In [296…
```
import pandas as pd

# Specify the path to your Excel file
excel_file_path = 'Germany ICP - Case study-Copy1.xlsx'  # Replace with your file p

# Read the Excel file into a DataFrame
df = pd.read_excel(excel_file_path)
```

In [ ]:
```
```

In [297…
```
df = df.drop('Company_country_name', axis=1)
df
```

Out[297]:

	ID	Amount_in_Euro	Close_Date	Revised_number_of_employees	Deal_Provenance	Deal_Orig
0	1	135.0	2022-12-30 23:59:00	15	Inbound	Signu Complete
1	2	412.5	2022-12-30 16:14:00	55	Inbound	Signu Complete
2	3	90.0	2022-12-27 11:45:00	15	Inbound	Chann inbour
3	4	180.0	2022-12-21 13:12:00	15	Inbound	Signu Complete
4	5	108.0	2022-12-29 12:45:00	15	Paid	Signu Complete
…	…	…	…	…	…	
257	258	149.0	2022-02-28 23:01:00	10	Outbound	Linked
258	259	1200.0	2022-07-12 13:48:00	100	Inbound	Inbour Dem
259	260	144.0	2022-02-04 10:27:00	16	Inbound	Inbour Pa
260	261	269.0	2022-10-21 17:31:00	35	Outbound	Linked
261	262	495.0	2022-11-25 15:43:00	55	Outbound	Linked

262 rows × 16 columns

In [298…
```
new_df = df.iloc[:, [3,4,5, 6,14,11,13]]
new_df
```

Out[298]:

	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_Date	Industry	Cl
0	15	Inbound	Signup Completed	2022-12-30 12:45:00	Other	
1	55	Inbound	Signup Completed	2022-12-30 12:00:00	Tourism	
2	15	Inbound	Channel inbound	2022-12-22 16:55:00	Consumer staples	
3	15	Inbound	Signup Completed	2022-12-21 11:10:00	Other	
4	15	Paid	Signup Completed	2022-12-19 16:43:00	Other	
...	...	...	...	...	...	
257	10	Outbound	LinkedIn	2021-11-18 12:04:00	Health care	
258	100	Inbound	Inbound Demo	2021-10-07 12:43:00	Energy	
259	16	Inbound	Inbound Paid	2021-09-30 17:58:00	Information technology	
260	35	Outbound	LinkedIn	2021-08-04 17:59:00	Communication	
261	55	Outbound	LinkedIn	2021-07-28 10:36:00	Industrials	

262 rows × 7 columns

In [17]:
```
#Converting objects into numricals

from sklearn import preprocessing

label_encoder = preprocessing.LabelEncoder()
new_df["Revised_number_of_employees"] = label_encoder.fit_transform(new_df["Revised
new_df["Deal_Provenance"] = label_encoder.fit_transform(new_df["Deal_Provenance"])
new_df["Deal_Origin"] = label_encoder.fit_transform(new_df["Deal_Origin"])
new_df["Industry"] = label_encoder.fit_transform(new_df["Industry"])
new_df["Decision_Maker"] = label_encoder.fit_transform(new_df["Decision_Maker"])
new_df["Champion"] = label_encoder.fit_transform(new_df["Champion"])
```

In [18]:
```
#renaming since it doesnt indentify
new_df = new_df.rename(columns={new_df.columns[2]: 'Deal_Origin'})
new_df = new_df.rename(columns={new_df.columns[6]: 'Decision_Maker'})
new_df = new_df.rename(columns={new_df.columns[0]: 'Revised_number_of_employees'})
new_df = new_df.rename(columns={new_df.columns[4]: 'Industry'})
```

In [19]:
```
new_df
```

Out[19]:

	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_Date	Industry	Champic
0	10	0	13	2022-12-30 12:45:00	15	
1	44	0	13	2022-12-30 12:00:00	18	
2	10	0	0	2022-12-22 16:55:00	4	
3	10	0	13	2022-12-21 11:10:00	15	
4	10	2	13	2022-12-19 16:43:00	15	
...	...	...	...	...	...	
257	6	1	9	2021-11-18 12:04:00	9	
258	55	0	5	2021-10-07 12:43:00	6	
259	11	0	8	2021-09-30 17:58:00	12	
260	29	1	9	2021-08-04 17:59:00	1	
261	44	1	9	2021-07-28 10:36:00	11	

262 rows × 7 columns

In [20]:
```
for i, col_name in enumerate(new_df.columns):
    print(f"Column {i}: {col_name}")
```

```
Column 0: Revised_number_of_employees
Column 1: Deal_Provenance
Column 2: Deal_Origin
Column 3: Create_Date
Column 4: Industry
Column 5: Champion
Column 6: Decision_Maker
```

In [21]:
```
new_df
```

Out[21]:

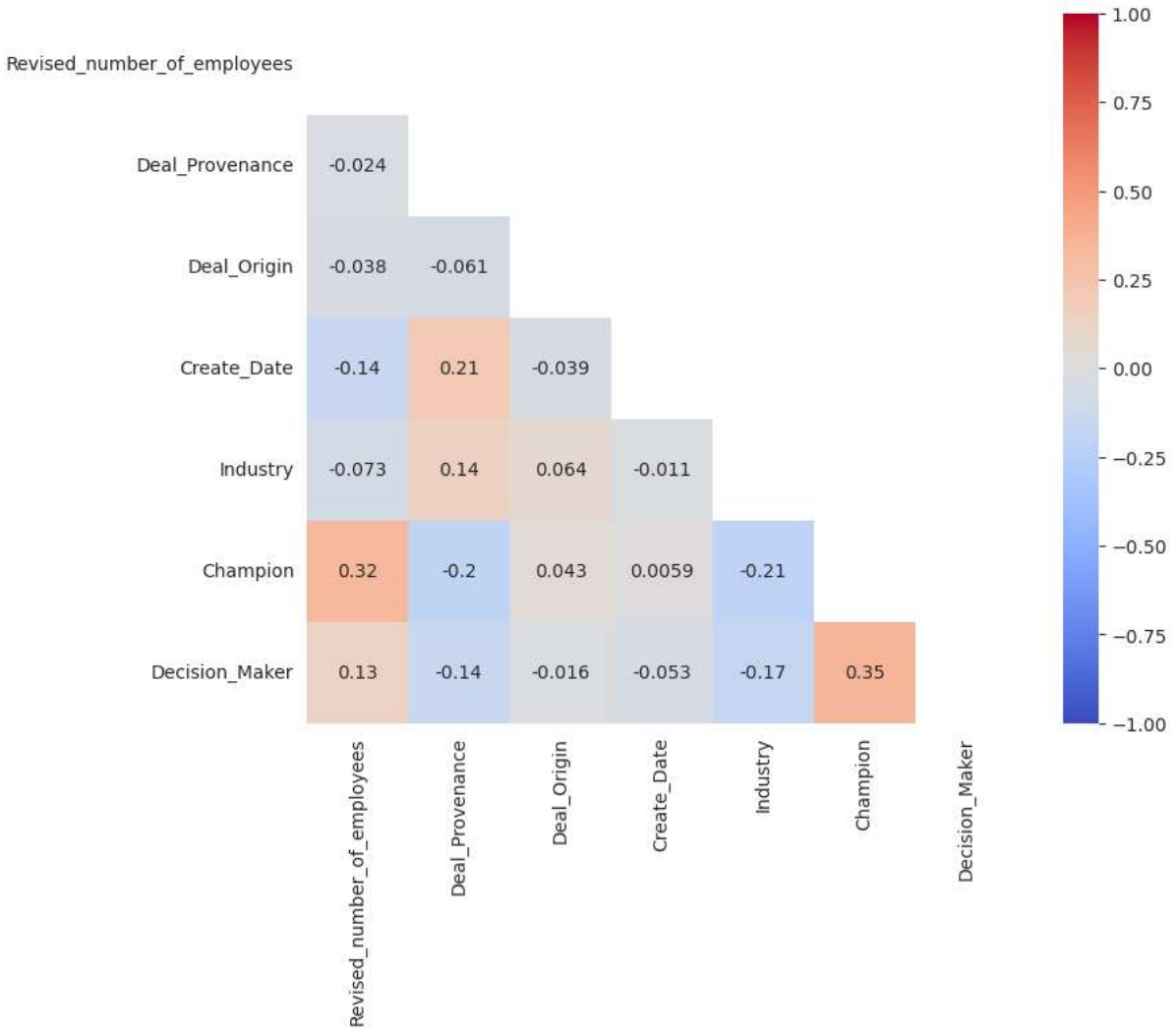
	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_Date	Industry	Champic
0	10	0	13	2022-12-30 12:45:00	15	
1	44	0	13	2022-12-30 12:00:00	18	
2	10	0	0	2022-12-22 16:55:00	4	
3	10	0	13	2022-12-21 11:10:00	15	
4	10	2	13	2022-12-19 16:43:00	15	
...	...	...	...	...	...	
257	6	1	9	2021-11-18 12:04:00	9	
258	55	0	5	2021-10-07 12:43:00	6	
259	11	0	8	2021-09-30 17:58:00	12	
260	29	1	9	2021-08-04 17:59:00	1	
261	44	1	9	2021-07-28 10:36:00	11	

262 rows × 7 columns

In [22]:

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

corr = new_df.corr()
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(9, 7))
    ax = sns.heatmap(corr, mask=mask,cmap='coolwarm', vmin=-1,vmax=1,annot=True, sc
```



In [23]:
```
new_df.corr()
```

Out[23]:

	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create
Revised_number_of_employees	1.000000	-0.023633	-0.037881	-0.1
Deal_Provenance	-0.023633	1.000000	-0.061155	0.2
Deal_Origin	-0.037881	-0.061155	1.000000	-0.0
Create_Date	-0.143190	0.206145	-0.039143	1.0
Industry	-0.073256	0.142201	0.063993	-0.0
Champion	0.321200	-0.197965	0.043499	0.0
Decision_Maker	0.125450	-0.142064	-0.015838	-0.0

In [24]:
```
#select industry, round(avg(Revised_number_of_employees),2),
#round(avg(Amount_in_Euro),2),round(sum(Amount_in_Euro),2) from Company group by ir
```

In [25]:
```
#-------------------------------------- INDUTRIAL SECTOR PROFIL-----------------
```

In [26]:
```
df["Revised_number_of_employees"].mean()
```

Out[26]:
```
30.041984732824428
```

In [27]:
```
import pandas as pd
```

```
# Group by the "industry" column and calculate the mean for "Revised_number_of_empl
mean_employees_by_industry = df.groupby('Industry')['Revised_number_of_employees'].

# Convert the result to a DataFrame (optional)
mean_employees_by_industry = mean_employees_by_industry.reset_index()

# Rename the columns for clarity (optional)
mean_employees_by_industry.columns = ['Industry', 'mean_employees']

# Display the result
print(mean_employees_by_industry)
```

	Industry	mean_employees
0	Agriculture	30.000000
1	Communication	37.947368
2	Construction	18.000000
3	Consumer discretionary	48.833333
4	Consumer staples	21.666667
5	E-learning	15.000000
6	Energy	32.857143
7	Entertainment	39.857143
8	Finance	27.000000
9	Health care	22.045455
10	Hospitality	22.666667
11	Industrials	38.976744
12	Information technology	23.833333
13	Legal services	23.500000
14	Non-Profit Organization Management	31.000000
15	Other	29.575758
16	Real estate	27.153846
17	Retail	12.000000
18	Tourism	55.000000
19	Transportation	18.250000

In [28]:
```
median_employees_by_industry = df.groupby('Industry')['Revised_number_of_employees'
median_employees_by_industry
```

Out[28]:
```
Industry
Agriculture                          30.0
Communication                        27.0
Construction                         18.0
Consumer discretionary               29.0
Consumer staples                     20.0
E-learning                           15.0
Energy                               20.0
Entertainment                        41.0
Finance                              15.5
Health care                          15.0
Hospitality                          18.0
Industrials                          27.0
Information technology               15.5
Legal services                       15.0
Non-Profit Organization Management   31.0
Other                                15.0
Real estate                          21.0
Retail                               12.0
Tourism                              55.0
Transportation                       15.0
Name: Revised_number_of_employees, dtype: float64
```

In [29]:
```
import pandas as pd

# Group by the "industry" column and calculate the mean for "Revised_number_of_empl
df_sume = df.groupby('Industry')['Amount_in_Euro'].sum()
df_sume
```

Out[29]:
```
Industry
Agriculture                              675.00
Communication                           6943.72
Construction                             108.00
Consumer discretionary                  2064.45
Consumer staples                         495.00
E-learning                                90.00
Energy                                  2526.00
Entertainment                           2903.00
Finance                                 2164.00
Health care                             5346.00
Hospitality                             1281.65
Industrials                            17364.00
Information technology                 10456.00
Legal services                          1913.00
Non-Profit Organization Management        93.00
Other                                  15424.43
Real estate                             3191.80
Retail                                   296.60
Tourism                                  412.50
Transportation                           695.00
Name: Amount_in_Euro, dtype: float64
```

In [30]:
```
import pandas as pd

# Group by the "industry" column and calculate the mean for "Revised_number_of_empl
df_mean_money = df.groupby('Industry')['Amount_in_Euro'].mean()
df_mean_money
```

Out[30]:
```
Industry
Agriculture                            337.500000
Communication                          365.458947
Construction                           108.000000
Consumer discretionary                 344.075000
Consumer staples                       165.000000
E-learning                              90.000000
Energy                                 360.857143
Entertainment                          414.714286
Finance                                270.500000
Health care                            243.000000
Hospitality                            213.608333
Industrials                            403.813953
Information technology                 248.952381
Legal services                         239.125000
Non-Profit Organization Management      93.000000
Other                                  233.703485
Real estate                            245.523077
Retail                                 148.300000
Tourism                                412.500000
Transportation                         173.750000
Name: Amount_in_Euro, dtype: float64
```

In [ ]:
```
```

In [31]:
```
# Combine DataFrames based on the "Industry" column
combined_df = mean_employees_by_industry.merge(median_employees_by_industry, on="Ir
combined_df = combined_df.merge(df_sume, on="Industry")

# Display the combined DataFrame
combined_df
```

Out[31]:

	Industry	mean_employees	Revised_number_of_employees	Amount_in_Euro
0	Agriculture	30.000000	30.0	675.00
1	Communication	37.947368	27.0	6943.72
2	Construction	18.000000	18.0	108.00
3	Consumer discretionary	48.833333	29.0	2064.45
4	Consumer staples	21.666667	20.0	495.00
5	E-learning	15.000000	15.0	90.00
6	Energy	32.857143	20.0	2526.00
7	Entertainment	39.857143	41.0	2903.00
8	Finance	27.000000	15.5	2164.00
9	Health care	22.045455	15.0	5346.00
10	Hospitality	22.666667	18.0	1281.65
11	Industrials	38.976744	27.0	17364.00
12	Information technology	23.833333	15.5	10456.00
13	Legal services	23.500000	15.0	1913.00
14	Non-Profit Organization Management	31.000000	31.0	93.00
15	Other	29.575758	15.0	15424.43
16	Real estate	27.153846	21.0	3191.80
17	Retail	12.000000	12.0	296.60
18	Tourism	55.000000	55.0	412.50
19	Transportation	18.250000	15.0	695.00

In [32]:

```
# Combine DataFrames based on the "Industry" column
combined_df = combined_df.merge(df_mean_money, on="Industry")

# Display the combined DataFrame
combined_df
```

Out[32]:

	Industry	mean_employees	Revised_number_of_employees	Amount_in_Euro_x	Amount_ir
0	Agriculture	30.000000	30.0	675.00	33
1	Communication	37.947368	27.0	6943.72	36
2	Construction	18.000000	18.0	108.00	10
3	Consumer discretionary	48.833333	29.0	2064.45	34
4	Consumer staples	21.666667	20.0	495.00	16
5	E-learning	15.000000	15.0	90.00	9(
6	Energy	32.857143	20.0	2526.00	36(
7	Entertainment	39.857143	41.0	2903.00	41
8	Finance	27.000000	15.5	2164.00	27(
9	Health care	22.045455	15.0	5346.00	24
10	Hospitality	22.666667	18.0	1281.65	21
11	Industrials	38.976744	27.0	17364.00	40
12	Information technology	23.833333	15.5	10456.00	24
13	Legal services	23.500000	15.0	1913.00	23
14	Non-Profit Organization Management	31.000000	31.0	93.00	9
15	Other	29.575758	15.0	15424.43	23
16	Real estate	27.153846	21.0	3191.80	24
17	Retail	12.000000	12.0	296.60	14
18	Tourism	55.000000	55.0	412.50	41
19	Transportation	18.250000	15.0	695.00	17

In [33]:
```
#renaming since it doesnt indentify
#combined_df = combined_df.rename(columns={combined_df.columns[2]: 'Median_number_c
combined_df = combined_df.rename(columns={combined_df.columns[3]: 'Sum_Amount_in_Eu
combined_df = combined_df.rename(columns={combined_df.columns[4]: 'Mean_Amount_in_E
combined_df = combined_df.rename(columns={combined_df.columns[2]: 'Median_number_of
```

In [34]:
```
df_industry_value_counts= df["Industry"].value_counts()
df_industry_value_counts
```

Out[34]:

Industry	
Other	66
Industrials	43
Information technology	42
Health care	22
Communication	19
Real estate	13
Finance	8
Legal services	8
Energy	7
Entertainment	7
Consumer discretionary	6
Hospitality	6
Transportation	4
Consumer staples	3
Agriculture	2
Retail	2
Tourism	1
Construction	1
E-learning	1
Non-Profit Organization Management	1

Name: count, dtype: int64

In [35]:
```
# Combine DataFrames based on the "Industry" column
combined_df = combined_df.merge(df_industry_value_counts, on="Industry")

# Display the combined DataFrame
combined_df
```

Out[35]:

	Industry	mean_employees	Median_number_of_employees	Sum_Amount_in_Euro	Mean_A
0	Agriculture	30.000000	30.0	675.00	
1	Communication	37.947368	27.0	6943.72	
2	Construction	18.000000	18.0	108.00	
3	Consumer discretionary	48.833333	29.0	2064.45	
4	Consumer staples	21.666667	20.0	495.00	
5	E-learning	15.000000	15.0	90.00	
6	Energy	32.857143	20.0	2526.00	
7	Entertainment	39.857143	41.0	2903.00	
8	Finance	27.000000	15.5	2164.00	
9	Health care	22.045455	15.0	5346.00	
10	Hospitality	22.666667	18.0	1281.65	
11	Industrials	38.976744	27.0	17364.00	
12	Information technology	23.833333	15.5	10456.00	
13	Legal services	23.500000	15.0	1913.00	
14	Non-Profit Organization Management	31.000000	31.0	93.00	
15	Other	29.575758	15.0	15424.43	
16	Real estate	27.153846	21.0	3191.80	
17	Retail	12.000000	12.0	296.60	
18	Tourism	55.000000	55.0	412.50	
19	Transportation	18.250000	15.0	695.00	

In [36]:

```
#renaming since it doesnt indentify
#combined_df = combined_df.rename(columns={combined_df.columns[2]: 'Median_number_c
combined_df = combined_df.rename(columns={combined_df.columns[5]: 'Count_industry'}
combined_df[["Industry","Count_industry","mean_employees","Median_number_of_employe
```

Out[36]:

	Industry	Count_industry	mean_employees	Median_number_of_employees	Sum_Amount
0	Agriculture	2	30.000000	30.0	
1	Communication	19	37.947368	27.0	
2	Construction	1	18.000000	18.0	
3	Consumer discretionary	6	48.833333	29.0	
4	Consumer staples	3	21.666667	20.0	
5	E-learning	1	15.000000	15.0	
6	Energy	7	32.857143	20.0	
7	Entertainment	7	39.857143	41.0	
8	Finance	8	27.000000	15.5	
9	Health care	22	22.045455	15.0	
10	Hospitality	6	22.666667	18.0	
11	Industrials	43	38.976744	27.0	
12	Information technology	42	23.833333	15.5	
13	Legal services	8	23.500000	15.0	
14	Non-Profit Organization Management	1	31.000000	31.0	
15	Other	66	29.575758	15.0	
16	Real estate	13	27.153846	21.0	
17	Retail	2	12.000000	12.0	
18	Tourism	1	55.000000	55.0	
19	Transportation	4	18.250000	15.0	

In [37]:

```
# Sort the DataFrame in descending order based on "Mean_Amount_in_Euro"
combined_df = combined_df.sort_values(by="Mean_Amount_in_Euro", ascending=False)

# Display the sorted DataFrame
combined_df
```

Out[37]:

	Industry	mean_employees	Median_number_of_employees	Sum_Amount_in_Euro	Mean_A
7	Entertainment	39.857143	41.0	2903.00	
18	Tourism	55.000000	55.0	412.50	
11	Industrials	38.976744	27.0	17364.00	
1	Communication	37.947368	27.0	6943.72	
6	Energy	32.857143	20.0	2526.00	
3	Consumer discretionary	48.833333	29.0	2064.45	
0	Agriculture	30.000000	30.0	675.00	
8	Finance	27.000000	15.5	2164.00	
12	Information technology	23.833333	15.5	10456.00	
16	Real estate	27.153846	21.0	3191.80	
9	Health care	22.045455	15.0	5346.00	
13	Legal services	23.500000	15.0	1913.00	
15	Other	29.575758	15.0	15424.43	
10	Hospitality	22.666667	18.0	1281.65	
19	Transportation	18.250000	15.0	695.00	
4	Consumer staples	21.666667	20.0	495.00	
17	Retail	12.000000	12.0	296.60	
2	Construction	18.000000	18.0	108.00	
14	Non-Profit Organization Management	31.000000	31.0	93.00	
5	E-learning	15.000000	15.0	90.00	

In [38]:
```
combined_df[["Industry","Count_industry","mean_employees","Median_number_of_employe
```

Out[38]:

	Industry	Count_industry	mean_employees	Median_number_of_employees	Sum_Amount
7	Entertainment	7	39.857143	41.0	
18	Tourism	1	55.000000	55.0	
11	Industrials	43	38.976744	27.0	
1	Communication	19	37.947368	27.0	
6	Energy	7	32.857143	20.0	
3	Consumer discretionary	6	48.833333	29.0	
0	Agriculture	2	30.000000	30.0	
8	Finance	8	27.000000	15.5	
12	Information technology	42	23.833333	15.5	
16	Real estate	13	27.153846	21.0	
9	Health care	22	22.045455	15.0	
13	Legal services	8	23.500000	15.0	
15	Other	66	29.575758	15.0	
10	Hospitality	6	22.666667	18.0	
19	Transportation	4	18.250000	15.0	
4	Consumer staples	3	21.666667	20.0	
17	Retail	2	12.000000	12.0	
2	Construction	1	18.000000	18.0	
14	Non-Profit Organization Management	1	31.000000	31.0	
5	E-learning	1	15.000000	15.0	

In [39]:
```
# relevant Enteirtainment only 7 accounts in this field with the hights avg price,
# tourism, just one account not relevant.
#Industrials third avg position and second main industry after 'Others' and hights
#Information and technology has an average price ordinal, but third biggest field s

#Clients have a 'medium' size company the avg and medain number of employees. It se
```

In [40]:
```
df["Amount_in_Euro"].sum() # total sume matches
```

Out[40]:
```
74443.15
```

In [41]:
```
combined_df.sum()  # total sume matches
```

Out[41]:
```
Industry                    EntertainmentTourism IndustrialsCommunicationE...
mean_employees                                                     575.163457
Median_number_of_employees                                            455.0
Sum_Amount_in_Euro                                                  74443.15
Mean_Amount_in_Euro                                               5111.381605
Count_industry                                                            262
dtype: object
```

In [42]:
```
via_new_df = df.iloc[:, [0,1,3,4,5, 6,15,14,11,13]]
via_new_df
```

Out[42]:

	ID	Amount_in_Euro	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_D
0	1	135.0	15	Inbound	Signup Completed	2022-12 12:45
1	2	412.5	55	Inbound	Signup Completed	2022-12 12:00
2	3	90.0	15	Inbound	Channel inbound	2022-12 16:55
3	4	180.0	15	Inbound	Signup Completed	2022-12 11:10
4	5	108.0	15	Paid	Signup Completed	2022-12 16:43
...	...	...	...	...	...	
257	258	149.0	10	Outbound	LinkedIn	2021-11 12:04
258	259	1200.0	100	Inbound	Inbound Demo	2021-10 12:43
259	260	144.0	16	Inbound	Inbound Paid	2021-09 17:58
260	261	269.0	35	Outbound	LinkedIn	2021-08 17:59
261	262	495.0	55	Outbound	LinkedIn	2021-07 10:36

262 rows × 10 columns

In [43]:
```
### correlation Emplyess vs money
```

In [44]:
```
easy_correlation_df = df.iloc[:, [3,1]]
easy_correlation_df
```

Out[44]:

	Revised_number_of_employees	Amount_in_Euro
0	15	135.0
1	55	412.5
2	15	90.0
3	15	180.0
4	15	108.0
...	...	...
257	10	149.0
258	100	1200.0
259	16	144.0
260	35	269.0
261	55	495.0

262 rows × 2 columns

In [45]:
```
# correlation 9 between money and number of emploeys
correlation_check = df['Revised_number_of_employees'].corr(df['Amount_in_Euro'])
correlation_check
```

Out[45]:
```
0.9243253430086269
```

In [46]:
```
# comapre correlation wihtin industry and heatmap
industry_correlation_df = df.iloc[:, [3,1,14]]
industry_correlation_df
```

Out[46]:

	Revised_number_of_employees	Amount_in_Euro	Industry
0	15	135.0	Other
1	55	412.5	Tourism
2	15	90.0	Consumer staples
3	15	180.0	Other
4	15	108.0	Other
...	...	...	...
257	10	149.0	Health care
258	100	1200.0	Energy
259	16	144.0	Information technology
260	35	269.0	Communication
261	55	495.0	Industrials

262 rows × 3 columns

In [47]:
```
# encoding categoricals into numericals for heatmap
industry_correlation_df["Industry"] = label_encoder.fit_transform(industry_correlat
industry_correlation_df["Industry"]
```

```
C:\Users\Stephan\AppData\Local\Temp\ipykernel_7872\1104235599.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  industry_correlation_df["Industry"] = label_encoder.fit_transform(industry_correlation_df["Industry"])
```

Out[47]:
```
0      15
1      18
2       4
3      15
4      15
       ..
257     9
258     6
259    12
260     1
261    11
Name: Industry, Length: 262, dtype: int32
```

In [48]:
```
industry_correlation_df
```

Out[48]:

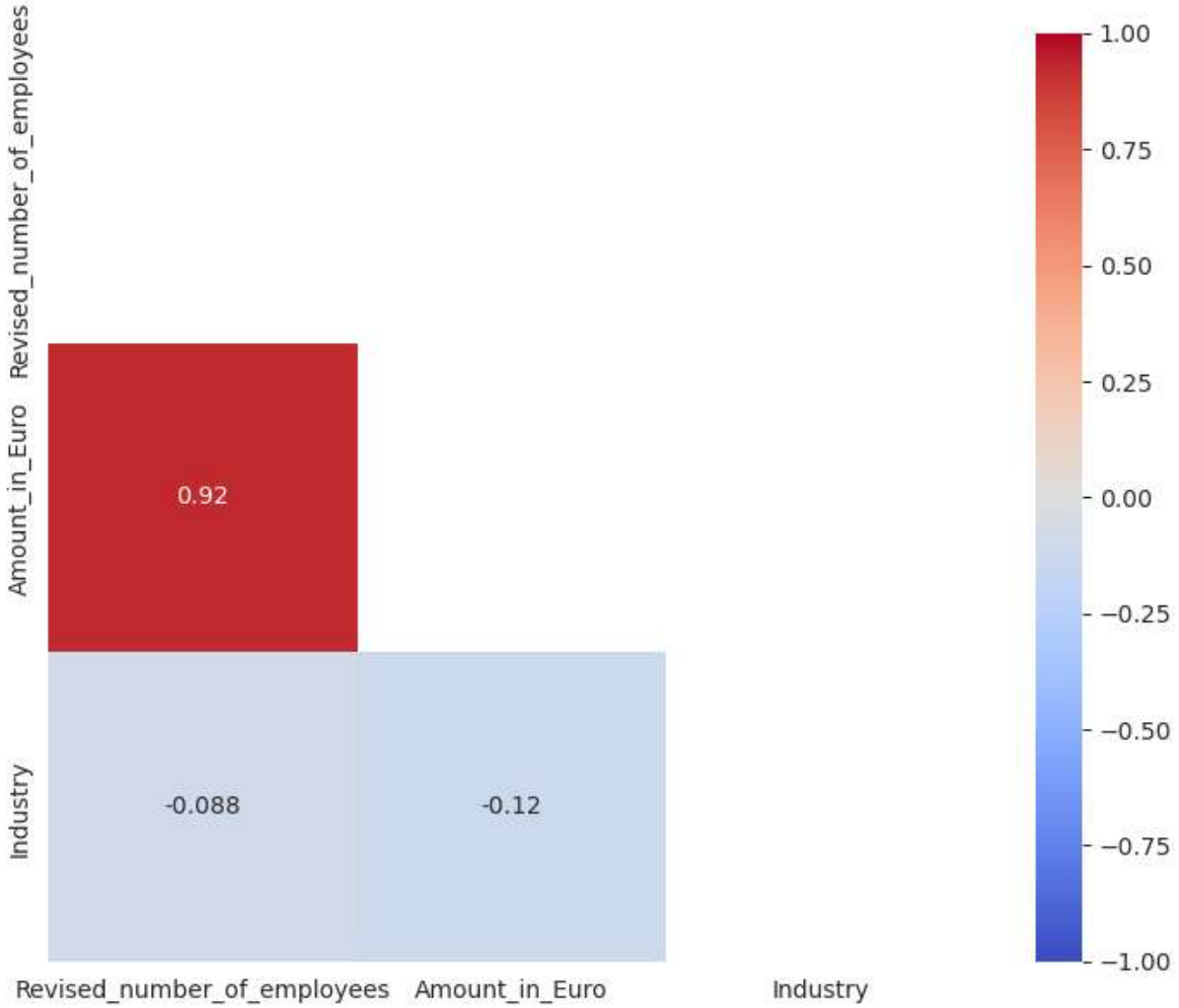
	Revised_number_of_employees	Amount_in_Euro	Industry
0	15	135.0	15
1	55	412.5	18
2	15	90.0	4
3	15	180.0	15
4	15	108.0	15
...	...	...	...
257	10	149.0	9
258	100	1200.0	6
259	16	144.0	12
260	35	269.0	1
261	55	495.0	11

262 rows × 3 columns

In [ ]:
```
# correlation Size & Investment
```

In [49]:
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

corr = industry_correlation_df.corr()
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(9, 7))
    ax = sns.heatmap(corr, mask=mask,cmap='coolwarm', vmin=-1,vmax=1,annot=True, sc
```



In [50]:
```
# correlation betewwn decisionmaking CEO and company size

decision_champion_size_correlation = new_df
decision_champion_size_correlation = decision_champion_size_correlation.drop('Deal_
decision_champion_size_correlation = decision_champion_size_correlation.drop('Creat
decision_champion_size_correlation = decision_champion_size_correlation.drop('Indus
decision_champion_size_correlation = decision_champion_size_correlation.drop('Deal_
decision_champion_size_correlation
```

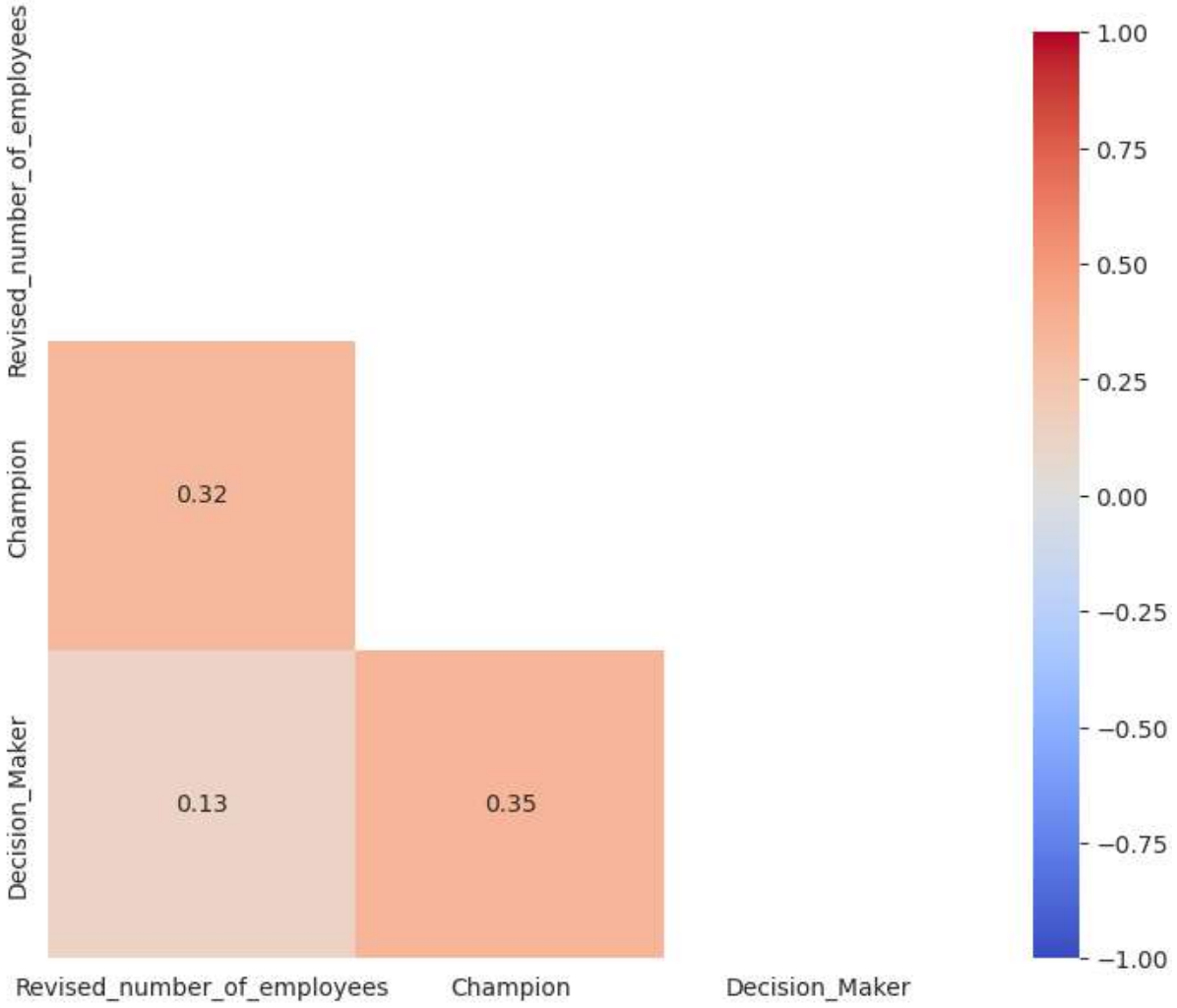
Out[50]:

	Revised_number_of_employees	Champion	Decision_Maker
0	10	3	1
1	44	1	1
2	10	3	3
3	10	1	1
4	10	3	1
...	...	...	...
257	6	3	3
258	55	4	4
259	11	1	1
260	29	3	1
261	44	3	3

262 rows × 3 columns

In [51]:
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

corr = decision_champion_size_correlation.corr()
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(9, 7))
    ax = sns.heatmap(corr, mask=mask,cmap='coolwarm', vmin=-1,vmax=1,annot=True, sc
```



In [134…
```
decision_champion_size_correlation.corr()
```

Out[134]:

	Revised_number_of_employees	Champion	Decision_Maker
Revised_number_of_employees	1.00000	0.321200	0.125450
Champion	0.32120	1.000000	0.345959
Decision_Maker	0.12545	0.345959	1.000000

Correlation experiment, Champion vs Company size = 0.3

I will check the avg, median and mode size per company grouped by champion

---

In [285…
```
### Relevant tech note aside:
### merged_correlation_test_X file at the bottom,
## is saved/run before the otehr dfs, since the Champion was saved a numerical inst
```

original DF reagrding correlations

In [263…
```
correlation_test = decision_champion_size_correlation
correlation_test
```

Out[263]:

	Revised_number_of_employees	Champion	Decision_Maker
0	10	3	1
1	44	1	1
2	10	3	3
3	10	1	1
4	10	3	1
...	...	...	...
257	6	3	3
258	55	4	4
259	11	1	1
260	29	3	1
261	44	3	3

262 rows × 3 columns

In [264…
```
# Calcular la media de empleados por empresa para cada 'champion'
correlation_test_avg= correlation_test
correlation_test_avg = df.groupby('Champion')['Revised_number_of_employees'].mean()
correlation_test_avg.columns = ['Champion', 'Avg_employees']
```

In [265…
```
merged_correlation_test_avg= correlation_test_avg
correlation_test_avg
```

Out[265]:

	Champion	Avg_employees
0	Administration	26.000000
1	CEO	22.791667
2	CFO	15.000000
3	HR Director	37.393443
4	HR team	39.081081
5	IT	25.000000
6	Office Manager	23.000000
7	Project Manager	83.750000

In [246…
```
# transform column Champion into numerical again
correlation_test_avg["Champion"] = label_encoder.fit_transform(correlation_test_avg
```

In [247…
```
correlation_test_avg
```

Out[247]:

	Champion	Avg_employees
0	0	26.000000
1	1	22.791667
2	2	15.000000
3	3	37.393443
4	4	39.081081
5	5	25.000000
6	6	23.000000
7	7	83.750000

In [248…

```
# Calcular la correlación entre la nueva columna 'moda_empleados' y 'champion'
correlation_test_avg = correlation_test_avg['Champion'].corr(correlation_test_avg['
correlation_test_avg
```

Out[248]:

```
0.5900151593316366
```

median

In [275…

```
correlation_test_median = correlation_test

# Calcular la mediana de empleados por empresa para cada 'champion'
correlation_test_median = df.groupby('Champion')['Revised_number_of_employees'].med
correlation_test_median.columns = ['Champion', 'Median_of_employees']

#for future merged
merged_correlation_test_median= correlation_test_median

correlation_test_median
```

Out[275]:

	Champion	Median_of_employees
0	Administration	24.5
1	CEO	15.0
2	CFO	15.0
3	HR Director	26.0
4	HR team	26.0
5	IT	16.0
6	Office Manager	23.0
7	Project Manager	35.0

In [250…

```
# transform column Champion into numerical again
correlation_test_median["Champion"] = label_encoder.fit_transform(correlation_test_
correlation_test_median
```

Out[250]:

	Champion	Median_of_employees
0	0	24.5
1	1	15.0
2	2	15.0
3	3	26.0
4	4	26.0
5	5	16.0
6	6	23.0
7	7	35.0

In [251…
```
# Calcular la correlación entre la nueva columna 'moda_empleados' y 'champion'
correlation_test_median = correlation_test_median['Champion'].corr(correlation_test
correlation_test_median
```

Out[251]:
```
0.48765370233385397
```

mode

In [252…
```
correlation_test_mode= correlation_test
```

In [260…
```
# Calcular la moda de empleados por empresa para cada 'champion'
correlation_test_mode = df.groupby('Champion')['Revised_number_of_employees'].apply
correlation_test_mode.columns = ['Champion', 'Mode_employees']

#for future mergedf
merged_correlation_test_mode= correlation_test_mode
correlation_test_mode
```

Out[260]:

	Champion	Mode_employees
0	Administration	20
1	CEO	15
2	CFO	15
3	HR Director	15
4	HR team	15
5	IT	15
6	Office Manager	23
7	Project Manager	15

In [254…
```
# transform column Champion into numerical again
correlation_test_mode["Champion"] = label_encoder.fit_transform(correlation_test_mo
correlation_test_mode
```

Out[254]:

	Champion	Mode_employees
0	0	20
1	1	15
2	2	15
3	3	15
4	4	15
5	5	15
6	6	23
7	7	15

In [255…

```
# Calcular la correlación entre la nueva columna 'moda_empleados' y 'champion'
correlation_test_mode = correlation_test_mode['Champion'].corr(correlation_test_mod
correlation_test_mode
```

Out[255]:

```
0.04682313004619228
```

In [ ]:

correlations merged

In [279…

```
# Combinar los DataFrames en función de la columna 'champion'
merged_test_df = pd.merge(merged_correlation_test_mode, merged_correlation_test_mec
merged_test_df = pd.merge(merged_test_df, merged_correlation_test_avg, on='Champior
merged_test_df
```

Out[279]:

	Champion	Mode_employees	Median_of_employees	Avg_employees
0	Administration	20	24.5	26.000000
1	CEO	15	15.0	22.791667
2	CFO	15	15.0	15.000000
3	HR Director	15	26.0	37.393443
4	HR team	15	26.0	39.081081
5	IT	15	16.0	25.000000
6	Office Manager	23	23.0	23.000000
7	Project Manager	15	35.0	83.750000

In [282…

```
saved_test_version_df = merged_test_df
saved_test_version_df
```

Out[282]:

	Champion	Mode_employees	Median_of_employees	Avg_employees
0	Administration	20	24.5	26.000000
1	CEO	15	15.0	22.791667
2	CFO	15	15.0	15.000000
3	HR Director	15	26.0	37.393443
4	HR team	15	26.0	39.081081
5	IT	15	16.0	25.000000
6	Office Manager	23	23.0	23.000000
7	Project Manager	15	35.0	83.750000

Global company size correlation/ Champion Vs Media, Mode & Mean/Champion

In [ ]:

In [286…

```
# transform column Champion into numerical again
merged_test_df["Champion"] = label_encoder.fit_transform(merged_test_df["Champion"]
merged_test_df
```

Out[286]:

	Champion	Mode_employees	Median_of_employees	Avg_employees
0	0	20	24.5	26.000000
1	1	15	15.0	22.791667
2	2	15	15.0	15.000000
3	3	15	26.0	37.393443
4	4	15	26.0	39.081081
5	5	15	16.0	25.000000
6	6	23	23.0	23.000000
7	7	15	35.0	83.750000

In [287…

```
merged_test_df.corr()
```

Out[287]:

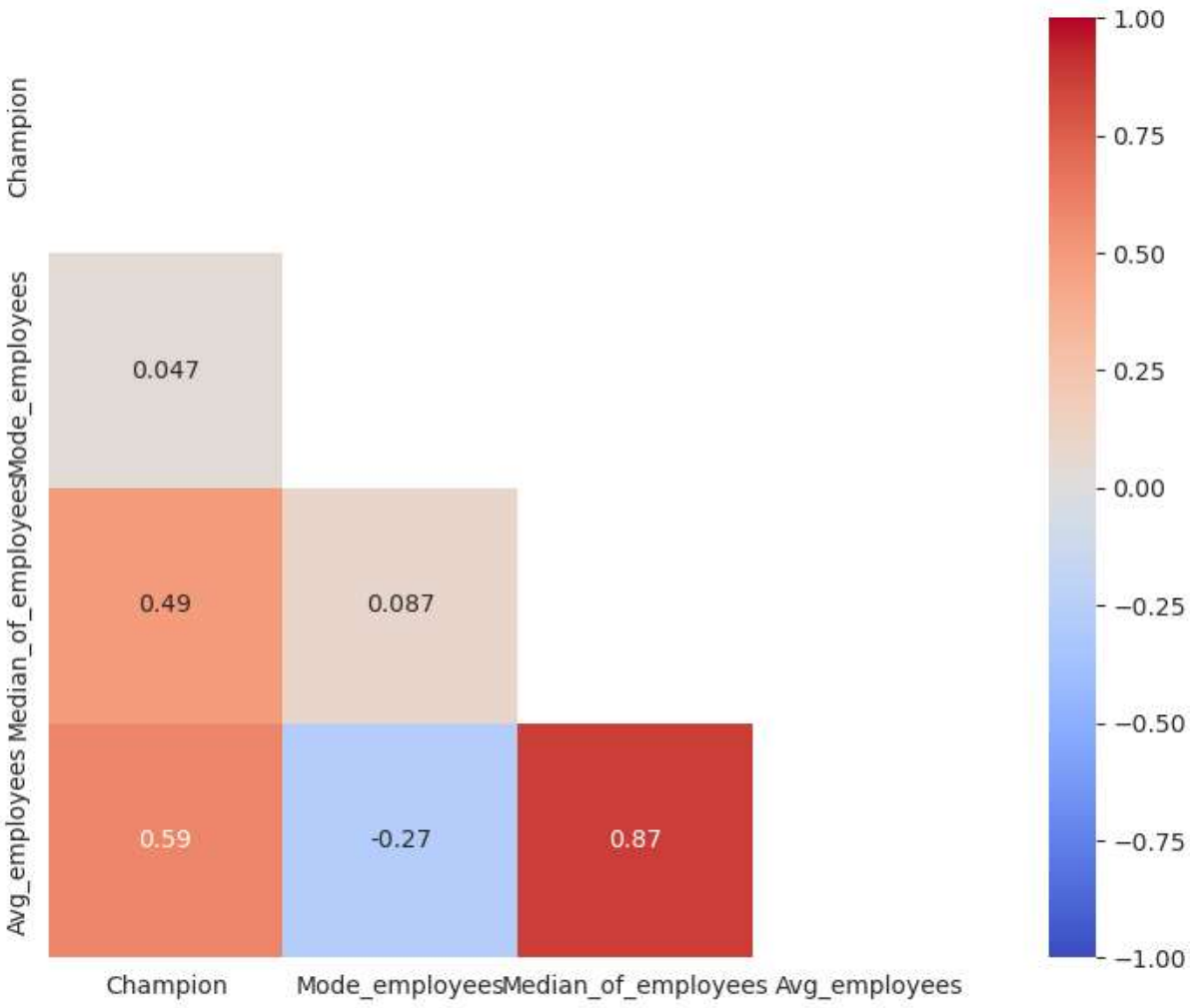
	Champion	Mode_employees	Median_of_employees	Avg_employees
Champion	1.000000	0.046823	0.487654	0.590015
Mode_employees	0.046823	1.000000	0.086846	-0.271958
Median_of_employees	0.487654	0.086846	1.000000	0.868889
Avg_employees	0.590015	-0.271958	0.868889	1.000000

In [288…

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

corr = merged_test_df.corr()
mask = np.zeros_like(corr)
```

```
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(9, 7))
    ax = sns.heatmap(corr, mask=mask,cmap='coolwarm', vmin=-1,vmax=1,annot=True, sc
```



In summary, there is a weak positive correlation between 'Champion' and company size, and the strength of the correlation varies from very weak to considerable depending on the specific company size metric (mode, median, or average) you are considering.

In [ ]:
```
# exporint file for excel
```

In [289…
```
# Specify the file path for the Excel file
excel_file_path = 'output_file.xlsx'

# Export the DataFrame to Excel
saved_test_version_df.to_excel(excel_file_path, index=False)
```

I grouped CEO & CFO vs 'Smaller position'

In [293…
```
import pandas as pd

# Creating the DataFrame
data = {'Champion': [1, 2, 2, 1, 1, 1, 1, 1],
        'Avg_employees': [26, 22.79166667, 15, 37.39344262, 39.08108108, 25, 23, 83

Bigvsothers_df = pd.DataFrame(data)
```

```
# Display the DataFrame
Bigvsothers_df
```

Out[293]:

	Champion	Avg_employees
0	1	26.000000
1	2	22.791667
2	2	15.000000
3	1	37.393443
4	1	39.081081
5	1	25.000000
6	1	23.000000
7	1	83.750000

In [295…

```
# Calcular la correlación entre la nueva columna 'moda_empleados' y 'champion'
Bigvsothers_df = Bigvsothers_df['Champion'].corr(Bigvsothers_df['Avg_employees'])
Bigvsothers_df

# Some tendecy
```

Out[295]:

```
-0.43171413014507537
```

TIME (month) AND INDUSTRIAL SECTOR

In [300…

```
inudstry_date_df = new_df
inudstry_date_df
```

Out[300]:

	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_Date	Industry	Cl
0	15	Inbound	Signup Completed	2022-12-30 12:45:00	Other	
1	55	Inbound	Signup Completed	2022-12-30 12:00:00	Tourism	
2	15	Inbound	Channel inbound	2022-12-22 16:55:00	Consumer staples	
3	15	Inbound	Signup Completed	2022-12-21 11:10:00	Other	
4	15	Paid	Signup Completed	2022-12-19 16:43:00	Other	
...	...	...	...	...	...	
257	10	Outbound	LinkedIn	2021-11-18 12:04:00	Health care	
258	100	Inbound	Inbound Demo	2021-10-07 12:43:00	Energy	
259	16	Inbound	Inbound Paid	2021-09-30 17:58:00	Information technology	
260	35	Outbound	LinkedIn	2021-08-04 17:59:00	Communication	
261	55	Outbound	LinkedIn	2021-07-28 10:36:00	Industrials	

262 rows × 7 columns

In [303…

```
# Convert the 'datetime_column' to a datetime object
inudstry_date_df['Create_Date'] = pd.to_datetime(inudstry_date_df['Create_Date'])

# Extract the month and create a new column 'month'
inudstry_date_df['month'] = inudstry_date_df['Create_Date'].dt.month

# Display the DataFrame with the extracted month
inudstry_date_df[['Create_Date', 'month']]
```

```
C:\Users\Stephan\AppData\Local\Temp\ipykernel_7872\2228689102.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  inudstry_date_df['Create_Date'] = pd.to_datetime(inudstry_date_df['Create_Date'])
C:\Users\Stephan\AppData\Local\Temp\ipykernel_7872\2228689102.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  inudstry_date_df['month'] = inudstry_date_df['Create_Date'].dt.month
```

Out[303]:

	Create_Date	month
0	2022-12-30 12:45:00	12
1	2022-12-30 12:00:00	12
2	2022-12-22 16:55:00	12
3	2022-12-21 11:10:00	12
4	2022-12-19 16:43:00	12
...	...	...
257	2021-11-18 12:04:00	11
258	2021-10-07 12:43:00	10
259	2021-09-30 17:58:00	9
260	2021-08-04 17:59:00	8
261	2021-07-28 10:36:00	7

262 rows × 2 columns

In [304…

```
inudstry_date_df
```

Out[304]:

	Revised_number_of_employees	Deal_Provenance	Deal_Origin	Create_Date	Industry	Cl
0	15	Inbound	Signup Completed	2022-12-30 12:45:00	Other	
1	55	Inbound	Signup Completed	2022-12-30 12:00:00	Tourism	
2	15	Inbound	Channel inbound	2022-12-22 16:55:00	Consumer staples	
3	15	Inbound	Signup Completed	2022-12-21 11:10:00	Other	
4	15	Paid	Signup Completed	2022-12-19 16:43:00	Other	
...	...	...	...	...	...	
257	10	Outbound	LinkedIn	2021-11-18 12:04:00	Health care	
258	100	Inbound	Inbound Demo	2021-10-07 12:43:00	Energy	
259	16	Inbound	Inbound Paid	2021-09-30 17:58:00	Information technology	
260	35	Outbound	LinkedIn	2021-08-04 17:59:00	Communication	
261	55	Outbound	LinkedIn	2021-07-28 10:36:00	Industrials	

262 rows × 8 columns

In [305…

```
# Group by 'industry' and 'month', then count the number of accounts
inudstry_date_df = inudstry_date_df.groupby(['Industry', 'month']).size().reset_inc
```

```
inudstry_date_df
```

Out[305]:

	Industry	month	number_of_accounts
0	Agriculture	9	1
1	Agriculture	10	1
2	Communication	2	1
3	Communication	3	1
4	Communication	4	2
...	...	...	...
105	Tourism	12	1
106	Transportation	7	1
107	Transportation	9	1
108	Transportation	10	1
109	Transportation	12	1

110 rows × 3 columns

In [308...

```
inudstry_date_df
```

Out[308]:

	Industry	month	number_of_accounts
0	Agriculture	9	1
1	Agriculture	10	1
2	Communication	2	1
3	Communication	3	1
4	Communication	4	2
...	...	...	...
105	Tourism	12	1
106	Transportation	7	1
107	Transportation	9	1
108	Transportation	10	1
109	Transportation	12	1

110 rows × 3 columns

In [309...

```
#renaming since it doesnt indentify
inudstry_date_df = inudstry_date_df.rename(columns={inudstry_date_df.columns[0]: '
```

In [312...

```
inudstry_date_df["Industry"]
```

Out[312]:
```
0          Agriculture
1          Agriculture
2        Communication
3        Communication
4        Communication
             ...
105          Tourism
106    Transportation
107    Transportation
108    Transportation
109    Transportation
Name: Industry, Length: 110, dtype: object
```

In [313…
```
# turning into numerical
inudstry_date_df["Industry"] = label_encoder.fit_transform(inudstry_date_df["Indust
inudstry_date_df["Industry"]
```

Out[313]:
```
0       0
1       0
2       1
3       1
4       1
       ..
105    18
106    19
107    19
108    19
109    19
Name: Industry, Length: 110, dtype: int32
```

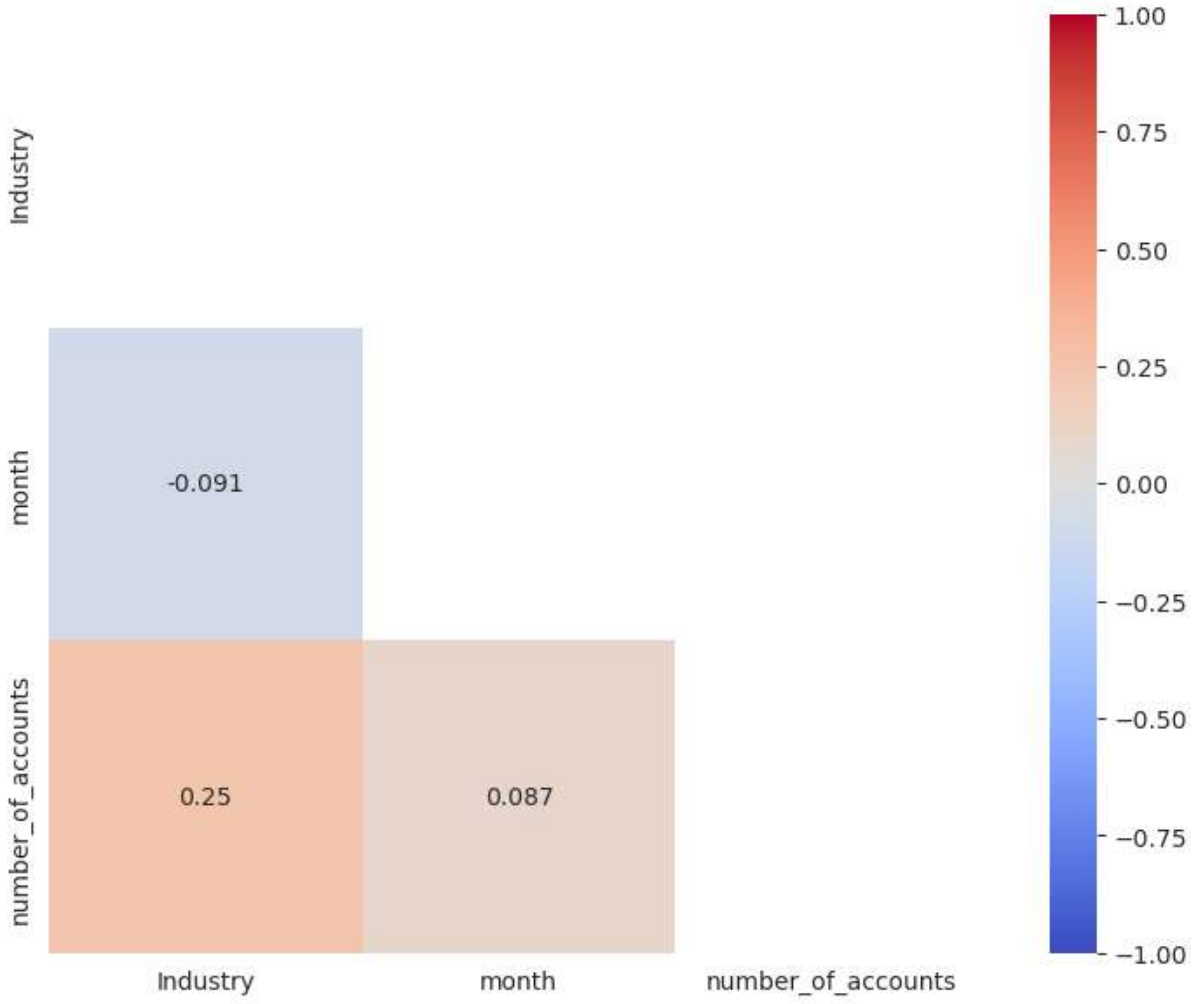
In [314…
```
inudstry_date_df.corr()
```

Out[314]:

	Industry	month	number_of_accounts
Industry	1.000000	-0.091183	0.246298
month	-0.091183	1.000000	0.087044
number_of_accounts	0.246298	0.087044	1.000000

In [315…
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

corr = inudstry_date_df.corr()
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(9, 7))
    ax = sns.heatmap(corr, mask=mask,cmap='coolwarm', vmin=-1,vmax=1,annot=True, sc
```


Industry
month
number_of_accounts
-0.091
0.25
0.087
Industry
month
number_of_accounts
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

In [ ]: