In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
```

EDA

In [2]:
```
#convertim fisierul csv in dataframe
df = pd.read_csv("train.csv")
```

In [3]:
```
#vizualizam primele 5 linii din tabel
df.head(5)
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [4]:
```
#Afisam statistici despre setul nostru de date
df.describe()  # doar pentru date continue
#df.describe(include = "all") ---> pentru toate
```

Out[4]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [5]:
```
#Afisam informatii despre dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [6]:
```
#Verificam daca avem linii duplicate
df.duplicated().value_counts()

#Observam ca avem False 891, numar care coincide cu numarul de linii din DataFrame, ceea ce inseamna ca nu avem
```

Out[6]:
```
False    891
Name: count, dtype: int64
```

Observ ca exista valori de null in 3 coloane: Age, Embarked, Cabin.

Plan initial:

Pentru coloana Age: o sa extrag statutul persoanelor din nume si o sa atribui media in functie de statut pentru valoriile nule.

Pentru coloana Embarked: avem doar 2 elemente de null, asadar o sa le pun cu random.

Pentru coloana Cabin, ma gandesc ca ar trebui scoasa cu totul, insa trebuie inspectat mai in detaliu

Pentru coloana Age

In [7]:
```
# PAS1 : Facem o coloana noua, unde sa extragem daca persoana este : Mr., Mrs., Miss., Master etc.

def extract_title(text):
    '''
    Functia are rolu de a extrage statul persoanei din coloana Name
    '''

    #Spargem textul in cuvinte
    res = text.split()

    #Definim posibilele titlurile sociale
    valid_titles = {"miss", "master", "mr", "mrs", "ms", "sir", "dr", "don", "col", "rev", "mlle", "mme", "majo

    #Spargem textul in cuvinte si aplicam .lower() pentru a face textul cu litere mici si .strip() pentru a sca
    words = [word.lower().strip('. ') for word in res]

    #Extragem titlurile sociale din cuvinte
    final_titles = [word for word in words if word in valid_titles]

    return final_titles[0] if final_titles else None

#Cream o coloana noua numita Title unde inseram statul extras din coloana Name
df["Title"] = df["Name"].apply(extract_title)   # aplicam functia de mai sus pe coloana Name

df.head()

#Rev -> un fel de preoti, cel mai probabil au sansa de supravietuire 0
#Mlle. -> Mademoiselle., rang mare, sanse mari de supravietuire
#Master -> pentru copii (baieti), probabil au sanse de supravietuire mai mari
#Mr. -> barbat fara un anume status social
#Dr. -> Doctor


# Miss -> femei <= 30 de ani nemaritate
# Mrs -> femeie maritata
```

Out[7]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	mr
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	mrs
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	miss
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	mrs
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	mr

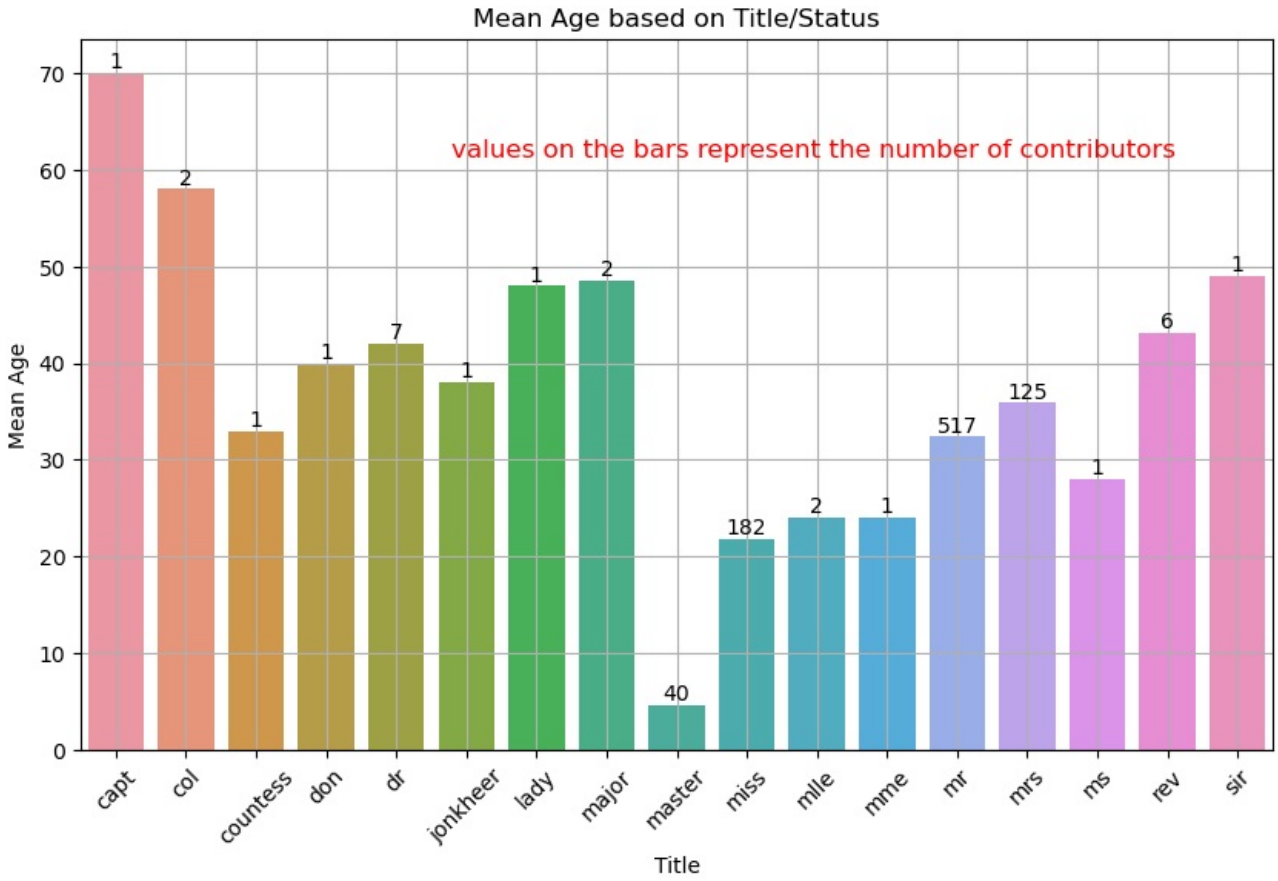
In [8]:
```
# PAS2 : Facem un nou df, unde calculam mediile de varsta pentru fiecare statut social
df_mean_age = round(df["Age"].groupby(df["Title"]).mean(), 1).reset_index()
df_mean_age = df_mean_age.rename(columns = {"Age" : "Mean_Age"}) #redenumim coloana corect
#print(df_mean_age)


#As vrea sa adaug si o coloana cu count sa vad cati indivizi au contribuit la obtinerea mediilor
df_count = df["Title"].value_counts().reset_index()
df_count = df_count.rename(columns = {"count" : "Mean_contributors"})
#print(df_count)

#Urmatorul pas este sa le dau merge, ca sa obtin un tabel cu urmatoarele informatii: Title, Mean_Age si Nr_cont
df_mean_age_count = df_mean_age.merge(df_count, how = "inner", on = "Title")
df_mean_age_count
```

Out[8]:

	Title	Mean_Age	Mean_contributors
0	capt	70.0	1
1	col	58.0	2
2	countess	33.0	1
3	don	40.0	1
4	dr	42.0	7
5	jonkheer	38.0	1
6	lady	48.0	1
7	major	48.5	2
8	master	4.6	40
9	miss	21.8	182
10	mlle	24.0	2
11	mme	24.0	1
12	mr	32.4	517
13	mrs	35.9	125
14	ms	28.0	1
15	rev	43.2	6
16	sir	49.0	1

Pentru a fi siguri totusi ca abordarea aleasa are sens, ar fi interesant daca am si vizualiza media varstei in functie de statutul persoanei. De asemenea, este extrem de important sa vedem si cati oameni au contribuit la obtinerea acestor medii. Cu cat este mai mare numarul de contribuitori, cu atat are mai mult sens ca un individ fara varsta, dar cu un anumit statut social, sa aiba media varstei obtinute pentru acea categorie.

In [9]:
```
#Facem un barplot ca sa vizualizam concret media in functie de statut
plt.figure(figsize=(10, 6))
sns.barplot(data = df_mean_age_count, x = df_mean_age_count["Title"], y = df_mean_age_count["Mean_Age"])

#Punem valoriile de count deasupra fiecarei coloane ca sa observam si mai bine numarul de contribuitori ai medi
for index, row in df_mean_age_count.iterrows():
    plt.text(index, row["Mean_Age"], str(row["Mean_contributors"]), ha='center', va='bottom')

plt.xlabel("Title")
plt.ylabel("Mean Age")
plt.title("Mean Age based on Title/Status")
plt.grid()
plt.xticks(rotation = 45)

#Adaug o legenda
plt.text(10, 62, "values on the bars represent the number of contributors ", ha='center', va='center', fontsize

plt.show()
```

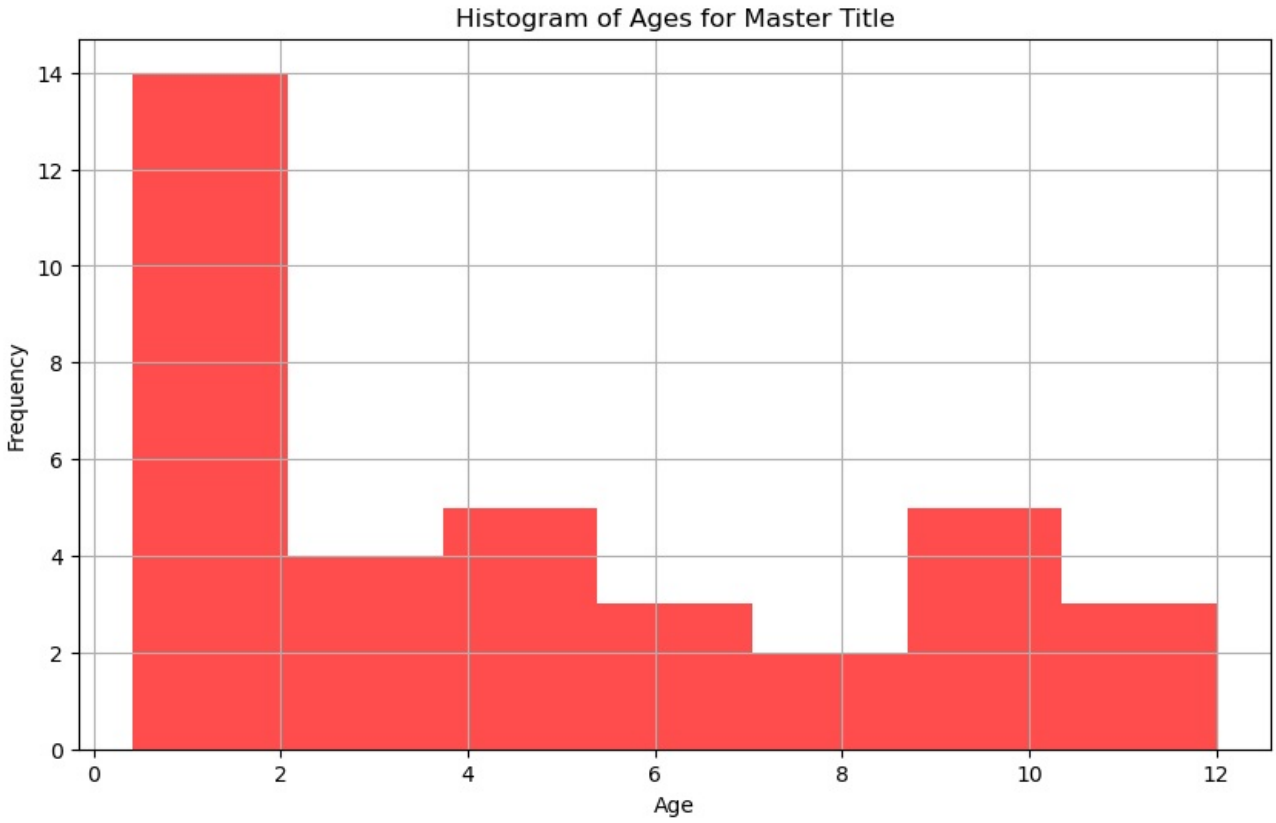


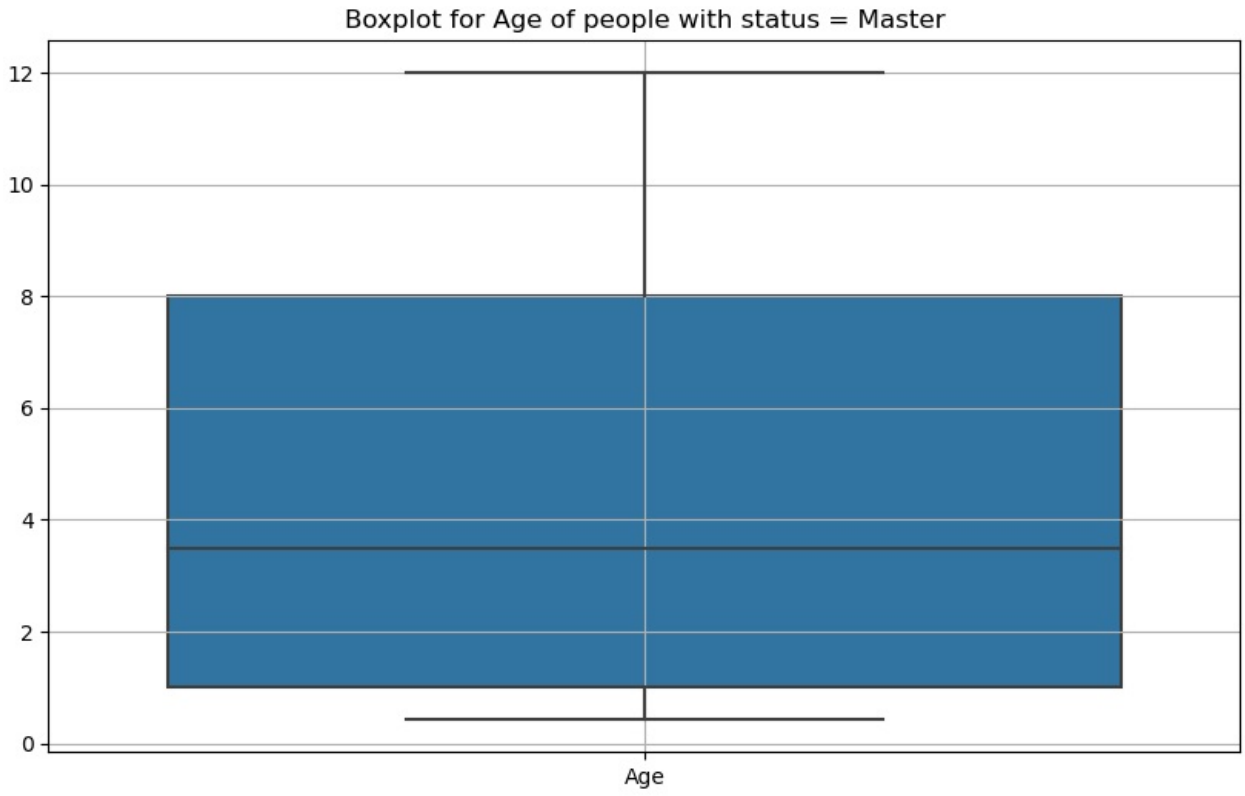
Urmatorul pas este, pentru categoriile care au mai mult de 20-30 de elemente, sa vizualizam histogramele. Ar ajuta sa vedem ce distributie urmaresc aceste varste si, in functie de asta, daca este normal, skewed to the left / right, sa decim ce valori punem.

De asemenea, ar fi important sa vizualizam si boxploturi sa vedem daca avem outliere sau nu.

Pentru Master

In [10]:
```
#Facem filtrul pentru a filtra dataframeul pentru a obtine doar varstele celor cu titlul Master
mask_age_master = ((df["Title"] == "master") & (df["Age"].notnull()))
df_hist_master = df[mask_age_master]

#Calculam numarul optim de binuri
nr_bins = int(np.floor(np.sqrt(df_hist_master["Age"].count())) + 1)


#Generam histograma pe setul filtrat
plt.figure(figsize=(10, 6))
plt.hist(data=df_hist_master, x="Age", bins=nr_bins, color="red", alpha=0.7)

plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Histogram of Ages for Master Title")
plt.grid()

plt.show()

#Generam boxplot cu varstele celor cu titlu = Master
plt.figure(figsize=(10, 6))
sns.boxplot(df_hist_master["Age"].reset_index().drop(columns = "index"))

plt.title("Boxplot for Age of people with status = Master")
plt.grid()

plt.show()
```



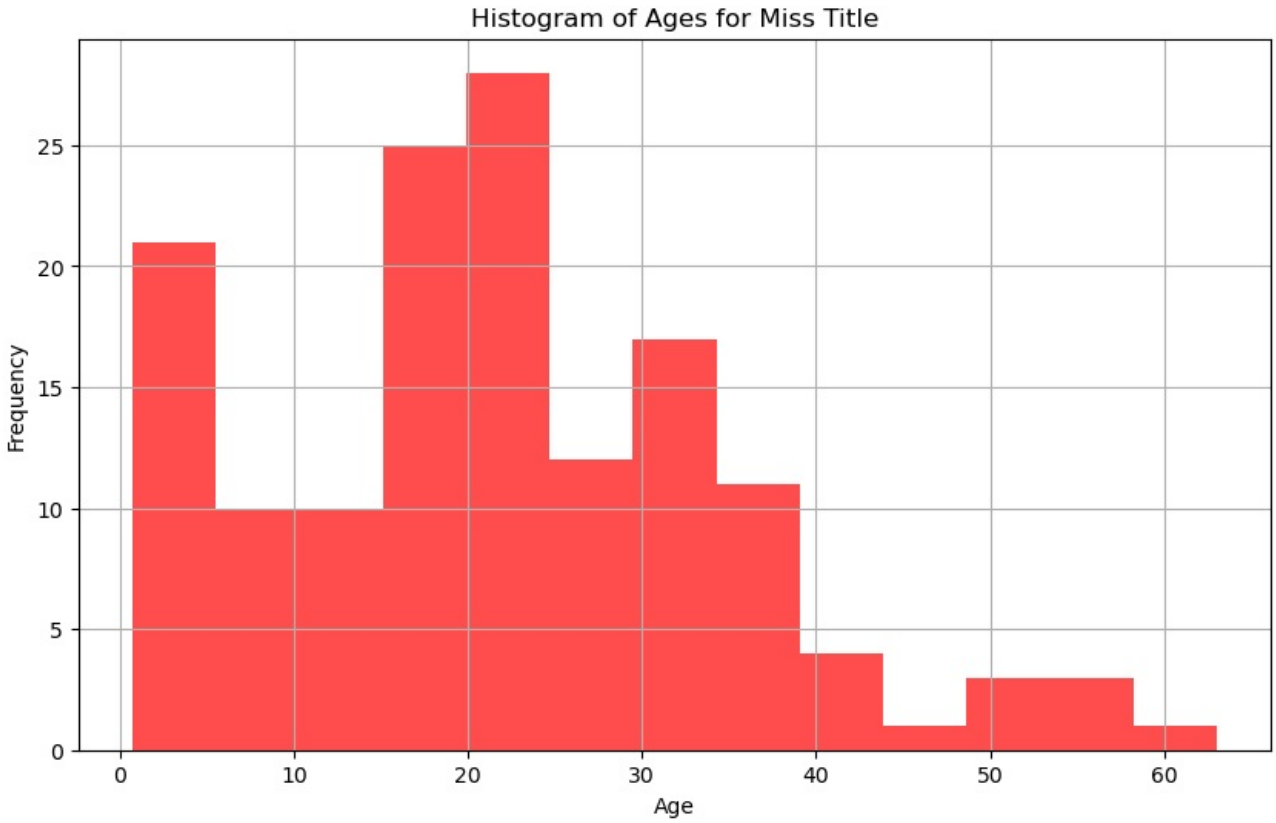


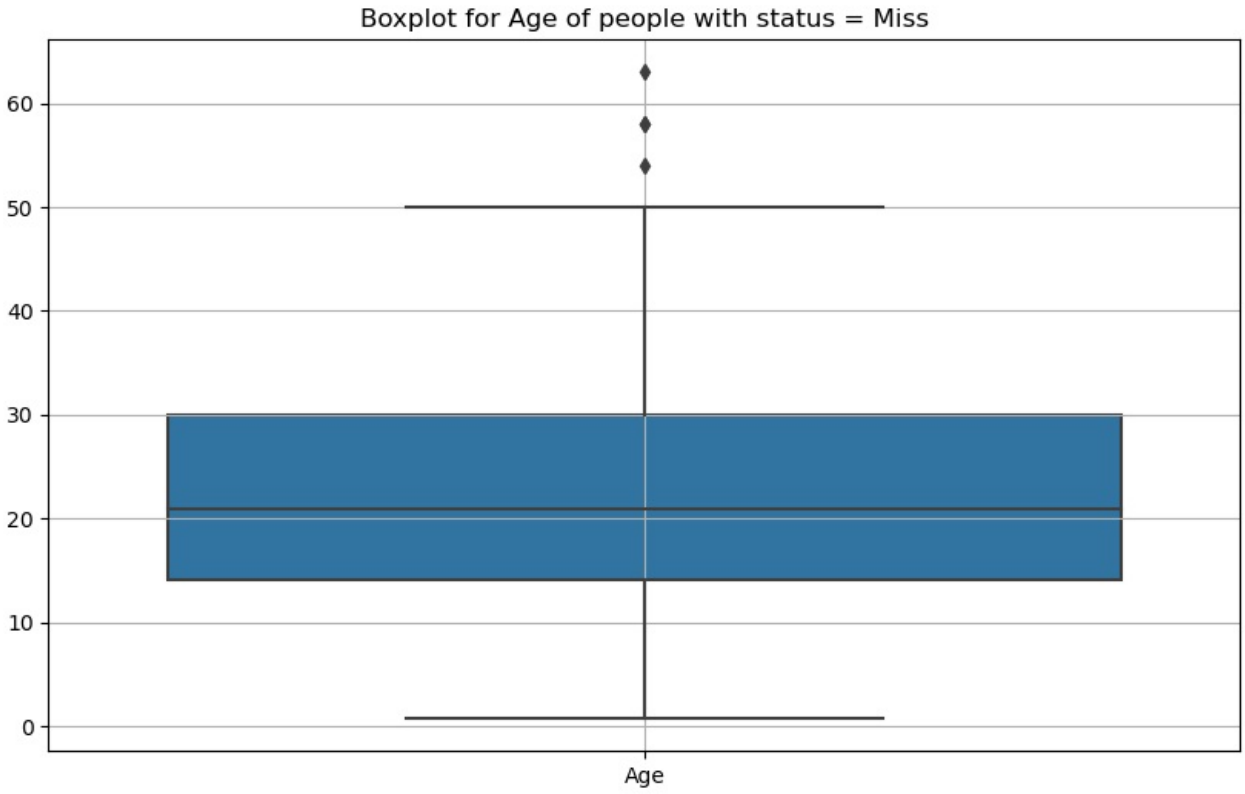
Observam ca distributia varstei pentru cei cu titlul de Master este skewed to the right. Ce e important este ca nu avem outliere, ceea ce inseamna ca valoriile de null vor putea fi inlocuite cu media (daca aveam outliere, le inlocuiam cu medianul). Mentionez ca ideea in sine de outlier are sens in acest context, intrucat Master se aplica pentru copiii intre 0-18 ani. Este gresit sa ai acest statut dupa varsta de 18 ani.

Pentru MIss

In [11]:
```
#Facem filtrul pentru a filtra dataframeul pentru a obtine doar varstele celor cu titlul Master
mask_age_miss = ((df["Title"] == "miss") & (df["Age"].notnull()))
df_hist_miss = df[mask_age_miss]

#Calculam numarul optim de binuri
nr_bins = int(np.floor(np.sqrt(df_hist_miss["Age"].count())) + 1)


#Generam histograma pe setul filtrat
plt.figure(figsize=(10, 6))
plt.hist(data=df_hist_miss, x="Age", bins=nr_bins, color="red", alpha=0.7)

plt.xlabel("Age")
```

```
plt.ylabel("Frequency")
plt.title("Histogram of Ages for Miss Title")
plt.grid()

plt.show()

#Generam boxplot cu varstele celor cu titlu = Master
plt.figure(figsize=(10, 6))
sns.boxplot(df_hist_miss["Age"].reset_index().drop(columns = "index"))

plt.title("Boxplot for Age of people with status = Miss")
plt.grid()

plt.show()
```





Daca am inteles bine, MIss. se foloseste pentru femei <= 30 de ani si NEMARITATE. Asta inseamna ca, pentru 25% din femei dupa cum se vede in boxplot, ar trebui schimbat titlul ori in Mrs (femeie maritata), ori in Ms (femeieis nemaritata si peste 30 SAU dacă preferă să fie adresată cu un titlu neutru din punct de vedere civil)

Raportat la ce valori punem la varsta, mai intai trebuie sa modificam statutul pentru femeile > 30 de ani si dupa sa recalculam media. Dupa aceea, valoriile de Nan vor lua noua medie calculata

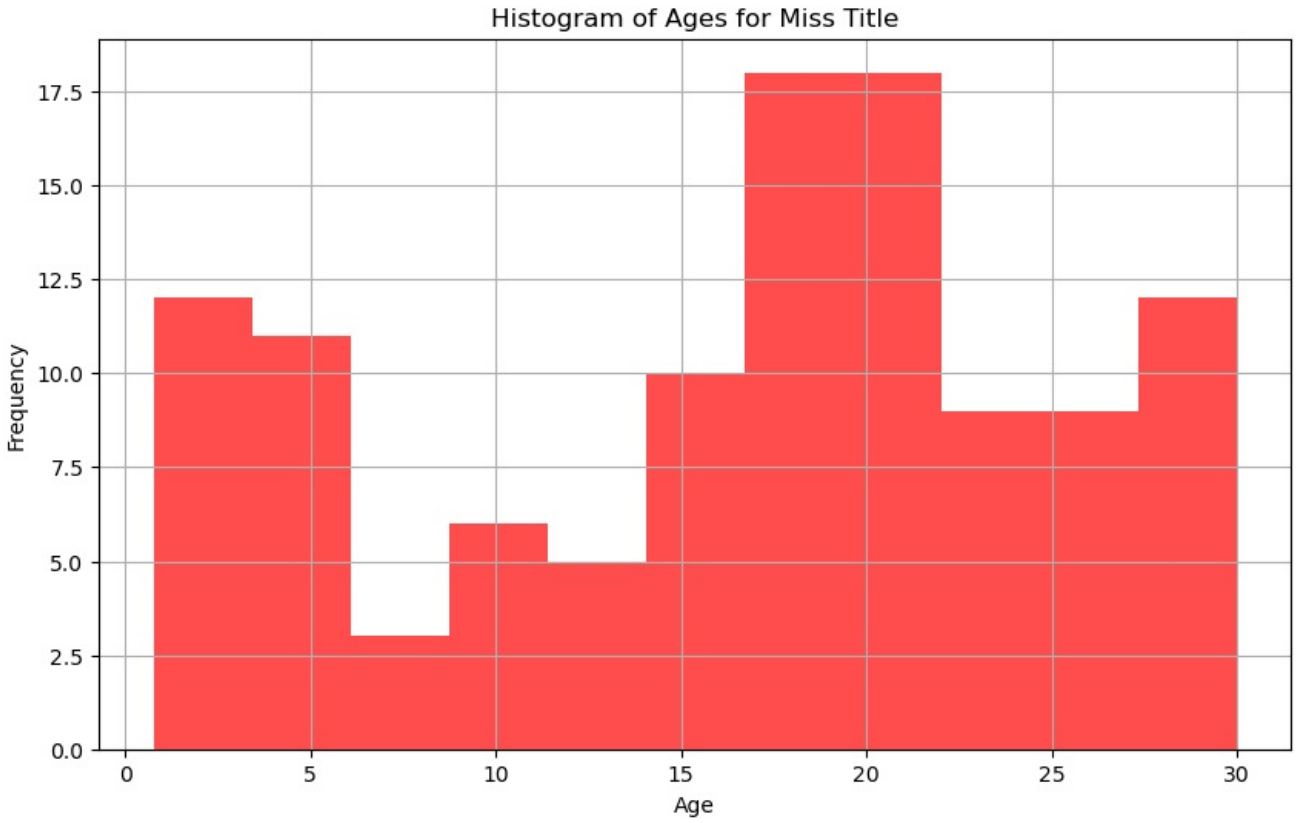
Am realizat ca trebuie sa mai fac o coloana, unde sa vad daca si-au schimbat numele sau nu (ca sa vad daca au fost maritate)

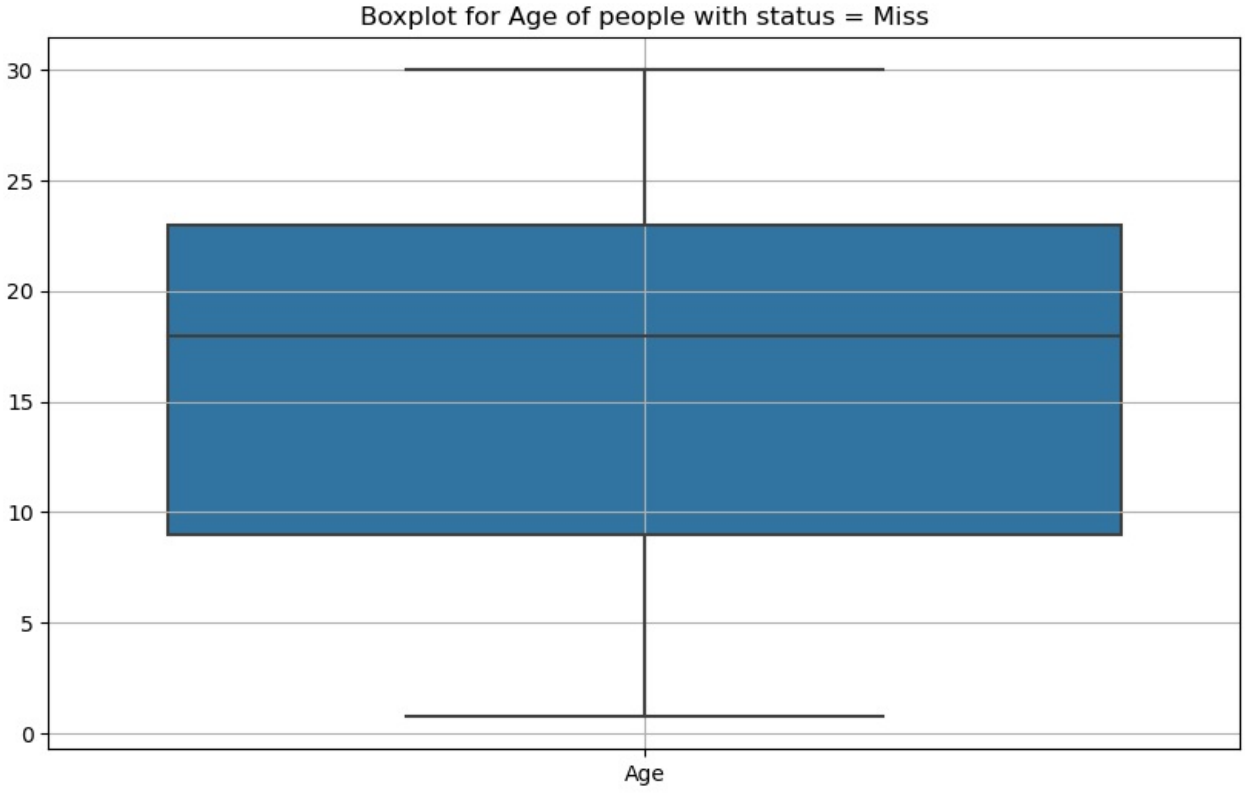
In [12]:
```
#### de citit mai mult despre regular expressions

import re

def extract_text_in_parentheses(text):

    '''
    Functia are rolul de a extragere numele vechi al persoanei.
    '''

    # Define a regular expression pattern to match text within parentheses
    pattern = r'\((.*?)\)'

    # Use re.findall to find all matches of the pattern in the text
    matches = re.findall(pattern, text)

    # Return the first match found (if any)
    if matches:
        return matches[0]
    else:
        return None

# Example usage
df["Previous_name"] = df["Name"].apply(extract_text_in_parentheses)
```

In [13]:
```
# Dupa cum observam, avem 3 cazuri care sa respecte conditia de mai jos: pentru 2 din aceste cazuri: 199, 427 o

mask_for_miss_married = (
    (df["Title"] == "miss") & (df["Age"].notnull()) &
    ((df["SibSp"] == 0) | (df["Age"] < 30)) &
    (df["Previous_name"].notnull())
)
df[mask_for_miss_married]

df.loc[[199,427], "Title"] = "mrs"
```

In [14]:
```
#La fel, verificam femeile peste 30 de ani care sunt miss SI care nu au numele schimbat -> vor deveni mrs
mask_for_miss_married = (
    (df["Title"] == "miss") & (df["Age"].notnull()) &
    (df["Age"] > 30) &
    (df["Previous_name"].isnull())
)
df.loc[mask_for_miss_married, "Title"] = "mrs"

# Nu avem femei cu conditia de mai sus si cu nume schimbat
```

In [15]:
```
## REFACEM HISTOGRAMA SI BOXPLOTUL SA VEDEM MODIFICARILE

#Facem filtrul pentru a filtra dataframeul pentru a obtine doar varstele celor cu titlul Master
mask_age_miss = ((df["Title"] == "miss") & (df["Age"].notnull()))
df_hist_miss = df[mask_age_miss]

#Calculam numarul optim de binuri
nr_bins = int(np.floor(np.sqrt(df_hist_miss["Age"].count())) + 1)


#Generam histograma pe setul filtrat
plt.figure(figsize=(10, 6))
plt.hist(data=df_hist_miss, x="Age", bins=nr_bins, color="red", alpha=0.7)

plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Histogram of Ages for Miss Title")
plt.grid()

plt.show()

#Generam boxplot cu varstele celor cu titlu = Master
plt.figure(figsize=(10, 6))
sns.boxplot(df_hist_miss["Age"].reset_index().drop(columns = "index"))

plt.title("Boxplot for Age of people with status = Miss")
plt.grid()

plt.show()
```



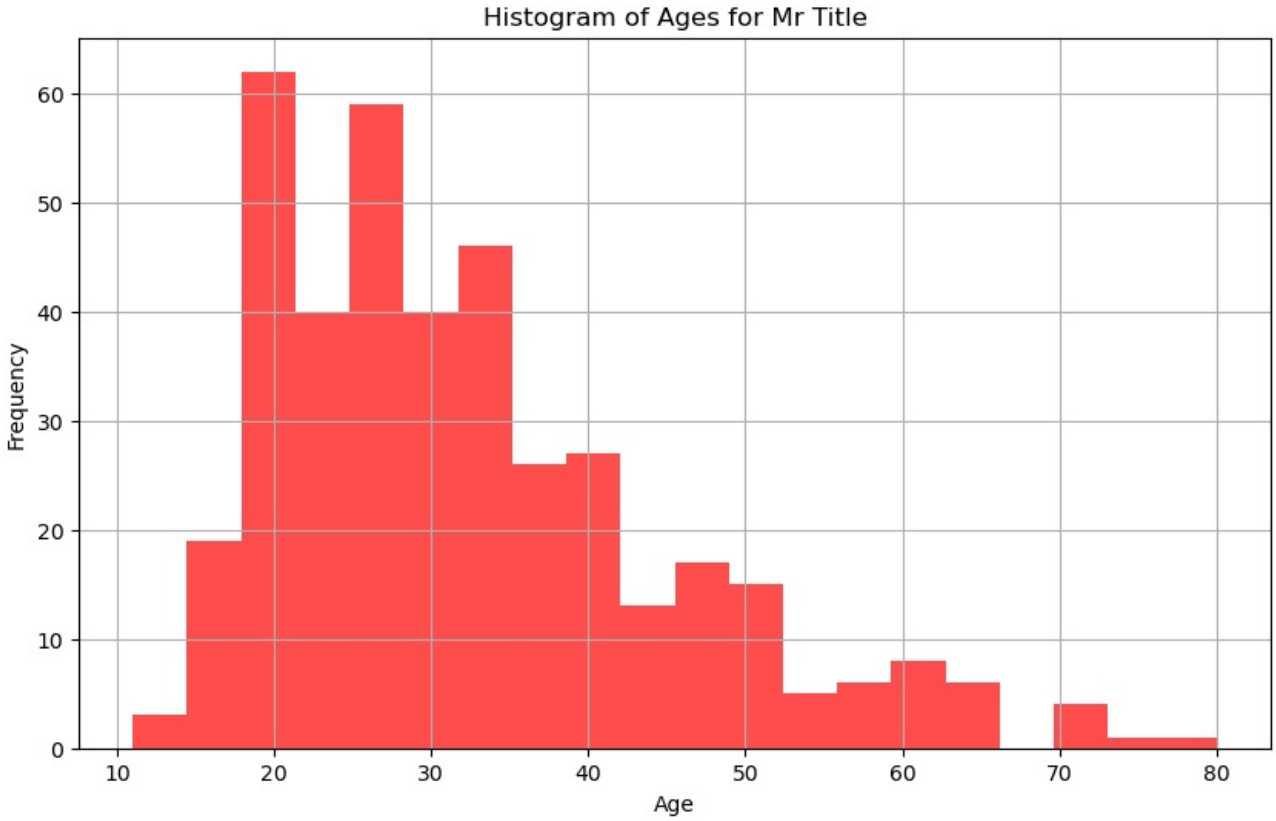


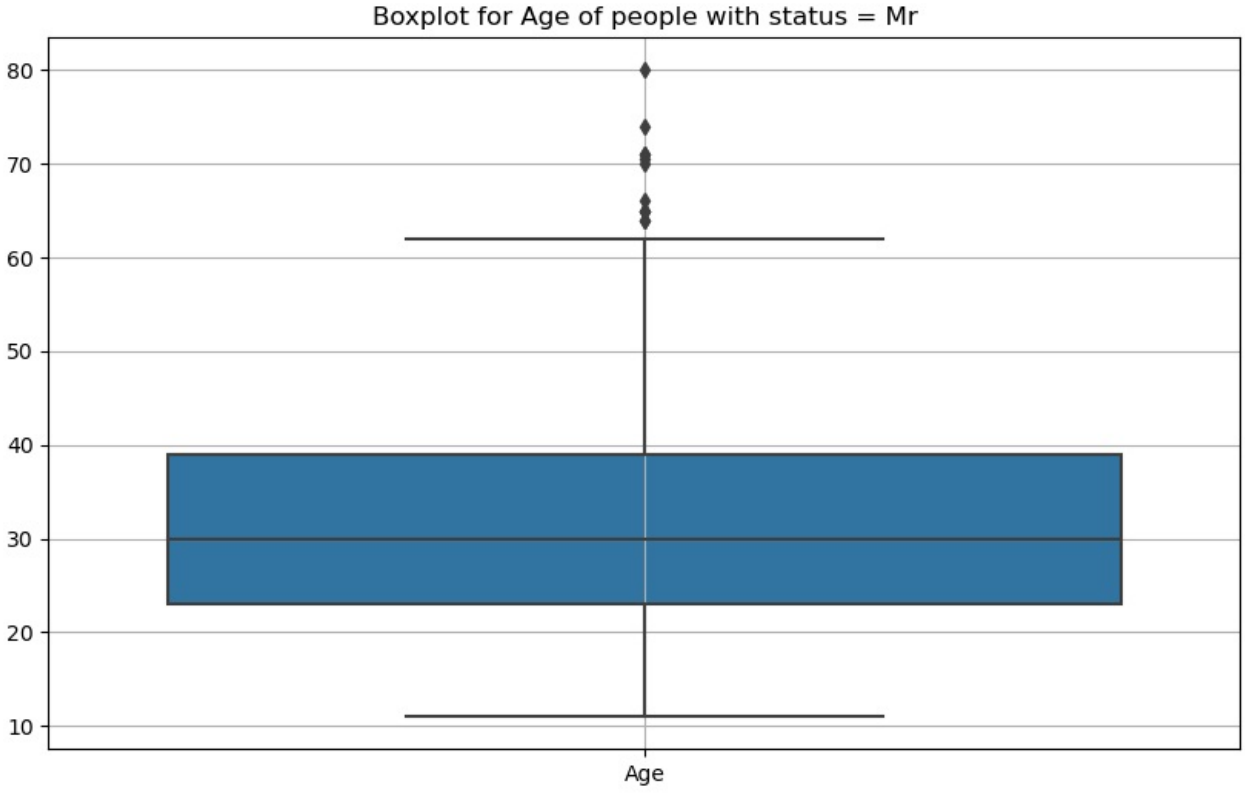
OBSERVAM CA BOXPLOTUL ARATA MULT MAI BINE.

Pentru Mr

In [16]:
```
#Facem filtrul pentru a filtra dataframeul pentru a obtine doar varstele celor cu titlul Master
mask_age_mr = ((df["Title"] == "mr") & (df["Age"].notnull()))
df_hist_mr = df[mask_age_mr]

#Calculam numarul optim de binuri
nr_bins = int(np.floor(np.sqrt(df_hist_mr["Age"].count())) + 1)


#Generam histograma pe setul filtrat
plt.figure(figsize=(10, 6))
plt.hist(data=df_hist_mr, x="Age", bins=nr_bins, color="red", alpha=0.7)

plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Histogram of Ages for Mr Title")
plt.grid()
```

```
plt.show()

#Generam boxplot cu varstele celor cu titlu = Master
plt.figure(figsize=(10, 6))
sns.boxplot(df_hist_mr["Age"].reset_index().drop(columns = "index"))

plt.title("Boxplot for Age of people with status = Mr")
plt.grid()

plt.show()
```





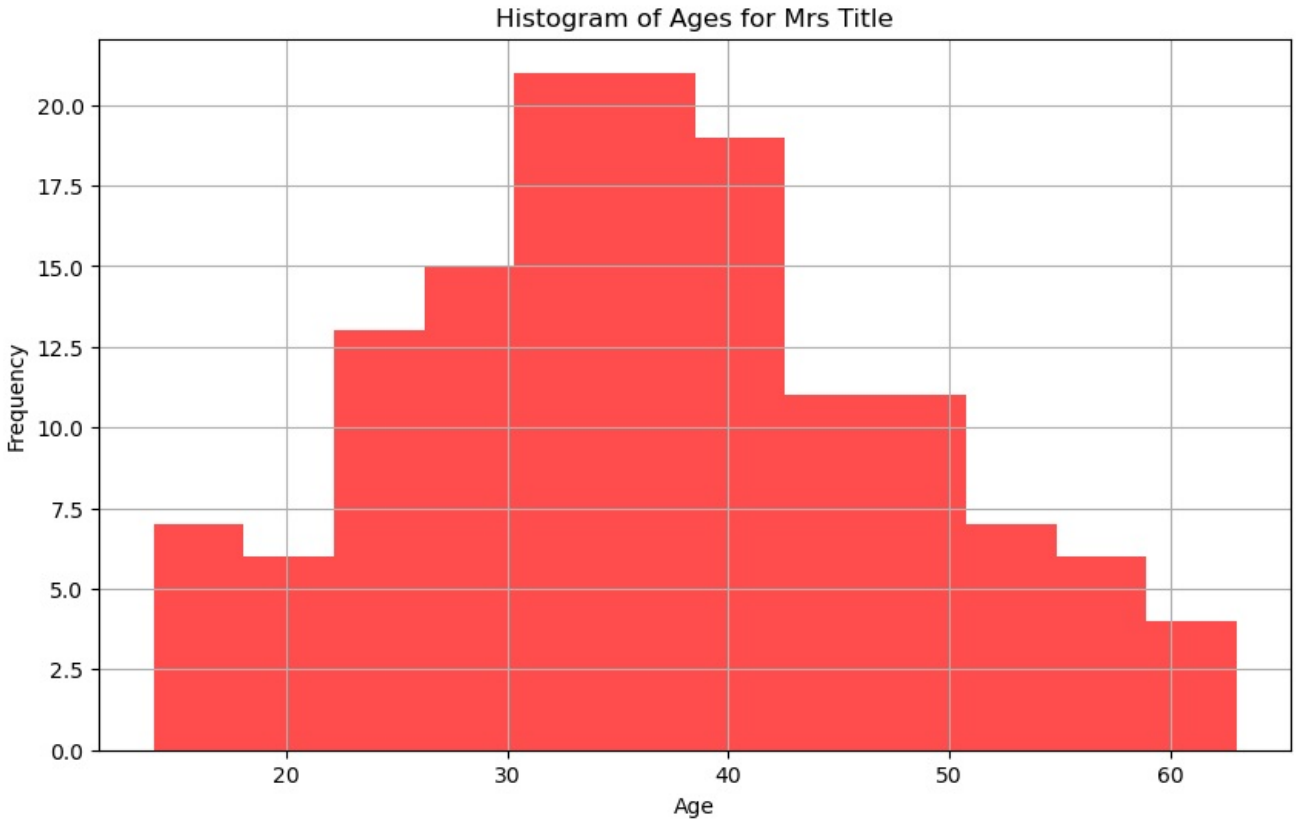
Aici pare totul ok, nu vad nimic dubios sau care sa nu fie in neregula.

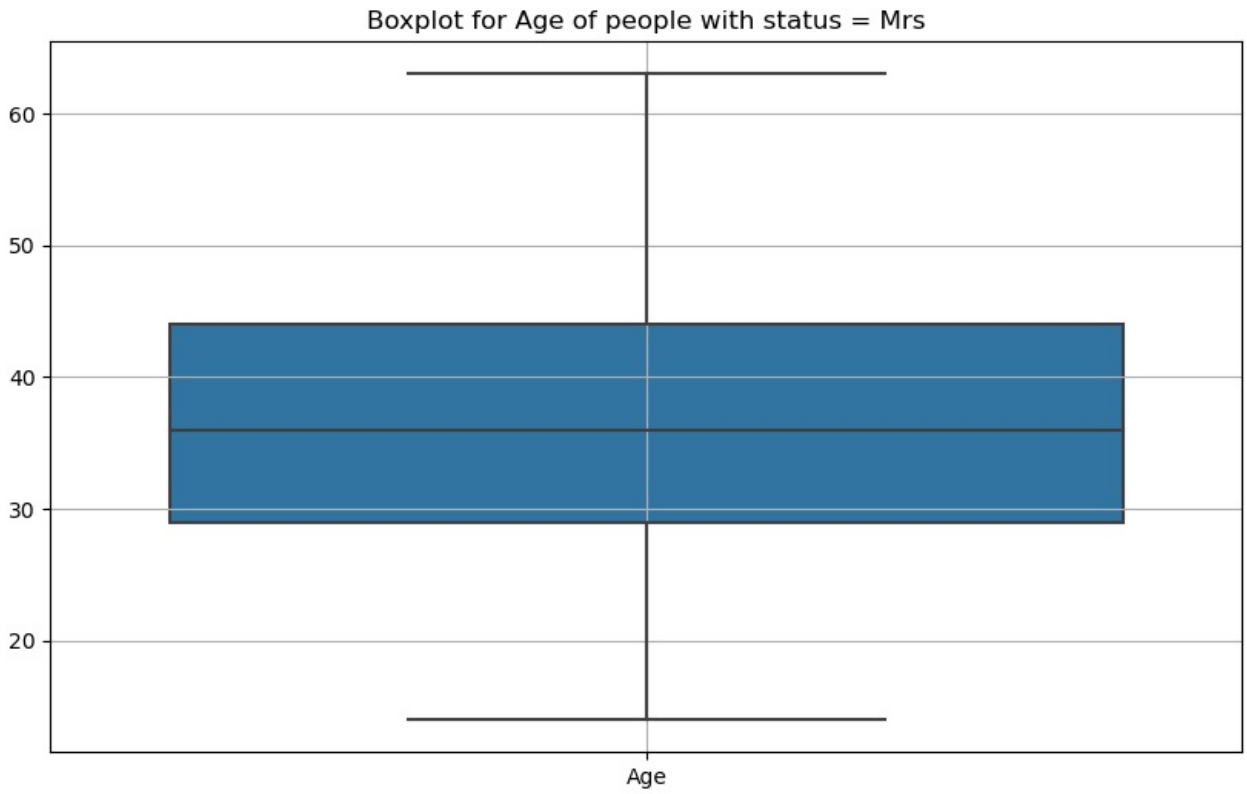
In [17]:
```
mask_mr = (df["Age"] < 18) & (df["Title"] == "mr")
df.loc[731, "Title"] = "master"  # are 11 ani deci il mutam
```

Aici pare totul ok, nu vad nimic dubios sau care sa nu fie in neregula.

Pentru Mrs

In [18]:
```
#Facem filtrul pentru a filtra dataframeul pentru a obtine doar varstele celor cu titlul Master
mask_age_mrs = ((df["Title"] == "mrs") & (df["Age"].notnull()))
df_hist_mrs = df[mask_age_mrs]

#Calculam numarul optim de binuri
nr_bins = int(np.floor(np.sqrt(df_hist_mrs["Age"].count())) + 1)


#Generam histograma pe setul filtrat
plt.figure(figsize=(10, 6))
plt.hist(data=df_hist_mrs, x="Age", bins=nr_bins, color="red", alpha=0.7)

plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Histogram of Ages for Mrs Title")
plt.grid()

plt.show()

#Generam boxplot cu varstele celor cu titlu = Master
plt.figure(figsize=(10, 6))
sns.boxplot(df_hist_mrs["Age"].reset_index().drop(columns = "index"))

plt.title("Boxplot for Age of people with status = Mrs")
plt.grid()

plt.show()
```





In [20]:
```
mask_mrs = (df["Age"] > 20) & (df["Title"] == "mrs")# & (df["Previous_name"].notnull())
df[mask_mrs]
```

Out[20]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title	Previous_name
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	mrs	Florence Briggs Thayer
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	mrs	Lily May Peel
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S	mrs	Elisabeth Vilhelmina Berg
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S	mrs	None
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.0000	NaN	S	mrs	Mary D Kingcome
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
871	872	1	1	Beckwith, Mrs. Richard Leonard (Sallie Monypeny)	female	47.0	1	1	11751	52.5542	D35	S	mrs	Sallie Monypeny
874	875	1	2	Abelson, Mrs. Samuel (Hannah Wizosky)	female	28.0	1	0	P/PP 3381	24.0000	NaN	C	mrs	Hannah Wizosky
879	880	1	1	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)	female	56.0	0	1	11767	83.1583	C50	C	mrs	Lily Alexenia Wilson
880	881	1	2	Shelley, Mrs. William (Imanita Parrish Hall)	female	25.0	0	1	230433	26.0000	NaN	S	mrs	Imanita Parrish Hall
885	886	0	3	Rice, Mrs. William (Margaret Norton)	female	39.0	0	5	382652	29.1250	NaN	Q	mrs	Margaret Norton

131 rows × 14 columns

Si aici pare totul ok

In [27]:
```
#Recalculam mediile cum am calculat mai sus cu noile modificari
# PAS2 : Facem un nou df, unde calculam mediile de varsta pentru fiecare statut social
df_mean_age = round(df["Age"].groupby(df["Title"]).mean(), 1).reset_index().set_index("Title")
df_mean_age
```

Out[27]:

Title	Age
capt	70.0
col	58.0
countess	33.0
don	40.0
dr	42.0
jonkheer	38.0
lady	48.0
major	48.5
master	4.7
miss	16.7
mlle	24.0
mme	24.0
mr	32.4
mrs	36.7
ms	28.0
rev	43.2
sir	49.0

In [28]:
```
# PAS3 : Modificam valoriile NaN cu mediile de varsta pentru fiecare statut social
```

```
mask_age = df["Age"].isnull()   #conditia dupa care filtram dataframe-ul

df.loc[mask_age, "Age"] = df.loc[mask_age,"Title"].map(df_mean_age["Age"])   # modificam valoriile NaN cu medii
```

OFICIAL GATA CU VARSTA