Displays calculations of values.
int_df2.describe()
```

Out[10]:

	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

In [11]:
```
# Displays calculations of values.
can_df2.describe()
```

Out[11]:

	JobTitle	Company	Location	Salary	PostDate	Summary	Jo
count	2971	2971	2971	444	2971	2971	2
unique	1421	1030	142	176	32	1811	2
top	Data Analyst	AbCellera Biologics	Toronto, ON	80, 000– 120,000 a year	30+ days ago	At AbCellera, we're solving tough problems and...	https://ca.indeed.com/rc/ jk=59f13e9da13b
freq	96	69	722	30	1251	27	

In [12]:
```
# Displays general information.
world_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9355 entries, 0 to 9354
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   work_year           9355 non-null   int64 
 1   job_title           9355 non-null   object
 2   job_category        9355 non-null   object
 3   salary_currency     9355 non-null   object
 4   salary              9355 non-null   int64 
 5   salary_in_usd       9355 non-null   int64 
 6   employee_residence  9355 non-null   object
 7   experience_level    9355 non-null   object
 8   employment_type     9355 non-null   object
 9   work_setting        9355 non-null   object
 10  company_location    9355 non-null   object
 11  company_size        9355 non-null   object
dtypes: int64(3), object(9)
memory usage: 877.2+ KB
```

In [13]:
```
# Displays general information.
usa_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 843 entries, 0 to 842
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Title          843 non-null    object 
 1   Min Salary     738 non-null    float64
 2   Max Salary     738 non-null    float64
 3   Salary Period  750 non-null    object 
 4   Company Name   819 non-null    object 
 5   State          536 non-null    object 
 6   Remote         843 non-null    object 
dtypes: float64(2), object(5)
memory usage: 46.2+ KB
```

In [14]:
```
# Displays general information.
uk_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 595 entries, 0 to 594
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   reference         595 non-null    int64
 1   title             595 non-null    object
 2   date_posted       595 non-null    object
 3   date_ending       595 non-null    object
 4   advertiser        595 non-null    object
 5   location          595 non-null    object
 6   city              595 non-null    object
 7   country           595 non-null    object
 8   salary            595 non-null    float64
 9   salary_min        595 non-null    float64
 10  salary_max        595 non-null    float64
 11  salary_frequency  595 non-null    object
 12  salary_currency   0 non-null      float64
 13  description       595 non-null    object
dtypes: float64(4), int64(1), object(9)
memory usage: 65.2+ KB
```

In [15]:
```
# Displays general information.
int_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 607 entries, 0 to 606
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Unnamed: 0          607 non-null    int64
 1   work_year           607 non-null    int64
 2   experience_level    607 non-null    object
 3   employment_type     607 non-null    object
 4   job_title           607 non-null    object
 5   salary              607 non-null    int64
 6   salary_currency     607 non-null    object
 7   salary_in_usd       607 non-null    int64
 8   employee_residence  607 non-null    object
 9   remote_ratio        607 non-null    int64
 10  company_location    607 non-null    object
 11  company_size        607 non-null    object
dtypes: int64(5), object(7)
memory usage: 57.0+ KB
```

In [16]:
```
# Displays general information.
can_df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2971 entries, 0 to 2970
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   JobTitle  2971 non-null   object
 1   Company   2971 non-null   object
 2   Location  2971 non-null   object
 3   Salary    444 non-null    object
 4   PostDate  2971 non-null   object
 5   Summary   2971 non-null   object
 6   JobUrl    2971 non-null   object
dtypes: object(7)
memory usage: 162.6+ KB
```

Data Cleaning

Retain Only Data Analyst Jobs

In [17]:
```
# Converts all letters in the job title to
# uppercase and displays the first 5 rows.
world_df2['job_title'] = world_df2[
    'job_title'].apply(
    lambda x: x.upper()).copy()

world_df2.head()
```

Out[17]:

	work_year	job_title	job_category	salary_currency	salary	salary_in_usd	employee_
0	2023	DATA DEVOPS ENGINEER	Data Engineering	EUR	88000	95012	
1	2023	DATA ARCHITECT	Data Architecture and Modeling	USD	186000	186000	Ur
2	2023	DATA ARCHITECT	Data Architecture and Modeling	USD	81800	81800	Ur
3	2023	DATA SCIENTIST	Data Science and Research	USD	212000	212000	Ur
4	2023	DATA SCIENTIST	Data Science and Research	USD	93300	93300	Ur

In [18]:

```
# Creates a copy of dataframe with rows
# that the job title attribute
# contains the word 'ANALYST'
# and displays the first 5 rows.
world_df = world_df2[
    world_df2['job_title'].str.contains(
    'ANALYST')].copy()

world_df.head()
```

Out[18]:

	work_year	job_title	job_category	salary_currency	salary	salary_in_usd	employee_r
15	2023	DATA ANALYST	Data Analysis	USD	95000	95000	Unit
16	2023	DATA ANALYST	Data Analysis	USD	75000	75000	Unit
23	2023	DATA ANALYST	Data Analysis	USD	155000	155000	Unit
24	2023	DATA ANALYST	Data Analysis	USD	110000	110000	Unit
41	2023	DATA ANALYST	Data Analysis	USD	176000	176000	Unit

In [19]:

```
# Calculates the difference in
# number of records removed from
# the dataframe and displays the results.
difference1 = len(world_df2) - len(world_df)

print("\n\tThere were", difference1,
      "records removed from the dataframe\n")
```

```
	There were 7742 records removed from the dataframe
```

In [20]:

```
# Converts all letters in the job title to
# uppercase and displays the first 5 rows.
usa_df2['Title'] = usa_df2['Title'].apply(
    lambda x: x.upper()).copy()

usa_df2.head()
```

Out[20]:

	Title	Min Salary	Max Salary	Salary Period	Company Name	State	Remote
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	NaN	No
1	DATA ANALYST	80783.0	103333.0	Yearly	DC Public Library	NaN	No
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	Horizon Blue Cross Blue Shield of New Jersey	NaN	No
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	First American Financial Corporation	CA	Yes
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	Concept Art House	NaN	Yes

In [21]:

```
# Creates a copy of dataframe with
# rows that the job title attribute
# contains the word 'ANALYST'
# and displays the first 5 rows.
usa_df = usa_df2[
    usa_df2['Title'].str.contains(
    'ANALYST')].copy()

usa_df.head()
```

Out[21]:

	Title	Min Salary	Max Salary	Salary Period	Company Name	State	Remote
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	NaN	No
1	DATA ANALYST	80783.0	103333.0	Yearly	DC Public Library	NaN	No
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	Horizon Blue Cross Blue Shield of New Jersey	NaN	No
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	First American Financial Corporation	CA	Yes
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	Concept Art House	NaN	Yes

In [22]:

```
# Calculates the difference in
# number of records removed from
# the dataframe and displays the results.
difference2 = len(usa_df2) - len(usa_df)

print("\n\tThere were", difference2,
      "records removed from the dataframe\n")
```

```
	There were 7 records removed from the dataframe
```

In [23]:

```
# Converts all letters in the job title to
# uppercase and displays the first 5 rows.
uk_df2['title'] = uk_df2['title'].apply(
    lambda x: x.upper()).copy()

uk_df2.head()
```

Out[23]:

	reference	title	date_posted	date_ending	advertiser	location	
0	41857664	DATA SCIENCE MANAGER	2021-01-26	2021-03-09T23:55:00.0000000	Charles Simon Associates Ltd	London	Cam
1	41924233	DATA SCIENCE RECRUITER	2021-02-03	2021-03-03T23:55:00.0000000	Crone Corkill	South East England	Lon
2	41752222	DATA SCIENCE LEAD	2021-01-14	2021-02-25T23:55:00.0000000	Harnham	South East England	Lon
3	41642513	DATA SCIENCE CONSULTANT	2020-12-27	2021-02-07T23:55:00.0000000	QUINTON DAVIES LIMITED	Avon	Bri

In [24]:

```
# Creates a copy of dataframe with
# rows that the job title attribute
# contains the word 'ANALYST'
# and displays the first 5 rows.
uk_df = uk_df2[
    uk_df2['title'].str.contains(
    'ANALYST')].copy()

uk_df.head()
```

Out[24]:

	reference	title	date_posted	date_ending	advertiser	location	
77	41730933	DATA SCIENTIST / DATA ANALYST	2021-01-12	2021-02-23T23:55:00.0000000	Cameo Consultancy	Oxfordshire	Ban
87	41666705	INSIGHT ANALYST	2021-01-04	2021-02-15T23:55:00.0000000	Harnham	West Yorkshire	Le
106	41863877	PATIENT DATA ANALYST	2021-01-27	2021-03-03T23:55:00.0000000	Hyper Recruitment Solutions Ltd	South East England	Berks
110	41813983	SENIOR INSIGHT ANALYST	2021-01-21	2021-03-04T23:55:00.0000000	Harnham	South East England	Lor
111	41651502	SENIOR INSIGHT ANALYST	2020-12-30	2021-02-10T23:55:00.0000000	Harnham	South East England	Lor

In [25]:

```
# Calculates the difference in
# number of records removed from
# the dataframe and displays the results.
difference3 = len(uk_df2) - len(uk_df)

print("\n\tThere were", difference3,
      "records removed from the dataframe\n")
```

```
	There were 533 records removed from the dataframe
```

In [26]:

```
# Converts all letters in the job title to
# uppercase and displays the first 5 rows.
int_df2['job_title'] = int_df2['job_title'].apply(
    lambda x: x.upper()).copy()

int_df2.head()
```

Out[26]:

	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	

In [27]:

```
# Creates a copy of dataframe with
# rows that the job title attribute
# contains the word 'ANALYST'
# and displays the first 5 rows.
int_df = int_df2[
    int_df2['job_title'].str.contains(
    'ANALYST')].copy()

int_df.head()
```

Out[27]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_cu
3	3	2020	MI	FT	PRODUCT DATA ANALYST	20000	
5	5	2020	EN	FT	DATA ANALYST	72000	
8	8	2020	MI	FT	BUSINESS DATA ANALYST	135000	
13	13	2020	MI	FT	LEAD DATA ANALYST	87000	
14	14	2020	MI	FT	DATA ANALYST	85000	

In [28]:

```
# Calculates the difference in
# number of records removed from
# the dataframe and displays the results.
difference4 = len(int_df2) -len(int_df)

print("\n\tThere were", difference4,
      "records removed from the dataframe\n")
```

```
	There were 488 records removed from the dataframe
```

In [29]:

```
# Converts all letters in the job title to
# uppercase and displays the first 5 rows.
can_df2['JobTitle'] = can_df2['JobTitle'].apply(
    lambda x: x.upper()).copy()

can_df2.head()
```

Out[29]:

	JobTitle	Company	Location	Salary	PostDate	Summary	
0	MACHINE LEARNING ENGINEER	Boast Capital	Montréal, QC	NaN	3 days ago	Fullstack development of AI features and appli...	https://ca.indeed
1	MACHINE LEARNING ENGINEER (CANADA, REMOTE)	Diligen	Toronto, ON	74, 062– 188,358 a year	2 days ago	Experience in machine learning techniques for ...	https://ca.indeed
2	SENIOR MACHINE LEARNING ENGINEER - INSIGHT - OPS	Iron Mountain	Remote	NaN	13 days ago	The ML engineer works closely with multiple in...	https://ca.indeed
3	JR. MACHINE LEARNING ENGINEER	Niricson Software Inc.	Vancouver, BC	50, 000– 60,000 a year	5 days ago	2+ years of industry experience in ML modellin...	https://ca.indeed.com/
4	RESEARCH ENGINEER, VISION	Facebook	Montréal, QC	NaN	30+ days ago	Experience developing machine learning algorit...	https://ca.indeed

In [30]:

```
# Creates a copy of dataframe with
# rows that the job title attribute
# contains the word 'ANALYST'
# and displays the first 5 rows.
can_df = can_df2[
    can_df2['JobTitle'].str.contains(
    'ANALYST')].copy()

can_df.head()
```

Out[30]:

	JobTitle	Company	Location	Salary	PostDate	Summary	
99	SENIOR STATISTICAL/ACTUARIAL ANALYST	Aviva	Montréal, QC	NaN	17 days ago	You will work on the proposal of innovative ma...	https://ca.i jk=(
139	SENIOR STATISTICAL/ACTUARIAL ANALYST	Aviva	Montréal, QC	NaN	17 days ago	You will work on the proposal of innovative ma...	https://ca.i jk=(
168	COMPENSATION ANALYST	AbCellera Biologics	Vancouver, BC	NaN	12 days ago	Our ideal candidate is results-driven and deta...	https://ca.i jk=f
194	COMPENSATION ANALYST	AbCellera	Vancouver, BC	NaN	11 days ago	Our ideal candidate is results-driven and deta...	https://ca.i jk=2
214	CORPORATE DEVELOPMENT ANALYST	AbCellera	Vancouver, BC	NaN	11 days ago	Proactively developing a broad knowledge of an...	https://ca.i jk=f

In [31]:

```
difference5 = len(can_df2) - len(can_df)

print("\n\tThere were", difference5,
      "records removed from the dataframe\n")
```

```
There were 1733 records removed from the dataframe
```

Converts & Creates Uniform Columns in dataframes

In [32]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
world_df.drop(world_df.columns[[2, 3, 4, 10]],
              axis = 1, inplace = True)

world_df.head()
```

Out[32]:

	work_year	job_title	salary_in_usd	employee_residence	experience_level	employment_
15	2023	DATA ANALYST	95000	United States	Entry-level	Ful
16	2023	DATA ANALYST	75000	United States	Entry-level	Ful
23	2023	DATA ANALYST	155000	United States	Mid-level	Ful
24	2023	DATA ANALYST	110000	United States	Mid-level	Ful
41	2023	DATA ANALYST	176000	United States	Senior	Ful

In [33]:

```
# Creates columns and populate with data
# based upon attributes, structure,
# and origin of the dataframe for uniformity,
# as well as displays the first 5 rows.
usa_df['experience_level'] = 'Mid-level'
usa_df['employee_residence'] = 'United States'
usa_df['employment_type'] = 'Full-time'
usa_df['work_year'] = '2022'

usa_df.head()
```

Out[33]:

	Title	Min Salary	Max Salary	Salary Period	Company Name	State	Remote	experience_level	em
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	NaN	No	Mid-level	
1	DATA ANALYST	80783.0	103333.0	Yearly	DC Public Library	NaN	No	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	Horizon Blue Cross Blue Shield of New Jersey	NaN	No	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	First American Financial Corporation	CA	Yes	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	Concept Art House	NaN	Yes	Mid-level	

In [34]:

```
# Renames columns to uniform names for
# merging and displays the first 5 rows.
usa_df = usa_df.rename(
    columns = {'Title':'job_title',
               'Remote':'work_setting',
               'State':'state',
               'Salary Period':'salary_period'})

usa_df.head()
```

Out[34]:

	job_title	Min Salary	Max Salary	salary_period	Company Name	state	work_setting	experienc
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	NaN	No	M
1	DATA ANALYST	80783.0	103333.0	Yearly	DC Public Library	NaN	No	M
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	Horizon Blue Cross Blue Shield of New Jersey	NaN	No	M
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	First American Financial Corporation	CA	Yes	M
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	Concept Art House	NaN	Yes	M

In [35]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
usa_df.drop(usa_df.columns[[4]],
            axis = 1, inplace = True)

usa_df.head()
```

Out[35]:

	job_title	Min Salary	Max Salary	salary_period	state	work_setting	experience_level	emp
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	No	Mid-level	
1	DATA ANALYST	80783.0	103333.0	Yearly	NaN	No	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	NaN	No	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	CA	Yes	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	NaN	Yes	Mid-level	

In [36]:

```
# Creates columns and populate with data
# based upon attributes, structure,
# and origin of the dataframe for uniformity,
# as well as displays the first 5 rows.
usa_df['work_setting'] = usa_df[
    'work_setting'].str.replace(
    'Yes', 'Remote').copy()

usa_df['work_setting'] = usa_df[
    'work_setting'].str.replace(
    'No','In-person').copy()

usa_df.head()
```

Out[36]:

	job_title	Min Salary	Max Salary	salary_period	state	work_setting	experience_level	emp
0	CARBON DATA ANALYST	NaN	NaN	NaN	NaN	In-person	Mid-level	
1	DATA ANALYST	80783.0	103333.0	Yearly	NaN	In-person	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	NaN	In-person	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	CA	Remote	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	NaN	Remote	Mid-level	

In [37]:

```
# Renames columns to uniform names for
# merging and displays the first 5 rows.
uk_df = uk_df.rename(
    columns = {'title':'job_title',
               'country':'employee_residence',
               'date_ending':'work_year'})

uk_df.head()
```

Out[37]:

	reference	job_title	date_posted	work_year	advertiser	location	
77	41730933	DATA SCIENTIST / DATA ANALYST	2021-01-12	2021-02-23T23:55:00.0000000	Cameo Consultancy	Oxfordshire	Ban
87	41666705	INSIGHT ANALYST	2021-01-04	2021-02-15T23:55:00.0000000	Harnham	West Yorkshire	Le
106	41863877	PATIENT DATA ANALYST	2021-01-27	2021-03-03T23:55:00.0000000	Hyper Recruitment Solutions Ltd	South East England	Berks
110	41813983	SENIOR INSIGHT ANALYST	2021-01-21	2021-03-04T23:55:00.0000000	Harnham	South East England	Lor
111	41651502	SENIOR INSIGHT ANALYST	2020-12-30	2021-02-10T23:55:00.0000000	Harnham	South East England	Lor

In [38]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
uk_df.drop(uk_df.columns[
    [0, 2, 4, 5, 6, 8, 11, 12, 13]],
    axis = 1, inplace = True)

uk_df.head()
```

Out[38]:

	job_title	work_year	employee_residence	salary_min	salary_max
77	DATA SCIENTIST / DATA ANALYST	2021-02-23T23:55:00.0000000	GB	40000.0	42000.0
87	INSIGHT ANALYST	2021-02-15T23:55:00.0000000	GB	30000.0	45000.0
106	PATIENT DATA ANALYST	2021-03-03T23:55:00.0000000	GB	56000.0	66000.0
110	SENIOR INSIGHT ANALYST	2021-03-04T23:55:00.0000000	GB	55000.0	65000.0
111	SENIOR INSIGHT ANALYST	2021-02-10T23:55:00.0000000	GB	55000.0	65000.0

In [39]:

```
# Creates columns and populate with data
# based upon attributes, structure,
# and origin of the dataframe for uniformity,
# as well as displays the first 5 rows.
uk_df['employee_residence'] = 'United Kingdom.'
uk_df['work_year'] = '2021'
uk_df['employment_type'] = 'Full-time'
uk_df['work_setting'] = 'In-person'

uk_df.head()
```

Out[39]:

	job_title	work_year	employee_residence	salary_min	salary_max	employment_type
77	DATA SCIENTIST / DATA ANALYST	2021	United Kingdom.	40000.0	42000.0	Full-time
87	INSIGHT ANALYST	2021	United Kingdom.	30000.0	45000.0	Full-time
106	PATIENT DATA ANALYST	2021	United Kingdom.	56000.0	66000.0	Full-time
110	SENIOR INSIGHT ANALYST	2021	United Kingdom.	55000.0	65000.0	Full-time
111	SENIOR INSIGHT ANALYST	2021	United Kingdom.	55000.0	65000.0	Full-time

In [40]:

```
# Renames columns to uniform names for
# merging and displays the first 5 rows.
int_df = int_df.rename(
    columns = {'remote_ratio':
               'work_setting'}).copy()

int_df.head()
```

Out[40]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_c
3	3	2020	MI	FT	PRODUCT DATA ANALYST	20000	
5	5	2020	EN	FT	DATA ANALYST	72000	
8	8	2020	MI	FT	BUSINESS DATA ANALYST	135000	
13	13	2020	MI	FT	LEAD DATA ANALYST	87000	
14	14	2020	MI	FT	DATA ANALYST	85000	

In [41]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
int_df.drop(int_df.columns[[0, 5, 6, 10]],
            axis = 1, inplace = True)

int_df.head()
```

Out[41]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_res
3	2020	MI	FT	PRODUCT DATA ANALYST	20000	
5	2020	EN	FT	DATA ANALYST	72000	
8	2020	MI	FT	BUSINESS DATA ANALYST	135000	
13	2020	MI	FT	LEAD DATA ANALYST	87000	
14	2020	MI	FT	DATA ANALYST	85000	

In [42]:

```
# Creates a column and populates with
# data based upon attribute, structure,
# and origin of the dataframe,
# as well as displays the first 5 rows.
int_df['employment_type'] = 'Full-time'

int_df.head()
```

Out[42]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_res
3	2020	MI	Full-time	PRODUCT DATA ANALYST	20000	
5	2020	EN	Full-time	DATA ANALYST	72000	
8	2020	MI	Full-time	BUSINESS DATA ANALYST	135000	
13	2020	MI	Full-time	LEAD DATA ANALYST	87000	
14	2020	MI	Full-time	DATA ANALYST	85000	

In [43]:

```
# Converts data in 'experience_level' column to uniform
# values for merging and displays the first 5 rows.
int_df.loc[int_df['experience_level'] == 'MI',
           'experience_level'] = 'Mid-level'
int_df.loc[int_df['experience_level'] == 'EN',
           'experience_level'] = 'Entry-level'
int_df.loc[int_df['experience_level'] == 'EX',
           'experience_level'] = 'Executive'
int_df.loc[int_df['experience_level'] == 'SE',
           'experience_level'] = 'Senior'

int_df.head()
```

Out[43]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_res
3	2020	Mid-level	Full-time	PRODUCT DATA ANALYST	20000	
5	2020	Entry-level	Full-time	DATA ANALYST	72000	
8	2020	Mid-level	Full-time	BUSINESS DATA ANALYST	135000	
13	2020	Mid-level	Full-time	LEAD DATA ANALYST	87000	
14	2020	Mid-level	Full-time	DATA ANALYST	85000	

In [44]:

```
# Converts all values in the 'employee_residence'
# column to uniform country names
# and displays the first 5 rows.
int_df.loc[int_df['employee_residence'] == 'HN',
           'employee_residence'] = 'Honduras'

int_df.loc[int_df['employee_residence'] == 'US',
           'employee_residence'] = 'United States'

int_df.loc[int_df['employee_residence'] == 'PK',
           'employee_residence'] = 'Pakistan'

int_df.loc[int_df['employee_residence'] == 'IN',
           'employee_residence'] = 'India'

int_df.loc[int_df['employee_residence'] == 'FR',
           'employee_residence'] = 'France'

int_df.loc[int_df['employee_residence'] == 'NG',
           'employee_residence'] = 'Nigeria'

int_df.loc[int_df['employee_residence'] == 'BG',
           'employee_residence'] = 'Bulgaria'

int_df.loc[int_df['employee_residence'] == 'GR',
           'employee_residence'] = 'Greece'

int_df.loc[int_df['employee_residence'] == 'HU',
           'employee_residence'] = 'Hungary'

int_df.loc[int_df['employee_residence'] == 'GB',
           'employee_residence'] = 'United Kingdom'

int_df.loc[int_df['employee_residence'] == 'ES',
           'employee_residence'] = 'Spain'

int_df.loc[int_df['employee_residence'] == 'KE',
           'employee_residence'] = 'Kenya'

int_df.loc[int_df['employee_residence'] == 'CA',
           'employee_residence'] = 'Canada'

int_df.loc[int_df['employee_residence'] == 'DE',
           'employee_residence'] = 'Germany'

int_df.loc[int_df['employee_residence'] == 'LU',
           'employee_residence'] = 'Luxembourg'

int_df.head()
```

Out[44]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_res
3	2020	Mid-level	Full-time	PRODUCT DATA ANALYST	20000	Ho
5	2020	Entry-level	Full-time	DATA ANALYST	72000	United
8	2020	Mid-level	Full-time	BUSINESS DATA ANALYST	135000	United
13	2020	Mid-level	Full-time	LEAD DATA ANALYST	87000	United
14	2020	Mid-level	Full-time	DATA ANALYST	85000	United

In [45]:

```
# Converts the 'work_setting' column datatype
# to string, converts all values in the column
# to uniform work setting values for merging
# dataframes, and displays the first 5 rows.
int_df['work_setting'] = int_df[
    'work_setting'].astype(str).copy()

int_df.loc[int_df['work_setting'] == '0',
           'work_setting'] = 'In-person'

int_df.loc[int_df['work_setting'] == '50',
           'work_setting'] = 'Hybrid'

int_df.loc[int_df['work_setting'] == '100',
           'work_setting'] = 'Remote'

int_df.head()
```

Out[45]:

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_res
3	2020	Mid-level	Full-time	PRODUCT DATA ANALYST	20000	Ho
5	2020	Entry-level	Full-time	DATA ANALYST	72000	United
8	2020	Mid-level	Full-time	BUSINESS DATA ANALYST	135000	United
13	2020	Mid-level	Full-time	LEAD DATA ANALYST	87000	United
14	2020	Mid-level	Full-time	DATA ANALYST	85000	United

In [46]:

```
# Renames columns to uniform names for
# merging and displays the first 5 rows.
can_df = can_df.rename(
    columns = {'JobTitle':'job_title',
               'Location':'employee_residence',
               'PostDate':'work_year',
               'Salary':'salary_in_usd'})

can_df.head()
```

Out[46]:

	job_title	Company	employee_residence	salary_in_usd	work_year	Su
99	SENIOR STATISTICAL/ACTUARIAL ANALYST	Aviva	Montréal, QC	NaN	17 days ago	p in
139	SENIOR STATISTICAL/ACTUARIAL ANALYST	Aviva	Montréal, QC	NaN	17 days ago	p in
168	COMPENSATION ANALYST	AbCellera Biologics	Vancouver, BC	NaN	12 days ago	C ca is dri
194	COMPENSATION ANALYST	AbCellera	Vancouver, BC	NaN	11 days ago	C ca is dri
214	CORPORATE DEVELOPMENT ANALYST	AbCellera	Vancouver, BC	NaN	11 days ago	Pro dev kn

In [47]:
```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
can_df.drop(can_df.columns[[1, 5, 6]],
            axis = 1, inplace = True)

can_df.head()
```

Out[47]:

	job_title	employee_residence	salary_in_usd	work_year
99	SENIOR STATISTICAL/ACTUARIAL ANALYST	Montréal, QC	NaN	17 days ago
139	SENIOR STATISTICAL/ACTUARIAL ANALYST	Montréal, QC	NaN	17 days ago
168	COMPENSATION ANALYST	Vancouver, BC	NaN	12 days ago
194	COMPENSATION ANALYST	Vancouver, BC	NaN	11 days ago
214	CORPORATE DEVELOPMENT ANALYST	Vancouver, BC	NaN	11 days ago

In [48]:
```
# Creates columns and populates with data
# based upon attributes, structure, and
# origin of the dataframe for uniformity,
# as well as displays the first 5 rows.
can_df['employee_residence'] = 'Canada'
can_df['work_year'] = '2020'
can_df['work_setting'] = 'In-person'
can_df['employment_type'] = 'Full-time'

can_df.head()
```

Out[48]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting
99	SENIOR STATISTICAL/ACTUARIAL ANALYST	Canada	NaN	2020	In-person
139	SENIOR STATISTICAL/ACTUARIAL ANALYST	Canada	NaN	2020	In-person
168	COMPENSATION ANALYST	Canada	NaN	2020	In-person
194	COMPENSATION ANALYST	Canada	NaN	2020	In-person
214	CORPORATE DEVELOPMENT ANALYST	Canada	NaN	2020	In-person

Drop Null Values

In [49]:

```
# Removes all null values from specified
# columns and displays the first 5 rows.
usa_df.dropna(subset = ['Min Salary'],
              inplace = True)

usa_df.dropna(subset = ['Max Salary'],
              inplace = True)

usa_df.head()
```

Out[49]:

	job_title	Min Salary	Max Salary	salary_period	state	work_setting	experience_level	emp
1	DATA ANALYST	80783.0	103333.0	Yearly	NaN	In-person	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	NaN	In-person	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	CA	Remote	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	NaN	Remote	Mid-level	
5	DATA ANALYST I	62000.0	62000.0	Yearly	NaN	In-person	Mid-level	

In [50]:
```
# Removes all null values from specified
# columns and displays the first 5 rows.
can_df.dropna(subset = ['salary_in_usd'],
              inplace = True)

can_df.dropna(subset = ['work_year'],
              inplace = True)

can_df.head()
```

Out[50]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employr
238	BUSINESS ANALYST	Canada	45, 000–60,000 a year	2020	In-person	
250	BUSINESS ANALYST	Canada	45, 000–60,000 a year	2020	In-person	
574	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	41.03–47.58 an hour	2020	In-person	
898	BILINGUAL BUSINESS ANALYST	Canada	45, 000–50,000 a year	2020	In-person	
1132	DATA ANALYST (CONTRACT)	Canada	4, 000–4,500 a month	2020	In-person	

Annual Salary Calculations and Conversions to U.S. Dollar

In [51]:

```
# Calculates the approximation of annual salaries for
# various work types (full-time, part-time, contractor)
# as well as salary periods (monthly, hourly, daily) for
# merging dataframes and displays the first 5 rows.
usa_df.loc[usa_df['salary_period'] == 'Monthly',
           'Min Salary'] = usa_df['Min Salary'] * 12

usa_df.loc[usa_df['salary_period'] == 'Monthly',
           'Max Salary'] = usa_df['Max Salary'] * 12

usa_df.loc[usa_df['salary_period'] == 'Hourly',
           'Min Salary'] = (
            usa_df['Min Salary'] * (40 * 52))

usa_df.loc[usa_df['salary_period'] == 'Hourly',
           'Max Salary'] = (
            usa_df['Max Salary'] * (40 * 52))

usa_df.loc[usa_df['salary_period'] == 'Daily',
           'Min Salary'] = (
            usa_df['Min Salary'] * (5 * 52))

usa_df.loc[usa_df['salary_period'] == 'Daily',
           'Max Salary'] = (
            usa_df['Max Salary'] * (5 * 52))

usa_df.head()
```

Out[51]:

	job_title	Min Salary	Max Salary	salary_period	state	work_setting	experience_level	emp
1	DATA ANALYST	80783.0	103333.0	Yearly	NaN	In-person	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	NaN	In-person	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	CA	Remote	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	NaN	Remote	Mid-level	
5	DATA ANALYST I	62000.0	62000.0	Yearly	NaN	In-person	Mid-level	

In [52]:

```
# Calculates an average of salaries for comparative
# analysis and displays the first 5 rows.
usa_df['salary_in_usd'] = ((usa_df['Min Salary'] +
                            usa_df['Max Salary']) /
                           2).astype('int').copy()

usa_df.head()
```

Out[52]:

	job_title	Min Salary	Max Salary	salary_period	state	work_setting	experience_level	emp
1	DATA ANALYST	80783.0	103333.0	Yearly	NaN	In-person	Mid-level	
2	DATA REPORTING ANALYST III	75300.0	100800.0	Yearly	NaN	In-person	Mid-level	
3	SENIOR DATA ANALYST	78700.0	163400.0	Yearly	CA	Remote	Mid-level	
4	PRODUCT DATA ANALYST	95200.0	121000.0	Yearly	NaN	Remote	Mid-level	
5	DATA ANALYST I	62000.0	62000.0	Yearly	NaN	In-person	Mid-level	

In [53]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
usa_df.drop(usa_df.columns[[1, 2, 3]],
            axis = 1, inplace = True)

usa_df.head()
```

Out[53]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
1	DATA ANALYST	NaN	In-person	Mid-level	United States	Full-time
2	DATA REPORTING ANALYST III	NaN	In-person	Mid-level	United States	Full-time
3	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
4	PRODUCT DATA ANALYST	NaN	Remote	Mid-level	United States	Full-time
5	DATA ANALYST I	NaN	In-person	Mid-level	United States	Full-time

In [54]:

```
# Calculates an average of salaries for comparative
# analysis and displays the first 5 rows.
uk_df['salary_in_usd'] = ((uk_df['salary_min'] +
                           uk_df['salary_max']) / 2
                           ).astype('int').copy()

uk_df.head()
```

Out[54]:

	job_title	work_year	employee_residence	salary_min	salary_max	employment_type
77	DATA SCIENTIST / DATA ANALYST	2021	United Kingdom.	40000.0	42000.0	Full-time
87	INSIGHT ANALYST	2021	United Kingdom.	30000.0	45000.0	Full-time
106	PATIENT DATA ANALYST	2021	United Kingdom.	56000.0	66000.0	Full-time
110	SENIOR INSIGHT ANALYST	2021	United Kingdom.	55000.0	65000.0	Full-time
111	SENIOR INSIGHT ANALYST	2021	United Kingdom.	55000.0	65000.0	Full-time

In [55]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
uk_df.drop(uk_df.columns[[3, 4]],
           axis = 1, inplace = True)

uk_df.head()
```

Out[55]:

	job_title	work_year	employee_residence	employment_type	work_setting	salary_in_
77	DATA SCIENTIST / DATA ANALYST	2021	United Kingdom.	Full-time	In-person	4
87	INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	3
106	PATIENT DATA ANALYST	2021	United Kingdom.	Full-time	In-person	6
110	SENIOR INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	6
111	SENIOR INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	6

In [56]:

```
# Converts the currency of the British
# Sterling Pound to the United States
# Dollar as datatype integer for comparative
# analysis and displays the first 5 rows.
uk_df['salary_in_usd'] = (uk_df['salary_in_usd'] *
                          1.27).astype(int).copy()

uk_df.head()
```

Out[56]:

	job_title	work_year	employee_residence	employment_type	work_setting	salary_in_
77	DATA SCIENTIST / DATA ANALYST	2021	United Kingdom.	Full-time	In-person	52
87	INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	47
106	PATIENT DATA ANALYST	2021	United Kingdom.	Full-time	In-person	77
110	SENIOR INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	76
111	SENIOR INSIGHT ANALYST	2021	United Kingdom.	Full-time	In-person	76

In [57]:

```
# Converts all the letters in the 'salary_in_usd'
# column to categorize for uniformity
# and displays the first 5 rows.
can_df['salary_in_usd'] = can_df[
    'salary_in_usd'].apply(
    lambda x: x.upper()).copy()

can_df.head()
```

Out[57]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employr
238	BUSINESS ANALYST	Canada	45, 000– 60,000 A YEAR	2020	In-person	
250	BUSINESS ANALYST	Canada	45, 000– 60,000 A YEAR	2020	In-person	
574	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	41.03–47.58 AN HOUR	2020	In-person	
898	BILINGUAL BUSINESS ANALYST	Canada	45, 000– 50,000 A YEAR	2020	In-person	
1132	DATA ANALYST (CONTRACT)	Canada	4, 000–4,500 A MONTH	2020	In-person	

In [58]:

```
# Creates a 'min_salary' value from
# a slice of the first 8 characters.
can_df['min_salary'] = can_df.salary_in_usd.str[:8].copy()

# Removes all unnecessary characters after
# specified delimiters for 'min_salary'.
can_df['min_salary'] = can_df[
    'min_salary'].str.replace('$', '').copy()

can_df['min_salary'] = can_df[
    'min_salary'].str.replace('-', '').copy()

can_df['min_salary'] = can_df[
    'min_salary'].str.replace(',', '').copy()

# Creates a 'max_salary' value from a slice of characters 9 - 18.
can_df['max_salary'] = can_df.salary_in_usd.str[9:18].copy()

# Removes all unnecessary characters after
# specified delimiters for 'max_salary'.
can_df['max_salary'] = can_df[
    'max_salary'].str.replace('$', '').copy()

can_df['max_salary'] = can_df[
    'max_salary'].str.replace(',', '').copy()

can_df.head()
```

Out[58]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employr
238	BUSINESS ANALYST	Canada	45, 000–60,000 A YEAR	2020	In-person	
250	BUSINESS ANALYST	Canada	45, 000–60,000 A YEAR	2020	In-person	
574	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	41.03–47.58 AN HOUR	2020	In-person	
898	BILINGUAL BUSINESS ANALYST	Canada	45, 000–50,000 A YEAR	2020	In-person	
1132	DATA ANALYST (CONTRACT)	Canada	4, 000–4,500 A MONTH	2020	In-person	

In [59]:

```
# Creates dataframes with the condition that specified salary
# period words are contained in the 'salary_in_usd' column.
yearly = can_df[
    can_df['salary_in_usd'].str.contains(
    'YEAR')].copy()

hourly = can_df[
    can_df['salary_in_usd'].str.contains(
        'HOUR')].copy()

monthly = can_df[
    can_df['salary_in_usd'].str.contains(
        'MONTH')].copy()
```

In [60]:

```
# Displays the first 5 rows of the yearly dataframe.
yearly.head()
```

Out[60]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employm
238	BUSINESS ANALYST	Canada	45, 000– 60,000 A YEAR	2020	In-person	
250	BUSINESS ANALYST	Canada	45, 000– 60,000 A YEAR	2020	In-person	
898	BILINGUAL BUSINESS ANALYST	Canada	45, 000– 50,000 A YEAR	2020	In-person	
1169	ANALYSTE DE DONNÉES / ANALYSTE FONCTIONNEL BI	Canada	75, 000– 110,000 A YEAR	2020	In-person	
1448	DATA ANALYST	Canada	50, 000– 90,000 A YEAR	2020	In-person	

In [61]:
```
# Displays the first 5 rows
# of the hourly dataframe.
hourly.head()
```

Out[61]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employr
574	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	41.03–47.58 AN HOUR	2020	In-person	
1316	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	41.03–47.58 AN HOUR	2020	In-person	
1706	ANALYSTE DE DONNÉES - DATA ANALYST	Canada	43–52 AN HOUR	2020	In-person	
1708	DATABASE ANALYST	Canada	30–35 AN HOUR	2020	In-person	
1713	ANALYSTE DE DONNÉES - DATA ANALYST	Canada	43–52 AN HOUR	2020	In-person	

In [62]:
```
# Displays the first 5 rows
# of the monthly dataframe.
monthly.head()
```

Out[62]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employme
1132	DATA ANALYST (CONTRACT)	Canada	4,000–4,500 A MONTH	2020	In-person	F
1711	DATA ANALYST	Canada	5,362–7,724 A MONTH	2020	In-person	F
1730	DATA ANALYST (CONTRACT)	Canada	4,000–4,500 A MONTH	2020	In-person	F
1741	TECHNICAL BUSINESS ANALYST II	Canada	6,779–8,026 A MONTH	2020	In-person	F
1748	TECHNICAL BUSINESS ANALYST II	Canada	6,779–8,026 A MONTH	2020	In-person	F

In [63]:

```
# Removes all unnecessary characters
# after specified delimiters for 'min_salary'.
hourly['min_salary'] = hourly[
    'min_salary'].str.split('.').str[0].copy()

hourly['min_salary'] = hourly[
    'min_salary'].str.split('-').str[0].copy()

hourly['min_salary'] = hourly[
    'min_salary'].str.split(' ').str[0].copy()

# Removes all unnecessary characters
# after specified delimiters for 'mAX_salary'.
hourly['max_salary'] = hourly[
    'max_salary'].str.split('.').str[0].copy()

hourly['max_salary'] = hourly[
    'max_salary'].str.split('-').str[0].copy()

hourly['max_salary'] = hourly[
    'max_salary'].str.split(' ').str[0].copy()

hourly['max_salary'] = hourly[
    'salary_in_usd'].str.split('-').str[-1].copy()

hourly['max_salary'] = hourly[
    'max_salary'].str.replace('$', '').copy()

hourly['max_salary'] = hourly[
    'max_salary'].str.rsplit('.').str[0].copy()

# Creates the 'max_salary' values to be
# a slice from the first 3 characters.
hourly['max_salary'] = hourly.max_salary.str[:3].copy()

# Converts the datatype for both columns to integer.
hourly['min_salary'] = hourly[
    'min_salary'].astype(int).copy()

hourly['max_salary'] = hourly[
    'max_salary'].astype(int).copy()

# Calculates the approximation of an
# annual salary for comparitive analysis.
hourly['min_salary'] = (hourly[
    'min_salary'] * (40 * 52)).copy()

hourly['max_salary'] = (hourly[
    'max_salary'] * (40 * 52)).copy()

# Displays the last 5 rows.
hourly.tail()
```

Out[63]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employm
2792	SENIOR BUSINESS SYSTEMS ANALYST (CLOUD CENTRE ...	Canada	70−75 AN HOUR	2020	In-person	
2804	SYSTEMS BUSINESS ANALYST	Canada	47.50−55.00 AN HOUR	2020	In-person	
2813	BUSINESS ANALYST - FUND ACCOUNTING (INVESTONE)	Canada	$70 AN HOUR	2020	In-person	
2819	SYSTEMS BUSINESS ANALYST	Canada	47.50−55.00 AN HOUR	2020	In-person	
2821	BUSINESS ANALYST - FUND ACCOUNTING (INVESTONE)	Canada	$70 AN HOUR	2020	In-person	

In [64]:

```
# Calculates the average salary and replaces
# the values in the 'salary_in_usd' column.
hourly['salary_in_usd'] = (
    (hourly['min_salary'] +
     hourly['max_salary']) / 2).copy()

hourly.head()
```

Out[64]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employr
574	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	91520.0	2020	In-person	
1316	SENIOR ANALYST, DATA VISUALIZATION AND ANALYTICS	Canada	91520.0	2020	In-person	
1706	ANALYSTE DE DONNÉES - DATA ANALYST	Canada	98800.0	2020	In-person	
1708	DATABASE ANALYST	Canada	67600.0	2020	In-person	
1713	ANALYSTE DE DONNÉES - DATA ANALYST	Canada	98800.0	2020	In-person	

In [65]:

```
# Removes all unecessary remaining characters
# after the specified space delimiter.
monthly['max_salary'] = monthly[
    'max_salary'].str.split(' ').str[0].copy()

# Converts the datatypes of both columns to integer.
monthly['min_salary'] = monthly[
    'min_salary'].astype(int).copy()

monthly['max_salary'] = monthly[
    'max_salary'].astype(int).copy()

# Calculates the approximation of an
# annual salary for comparitive analysis.
monthly['min_salary'] = (monthly[
    'min_salary'] * 12).copy()
monthly['max_salary'] = (monthly[
    'max_salary'] * 12).copy()

# Displays the first 5 rows.
monthly.head()
```

Out[65]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employme
1132	DATA ANALYST (CONTRACT)	Canada	4,000–4,500 A MONTH	2020	In-person	F
1711	DATA ANALYST	Canada	5,362–7,724 A MONTH	2020	In-person	F
1730	DATA ANALYST (CONTRACT)	Canada	4,000–4,500 A MONTH	2020	In-person	F
1741	TECHNICAL BUSINESS ANALYST II	Canada	6,779–8,026 A MONTH	2020	In-person	F
1748	TECHNICAL BUSINESS ANALYST II	Canada	6,779–8,026 A MONTH	2020	In-person	F

In [66]:

```
# Calculates the average salary and replaces
# the values in the 'salary_in_usd' column.
monthly['salary_in_usd'] = (
    (monthly['min_salary'] +
    monthly['max_salary']) / 2).copy()

# Displays the first 5 rows.
monthly.head()
```

Out[66]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employme
1132	DATA ANALYST (CONTRACT)	Canada	51000.0	2020	In-person	F
1711	DATA ANALYST	Canada	78516.0	2020	In-person	F
1730	DATA ANALYST (CONTRACT)	Canada	51000.0	2020	In-person	F
1741	TECHNICAL BUSINESS ANALYST II	Canada	88830.0	2020	In-person	F
1748	TECHNICAL BUSINESS ANALYST II	Canada	88830.0	2020	In-person	F

In [67]:

```
# Converts the datatype of the 'salary_in_usd'
# column for both dataframes to integer.
monthly['salary_in_usd'] = monthly[
    'salary_in_usd'].astype(int).copy()

hourly['salary_in_usd'] = hourly[
    'salary_in_usd'].astype(int).copy()
```

In [68]:

```
# Creates a dataframe from the merge
# of monthly and hourly dataframes
# and displays the first 5 rows.
temp_df = pd.merge(monthly, hourly,
                   how = 'outer',
                   on = ['job_title',
                         'employee_residence',
                         'employment_type',
                         'work_setting',
                         'salary_in_usd',
                         'work_year']).copy()

temp_df.head()
```

Out[68]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_t
0	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
1	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
2	DATA ANALYST	Canada	78516	2020	In-person	Full-
3	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-
4	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-

In [69]:

```
# Creates a dataframe with the
# condition 'YEAR' is contained
# within the 'max_salary' column
# and displays the first 5 rows.
yearly2 = yearly[yearly[
    'max_salary'].str.contains('YEAR')].copy()

yearly2.head()
```

Out[69]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employmen
1494	DATA ANALYST	Canada	$85,000 A YEAR	2020	In-person	Fu
1701	DATA ANALYST	Canada	$70,000 A YEAR	2020	In-person	Fu
1707	RESEARCH ANALYST, HEALTH POLICY	Canada	$80,000 A YEAR	2020	In-person	Fu
1725	DATA ANALYST	Canada	$73,068 A YEAR	2020	In-person	Fu
1726	DATA ANALYST	Canada	$45,000 A YEAR	2020	In-person	Fu

In [70]:

```
# Creates a dataframe with the
# condition 'A YEAR' is contained
# within the 'max_salary' column
# and displays the first 5 rows.
yearly3 = yearly[yearly[
    'max_salary'].str.contains('A YEAR')].copy()

yearly3.head()
```

Out[70]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_
2808	DAX SENIOR BUSINESS ANALYST	Canada	$100,000 A YEAR	2020	In-person	Full
2822	DAX SENIOR BUSINESS ANALYST	Canada	$100,000 A YEAR	2020	In-person	Full

In [71]:

```
# Replaces the values in 'salary_in_usd'
# with the previously cleaned value in
# 'min_salary' as datatype integer.
yearly2['salary_in_usd'] = yearly2[
    'min_salary'].astype(int).copy()

# Removes all unecessary columns.
yearly2.drop(yearly2.columns[[6,7]],
             axis = 1, inplace = True)

# Displays the first 5 rows.
yearly2.head()
```

Out[71]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employmen
1494	DATA ANALYST	Canada	85000	2020	In-person	Fu
1701	DATA ANALYST	Canada	70000	2020	In-person	Fu
1707	RESEARCH ANALYST, HEALTH POLICY	Canada	80000	2020	In-person	Fu
1725	DATA ANALYST	Canada	73068	2020	In-person	Fu
1726	DATA ANALYST	Canada	45000	2020	In-person	Fu

In [72]:

```
# Replaces the values in 'salary_in_usd'
# with the previously cleaned value in
# 'min_salary' as datatype integer.
yearly3['salary_in_usd'] = yearly3[
    'min_salary'].astype(int).copy()

# # Removes all unecessary columns.
yearly3.drop(yearly3.columns[[6, 7]],
             axis = 1, inplace = True)

# Displays the first 5 rows.
yearly3.head()
```

Out[72]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_
2808	DAX SENIOR BUSINESS ANALYST	Canada	100000	2020	In-person	Full
2822	DAX SENIOR BUSINESS ANALYST	Canada	100000	2020	In-person	Full

In [73]:

```
# Creates a dataframe from the merge of yearly2 and
# yearly3 dataframes and displays the first 5 rows.
yearly4 = pd.merge(
    yearly2, yearly3,
           how = 'outer',
           on = ['job_title',
                 'work_setting',
                 'employment_type',
                 'employee_residence',
                 'salary_in_usd','work_year']).copy()

yearly4.head()
```

Out[73]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_type
0	DATA ANALYST	Canada	85000	2020	In-person	Full-time
1	DATA ANALYST	Canada	70000	2020	In-person	Full-time
2	DATA ANALYST	Canada	70000	2020	In-person	Full-time
3	DATA ANALYST	Canada	70000	2020	In-person	Full-time
4	DATA ANALYST	Canada	70000	2020	In-person	Full-time

In [74]:

```
# Creates a dataframe from the merge
# of temp_df and yearly4 dataframe
# and displays the first 5 rows.
temp_df2 = pd.merge(
    temp_df, yearly4,
            how = 'outer',
            on = ['job_title',
                  'employee_residence',
                  'salary_in_usd',
                  'work_year',
                  'work_setting',
                  'employment_type']).copy()

temp_df2.head()
```

Out[74]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_t
0	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
1	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
2	DATA ANALYST	Canada	78516	2020	In-person	Full-
3	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-
4	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-

In [75]:

```
# Removes all unecessary columns
# and displays the first 5 rows.
temp_df2.drop(
    temp_df2.columns[[6, 7, 8, 9]],
    axis = 1, inplace = True)

temp_df2.head()
```

Out[75]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_t
0	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
1	DATA ANALYST (CONTRACT)	Canada	51000	2020	In-person	Full-
2	DATA ANALYST	Canada	78516	2020	In-person	Full-
3	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-
4	TECHNICAL BUSINESS ANALYST II	Canada	88830	2020	In-person	Full-

In [76]:

```
# Converts the Canadian Dollar to the
# United States Dollar as datatype
# integer for comparative analysis
# and displays the first 5 rows.
temp_df2['salary_in_usd'] = (
    temp_df2['salary_in_usd']
    * .74).astype(int).copy()

temp_df2.head()
```

Out[76]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_t
0	DATA ANALYST (CONTRACT)	Canada	37740	2020	In-person	Full-
1	DATA ANALYST (CONTRACT)	Canada	37740	2020	In-person	Full-
2	DATA ANALYST	Canada	58101	2020	In-person	Full-
3	TECHNICAL BUSINESS ANALYST II	Canada	65734	2020	In-person	Full-
4	TECHNICAL BUSINESS ANALYST II	Canada	65734	2020	In-person	Full-

Clean and Merge All of the dataframes

In [77]:

```
# Creates a cleaned copy of the dataframe
# and displays the first 5 rows.
can_df = temp_df2.copy()

can_df.head()
```

Out[77]:

	job_title	employee_residence	salary_in_usd	work_year	work_setting	employment_t
0	DATA ANALYST (CONTRACT)	Canada	37740	2020	In-person	Full-
1	DATA ANALYST (CONTRACT)	Canada	37740	2020	In-person	Full-
2	DATA ANALYST	Canada	58101	2020	In-person	Full-
3	TECHNICAL BUSINESS ANALYST II	Canada	65734	2020	In-person	Full-
4	TECHNICAL BUSINESS ANALYST II	Canada	65734	2020	In-person	Full-

In [78]:

```
# Replaces all NaN values with
# a 0 (zero) in both dataframes
# and displays the first 5 rows.
usa_df = usa_df.fillna(0).copy()

world_df = world_df.fillna(0).copy()

usa_df.head()
```

Out[78]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
1	DATA ANALYST	0	In-person	Mid-level	United States	Full-time
2	DATA REPORTING ANALYST III	0	In-person	Mid-level	United States	Full-time
3	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
4	PRODUCT DATA ANALYST	0	Remote	Mid-level	United States	Full-time
5	DATA ANALYST I	0	In-person	Mid-level	United States	Full-time

In [79]:

```
# Converts the 'work_year' attribute
# datatype to an integer.
usa_df['work_year'] = usa_df[
    'work_year'].astype(int).copy()

# Creates a dataframe from the merge
# of usa_df and world_df dataframes
# and displays the first 5 rows.
all_df = pd.merge(
    usa_df, world_df,
          how = 'outer',
          on = ['job_title',
                'work_setting',
                'salary_in_usd',
                'work_year',
                'employment_type',
                'experience_level',
                'employee_residence']).copy()

all_df.head()
```

Out[79]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
0	DATA ANALYST	0	In-person	Mid-level	United States	Full-time
1	DATA REPORTING ANALYST III	0	In-person	Mid-level	United States	Full-time
2	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
3	PRODUCT DATA ANALYST	0	Remote	Mid-level	United States	Full-time
4	DATA ANALYST I	0	In-person	Mid-level	United States	Full-time

In [80]:

```
# Converts the 'work_year' attribute datatype
# to an integer for both dataframes for merging.
all_df['work_year'] = all_df[
    'work_year'].astype(int).copy()

uk_df['work_year'] = uk_df[
    'work_year'].astype(int).copy()

# Creates a dataframe from the merge
# of all_df and uk_df dataframes
# and displays the first 5 rows.
all_df1 = pd.merge(
    all_df, uk_df,
           how = 'outer',
           on = ['job_title',
                 'work_setting',
                 'work_year',
                 'salary_in_usd',
                 'employment_type',
                 'employee_residence']).copy()

all_df1.head()
```

Out[80]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
0	DATA ANALYST	0	In-person	Mid-level	United States	Full-time
1	DATA REPORTING ANALYST III	0	In-person	Mid-level	United States	Full-time
2	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
3	PRODUCT DATA ANALYST	0	Remote	Mid-level	United States	Full-time
4	DATA ANALYST I	0	In-person	Mid-level	United States	Full-time

In [81]:

```
# Converts the 'work_year' attribute
# datatype to an integer.
int_df['work_year'] = int_df[
    'work_year'].astype(int).copy()

# Creates a dataframe from the merge
# of all_df1 and int_df dataframes
# and displays the first 5 rows.
all_df2 = pd.merge(
    all_df1, int_df,
           how = 'outer',
           on = ['job_title',
                 'work_setting',
                 'work_year',
                 'experience_level',
                 'salary_in_usd',
                 'employment_type',
                 'employee_residence',
                 'company_size']).copy()

all_df2.head()
```

Out[81]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
0	DATA ANALYST	0	In-person	Mid-level	United States	Full-time
1	DATA REPORTING ANALYST III	0	In-person	Mid-level	United States	Full-time
2	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
3	PRODUCT DATA ANALYST	0	Remote	Mid-level	United States	Full-time
4	DATA ANALYST I	0	In-person	Mid-level	United States	Full-time

In [82]:

```
# Replaces all NaN values with a 0 (zero)
# and displays the first 5 rows.
all_df2 = all_df2.replace([np.nan], 0).copy()

all_df2.head()
```

Out[82]:

	job_title	state	work_setting	experience_level	employee_residence	employment_type
0	DATA ANALYST	0	In-person	Mid-level	United States	Full-time
1	DATA REPORTING ANALYST III	0	In-person	Mid-level	United States	Full-time
2	SENIOR DATA ANALYST	CA	Remote	Mid-level	United States	Full-time
3	PRODUCT DATA ANALYST	0	Remote	Mid-level	United States	Full-time
4	DATA ANALYST I	0	In-person	Mid-level	United States	Full-time

In [83]:

```
# Changes the values in the entire 'job_title'
# column to Data Analyst.
all_df2['job_title'] = 'Data Analyst'

# Removes all remaining unwanted
# characters after cleaning.
all_df2['employee_residence'] = all_df2[
    'employee_residence'].str.split(
    '.').str[0].copy()

# Removes unecessary columns and
# displays the first 5 rows.
all_df2.drop(all_df2.columns[[5]],
             axis = 1, inplace = True)

all_df2.head()
```

Out[83]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

Calculate Z-Scores

In [84]:

```
# Creates a copy of the dataframe
# and displays the first 5 rows.
final_df = all_df2.copy()

final_df.tail()
```

Out[84]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary
2477	Data Analyst	0	Remote	Mid-level	Greece	2022	
2478	Data Analyst	0	Remote	Mid-level	India	2022	
2479	Data Analyst	0	Hybrid	Mid-level	Canada	2022	
2480	Data Analyst	0	In-person	Mid-level	United Kingdom	2022	
2481	Data Analyst	0	In-person	Mid-level	United Kingdom	2022	

In [85]:

```
# Creates a copy of the dataframe
# and displays the first 5 rows.
final_df20 = final_df.copy()

final_df20.head()
```

Out[85]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [86]:

```
# Creates a new dataframe to calculate the job
# count using the groupby function for employee
# residence and displays the first 5 rows
final_df21 = pd.concat([final_df20.groupby(
    'employee_residence').count()]).reset_index()

final_df21.head()
```

Out[86]:

	employee_residence	job_title	state	work_setting	experience_level	work_year	salary_in
0	Argentina	3	3	3	3	3	
1	Armenia	1	1	1	1	1	
2	Australia	5	5	5	5	5	
3	Brazil	2	2	2	2	2	
4	Bulgaria	1	1	1	1	1	

In [87]:

```
# Removes 6 unnecessary columns
# and displays the first 5 rows.
final_df21.drop(
    final_df21.columns[[2, 3, 4, 5, 6, 7]],
    axis = 1, inplace = True)

final_df21.head()
```

Out[87]:

	employee_residence	job_title
0	Argentina	3
1	Armenia	1
2	Australia	5
3	Brazil	2
4	Bulgaria	1

In [88]:

```
# Creates a dataframe from merging
# and displays the first 5 rows.
final_df22 = final_df20.merge(
    final_df21,
    how = 'left',
    on = 'employee_residence' ).copy()

final_df22.head()
```

Out[88]:

	job_title_x	state	work_setting	experience_level	employee_residence	work_year	salary_
0	Data Analyst	0	In-person	Mid-level	United States	2022	
1	Data Analyst	0	In-person	Mid-level	United States	2022	
2	Data Analyst	CA	Remote	Mid-level	United States	2022	
3	Data Analyst	0	Remote	Mid-level	United States	2022	
4	Data Analyst	0	In-person	Mid-level	United States	2022	

In [89]:

```
# Rename 2 columns and displays the first 5 rows.
final_df22 = final_df22.rename(
    columns = {'job_title_x':'job_title',
               'job_title_y':'job_count'}).copy()

final_df22.head()
```

Out[89]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [90]:

```
# Creates a column with the z-score calculation
# values and displays the first 5 rows.
final_df22['country_job_z_score'] = stats.zscore(
    final_df22.job_count).copy()

final_df22.tail()
```

Out[90]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary
2477	Data Analyst	0	Remote	Mid-level	Greece	2022	
2478	Data Analyst	0	Remote	Mid-level	India	2022	
2479	Data Analyst	0	Hybrid	Mid-level	Canada	2022	
2480	Data Analyst	0	In-person	Mid-level	United Kingdom	2022	
2481	Data Analyst	0	In-person	Mid-level	United Kingdom	2022	

In [91]:

```
# Creates a new dataframe to calculate the job
# count using the groupby function for State
# and displays the first 5 rows
final_df23 = pd.concat(
    [final_df22.groupby('state').count()]).reset_index()

# Removes 8 unnecessary columns.
final_df23.drop(
    final_df23.columns[[2, 3, 4, 5, 6, 7, 8, 9]],
    axis = 1, inplace = True)

# Renames 2 columns and displays the first 5 rows.
final_df23.rename(columns = {'job_title':'job_count'},
                  inplace = True)

final_df23.head()
```

Out[91]:

	state	job_count
0	0	1991
1	AL	3
2	AR	4
3	AZ	10
4	CA	71

In [92]:

```
# Creates a dataframe from valid
# States using the .loc method.
final_df23 = final_df23.loc[
    final_df23['state'] != 0].copy()

# Calculates the state job z-score
# and displays the first 5 rows.
final_df23['state_job_z_score'] = stats.zscore(
    final_df23.job_count).copy()

final_df23.head()
```

Out[92]:

	state	job_count	state_job_z_score
1	AL	3	-0.560592
2	AR	4	-0.485313
3	AZ	10	-0.033636
4	CA	71	4.558416
5	CO	9	-0.108915

In [93]:

```
# Creates a datframe form merging
# and displays the first 5 rows.
final_df25 = final_df22.merge(
    final_df23,
    how = 'left',
    on = 'state').copy()

final_df25.head()
```

Out[93]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [94]:

```
# Replaces all NaN values with a 0 (zero).
final_df25 = final_df25.replace(
    [np.nan], 0).copy()

# Renames the column to country_job_count.
final_df25['country_job_count'] = final_df25[
    'job_count_x'].copy()

# Renames the column to state_job_count
# and converts to datatype integer.
final_df25['state_job_count'] = final_df25[
    'job_count_y'].astype(int).copy()

# Removes 2 unnecessary columns.
final_df25.drop(final_df25.columns[[8, 10]],
                axis = 1, inplace = True)

final_df25.head()
```

Out[94]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [95]:

```
# Makes a copy of the dataframe and
# displays the first 5 rows.
final_df26 = final_df25.copy()

final_df26.head()
```

Out[95]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [96]:

```
# Creates a new dataframe to
# calculate the z-scores for
# salaries using the groupby
# function for State and
# displays the first 5 rows
final_df27 = pd.concat(
    [final_df26.groupby(
    ['state',
    'work_setting']).sum()]).reset_index()

final_df27.head()
```

Out[96]:

	state	work_setting	work_year	salary_in_usd	country_job_z_score	state_job_z_score	c
0	0	Hybrid	68744	2012427	-60.394928	0.000000	
1	0	In-person	2234970	114525460	-153.132209	0.000000	
2	0	Remote	1723226	90208271	24.091611	0.000000	
3	AL	In-person	2022	31200	0.385816	-0.560592	
4	AL	Remote	4044	156600	0.771631	-1.121185	

In [97]:

```
# Removes all unnecessary columns from the
# dataframe and displays the first 5 rows.
final_df27.drop(
    final_df27.columns[[2, 4, 5, 6, 7]],
    axis = 1, inplace = True)

final_df27.head()
```

Out[97]:

	state	work_setting	salary_in_usd
0	0	Hybrid	2012427
1	0	In-person	114525460
2	0	Remote	90208271
3	AL	In-person	31200
4	AL	Remote	156600

In [98]:
```
# Creates a dataframe of valid States
# and displays the first 5 rows.
final_df28 = final_df27[
    final_df27.state != 0]

final_df28.head()
```

Out[98]:

	state	work_setting	salary_in_usd
3	AL	In-person	31200
4	AL	Remote	156600
5	AR	In-person	250300
6	AR	Remote	52600
7	AZ	In-person	724650

In [99]:
```
# Makes a copy of the dataframe
# and displays the first 5 rows.
df100 = final_df28

df100.head()
```

Out[99]:

	state	work_setting	salary_in_usd
3	AL	In-person	31200
4	AL	Remote	156600
5	AR	In-person	250300
6	AR	Remote	52600
7	AZ	In-person	724650

In [100]:
```
# Creates a dataframe and
# displays the first 5 rows.
dfremote = df100[
    df100.work_setting == "Remote"]

dfremote.head()
```

Out[100]:

	state	work_setting	salary_in_usd
4	AL	Remote	156600
6	AR	Remote	52600
8	AZ	Remote	128250
10	CA	Remote	2149518
12	CO	Remote	252738

In [101]:
```
# Creates a dataframe and
# displays the first 5 rows.
dfinperson = df100[
    df100.work_setting == "In-person"]

dfinperson.head()
```

Out[101]:

	state	work_setting	salary_in_usd
3	AL	In-person	31200
5	AR	In-person	250300
7	AZ	In-person	724650
9	CA	In-person	4840850
11	CO	In-person	525148

In [102]:
```
# Calculates the z-score for each state
# and displays the first 5 rows.
dfremote[
    'state_remote_salary_z_score'] = stats.zscore(
    dfremote.salary_in_usd).copy()

dfremote.head()
```

Out[102]:

	state	work_setting	salary_in_usd	state_remote_salary_z_score
4	AL	Remote	156600	-0.463614
6	AR	Remote	52600	-0.694958
8	AZ	Remote	128250	-0.526677
10	CA	Remote	2149518	3.969566
12	CO	Remote	252738	-0.249758

In [103]:
```
# Calculates the z-score for each state
# and displays the first 5 rows.
dfinperson[
    'state_inperson_salary_z_score'] = stats.zscore(
    dfinperson.salary_in_usd).copy()

dfinperson.head()
```

Out[103]:

	state	work_setting	salary_in_usd	state_inperson_salary_z_score
3	AL	In-person	31200	-0.734369
5	AR	In-person	250300	-0.493124
7	AZ	In-person	724650	0.029171
9	CA	In-person	4840850	4.561415
11	CO	In-person	525148	-0.190495

In [104]:

```
# Creates a finalized dataframe from merging the
# 2 dataframes and displays the first 5 rows.
df_ws = pd.merge(
    dfremote, dfinperson,
    how = 'outer',
    on = ['state',
          'work_setting']).copy()

# Removes 2 unnecessary columns.
df_ws.drop(df_ws.columns[[2, 4]],
           axis = 1, inplace = True)

df_ws.head()
```

Out[104]:

	state	work_setting	state_remote_salary_z_score	state_inperson_salary_z_score
0	AL	Remote	-0.463614	NaN
1	AR	Remote	-0.694958	NaN
2	AZ	Remote	-0.526677	NaN
3	CA	Remote	3.969566	NaN
4	CO	Remote	-0.249758	NaN

In [105]:

```
# Creates a finalized dataframe from merging the
# 2 dataframes and displays the first 5 rows.
df_final = pd.merge(
    final_df26, df_ws,
    how = 'left',
    on = ['state',
          'work_setting']).copy()

df_final.head()
```

Out[105]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [106]:
```
# Creates a new dataframe to calculate the z-scores
# for salaries using the groupby function for State
# and displays the first 5 rows
final_df29 = pd.concat(
    [final_df26.groupby(
    ['state',
     'salary_in_usd']).sum()]).reset_index()

final_df29.tail()
```

Out[106]:

	state	salary_in_usd	work_year	country_job_z_score	state_job_z_score	country_job_
1164	WI	71140	2022	0.385816	-0.485313	
1165	WI	78400	2022	0.385816	-0.485313	
1166	WI	97500	2022	0.385816	-0.485313	
1167	WY	73200	2022	0.385816	-0.635872	
1168	WY	88850	2022	0.385816	-0.635872	

In [107]:
```
# Removes 5 unnecessary columns.
final_df29.drop(
    final_df29.columns[[2, 3, 4, 5 ,6]],
    axis = 1, inplace = True)

final_df29.head()
```

Out[107]:

	state	salary_in_usd
0	0	6072
1	0	8000
2	0	9272
3	0	10000
4	0	10354

In [108]:
```
# Creates a dataframe of all valid States.
final_df30 = final_df29[
    final_df29.state != 0].copy()

final_df30.head()
```

Out[108]:

	state	salary_in_usd
715	AL	31200
716	AL	62700
717	AL	93900
718	AR	52600
719	AR	73400

In [109]:
```
# Calculates the salary z-score for each state.
final_df30['state_salary_z_score'] = stats.zscore(
    final_df.salary_in_usd).copy()

final_df30.head()
```

Out[109]:

	state	salary_in_usd	state_salary_z_score
715	AL	31200	-0.019920
716	AL	62700	-0.098118
717	AL	93900	-0.501325
718	AR	52600	-0.374254
719	AR	73400	0.239109

Create a Finalized Copy of the dataframe and a .csv file

In [110]:
```
# Creates a finalized dataframe from merging the
# 2 dataframes and displays the first 5 rows.
finaldf = pd.merge(
    df_final, final_df30,
    how = 'left',
    on = ['state', 'salary_in_usd']).copy()

finaldf.head()
```

Out[110]:

	job_title	state	work_setting	experience_level	employee_residence	work_year	salary_in
0	Data Analyst	0	In-person	Mid-level	United States	2022	9
1	Data Analyst	0	In-person	Mid-level	United States	2022	8
2	Data Analyst	CA	Remote	Mid-level	United States	2022	12
3	Data Analyst	0	Remote	Mid-level	United States	2022	10
4	Data Analyst	0	In-person	Mid-level	United States	2022	6

In [111]:
```
# Creates a .csv file
finaldf.to_csv(
    'World Data Analyst Jobs and Salaries 2020 - 2023.csv')
```

In [112]:

```
# Basic description of the dataframe.
finaldf.info()
```

```
Data columns (total 15 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   job_title                     2482 non-null   object
 1   state                         2482 non-null   object
 2   work_setting                  2482 non-null   object
 3   experience_level              2482 non-null   object
 4   employee_residence            2482 non-null   object
 5   work_year                     2482 non-null   int32
 6   salary_in_usd                 2482 non-null   int64
 7   company_size                  2482 non-null   object
 8   country_job_z_score           2482 non-null   float64
 9   state_job_z_score             2482 non-null   float64
 10  country_job_count             2482 non-null   int64
 11  state_job_count               2482 non-null   int32
 12  state_remote_salary_z_score   125 non-null    float64
 13  state_inperson_salary_z_score 366 non-null    float64
 14  state_salary_z_score          491 non-null    float64
dtypes: float64(5), int32(2), int64(2), object(6)
memory usage: 290.9+ KB
```