In [ ]:

```

```

About Dataset

Hello My name is Ben Roshan D, doing MBA in Business Analytics at Jain University Bangalore . We have practical sessions in Python,R as subjects. Faculties provide us with such data sets to work on with it, So here is one of the data set which our class worked on

- What is in it?

This data set consists of Placement data of students in a XYZ campus. It includes secondary and higher secondary school percentage and specialization. It also includes degree specialization, type and Work experience and salary offers to the placed students

- Acknowledgement

I would like to thank Dr. Dhimant Ganatara, Professor Jain University for helping the students by providing this data for us to train R programming

- Questions

1.Which factor influenced a candidate in getting placed?

2.Does percentage matters for one to get placed?

3.Which degree specialization is much demanded by corporate?

4.Play with the data conducting all statistical tests.

In [1]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```python
# display all the columns of dataframe
pd.pandas.set_option("display.max_columns",None)
```

In [3]:

```python
# import the dataset
data=pd.read_csv(r"C:\Users\Mukul\Downloads\Placement Requirement Dataset Classification
```

In [4]:

```
# top 5 rows
data.head()
```

Out[4]:

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No
3	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No

In [5]:

```
# last 5 rows
data.tail()
```

Out[5]:

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	worke
210	211	M	80.6	Others	82.0	Others	Commerce	77.6	Comm&Mgmt	N
211	212	M	58.0	Others	60.0	Others	Science	72.0	Sci&Tech	N
212	213	M	67.0	Others	67.0	Others	Commerce	73.0	Comm&Mgmt	Y
213	214	F	74.0	Others	66.0	Others	Commerce	58.0	Comm&Mgmt	N
214	215	M	62.0	Central	58.0	Others	Science	53.0	Comm&Mgmt	N

In [6]:

```
# check the columns
data.columns
```

Out[6]:

```
Index(['sl_no', 'gender', 'ssc_p', 'ssc_b', 'hsc_p', 'hsc_b', 'hsc_s',
       'degree_p', 'degree_t', 'workex', 'etest_p', 'specialisation', 'mba
_p',
       'status', 'salary'],
      dtype='object')
```

Description of the column labels in this dataset:

sl_no = Serial Number

gender = Male - 'M' and Female - 'F'

ssc_p = Secondary Education Percentage (Middle School)

ssc_b = Board of Education (middle)

hsc_p = Higher Secondary School Percentage (Senior)

hsc_b = Board of Education (senior)

hsc_s = Specialization in Higher Secondary Education

degree_p = Degree Percentage

degree_t = Degree type- Field of degree education

workex = Work Experience

etest_p = Employability test percentage (conducted by college)

specialisation = Post Graduation(MBA)- Specialization

mba_p = MBA percentage

status = Status of placement- Placed/Not placed

salary = Salary offered by corporate to candidates

Change the columns name

In [7]:

```
data.columns
```

Out[7]:

```
Index(['sl_no', 'gender', 'ssc_p', 'ssc_b', 'hsc_p', 'hsc_b', 'hsc_s',
       'degree_p', 'degree_t', 'workex', 'etest_p', 'specialisation', 'mba
_p',
       'status', 'salary'],
      dtype='object')
```

In [8]:

```
data.rename(columns={'ssc_p':"Secondary Education Percentage (middle school)",'ssc_b':"B
                    'hsc_b':"Board of Education (senior)",'hsc_s':"Specialization in Hig
                    'degree_t':"Degree type- Field of degree education",'workex':"Work E
                    'mba_p':"MBA Percentage",'status':"Status of placement"},inplace=Tru
```

In [9]:

```
data.columns
```

Out[9]:

```
Index(['sl_no', 'gender', 'Secondary Education Percentage (middle schoo
l)',
       'Board of Education (middle)',
       'Higher Secondary School Percentage (senior)',
       'Board of Education (senior)',
       'Specialization in Higher Secondary Education', 'Degree Percentag
e',
       'Degree type- Field of degree education', 'Work Experience',
       'Employability test percentage (conducted by college)',
       'specialisation', 'MBA Percentage', 'Status of placement', 'salar
y'],
      dtype='object')
```

In [10]:

```
len(data.columns)
```

Out[10]:

```
15
```

In [11]:

```
# check the shape of  dataset
data.shape
```

Out[11]:

```
(215, 15)
```

In [12]:

```
# check the information of dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 15 columns):
 #   Column                                               Non-Null Count  
Dtype  
---  ------                                               --------------  
-----  
 0   sl_no                                                215 non-null    
int64  
 1   gender                                               215 non-null    
object 
 2   Secondary Education Percentage (middle school)       215 non-null    
float64
 3   Board of Education (middle)                          215 non-null    
object 
 4   Higher Secondary School Percentage (senior)          215 non-null    
float64
 5   Board of Education (senior)                          215 non-null    
object 
 6   Specialization in Higher Secondary Education         215 non-null    
object 
 7   Degree Percentage                                    215 non-null    
float64
 8   Degree type- Field of degree education               215 non-null    
object 
 9   Work Experience                                      215 non-null    
object 
 10  Employability test percentage (conducted by college)  215 non-null    
float64
 11  specialisation                                       215 non-null    
object 
 12  MBA Percentage                                       215 non-null    
float64
 13  Status of placement                                  215 non-null    
object 
 14  salary                                               148 non-null    
float64
dtypes: float64(6), int64(1), object(8)
memory usage: 25.3+ KB
```

In [13]:

```
# drop the sl_no columns
data.drop(['sl_no'],axis=1,inplace=True)
```

In [14]:

```
# check the missing value
data.isnull().sum()
```

Out[14]:

```
gender                                                 0
Secondary Education Percentage (middle school)         0
Board of Education (middle)                            0
Higher Secondary School Percentage (senior)            0
Board of Education (senior)                            0
Specialization in Higher Secondary Education           0
Degree Percentage                                      0
Degree type- Field of degree education                 0
Work Experience                                        0
Employability test percentage (conducted by college)   0
specialisation                                         0
MBA Percentage                                         0
Status of placement                                    0
salary                                                67
dtype: int64
```

- it is indicated the 67 missing value in salary columns

In [15]:

```
# check the % of missing values
data.isnull().mean()*100
```

Out[15]:

```
gender                                                 0.000000
Secondary Education Percentage (middle school)         0.000000
Board of Education (middle)                            0.000000
Higher Secondary School Percentage (senior)            0.000000
Board of Education (senior)                            0.000000
Specialization in Higher Secondary Education           0.000000
Degree Percentage                                      0.000000
Degree type- Field of degree education                 0.000000
Work Experience                                        0.000000
Employability test percentage (conducted by college)   0.000000
specialisation                                         0.000000
MBA Percentage                                         0.000000
Status of placement                                    0.000000
salary                                                31.162791
dtype: float64
```

- it is indiacted the 31.16 % missing value in salary columns

In [ ]:

```

```

In [16]:

```
# check the unique value
data.nunique()
```

Out[16]:

```
gender                                                   2
Secondary Education Percentage (middle school)         103
Board of Education (middle)                              2
Higher Secondary School Percentage (senior)             97
Board of Education (senior)                              2
Specialization in Higher Secondary Education             3
Degree Percentage                                       89
Degree type- Field of degree education                   3
Work Experience                                          2
Employability test percentage (conducted by college)   100
specialisation                                           2
MBA Percentage                                         205
Status of placement                                      2
salary                                                  45
dtype: int64
```

In [17]:

```
#  check the duplicated values
data.duplicated().sum()
```

Out[17]:

```
0
```

In [18]:

```
# check the memory usage
data.memory_usage()
```

Out[18]:

```
Index                                                    128
gender                                                  1720
Secondary Education Percentage (middle school)          1720
Board of Education (middle)                             1720
Higher Secondary School Percentage (senior)             1720
Board of Education (senior)                             1720
Specialization in Higher Secondary Education            1720
Degree Percentage                                       1720
Degree type- Field of degree education                  1720
Work Experience                                         1720
Employability test percentage (conducted by college)    1720
specialisation                                          1720
MBA Percentage                                          1720
Status of placement                                     1720
salary                                                  1720
dtype: int64
```

In [19]:

```
# Getting the count of each category from data
for feature in data.columns:
    print(data[feature].value_counts())
```

```
M    139
F     76
Name: gender, dtype: int64
62.00    11
63.00    10
67.00     9
52.00     9
73.00     9
         ..
69.70     1
80.92     1
83.00     1
86.50     1
80.60     1
Name: Secondary Education Percentage (middle school), Length: 103, dtyp
e: int64
Central    116
Others      99
Name: Board of Education (middle), dtype: int64
```

In [20]:

```
# Getting the count of each categories from unique data
for feature in data.columns:
    print(data[feature].unique())
```

```
['M' 'F']
[67.   79.33 65.   56.   85.8  55.   46.   82.   73.   58.   69.6  47.
 77.   62.   63.   60.   79.   69.8  77.4  76.5  52.58 71.   76.76 64.
 61.   87.   69.   51.   81.   78.   74.   49.   76.   70.89 50.   75.2
 54.4  40.89 80.   60.4  68.   52.6  84.2  86.5  54.   83.   80.92 69.7
 75.   84.86 64.6  56.6  59.   66.5  84.   81.7  70.   83.84 59.6  66.
 85.   52.   60.23 70.5  45.   61.08 69.5  73.96 68.2  60.8  72.   80.4
 76.7  74.9  77.44 77.67 89.4  44.   75.4  53.   51.57 55.6  74.2  67.16
 63.3  67.9  48.   59.96 63.4  73.24 77.8  56.28 88.   78.5  61.8  65.2
 83.96 54.2  55.68 41.   83.33 43.   80.6 ]
['Others' 'Central']
[91.   78.33 68.   52.   73.6  49.8  49.2  64.   79.   70.   61.   68.4
 55.   87.   47.   75.   66.2  67.   66.   65.   76.   60.8  60.   97.7
 54.6  76.5  73.5  53.   81.   51.   78.   44.   58.   77.   63.16 39.
 73.   71.98 62.   37.   73.2  61.12 45.83 66.6  71.4  65.58 73.4  64.2
 74.   78.5  70.29 83.83 64.8  70.4  80.   90.9  63.   89.83 90.   57.
 69.   62.5  82.   72.   50.   54.   72.8  40.   66.8  59.   71.   89.7
 92.   56.   64.89 65.66 86.   58.66 60.5  74.66 69.4  49.   87.6  72.5
 42.16 67.2  50.83 97.   71.5  60.33 62.83 65.5  77.6  70.2  61.4  61.33
 42.  ]
['Others' 'Central']
['Commerce' 'Science' 'Arts']
[58.   77.48 64.   52.   73.3  67.25 79.   66.   72.   61.   60.   78.3
 65.   59.   50.   69.   65.6  70.   85.   72.23 64.74 78.86 50.2  67.5
 73.   66.4  81.   57.   80.   68.   68.4  56.2  53.   61.4  74.   72.11
 66.89 67.4  75.   67.   72.7  62.   71.   78.   71.72 70.2  77.5  71.93
 64.5  77.2  82.   50.8  54.   76.   63.   83.   66.6  64.6  69.6  69.3
 64.33 75.5  77.72 77.   69.5  73.43 70.67 71.25 56.   55.   84.   59.9
 60.9  57.5  77.25 63.35 61.26 64.27 64.2  62.8  64.21 59.79 54.38 69.2
 64.8  56.3  91.   56.87 77.6 ]
['Sci&Tech' 'Comm&Mgmt' 'Others']
['No' 'Yes']
[55.   86.5  75.   66.   96.8  74.28 67.   91.34 54.   62.   60.   68.
 76.   72.   50.48 50.   95.   55.53 92.   97.4  94.   73.35 77.   52.
 64.   50.89 88.   68.44 71.   58.   53.7  93.   65.   63.   89.   78.
 71.2  87.   80.   74.   57.6  61.6  59.   68.5  61.   89.69 68.92 68.71
 79.   70.   95.5  86.   84.27 69.   86.04 82.   84.   78.74 53.88 95.46
 93.91 56.39 57.5  85.   57.2  72.15 96.   97.   82.66 73.   55.67 80.4
 55.5  81.2  90.   74.4  55.6  56.   83.   57.   64.25 98.   56.15 93.4
 57.63 75.2  53.04 58.1  54.48 58.06 63.79 87.5  75.5  95.65 59.32 87.55
 61.28 88.56 92.66 91.  ]
['Mkt&HR' 'Mkt&Fin']
[58.8  66.28 57.8  59.43 55.5  51.58 53.29 62.14 61.29 52.21 60.85 63.7
 65.04 68.63 54.96 64.66 62.54 67.28 64.08 77.89 56.7  69.06 68.81 63.62
 74.01 65.33 57.55 57.69 64.15 51.29 58.32 62.21 72.78 62.77 62.74 51.45
 55.47 56.86 62.56 66.72 69.76 51.21 62.9  69.7  66.53 71.63 54.55 62.46
 56.11 62.98 62.65 65.49 71.04 65.56 52.71 66.88 63.59 57.99 56.66 57.24
 62.48 59.69 59.5  58.78 57.1  58.46 60.99 59.24 68.07 65.45 66.94 68.53
 59.75 67.2  67.   64.27 57.65 59.42 67.99 62.35 70.2  60.44 66.69 62.
 76.18 57.03 59.08 64.36 62.36 68.03 62.79 59.47 55.41 54.97 62.16 64.44
 69.03 57.31 64.95 61.31 65.83 58.23 55.3  65.69 73.52 58.31 56.09 54.8
 60.64 53.94 63.08 55.01 60.5  70.85 67.05 70.48 64.34 58.81 71.49 71.
 61.26 73.33 68.2  58.4  76.26 68.55 60.78 53.49 60.98 67.13 65.63 61.58
 60.41 71.77 54.43 56.94 61.9  60.39 58.52 63.23 55.14 62.28 58.54 61.3
 58.87 65.25 53.2  65.99 52.72 55.03 61.87 60.59 72.29 62.72 66.06 66.46
 65.52 74.56 52.38 75.71 58.79 65.48 69.28 66.04 52.64 59.32 66.23 60.69
 57.9  70.81 72.14 56.6  60.02 59.81 61.82 57.29 71.43 62.93 64.86 56.13
 62.5  61.01 57.34 56.63 64.74 58.95 54.48 69.71 71.96 55.8  52.81 58.44
 60.11 58.3  67.69 56.81 53.39 71.55 62.92 56.49 74.49 53.62 69.72 60.23
 60.22]
['Placed' 'Not Placed']
```

```
[270000. 200000. 250000.     nan 425000. 252000. 231000. 260000. 218000.
 300000. 236000. 265000. 393000. 360000. 240000. 350000. 278000. 320000.
 411000. 287000. 204000. 450000. 216000. 220000. 268000. 275000. 336000.
 230000. 500000. 400000. 210000. 420000. 380000. 280000. 276000. 940000.
 225000. 233000. 690000. 340000. 255000. 285000. 290000. 650000. 264000.
 295000.]
```

Handling the Missing Value

In [21]:

```
data.isnull().sum()
```

Out[21]:

```
gender                                                    0
Secondary Education Percentage (middle school)            0
Board of Education (middle)                               0
Higher Secondary School Percentage (senior)               0
Board of Education (senior)                               0
Specialization in Higher Secondary Education              0
Degree Percentage                                         0
Degree type- Field of degree education                    0
Work Experience                                           0
Employability test percentage (conducted by college)      0
specialisation                                            0
MBA Percentage                                            0
Status of placement                                       0
salary                                                   67
dtype: int64
```

In [22]:

```
data["salary"].unique()
```

Out[22]:

```
array([270000., 200000., 250000.,     nan, 425000., 252000., 231000.,
       260000., 218000., 300000., 236000., 265000., 393000., 360000.,
       240000., 350000., 278000., 320000., 411000., 287000., 204000.,
       450000., 216000., 220000., 268000., 275000., 336000., 230000.,
       500000., 400000., 210000., 420000., 380000., 280000., 276000.,
       940000., 225000., 233000., 690000., 340000., 255000., 285000.,
       290000., 650000., 264000., 295000.])
```

In [23]:

```
data['salary'].value_counts()
```

Out[23]:

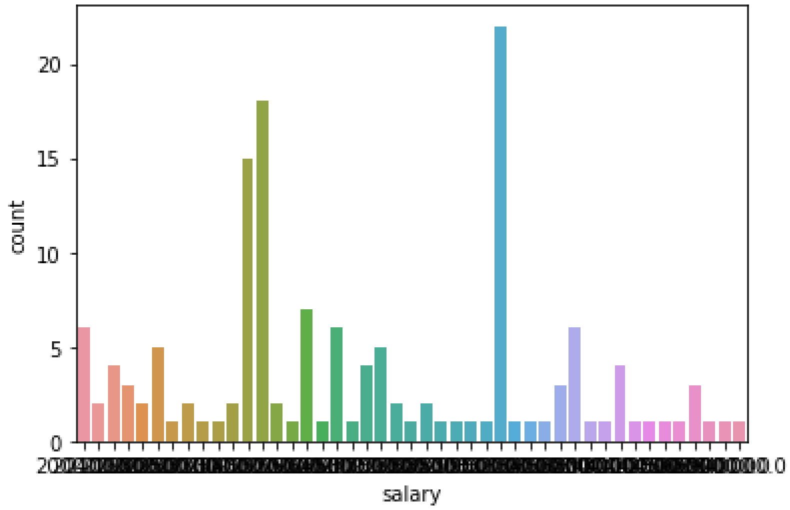
```
300000.0    22
250000.0    18
240000.0    15
260000.0     7
360000.0     6
200000.0     6
265000.0     6
220000.0     5
275000.0     5
210000.0     4
400000.0     4
270000.0     4
216000.0     3
350000.0     3
500000.0     3
252000.0     2
236000.0     2
230000.0     2
280000.0     2
218000.0     2
204000.0     2
276000.0     2
255000.0     1
285000.0     1
340000.0     1
690000.0     1
233000.0     1
290000.0     1
650000.0     1
264000.0     1
225000.0     1
940000.0     1
393000.0     1
380000.0     1
420000.0     1
425000.0     1
336000.0     1
231000.0     1
268000.0     1
450000.0     1
287000.0     1
411000.0     1
320000.0     1
278000.0     1
295000.0     1
Name: salary, dtype: int64
```

In [24]:

```
sns.countplot(data['salary'])
```

Out[24]:

```
<AxesSubplot:xlabel='salary', ylabel='count'>
```



In [25]:

```
data['salary'].dtypes
```

Out[25]:

```
dtype('float64')
```

- We are observerd the salary columns is missing value and it is indicated the float dtypes.
- Salary------> This feature are float dtypes. We decided the filling missing values by means()

In [26]:

```
data['salary'] = data['salary'].fillna(data['salary'].mean())
```

In [27]:

```
data.isnull().sum()
```

Out[27]:

```
gender                                                   0
Secondary Education Percentage (middle school)           0
Board of Education (middle)                              0
Higher Secondary School Percentage (senior)              0
Board of Education (senior)                              0
Specialization in Higher Secondary Education             0
Degree Percentage                                        0
Degree type- Field of degree education                   0
Work Experience                                          0
Employability test percentage (conducted by college)    0
specialisation                                           0
MBA Percentage                                           0
Status of placement                                      0
salary                                                   0
dtype: int64
```

- no missing values

Segregate the Numerical and Categorical columns

In [28]:

```
# Numerical Columns

num_columns = [feature for feature in data.columns if data[feature].dtypes != 'object']
print("Number of Numerical Feature ",  len(num_columns))
print(num_columns)
```

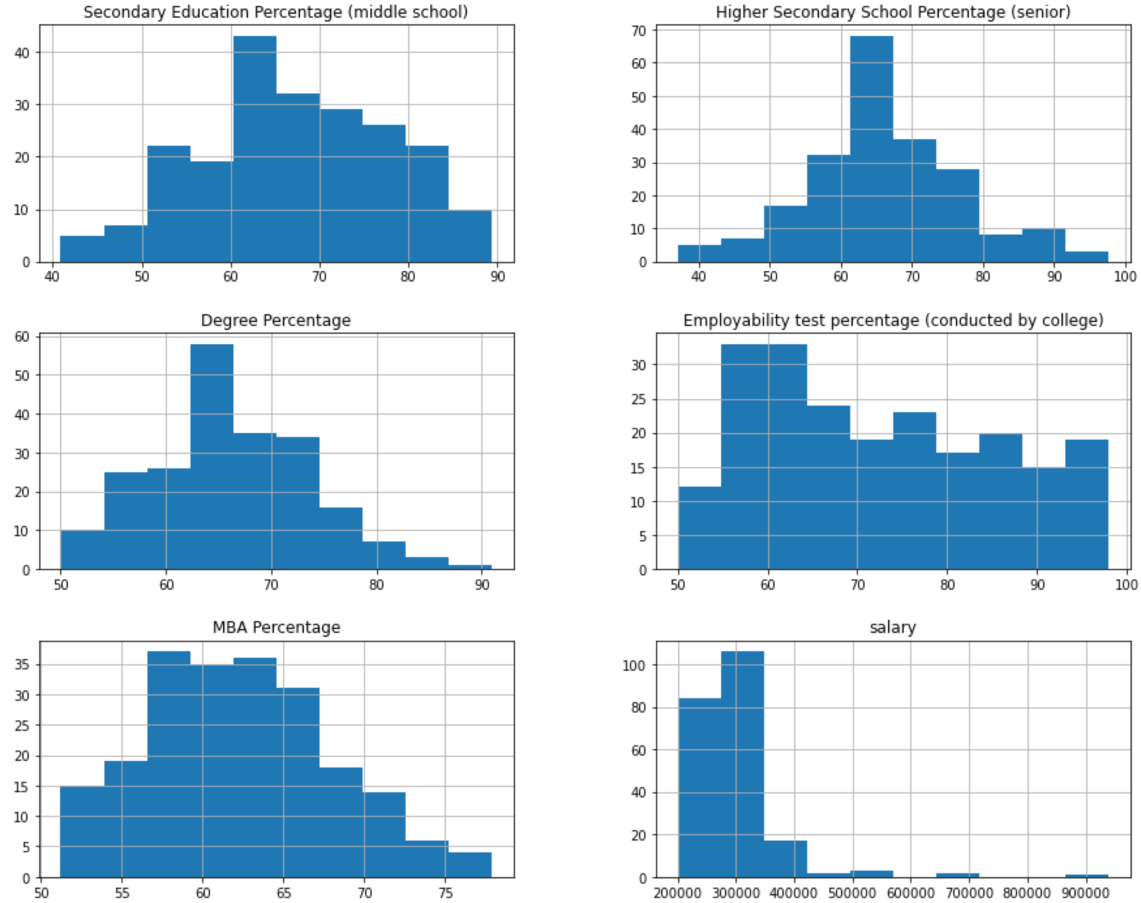
```
Number of Numerical Feature  6
['Secondary Education Percentage (middle school)', 'Higher Secondary Schoo
l Percentage (senior)', 'Degree Percentage', 'Employability test percentag
e (conducted by college)', 'MBA Percentage', 'salary']
```

In [29]:

```
data.hist(figsize=(15,12))
```

Out[29]:

```
array([[<AxesSubplot:title={'center':'Secondary Education Percentage (midd
le school)'}>,
        <AxesSubplot:title={'center':'Higher Secondary School Percentage
(senior)'}>],
       [<AxesSubplot:title={'center':'Degree Percentage'}>,
        <AxesSubplot:title={'center':'Employability test percentage (condu
cted by college)'}>],
       [<AxesSubplot:title={'center':'MBA Percentage'}>,
        <AxesSubplot:title={'center':'salary'}>]], dtype=object)
```



In [30]:

```
# Categorical Columns

cat_columns = [feature for feature in data.columns if data[feature].dtypes=='object']
print("Number of Categorical Columns" , len(cat_columns))
print(cat_columns)
```

```
Number of Categorical Columns 8
['gender', 'Board of Education (middle)', 'Board of Education (senior)',
'Specialization in Higher Secondary Education', 'Degree type- Field of deg
ree education', 'Work Experience', 'specialisation', 'Status of placemen
t']
```

In [31]:

```
# Fetching the Unique value of categorical data
for i in cat_columns:
    print(i , data[i].unique())
    print()
```

```
gender ['M' 'F']

Board of Education (middle) ['Others' 'Central']

Board of Education (senior) ['Others' 'Central']

Specialization in Higher Secondary Education ['Commerce' 'Science' 'Arts']

Degree type- Field of degree education ['Sci&Tech' 'Comm&Mgmt' 'Others']

Work Experience ['No' 'Yes']

specialisation ['Mkt&HR' 'Mkt&Fin']

Status of placement ['Placed' 'Not Placed']
```

In [32]:

```
# Fetch the only categorical columns
cat_df = data[cat_columns]
cat_df
```

Out[32]:

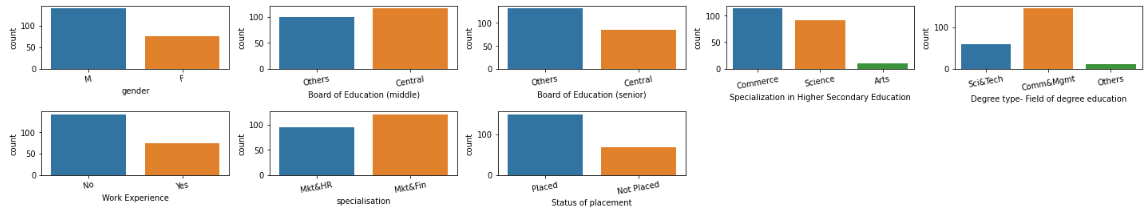
	gender	Board of Education (middle)	Board of Education (senior)	Specialization in Higher Secondary Education	Degree type- Field of degree education	Work Experience	specialisation
0	M	Others	Others	Commerce	Sci&Tech	No	Mkt&HR
1	M	Central	Others	Science	Sci&Tech	Yes	Mkt&Fin
2	M	Central	Central	Arts	Comm&Mgmt	No	Mkt&Fin
3	M	Central	Central	Science	Sci&Tech	No	Mkt&HR
4	M	Central	Central	Commerce	Comm&Mgmt	No	Mkt&Fin
...	...	...	...	...	...	...	...
210	M	Others	Others	Commerce	Comm&Mgmt	No	Mkt&Fin
211	M	Others	Others	Science	Sci&Tech	No	Mkt&Fin
212	M	Others	Others	Commerce	Comm&Mgmt	Yes	Mkt&Fin
213	F	Others	Others	Commerce	Comm&Mgmt	No	Mkt&HR
214	M	Central	Others	Science	Comm&Mgmt	No	Mkt&HR

215 rows × 8 columns

Visulization of Categorical Columns

In [33]:

```
plt.figure(figsize=(20,18))
cat_columns = ['gender','Board of Education (middle)','Board of Education (senior)','Spe
for i in range (0 , len(cat_columns)):
    plt.subplot(10,5,i+1)
    sns.countplot(x=data[cat_columns[i]])
    plt.xlabel(cat_columns[i])
    plt.xticks(rotation=10)
    plt.tight_layout()
```



In [34]:

```
data['specialisation'].value_counts()
```

Out[34]:

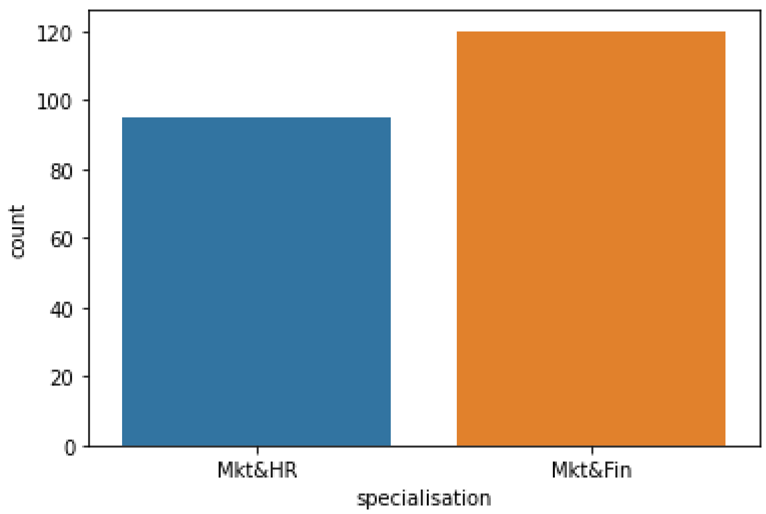
```
Mkt&Fin    120
Mkt&HR      95
Name: specialisation, dtype: int64
```

In [35]:

```
sns.countplot(data['specialisation'])
```

Out[35]:

<AxesSubplot:xlabel='specialisation', ylabel='count'>



In [36]:

```
data['Specialization in Higher Secondary Education'].value_counts()
```

Out[36]:

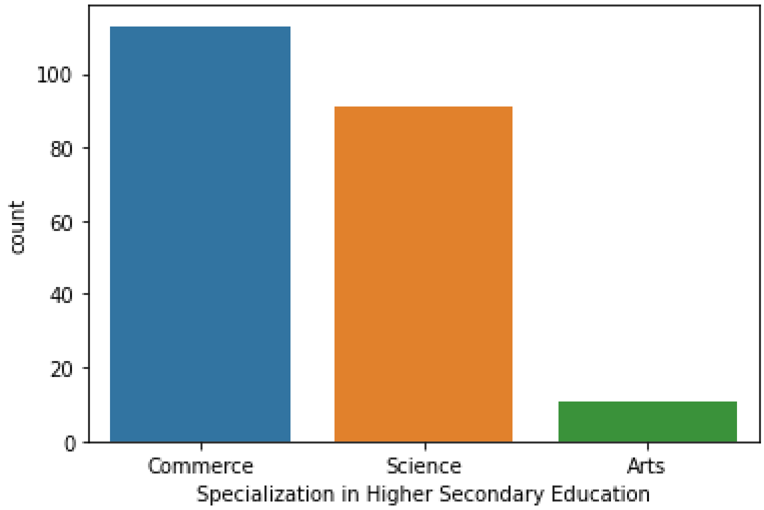
```
Commerce    113
Science      91
Arts         11
Name: Specialization in Higher Secondary Education, dtype: int64
```

In [37]:

```
sns.countplot(data['Specialization in Higher Secondary Education'])
```

Out[37]:

```
<AxesSubplot:xlabel='Specialization in Higher Secondary Education', ylabel
='count'>
```



- it is indicated the Art specialization is low category and Commerce specialization is Higher Category and Science specialization is more than intermediate category

In [38]:

```
data['Work Experience'].value_counts()
```

Out[38]:

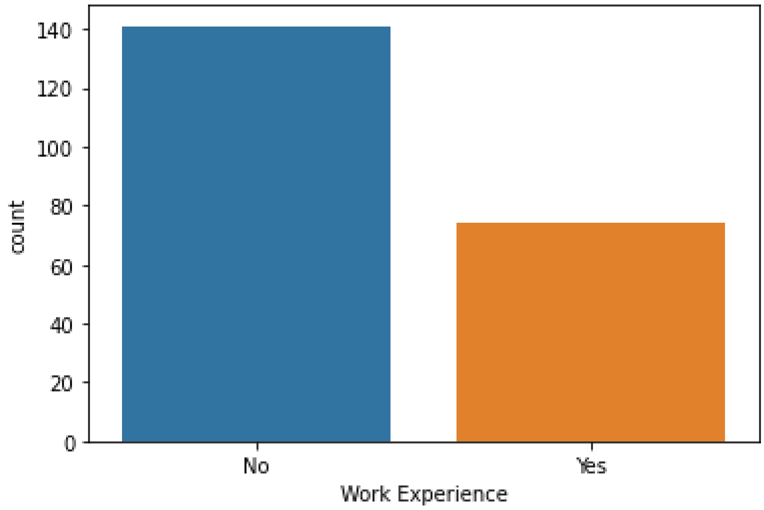
```
No     141
Yes     74
Name: Work Experience, dtype: int64
```

In [39]:

```
sns.countplot(data['Work Experience'])
```

Out[39]:

```
<AxesSubplot:xlabel='Work Experience', ylabel='count'>
```

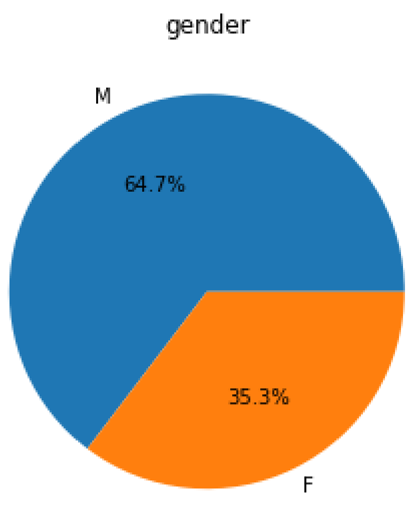


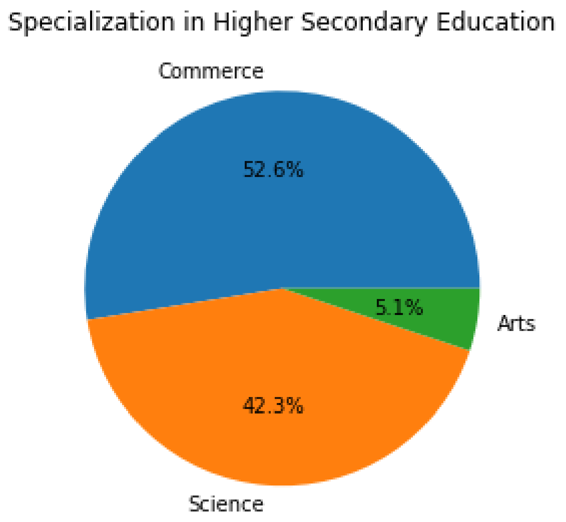
In [40]:

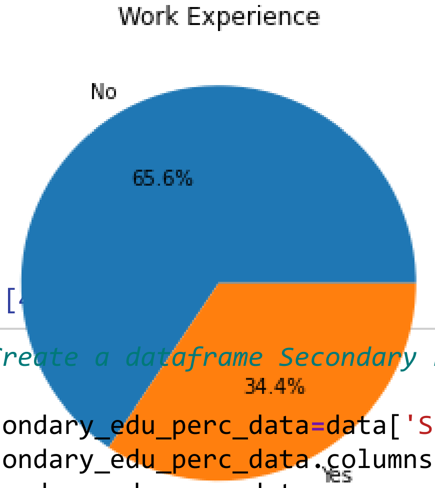
```
# gender  , # Secondary Eduaction Percentage, #Work Experience


# gender
plt.figure(figsize=(15,12))
plt.subplot(1,3,1)
labels = data['gender'].value_counts().index
size=data['gender'].value_counts()
explode = None
plt.pie(size,labels=labels,explode=explode,autopct='%1.1f%%')
plt.title('gender')

# Secondarey Education Percentage
plt.figure(figsize=(15,12))
plt.subplot(1,3,2)
labels = data['Specialization in Higher Secondary Education'].value_counts().index
size=data['Specialization in Higher Secondary Education'].value_counts()
explode=None
plt.pie(size,labels=labels,explode=explode,autopct='%1.1f%%')
plt.title('Specialization in Higher Secondary Education')

# Work Experience
plt.figure(figsize=(15,12))
plt.subplot(1,3,3)
labels=data['Work Experience'].value_counts().index
size=data['Work Experience'].value_counts()
explode = None
plt.pie(size,labels=labels , explode=explode,autopct='%1.1f%%')
plt.title('Work Experience')
```

Out[40]:

```
Text(0.5, 1.0, 'Work Experience')
```







In [4

```
# Create a dataframe Secondary Education Percentage and find out how many total student g
secondary_edu_perc_data=data['Secondary Education Percentage (middle school)'].value_cou
secondary_edu_perc_data.columns = ['Secondary Education Percentage (middle school)' , 'T
secondary_edu_perc_data
```

Out[41]:

	Secondary Education Percentage (middle school)	Total Student
0	62.00	11
1	63.00	10
2	67.00	9
3	52.00	9
4	73.00	9
...	...	...
98	69.70	1
99	80.92	1
100	83.00	1
101	86.50	1
102	80.60	1

103 rows × 2 columns

In [42]:

```
# check the which type of more dergee in gender category

data.groupby(['gender'])['Degree type- Field of degree education'].value_counts()
```

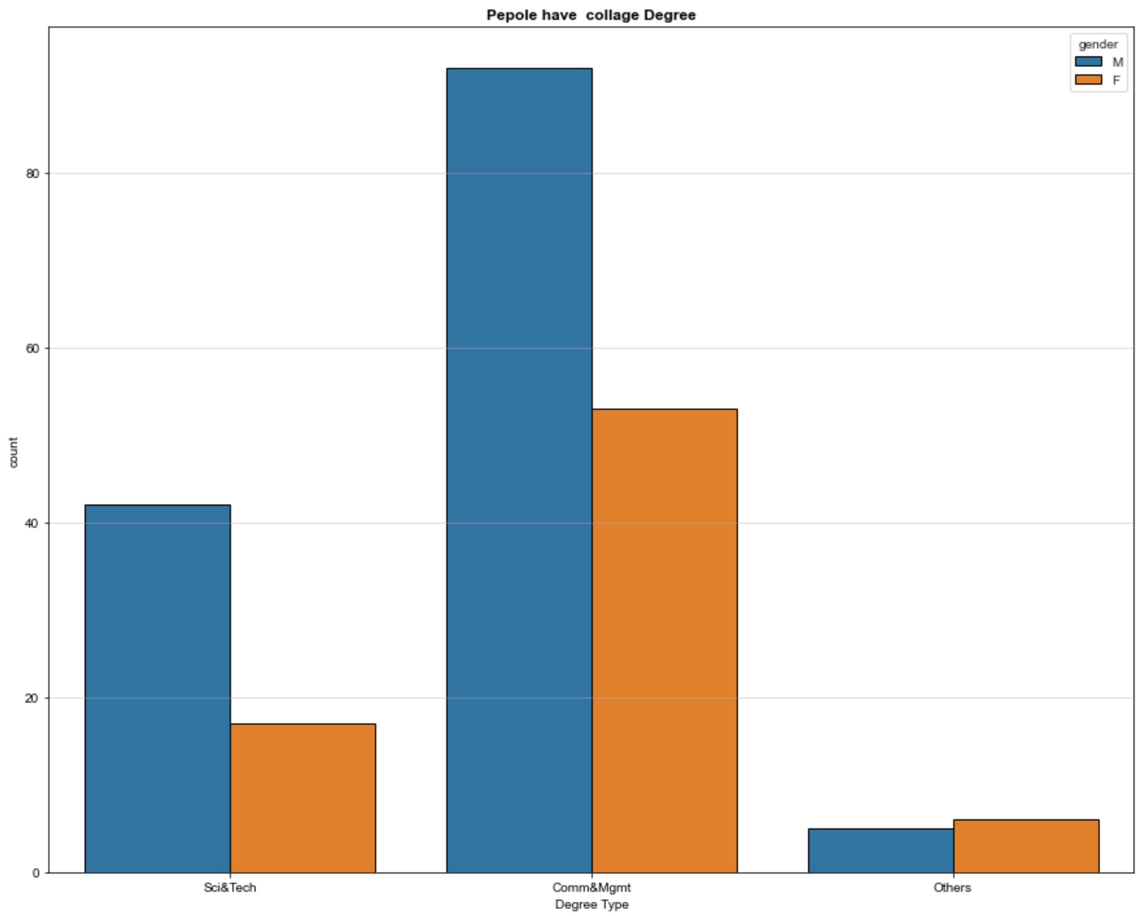
Out[42]:

```
gender  Degree type- Field of degree education
F       Comm&Mgmt                                 53
        Sci&Tech                                  17
        Others                                     6
M       Comm&Mgmt                                 92
        Sci&Tech                                  42
        Others                                     5
Name: Degree type- Field of degree education, dtype: int64
```

- it is indicated the Female have Commerce and Management 53 degree , Science and Technology 17 degree and other categegory is 6 degree
- Male have Commerce and Management 92 degree , Science and Technology 42 degree and other category is 6 degree

In [43]:

```
# plot barcahrt

plt.subplots(figsize=(15,12))
sns.set_style('whitegrid')
sns.countplot(x='Degree type- Field of degree education',hue='gender',data=data,ec='blac
plt.title('Pepole have  collage Degree' , weight='bold')
plt.xlabel('Degree Type')
plt.grid(alpha=0.5,axis='y')
plt.show()
```



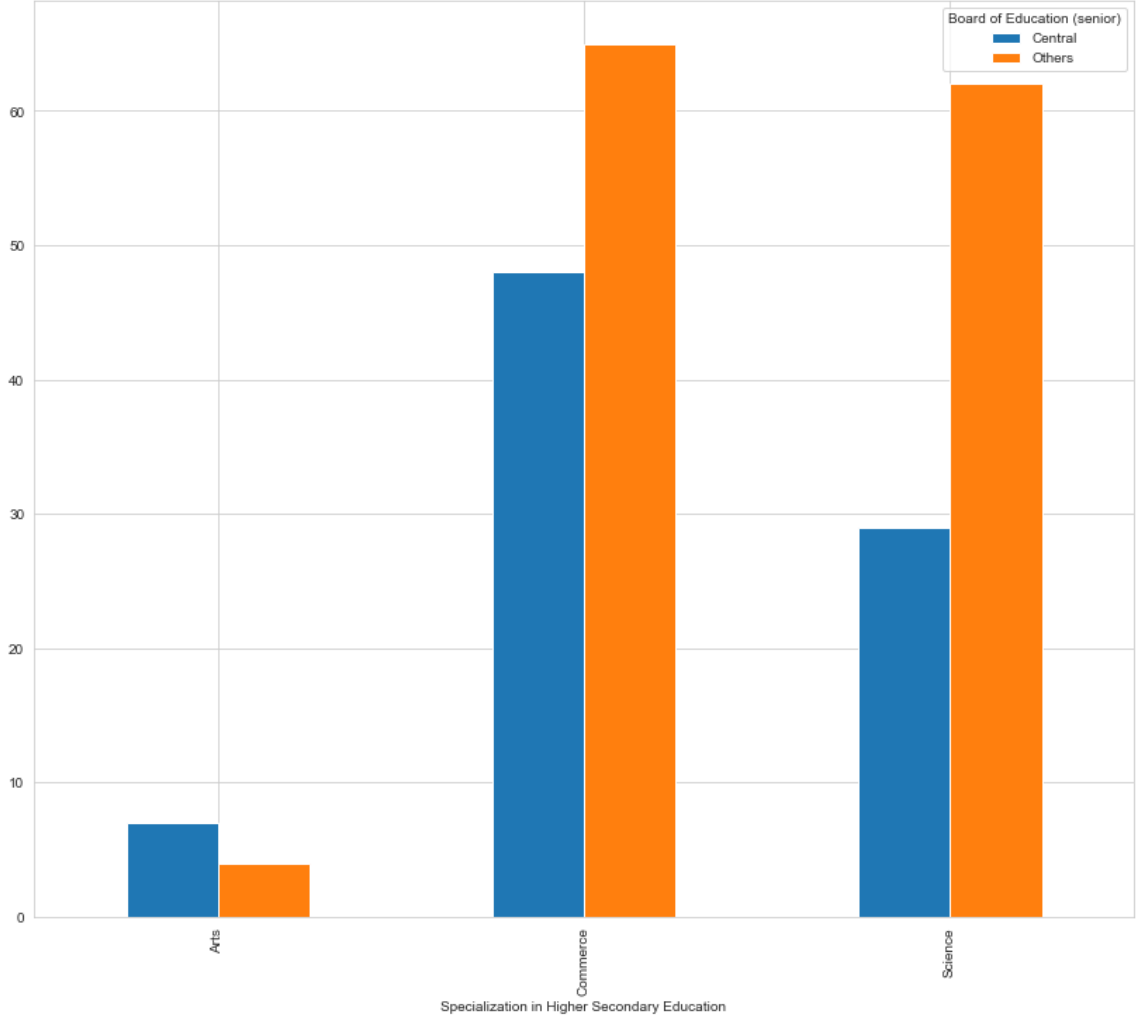
In [44]:

```
# check the Specialization in Higher Secondary Education

plt.figure(figsize=(30,20))
secon_edu=pd.crosstab(data['Specialization in Higher Secondary Education'],data['Board o
secon_edu.plot(kind='bar',figsize=(14,12))
```

Out[44]:

```
<AxesSubplot:xlabel='Specialization in Higher Secondary Education'>
```

```
<Figure size 2160x1440 with 0 Axes>
```



- we are observed that other Board of Education in high demand Commerce and Science specialization in Higher Secondary education and the other Board of education in low demand in Art Background Student
- Central Board of education high demand in Commerece,Science and Art Background Student

In [45]:

```
data.columns
```

Out[45]:

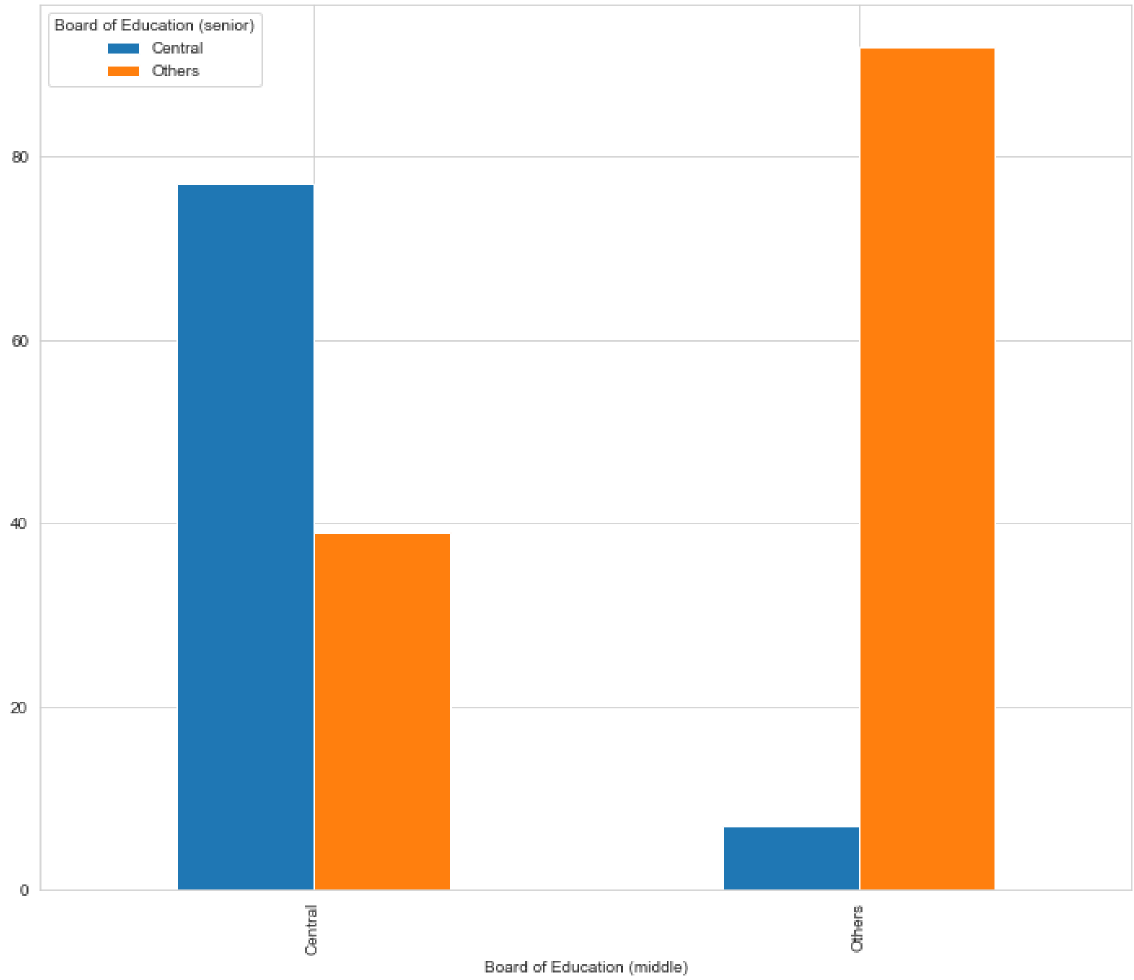
```
Index(['gender', 'Secondary Education Percentage (middle school)',
       'Board of Education (middle)',
       'Higher Secondary School Percentage (senior)',
       'Board of Education (senior)',
       'Specialization in Higher Secondary Education', 'Degree Percentag
e',
       'Degree type- Field of degree education', 'Work Experience',
       'Employability test percentage (conducted by college)',
       'specialisation', 'MBA Percentage', 'Status of placement', 'salar
y'],
      dtype='object')
```

In [46]:

```
board_edu = pd.crosstab(data['Board of Education (middle)'],data['Board of Education (se
board_edu.plot(kind='bar',figsize=(12,10))
```

Out[46]:

```
<AxesSubplot:xlabel='Board of Education (middle)'>
```



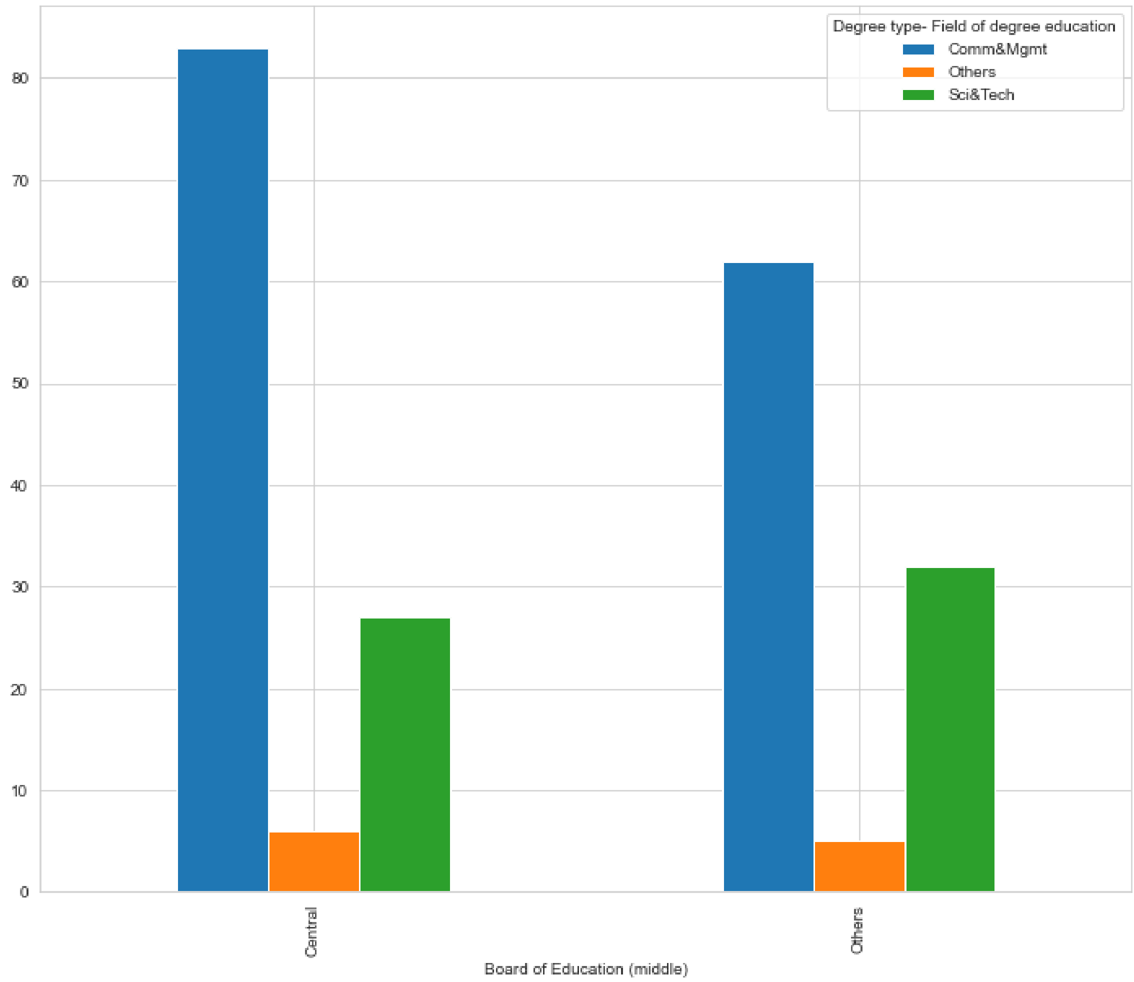
Which Board of Education have a more specilization

In [47]:

```
board_education = pd.crosstab(data['Board of Education (middle)'],data['Degree type- Fie
board_education.plot(kind='bar',figsize=(12,10))
```

Out[47]:

```
<AxesSubplot:xlabel='Board of Education (middle)'>
```



- we are observed that Central Borad of Education have a more degree type Commerce and Management Background of students , 28 % ofdegree type of Science and Technology background students, or 5 % of degree type of other background of students
- Other Board of Education have a more degree type Commerce and Management Background of students , 32 % of Degree types of Science and Technology background of student or 4% of degree type of other background of students

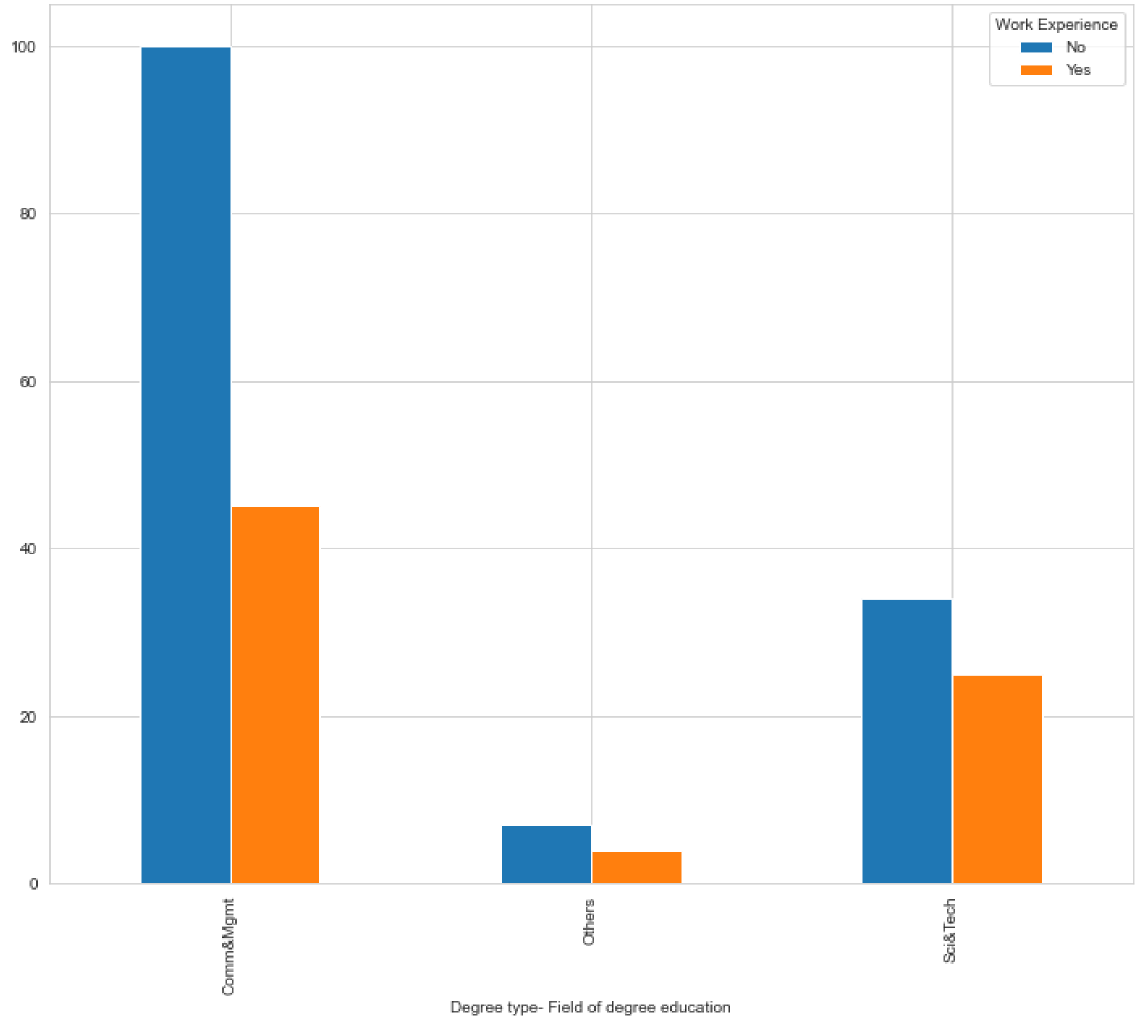
Which Degree type- Field of degree education have a more work experience?

In [48]:

```
degree_type = pd.crosstab(data['Degree type- Field of degree education'],data['Work Expe
degree_type.plot(kind='bar',figsize=(12,10))
```

Out[48]:

```
<AxesSubplot:xlabel='Degree type- Field of degree education'>
```



What is the Degree type- Field of degree education is each distribution of Gender Category

In [49]:

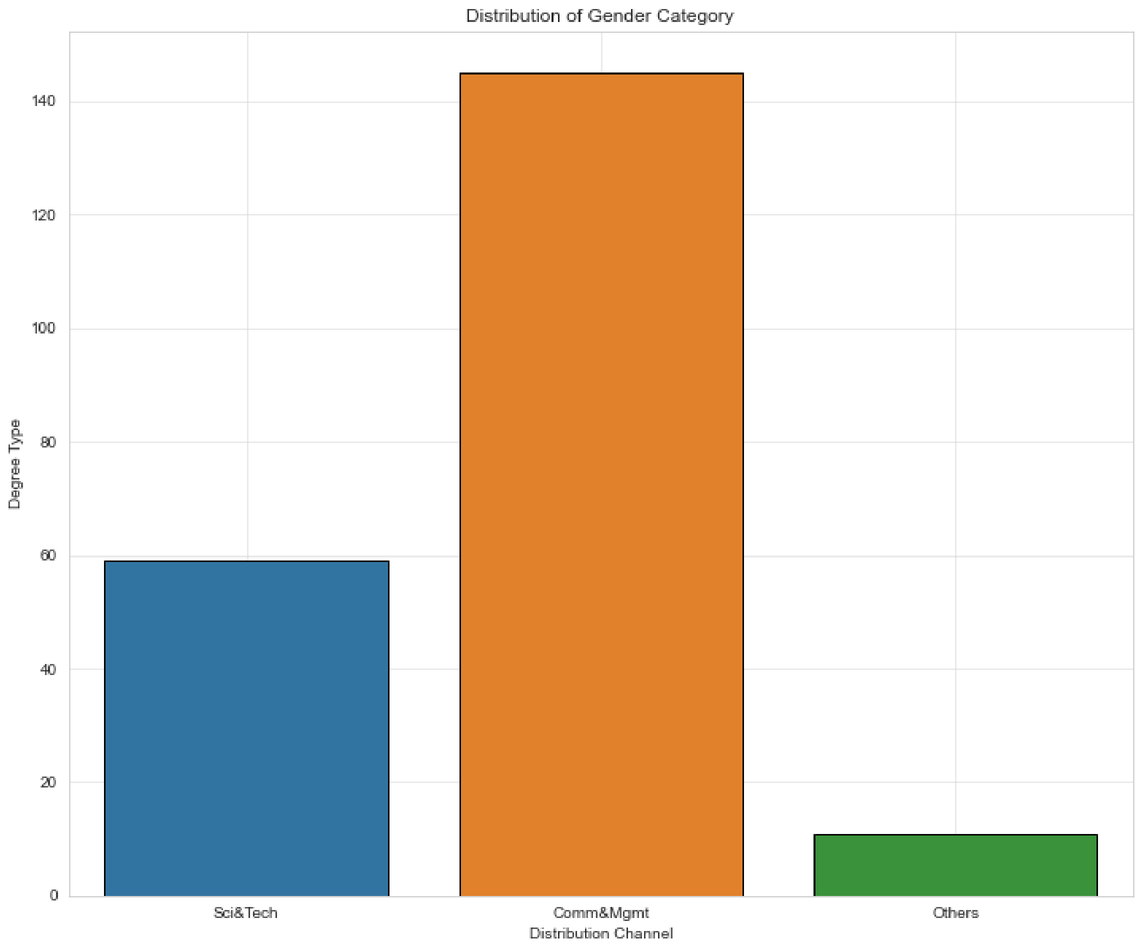
```
data.groupby(['Degree type- Field of degree education'])['gender'].value_counts()
```

Out[49]:

```
Degree type- Field of degree education  gender
Comm&Mgmt                               M         92
                                        F         53
Others                                  F          6
                                        M          5
Sci&Tech                                M         42
                                        F         17
Name: gender, dtype: int64
```

In [50]:

```
plt.subplots(figsize=(12,10))
sns.countplot(x='Degree type- Field of degree education',data=data , ec='black')
plt.title('Distribution of Gender Category')
plt.xlabel('Distribution Channel')
plt.ylabel('Degree Type')
plt.grid(alpha=0.5)
```



What is the Employability test percentage each distribution of Degree type field of degree education?

In [51]:

```
data.groupby(['Degree type- Field of degree education'])['Employability test percentage
```

Out[51]:

```
Degree type- Field of degree education  Employability test percentage (con
ducted by college)
Comm&Mgmt                               60.0
11
                                        67.0
5
                                        68.0
5
                                        89.0
5
                                        62.0
4

..
Sci&Tech                                95.0
1
                                        96.0
1
                                        97.0
1
                                        97.4
1
                                        98.0
1
Name: Employability test percentage (conducted by college), Length: 136, d
type: int64
```

- We are observed that Highest Employability Test Percentage is Science and Technology background of student 98 %

WHich background of student come under the top 10 percentage?

In [137]:

```
employability_test=data.groupby(['Employability test percentage (conducted by college)']
employability_test.sort_values(ascending=False)[0:10]
```

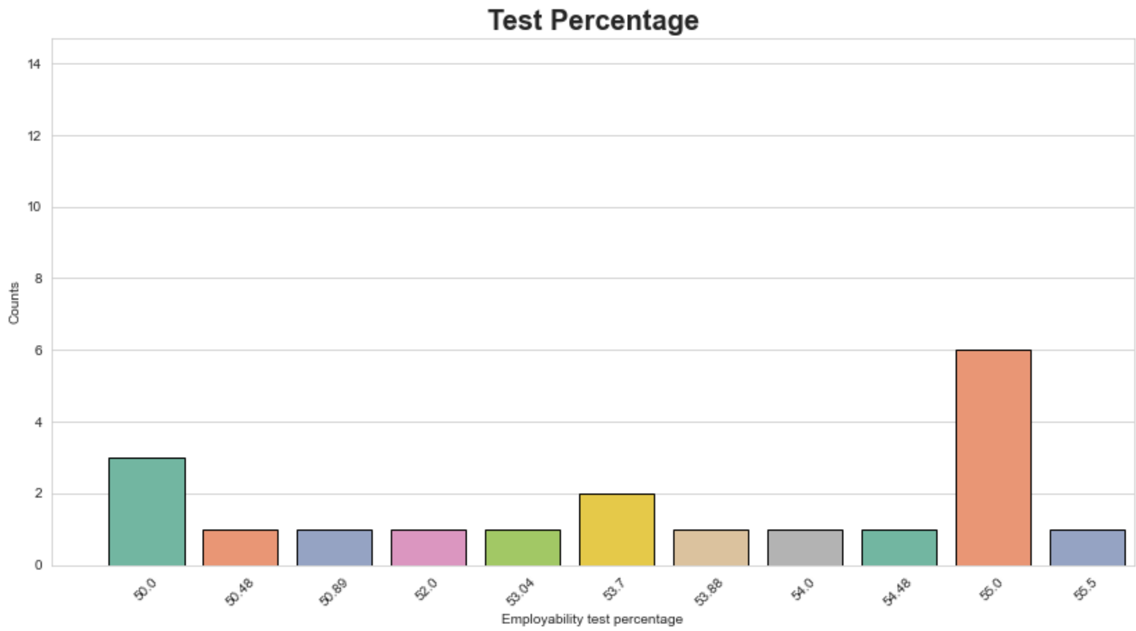
Out[137]:

```
Employability test percentage (conducted by college)  Degree type- Field o
f degree education
60.0                                                  0
11
67.0                                                  0
5
89.0                                                  0
5
68.0                                                  0
5
72.0                                                  0
4
62.0                                                  0
4
78.0                                                  0
4
84.0                                                  0
4
80.0                                                  0
4
75.0                                                  2
3
Name: Degree type- Field of degree education, dtype: int64
```

- it is indicated the 9 student is Commerce and Management background and 1 student is Science and Technology backgrounfd student

In [53]:

```
plt.subplots(figsize=(14,7))
sns.countplot(x='Employability test percentage (conducted by college)',data=data,ec='bla
plt.title('Test Percentage',weight='bold',fontsize=20)
plt.ylabel('Counts')
plt.xlabel('Employability test percentage')
plt.xticks(rotation=45)
plt.xlim(-1,10.5)
plt.show()
```



Which background of student is the top 10 highest percentage ?

In [54]:

```
employability_test.sort_values(ascending=False)[0:10]
```

Out[54]:

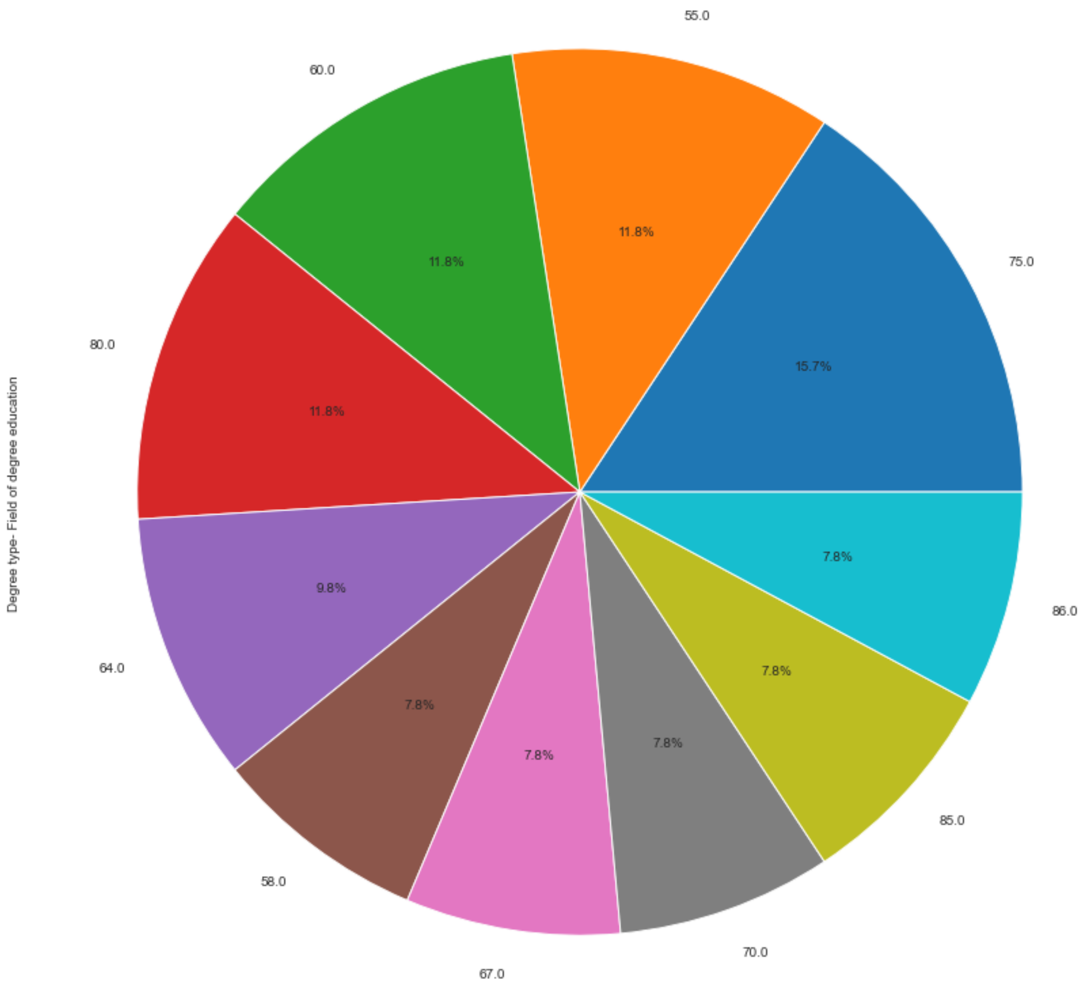
```
Employability test percentage (conducted by college)  Degree type- Field o
f degree education
60.0                                                  Comm&Mgmt
11
67.0                                                  Comm&Mgmt
5
89.0                                                  Comm&Mgmt
5
68.0                                                  Comm&Mgmt
5
72.0                                                  Comm&Mgmt
4
62.0                                                  Comm&Mgmt
4
78.0                                                  Comm&Mgmt
4
84.0                                                  Comm&Mgmt
4
80.0                                                  Comm&Mgmt
4
75.0                                                  Sci&Tech
3
Name: Degree type- Field of degree education, dtype: int64
```

In [136]:

```
data.groupby(['Employability test percentage (conducted by college)'])['Degree type- Fie
```

Out[136]:

```
<AxesSubplot:ylabel='Degree type- Field of degree education'>
```



In [ ]:

In [ ]:

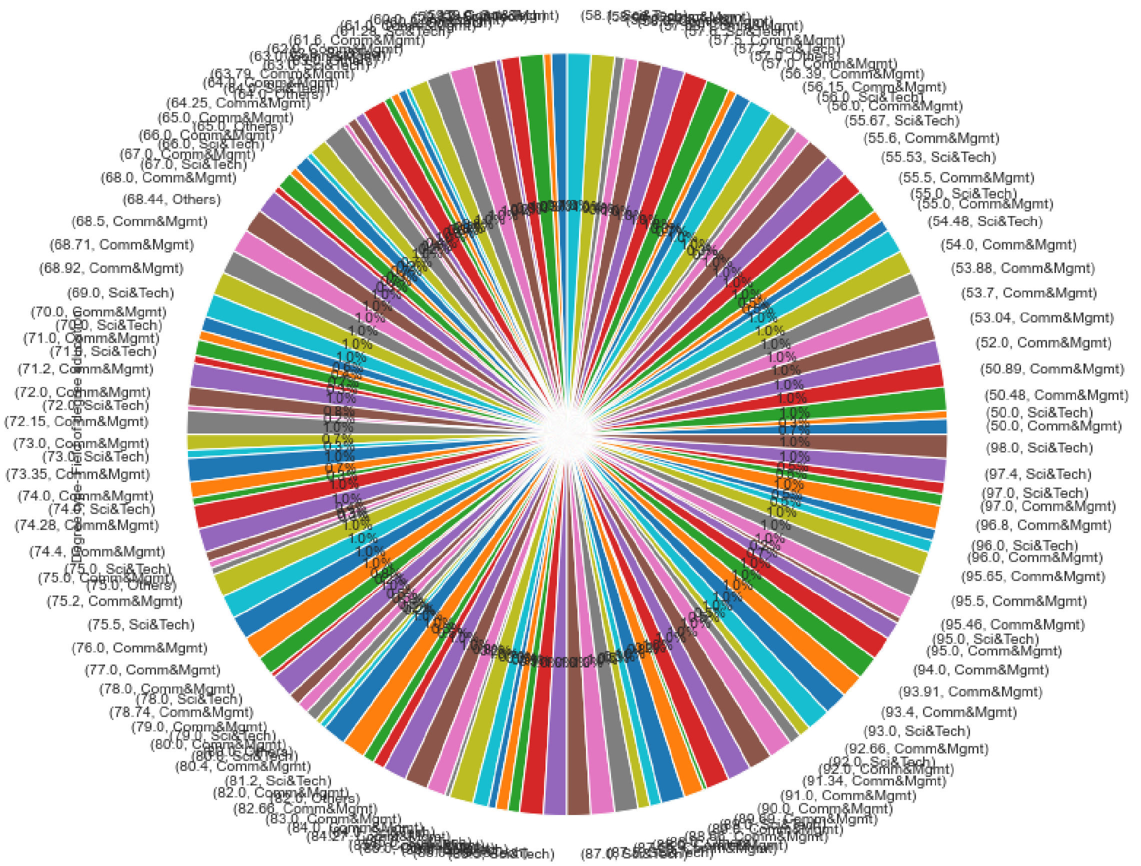
In [ ]:

In [55]:

```
data.groupby(['Employability test percentage (conducted by college)'])['Degree type- Fie
```

Out[55]:

<AxesSubplot:ylabel='Degree type- Field of degree education'>



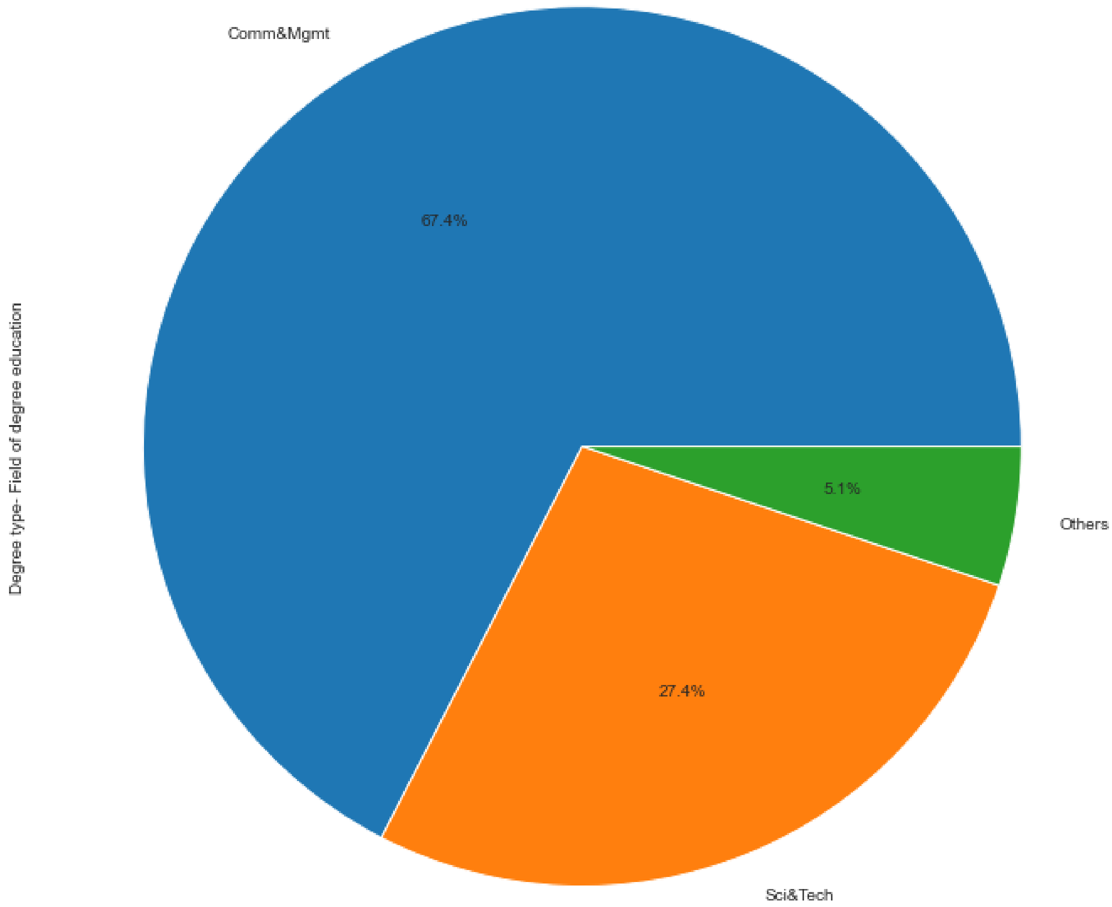
which is most popular degree type - field of degree education?

In [56]:

```
data['Degree type- Field of degree education'].value_counts().plot.pie(figsize=(15,12),a
```

Out[56]:

```
<AxesSubplot:ylabel='Degree type- Field of degree education'>
```



- it is indicated the most popular degree Commerce and Managemat Background of student 67.4% , Science and Technology background of student is 27.4% and other background of student is 5.1 %

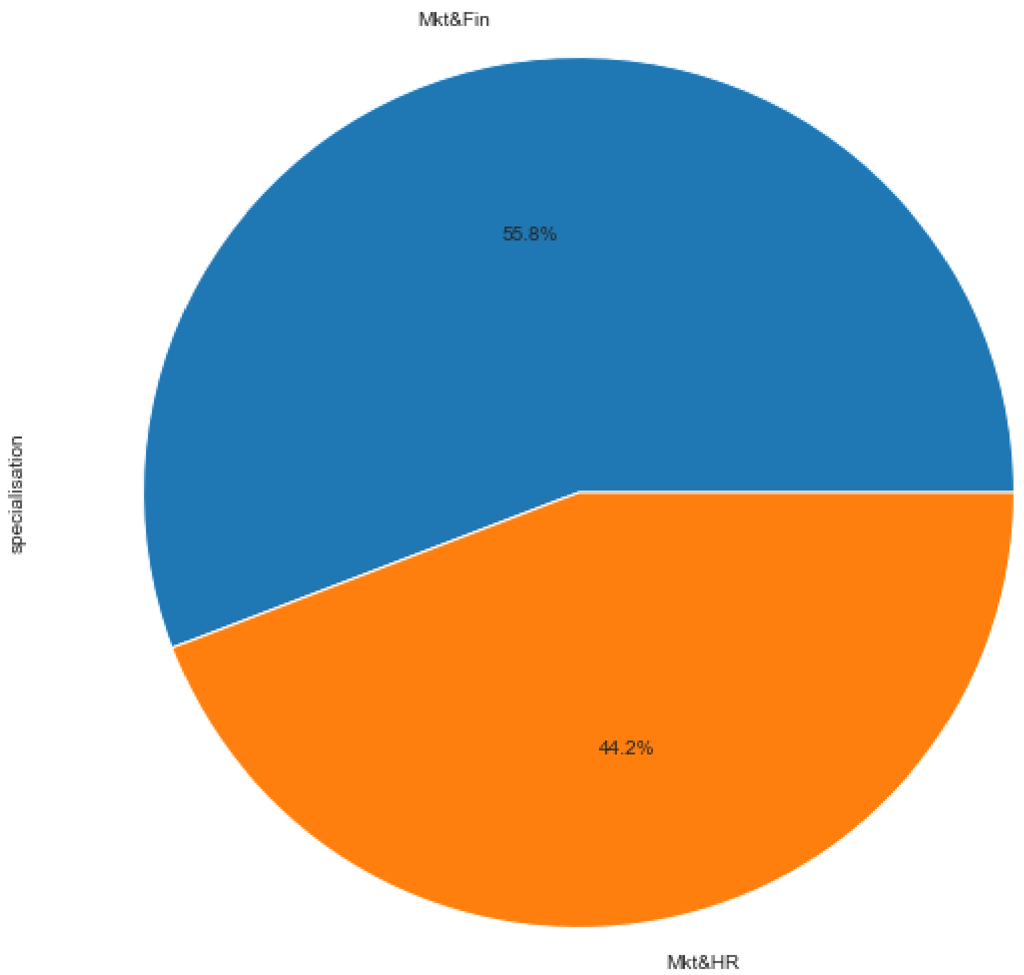
Which education is the most specialisation of students?

In [57]:

```
data['specialisation'].value_counts().plot.pie(figsize=(12,10),autopct="%1.1f%%")
```

Out[57]:

```
<AxesSubplot:ylabel='specialisation'>
```



Chcek the which specialisation is highest MBA Percentage

In [58]:

```
data.groupby(['MBA Percentage'])['specialisation'].value_counts()
```

Out[58]:

```
MBA Percentage  specialisation
51.21           Mkt&Fin          1
51.29           Mkt&Fin          1
51.45           Mkt&Fin          1
51.58           Mkt&Fin          1
52.21           Mkt&Fin          1
                                ..
74.56           Mkt&Fin          1
75.71           Mkt&Fin          1
76.18           Mkt&Fin          1
76.26           Mkt&HR           1
77.89           Mkt&Fin          1
Name: specialisation, Length: 208, dtype: int64
```

In [59]:

```
mba_perc=data.groupby(['MBA Percentage'])['specialisation'].value_counts()
mba_perc.sort_values(ascending=False)

mba_top_10_percentage=mba_perc.head(int(len(mba_perc)*0.1))
```

In [60]:

```
mba_top_10_percentage
```

Out[60]:

```
MBA Percentage  specialisation
51.21           Mkt&Fin          1
51.29           Mkt&Fin          1
51.45           Mkt&Fin          1
51.58           Mkt&Fin          1
52.21           Mkt&Fin          1
52.38           Mkt&HR           1
52.64           Mkt&HR           1
52.71           Mkt&HR           1
52.72           Mkt&HR           1
52.81           Mkt&Fin          1
53.20           Mkt&Fin          1
53.29           Mkt&Fin          1
53.39           Mkt&Fin          1
53.49           Mkt&HR           1
53.62           Mkt&Fin          1
53.94           Mkt&HR           1
54.43           Mkt&Fin          1
54.48           Mkt&Fin          1
54.55           Mkt&Fin          1
54.80           Mkt&HR           1
Name: specialisation, dtype: int64
```

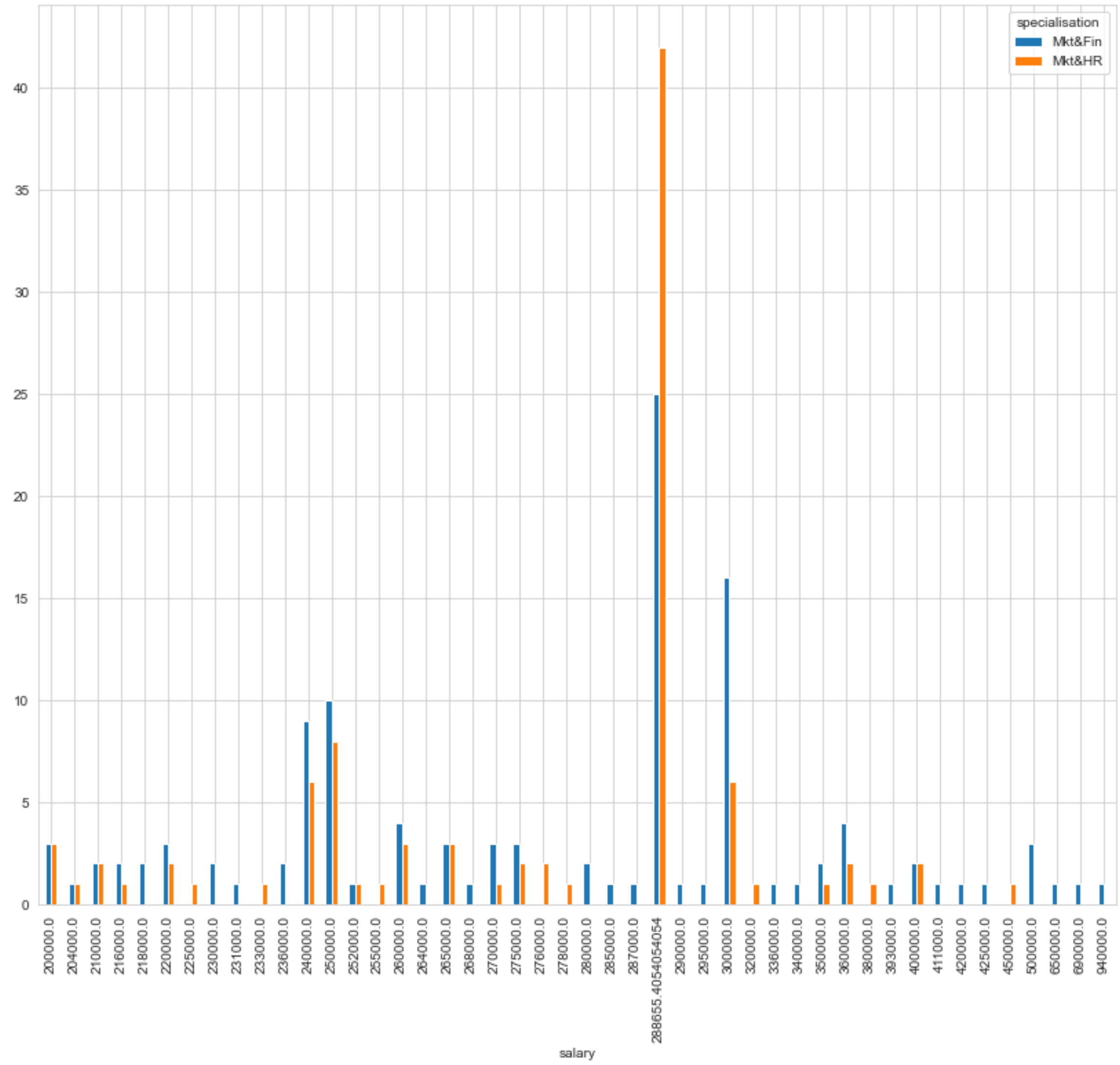
check the which specialistaion is higest salary

In [61]:

```
salary = pd.crosstab(data['salary'],data['specialisation'])
salary.plot(kind='bar', figsize=(14,12))
```

Out[61]:

```
<AxesSubplot:xlabel='salary'>
```



- we are observed that Marketing and HR sector is higest salary

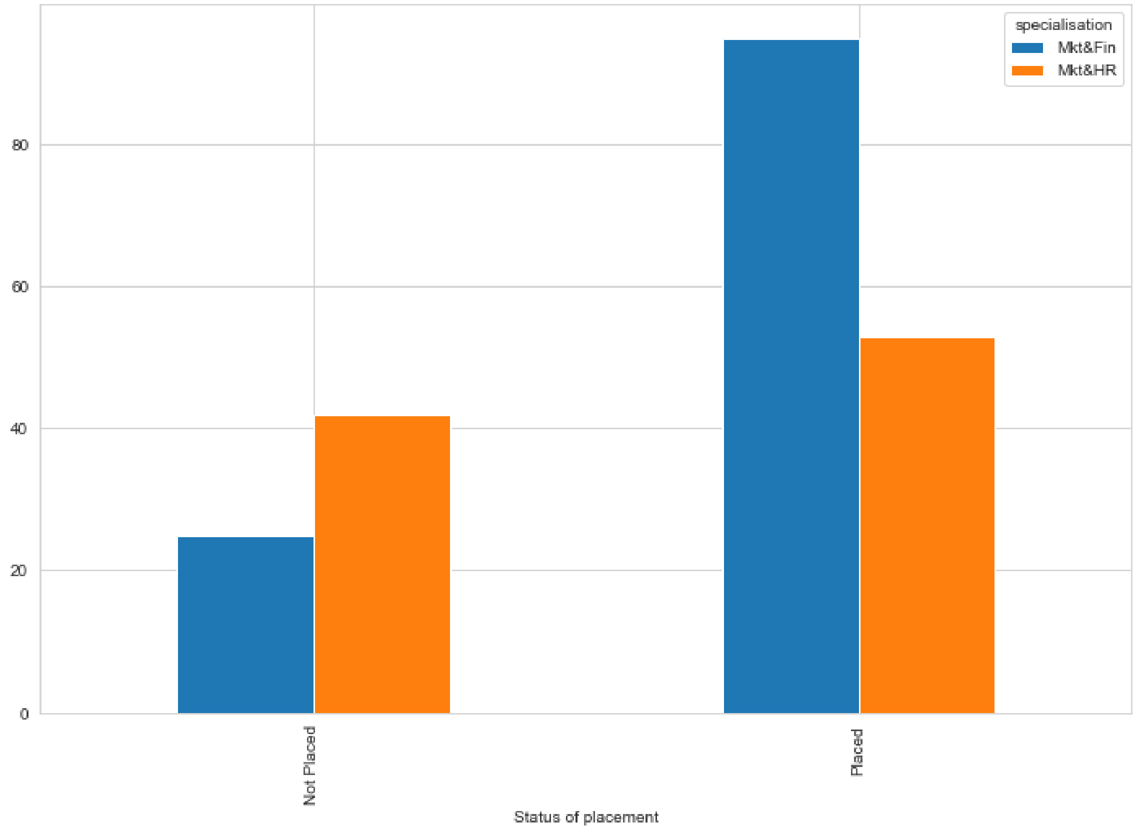
check the what is status of palcement of MBA Students

In [62]:

```
placement = pd.crosstab(data['Status of placement'],data['specialisation'])
placement.plot(kind='bar',figsize=(12,8))
```

Out[62]:

```
<AxesSubplot:xlabel='Status of placement'>
```



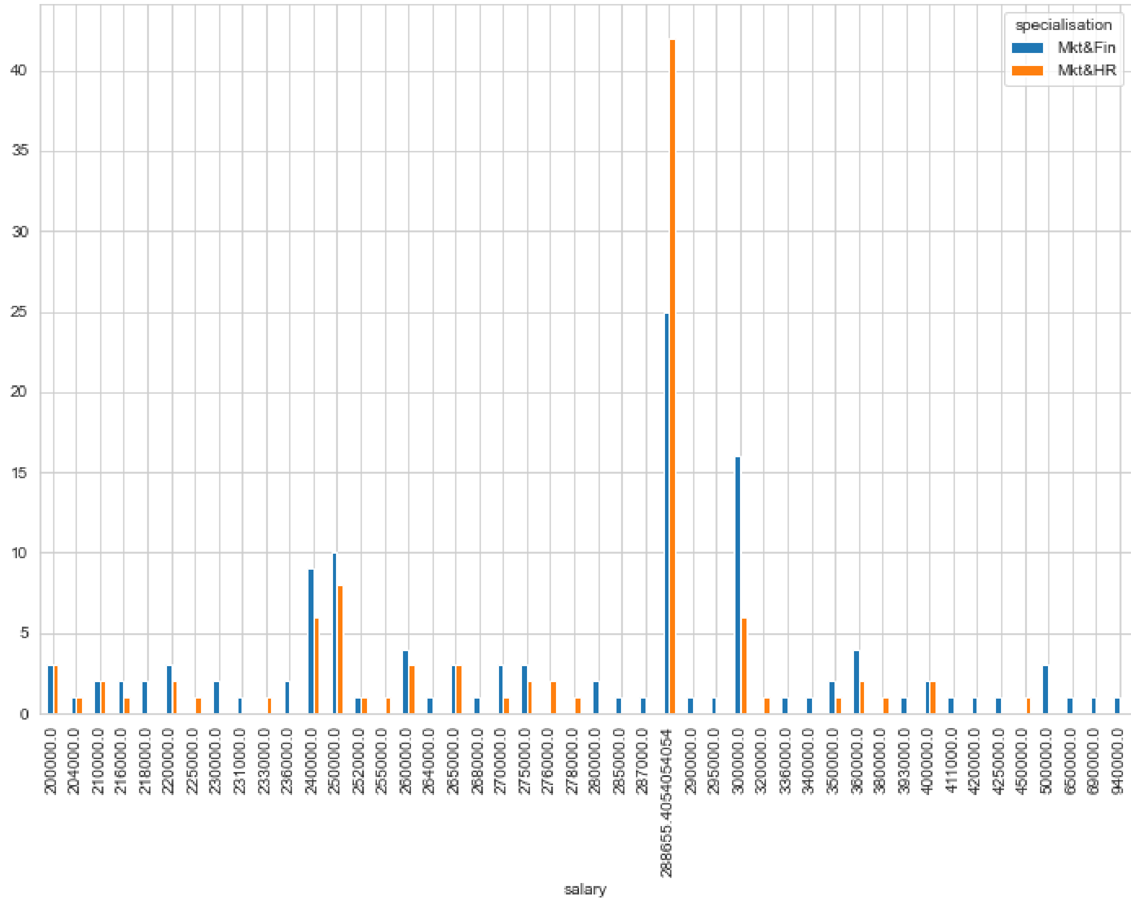
Check the which MBA specialisation is get higest package?

In [63]:

```
salary = pd.crosstab(data['salary'],data['specialisation'])
salary.plot(kind='bar',figsize=(12,8))
```

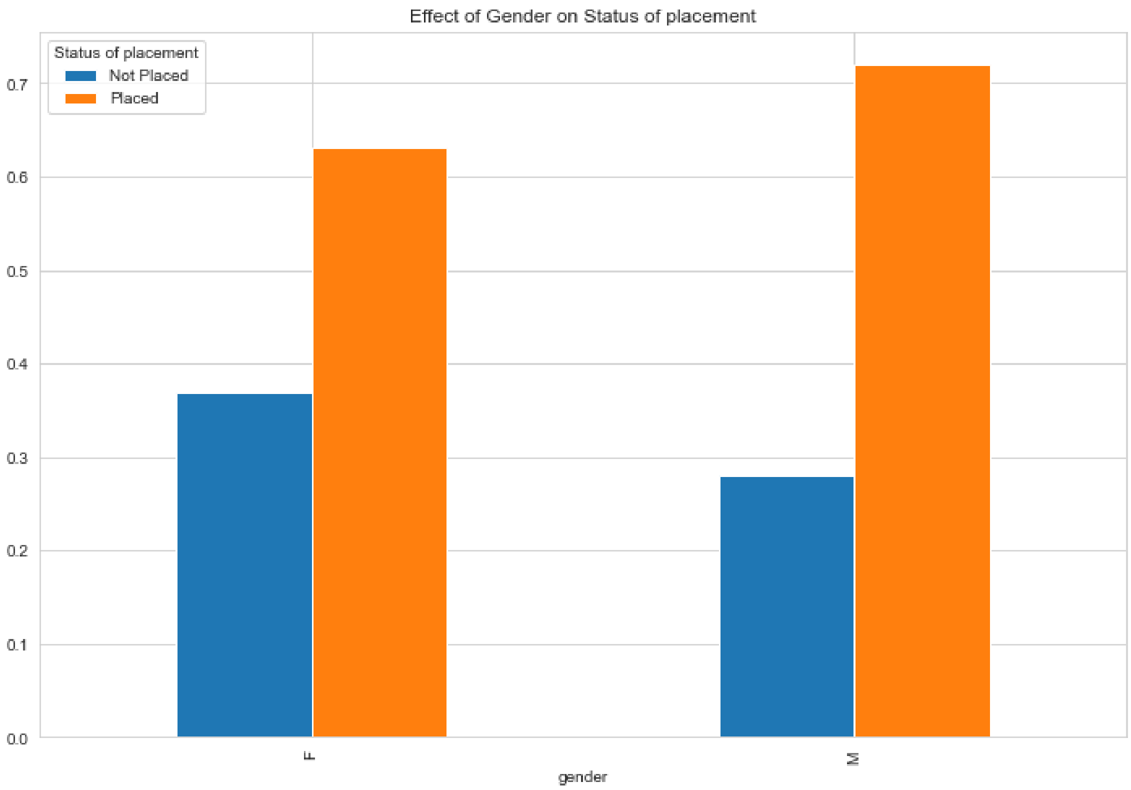
Out[63]:

<AxesSubplot:xlabel='salary'>



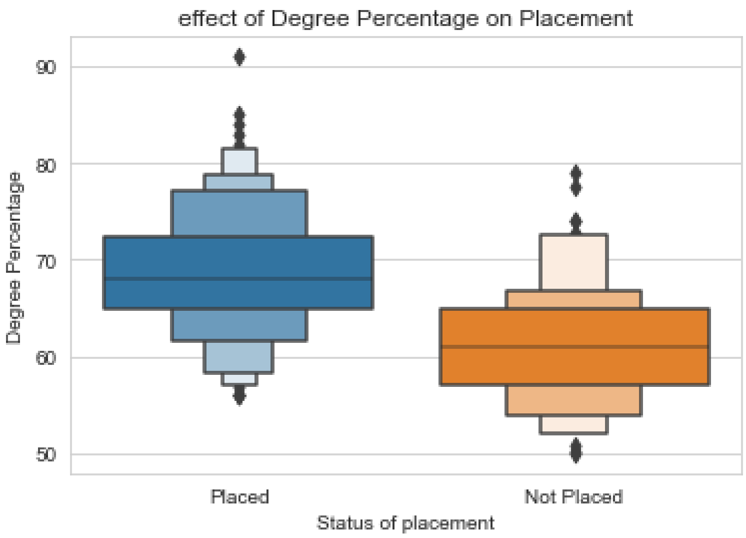
In [64]:

```
# Gender and Status of placement

x= pd.crosstab(data['gender'],data['Status of placement'])
x.div(x.sum(1).astype(float),axis=0).plot(kind='bar',stacked=False,figsize=(12,8))
plt.title('Effect of Gender on Status of placement')
plt.show()
```



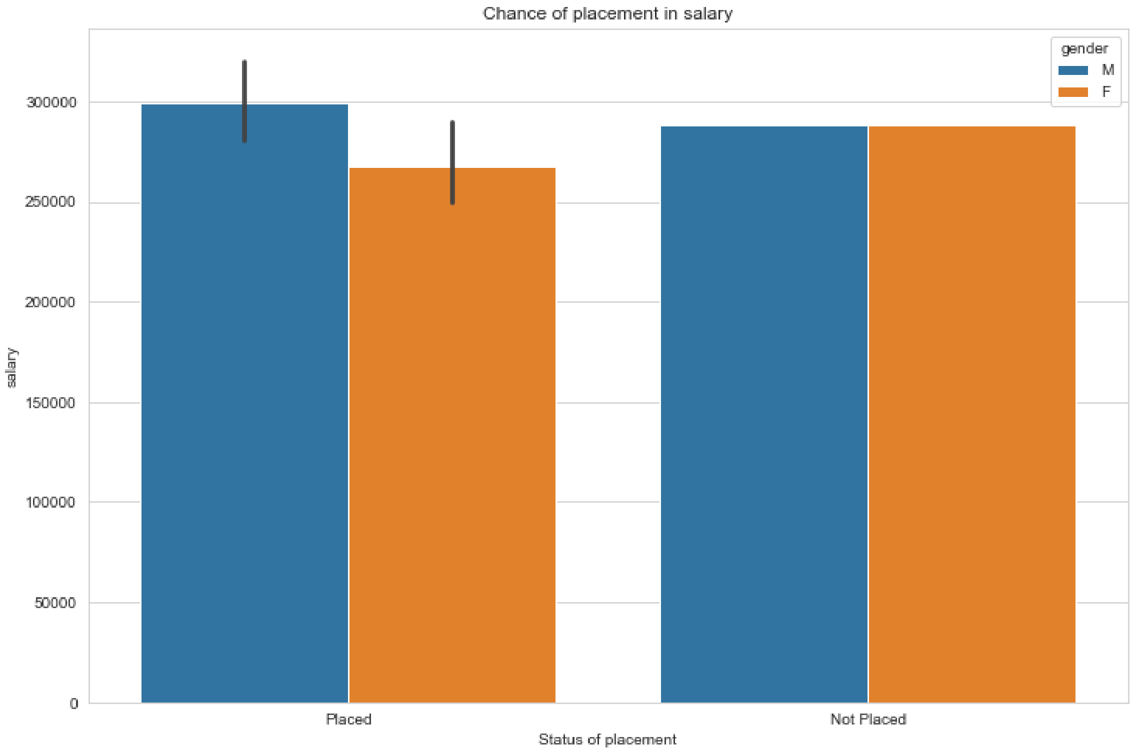
In [65]:

```
#  Degree Percentage and Status of placement

sns.boxenplot(data['Status of placement'],data['Degree Percentage'])
plt.title('effect of Degree Percentage on Placement')
plt.show()
```



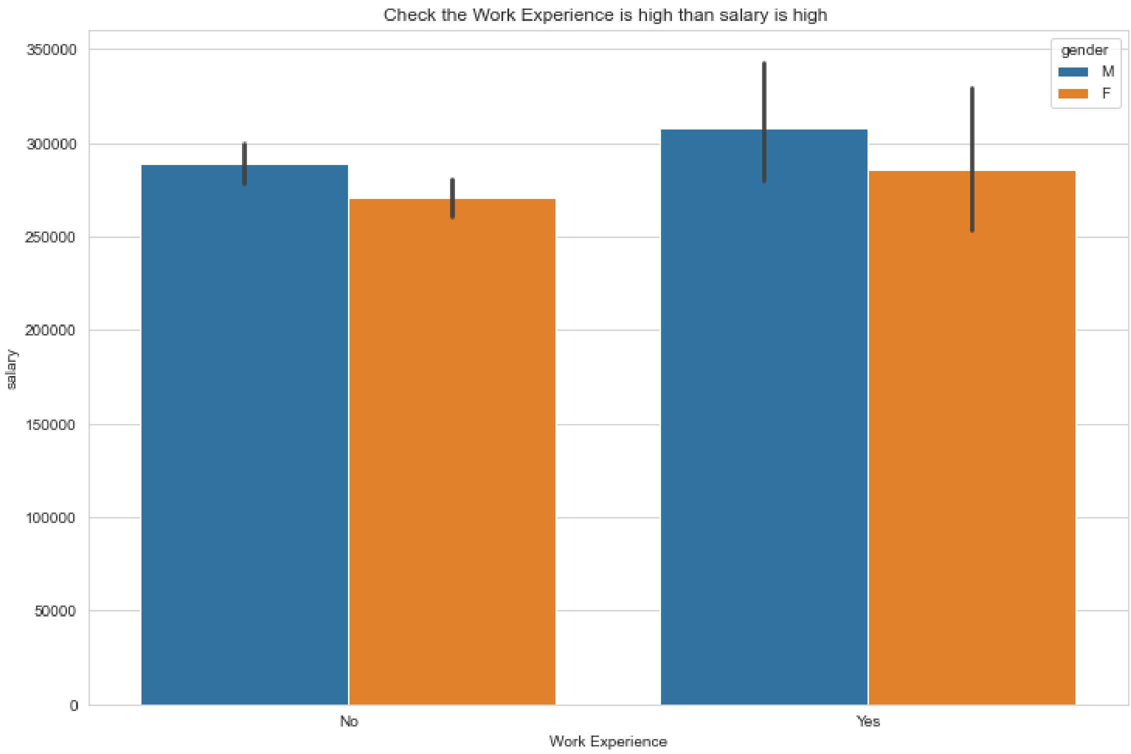
In [66]:

```
# salary and  #Stauts of placement

plt.figure(figsize=(12,8))
sns.barplot(data['Status of placement'],data['salary'],hue=data['gender'])
plt.title('Chance of placement in salary')
plt.xlabel('Status of placement')
plt.ylabel('salary')
plt.show()
```



In [67]:

```
# salary #Work Experience

plt.figure(figsize=(12,8))
sns.barplot(data['Work Experience'],data['salary'],hue=data['gender'])
plt.title('Check the Work Experience is high than salary is high ')
plt.xlabel('Work Experience')
plt.ylabel('salary')
plt.show()
```



In [68]:

```
cat_columns
```

Out[68]:

```
['gender',
 'Board of Education (middle)',
 'Board of Education (senior)',
 'Specialization in Higher Secondary Education',
 'Degree type- Field of degree education',
 'Work Experience',
 'specialisation',
 'Status of placement']
```

In [69]:

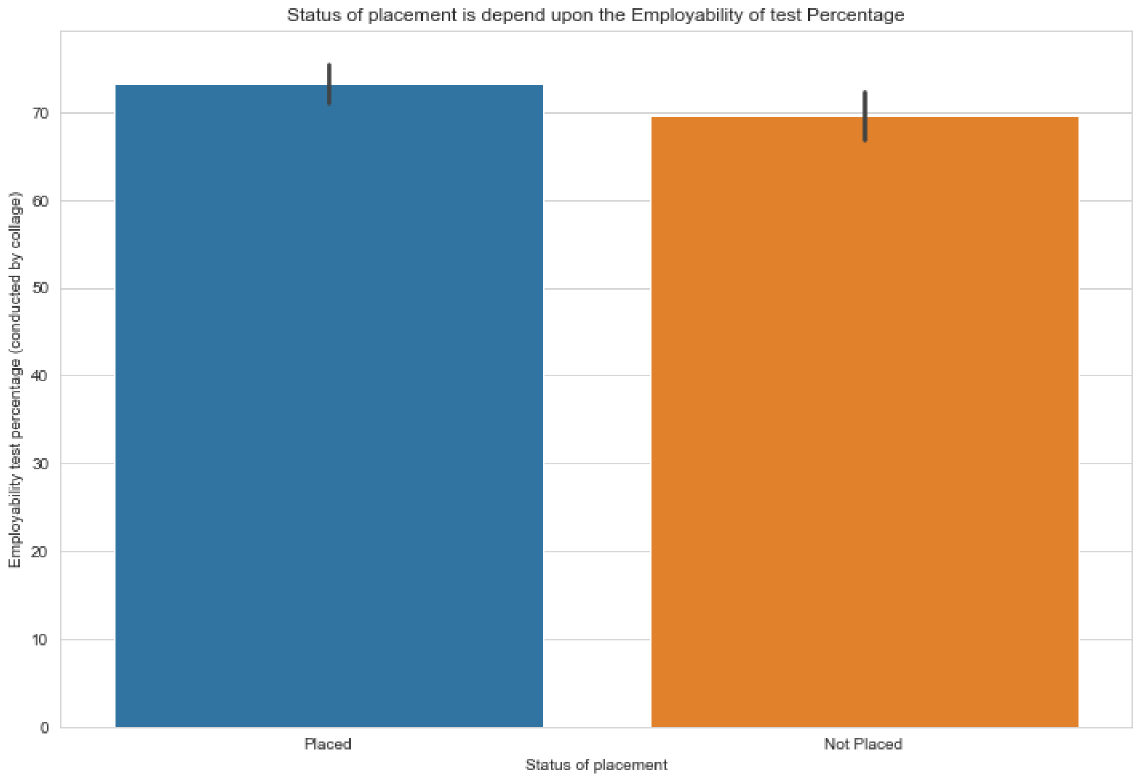
```
num_columns
```

Out[69]:

```
['Secondary Education Percentage (middle school)',
 'Higher Secondary School Percentage (senior)',
 'Degree Percentage',
 'Employability test percentage (conducted by college)',
 'MBA Percentage',
 'salary']
```

In [70]:

```
plt.figure(figsize=(12,8))
sns.barplot(data['Status of placement'],data['Employability test percentage (conducted b
plt.title('Status of placement is depend upon the Employability of test Percentage')
plt.xlabel('Status of placement')
plt.ylabel('Employability test percentage (conducted by collage)')
plt.show()
```



Target Columns

In [71]:

```
data['Status of placement'].value_counts()
```

Out[71]:

```
Placed        148
Not Placed     67
Name: Status of placement, dtype: int64
```

In [72]:

```
data['Status of placement'].unique()
```

Out[72]:

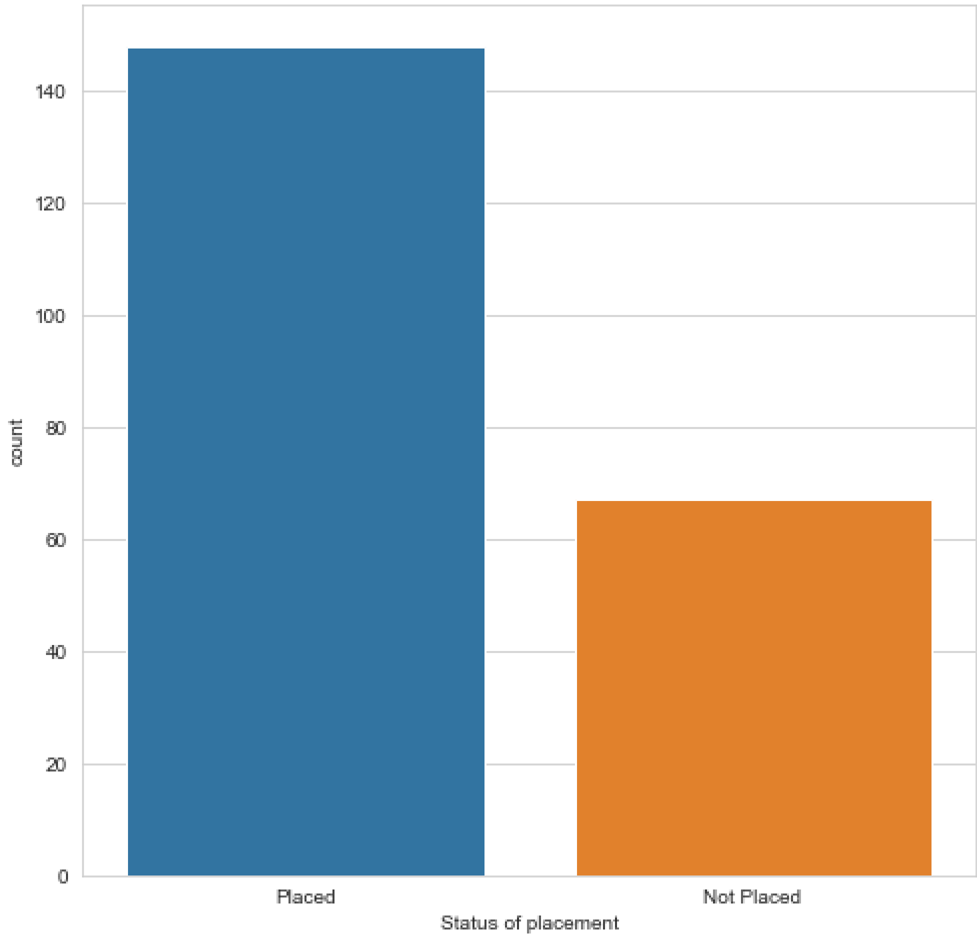
```
array(['Placed', 'Not Placed'], dtype=object)
```

In [73]:

```
plt.figure(figsize=(8,8))
sns.countplot(data['Status of placement'])
```

Out[73]:

```
<AxesSubplot:xlabel='Status of placement', ylabel='count'>
```



it is imblance data

We need s handle imblance data

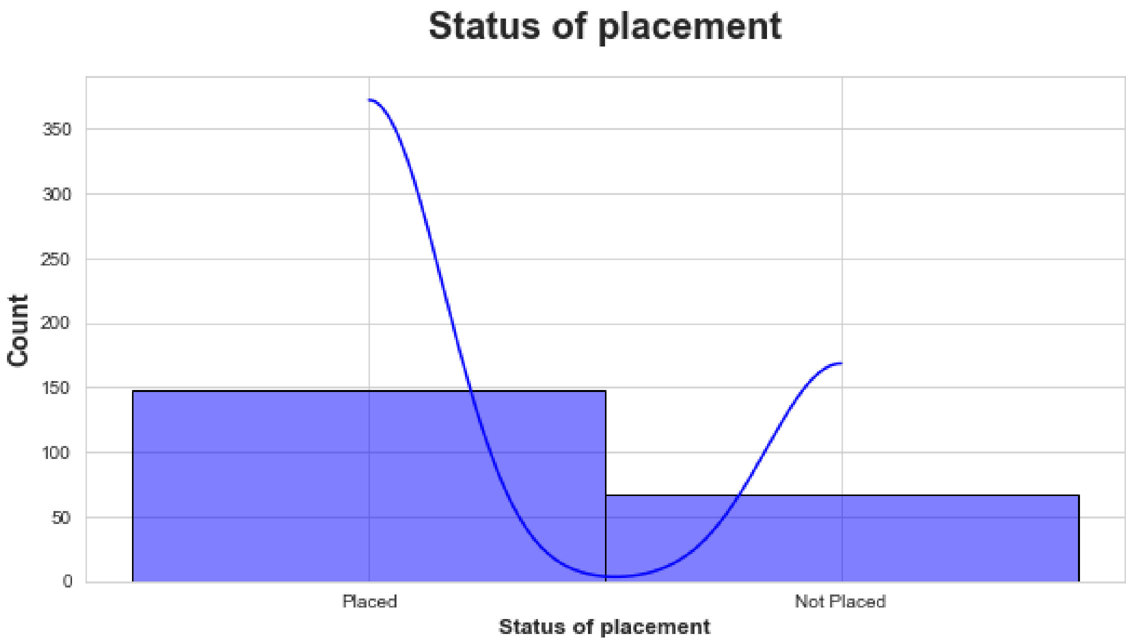
In [ ]:

```
```

Visulization of Target Columns

In [74]:

```
plt.subplots(figsize=(10,5))
sns.histplot(data['Status of placement'],ec='Black',color='blue',kde=True)
plt.title('Status of placement', weight='bold',fontsize=20,pad=20)
plt.ylabel('Count',weight='bold',fontsize=14)
plt.xlabel('Status of placement',weight='bold',fontsize=12)
plt.show()
```



Statistical Based Analysis

In [75]:

```
# check summary of stats columns
data.describe()
```

Out[75]:

	Secondary Education Percentage (middle school)	Higher Secondary School Percentage (senior)	Degree Percentage	Employability test percentage (conducted by college)	MBA Percentage	salary
count	215.000000	215.000000	215.000000	215.000000	215.000000	215.000000
mean	67.303395	66.333163	66.370186	72.100558	62.278186	288655.405405
std	10.827205	10.897509	7.358743	13.275956	5.833385	77457.900102
min	40.890000	37.000000	50.000000	50.000000	51.210000	200000.000000
25%	60.600000	60.900000	61.000000	60.000000	57.945000	250000.000000
50%	67.000000	65.000000	66.000000	71.000000	62.000000	288655.405405
75%	75.700000	73.000000	72.000000	83.500000	66.255000	288655.405405
max	89.400000	97.700000	91.000000	98.000000	77.890000	940000.000000

In [76]:

```
# check the summary of stats categorical columns
data.describe(include='object')
```

Out[76]:

	gender	Board of Education (middle)	Board of Education (senior)	Specialization in Higher Secondary Education	Degree type-Field of degree education	Work Experience	specialisatio
count	215	215	215	215	215	215	21
unique	2	2	2	3	3	2	
top	M	Central	Others	Commerce	Comm&Mgmt	No	Mkt&Fi
freq	139	116	131	113	145	141	12

In [77]:

```
# check the transpose summary of stats columns
data.describe().T
```

Out[77]:

	count	mean	std	min	25%	50%	
Secondary Education Percentage (middle school)	215.0	67.303395	10.827205	40.89	60.600	67.000000	
Higher Secondary School Percentage (senior)	215.0	66.333163	10.897509	37.00	60.900	65.000000	
Degree Percentage	215.0	66.370186	7.358743	50.00	61.000	66.000000	
Employability test percentage (conducted by college)	215.0	72.100558	13.275956	50.00	60.000	71.000000	
MBA Percentage	215.0	62.278186	5.833385	51.21	57.945	62.000000	
salary	215.0	288655.405405	77457.900102	200000.00	250000.000	288655.405405	28

In [78]:

```
# check the covariance
data.cov()
```

Out[78]:

	Secondary Education Percentage (middle school)	Higher Secondary School Percentage (senior)	Degree Percentage	Employability test percentage (conducted by college)	MBA Percentage	
Secondary Education Percentage (middle school)	117.228377	60.348373	42.897137	37.659225	24.535952	1.976
Higher Secondary School Percentage (senior)	60.348373	118.755706	34.819820	35.461678	22.555846	4.600
Degree Percentage	42.897137	34.819820	54.151103	21.929469	17.272020	-8.064
Employability test percentage (conducted by college)	37.659225	35.461678	21.929469	176.251018	16.886973	1.571
MBA Percentage	24.535952	22.555846	17.272020	16.886973	34.028376	6.611
salary	19767.643976	46008.057275	-8064.357161	157157.850783	66115.509283	5.999

In [79]:

```
# check the standard deviation
data.std()
```

Out[79]:

```
Secondary Education Percentage (middle school)              10.827205
Higher Secondary School Percentage (senior)                 10.897509
Degree Percentage                                            7.358743
Employability test percentage (conducted by college)        13.275956
MBA Percentage                                               5.833385
salary                                                   77457.900102
dtype: float64
```

In [80]:

```
# check the skewness
data.skew()
```

Out[80]:

```
Secondary Education Percentage (middle school)         -0.132649
Higher Secondary School Percentage (senior)             0.163639
Degree Percentage                                       0.244917
Employability test percentage (conducted by college)    0.282308
MBA Percentage                                          0.313576
salary                                                  4.288799
dtype: float64
```

In [81]:

```
# check the correlation
data.corr()
```

Out[81]:

	Secondary Education Percentage (middle school)	Higher Secondary School Percentage (senior)	Degree Percentage	Employability test percentage (conducted by college)	MBA Percentage	salary
Secondary Education Percentage (middle school)	1.000000	0.511472	0.538404	0.261993	0.388478	0.023571
Higher Secondary School Percentage (senior)	0.511472	1.000000	0.434206	0.245113	0.354823	0.054506
Degree Percentage	0.538404	0.434206	1.000000	0.224470	0.402364	-0.014148
Employability test percentage (conducted by college)	0.261993	0.245113	0.224470	1.000000	0.218055	0.152829
MBA Percentage	0.388478	0.354823	0.402364	0.218055	1.000000	0.146324
salary	0.023571	0.054506	-0.014148	0.152829	0.146324	1.000000

In [82]:

```
# check the cumalative value
data.cummax()
```

Out[82]:

	gender	Secondary Education Percentage (middle school)	Board of Education (middle)	Higher Secondary School Percentage (senior)	Board of Education (senior)	Specialization in Higher Secondary Education	Degree Percentage	D Fi d educ
0	M	67.00	Others	91.0	Others	Commerce	58.00	Sci
1	M	79.33	Others	91.0	Others	Science	77.48	Sci
2	M	79.33	Others	91.0	Others	Science	77.48	Sci
3	M	79.33	Others	91.0	Others	Science	77.48	Sci
4	M	85.80	Others	91.0	Others	Science	77.48	Sci
...	...	...	...	...	...	...	...	
210	M	89.40	Others	97.7	Others	Science	91.00	Sci
211	M	89.40	Others	97.7	Others	Science	91.00	Sci
212	M	89.40	Others	97.7	Others	Science	91.00	Sci
213	M	89.40	Others	97.7	Others	Science	91.00	Sci
214	M	89.40	Others	97.7	Others	Science	91.00	Sci

215 rows × 14 columns

In [83]:

```
# check the Kurotosis
data.kurt()
```

Out[83]:

```
Secondary Education Percentage (middle school)         -0.607510
Higher Secondary School Percentage (senior)             0.450765
Degree Percentage                                       0.052143
Employability test percentage (conducted by college)   -1.088580
MBA Percentage                                         -0.470723
salary                                                 28.012383
dtype: float64
```

In [84]:

```
# check the quantile
data.quantile()
```

Out[84]:

```
Secondary Education Percentage (middle school)            67.000000
Higher Secondary School Percentage (senior)               65.000000
Degree Percentage                                         66.000000
Employability test percentage (conducted by college)      71.000000
MBA Percentage                                            62.000000
salary                                                288655.405405
Name: 0.5, dtype: float64
```

In [85]:

```
#check the mean absolute deviation
data.mad()
```

Out[85]:

```
Secondary Education Percentage (middle school)            8.882360
Higher Secondary School Percentage (senior)               8.342326
Degree Percentage                                         5.757694
Employability test percentage (conducted by college)     11.391207
MBA Percentage                                            4.750406
salary                                                39441.106223
dtype: float64
```

In [86]:

```
# check the mean
data.mean()
```

Out[86]:

```
Secondary Education Percentage (middle school)            67.303395
Higher Secondary School Percentage (senior)               66.333163
Degree Percentage                                         66.370186
Employability test percentage (conducted by college)      72.100558
MBA Percentage                                            62.278186
salary                                                288655.405405
dtype: float64
```

In [87]:

```
# check the median
data.median()
```

Out[87]:

```
Secondary Education Percentage (middle school)            67.000000
Higher Secondary School Percentage (senior)               65.000000
Degree Percentage                                         66.000000
Employability test percentage (conducted by college)      71.000000
MBA Percentage                                            62.000000
salary                                                288655.405405
dtype: float64
```

In [88]:

```
# check the min values
data.min()
```

Out[88]:

```
gender                                                               F
Secondary Education Percentage (middle school)                   40.89
Board of Education (middle)                                    Central
Higher Secondary School Percentage (senior)                       37.0
Board of Education (senior)                                    Central
Specialization in Higher Secondary Education                      Arts
Degree Percentage                                                 50.0
Degree type- Field of degree education                       Comm&Mgmt
Work Experience                                                     No
Employability test percentage (conducted by college)              50.0
specialisation                                                 Mkt&Fin
MBA Percentage                                                   51.21
Status of placement                                         Not Placed
salary                                                        200000.0
dtype: object
```

In [89]:

```
# check the maximum value
data.max()
```

Out[89]:

```
gender                                                             M
Secondary Education Percentage (middle school)                  89.4
Board of Education (middle)                                   Others
Higher Secondary School Percentage (senior)                     97.7
Board of Education (senior)                                   Others
Specialization in Higher Secondary Education                 Science
Degree Percentage                                               91.0
Degree type- Field of degree education                      Sci&Tech
Work Experience                                                  Yes
Employability test percentage (conducted by college)            98.0
specialisation                                                Mkt&HR
MBA Percentage                                                 77.89
Status of placement                                           Placed
salary                                                      940000.0
dtype: object
```

In [90]:

```
# Calculate the value range of ecah variable that is the difference between the maximum 

#data.max() - data.min()
```

Plot the Heatmap

In [91]:

```
data.corr()
```

Out[91]:

	Secondary Education Percentage (middle school)	Higher Secondary School Percentage (senior)	Degree Percentage	Employability test percentage (conducted by college)	MBA Percentage	salary
Secondary Education Percentage (middle school)	1.000000	0.511472	0.538404	0.261993	0.388478	0.023571
Higher Secondary School Percentage (senior)	0.511472	1.000000	0.434206	0.245113	0.354823	0.054506
Degree Percentage	0.538404	0.434206	1.000000	0.224470	0.402364	-0.014148
Employability test percentage (conducted by college)	0.261993	0.245113	0.224470	1.000000	0.218055	0.152829
MBA Percentage	0.388478	0.354823	0.402364	0.218055	1.000000	0.146324
salary	0.023571	0.054506	-0.014148	0.152829	0.146324	1.000000

In [92]:

```
plt.figure(figsize=(12,10))
sns.heatmap(data.corr(),annot = True)
```

Out[92]:

```
<AxesSubplot:>
```

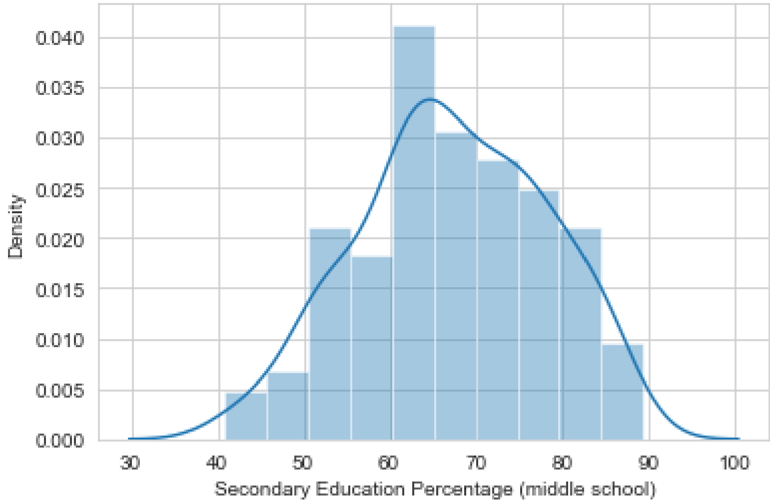
check the Distribution of Numerical Columns

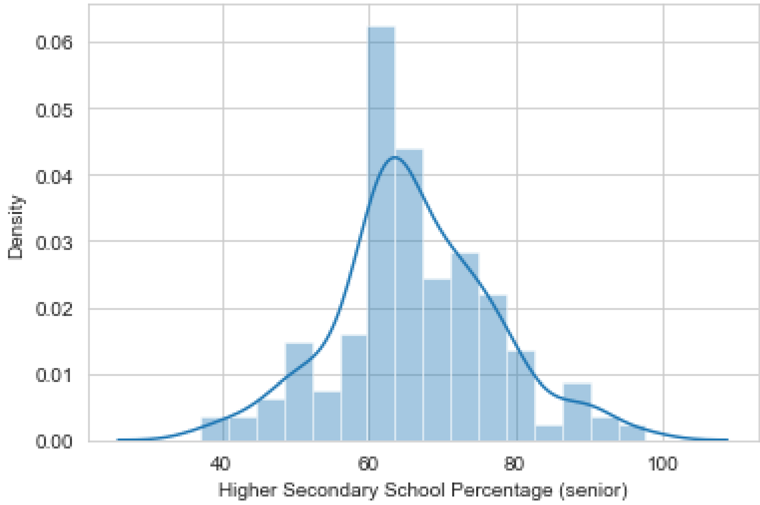
In [ ]:

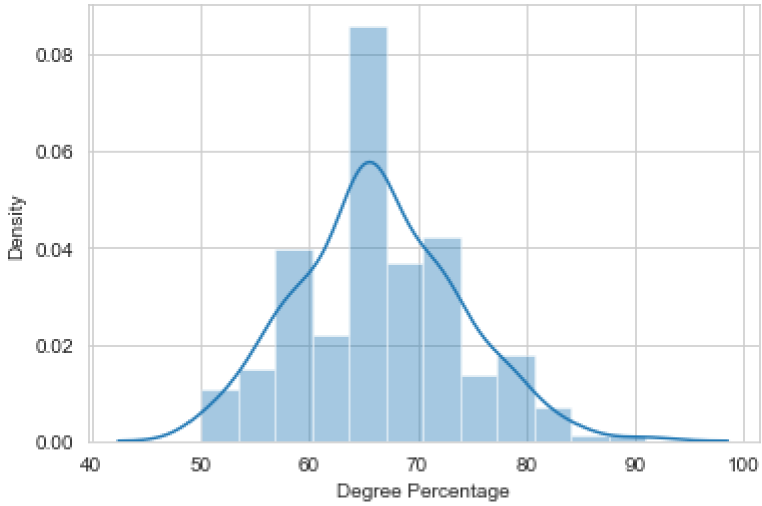
```

```

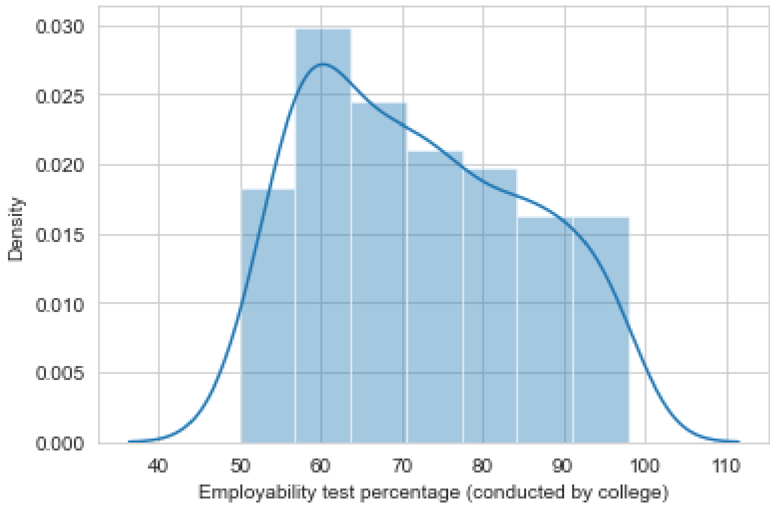
In [93]:

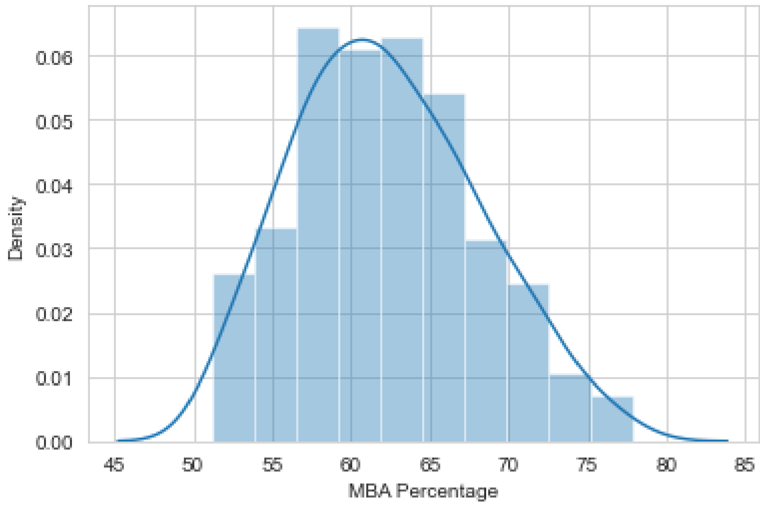
```
for i in num_columns:
    sns.distplot(data[i])
    plt.show()
```

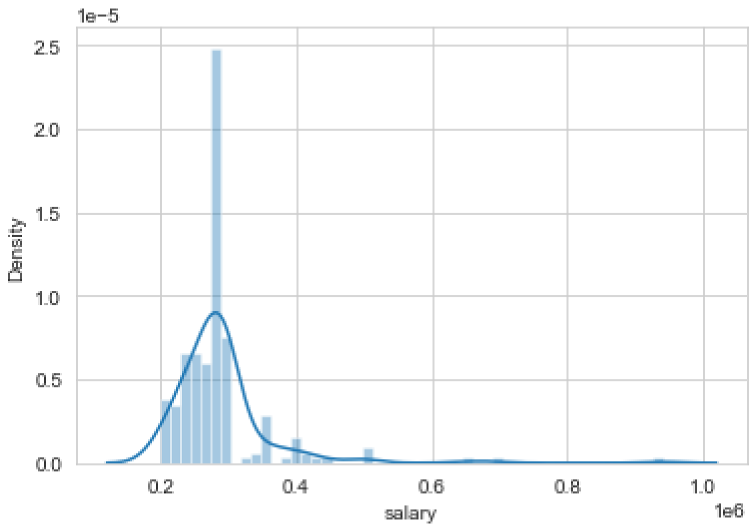






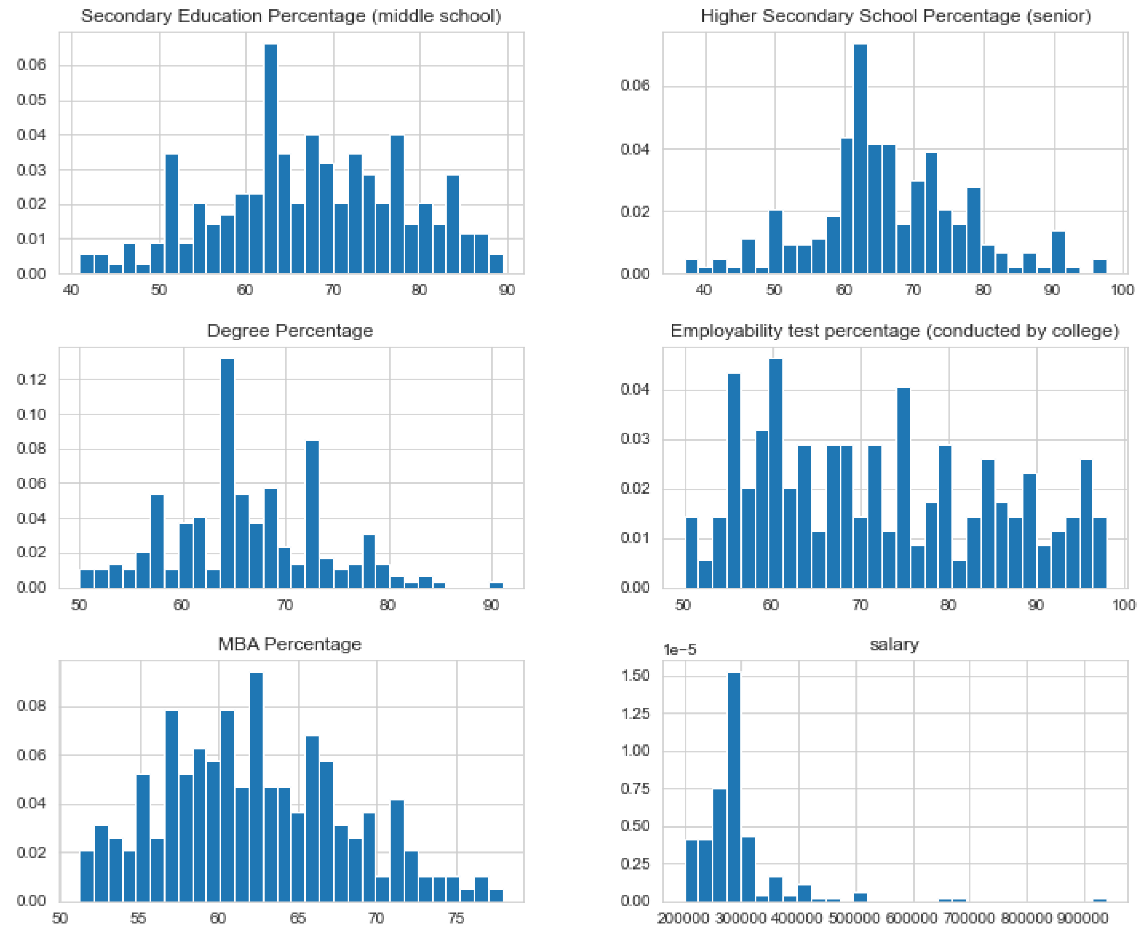

0.030
0.025
0.020
0.015
0.010
0.005
0.000
Density
40
50
60
70
80
90
100
110
Employability test percentage (conducted by college)


0.06
0.05
0.04
0.03
0.02
0.01
0.00
Density
45
50
55
60
65
70
75
80
85
MBA Percentage


1e−5
2.5
2.0
1.5
1.0
0.5
0.0
Density
0.2
0.4
0.6
0.8
1.0
1e6
salary

In [94]:

```
# Visualization the variable distribution with histograme
data.hist(bins=30,figsize=(12,10),density=True)
plt.show()
```



Check Multicollinearity for Categorical Feature

- A Chi-squared test(also chi-squared or x2 test) is a statistical hypothesis test that is valid to perform when the test statistic is chi-squared distribution under the null hypothesis, specifically Pearson's Chi-Squared Test
- A Chi- Squared statistic is one way to show a relationship between two Categorical Variable.
- Here we test correlation of Categorical Columns with Target columns .i.e. Status of placement

In [95]:

```
from scipy.stats import chi2_contingency

chi2_test = []
for feature in cat_columns:
    if chi2_contingency(pd.crosstab(data['Status of placement'], data[feature]))[1] <0.0
        chi2_test.append('Reject the Null Hypothesis')
    else:
        chi2_test.append('Fail to Reject Null Hypothesis')
result = pd.DataFrame(data=[cat_columns,chi2_test]).T
result.columns = ['Column' ,'Hypothesis Result']
result
```

Out[95]:

	Column	Hypothesis Result
0	gender	Fail to Reject Null Hypothesis
1	Board of Education (middle)	Fail to Reject Null Hypothesis
2	Board of Education (senior)	Fail to Reject Null Hypothesis
3	Specialization in Higher Secondary Education	Fail to Reject Null Hypothesis
4	Degree type- Field of degree education	Fail to Reject Null Hypothesis
5	Work Experience	Reject the Null Hypothesis
6	specialisation	Reject the Null Hypothesis
7	Status of placement	Reject the Null Hypothesis

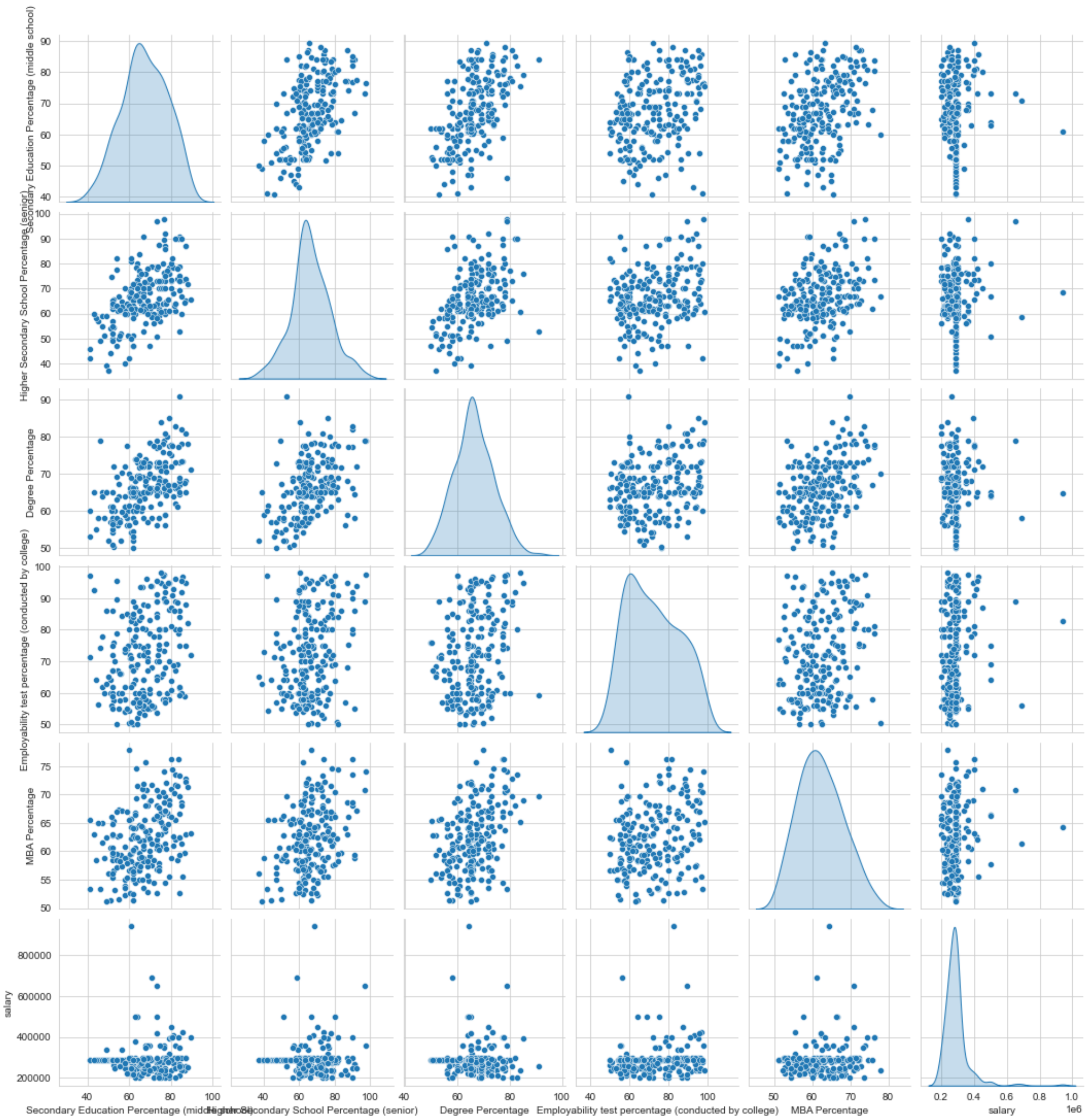
Pair plot

In [96]:

```
sns.pairplot(data , diag_kind = 'kde')
```

Out[96]:

<seaborn.axisgrid.PairGrid at 0x13ef9cda190>



In [139]:

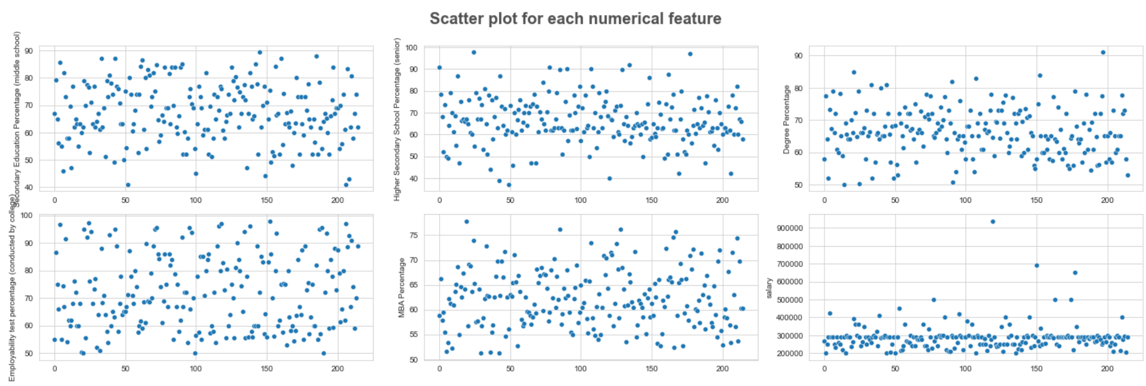
```
plt.figure(figsize=(20,15))
sns.pairplot(data, hue='Status of placement', size=1.5)
plt.show()
```

```
    552             "or a callable."
    553     raise ValueError(msg)
--> 555 self._compute_covariance()

File ~\anaconda3\lib\site-packages\scipy\stats\_kde.py:567, in gaussian
_kde._compute_covariance(self)
    563 if not hasattr(self, '_data_inv_cov'):
    564     self._data_covariance = atleast_2d(cov(self.dataset, rowvar
=1,
    565                                        bias=False,
    566                                        aweights=self.weights))
--> 567     self._data_inv_cov = linalg.inv(self._data_covariance)
    569 self.covariance = self._data_covariance * self.factor**2
    570 self.inv_cov = self._data_inv_cov / self.factor**2

File ~\anaconda3\lib\site-packages\scipy\linalg\_basic.py:956, in inv
(a, overwrite_a, check_finite)
    954     inv_a, info = getri(lu, piv, lwork=lwork, overwrite_lu=1)
    955 if info > 0:
```

Scatter Plot

In [97]:

```
plt.figure(figsize=(20,15))
plt.suptitle('Scatter plot for each numerical feature', fontsize = 20 , fontweight = 'bo
for i in range(0 , len(num_columns)):
    plt.subplot(5,3,i+1)
    sns.scatterplot(y=num_columns[i],x=data.index, data = data)
    plt.tight_layout()
```

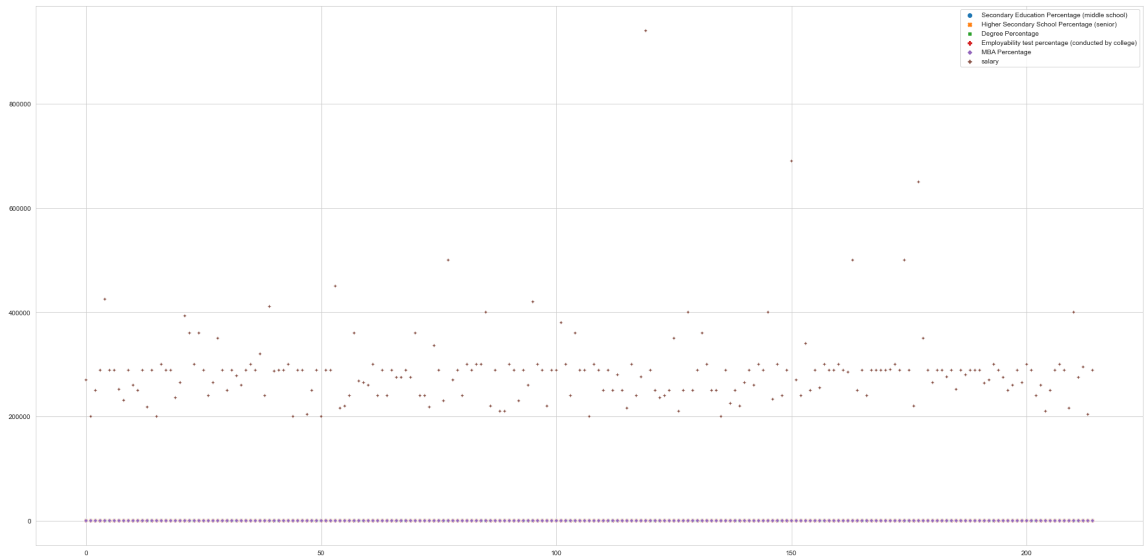


In [98]:

```
# check the scatter plot
plt.figure(figsize=(30,15))
sns.scatterplot(data=data)
```

Out[98]:

<AxesSubplot:>



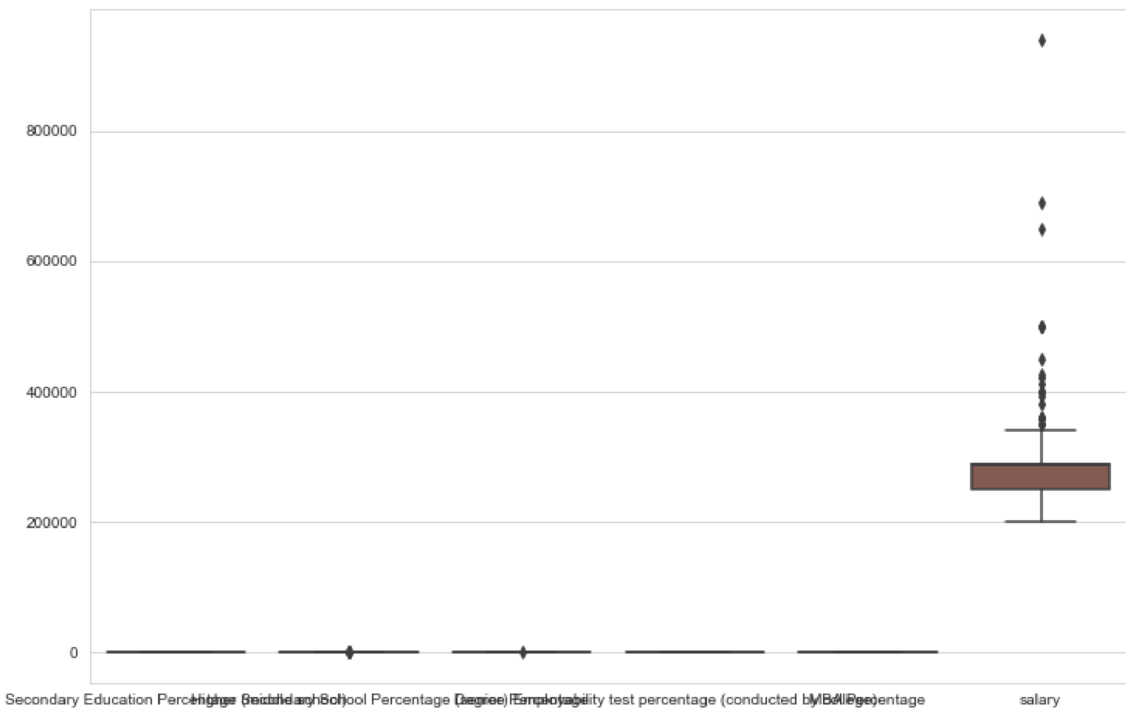
Box PLot

In [99]:

```
plt.figure(figsize=(12,8))
sns.boxplot(data=data , orient = 'v')
```

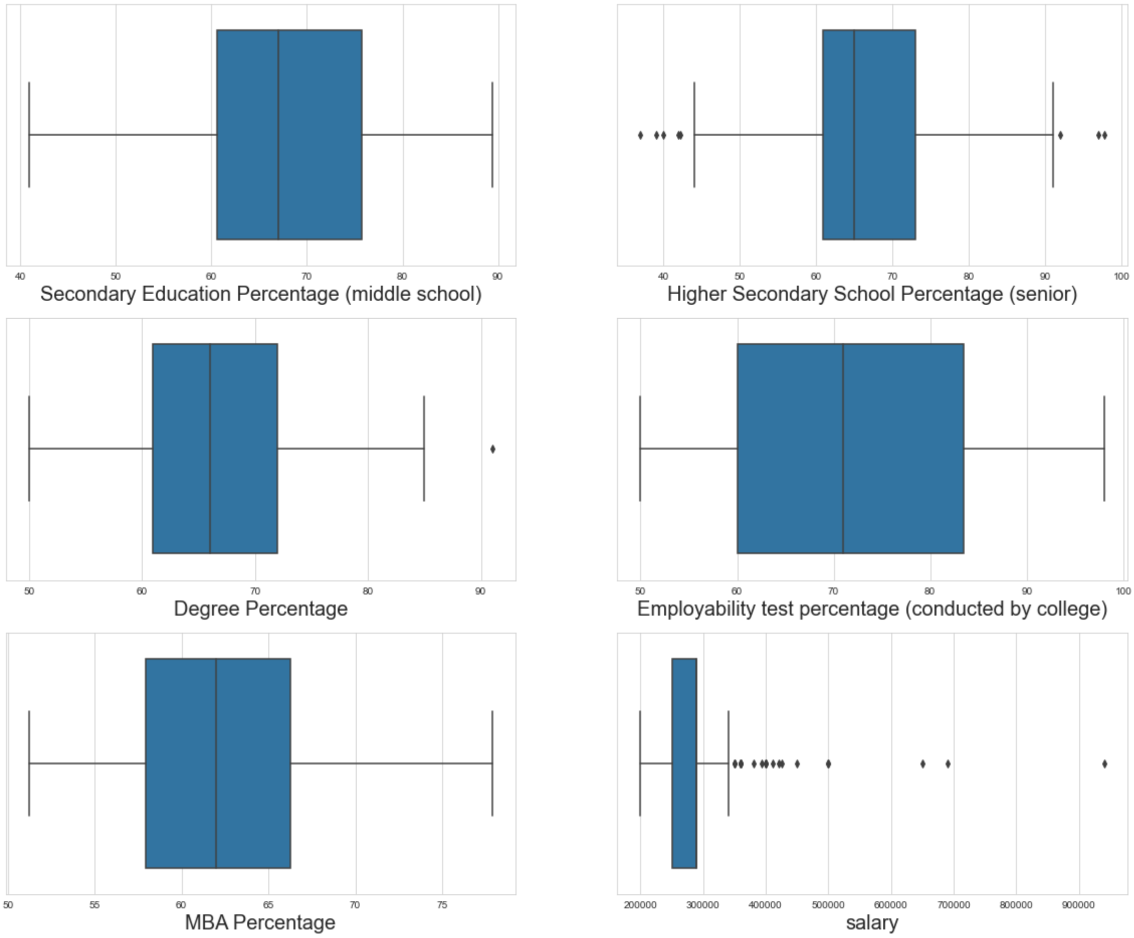
Out[99]:

<AxesSubplot:>



In [100]:

```
plt.figure(figsize=(20,45))
plotnumber=1

for i in num_columns:
    if plotnumber<=7:
        ax=plt.subplot(8,2,plotnumber)
        sns.boxplot(data[i])
        plt.xlabel(i,fontsize=20)

    plotnumber+=1
plt.show()
```



- it is indicated the two columns are outliers 1. Higher Secondary Sechool Percentage and 2 . Salary

Droping the Outliers

In [101]:

```
IQR = data['Higher Secondary School Percentage (senior)'].quantile(0.75) - data['Higher
lower_fence = data['Higher Secondary School Percentage (senior)'].quantile(0.25) - 1.5*I
upper_fence = data['Higher Secondary School Percentage (senior)'].quantile(0.75) + 1.5*I

lower_fence , upper_fence
```

Out[101]:

```
(42.75, 91.15)
```

In [102]:

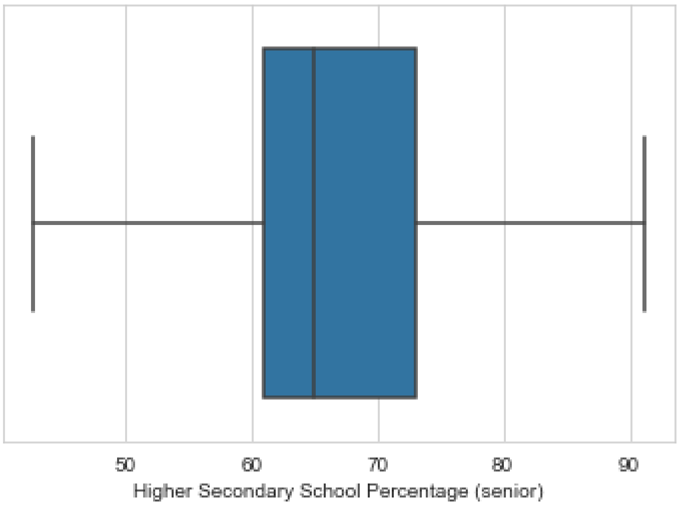
```
data['Higher Secondary School Percentage (senior)']=np.where(data['Higher Secondary Scho
```

In [103]:

```
sns.boxplot(data['Higher Secondary School Percentage (senior)'])
```

Out[103]:

```
<AxesSubplot:xlabel='Higher Secondary School Percentage (senior)'>
```



In [104]:

```
IQR = data['salary'].quantile(0.75) - data['salary'].quantile(0.25)
lower_fence = data['salary'].quantile(0.25) - 1.5*IQR
upper_fence = data['salary'].quantile(0.75) + 1.5*IQR

lower_fence , upper_fence
```

Out[104]:

```
(192016.89189189192, 346638.5135135135)
```

In [105]:

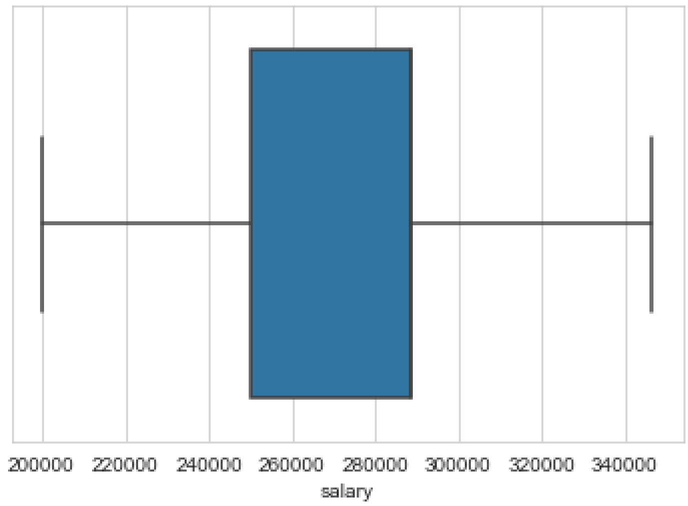
```
data['salary'] = np.where(data['salary']>upper_fence , upper_fence , np.where(data['sala
```

In [106]:

```
sns.boxplot(data['salary'])
```

Out[106]:

<AxesSubplot:xlabel='salary'>



Encoding Categorical Variables

In [107]:

```
cat_column = [feature for feature in data.columns if data[feature].dtype=='object']
print('Number of Categorical Feature : ', len(cat_column))
```

```
Number of Categorical Feature :  8
```

In [108]:

```
# checking the unique value categories in each columns

for feature in cat_column:
    print(feature , ':' , data[feature].unique())
```

```
gender : ['M' 'F']
Board of Education (middle) : ['Others' 'Central']
Board of Education (senior) : ['Others' 'Central']
Specialization in Higher Secondary Education : ['Commerce' 'Science' 'Art
s']
Degree type- Field of degree education : ['Sci&Tech' 'Comm&Mgmt' 'Others']
Work Experience : ['No' 'Yes']
specialisation : ['Mkt&HR' 'Mkt&Fin']
Status of placement : ['Placed' 'Not Placed']
```

In [109]:

```
# indicated the unique value in each categorical column

for feature in cat_column:
    print(feature , ':' , data[feature].nunique())
```

```
gender : 2
Board of Education (middle) : 2
Board of Education (senior) : 2
Specialization in Higher Secondary Education : 3
Degree type- Field of degree education : 3
Work Experience : 2
specialisation : 2
Status of placement : 2
```

Binary Encoding

In [110]:

```
data['gender'] = data['gender'].replace({'M':0 , 'F':1})
data['Board of Education (middle)'] = data['Board of Education (middle)'].replace({'Othe
data['Board of Education (senior)'] = data['Board of Education (senior)'].replace({'Othe
data['Work Experience'] = data['Work Experience'].replace({'No':0,"Yes":1})
data['specialisation'] = data['specialisation'].replace({'Mkt&HR':0,'Mkt&Fin':1})
data['Status of placement'] = data['Status of placement'].replace({'Placed':1,'Not Place
```

Label Encoder

In [111]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [112]:

```
data['Specialization in Higher Secondary Education'] = le.fit_transform(data['Specializa
data['Degree type- Field of degree education'] = le.fit_transform(data['Degree type- Fie
```

In [113]:

```
data.head()
```

Out[113]:

	gender	Secondary Education Percentage (middle school)	Board of Education (middle)	Higher Secondary School Percentage (senior)	Board of Education (senior)	Specialization in Higher Secondary Education	Degree Percentage	Deg ty Fielc deg educat
0	0	67.00	0	91.00	0	1	58.00	
1	0	79.33	1	78.33	0	2	77.48	
2	0	65.00	1	68.00	1	0	64.00	
3	0	56.00	1	52.00	1	2	52.00	
4	0	85.80	1	73.60	1	1	73.30	

Train Test Split

- Split the Dataframe X and Y
- X-- is independent columns and variable and Y is dependent columns "Status of placement"

In [140]:

```
data.shape
```

Out[140]:

```
(215, 14)
```

In [116]:

```
from sklearn.model_selection import train_test_split
```

In [144]:

```
X=data.drop(['Status of placement'],axis=1)
y=data['Status of placement']
```

In [145]:

```
X_train,X_test , y_train,y_test = train_test_split(X,y,test_size=0.30 , random_state = 3
```

In [146]:

```
X_train.shape , y_train.shape
```

Out[146]:

```
((150, 13), (150,))
```

In [147]:

```
X_test.shape , y_test.shape
```

Out[147]:

```
((65, 13), (65,))
```

Standardization or Feature Scaling

In [148]:

```
from sklearn.preprocessing import StandardScaler
```

In [149]:

```
std_scaler = StandardScaler()
std_scaler
```

Out[149]:

```
StandardScaler()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [150]:

```
X_train = std_scaler.fit_transform(X_train)
X_train
```

Out[150]:

```
array([[-0.72843136, -1.08682979, -1.1751393 , ..., -1.1751393 ,
         0.05028537,  0.29231618],
       [-0.72843136,  0.85471545,  0.85096294, ...,  0.85096294,
         1.00170375, -0.69945877],
       [-0.72843136, -0.86599052,  0.85096294, ...,  0.85096294,
        -1.21446462, -0.44289053],
       ...,
       [-0.72843136, -0.22187599,  0.85096294, ..., -1.1751393 ,
        -0.09658882, -0.03238134],
       [ 1.37281295, -0.22187599,  0.85096294, ...,  0.85096294,
         0.36688085, -1.9823    ],
       [-0.72843136, -1.21565269,  0.85096294, ..., -1.1751393 ,
        -0.64818301,  0.29231618]])
```

In [151]:

```
X_test = std_scaler.fit_transform(X_test)
X_test
```

Out[151]:

```
array([[ 1.30703226, -1.88396545,  1.11417203, -0.36336831,  1.4474937
3,
        -2.54950976, -0.07489567, -0.6494227 , -0.64268459,  0.6105495
8,
         0.98473193, -0.07755965,  0.31256944],
       [ 1.30703226,  1.00946232,  1.11417203,  0.11416788,  1.4474937
3,
         1.09264704, -0.04700556,  1.63233274, -0.64268459,  0.4150572
6,
        -1.0155048 ,  2.0241752 ,  0.31256944],
       [-0.76509206,  1.21613573, -0.89752747,  0.97373304, -0.6908492
8,
        -0.72843136,  1.45906038, -0.6494227 , -0.64268459, -0.7187981
9,
         0.98473193, -0.63483783, -1.11174583],
       [-0.76509206,  0.91176216, -0.89752747,  1.30800838, -0.6908492
8,
        -0.72843136,  0.41318125, -0.6494227 ,  1.55597321,  0.2469338
```

Model Building

In [152]:

```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(X_train,y_train)
y_pred_log_reg = log_reg.predict(X_test)
y_pred_log_reg
```

Out[152]:

```
array([0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1],
      dtype=int64)
```

In [ ]:

In [158]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier

from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score , classification_report, ConfusionMatrixDispl
```

In [167]:

```
models = {
    "Logistic Regression": LogisticRegression(),
    "Support Vector Classifier": SVC(),
    "Decision Tree Classifier ": DecisionTreeClassifier(),
    "Random Forest Classifier ": RandomForestClassifier(),
    "Ada Boost Classifier ": AdaBoostClassifier(),
    " Navie Bayes" : GaussianNB()
}

for i in range(len(list(models))):
    model = list(models.values())[i]
    model.fit(X_train,y_train)

    # prediction

    y_train_pred = model.predict(X_train)
    y_test_pred = model.predict(X_test)

    # training set performance
    model_train_accuracy = accuracy_score(y_train,y_train_pred)
    model_train_f1 = f1_score(y_train,y_train_pred)
    model_train_precision = precision_score(y_train,y_train_pred)
    model_train_recall = recall_score(y_train,y_train_pred)



    # Test set performance
    model_test_accuracy = accuracy_score(y_test, y_test_pred)
    model_test_f1 = f1_score(y_test, y_test_pred)
    model_test_precision = precision_score(y_test, y_test_pred)
    model_test_recall = recall_score(y_test, y_test_pred)


    print(list(models.keys())[i])

    print('Model performance for Training set')
    print("- Accuracy: {:.4f}".format(model_train_accuracy))
    print('- F1 score: {:.4f}'.format(model_train_f1))
    print('- Precision: {:.4f}'.format(model_train_precision))
    print('- Recall: {:.4f}'.format(model_train_recall))

    print('----------------------------------')

    print('Model performance for Test set')
    print('- Accuracy: {:.4f}'.format(model_test_accuracy))
    print('- F1 score: {:.4f}'.format(model_test_f1))
    print('- Precision: {:.4f}'.format(model_test_precision))
    print('- Recall: {:.4f}'.format(model_test_recall))

    print("="*20)
    print('\n')
```

```
Logistic Regression
Model performance for Training set
- Accuracy: 0.9133
- F1 score: 0.9406
- Precision: 0.9196
- Recall: 0.9626
----------------------------------
Model performance for Test set
- Accuracy: 0.8462
- F1 score: 0.8780
- Precision: 0.8780
- Recall: 0.8780
====================


Support Vector Classifier
Model performance for Training set
- Accuracy: 0.9533
- F1 score: 0.9683
- Precision: 0.9386
- Recall: 1.0000
----------------------------------
Model performance for Test set
- Accuracy: 0.9385
- F1 score: 0.9535
- Precision: 0.9111
- Recall: 1.0000
====================


Decision Tree Classifier
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.0000
- Precision: 1.0000
- Recall: 1.0000
----------------------------------
Model performance for Test set
- Accuracy: 0.9846
- F1 score: 0.9877
- Precision: 1.0000
- Recall: 0.9756
====================


Random Forest Classifier
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.0000
- Precision: 1.0000
- Recall: 1.0000
----------------------------------
Model performance for Test set
- Accuracy: 0.8923
- F1 score: 0.9213
- Precision: 0.8542
- Recall: 1.0000
====================


Ada Boost Classifier
```

```
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.0000
- Precision: 1.0000
- Recall: 1.0000
----------------------------------
Model performance for Test set
- Accuracy: 0.9846
- F1 score: 0.9877
- Precision: 1.0000
- Recall: 0.9756
===================================


 Navie Bayes
Model performance for Training set
- Accuracy: 1.0000
- F1 score: 1.0000
- Precision: 1.0000
- Recall: 1.0000
----------------------------------
Model performance for Test set
- Accuracy: 0.6308
- F1 score: 0.7736
- Precision: 0.6308
- Recall: 1.0000
===================================
```

We are observed that Our Model Decision Tree , Ada Boost and Navie Baise are accuracy is 100%

- Made By:---Mukul Singh
- Thank You

In [ ]:

```

```