Exploration Data Analysis

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

Importiamo le librerie:

> pandas: dataframe e series numpy: lavorare con gli array matplotlib & seaborn: visualizzazione del dato

Da una vista in SQL abbiamo creato una nuova tabella con i dati che occorrono per fare esplorazione del dato

**Lettura del file .csv

In [20]:
```
df = pd.read_csv("exploration.csv",delimiter=';',header=0)
```

Con la funzione **head()**, visualizziamo i **primi 5 record del dataset**

In [21]:
```
df.head()
```

Out[21]:

	GSP0001	F	14	2023	ETNA	AQUILA	UCCELLI	In Ottima Salute
0	GSP0002	M	2	2023	ETNA	AQUILA	UCCELLI	In Buona Salute
1	GSP0003	M	3	2023	ADDA NORD	AQUILA	UCCELLI	In Ottima Salute
2	GSP0004	F	2	2023	PREALPI GIULIE	AQUILA	UCCELLI	In Ottima Salute
3	GSP0005	F	14	2023	ALPI APUANE	AQUILA	UCCELLI	In Buona Salute
4	GSP0006	M	16	2023	ALPI LIGURI	AQUILA	UCCELLI	In Buona Salute

!!**ATTENZIONE!!** Da una prima analisi, le colonne sono sprovviste di un header. Provvediamo a dare un nome della colonne!

In [22]:
```
df.columns=['IDAnimale','Sesso','Età','AnnoNascita','NomeParco','SpecieAnimale','OrdineAnimale','StatoSalute']
```

Controlliamo se l'header è stato inserito. Con la funzione **tail()**, visualizziamo gli **ultimi 5 record del dataset**

In [24]:
```
df.tail()
```

Out[24]:

	IDAnimale	Sesso	Età	AnnoNascita	NomeParco	SpecieAnimale	OrdineAnimale	StatoSalute
1494	GSP1496	M	1	1966	MONTI SIMBRUINI	VOLPE	MAMMIFERI	In Buona Salute
1495	GSP1497	F	3	1959	PANEVEGGIO PALE DI S. MARTINO	VOLPE	MAMMIFERI	In Buona Salute
1496	GSP1498	M	4	1950	ADAMELLO	VOLPE	MAMMIFERI	In Cura
1497	GSP1499	F	3	1949	MONTI SIMBRUINI	VOLPE	MAMMIFERI	In Salute
1498	GSP1500	F	1	1936	NEBRODI	VOLPE	MAMMIFERI	Pessima Salute

Per avere un quadro generale dei data type, del numero di record per colonne e la presenza di Null, utilizziamo la funzione: **info()**

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1499 entries, 0 to 1498
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   IDAnimale       1499 non-null   object
 1   Sesso           1499 non-null   object
 2   Età             1499 non-null   int64 
 3   AnnoNascita     1499 non-null   int64 
 4   NomeParco       1499 non-null   object
 5   SpecieAnimale   1499 non-null   object
 6   OrdineAnimale   1499 non-null   object
 7   StatoSalute     1499 non-null   object
dtypes: int64(2), object(6)
memory usage: 93.8+ KB
```

La funzione **nunique()** conta i valori univoci. Possiamo controllare la presenza di dati duplicati e capire, qualora dovesse servire, come eliminarli.

In [26]:
```
df.nunique()
```

Out[26]:
```
IDAnimale        1499
Sesso               2
Età                49
AnnoNascita        49
NomeParco          38
SpecieAnimale      24
OrdineAnimale       5
StatoSalute         5
dtype: int64
```

Successivamente, tramite le funzioni **.isnull().sum()** controlliamo la presenza dei valori NULL.

In [27]:
```
df.isnull().sum()
```

Out[27]:
```
IDAnimale       0
Sesso           0
Età             0
AnnoNascita     0
NomeParco       0
SpecieAnimale   0
OrdineAnimale   0
StatoSalute     0
dtype: int64
```

Anche espresso in **%**

In [28]:
```
(df.isnull().sum()/(len(df)))*100
```

Out[28]:
```
IDAnimale       0.0
Sesso           0.0
Età             0.0
AnnoNascita     0.0
NomeParco       0.0
SpecieAnimale   0.0
OrdineAnimale   0.0
StatoSalute     0.0
dtype: float64
```

Statistic Summery: con la funzione .describe() abbiamo una visione generale degli indici statistici dei valori quantitativi del dataset.

In [29]:
```
df.describe()
```

Out[29]:

	Età	AnnoNascita
count	1499.000000	1499.000000
mean	8.255504	2015.440294
std	8.257502	8.483370
min	1.000000	1936.000000
25%	3.000000	2012.000000
50%	6.000000	2018.000000
75%	11.000000	2021.000000
max	90.000000	2023.000000

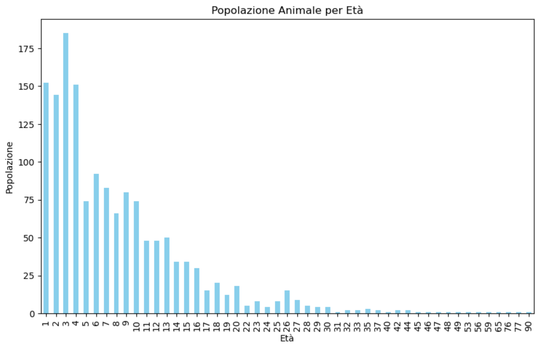
Separiamo le variabile quantitative e qualitative:

In [30]:
```
cat_cols=df.select_dtypes(include=['object']).columns
num_cols = df.select_dtypes(include=np.number).columns.tolist()
print("Qualitative:")
print(cat_cols)
print("Quantitative:")
print(num_cols)
```

```
Qualitative:
Index(['IDAnimale', 'Sesso', 'NomeParco', 'SpecieAnimale', 'OrdineAnimale',
       'StatoSalute'],
      dtype='object')
Quantitative:
['Età', 'AnnoNascita']
```

Il grafico qui sotto ci mostrerà la **distribuzione dell'età in fuzione alla Popolazione Animale:**

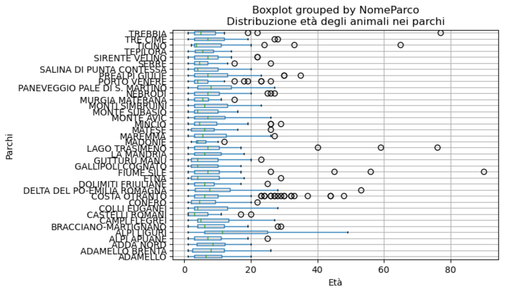
In [33]:
```
età = df['Età']
distribuzione_età = età.value_counts().sort_index()
plt.figure(figsize=(10, 6))
distribuzione_età.plot(kind='bar', color='skyblue')
plt.title('Popolazione Animale per Età')
plt.xlabel('Età')
plt.ylabel('Popolazione')
plt.show()
```



Successivamente, il grafico seguente ci mostrerà la **Distribuzione dell'età degli animali nei parchi:**

In [37]:
```
plt.figure(figsize=(10, 6))
df.boxplot(column='Età', by='NomeParco', vert=False)
plt.title('Distribuzione età degli animali nei parchi')
plt.xlabel('Età')
plt.ylabel('Parchi')
plt.show()
```

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



E' stata eseguita una breve EDA, sicuramente è possibile effettuare ancora altri studi e analisi univariate, bivariate e multivariate.