[1]Universidad Sergio Arboleda

Análisis de datos a partir de información encontrada para cada base de datos

La primera entrega de datos, con el archivo **‘Colombia-Feb21.xlsx’**, el cual es un archivo Excel, nuestro primer objetivo es identificar nuestra data y saber cuáles es su formato, cuantas variables categóricas y numéricas tenemos como son sus distribuciones.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 15 columns):
 #   Column                                  Non-Null Count  Dtype  
---  ------                                  --------------  -----  
 0   Organization Name                       1000 non-null   object 
 1   Organization Name URL                   1000 non-null   object 
 2   Industries                              980 non-null    object 
 3   Headquarters Location                   1000 non-null   object 
 4   Description                             1000 non-null   object 
 5   CB Rank (Company)                       999 non-null    object 
 6   Founded Date                            894 non-null    object 
 7   Founded Date Precision                  894 non-null    object 
 8   Last Funding Date                       367 non-null    object 
 9   Last Funding Amount                     227 non-null    object 
 10  Last Funding Amount Currency            227 non-null    object 
 11  Last Funding Amount Currency (in USD)   227 non-null    float64
 12  Last Equity Funding Amount              204 non-null    object 
 13  Last Equity Funding Amount Currency     204 non-null    object 
 14  Last Equity Funding Amount Currency (in USD)  203 non-null    float64
dtypes: float64(2), object(13)
memory usage: 117.3+ KB
```

Encontramos:

- 15 variables
- 1000 Entradas
- 2 variables numéricas tipo float64
- 12 variables categóricas tipo object
- Encontramos información de nombres de empresas, en que industria trabaja cada una de estas compañías, cuales fueron las fechas de inicio y de finalización de cada monto invertido.

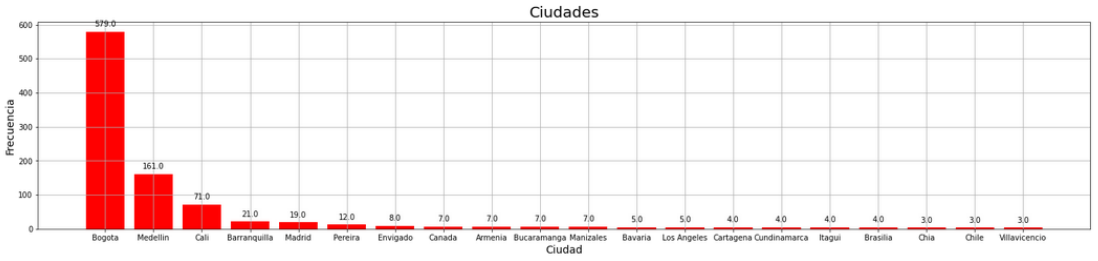
[1] Bitacora-Mineria de datos

Funcion para crear graficos de barras, como funciona:

- x es un strin
- y es la frecuencia con la que encuentras cada entrada de x
- ax
- title el titulo de la grafica
- x_label nombre del eje x
- y_label nombre del eje y

In [15]:
```python
def bar_plot(x, y, ax, title, x_label, y_label, ymax):

    ax.bar(x,y, color = 'red')
    ax.set_title(title, fontsize=20)
    ax.set_xlabel(x_label, fontsize=14)
    ax.set_ylabel(y_label, fontsize=14)
    ax.grid(True)

    for rect in ax.patches:
        # Get X and Y placement of label from rect.
        Y = rect.get_height()
        X = rect.get_x() + rect.get_width() / 2

        # Number of points between bar and label. Change to your liking.
        space = 5
        # Vertical alignment for positive values
        va = 'bottom'

        # If value of bar is negative: Place label below bar
        if Y < 0:
            # Invert space to place label below
            space *= -1
            # Vertically align label at top
            va = 'top'

        # Use Y value as label and format number with one decimal place
        label = "{:.1f}".format(Y)

        # Create annotation
        ax.annotate(
            label,
            (X, Y),
            xytext=(0, space),
            textcoords="offset points",
            ha='center',
            va=va)
```

In [16]:
```python
fig,ax = plt.subplots(1,1, figsize=(25, 5))

a_graficar = 'Ciudad'
x=df[a_graficar].value_counts(dropna=True).keys()
y=df[a_graficar].value_counts(dropna=True).to_list()
bar_plot(x[:20], y[:20], ax, title='Ciudades', x_label=a_graficar, y_label = 'Frecuencia', ymax=700)
```



Análisis de datos para 11 tablas con 11 países de latinoamérica, el objetivo principal de este análisis es reconocer y eliminar NaN.

Función para detectar el porcentaje de NaN por cada columna para los 11 archivos.
Entradas:

- datos, los data frame que se quieren analizar
- **per nulls** cuanto es lo mínimo de porcentaje que es tolerable para el análisis.

para el análisis de estos datos notamos que lo mínimo de consideración es el 50%, pero al nivel de levantamiento de los datos el negocio dejó claro que para ellos lo mínimo es el 70% de los NaN

```
In [3]: df_ar['Number of Events'].isna().sum()/len(df_ar['Number of Events']) # porcentaje de datos nulos en la columna
Out[3]: 0.917
```

```
In [4]: def cols_90per_nulls(data):
            count = 0
            cols_to_drop = {}
            for col in data.columns:
                per_nulls = data[col].isna().sum()/len(data[col])
                if per_nulls >= 0.5:
                    cols_to_drop[col] = per_nulls
                    # print(col, per_nulls)
                    count+=1
                else:
                    None

            print('Number of cols with >50% nulls:', count)
            return cols_to_drop
```

¿Cual es la salida?

nos entrega un análisis por variable con la cantidad en valor del porcentaje del total de NaN.

Out[7]:

	Arg	Bra	Chi	Col	Ger	Isr	Mex	Spa	Swi	Uru	Usa
Estimated Revenue Range	0.543	NaN	0.621	0.603	NaN	NaN	0.604	NaN	NaN	0.559846	NaN
Exit Date	0.900	0.912	0.940	0.938	0.832	0.810	0.906	0.894	0.880	0.930502	0.706
Exit Date Precision	0.900	0.912	0.940	0.938	0.832	0.810	0.906	0.894	0.880	0.930502	0.706
Closed Date	0.985	0.989	0.984	0.989	0.997	0.992	0.992	0.994	0.993	0.992278	0.992
Closed Date Precision	0.971	0.980	0.965	0.978	0.996	0.991	0.980	0.988	0.992	0.984556	0.992
...	...	...	...	...	...	...	...	...	...	...	...
Aberdeen - IT Spend Currency (in USD)	1.000	1.000	1.000	1.000	0.727	1.000	1.000	0.836	0.847	1.000000	NaN
School Method	1.000	0.998	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000000	1.000
Number of Founders	NaN	NaN	NaN	0.564	NaN	NaN	NaN	NaN	NaN	0.621622	NaN
Founders	NaN	NaN	NaN	0.564	NaN	NaN	NaN	NaN	NaN	0.621622	NaN
Headquarters Regions	NaN	NaN	NaN	NaN	NaN	1.000	NaN	NaN	1.000	NaN	NaN

Función para corregir espacios, que recibe:

- la palabra para borrar el espacio en blanco

su salida, es una palabra sin espacios.

In [24]:
```
#Funcion que corrige espacios
def correct_word(word):

    new_word = word.split()[0]
    return new_word
```