Introduction:

Analyzing the Jod Satisfaction and Salary disparities in data-related roles using the Data Professional Survey by Alex Freberg

About the Data Set:

This dataset was compiled by YouTuber Alex Freberg, more commonly recognized as Alex The Analyst. He conducted a survey among his audience, comprised mainly of data professionals, to assess the current landscape of data and those working within the field. The survey had the following questions;

- Date_Taken- Date Taken (America/New_York)
- List item - Time Taken (America/New_York)
- Time_Spent - Time Spent (America/New_York)
- Current_Role - Did you switch careers into Data?
- Switched_Careers - Did you switch careers into Data?
- Yearly_Salary - What is your currently annual salary?
- Average_yearly_Salary- this is the average of the current annual salary range
- Industry - What Industry do you work in?
- Programming_Language - Favorite Programming Language
- Satisfaction level - On a scale of 1-10, How happy are you with {Salary, Work/Life balance, Management, Co-Workers, Upward mobility, Opportunity to learn)
- Break_into_Data - The level of difficulty in breaking into data?
- New_Job_Type - If you were to look for a new job today, what would be the most important thing to you?
- Sex - Male or Female
- Age - Current Age
- Country - Which Country do you live in?
- Education - Highest Level of Education
- Ethnicity

Problem Statement Identification:

Investigating salary discrepancies and job satisfaction amongst data professionals based on factors such as job role, industry, demographic, sex, programming language, relationship with coworkers, management, upword mobiliity opportunities. learning environment and ethnicity.

Import Libraries:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the Dataset:

```
df = pd.read_csv("/content/data_professionals_survey.csv")
df
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Browser	OS	City	Country	Referrer	...	Managemen
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	NaN	NaN	NaN	NaN	NaN	...	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	NaN	NaN	NaN	NaN	NaN	...	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	NaN	NaN	NaN	NaN	NaN	...	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	NaN	NaN	NaN	NaN	NaN	...	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	NaN	NaN	NaN	NaN	NaN	...	
...	...	...	...	...	...	...	...	...	...	...	...	
625	62b525563f28f20328aeee5c	anonymous	6/23/2022	10:45:00 PM	12:00:50 AM	NaN	NaN	NaN	NaN	NaN	...	
626	62b5a3e29bc428d5345f6e89	anonymous	6/24/2022	7:45:00 AM	12:03:12 AM	NaN	NaN	NaN	NaN	NaN	...	
627	62b71083f31287f32e189026	anonymous	6/25/2022	9:41:00 AM	12:04:43 AM	NaN	NaN	NaN	NaN	NaN	...	
628	62b795033b026e423f287ecd	anonymous	6/25/2022	7:06:00 PM	12:02:17 AM	NaN	NaN	NaN	NaN	NaN	...	
629	62b89039377223ff07b80fb5	anonymous	6/26/2022	12:58:00 PM	12:01:19 AM	NaN	NaN	NaN	NaN	NaN	...	

630 rows × 29 columns

Data Exploration:

```
df.shape
```

```
(630, 29)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 630 entries, 0 to 629
Data columns (total 29 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Unique ID                       630 non-null    object
 1   Email                           630 non-null    object
```

```
 2   Date_Taken                      630 non-null    object
 3   Time_Taken                      630 non-null    object
 4   Time_Spent                      630 non-null    object
 5   Browser                         0 non-null      float64
 6   OS                              0 non-null      float64
 7   City                            0 non-null      float64
 8   Country                         0 non-null      float64
 9   Referrer                        0 non-null      float64
 10  Current_Role                    630 non-null    object
 11  Switched_Careers                630 non-null    object
 12  Yearly_Salary                   630 non-null    object
 13  Average_yearly_Salary           628 non-null    float64
 14  Industry                        630 non-null    object
 15  Programming_Language            630 non-null    object
 16  Salary_Satisfaction             623 non-null    float64
 17  Work/Life_Balance_Satisfaction  620 non-null    float64
 18  Coworkers_Satisfaction          619 non-null    float64
 19  Management_Satisfaction         618 non-null    float64
 20  Upward_Mobility_Satisfaction    617 non-null    float64
 21  Learning_New_Things_Satisfaction 625 non-null   float64
 22  Break_into_Data                 630 non-null    object
 23  New_Job_Type                    630 non-null    object
 24  Sex                             630 non-null    object
 25  Age                             630 non-null    int64
 26  Country.1                       630 non-null    object
 27  Education_Level                 578 non-null    object
 28  Ethnicity                       630 non-null    object
dtypes: float64(12), int64(1), object(16)
memory usage: 142.9+ KB
```

Data Preprocessing:

```
df.isnull().sum()
```

```
Unique ID                           0
Email                               0
Date_Taken                          0
Time_Taken                          0
Time_Spent                          0
Browser                           630
OS                                630
City                              630
Country                           630
Referrer                          630
Current_Role                        0
Switched_Careers                    0
Yearly_Salary                       0
Average_yearly_Salary               2
Industry                            0
Programming_Language                0
Salary_Satisfaction                 7
Work/Life_Balance_Satisfaction     10
Coworkers_Satisfaction             11
Management_Satisfaction            12
Upward_Mobility_Satisfaction       13
Learning_New_Things_Satisfaction    5
Break_into_Data                     0
New_Job_Type                        0
Sex                                 0
Age                                 0
Country.1                           0
Education_Level                    52
Ethnicity                           0
dtype: int64
```

Drop Empty Columns:

```
df.dropna(axis=1, how='all', inplace=True)
```

```
df.isnull().sum()
```

```
Unique ID                           0
Email                               0
Date_Taken                          0
Time_Taken                          0
Time_Spent                          0
Current_Role                        0
Switched_Careers                    0
Yearly_Salary                       0
Average_yearly_Salary               2
Industry                            0
Programming_Language                0
Salary_Satisfaction                 7
```

```
Work/Life_Balance_Satisfaction      10
Coworkers_Satisfaction              11
Management_Satisfaction             12
Upward_Mobility_Satisfaction        13
Learning_New_Things_Satisfaction     5
Break_into_Data                      0
New_Job_Type                         0
Sex                                  0
Age                                  0
Country.1                            0
Education_Level                     52
Ethnicity                            0
dtype: int64
```

```
df["Average_yearly_Salary"]
```

```
0      116.0
1       53.0
2       20.0
3      188.0
4       53.0
       ...
625    138.0
626     20.0
627     20.0
628     20.0
629     53.0
Name: Average_yearly_Salary, Length: 630, dtype: float64
```

Impute with Constant Value:

```
df["Average_yearly_Salary"].fillna("undefined", inplace=True)
```

```
df["Average_yearly_Salary"].isnull().sum()
```

```
0
```

```
df["Education_Level"].unique()
```

```
array([nan, 'High School', 'Bachelors', 'Masters', 'Associates', 'PhD'],
      dtype=object)
```

```
df["Education_Level"].fillna("undefined", inplace=True)
```

```
df["Education_Level"].isnull().sum()
```

```
0
```

Impute with mode:

```
df["Salary_Satisfaction"].unique()
```

```
array([ 9.,  1.,  0., 10.,  2.,  4.,  3.,  7.,  5.,  6.,  8., nan])
```

```
salary_filling = df["Salary_Satisfaction"].mode().iloc[0]
salary_filling
```

```
3.0
```

```
df["Salary_Satisfaction"].fillna("salary_filling", inplace=True)
```

```
df["Salary_Satisfaction"].isnull().sum()
```

```
0
```

```
df.isnull().sum()
```

```
Unique ID                           0
Email                               0
Date_Taken                          0
Time_Taken                          0
Time_Spent                          0
Current_Role                        0
Switched_Careers                    0
Yearly_Salary                       0
Average_yearly_Salary               0
Industry                            0
```

```
Programming_Language                0
Salary_Satisfaction                 0
Work/Life_Balance_Satisfaction     10
Coworkers_Satisfaction             11
Management_Satisfaction            12
Upward_Mobility_Satisfaction       13
Learning_New_Things_Satisfaction    5
Break_into_Data                     0
New_Job_Type                        0
Sex                                 0
Age                                 0
Country.1                           0
Education_Level                     0
Ethnicity                           0
dtype: int64
```

```
df["Work/Life_Balance_Satisfaction"].unique()
```

```
array([ 9.,  2.,  8.,  6.,  4.,  3.,  5.,  0., 10.,  1.,  7., nan])
```

```
work_filling = df["Work/Life_Balance_Satisfaction"].mode().iloc[0]
work_filling
```

```
6.0
```

```
df["Work/Life_Balance_Satisfaction"].fillna("work_filling", inplace=True)
```

```
df["Work/Life_Balance_Satisfaction"].isnull().sum()
```

```
0
```

```
coworkers_filling = df["Coworkers_Satisfaction"].mode().iloc[0]
coworkers_filling
```

```
5.0
```

```
df["Coworkers_Satisfaction"].fillna("coworkers_filling", inplace=True)
```

```
df["Coworkers_Satisfaction"].isnull().sum()
```

```
0
```

```
management_filling = df["Management_Satisfaction"].mode().iloc[0]
management_filling
```

```
5.0
```

```
df["Management_Satisfaction"].fillna("management_filling", inplace=True)
```

```
df["Management_Satisfaction"].isnull().sum()
```

```
0
```

```
upward_filling = df["Upward_Mobility_Satisfaction"].mode().iloc[0]
upward_filling
```

```
5.0
```

```
df["Upward_Mobility_Satisfaction"].fillna("upward_filling", inplace=True)
```

```
df["Upward_Mobility_Satisfaction"].isnull().sum()
```

```
0
```

```
learning_filling = df["Learning_New_Things_Satisfaction"].mode().iloc[0]
learning_filling
```

```
10.0
```

```
df["Learning_New_Things_Satisfaction"].fillna("learning_filling", inplace=True)
```

```
df["Learning_New_Things_Satisfaction"].isnull().sum()
```

```
0
```

```
df.isnull().sum()
```

```
Unique ID                           0
Email                               0
Date_Taken                          0
Time_Taken                          0
Time_Spent                          0
Current_Role                        0
Switched_Careers                    0
Yearly_Salary                       0
Average_yearly_Salary               0
Industry                            0
Programming_Language                0
Salary_Satisfaction                 0
Work/Life_Balance_Satisfaction      0
Coworkers_Satisfaction              0
Management_Satisfaction             0
Upward_Mobility_Satisfaction        0
Learning_New_Things_Satisfaction    0
Break_into_Data                     0
New_Job_Type                        0
Sex                                 0
Age                                 0
Country.1                           0
Education_Level                     0
Ethnicity                           0
dtype: int64
```

Data Type Issues:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 630 entries, 0 to 629
Data columns (total 24 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   Unique ID                         630 non-null    object
 1   Email                             630 non-null    object
 2   Date_Taken                        630 non-null    object
 3   Time_Taken                        630 non-null    object
 4   Time_Spent                        630 non-null    object
 5   Current_Role                      630 non-null    object
 6   Switched_Careers                  630 non-null    object
 7   Yearly_Salary                     630 non-null    object
 8   Average_yearly_Salary             630 non-null    object
 9   Industry                          630 non-null    object
 10  Programming_Language              630 non-null    object
 11  Salary_Satisfaction               630 non-null    object
 12  Work/Life_Balance_Satisfaction    630 non-null    object
 13  Coworkers_Satisfaction            630 non-null    object
 14  Management_Satisfaction           630 non-null    object
 15  Upward_Mobility_Satisfaction      630 non-null    object
 16  Learning_New_Things_Satisfaction  630 non-null    object
 17  Break_into_Data                   630 non-null    object
 18  New_Job_Type                      630 non-null    object
 19  Sex                               630 non-null    object
 20  Age                               630 non-null    int64
 21  Country.1                         630 non-null    object
 22  Education_Level                   630 non-null    object
 23  Ethnicity                         630 non-null    object
dtypes: int64(1), object(23)
memory usage: 118.2+ KB
```

```
df["Average_yearly_Salary"].unique()
```

```
array([116.0, 53.0, 20.0, 188.0, 138.0, 96.0, 76.0, 'undefined'],
      dtype=object)
```

```
df['Average_yearly_Salary'] = df['Average_yearly_Salary'].replace("undefined", 0)
```

```
df['Average_yearly_Salary'] = df['Average_yearly_Salary'].astype(int)
```

```
df["Average_yearly_Salary"].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 630 entries, 0 to 629
Series name: Average_yearly_Salary
Non-Null Count  Dtype
--------------  -----
630 non-null    int64
```

```
dtypes: int64(1)
memory usage: 5.0 KB
```

Duplicates:

```
df.duplicated().sum()
```

```
0
```

Outliers:

IQR:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 630 entries, 0 to 629
Data columns (total 24 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   Unique ID                         630 non-null    object
 1   Email                             630 non-null    object
 2   Date_Taken                        630 non-null    object
 3   Time_Taken                        630 non-null    object
 4   Time_Spent                        630 non-null    object
 5   Current_Role                      630 non-null    object
 6   Switched_Careers                  630 non-null    object
 7   Yearly_Salary                     630 non-null    object
 8   Average_yearly_Salary             630 non-null    int64
 9   Industry                          630 non-null    object
 10  Programming_Language              630 non-null    object
 11  Salary_Satisfaction               630 non-null    object
 12  Work/Life_Balance_Satisfaction    630 non-null    object
 13  Coworkers_Satisfaction            630 non-null    object
 14  Management_Satisfaction           630 non-null    object
 15  Upward_Mobility_Satisfaction      630 non-null    object
 16  Learning_New_Things_Satisfaction  630 non-null    object
 17  Break_into_Data                   630 non-null    object
 18  New_Job_Type                      630 non-null    object
 19  Sex                               630 non-null    object
 20  Age                               630 non-null    int64
 21  Country.1                         630 non-null    object
 22  Education_Level                   630 non-null    object
 23  Ethnicity                         630 non-null    object
dtypes: int64(2), object(22)
memory usage: 118.2+ KB
```

```
df.describe()
```

	Average_yearly_Salary	Age
count	630.000000	630.000000
mean	53.350794	29.866667
std	38.352633	7.245941
min	0.000000	18.000000
25%	20.000000	25.000000
50%	53.000000	28.000000
75%	76.000000	33.000000
max	188.000000	92.000000

```
Q1 = df["Average_yearly_Salary"].quantile(0.25)
Q3 = df["Average_yearly_Salary"].quantile(0.75)
IQR = Q3 - Q1
```

```
lower_threshold = Q1 - 1.5 * IQR
upper_threshold = Q3 - 1.5 * IQR
```

```
outliers = df[(df["Average_yearly_Salary"] < lower_threshold) | (df["Average_yearly_Salary"] > upper_threshold)]
outliers
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Averag
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	Data Analyst	No	41k-65k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	Data Analyst	Yes	41k-65k	
...	...	...	...	...	...	...	...	...	
625	62b525563f28f20328aeee5c	anonymous	6/23/2022	10:45:00 PM	12:00:50 AM	Data Analyst	Yes	125k-150k	
626	62b5a3e29bc428d5345f6e89	anonymous	6/24/2022	7:45:00 AM	12:03:12 AM	Other	No	0-40k	
627	62b71083f31287f32e189026	anonymous	6/25/2022	9:41:00 AM	12:04:43 AM	Student	Yes	0-40k	
628	62b795033b026e423f287ecd	anonymous	6/25/2022	7:06:00 PM	12:02:17 AM	Data Engineer	No	0-40k	
629	62b89039377223ff07b80fb5	anonymous	6/26/2022	12:58:00 PM	12:01:19 AM	Data Analyst	No	41k-65k	

630 rows × 24 columns

```
print(Q1)
print(Q3)
print(IQR)
```

```
20.0
76.0
56.0
```

```
Q1 = df["Age"].quantile(0.25)
Q3 = df["Age"].quantile(0.75)
IQR = Q3 - Q1
```

```
lower_threshold = Q1 - 1.5 * IQR
upper_threshold = Q3 - 1.5 * IQR
```

```
outliers = df[(df["Age"] < lower_threshold) | (df["Age"] > upper_threshold)]
outliers
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Averag
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	Data Analyst	No	41k-65k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	Data Analyst	Yes	41k-65k	
...	...	...	...	...	...	...	...	...	
624	62b4404df31287f32e14d1c1	anonymous	6/23/2022	6:28:00 AM	12:01:20 AM	Database Developer	Yes	0-40k	
625	62b525563f28f20328aeee5c	anonymous	6/23/2022	10:45:00 PM	12:00:50 AM	Data Analyst	Yes	125k-150k	
627	62b71083f31287f32e189026	anonymous	6/25/2022	9:41:00 AM	12:04:43 AM	Student	Yes	0-40k	
628	62b795033b026e423f287ecd	anonymous	6/25/2022	7:06:00 PM	12:02:17 AM	Data Engineer	No	0-40k	
629	62b89039377223ff07b80fb5	anonymous	6/26/2022	12:58:00 PM	12:01:19 AM	Data Analyst	No	41k-65k	

611 rows × 24 columns

```
print(Q1)
print(Q3)
print(IQR)
```

```
25.0
33.0
8.0
```

Z-Score:

```
mean = np.nanmean(df.Average_yearly_Salary.values.tolist())
std = np.nanstd(df.Average_yearly_Salary.values.tolist())
```

```
print(mean)
print(std)
```

```
53.35079365079365
38.32218211724459
```

```
df["Zscor_Average_yearly_Salary"] = (df.Average_yearly_Salary - mean)
```

```
df.head()
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Average_
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	Data Analyst	No	41k-65k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	Data Analyst	Yes	41k-65k	

5 rows × 25 columns

```
# extract outliers
z_outliers = df[(df.Zscor_Average_yearly_Salary < -3) | (df.Zscor_Average_yearly_Salary > 3)]
```

```
z_outliers
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Averag
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
5	62a33cb6cf25554317300177	anonymous	6/10/2022	8:44:00 AM	12:02:34 AM	Data Analyst	Yes	0-40k	
6	62a33cb72e54c9003e531c65	anonymous	6/10/2022	8:44:00 AM	12:01:15 AM	Data Scientist	Yes	0-40k	
...	...	...	...	...	...	...	...	...	
624	62b4404df31287f32e14d1c1	anonymous	6/23/2022	6:28:00 AM	12:01:20 AM	Database Developer	Yes	0-40k	
625	62b525563f28f20328aeee5c	anonymous	6/23/2022	10:45:00 PM	12:00:50 AM	Data Analyst	Yes	125k-150k	
626	62b5a3e29bc428d5345f6e89	anonymous	6/24/2022	7:45:00 AM	12:03:12 AM	Other	No	0-40k	
627	62b71083f31287f32e189026	anonymous	6/25/2022	9:41:00 AM	12:04:43 AM	Student	Yes	0-40k	
628	62b795033b026e423f287ecd	anonymous	6/25/2022	7:06:00 PM	12:02:17 AM	Data Engineer	No	0-40k	

481 rows × 25 columns

```
mean = np.nanmean(df.Age.values.tolist())
std = np.nanstd(df.Age.values.tolist())
```

```
print(mean)
print(std)
```

```
29.866666666666667
7.24018766747917
```

```
df["Zscor_Age"] = (df.Age - mean)
```

```
df.head()
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Average_
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	Data Analyst	No	41k-65k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	Data Analyst	Yes	41k-65k	

5 rows × 26 columns

```
# extract outliers
z_outliers = df[(df.Zscor_Age < -3) | (df.Zscor_Age > 3)]
```

```
z_outliers
```

	Unique ID	Email	Date_Taken	Time_Taken	Time_Spent	Current_Role	Switched_Careers	Yearly_Salary	Averag
0	62a33b3db4da29969c62df3d	anonymous	6/10/2022	8:38:00 AM	12:00:44 AM	Data Analyst	Yes	106k-125k	
1	62a33ba1bae91e4b8b82e35c	anonymous	6/10/2022	8:40:00 AM	12:01:30 AM	Data Analyst	No	41k-65k	
2	62a33c2cbc6861bf3176bec1	anonymous	6/10/2022	8:42:00 AM	12:02:18 AM	Data Engineer	No	0-40k	
3	62a33c8624a26260273822f9	anonymous	6/10/2022	8:43:00 AM	12:02:10 AM	Other	Yes	150k-225k	
4	62a33c91f3072dd892621e03	anonymous	6/10/2022	8:44:00 AM	12:01:51 AM	Data Analyst	Yes	41k-65k	
...	...	...	...	...	...	...	...	...	
625	62b525563f28f20328aeee5c	anonymous	6/23/2022	10:45:00 PM	12:00:50 AM	Data Analyst	Yes	125k-150k	
626	62b5a3e29bc428d5345f6e89	anonymous	6/24/2022	7:45:00 AM	12:03:12 AM	Other	No	0-40k	
627	62b71083f31287f32e189026	anonymous	6/25/2022	9:41:00 AM	12:04:43 AM	Student	Yes	0-40k	
628	62b795033b026e423f287ecd	anonymous	6/25/2022	7:06:00 PM	12:02:17 AM	Data Engineer	No	0-40k	
629	62b89039377223ff07b80fb5	anonymous	6/26/2022	12:58:00 PM	12:01:19 AM	Data Analyst	No	41k-65k	

410 rows × 26 columns

Exploratory Data Analysis

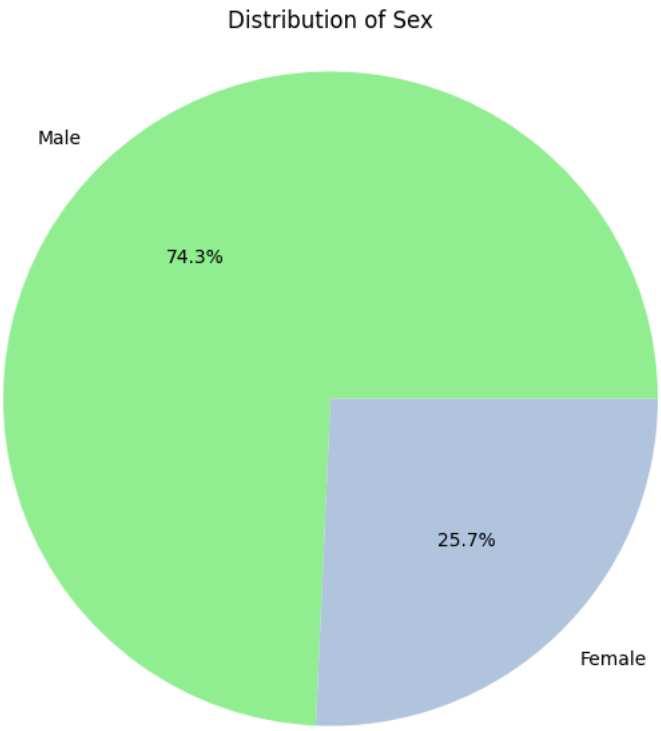
```
df.columns
```

```
Index(['Unique ID', 'Email', 'Date_Taken', 'Time_Taken', 'Time_Spent',
       'Current_Role', 'Switched_Careers', 'Yearly_Salary',
       'Average_yearly_Salary', 'Industry', 'Programming_Language',
       'Salary_Satisfaction', 'Work/Life_Balance_Satisfaction',
       'Coworkers_Satisfaction', 'Management_Satisfaction',
       'Upward_Mobility_Satisfaction', 'Learning_New_Things_Satisfaction',
       'Break_into_Data', 'New_Job_Type', 'Sex', 'Age', 'Country.1',
       'Education_Level', 'Ethnicity', 'Zscor_Average_yearly_Salary',
       'Zscor_Age'],
      dtype='object')
```

What is the Gender ratio of the Data Professionals?

```
# Get value counts for the "Sex" column
sex_rate = df["Sex"].value_counts()
```

```
# Plotting
plt.figure(figsize=(8, 7))  # Set the size of the plot
plt.pie(sex_rate.values, labels=sex_rate.index, colors=["lightgreen", "lightsteelblue"], autopct="%1.1f%%")
plt.title("Distribution of Sex")
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



Insight: There's a significant disproportion between the number of male and female data professionals who participated in the survey.

Double-click (or enter) to edit

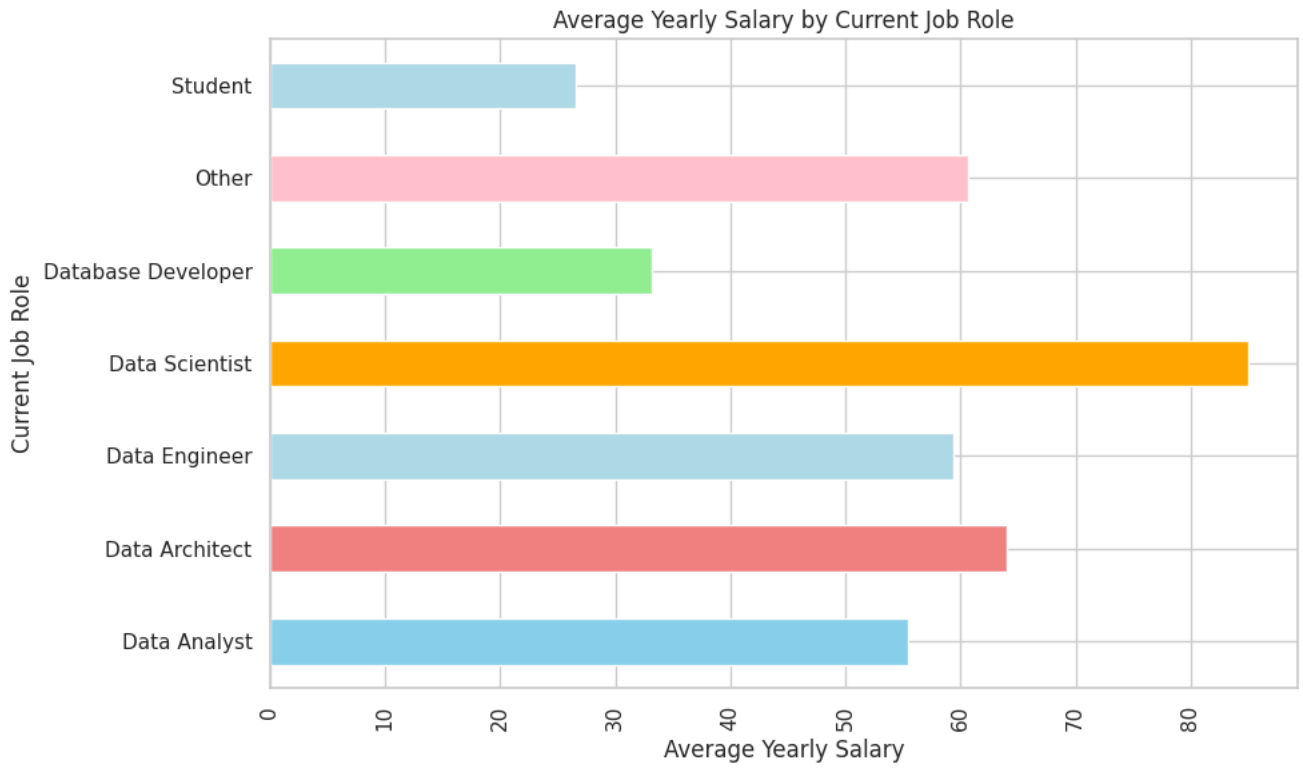
How does average yearly salary vary across different current job roles?

```
# Get value counts for the "Current_Role" column
Current_Role_rate = df["Current_Role"].value_counts()
```

```
# Calculate average salary for each current job role
avg_salary = df.groupby("Current_Role")["Average_yearly_Salary"].mean()
```

```
# Define colors for each bar
colors = ['skyblue', 'lightcoral', 'lightblue', 'orange', 'lightgreen', 'pink', 'lightblue']

# Plotting
plt.figure(figsize=(10, 6))  # Set the size of the plot
avg_salary.plot(kind='barh', color=colors)
plt.title("Average Yearly Salary by Current Job Role")
plt.xlabel("Average Yearly Salary")
plt.ylabel("Current Job Role")
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.tight_layout()  # Adjust layout to prevent labels from being cut off
plt.show()
```



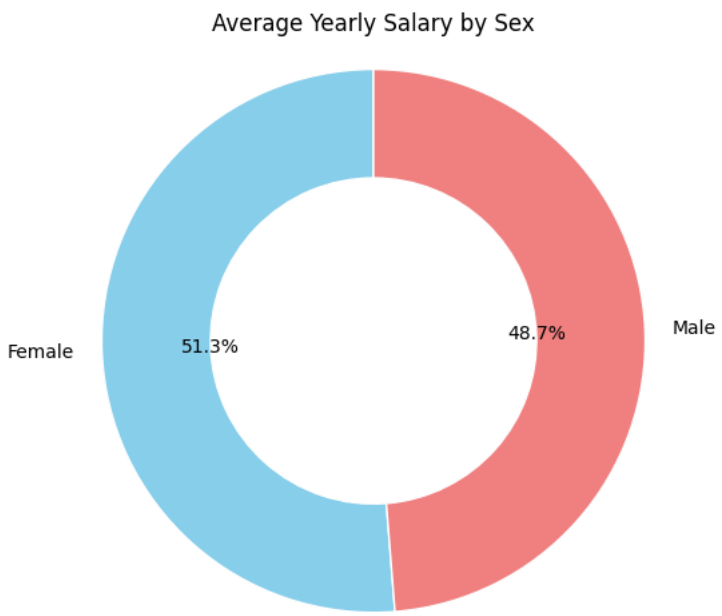
Insight: Data scientists consistently earn the highest salaries compared to other roles within the data industry, indicating that their specialized skills and expertise are highly valued in the job market. This suggests that individuals pursuing careers in data science may have greater earning potential relative to other data-related professions.

Double-click (or enter) to edit

What are the differences in average yearly salary between different genders?

```
# Calculate average salary for each sex
avg_salary_by_sex = df.groupby("Sex")["Average_yearly_Salary"].mean()
```

```
plt.figure(figsize=(8, 5))  # Set the size of the plot
plt.pie(avg_salary_by_sex.values, labels=avg_salary_by_sex.index, colors=['skyblue', 'lightcoral'], autopct='%1.1f%%', startangle=90, we
plt.title("Average Yearly Salary by Sex")
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.tight_layout()
plt.show()
```



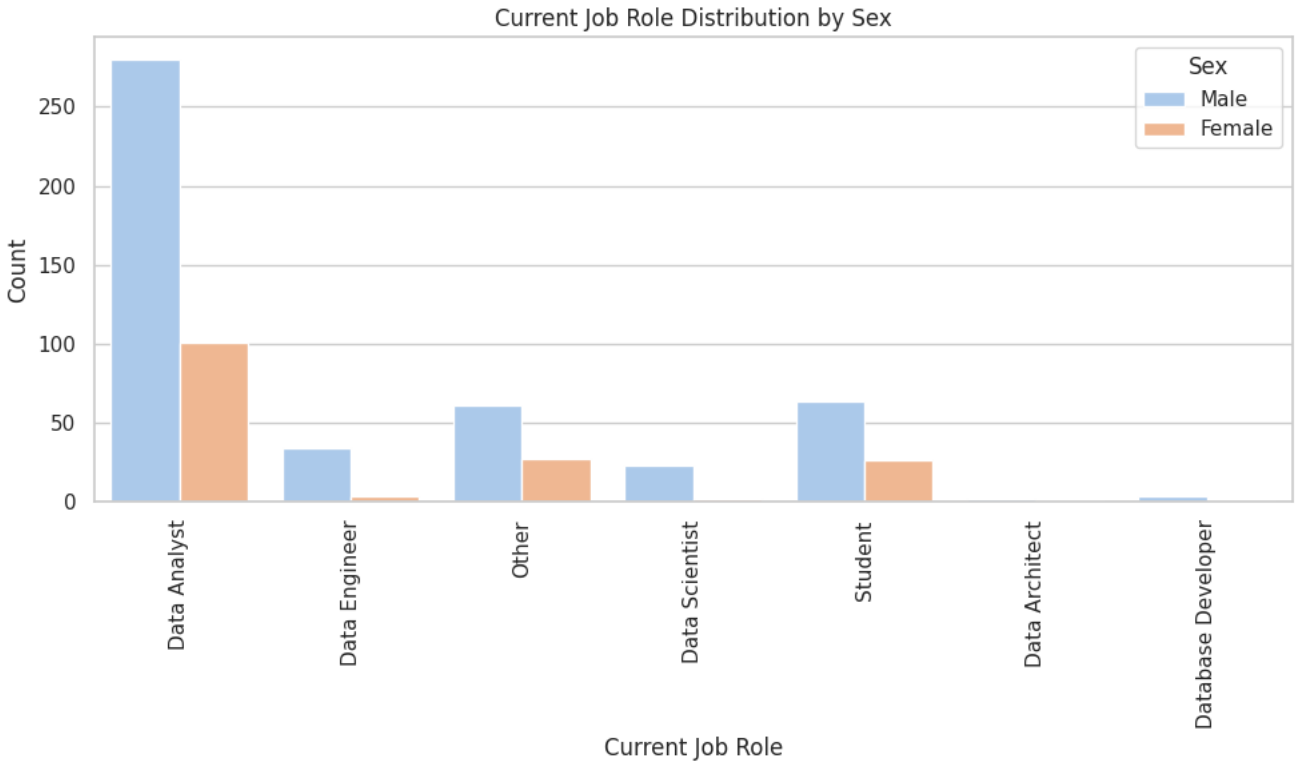
Insight: In the realm of data professions, women outpaced men in earnings, marking a significant shift in gender-based income differentials within the field. This trend underscores a promising stride towards gender parity and recognition of female expertise in data-related roles.

Double-click (or enter) to edit

How is the distribution of current job roles different between genders?

```
# Set seaborn style
sns.set(style="whitegrid")
```

```
# Plotting
plt.figure(figsize=(10, 6))  # Set the size of the plot
sns.countplot(data=df, x="Current_Role", hue="Sex", palette="pastel")
plt.title("Current Job Role Distribution by Sex")
plt.xlabel("Current Job Role")
plt.ylabel("Count")
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.legend(title="Sex")
plt.tight_layout()
plt.show()
```



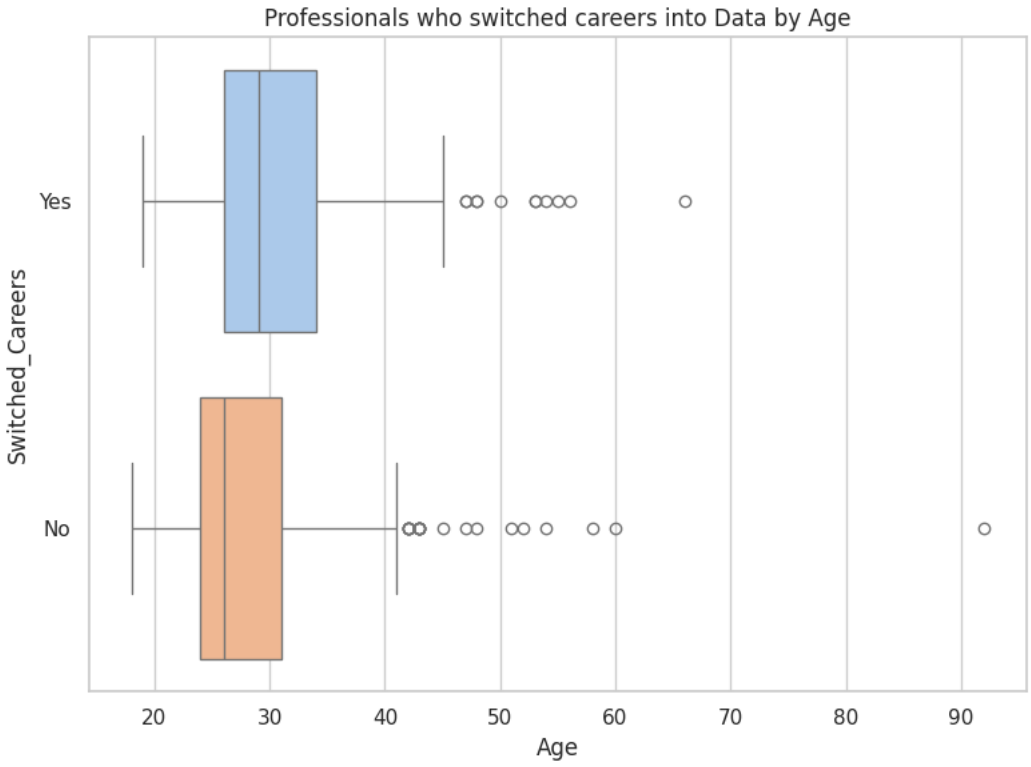
Insight: Males significantly outnumber females especially in the Data Analysis profession, indicating a gender imbalance within this specific profession. Addressing this gap is crucial for fostering diversity and inclusivity in data-driven industries.

Double-click (or enter) to edit

** What is the distribution of professionals who transitioned into data-related careers based on their age?**

```
# Set seaborn style
sns.set(style="whitegrid")
```

```
# Plotting
plt.figure(figsize=(8, 6))  # Set the size of the plot
sns.boxplot(data=df, x="Age", y="Switched_Careers", palette="pastel", hue="Switched_Careers")
plt.title("Professionals who switched careers into Data by Age")
plt.xlabel("Age")
plt.ylabel("Switched_Careers")
plt.tight_layout()
plt.show()
```

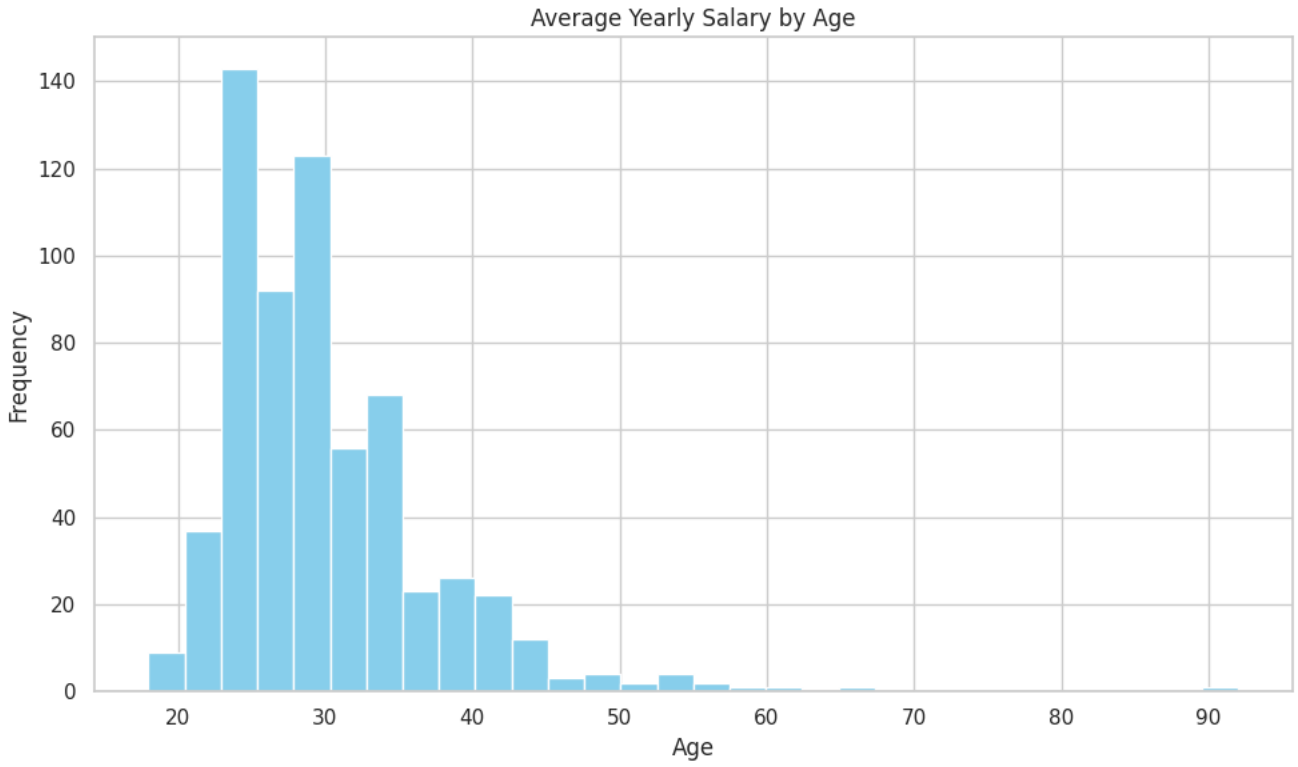


Insight: There are more outliers among older individuals who didnt transition into data professions compared to those who did switch careers meanig those who remained in their original professions have more extreme values (outliers) compared to those who transitioned into data-related professions.

Double-click (or enter) to edit

** What is the relationship between Average Yearly Salary and Age?**

```
# Plotting
plt.figure(figsize=(10, 6))  # Set the size of the plot
plt.hist(df["Age"], bins=30, color='skyblue', edgecolor= None )
plt.title("Average Yearly Salary by Age")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.tight_layout()
plt.show()
```

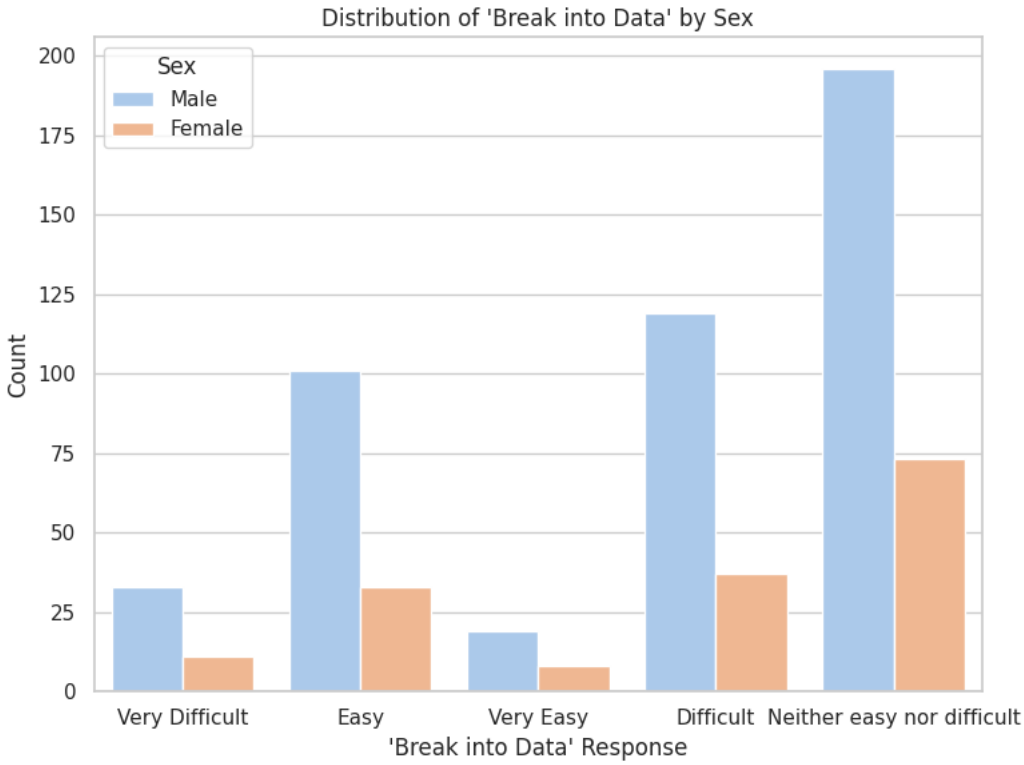


Insight: Data professionals aged between 25 and 35 earn the highest average yearly salary. This suggests that individuals within this age range tend to command higher salaries in the data industry compared to other age groups.

How does the distribution of The level of difficulty in breaking into data differ between genders

```
# Set seaborn style
sns.set(style="whitegrid")

# Plotting
plt.figure(figsize=(8, 6))  # Set the size of the plot
sns.countplot(data=df, x="Break_into_Data", hue="Sex", palette="pastel")
plt.title("Distribution of 'Break into Data' by Sex")
plt.xlabel("'Break into Data' Response")
plt.ylabel("Count")
plt.legend(title="Sex")
plt.tight_layout()
plt.show()
```



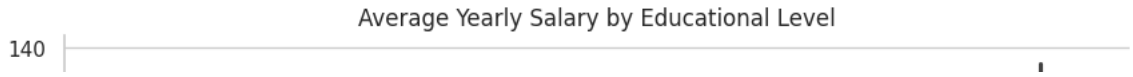
Insight: Both male and female data professionals rated the level of difficulty in breaking into the data field as moderate, with the highest response frequency. This indicates that both genders perceive the challenge of entering the data industry similarly, neither finding it excessively easy nor overly difficult.

What is the correlation between educational level and average yearly salary?

```
# Set seaborn style
sns.set(style="whitegrid")

# Plotting
plt.figure(figsize=(10, 6))  # Set the size of the plot
sns.catplot(data=df, x="Education_Level", y="Average_yearly_Salary", hue="Education_Level", kind="bar", palette="muted", height=6, aspe
plt.title("Average Yearly Salary by Educational Level")
plt.xlabel("Educational Level")
plt.ylabel("Average Yearly Salary")
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```



Insight: Data professionals with a Ph.D. education level earned significantly higher salaries compared to those with lower levels of education. This suggests that advanced educational qualifications, such as a Ph.D., are associated with higher earning potential in the data industry.

Double-click (or enter) to edit

What is the relationship between ethnicity and average yearly salary?

```
# Set seaborn style
sns.set(style="whitegrid")

# Plotting
plt.figure(figsize=(12, 8))  # Set the size of the plot
sns.lineplot(data=df, x="Programming_Language", y="Average_yearly_Salary", hue="Country.1", marker='o', palette="muted")
```