LinkedIn Data Analysis

Author: AKILESH S

As an active user on LinkedIn with more than 1000 connections, I was curious about the statistics of my network. In this project, I utilized exploratory analysis and data visualizations to gain insights from my own LinkedIn data.

Data Preparation

First, let's import the necessary libraries for this project:

```
# Import the libraries
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
```

Next, we can load the data that is already downloaded as a .csv file. To download your own data, you can go

```
# Load the data
df = pd.read_csv("connections.csv")
df.head(10)
```

	First Name	Last Name	Email Address	Company	Position	Connected On
0	Mohan venkata	sai	NaN	NaN	NaN	13-Mar-23
1	Yeswanth	Penmetsa	NaN	NaN	NaN	13-Mar-23
2	Kunal	Raj	NaN	NaN	NaN	12-Mar-23
3	Gohula	Krishnan	NaN	NaN	NaN	12-Mar-23
4	Jaya	Kumar	NaN	NaN	NaN	12-Mar-23
5	CHANDAN KUMAR	TRIVEDI	NaN	NaN	NaN	12-Mar-23
6	HARRSAVARTHINI	K	NaN	NaN	NaN	11-Mar-23
7	Ankita	Kumari	NaN	NaN	NaN	11-Mar-23
8	Shalu	Kumari	NaN	NaN	NaN	11-Mar-23

The DataFrame above displays only my 10 latest connections on LinkedIn. The Connected On column indicates the date that I connect to that person.

```
# Describe the data
df.describe()
```

	First Name	Last Name	Email Address	Company	Position	Connected On
count	1085	1085	11	754	754	1092
unique	960	764	11	516	495	243
top	Arun	S	9921004758@klu.ac.in	Cognizant	Associate Software Engineer	13-Dec-21
freq	7	39	1	28	28	60

Date Connected

Let's take a closer look on the Connected On column. But before that, we need to convert that column into a datetime format .

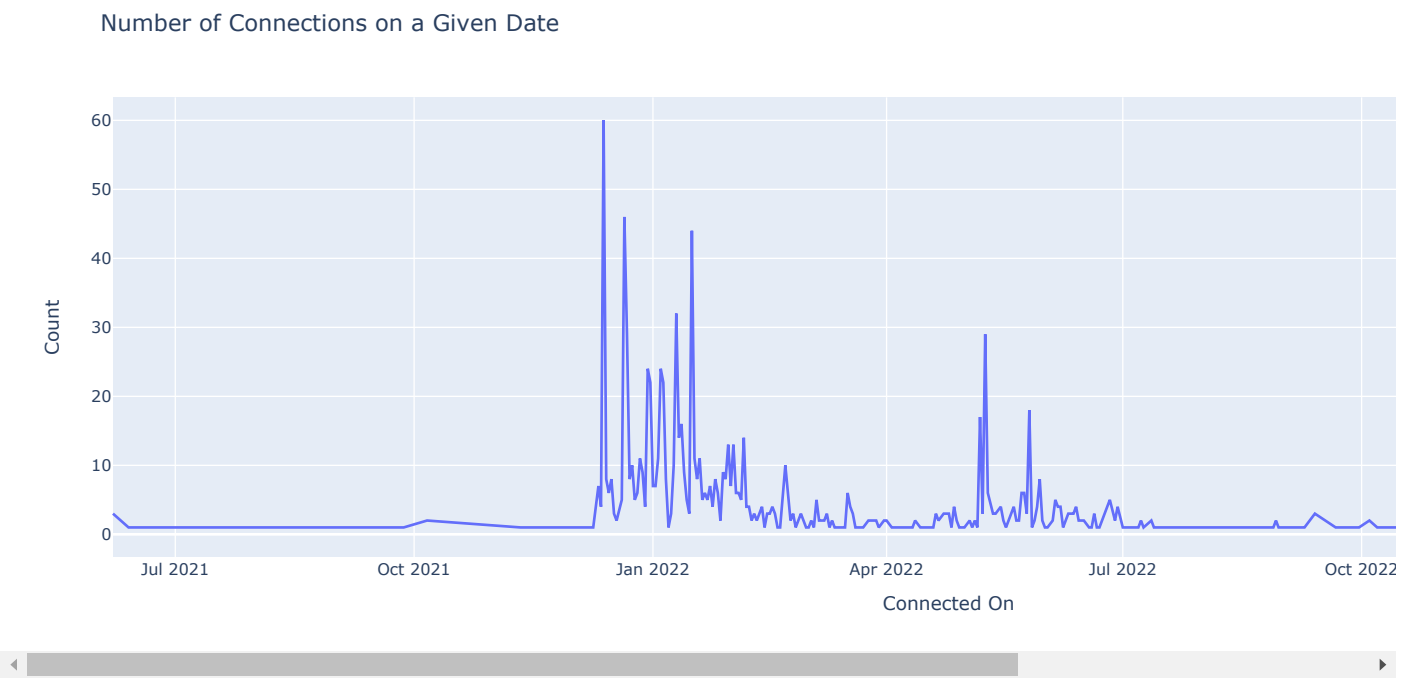
```
# Convert the 'Connected On' column to datetime format
df["Connected On"] = pd.to_datetime(df["Connected On"])
df["Connected On"]
```

```
0      2023-03-13
1      2023-03-13
2      2023-03-12
3      2023-03-12
4      2023-03-12
```

```
        ...
1087   2021-07-14
1088   2021-06-13
1089   2021-06-07
1090   2021-06-07
1091   2021-06-07
Name: Connected On, Length: 1092, dtype: datetime64[ns]
```

Now, we can visualize the number of connections on a given date using Plotly's line plot.

```
# Create a line plot to visualize the number of connections on a given date
fig1 = px.line(df.groupby(by="Connected On").count().reset_index(),x="Connected On",y="First Name",labels={"First Name": "Count"},title="Numb
fig1.show()
```



From the line plot above, we can see that there is a peak in the number of connections per day on 13 December 2021. It also seems that December 2021 is the period when I was the most active on LinkedIn.

Company

Which companies/organizations do the people in my network mainly come from?

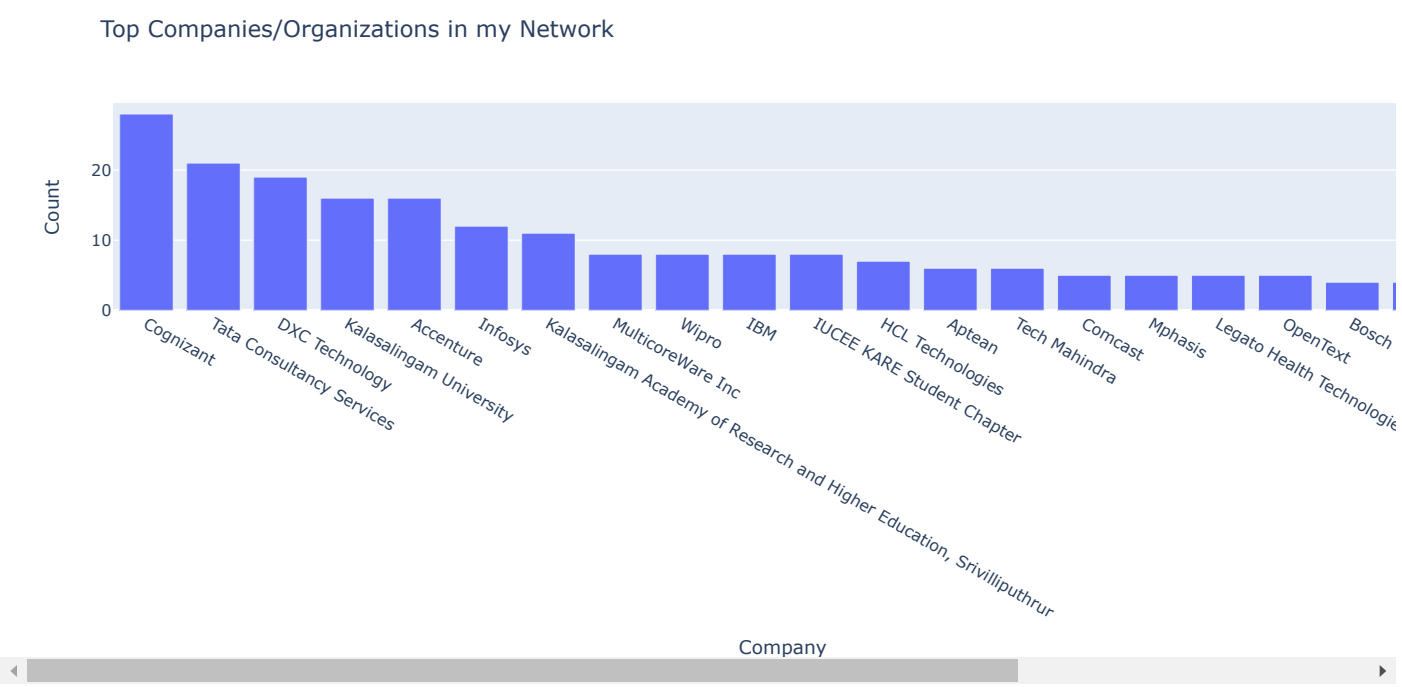
To answer that question, we need to first group and sort the data based on the companies

```
# Group and sort the data by company
df_by_company = df.groupby(by="Company").count().reset_index().sort_values(by="First Name", ascending=False).reset_index(drop=True)
df_by_company
```

	Company	First Name	Last Name	Email Address	Position	Connected On
0	Cognizant	28	28	0	28	28

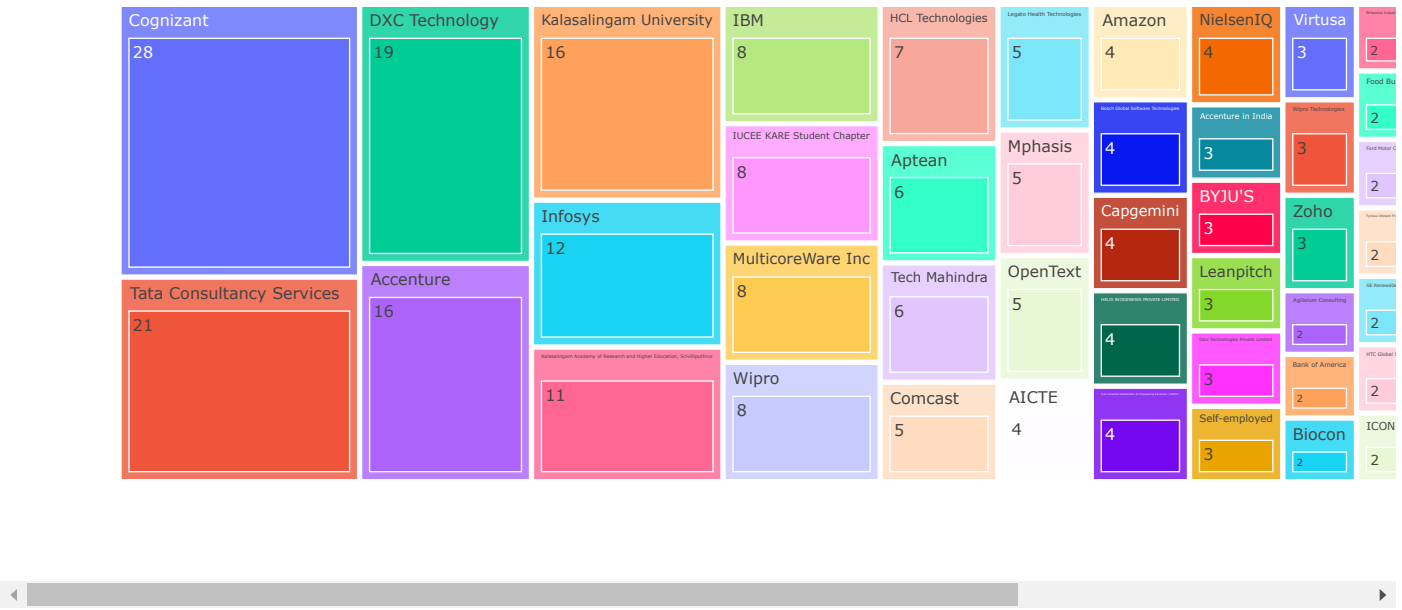
Now that we have our data grouped and sorted based on the companies, we can visualize it using Plotly's bar plot

```
# Create a bar plot for the top companies
fig2 = px.bar(df_by_company[:20],x="Company",y="First Name",labels={"First Name": "Count"},title="Top Companies/Organizations in my Network")
fig2.show()
```



It worked just fine, but perhaps Plotly's treemap will do a better job in visualizing the companies in this case.

```
# Create a treemap for the top companies
fig3 = px.treemap(df_by_company[:100], path=["Company", "Position"],values="First Name",labels={"First Name": "Count"})
fig3.show()
```



Using the treemap above, it is easier to compare the proportion of one company/organization to the others. It looks like the largest proportion of my network is from my university.

Position

What are the top common positions of people in my network?

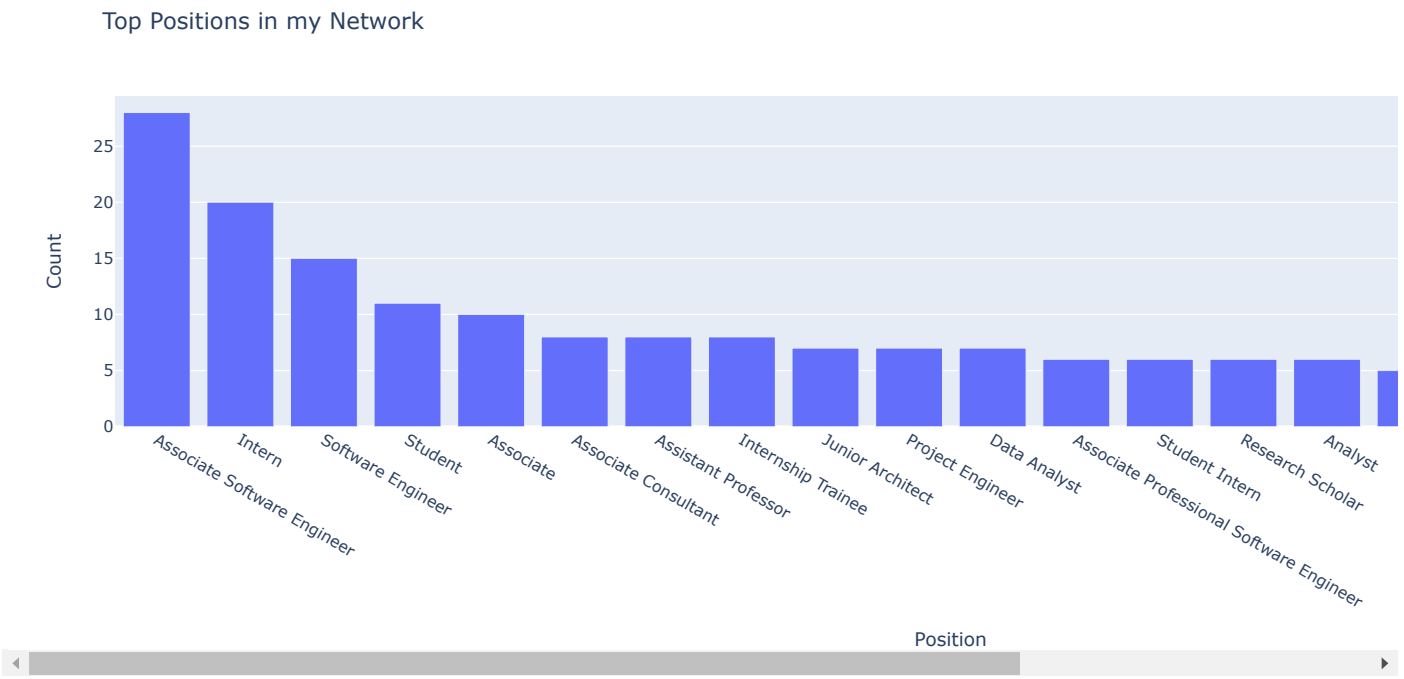
To answer that question, we can create similar visualizations for the Position column

```
# Group and sort the data by position
df_by_position = df.groupby(by="Position").count().reset_index().sort_values(by="First Name", ascending=False).reset_index(drop=True)
df_by_position
```

	Position	First Name	Last Name	Email Address	Company	Connected On
0	Associate Software Engineer	28	28	0	28	28
1	Intern	20	20	1	20	20
2	Software Engineer	15	15	0	15	15
3	Student	11	11	0	11	11
4	Associate	10	10	0	10	10
...	...	...	...	...	...	...
490	Full Stack Engineer	1	1	0	1	1
491	Frontend Web Developer	1	1	0	1	1
492	Front end Application Developer (Consultant)	1	1	0	1	1
493	Freelance Graphic Designer	1	1	0	1	1
494	worked as Analytical executive	1	1	0	1	1

495 rows × 6 columns

```
# Create a bar plot for the top positions
fig4 = px.bar(df_by_position[:20],x="Position",y="First Name",labels={"First Name": "Count"},title="Top Positions in my Network")
fig4.show()
```




```
# Create a treemap for the top positions
fig5 = px.treemap(df_by_position[:100], path=["Position", "Company"],values="First Name",labels={"First Name": "Count"})
fig5.show()
```

The top position in my network is Associative Software Engineer, It is great to know that the top common positions in my network are my target group for networking.

```
# Find all positions that contains 'Data Analysts'
df["Position"].str.contains("Data Analyst").sum()
```

```
8
```

Wow, I didn't expect to see that many data Analysts in my network!

It is always fun and interesting to analyze your own data as you might be surprised by what you see and learned something helpful. Personally, these treemaps made me realize that my LinkedIn network is much more diverse than I had thought.