In [5]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import numpy as np
import warnings
warnings.filterwarnings('ignore')

from scipy import stats
```

In [6]:
```
df = pd.read_csv("data_1.csv")
```

In [7]:
```
df
```

Out[7]:

	Unnamed: 0	ID	Salary	DOJ	DOL	Designation	JobCity	Gender
0	train	203097	420000.0	01-06-2012 00:00	present	senior quality engineer	Bangalore	f
1	train	579905	500000.0	01-09-2013 00:00	present	assistant manager	Indore	m
2	train	810601	325000.0	01-06-2014 00:00	present	systems engineer	Chennai	f
3	train	267447	1100000.0	01-07-2011 00:00	present	senior software engineer	Gurgaon	m
4	train	343523	200000.0	01-03-2014 00:00	01-03-2015 00:00	get	Manesar	m
...	...	...	...	...	...	...	...	...
3993	train	47916	280000.0	01-10-2011 00:00	01-10-2012 00:00	software engineer	New Delhi	m
3994	train	752781	100000.0	01-07-2013 00:00	01-07-2013 00:00	technical writer	Hyderabad	f
3995	train	355888	320000.0	01-07-2013 00:00	present	associate software engineer	Bangalore	m
3996	train	947111	200000.0	01-07-2014 00:00	01-01-2015 00:00	software developer	Asifabadbanglore	f
3997	train	324966	400000.0	01-02-2013 00:00	present	senior systems engineer	Chennai	f

3998 rows × 39 columns

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3998 entries, 0 to 3997
Data columns (total 39 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Unnamed: 0             3998 non-null   object 
 1   ID                     3998 non-null   int64  
 2   Salary                 3998 non-null   float64
 3   DOJ                    3998 non-null   object 
 4   DOL                    3998 non-null   object 
 5   Designation            3998 non-null   object 
 6   JobCity                3998 non-null   object 
 7   Gender                 3998 non-null   object 
 8   DOB                    3998 non-null   object 
 9   10percentage           3998 non-null   float64
 10  10board                3998 non-null   object 
 11  12graduation           3998 non-null   int64  
 12  12percentage           3998 non-null   float64
 13  12board                3998 non-null   object 
 14  CollegeID              3998 non-null   int64  
 15  CollegeTier            3998 non-null   int64  
 16  Degree                 3998 non-null   object 
 17  Specialization         3998 non-null   object 
 18  collegeGPA             3998 non-null   float64
 19  CollegeCityID          3998 non-null   int64  
 20  CollegeCityTier        3998 non-null   int64  
 21  CollegeState           3998 non-null   object 
 22  GraduationYear         3998 non-null   int64  
 23  English                3998 non-null   int64  
 24  Logical                3998 non-null   int64  
 25  Quant                  3998 non-null   int64  
 26  Domain                 3998 non-null   float64
 27  ComputerProgramming    3998 non-null   int64  
 28  ElectronicsAndSemicon  3998 non-null   int64  
 29  ComputerScience        3998 non-null   int64  
 30  MechanicalEngg         3998 non-null   int64  
 31  ElectricalEngg         3998 non-null   int64  
 32  TelecomEngg            3998 non-null   int64  
 33  CivilEngg              3998 non-null   int64  
 34  conscientiousness      3998 non-null   float64
 35  agreeableness          3998 non-null   float64
 36  extraversion           3998 non-null   float64
 37  nueroticism            3998 non-null   float64
 38  openess_to_experience  3998 non-null   float64
dtypes: float64(10), int64(17), object(12)
memory usage: 1.2+ MB
```

In [8]:
```
df = df.drop('Unnamed: 0',axis=1)
```

In [9]:
```
df.columns
```

Out[9]:
```
Index(['ID', 'Salary', 'DOJ', 'DOL', 'Designation', 'JobCity', 'Gender',
'DOB',
       '10percentage', '10board', '12graduation', '12percentage', '12boar
d',
       'CollegeID', 'CollegeTier', 'Degree', 'Specialization', 'collegeGP
A',
       'CollegeCityID', 'CollegeCityTier', 'CollegeState', 'GraduationYea
r',
       'English', 'Logical', 'Quant', 'Domain', 'ComputerProgramming',
       'ElectronicsAndSemicon', 'ComputerScience', 'MechanicalEngg',
       'ElectricalEngg', 'TelecomEngg', 'CivilEngg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience'],
      dtype='object')
```

In [10]:
```
df.columns = df.columns.str.lower()
```

In [12]:
```
df.head()
```

Out[12]:

	id	salary	doj	dol	designation	jobcity	gender	dob	10percentage
0	203097	420000.0	01-06-2012 00:00	present	senior quality engineer	Bangalore	f	19-02-1990 00:00	84.3
1	579905	500000.0	01-09-2013 00:00	present	assistant manager	Indore	m	04-10-1989 00:00	85.4
2	810601	325000.0	01-06-2014 00:00	present	systems engineer	Chennai	f	03-08-1992 00:00	85.0
3	267447	1100000.0	01-07-2011 00:00	present	senior software engineer	Gurgaon	m	05-12-1989 00:00	85.6
4	343523	200000.0	01-03-2014 00:00	01-03-2015 00:00	get	Manesar	m	27-02-1991 00:00	78.0

5 rows × 38 columns

In [13]:
```
df['doj'] = pd.to_datetime(df['doj'])
```

In [15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3998 entries, 0 to 3997
Data columns (total 38 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     3998 non-null   int64
 1   salary                 3998 non-null   float64
 2   doj                    3998 non-null   datetime64[ns]
 3   dol                    3998 non-null   object
 4   designation            3998 non-null   object
 5   jobcity                3998 non-null   object
 6   gender                 3998 non-null   object
 7   dob                    3998 non-null   object
 8   10percentage           3998 non-null   float64
 9   10board                3998 non-null   object
 10  12graduation           3998 non-null   int64
 11  12percentage           3998 non-null   float64
 12  12board                3998 non-null   object
 13  collegeid              3998 non-null   int64
 14  collegetier            3998 non-null   int64
 15  degree                 3998 non-null   object
 16  specialization         3998 non-null   object
 17  collegegpa             3998 non-null   float64
 18  collegecityid          3998 non-null   int64
 19  collegecitytier        3998 non-null   int64
 20  collegestate           3998 non-null   object
 21  graduationyear         3998 non-null   int64
 22  english                3998 non-null   int64
 23  logical                3998 non-null   int64
 24  quant                  3998 non-null   int64
 25  domain                 3998 non-null   float64
 26  computerprogramming    3998 non-null   int64
 27  electronicsandsemicon  3998 non-null   int64
 28  computerscience        3998 non-null   int64
 29  mechanicalengg         3998 non-null   int64
 30  electricalengg         3998 non-null   int64
 31  telecomengg            3998 non-null   int64
 32  civilengg              3998 non-null   int64
 33  conscientiousness      3998 non-null   float64
 34  agreeableness          3998 non-null   float64
 35  extraversion           3998 non-null   float64
 36  nueroticism            3998 non-null   float64
 37  openess_to_experience  3998 non-null   float64
dtypes: datetime64[ns](1), float64(10), int64(17), object(10)
memory usage: 1.2+ MB
```

In [14]:
```
df.shape
```

Out[14]: (3998, 38)

1.DATA CLEANING

In [15]:
```
unique_cities = df['jobcity'].unique()
unique_cities
```

Out[15]:
```
array(['Bangalore', 'Indore', 'Chennai', 'Gurgaon', 'Manesar',
       'Hyderabad', 'Banglore', 'Noida', 'Kolkata', 'Pune', '-1',
       'mohali', 'Jhansi', 'Delhi', 'Hyderabad ', 'Bangalore ', 'noida',
       'delhi', 'Bhubaneswar', 'Navi Mumbai', 'Mumbai', 'New Delhi',
       'Mangalore', 'Rewari', 'Gaziabaad', 'Bhiwadi', 'Mysore', 'Rajkot',
       'Greater Noida', 'Jaipur', 'noida ', 'HYDERABAD', 'mysore',
       'THANE', 'Maharajganj', 'Thiruvananthapuram', 'Punchkula',
       'Bhubaneshwar', 'Pune ', 'coimbatore', 'Dhanbad', 'Lucknow',
       'Trivandrum', 'kolkata', 'mumbai', 'Gandhi Nagar', 'Una',
       'Daman and Diu', 'chennai', 'GURGOAN', 'vsakhapttnam', 'pune',
       'Nagpur', 'Bhagalpur', 'new delhi - jaisalmer', 'Coimbatore',
       'Ahmedabad', 'Kochi/Cochin', 'Bankura', 'Bengaluru', 'Mysore ',
       'Kanpur ', 'jaipur', 'Gurgaon ', 'bangalore', 'CHENNAI',
       'Vijayawada', 'Kochi', 'Beawar', 'Alwar', 'NOIDA', 'Greater noid
a',
       'Siliguri ', 'raipur', 'gurgaon', 'Bhopal', 'Faridabad', 'Jodhpu
r',
       'udaipur', 'Muzaffarpur', 'Kolkata`', 'Bulandshahar', 'Haridwar',
       'Raigarh', 'Visakhapatnam', 'Jabalpur', 'hyderabad', 'Unnao',
       'KOLKATA', 'Thane', 'Aurangabad', 'Belgaum', 'gurgoan', 'Dehradu
n',
       'Rudrapur', 'Jamshedpur', 'vizag', 'Nouda', 'Dharamshala',
       'Banagalore', 'Hissar', 'Ranchi', 'BANGALORE', 'Madurai', 'Gurga',
       'Chandigarh', 'Australia', ' Chennai', 'CHEYYAR', 'Mumbai ',
       'sonepat', 'Ghaziabad', 'Pantnagar', 'Siliguri', 'mumbai ',
       'Jagdalpur', 'Chennai ', 'angul', 'Baroda', ' ariyalur', 'Jowai',
       'Kochi/Cochin, Chennai and Coimbatore', 'bhubaneswar', 'Neemrana',
       'VIZAG', 'Tirupathi', 'Lucknow ', 'Ahmedabad ', 'Bhubneshwar',
       'Noida ', 'pune ', 'Calicut', 'Gandhinagar', 'LUCKNOW', 'Dubai',
       'bengaluru', 'MUMBAI', 'Ahmednagar', 'Nashik', 'New delhi',
       'Bellary', 'Ludhiana', 'New Delhi ', 'Muzaffarnagar', 'BHOPAL',
       'Gurgoan', 'Gagret', 'Indirapuram, Ghaziabad', 'Gwalior',
       'new delhi', 'TRIVANDRUM', 'Chennai & Mumbai', 'Rajasthan',
       'Sonipat', 'Bareli', 'Kanpur', 'Hospete', 'Miryalaguda', ' mumba
i',
       'Dharuhera', 'lucknow', 'meerut', 'dehradun', 'Ganjam', 'Hubli',
       'bangalore ', 'NAVI MUMBAI', 'ncr', 'Agra', 'Trichy',
       'kudankulam ,tarapur', 'Ongole', 'Sambalpur', 'Pondicherry',
       'Bundi', 'SADULPUR,RAJGARH,DISTT-CHURU,RAJASTHAN', 'AM', 'Bikane
r',
       'Vadodara', 'BAngalore', 'india', 'Asansol', 'Tirunelvelli',
       'Ernakulam', 'DELHI', 'Bilaspur', 'Chandrapur', 'Nanded',
       'Dharmapuri', 'Vandavasi', 'Rohtak', 'trivandrum', 'Nagpur ',
       'Udaipur', 'Patna', 'banglore', 'indore', 'Salem', 'Nasikcity',
       'Gandhinagar ', 'Technopark, Trivandrum', 'Bharuch', 'Tornagallu',
       'Raipur', 'Kolkata ', 'Jaspur', 'Burdwan', 'Bhubaneswar ',
       'Shimla', 'ahmedabad', 'Gajiabaad', 'Jammu', 'Shahdol',
       'Muvattupuzha', 'Al Jubail,Saudi Arabia', 'Kalmar, Sweden',
       'Secunderabad', 'A-64,sec-64,noida', 'Ratnagiri', 'Jhajjar',
       'Gulbarga', 'hyderabad(bhadurpally)', 'Nalagarh', 'Chandigarh ',
       'Jaipur ', 'Jeddah Saudi Arabia', ' Delhi', 'PATNA', 'SHAHDOL',
       'Chennai, Bangalore', 'Bhopal ', 'Jamnagar', 'PUNE', 'Tirupati',
       'Gonda', 'jamnagar', 'chennai ', 'orissa', 'kharagpur',
       'Trivandrum ', 'Navi Mumbai , Hyderabad', 'Joshimath',
       'chandigarh', 'Bathinda', 'Johannesburg', 'kala amb ', 'Karnal',
       'LONDON', 'Kota', 'Panchkula', 'Baddi HP', 'Nagari',
       'Mettur, Tamil Nadu ', 'Durgapur', 'pondi', 'Surat', 'Kurnool',
       'kolhapur', 'Madurai ', 'GREATER NOIDA', 'Bhilai', ' Pune',
       'hderabad', 'KOTA', 'thane', 'Vizag', 'Bahadurgarh',
       'Rayagada, Odisha', 'kakinada', 'GURGAON', 'Varanasi', 'punr',
       'Nellore', 'patna', 'Meerut', 'hyderabad ', 'Sahibabad', 'Howrah',
```

```
       'BHUBANESWAR', 'Trichur', 'Ambala', 'Khopoli', 'keral', 'Roorkee',
       'Greater NOIDA', 'Navi mumbai', 'ghaziabad', 'Allahabad',
       'Delhi/NCR', 'Panchkula ', 'Ranchi ', 'Jalandhar', 'manesar',
       'vapi', 'PILANI', 'muzzafarpur', 'RAS AL KHAIMAH', 'bihar',
       'singaruli', 'KANPUR', 'Banglore ', 'pondy', 'Mohali', 'Phagwara',
       ' Mumbai', ' bangalore', 'GURAGAON', 'Baripada', 'MEERUT',
       'Yamuna Nagar', 'shahibabad', 'sampla', 'Guwahati', 'Rourkela',
       'Banaglore', 'Vellore', 'Dausa', 'latur (Maharashtra )',
       'NEW DELHI', 'kanpur', 'Mainpuri', 'karnal', 'Dammam', 'Haldia',
       'sambalpur', 'RAE BARELI', 'ranchi', 'jAipur', 'BANGLORE',
       'Patiala', 'Gorakhpur', 'new dehli', 'BANGALORE ', 'Ambala City',
       'Karad', 'Rajpura', 'Pilani', 'haryana', 'Asifabadbanglore'],
      dtype=object)
```

In [16]:
```
df.jobcity = df.jobcity.str.strip().str.lower()

unique_cities_cleaned = df['jobcity'].unique()
print(unique_cities_cleaned)
```

```
['bangalore' 'indore' 'chennai' 'gurgaon' 'manesar' 'hyderabad' 'banglor
e'
 'noida' 'kolkata' 'pune' '-1' 'mohali' 'jhansi' 'delhi' 'bhubaneswar'
 'navi mumbai' 'mumbai' 'new delhi' 'mangalore' 'rewari' 'gaziabaad'
 'bhiwadi' 'mysore' 'rajkot' 'greater noida' 'jaipur' 'thane'
 'maharajganj' 'thiruvananthapuram' 'punchkula' 'bhubaneshwar'
 'coimbatore' 'dhanbad' 'lucknow' 'trivandrum' 'gandhi nagar' 'una'
 'daman and diu' 'gurgoan' 'vsakhapttnam' 'nagpur' 'bhagalpur'
 'new delhi - jaisalmer' 'ahmedabad' 'kochi/cochin' 'bankura' 'bengaluru'
 'kanpur' 'vijayawada' 'kochi' 'beawar' 'alwar' 'siliguri' 'raipur'
 'bhopal' 'faridabad' 'jodhpur' 'udaipur' 'muzaffarpur' 'kolkata``'
 'bulandshahar' 'haridwar' 'raigarh' 'visakhapatnam' 'jabalpur' 'unnao'
 'aurangabad' 'belgaum' 'dehradun' 'rudrapur' 'jamshedpur' 'vizag' 'noud
a'
 'dharamshala' 'banagalore' 'hissar' 'ranchi' 'madurai' 'gurga'
 'chandigarh' 'australia' 'cheyyar' 'sonepat' 'ghaziabad' 'pantnagar'
 'jagdalpur' 'angul' 'baroda' 'ariyalur' 'jowai'
 'kochi/cochin, chennai and coimbatore' 'neemrana' 'tirupathi'
 'bhubneshwar' 'calicut' 'gandhinagar' 'dubai' 'ahmednagar' 'nashik'
 'bellary' 'ludhiana' 'muzaffarnagar' 'gagret' 'indirapuram, ghaziabad'
 'gwalior' 'chennai & mumbai' 'rajasthan' 'sonipat' 'bareli' 'hospete'
 'miryalaguda' 'dharuhera' 'meerut' 'ganjam' 'hubli' 'ncr' 'agra' 'trich
y'
 'kudankulam ,tarapur' 'ongole' 'sambalpur' 'pondicherry' 'bundi'
 'sadulpur,rajgarh,distt-churu,rajasthan' 'am' 'bikaner' 'vadodara'
 'india' 'asansol' 'tirunelvelli' 'ernakulam' 'bilaspur' 'chandrapur'
 'nanded' 'dharmapuri' 'vandavasi' 'rohtak' 'patna' 'salem' 'nasikcity'
 'technopark, trivandrum' 'bharuch' 'tornagallu' 'jaspur' 'burdwan'
 'shimla' 'gajiabaad' 'jammu' 'shahdol' 'muvattupuzha'
 'al jubail,saudi arabia' 'kalmar, sweden' 'secunderabad'
 'a-64,sec-64,noida' 'ratnagiri' 'jhajjar' 'gulbarga'
 'hyderabad(bhadurpally)' 'nalagarh' 'jeddah saudi arabia'
 'chennai, bangalore' 'jamnagar' 'tirupati' 'gonda' 'orissa' 'kharagpur'
 'navi mumbai , hyderabad' 'joshimath' 'bathinda' 'johannesburg'
 'kala amb' 'karnal' 'london' 'kota' 'panchkula' 'baddi hp' 'nagari'
 'mettur, tamil nadu' 'durgapur' 'pondi' 'surat' 'kurnool' 'kolhapur'
 'bhilai' 'hderabad' 'bahadurgarh' 'rayagada, odisha' 'kakinada'
 'varanasi' 'punr' 'nellore' 'sahibabad' 'howrah' 'trichur' 'ambala'
 'khopoli' 'keral' 'roorkee' 'allahabad' 'delhi/ncr' 'jalandhar' 'vapi'
 'pilani' 'muzzafarpur' 'ras al khaimah' 'bihar' 'singaruli' 'pondy'
 'phagwara' 'guragaon' 'baripada' 'yamuna nagar' 'shahibabad' 'sampla'
 'guwahati' 'rourkela' 'banaglore' 'vellore' 'dausa'
 'latur (maharashtra )' 'mainpuri' 'dammam' 'haldia' 'rae bareli'
 'patiala' 'gorakhpur' 'new dehli' 'ambala city' 'karad' 'rajpura'
 'haryana' 'asifabadbanglore']
```

In [3]:
```
city_mapping = {
    'bangalore':'Bangalore',
    'banglore': 'Bangalore',
    'banagalore':'Bangalore',
    'bengaluru':'Bangalore',
    'asifabadbanglore':'Bangalore',
    'indore':'Indore',
    'chennai':'Chennai',
    'gurgaon':'Gurgaon',
    'gurgoan':'Gurgaon',
    'gurga':'Gurgaon',
    'manesar':'Manesar',
    'hyderabad':'Hyderabad',
    'hderabad':'Hyderabad',
    'hyderabad(bhadurpally)' :'Hyderabad',
    'noida':'Noida',
    'nouda':'Noida',
     'kolkata': 'Kolkata',
 'kolkata`': 'Kolkata',
 'pune': 'Pune',
 '-1': 'Unknown',
 'mohali': 'Mohali',
 'jhansi': 'Jhansi',
 'delhi': 'Delhi',
 'new delhi': 'New Delhi',
 'bhubaneswar': 'Bhubaneswar',
 'bhubaneshwar': 'Bhubaneswar',
 'navi mumbai': 'Navi Mumbai',
 'mumbai': 'Mumbai',
 'mangalore': 'Mangalore',
 'rewari': 'Rewari',
 'gaziabaad': 'Ghaziabad',
 'ghaziabad': 'Ghaziabad',
 'bhiwadi': 'Bhiwadi',
 'mysore': 'Mysore',
 'rajkot': 'Rajkot',
 'greater noida': 'Greater Noida',
 'jaipur': 'Jaipur',
 'thane': 'Thane',
 'maharajganj': 'Maharajganj',
 'thiruvananthapuram': 'Thiruvananthapuram',
 'punchkula': 'Panchkula',
 'coimbatore': 'Coimbatore',
 'dhanbad': 'Dhanbad',
 'lucknow': 'Lucknow',
 'trivandrum': 'Thiruvananthapuram',
 'gandhi nagar': 'Gandhinagar',
 'una': 'Una',
 'daman and diu': 'Daman and Diu',
 'vsakhapttnam': 'Visakhapatnam',
 'nagpur': 'Nagpur',
'bhagalpur': 'Bhagalpur',
 'new delhi- jaisalmer': 'New Delhi',
 'ahmedabad': 'Ahmedabad',
 'kochi/cochin': 'Kochi',
 'bankura': 'Bankura',
 'kanpur': 'Kanpur',
 'vijayawada': 'Vijayawada',
 'kochi': 'Kochi',
 'beawar': 'Beawar',
 'alwar': 'Alwar',
```

```
 'siliguri': 'Siliguri',
 'raipur': 'Raipur',
 'bhopal': 'Bhopal',
 'faridabad': 'Faridabad',
 'jodhpur': 'Jodhpur',
 'udaipur': 'Udaipur',
 'muzaffarpur': 'Muzaffarpur',
 'bulandshahar': 'Bulandshahar',
 'haridwar': 'Haridwar',
 'raigarh': 'Raigarh',
 'visakhapatnam': 'Visakhapatnam',
 'jabalpur': 'Jabalpur',
 'unnao': 'Unnao',
 'aurangabad': 'Aurangabad',
 'belgaum': 'Belgaum',
 'dehradun': 'Dehradun',
 'rudrapur': 'Rudrapur',
 'jamshedpur': 'Jamshedpur',
 'vizag': 'Visakhapatnam',
 'nouda': 'Noida',
 'dharamshala': 'Dharamshala',
 'hissar': 'Hisar',
 'ranchi': 'Ranchi',
 'madurai': 'Madurai',
 'chandigarh': 'Chandigarh',
 'australia': 'Australia',
 'cheyyar': 'Cheyyar',
 'sonepat': 'Sonepat',
 'pantnagar': 'Pantnagar',
 'jagdalpur': 'Jagdalpur',
 'angul': 'Angul',
 'baroda': 'Vadodara',
 'ariyalur': 'Ariyalur',
 'jowai': 'Jowai',
 'neemrana': 'Neemrana',
 'tirupathi': 'Tirupati',
 'bhubneshwar': 'Bhubaneswar',
'calicut': 'Kozhikode',
 'gandhinagar': 'Gandhinagar',
 'dubai': 'Dubai',
 'ahmednagar': 'Ahmednagar',
 'nashik': 'Nashik',
 'bellary': 'Bellary',
 'ludhiana': 'Ludhiana',
 'muzaffarnagar': 'Muzaffarnagar',
 'gagret': 'Gagret',
 'indirapuram, ghaziabad': 'Ghaziabad',
 'gwalior': 'Gwalior',
 'chennai & mumbai': 'Chennai',
 'rajasthan': 'Rajasthan',
 'sonipat': 'Sonipat',
 'bareli': 'Bareli',
 'hospete': 'Hospete',
 'miryalaguda': 'Miryalaguda',
 'dharuhera': 'Dharuhera',
 'meerut': 'Meerut',
 'ganjam': 'Ganjam',
 'hubli': 'Hubli',
 'ncr': 'NCR',
 'agra': 'Agra',
 'trichy': 'Tiruchirappalli',
```

```
 'kudankulam ,tarapur': 'Kudankulam',
 'ongole': 'Ongole',
 'sambalpur': 'Sambalpur',
 'pondicherry': 'Puducherry',
 'bundi': 'Bundi',
 'sadulpur,rajgarh,distt-churu,rajasthan': 'Rajasthan',
 'am': 'Am',
 'bikaner': 'Bikaner',
 'vadodara': 'Vadodara',
 'india': 'India',
 'asansol': 'Asansol',
 'tirunelvelli': 'Tirunelveli',
 'ernakulam': 'Ernakulam',
 'bilaspur': 'Bilaspur',
 'chandrapur': 'Chandrapur',
 'nanded': 'Nanded',
 'dharmapuri': 'Dharmapuri',
 'vandavasi': 'Vandavasi',
 'rohtak': 'Rohtak',
 'patna': 'Patna',
 'salem': 'Salem',
 'nasikcity': 'Nashik',
 'technopark, trivandrum': 'Trivandrum',
'bharuch': 'Bharuch',
 'tornagallu': 'Tornagallu',
 'jaspur': 'Jaspur',
 'burdwan': 'Burdwan',
 'shimla': 'Shimla',
 'gajiabaad': 'Ghaziabad',
 'jammu': 'Jammu',
 'shahdol': 'Shahdol',
 'muvattupuzha': 'Muvattupuzha',
 'al jubail,saudi arabia': 'Al Jubail',
 'kalmar, sweden': 'Kalmar',
 'secunderabad': 'Secunderabad',
 'a-64,sec-64,noida': 'Noida',
 'ratnagiri': 'Ratnagiri',
 'jhajjar': 'Jhajjar',
 'gulbarga': 'Gulbarga',
 'hyderabad(bhadurpally)': 'Hyderabad',
 'nalagarh': 'Nalagarh',
 'jeddah saudi arabia': 'Jeddah',
 'chennai, bangalore': 'Chennai',
 'jamnagar': 'Jamnagar',
 'tirupati': 'Tirupati',
 'gonda': 'Gonda',
 'orissa': 'Odisha',
 'kharagpur': 'Kharagpur',
 'navi mumbai , hyderabad': 'Navi Mumbai',
 'joshimath': 'Joshimath',
 'bathinda': 'Bathinda',
 'johannesburg': 'Johannesburg',
 'kala amb': 'Kala Amb',
 'karnal': 'Karnal',
 'london': 'London',
 'kota': 'Kota',
 'dehraj': 'Dehradun',
 'melbourne': 'Melbourne',
 'moradabad': 'Moradabad',
 'delhi-gurgaon': 'Delhi',
 'ambala': 'Ambala',
```

```
 'faridkot': 'Faridkot',
 'rohtak, haryana': 'Rohtak',
 'khammam': 'Khammam',
 'khurda': 'Khurda',
 'jhalawar': 'Jhalawar',
 'kaithal': 'Kaithal',
 'sonbhadra': 'Sonbhadra',
 'fatehgarh sahib': 'Fatehgarh Sahib',
 'kaithal-haryana': 'Kaithal',
'bhilwara': 'Bhilwara',
 'coimbatore, tirupur': 'Coimbatore',
 'sri ganganagar': 'Sri Ganganagar',
 'manipal': 'Manipal',
 'tirupathi': 'Tirupati',
 'kharagpur, west bengal': 'Kharagpur',
 'kolkata': 'Kolkata',
 'trichy-tiruchirappalli': 'Tiruchirappalli',
}
```

In [17]:
```
df['jobcity'] = df['jobcity'].replace(city_mapping)
```

In [18]:
```
df['jobcity'] = df.jobcity.str.strip().str.lower()
```

In [19]:
```
df
```

Out[19]:

	id	salary	doj	dol	designation	jobcity	gender	dob	10percentag
0	203097	420000.0	2012-01-06	present	senior quality engineer	bangalore	f	19-02-1990 00:00	84.3
1	579905	500000.0	2013-01-09	present	assistant manager	indore	m	04-10-1989 00:00	85.4
2	810601	325000.0	2014-01-06	present	systems engineer	chennai	f	03-08-1992 00:00	85.0
3	267447	1100000.0	2011-01-07	present	senior software engineer	gurgaon	m	05-12-1989 00:00	85.6
4	343523	200000.0	2014-01-03	01-03-2015 00:00	get	manesar	m	27-02-1991 00:00	78.0
...	...	...	...	...	...	...	...	...	.
3993	47916	280000.0	2011-01-10	01-10-2012 00:00	software engineer	new delhi	m	15-04-1987 00:00	52.0
3994	752781	100000.0	2013-01-07	01-07-2013 00:00	technical writer	hyderabad	f	27-08-1992 00:00	90.0
3995	355888	320000.0	2013-01-07	present	associate software engineer	bangalore	m	03-07-1991 00:00	81.8
3996	947111	200000.0	2014-01-07	01-01-2015 00:00	software developer	bangalore	f	20-03-1992 00:00	78.7
3997	324966	400000.0	2013-01-02	present	senior systems engineer	chennai	f	26-02-1991 00:00	70.6

3998 rows × 38 columns

In [24]:
```
# replace the date values with "left " in dol
df['dol'] = df['dol'].apply(lambda x: "Left" if x != "present" else x)
```

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

Out[25]:

dol	designation	jobcity	gender	dob	10percentage	10board	...	computerscience
present	senior quality engineer	bangalore	f	19-02-1990 00:00	84.3	board ofsecondary education,ap	...	-1
present	assistant manager	indore	m	04-10-1989 00:00	85.4	cbse	...	-1
present	systems engineer	chennai	f	03-08-1992 00:00	85.0	cbse	...	-1
present	senior software engineer	gurgaon	m	05-12-1989 00:00	85.6	cbse	...	-1
Left	get	manesar	m	27-02-1991 00:00	78.0	cbse	...	-1

In [26]:
```
df['dol'].value_counts()
```

Out[26]:
```
Left       2123
present    1875
Name: dol, dtype: int64
```

In [27]:
```
df['salary'].mean().round(2)
```

Out[27]:
```
307699.85
```

In [28]:
```
df['salary'].max()
```

Out[28]:
```
4000000.0
```

In [29]:
```
df['salary'].min()
```

Out[29]:
```
35000.0
```

In [30]:
```
df['gender'].value_counts()
```

Out[30]:
```
m    3041
f     957
Name: gender, dtype: int64
```

In [32]:
```
df['computerscience'] = df['computerscience'].replace(-1,0)
df['mechanicalengg'] = df['mechanicalengg'].replace(-1,0)
df['electricalengg'] = df['electricalengg'].replace(-1,0)
df['telecomengg'] = df['telecomengg'].replace(-1,0)
df['civilengg'] = df['civilengg'].replace(-1,0)
```

In [33]:
```
df.head()
```

Out[33]:

computerscience	mechanicalengg	electricalengg	telecomengg	civilengg	conscientiousness	a
0	0	0	0	0	0.9737	
0	0	0	0	0	-0.7335	
0	0	0	0	0	0.2718	
0	0	0	0	0	0.0464	
0	0	0	0	0	-0.8810	

In [35]:
```
df['salary'].describe()
```

Out[35]:
```
count    3.998000e+03
mean     3.076998e+05
std      2.127375e+05
min      3.500000e+04
25%      1.800000e+05
50%      3.000000e+05
75%      3.700000e+05
max      4.000000e+06
Name: salary, dtype: float64
```

In [36]:
```
pd.options.display.float_format = '{:,.0f}'.format

#display  the describe() outputfor the 'salary' column
df.describe().transpose()
```

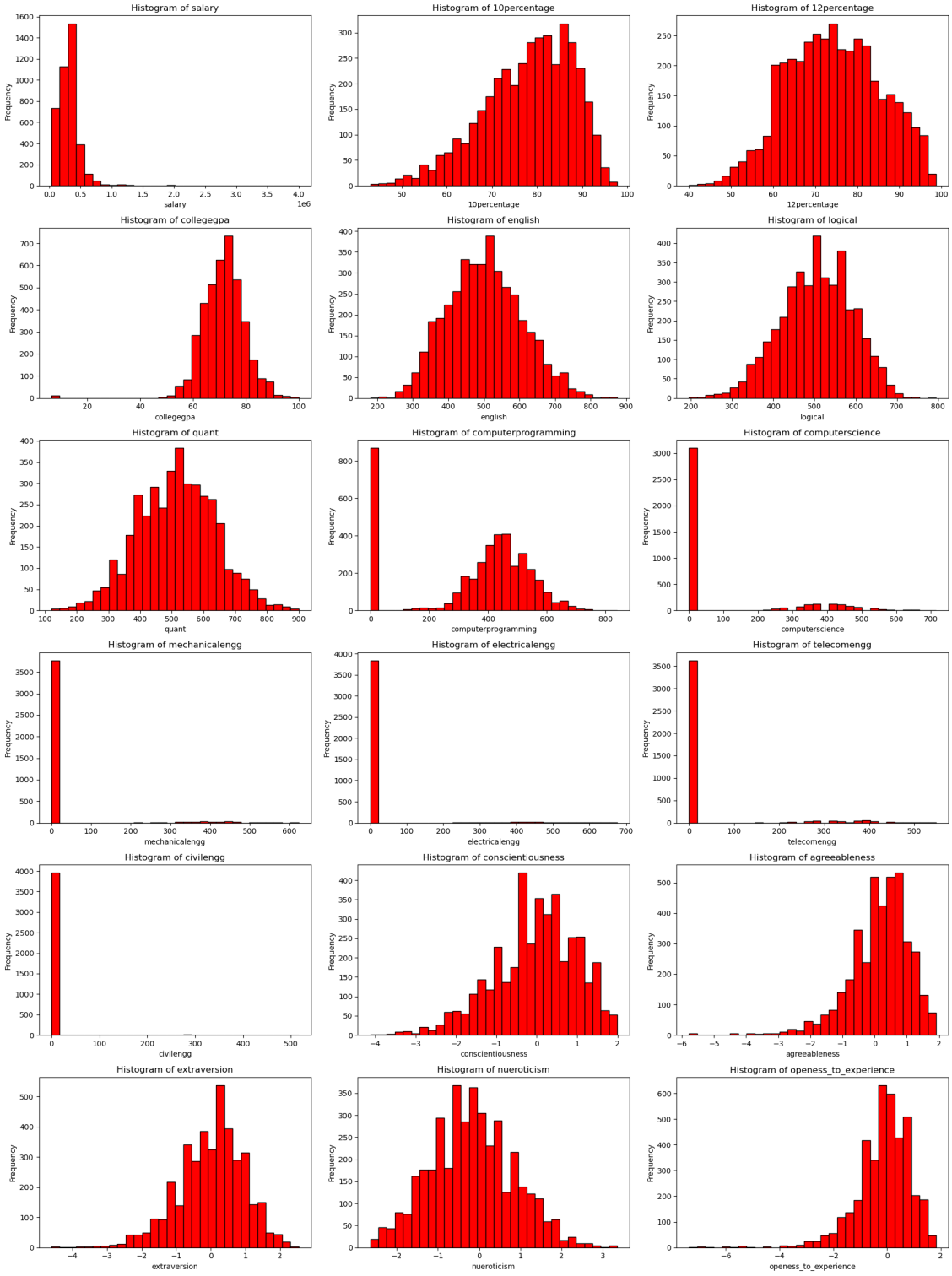
Out[36]:

	count	mean	std	min	25%	50%	75%	ma
id	3,998	663,795	363,218	11,244	334,284	639,600	990,480	1,298,27
salary	3,998	307,700	212,737	35,000	180,000	300,000	370,000	4,000,00
10percentage	3,998	78	10	43	72	79	86	9
12graduation	3,998	2,008	2	1,995	2,007	2,008	2,009	2,01
12percentage	3,998	74	11	40	66	74	83	9
collegeid	3,998	5,157	4,802	2	494	3,879	8,818	18,40
collegetier	3,998	2	0	1	2	2	2	
collegegpa	3,998	71	8	6	66	72	76	10
collegecityid	3,998	5,157	4,802	2	494	3,879	8,818	18,40
collegecitytier	3,998	0	0	0	0	0	1	
graduationyear	3,998	2,012	32	0	2,012	2,013	2,014	2,01
english	3,998	502	105	180	425	500	570	87
logical	3,998	502	87	195	445	505	565	79
quant	3,998	513	122	120	430	515	595	90
domain	3,998	1	0	-1	0	1	1	
computerprogramming	3,998	353	205	-1	295	415	495	84
electronicsandsemicon	3,998	95	158	-1	-1	-1	233	61
computerscience	3,998	92	175	0	0	0	0	71
mechanicalengg	3,998	24	98	0	0	0	0	62
electricalengg	3,998	17	87	0	0	0	0	67
telecomengg	3,998	33	105	0	0	0	0	54
civilengg	3,998	4	37	0	0	0	0	51
conscientiousness	3,998	-0	1	-4	-1	0	1	
agreeableness	3,998	0	1	-6	-0	0	1	
extraversion	3,998	0	1	-5	-1	0	1	
nueroticism	3,998	-0	1	-3	-1	-0	1	
openess_to_experience	3,998	-0	1	-7	-1	-0	1	

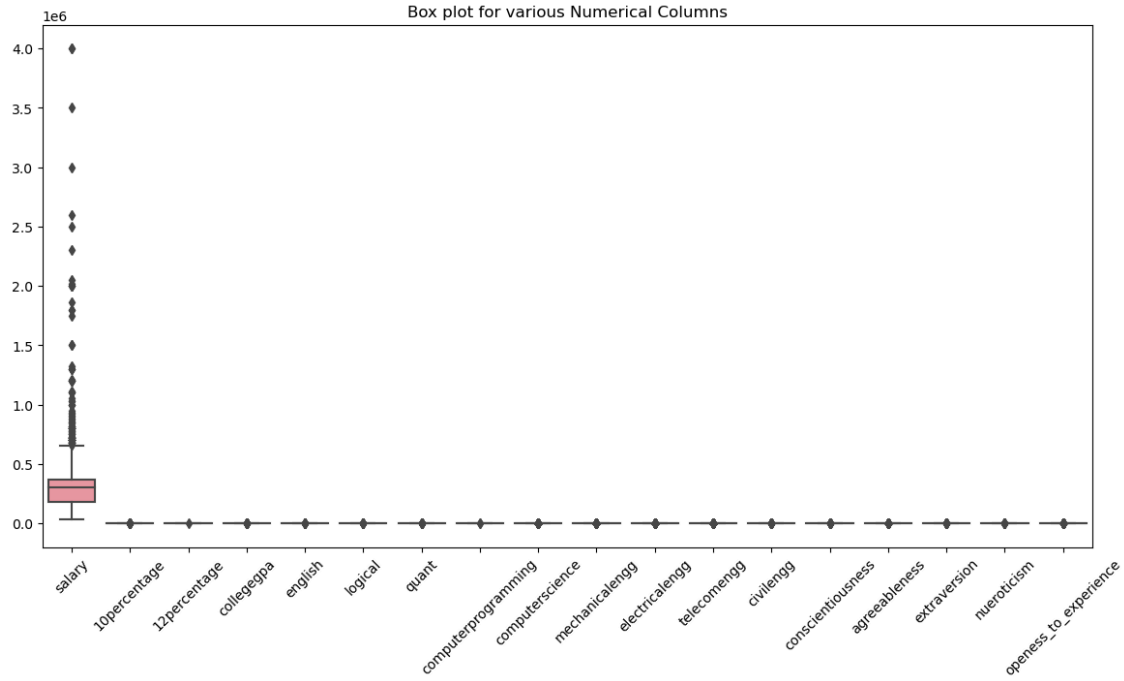
In [37]:
```
df.columns
```

Out[37]:
```
Index(['id', 'salary', 'doj', 'dol', 'designation', 'jobcity', 'gender',
'dob',
       '10percentage', '10board', '12graduation', '12percentage', '12boar
d',
       'collegeid', 'collegetier', 'degree', 'specialization', 'collegegp
a',
       'collegecityid', 'collegecitytier', 'collegestate', 'graduationyea
r',
       'english', 'logical', 'quant', 'domain', 'computerprogramming',
       'electronicsandsemicon', 'computerscience', 'mechanicalengg',
       'electricalengg', 'telecomengg', 'civilengg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience'],
      dtype='object')
```

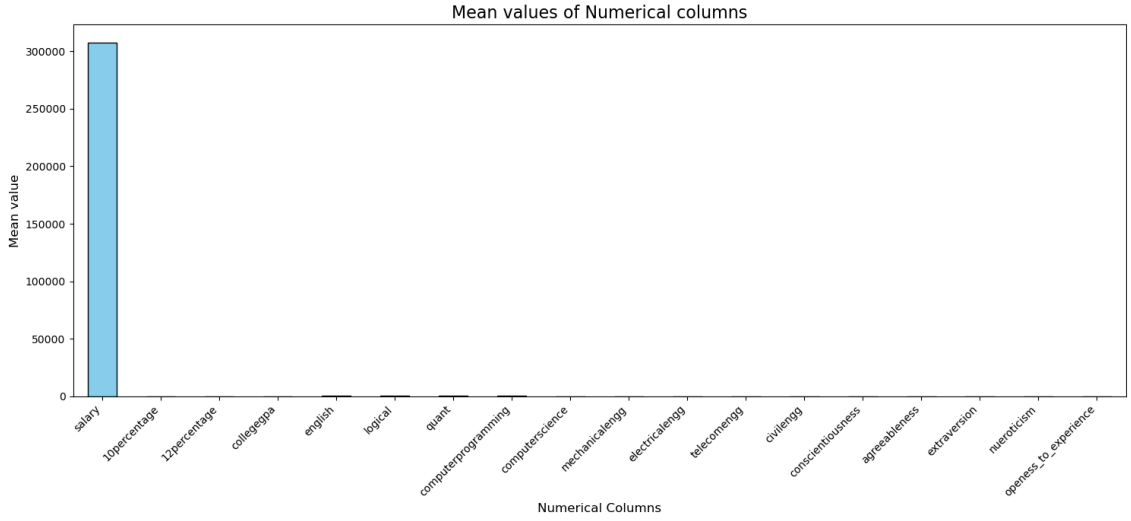
In [45]:
```
# select the columns you want to plot
columns_to_plot = ['salary' , '10percentage' , '12percentage','collegegpa'
                   'quant','computerprogramming','computerscience','mechani
                   'civilengg','conscientiousness',
 'agreeableness', 'extraversion', 'nueroticism',
 'openess_to_experience']

# set up the figure and axes for subplots
fig , axes = plt.subplots(nrows=6, ncols=3, figsize=(18,24))
axes = axes.flatten()  #flatten the 2D array of axes into 1D for easier it

# loop through each column and its respective axis
for i , column in enumerate(columns_to_plot):
    axes[i].hist(df[column].dropna(),bins=30,color='red',edgecolor='black'
    axes[i].set_title(f'Histogram of {column}')  # set each title for subp
    axes[i].set_xlabel(column)  #X - axis label
    axes[i].set_ylabel('Frequency')  # y- axis label


# remove any unused subplots (if there are more axes than columns)
for j in range(i+1,len(axes)):
    fig.delaxes(axes[j])

# Adjust layout to prevent overlapping
plt.tight_layout()

#show the plot
plt.show()
```


Histogram of salary
Frequency
salary
1e6
Histogram of 10percentage
10percentage
Histogram of 12percentage
12percentage
Histogram of collegegpa
collegegpa
Histogram of english
english
Histogram of logical
logical
Histogram of quant
quant
Histogram of computerprogramming
computerprogramming
Histogram of computerscience
computerscience
Histogram of mechanicalengg
mechanicalengg
Histogram of electricalengg
electricalengg
Histogram of telecomengg
telecomengg
Histogram of civilengg
civilengg
Histogram of conscientiousness
conscientiousness
Histogram of agreeableness
agreeableness
Histogram of extraversion
extraversion
Histogram of nueroticism
nueroticism
Histogram of openess_to_experience
openess_to_experience

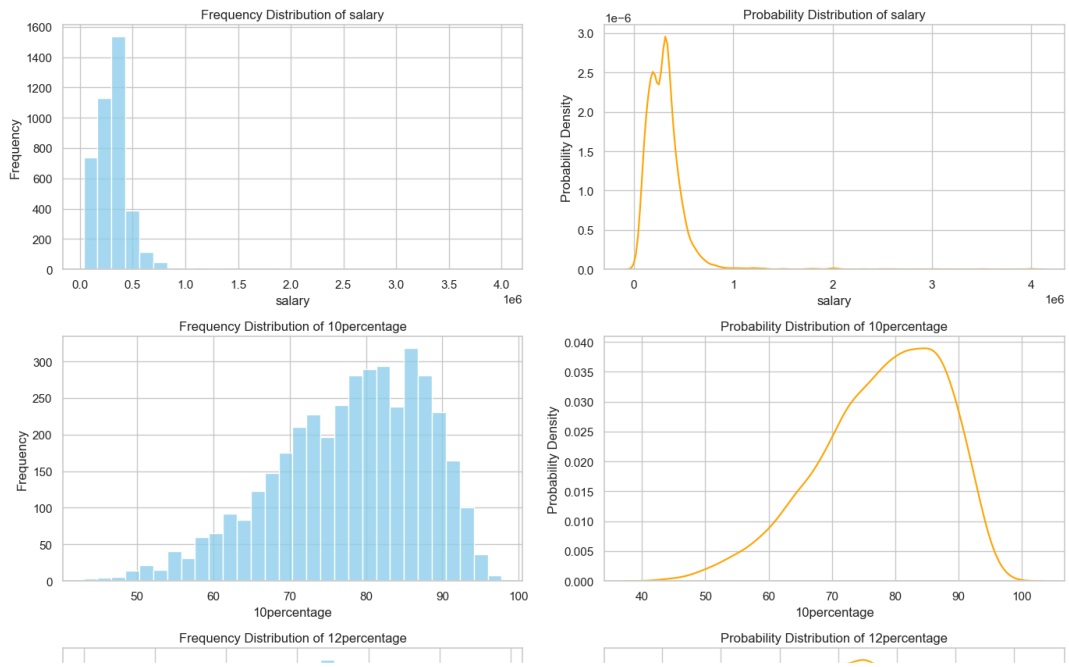
In [46]:
```
# correct list of columns  to plot (only numerical columns )
columns_to_plot =  ['salary', '10percentage', '12percentage', 'collegegpa'
 'english', 'logical', 'quant', 'computerprogramming',
 'computerscience', 'mechanicalengg', 'electricalengg',
 'telecomengg', 'civilengg', 'conscientiousness',
 'agreeableness', 'extraversion', 'nueroticism',
 'openess_to_experience']

# plot the box plot with valid columns
plt.figure(figsize=(14,7))
sns.boxplot(data=df[columns_to_plot])
plt.title('Box plot for various Numerical Columns')
plt.xticks(rotation=45)
plt.show()
```



In [53]:
```
import matplotlib.pyplot as plt
# select only numerical values
columns_to_plot = ['salary', '10percentage', '12percentage', 'collegegpa',
 'english', 'logical',
 'quant', 'computerprogramming', 'computerscience',
 'mechanicalengg',
'electricalengg', 'telecomengg', 'civilengg',
 'conscientiousness',
 'agreeableness', 'extraversion', 'nueroticism',
 'openess_to_experience']

# calculate the mean of each numerical column
mean_values = df[columns_to_plot].mean()

# create the bar plot
plt.figure(figsize=(15,7))  # set the fig size
mean_values.plot(kind='bar',color='skyblue',edgecolor='black')

#customize the plot
plt.title('Mean values of Numerical columns',fontsize=16)
plt.xlabel('Numerical Columns' , fontsize=12)
plt.ylabel('Mean value',fontsize=12)
plt.xticks(rotation=45, ha='right') #rotate x labesl for better visibility

#show the plot
plt.tight_layout()
plt.show()
```

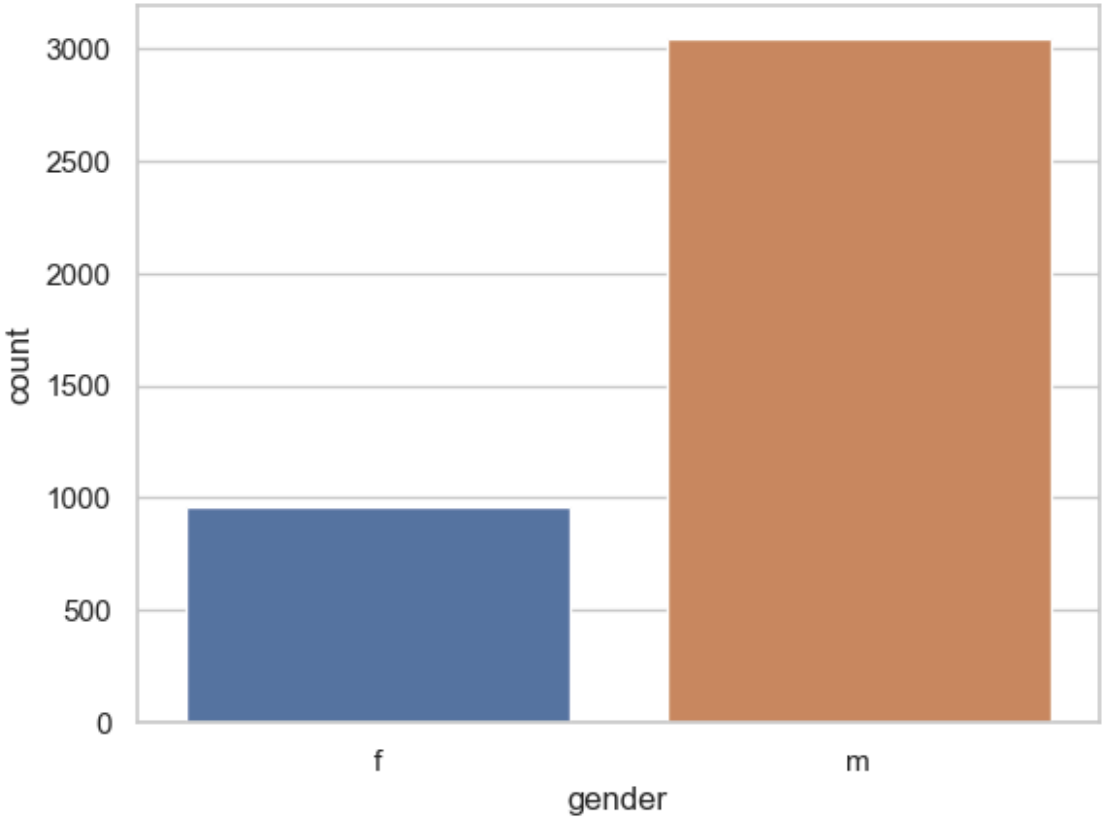


In [61]:
```
# Set the style of seaborn
sns.set(style="whitegrid")

 # Define the columns for plotting
columns_to_plot = ['salary', '10percentage', '12percentage', 'collegegpa',
 'english', 'logical',
'quant', 'computerprogramming', 'computerscience',
 'mechanicalengg',
 'electricalengg', 'telecomengg', 'civilengg',
 'conscientiousness',
 'agreeableness', 'extraversion', 'nueroticism',
 'openess_to_experience']

 # Create a figure with subplots
fig, axes = plt.subplots(nrows=len(columns_to_plot), ncols=2, figsize=(14,

 # Loop through each numerical column to plot
for i, column in enumerate(columns_to_plot):

 # Frequency Distribution
 sns.histplot(df[column], ax=axes[i, 0], bins=30, kde=False, color='skyblu
 axes[i, 0].set_title(f'Frequency Distribution of {column}', fontsize=12)
 axes[i, 0].set_xlabel(column)
 axes[i, 0].set_ylabel('Frequency')

 # Probability Distribution (KDE)
 sns.kdeplot(df[column], ax=axes[i, 1], color='orange')
 axes[i, 1].set_title(f'Probability Distribution of {column}', fontsize=12
 axes[i, 1].set_xlabel(column)
 axes[i, 1].set_ylabel('Probability Density')

 # Adjust layout
plt.tight_layout()
plt.show()
```



In [62]:
```
sns.countplot(x=df['gender'])
```

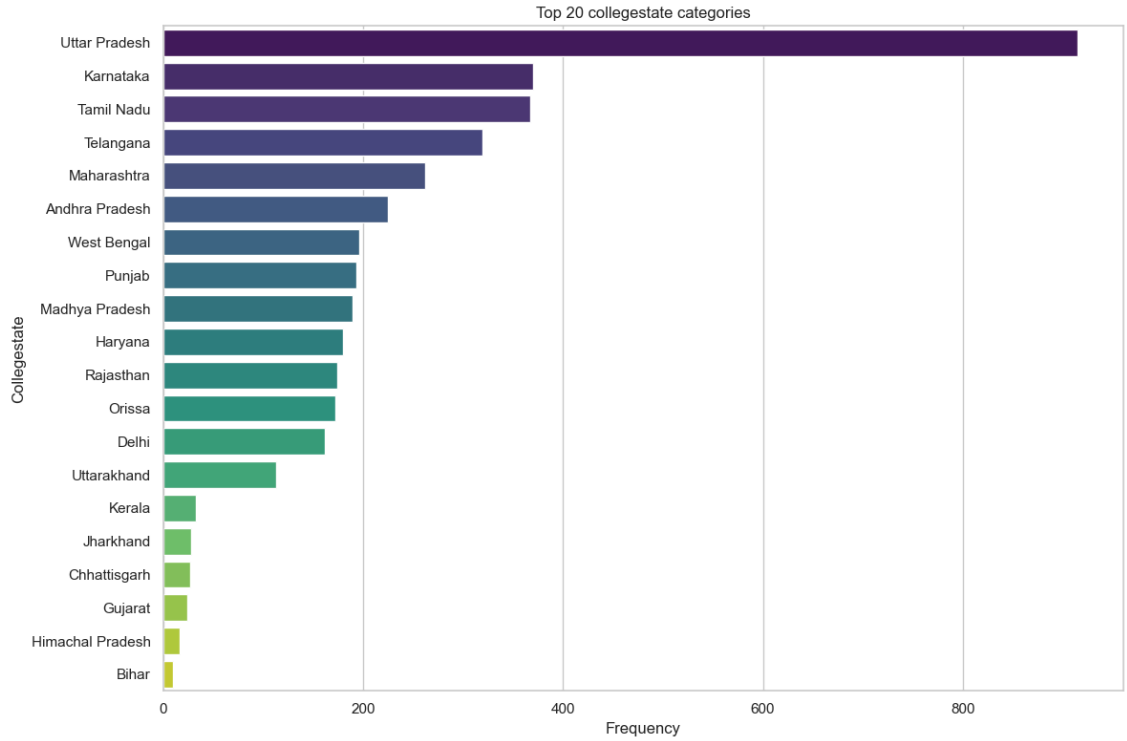
Out[62]:
```
<Axes: xlabel='gender', ylabel='count'>
```



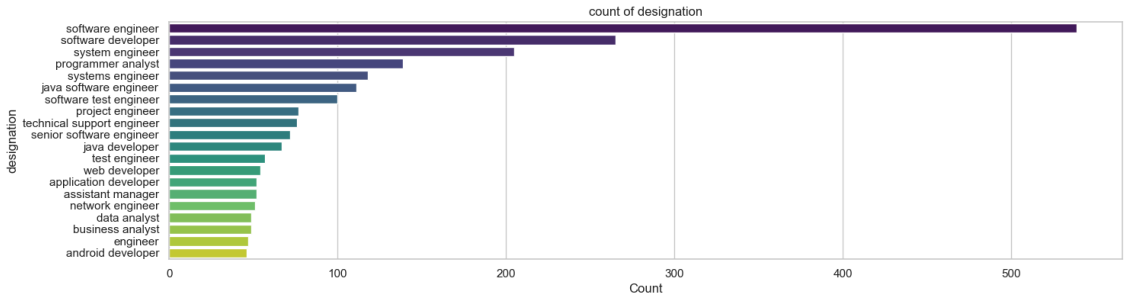
In [63]:
```
df.columns
```

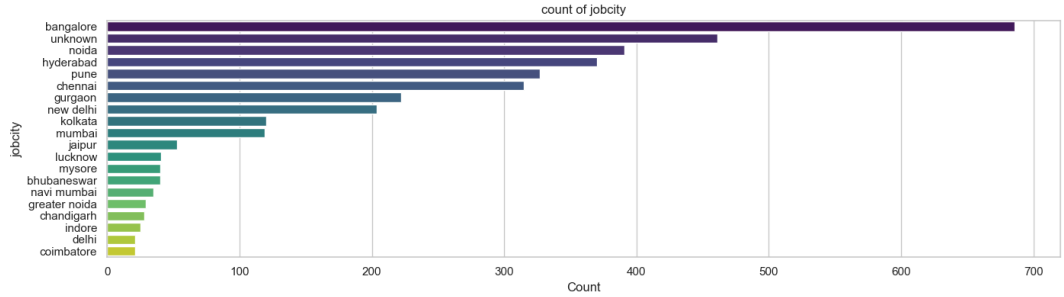
Out[63]:
```
Index(['id', 'salary', 'doj', 'dol', 'designation', 'jobcity', 'gender',
'dob',
       '10percentage', '10board', '12graduation', '12percentage', '12boar
d',
       'collegeid', 'collegetier', 'degree', 'specialization', 'collegegp
a',
       'collegecityid', 'collegecitytier', 'collegestate', 'graduationyea
r',
       'english', 'logical', 'quant', 'domain', 'computerprogramming',
       'electronicsandsemicon', 'computerscience', 'mechanicalengg',
       'electricalengg', 'telecomengg', 'civilengg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience'],
      dtype='object')
```

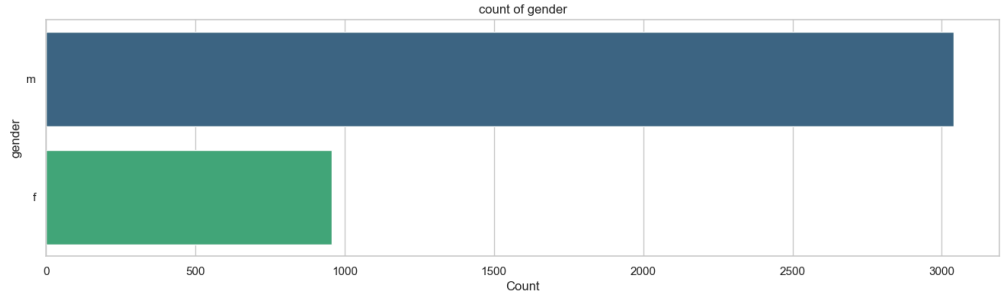
In [65]:
```
top_collegestates = df['collegestate'].value_counts().nlargest(20)
plt.figure(figsize=(12,8))
sns.countplot(y='collegestate',data=df[df['collegestate'].isin(top_college
palette='viridis',
order=top_collegestates.index)

plt.title('Top 20 collegestate categories')
plt.xlabel('Frequency')
plt.ylabel('Collegestate')
plt.tight_layout()
plt.show()
```

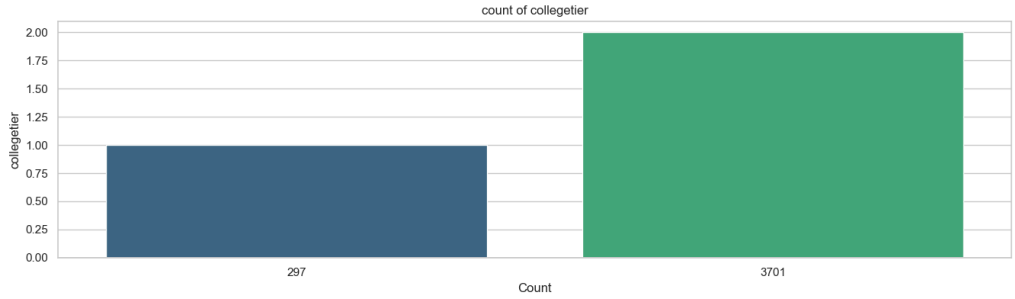


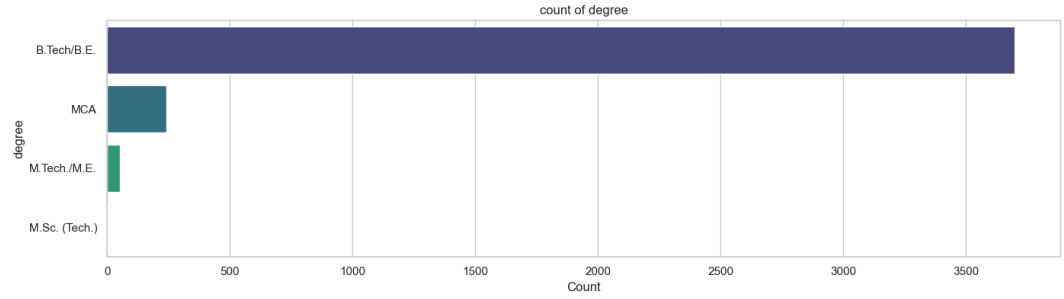
In [68]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# set the aesthetics for the plots
sns.set(style='whitegrid')

#list of important categorical columns
important_categorical_columns = ['designation','jobcity','gender','college

# create a bar plot for each important categorical column
plt.figure(figsize=(15,20))  #Adjust the figure size as needed

for i , column in enumerate(important_categorical_columns):
    plt.subplot(len(important_categorical_columns),1,i+1)  # create a subp
    top_values = df[column].value_counts().nlargest(20)  # get top 20 valu
    sns.barplot(x=top_values.values, y=top_values.index, palette='viridis'
    plt.title(f'count of {column}')  # set the title
    plt.xlabel('Count')  # label for x-axis
    plt.ylabel(column)  # label for y-axis

plt.tight_layout()  # adjust layout to prevent clipping of tick-labels
plt.show()
```





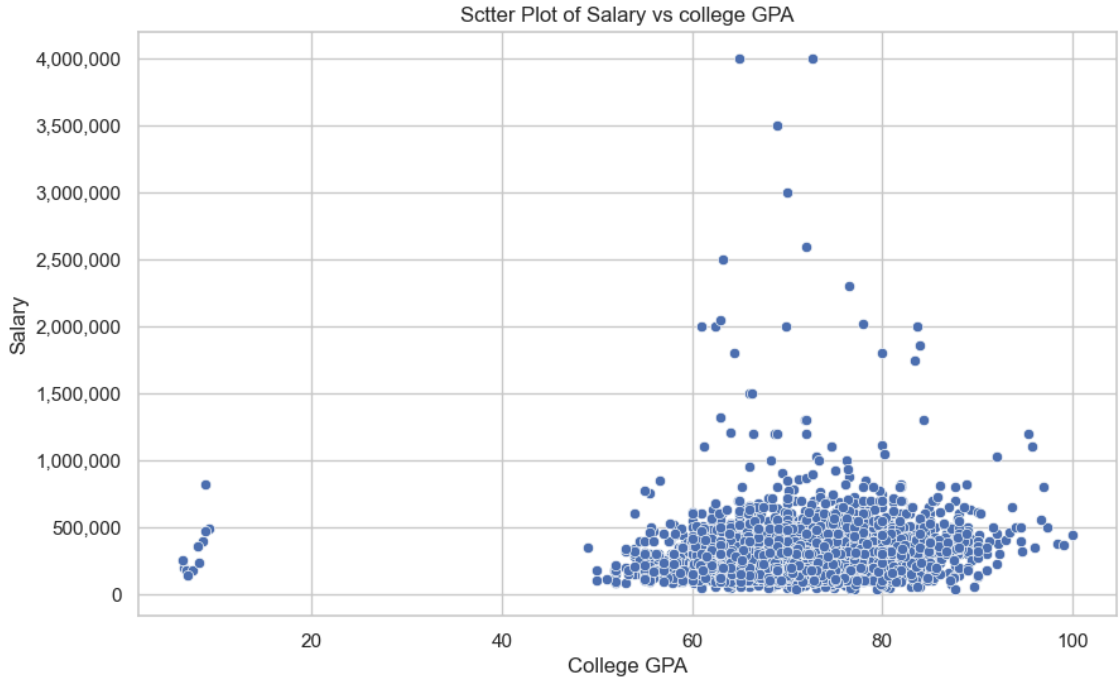




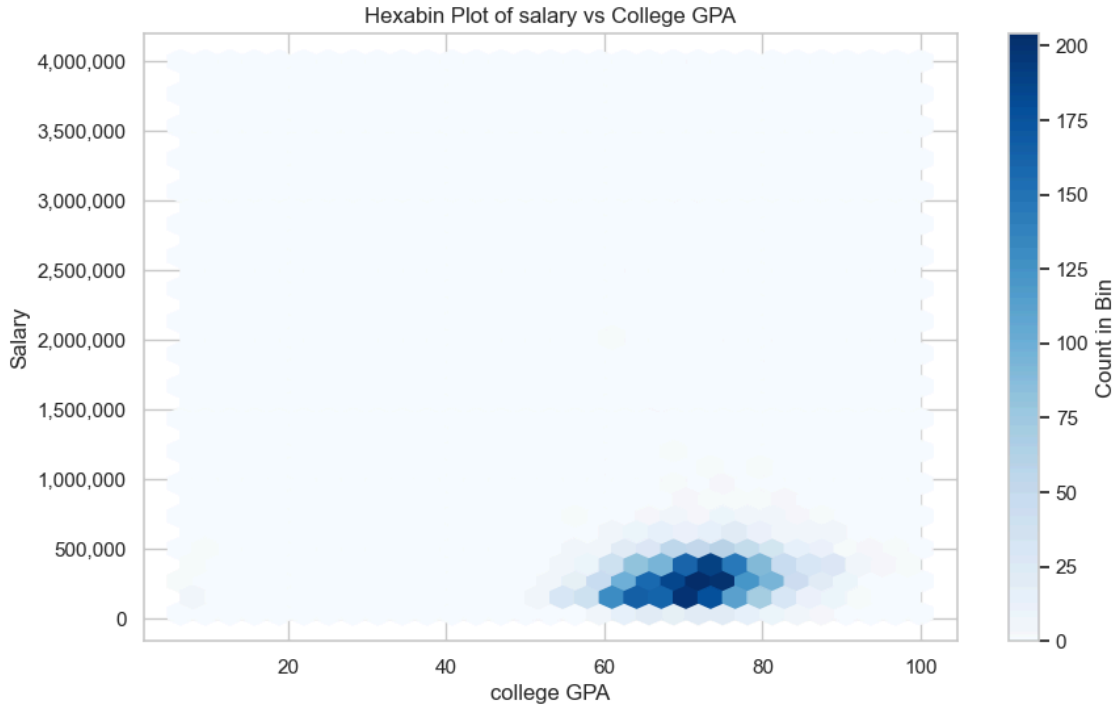


2. **Bivariate Analysis**

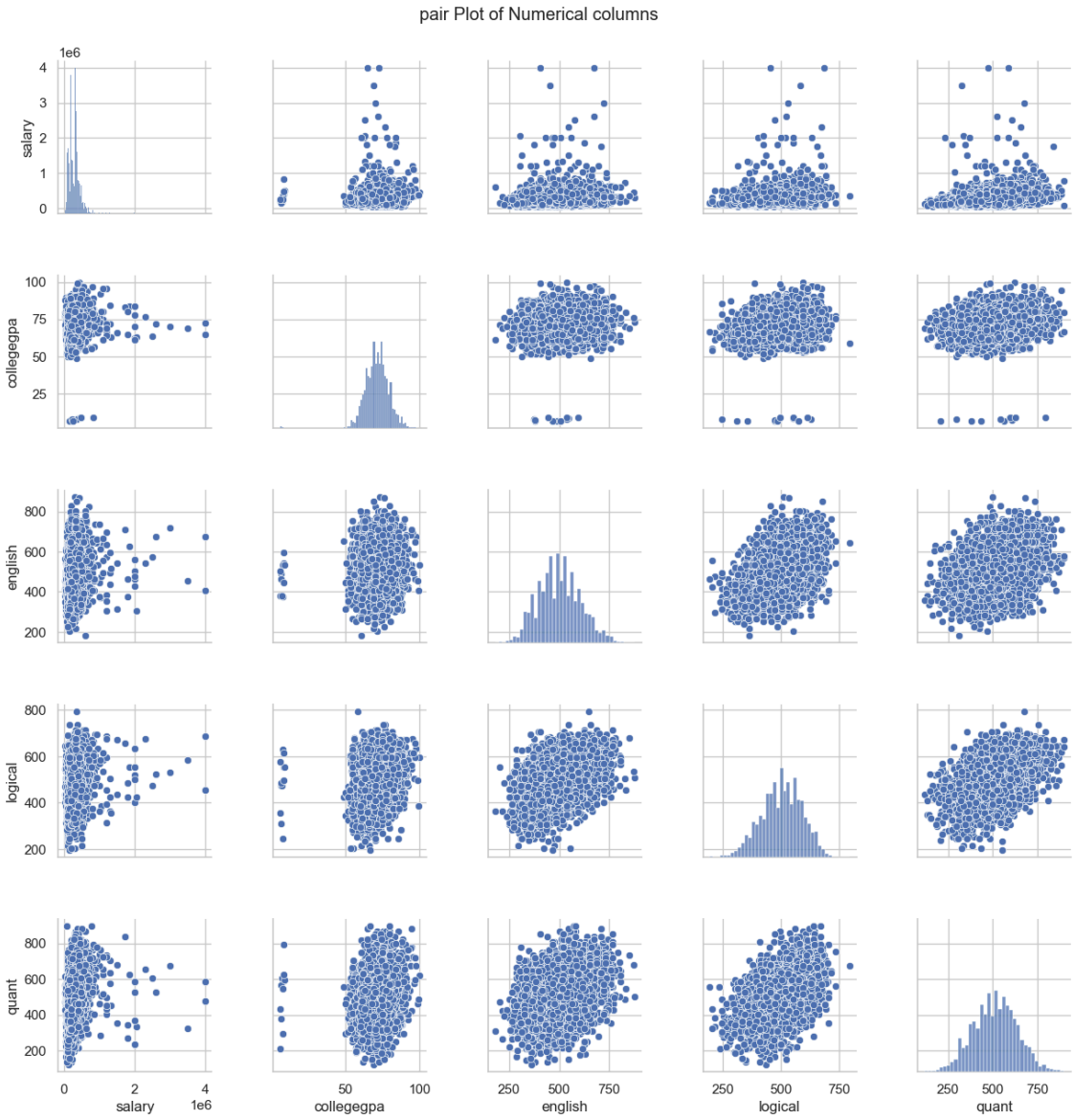
In [69]:
```
from matplotlib.ticker import FuncFormatter

# function to format y-axis labels
def currency(x,_):
    return f'{int(x):,}'  # Format as integer with commas

plt.figure(figsize=(10,6))
sns.scatterplot(data=df,x='collegegpa',y='salary')
plt.title('Sctter Plot of Salary vs college GPA')
plt.xlabel('College GPA')
plt.ylabel('Salary')
plt.grid('True')

# apply the formatter to the y-axis
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```



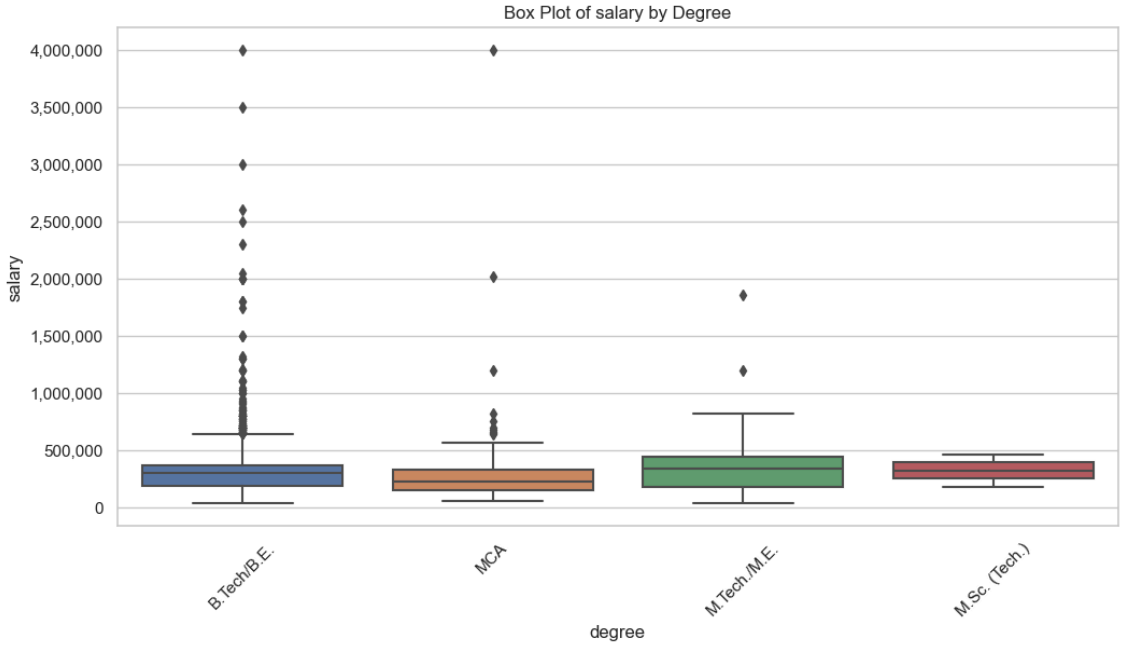
In [71]:
```
plt.figure(figsize=(10,6))
plt.hexbin(df['collegegpa'],df['salary'],gridsize=30,cmap='Blues')
plt.colorbar(label='Count in Bin')
plt.title('Hexabin Plot of salary vs College GPA')
plt.xlabel('college GPA')
plt.ylabel('Salary')
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```



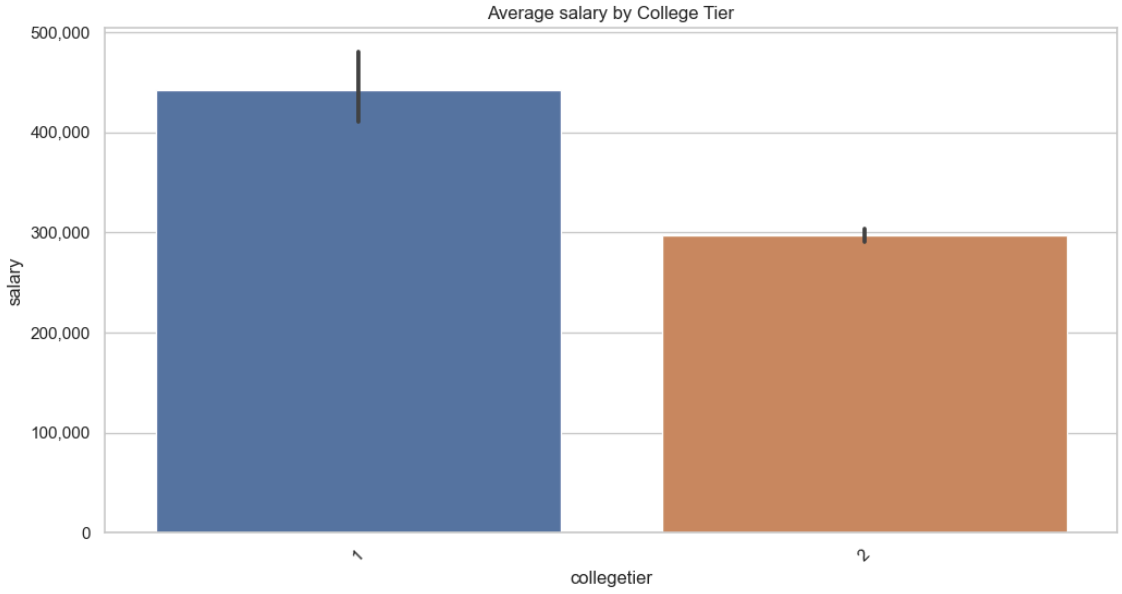
In [72]:
```
numerical_columns = ['salary', 'collegegpa', 'english', 'logical', 'quant'
sns.set(style='whitegrid')
pair_plot = sns.pairplot(df[numerical_columns])
plt.suptitle('pair Plot of Numerical columns', y= 1.02)
plt.subplots_adjust(hspace=0.4,wspace=0.4)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))

plt.show()
```



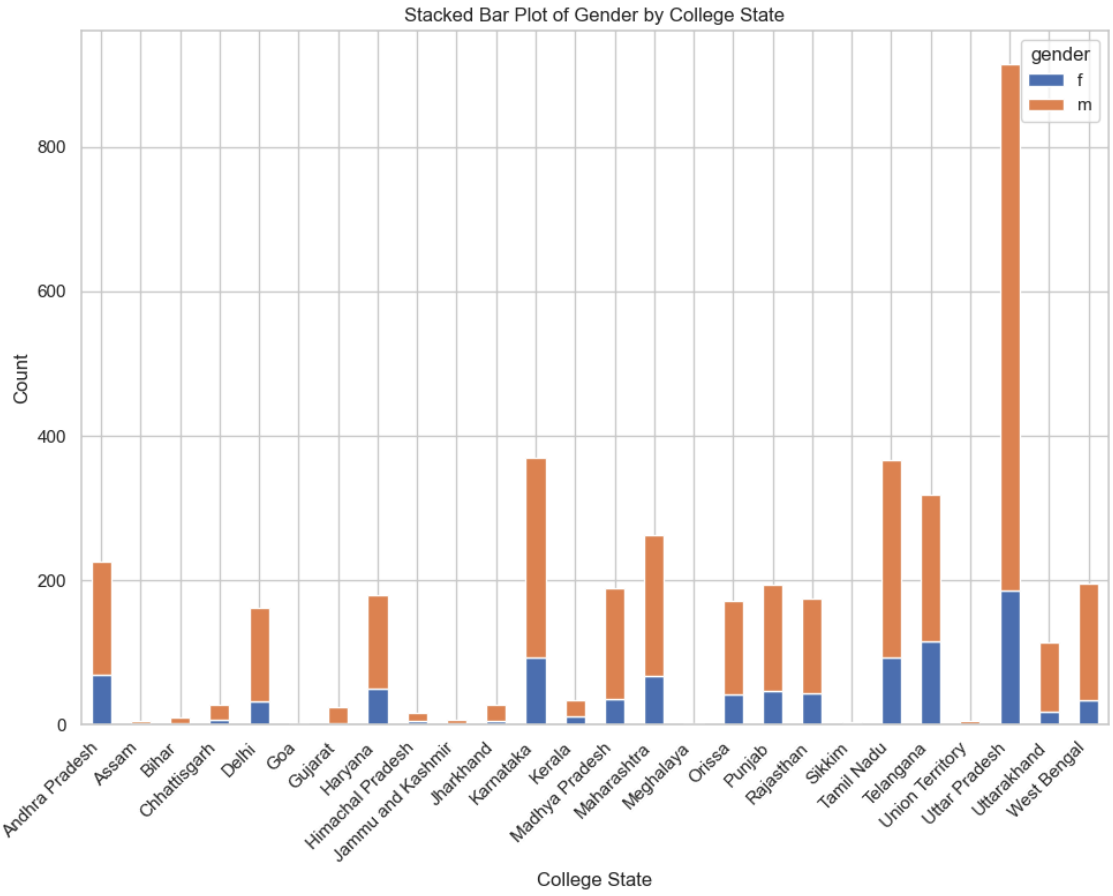
In [74]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='degree',y='salary')
plt.title('Box Plot of salary by Degree')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```



In [75]:
```
plt.figure(figsize=(12,6))
sns.barplot(data=df, x='collegetier',y='salary',estimator=np.mean)
plt.title('Average salary by College Tier')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```



In [76]:
```
pivot_table = df.pivot_table(index='collegestate', columns='gender',
values='salary', aggfunc='count').fillna(0)
pivot_table.plot(kind='bar', stacked=True, figsize=(10, 8))
plt.title('Stacked Bar Plot of Gender by College State')
plt.xlabel('College State')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right') # Adjusted alignment to 'right'
plt.tight_layout() # Adjust layout to prevent clipping
plt.show()
```



3 Step- 5- Research Questions

In [77]:
```
df.columns
```

Out[77]:
```
Index(['id', 'salary', 'doj', 'dol', 'designation', 'jobcity', 'gender',
'dob',
       '10percentage', '10board', '12graduation', '12percentage', '12boar
d',
       'collegeid', 'collegetier', 'degree', 'specialization', 'collegegp
a',
       'collegecityid', 'collegecitytier', 'collegestate', 'graduationyea
r',
       'english', 'logical', 'quant', 'domain', 'computerprogramming',
       'electronicsandsemicon', 'computerscience', 'mechanicalengg',
       'electricalengg', 'telecomengg', 'civilengg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience'],
      dtype='object')
```

In [80]:
```
from scipy import stats

 # Specify the claimed salary range
lower_bound = 2.5 * 100000 # converting lakhs to actual number
upper_bound = 3 * 100000

 # Filter data for specified job titles
job_titles = ['Programming Analyst', 'Software Engineer', 'Hardware Engine
filtered_data = df[df['designation'].isin(job_titles)]

 # Perform one-sample t-test on salary
if not filtered_data.empty:
    t_statistic, p_value = stats.ttest_1samp(filtered_data['salary'],lower

 # Display the results
    print(f"T-statistic: {t_statistic}, P-value: {p_value}")
 # Interpret the p-value
    alpha = 0.05
    if p_value < alpha:
        print("Reject the null hypothesis: Average salary significantly di
    else:
        print("Fail to reject the null hypothesis: Average salary does not
else:
    print("No data found for the specified job titles.")
```

```
No data found for the specified job titles.
```

In [83]:
```
# Assuming df is your DataFrame containing the data
job_titles = ['Programming Analyst', 'Software Engineer', 'Hardware Engine
salary_data = df[df['designation'].isin(job_titles)]

 # Calculate the average salary for each job title
average_salaries = salary_data.groupby('designation')['salary'].mean().res

# Check if average salaries are within the claimed range of 2.5 to 3 Lakhs
average_salaries['within_claimed_range'] = average_salaries['salary'].appl
print("Average Salaries for Specified Job Titles:")
print(average_salaries)
print("\nAverage Salaries within Claimed Range:")
print(average_salaries[average_salaries['within_claimed_range']])
```

```
Average Salaries for Specified Job Titles:
Empty DataFrame
Columns: [designation, salary, within_claimed_range]
Index: []

Average Salaries within Claimed Range:
Empty DataFrame
Columns: []
Index: []
```

In [84]:
```
 # Create a contingency table
contingency_table = pd.crosstab(df['gender'], df['specialization'])
 # Display the contingency table
print("Contingency Table:")
print(contingency_table)
 # Perform Chi-Square test
chi2_stat, p_value, dof, expected = stats.chi2_contingency(contingency_tab
 # Create a results DataFrame with reset index
results = pd.DataFrame({
 'Metric': ['Chi-Squared Statistic', 'P-value', 'Degrees of Freedom','Conc
 'Value': [
 chi2_stat,
 p_value,
 dof,
 "Reject the null hypothesis" if p_value < 0.05 else "Fail to reject the nu
 ]
 })
 # Reset the index of the results DataFrame
results.reset_index(drop=True, inplace=True)

# Display the results
print("\nChi-Square Test Results:")
print(results)
```

```
Contingency Table:
specialization  aeronautical engineering  \
gender
f                                       1
m                                       2

specialization  applied electronics and instrumentation  \
gender
f                                                     2
m                                                     7

specialization  automobile/automotive engineering  biomedical engineering
\
gender
f                                                 0                      2
m                                                 5                      0

specialization  biotechnology  ceramic engineering  chemical engineering
\
gender
f                           9                   0                     1
m                           6                   1                     8

specialization  civil engineering  computer and communication engineering
\
gender
f                              6                                       0
m                             23                                       1

specialization  computer application  ...  internal combustion engine  \
gender                                 ...
f                                59  ...                           0
m                               185  ...                           1

specialization  mechanical & production engineering  \
gender
f                                                  0
m                                                  1

specialization  mechanical and automation  mechanical engineering  \
gender
f                                       0                     10
m                                       5                    191

specialization  mechatronics  metallurgical engineering  other  \
gender
f                           1                          0      0
m                           3                          2     13

specialization  polymer technology  power systems and automation  \
gender
f                                0                             0
m                                1                             1

specialization  telecommunication engineering
gender
f                                          1
m                                          5

[2 rows x 46 columns]
```

```
Chi-Square Test Results:
                  Metric                        Value
0  Chi-Squared Statistic                          104
1                P-value                            0
2     Degrees of Freedom                           45
3             Conclusion  Reject the null hypothesis
```

In [ ]: