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

In [2]:
```
data = pd.read_csv('HRDataset_v14.csv', sep=",")
```

In [3]:
```
print("Размер таблицы: ", data.shape)
```

```
Размер таблицы:  (311, 36)
```

In [4]:
```
data.head()
```

Out[4]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScore
0	Adinolfi, Wilson K	10026	0.0	0	1	1	5	
1	Ait Sidi, Karthikeyan	10084	NaN	1	1	5	3	
2	Akinkuolie, Sarah	10196	NaN	1	0	5	5	
3	Alagbe,Trina	10088	1.0	1	0	1	5	
4	Anderson, Carol	10069	0.0	2	0	5	5	

5 rows × 36 columns

In [5]:
```
data.dtypes
```

Out[5]:
```
Employee_Name                object
EmpID                         int64
MarriedID                   float64
MaritalStatusID               int64
GenderID                      int64
EmpStatusID                   int64
DeptID                        int64
PerfScoreID                   int64
FromDiversityJobFairID        int64
Salary                        int64
Termd                         int64
PositionID                    int64
Position                     object
State                        object
Zip                           int64
DOB                          object
Sex                          object
MaritalDesc                  object
CitizenDesc                  object
HispanicLatino               object
RaceDesc                     object
DateofHire                   object
DateofTermination            object
TermReason                   object
EmploymentStatus             object
Department                   object
ManagerName                  object
ManagerID                   float64
RecruitmentSource            object
```

```
PerformanceScore              object
EngagementSurvey             float64
EmpSatisfaction                int64
SpecialProjectsCount           int64
LastPerformanceReview_Date    object
DaysLateLast30                 int64
Absences                       int64
dtype: object
```

In [6]:
```
# Сумма пропущенных значений
isnull = data.isnull().sum()
print(isnull)
```

```
Employee_Name                  0
EmpID                          0
MarriedID                     20
MaritalStatusID                0
GenderID                       0
EmpStatusID                    0
DeptID                         0
PerfScoreID                    0
FromDiversityJobFairID         0
Salary                         0
Termd                          0
PositionID                     0
Position                       0
State                          0
Zip                            0
DOB                            0
Sex                            0
MaritalDesc                    0
CitizenDesc                    0
HispanicLatino                 0
RaceDesc                      20
DateofHire                     0
DateofTermination            207
TermReason                   207
EmploymentStatus               0
Department                     0
ManagerName                    0
ManagerID                      8
RecruitmentSource              0
PerformanceScore               0
EngagementSurvey               0
EmpSatisfaction                0
SpecialProjectsCount           0
LastPerformanceReview_Date     0
DaysLateLast30                 0
Absences                       0
dtype: int64
```

2. Обработка данных

2.1. Удаление значений

In [8]:
```
# Удаление столбцов
newdata1 = data.dropna(axis=1)
newdata1.shape
```

Out[8]: (311, 31)

In [9]:
```
newdata1.dtypes
```

Out[9]:
```
Employee_Name                object
EmpID                         int64
MaritalStatusID               int64
GenderID                      int64
EmpStatusID                   int64
DeptID                        int64
PerfScoreID                   int64
FromDiversityJobFairID        int64
Salary                        int64
Termd                         int64
PositionID                    int64
Position                     object
State                        object
Zip                           int64
DOB                          object
Sex                          object
MaritalDesc                  object
CitizenDesc                  object
HispanicLatino               object
DateofHire                   object
EmploymentStatus             object
Department                   object
ManagerName                  object
RecruitmentSource            object
PerformanceScore             object
EngagementSurvey            float64
EmpSatisfaction               int64
SpecialProjectsCount          int64
LastPerformanceReview_Date   object
DaysLateLast30                int64
Absences                      int64
dtype: object
```

In [10]:
```
# Удаление строк
newdata2 = data.dropna(axis=0)
newdata2.shape
```

Out[10]:
```
(88, 36)
```

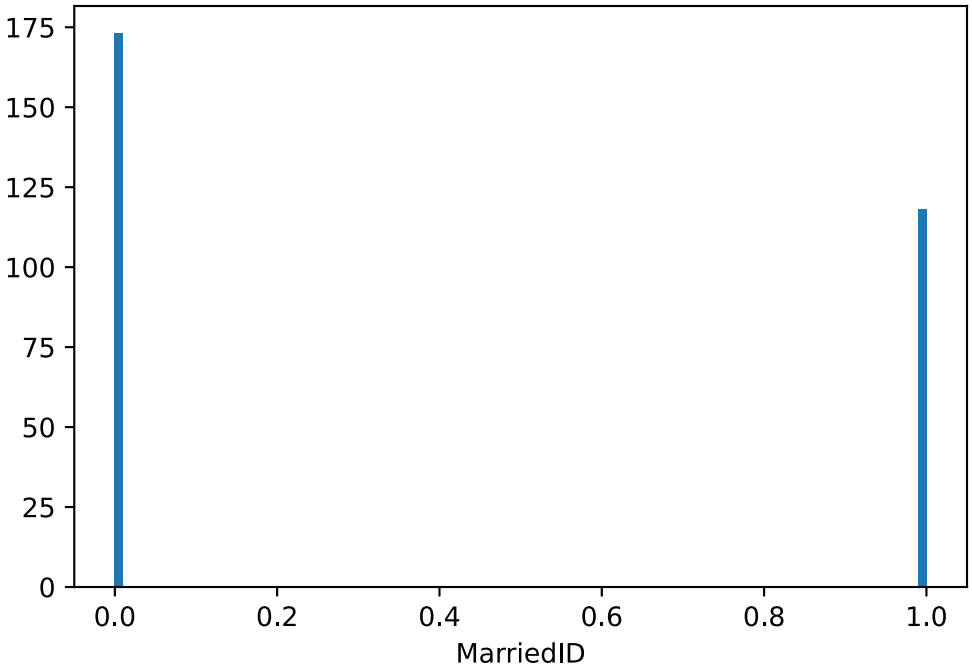
In [11]:
```
mass1 = []
mass2 = []
mass3 = []
for key in isnull.keys():
    elem = []
    if isnull[key] != 0:
        elem.append(key)
        elem.append(data[key].dtype)
        elem.append(isnull[key])
        elem.append(round(isnull[key] / data.shape[0] * 100, 5))
        mass1.append(elem)
    if isnull[key] != 0 and (str(data[key].dtype) == 'float64' or str(data[key].dtyp
        mass2.append(elem)
    if isnull[key] != 0 and str(data[key].dtype) == 'object':
        mass3.append(elem)
data_num_obj = []
for key in mass1:
    data_num_obj.append(key[0])
    print("{} - {} - ({}) {}%".format(key[0], key[1], key[2], key[3]))
```

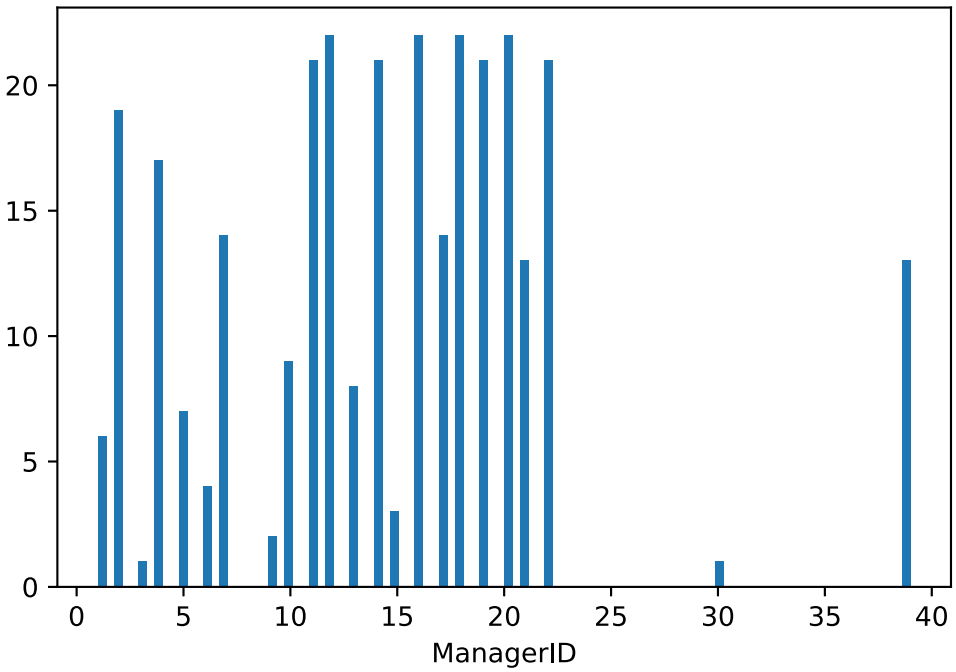
```
MarriedID - float64 - (20) 6.43087%
RaceDesc - object - (20) 6.43087%
DateofTermination - object - (207) 66.55949%
TermReason - object - (207) 66.55949%
ManagerID - float64 - (8) 2.57235%
```

In [12]:
```
data_num = []
for key in mass2:
    data_num.append(key[0])
    print("{} - {} - ({}) {}%".format(key[0], key[1], key[2], key[3]))
```

```
MarriedID - float64 - (20) 6.43087%
ManagerID - float64 - (8) 2.57235%
```

In [13]:
```
for key in mass2:
    plt.hist(data[key[0]], 100)
    plt.xlabel(key[0])
    plt.show()
```





SimpleImputer

In [15]:
```
from sklearn.impute import SimpleImputer
```

```
from sklearn.impute import MissingIndicator
```

In [16]:

```
sort_null_data = data[data_num]
data_MarriedID = sort_null_data[['MarriedID']]
data_MarriedID.head()
```

Out[16]:

	MarriedID
0	0.0
1	NaN
2	NaN
3	1.0
4	0.0

In [17]:

```
implicator = MissingIndicator()
values = implicator.fit_transform(data_MarriedID)
values
```

Out[17]:

```
array([[False],
       [ True],
       [ True],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [ True],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
```

```
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
                [ True],
                [False],
                [False],
                [False],
                [False],
```

```
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [ True],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [ True],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
       [False],
```

```
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[ True],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[ True],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[ True],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
[False],
```

```
        [False],
        [False],
        [False],
        [ True],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [ True],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [ True],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False],
        [ True],
        [False],
        [False],
        [False],
        [False],
        [False],
        [False]])
```

In [18]:

```
strategies = ['mean', 'median', 'most_frequent']
```

In [19]:

```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
```

```
    data_num_imp = imp_num.fit_transform(data_MarriedID)
    return data_num_imp[values]
```

In [20]:
```
strategies[0], test_num_impute(strategies[0])
```

Out[20]:
```
('mean',
 array([0.40549828, 0.40549828, 0.40549828, 0.40549828, 0.40549828,
        0.40549828, 0.40549828, 0.40549828, 0.40549828, 0.40549828,
        0.40549828, 0.40549828, 0.40549828, 0.40549828, 0.40549828,
        0.40549828, 0.40549828, 0.40549828, 0.40549828, 0.40549828]))
```

In [21]:
```
strategies[1], test_num_impute(strategies[1])
```

Out[21]:
```
('median',
 array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0.]))
```

In [22]:
```
strategies[2], test_num_impute(strategies[2])
```

Out[22]:
```
('most_frequent',
 array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0.]))
```

Обработка категориальных признаков

In [24]:
```
for key in mass3:
    print("{} - {} - ({}) {}%".format(key[0], key[1], key[2], key[3]))
```

```
RaceDesc - object - (20) 6.43087%
DateofTermination - object - (207) 66.55949%
TermReason - object - (207) 66.55949%
```

In [25]:
```
sort_null_data_obj = data[data_num_obj]
data_RaceDesc = sort_null_data_obj[['RaceDesc']]
data_RaceDesc.head()
```

Out[25]:

	RaceDesc
0	White
1	NaN
2	White
3	White
4	White

In [26]:
```
implicator = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='N
RaceDesc_values = implicator.fit_transform(data_RaceDesc)
RaceDesc_values
```

Out[26]:
```
array([['White'],
       ['NA'],
       ['White'],
       ['White'],
       ['White'],
       ['White'],
```

```
['White'],
['NA'],
['Black or African American'],
['White'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['Two or more races'],
['White'],
['White'],
['White'],
['White'],
['White'],
['NA'],
['White'],
['Asian'],
['White'],
['NA'],
['White'],
['Black or African American'],
['NA'],
['Asian'],
['White'],
['Black or African American'],
['White'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['NA'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Two or more races'],
['White'],
['White'],
['Black or African American'],
['Two or more races'],
['NA'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
```

```
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['NA'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['Black or African American'],
['White'],
['White'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['Two or more races'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['White'],
['NA'],
['Asian'],
['White'],
['White'],
['Black or African American'],
['Two or more races'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['American Indian or Alaska Native'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['Asian'],
['White'],
['Black or African American'],
['White'],
['American Indian or Alaska Native'],
['White'],
['White'],
```

```
['Asian'],
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['Asian'],
['Black or African American'],
['White'],
['White'],
['Asian'],
['White'],
['Black or African American'],
['White'],
['Asian'],
['Asian'],
['White'],
['Asian'],
['Black or African American'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['White'],
['Two or more races'],
['White'],
['Asian'],
['White'],
['Black or African American'],
['Black or African American'],
['Two or more races'],
['Black or African American'],
['White'],
['Hispanic'],
['NA'],
['White'],
['Black or African American'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['Black or African American'],
['Asian'],
['Asian'],
['Black or African American'],
['White'],
['White'],
['Black or African American'],
['Two or more races'],
['White'],
['White'],
['Two or more races'],
['Asian'],
['White'],
['White'],
['Black or African American'],
['White'],
['White'],
['Black or African American'],
```

```
['White'],
['Black or African American'],
['Asian'],
['White'],
['White'],
['White'],
['Asian'],
['NA'],
['Asian'],
['NA'],
['Black or African American'],
['Asian'],
['White'],
['NA'],
['Black or African American'],
['Black or African American'],
['White'],
['White'],
['NA'],
['Asian'],
['NA'],
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['White'],
['White'],
['Black or African American'],
['White'],
['NA'],
['White'],
['White'],
['Black or African American'],
['Black or African American'],
['Asian'],
['White'],
['White'],
['White'],
['White'],
['White'],
['White'],
['Black or African American'],
['Black or African American'],
['White'],
['Black or African American'],
['Asian'],
['White'],
['White'],
['White'],
['Two or more races'],
['White'],
['White'],
['Asian'],
['NA'],
['Black or African American'],
['White'],
['Asian'],
['White'],
['White'],
['Black or African American'],
['White'],
['Black or African American'],
['White'],
['White'],
['Black or African American'],
```

```
       ['White'],
       ['White'],
       ['White'],
       ['Black or African American'],
       ['White'],
       ['White'],
       ['Two or more races'],
       ['White'],
       ['White'],
       ['Asian'],
       ['NA'],
       ['White'],
       ['White'],
       ['Black or African American'],
       ['Black or African American'],
       ['White'],
       ['Asian'],
       ['Asian'],
       ['White'],
       ['White'],
       ['White'],
       ['NA'],
       ['White'],
       ['White'],
       ['White'],
       ['NA'],
       ['White'],
       ['White'],
       ['Asian']], dtype=object)
```

In [27]:
```
np.unique(RaceDesc_values)
```

Out[27]:
```
array(['American Indian or Alaska Native', 'Asian',
       'Black or African American', 'Hispanic', 'NA', 'Two or more races',
       'White'], dtype=object)
```

Кодирование категориальных признаков

In [30]:
```
data_frame = pd.DataFrame({'RaceDesc': RaceDesc_values.T[0]})
```

In [31]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

In [33]:
```
le = LabelEncoder()
data_label_en = le.fit_transform(data_frame)
```

In [34]:
```
data_frame['RaceDesc'].unique()
```

Out[34]:
```
array(['White', 'NA', 'Black or African American', 'Two or more races',
       'Asian', 'American Indian or Alaska Native', 'Hispanic'],
      dtype=object)
```

In [35]:
```
data_label_en
```

Out[35]:
```
array([6, 4, 6, 6, 6, 6, 6, 4, 2, 6, 2, 2, 2, 5, 6, 6, 6, 6, 6, 4, 6, 1,
       6, 4, 6, 2, 4, 1, 6, 2, 6, 2, 2, 2, 6, 6, 6, 6, 2, 2, 6, 6, 6, 6,
       6, 2, 6, 2, 6, 6, 6, 6, 6, 2, 6, 4, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6,
       2, 5, 4, 6, 6, 6, 6, 2, 6, 6, 2, 6, 6, 6, 6, 6, 6, 6, 6, 6, 4, 2,
       2, 6, 6, 6, 6, 6, 6, 6, 2, 6, 2, 6, 6, 2, 2, 2, 2, 2, 5, 6, 6, 6,
       6, 2, 2, 6, 6, 6, 4, 1, 6, 6, 2, 5, 6, 6, 6, 6, 6, 6, 0, 6, 6, 6,
```

```
2, 6, 6, 6, 6, 1, 6, 2, 6, 0, 6, 6, 1, 6, 2, 6, 6, 6, 6, 6, 6, 2,
1, 2, 6, 6, 1, 6, 2, 6, 1, 1, 6, 1, 2, 6, 6, 6, 2, 6, 2, 2, 2, 6,
6, 6, 5, 6, 1, 6, 2, 2, 5, 2, 6, 3, 4, 6, 2, 2, 2, 6, 6, 2, 1, 1,
2, 6, 6, 2, 5, 6, 6, 5, 1, 6, 6, 2, 6, 6, 2, 6, 2, 1, 6, 6, 6, 1,
4, 1, 4, 2, 1, 6, 4, 2, 2, 6, 6, 4, 1, 4, 6, 2, 6, 6, 6, 6, 2, 6,
6, 6, 2, 6, 4, 6, 6, 2, 2, 1, 6, 6, 6, 6, 6, 6, 2, 2, 6, 2, 1, 6,
6, 6, 5, 6, 6, 1, 4, 2, 6, 1, 6, 6, 2, 6, 2, 6, 6, 2, 6, 6, 6, 2,
6, 6, 5, 6, 6, 1, 4, 6, 6, 2, 2, 6, 1, 1, 6, 6, 6, 4, 6, 6, 6, 4,
6, 6, 1])
```

In [36]:
```
one = OneHotEncoder()
data_label_hot = one.fit_transform(data_frame)
```

In [37]:
```
data_label_hot
```

Out[37]:
```
<311x7 sparse matrix of type '<class 'numpy.float64'>'
        with 311 stored elements in Compressed Sparse Row format>
```

In [38]:
```
data_label_hot.todense()[0:10]
```

Out[38]:
```
matrix([[0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 1., 0., 0.],
        [0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 0., 0., 1.],
        [0., 0., 0., 0., 1., 0., 0.],
        [0., 0., 1., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 1.]])
```

In [39]:
```
data_frame.head(10)
```

Out[39]:

	RaceDesc
0	White
1	NA
2	White
3	White
4	White
5	White
6	White
7	NA
8	Black or African American
9	White

С помощью Pands

In [41]:
```
pd.get_dummies(data_frame).head()
```

Out[41]:

	RaceDesc_American Indian or Alaska Native	RaceDesc_Asian	RaceDesc_Black or African American	RaceDesc_Hispanic	RaceDesc_NA	RaceDesc_ or more
0	0	0	0	0	0	
1	0	0	0	0	1	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	

In [42]:
```
pd.get_dummies(data_frame, dummy_na=True).head()
```

Out[42]:

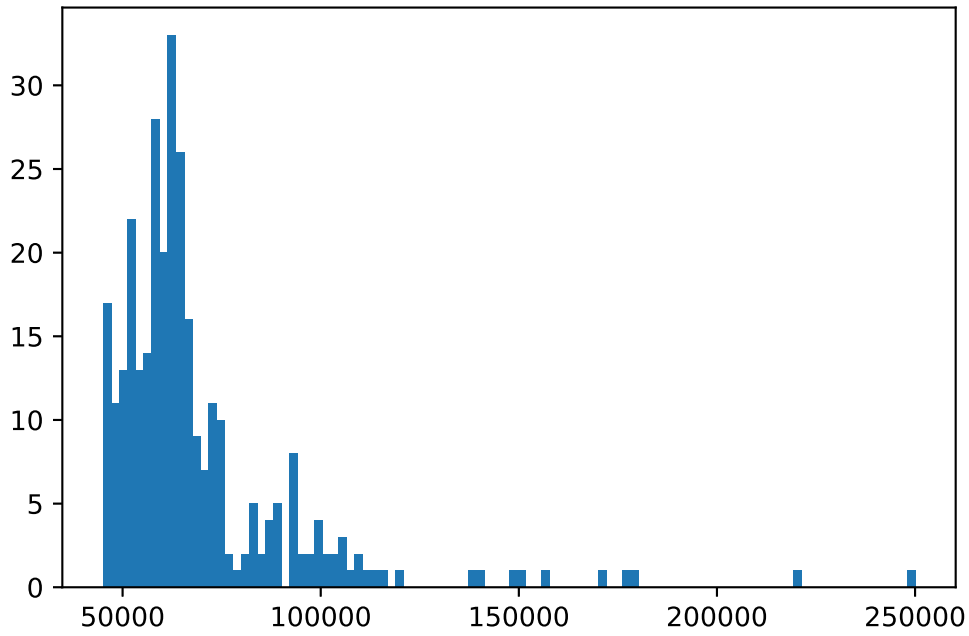
	RaceDesc_American Indian or Alaska Native	RaceDesc_Asian	RaceDesc_Black or African American	RaceDesc_Hispanic	RaceDesc_NA	RaceDesc_ or more
0	0	0	0	0	0	
1	0	0	0	0	1	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	

Маштабирование данных

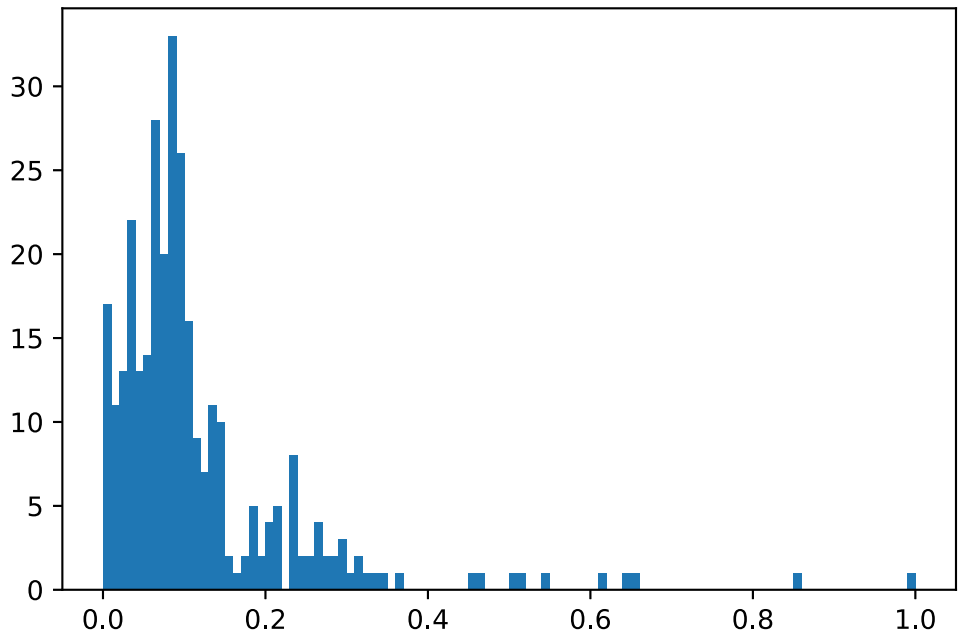
In [44]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

In [45]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['Salary']])
```

In [46]:
```
plt.hist(data[['Salary']], 100)
plt.show()
```



In [47]:
```
plt.hist(sc1_data, 100)
plt.show()
```



In [ ]: