Introduction

Mental health is the ability to deal with ups and downs in our life. Your mental health always fluctuates as circumstances change as you move through different stages in your life. OSMI, an organization is working to help people to identify and overcome mental health disorders while working in a tech space. They perform surveys to examine the frequency of mental health disorders among tech workers.

Problem Statement

A survey is done to find the attitudes towards mental health in the tech workplace.

The description of the features collected during the survey is given below:

Id	Features	Description
01	Timestamp	Time the survey was submitted.
02	Age	The age of the person.
03	Gender	The gender of the person.
04	Country	The country name where person belongs to.
05	state	The state name where person belongs to.
06	self_employed	Is the person self employed or not.
07	family_history	Does the person's family history had mental illness or not?
08	treatment	Have you sought treatment for a mental health condition?
09	work_intefere	If you have a mental health condition, do you feel that it interferes with your work?
10	no_employees	How many employees does your company or organization have?
11	remote_work	Do you work remotely (outside of an office) at least 50% of the time?
12	tech_company	Is your employer primarily a tech company/organization?
13	benifits	Does your employer provide mental health benefits?
14	care_options	Do you know the options for mental health care your employer provides?
15	wellness_program	Has your employer ever discussed mental health as part of an employee wellness program?
16	seek_help	Does your employer provide resources to learn more about mental health issues and how to seek help?
17	anonymity	Is your anonymity protected if you choose to take advantage of mental health or substance abuse treatment resources?
18	leave	How easy is it for you to take medical leave for a mental health condition?
19	mental_health_consequence	Do you think that discussing a mental health issue with your employer would have negative consequences?

Id	Features	Description
20	phy_health_consequence	Do you think that discussing a physical health issue with your employer would have negative consequences?
21	coworkers	Would you be willing to discuss a mental health issue with your coworkers?
22	supervisor	Would you be willing to discuss a mental health issue with your direct supervisor(s)?
23	mental_health_interview	Would you bring up a mental health issue with a potential employer in an interview?
24	phs_health_interivew	Would you bring up a physical health issue with a potential employer in an interview?
25	mental_vs_physical	Do you feel that your employer takes mental health as seriously as physical health?
26	obs_consequence	Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?
27	comments	Any additional notes or comments.

Analysis is to be done to find the set of parameters affecting the mental health of the people.

The process consists of the following steps - Data collection, Data Pre-profiling, Data cleaning, Exploratory Data Analysis, and Summarization

Installing Libraries

In [1]:
```
# Package that is required by pandas profiling
!pip install -q datascience

!pip install -q --upgrade pandas-profiling
```

Importing Libraries

In [2]:
```
import numpy as np                      # Importing numpy library

import pandas as pd                     # Importing for panel data analysis

import matplotlib.pyplot as plt         # Importing pyplot interface using matplotlib
%matplotlib inline

import seaborn as sns                   # Importing seaborm library for interactive v

import warnings                         # Importing warnings to disable runtime warni
warnings.filterwarnings("ignore")
```

Data Collection

In [3]:
```
# download the dataset to pandas dataframe
data = pd.read_csv('survey.csv')
```

In [4]:
```
# setting to display all the rows and columns
pd.set_option("display.max_columns", None)
```

```python
pd.set_option("display.max_colwidth", None)
```

In [5]:
```python
#display the first 5 rows
data.head()
```

Out[5]:

	Timestamp	Age	Gender	Country	state	self_employed	family_history	treatment	work_inte
0	2014-08-27 11:29:31	37	Female	United States	IL	NaN	No	Yes	(
1	2014-08-27 11:29:37	44	M	United States	IN	NaN	No	No	R
2	2014-08-27 11:29:44	32	Male	Canada	NaN	NaN	No	No	R
3	2014-08-27 11:29:46	31	Male	United Kingdom	NaN	NaN	Yes	Yes	(
4	2014-08-27 11:30:22	31	Male	United States	TX	NaN	No	No	N

In [6]:
```python
# shape or dimension of the dataframe - number of rows and columns
print('Data Shape:', data.shape)
```

```
Data Shape: (1259, 27)
```

Observation:

There are 1259 rows and 27 columns

In [7]:
```python
# information about the dataframe
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Timestamp                  1259 non-null   object
 1   Age                        1259 non-null   int64 
 2   Gender                     1259 non-null   object
 3   Country                    1259 non-null   object
 4   state                      744 non-null    object
 5   self_employed              1241 non-null   object
 6   family_history             1259 non-null   object
 7   treatment                  1259 non-null   object
 8   work_interfere             995 non-null    object
 9   no_employees               1259 non-null   object
 10  remote_work                1259 non-null   object
 11  tech_company               1259 non-null   object
 12  benefits                   1259 non-null   object
 13  care_options               1259 non-null   object
 14  wellness_program           1259 non-null   object
 15  seek_help                  1259 non-null   object
 16  anonymity                  1259 non-null   object
 17  leave                      1259 non-null   object
 18  mental_health_consequence  1259 non-null   object
 19  phys_health_consequence    1259 non-null   object
 20  coworkers                  1259 non-null   object
 21  supervisor                 1259 non-null   object
 22  mental_health_interview    1259 non-null   object
 23  phys_health_interview      1259 non-null   object
 24  mental_vs_physical         1259 non-null   object
 25  obs_consequence            1259 non-null   object
 26  comments                   164 non-null    object
dtypes: int64(1), object(26)
memory usage: 265.7+ KB
```

Observation:

The data type of Timestamp column is object. It is to be rectified.

In [8]:
```
# description of the numerical data in the dataframe
data.describe()
```

Out[8]:

	Age
count	1.259000e+03
mean	7.942815e+07
std	2.818299e+09
min	-1.726000e+03
25%	2.700000e+01
50%	3.100000e+01
75%	3.600000e+01
max	1.000000e+11

Obervation:

There are some absurd values in Age column. The average age of the person is found to be 7.942815e+07 years and it is absurd. The minimum and maximum ages are found to be negative and very large numbers.

Data Pre-Profiling

In [9]:
```
# Pandas Profiling
from ydata_profiling import ProfileReport
```

In [10]:
```
#Generates profile reports from a pandas DataFrame
profile = ProfileReport(df=data, minimal=True)

#statistics are presented in an interactive HTML report
profile.to_file(output_file="Pre Profiling Report.html")
print("Report exported successfully!")
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
Report exported successfully!
```

Data Cleaning

Handling Missing Data

In [11]:
```
# identifying the count of missing values in each column in ascending order
data.isna().sum().sort_values(ascending=False)
```

Out[11]:
```
comments                     1095
state                         515
work_interfere                264
self_employed                  18
seek_help                       0
obs_consequence                 0
mental_vs_physical              0
phys_health_interview           0
mental_health_interview         0
supervisor                      0
coworkers                       0
phys_health_consequence         0
mental_health_consequence       0
leave                           0
anonymity                       0
Timestamp                       0
wellness_program                0
Age                             0
benefits                        0
tech_company                    0
remote_work                     0
no_employees                    0
treatment                       0
family_history                  0
Country                         0
Gender                          0
care_options                    0
dtype: int64
```

Obervation:

There are four columns which consists of null values

In [12]:
```
#create a new frame to get the details of null values in each column
null_frame = pd.DataFrame(index = data.columns)

null_frame['Frequency'] = data.isna().sum()

null_frame['Missing Percentage'] = (data.isna().sum()/data.shape[0])*100
null_frame
```

Out[12]:

	Frequency	Missing Percentage
Timestamp	0	0.000000
Age	0	0.000000
Gender	0	0.000000
Country	0	0.000000
state	515	40.905481
self_employed	18	1.429706
family_history	0	0.000000
treatment	0	0.000000
work_interfere	264	20.969023
no_employees	0	0.000000
remote_work	0	0.000000
tech_company	0	0.000000
benefits	0	0.000000
care_options	0	0.000000
wellness_program	0	0.000000
seek_help	0	0.000000
anonymity	0	0.000000
leave	0	0.000000
mental_health_consequence	0	0.000000
phys_health_consequence	0	0.000000
coworkers	0	0.000000
supervisor	0	0.000000
mental_health_interview	0	0.000000
phys_health_interview	0	0.000000
mental_vs_physical	0	0.000000
obs_consequence	0	0.000000
comments	1095	86.973789

Observations:

Feature	Missing %	Solution
state	40.9%	Replace with mode

Feature	Missing %	Solution
self_employed	1.43%	Replace with mode
work_interfere	20.97%	Replace with mode
comments	86.97%	Drop the feature as the missing percentage is too high

Handling state column

In [13]:
```
#check the mode values in state column
data.state.mode()
```

Out[13]:
```
0    CA
Name: state, dtype: object
```

In [14]:
```
#fill the missing values in state column with mode
data.state.fillna(data.state.mode()[0], inplace = True)
```

Handling self_employed column

In [15]:
```
#check the mode values in self_employed column
data.self_employed.mode()
```

Out[15]:
```
0    No
Name: self_employed, dtype: object
```

In [16]:
```
#fill the missing values in self_employed column with mode
data.self_employed.fillna(data.self_employed.mode()[0], inplace=True)
```

Handling work_interfere column

In [17]:
```
#check the mode values in work_interfere column
data.work_interfere.mode()
```

Out[17]:
```
0    Sometimes
Name: work_interfere, dtype: object
```

In [18]:
```
#fill the missing values in work_interfere column with mode
data.work_interfere.fillna(data.work_interfere.mode()[0], inplace=True)
```

Handling comments column

In [19]:
```
#drop comments column
data.drop(['comments'], axis=1, inplace=True)
```

verification after handling missing data

In [20]:
```
# identifying the count of missing data in each column in ascending order
data.isna().sum().sort_values(ascending=False)
```

Out[20]:
```
Timestamp                    0
Age                          0
mental_vs_physical           0
phys_health_interview        0
mental_health_interview      0
supervisor                   0
coworkers                    0
phys_health_consequence      0
mental_health_consequence    0
leave                        0
anonymity                    0
seek_help                    0
wellness_program             0
care_options                 0
benefits                     0
tech_company                 0
remote_work                  0
no_employees                 0
work_interfere               0
treatment                    0
family_history               0
self_employed                0
state                        0
Country                      0
Gender                       0
obs_consequence              0
dtype: int64
```

Handling Redundant Data

In [21]:
```
# finding duplicated rows in the dataframe
data.duplicated().any()
```

Out[21]:
```
False
```

In [22]:
```
data.duplicated().sum()
```

Out[22]:
```
0
```

Observation:

There are no duplicated rows.

Handling Inconsistent Data

It was noticed that Timestamp feature was identified as Object

In [23]:
```
#changing the data type of Timestamp column
data['Timestamp'] = pd.to_datetime(data.Timestamp)
```

verification after Handling of Timestamp column

In [24]:
```
# information about the dataframe
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 26 columns):
 #   Column                     Non-Null Count  Dtype         
---  ------                     --------------  -----         
 0   Timestamp                  1259 non-null   datetime64[ns]
 1   Age                        1259 non-null   int64         
 2   Gender                     1259 non-null   object        
 3   Country                    1259 non-null   object        
 4   state                      1259 non-null   object        
 5   self_employed              1259 non-null   object        
 6   family_history             1259 non-null   object        
 7   treatment                  1259 non-null   object        
 8   work_interfere             1259 non-null   object        
 9   no_employees               1259 non-null   object        
 10  remote_work                1259 non-null   object        
 11  tech_company               1259 non-null   object        
 12  benefits                   1259 non-null   object        
 13  care_options               1259 non-null   object        
 14  wellness_program           1259 non-null   object        
 15  seek_help                  1259 non-null   object        
 16  anonymity                  1259 non-null   object        
 17  leave                      1259 non-null   object        
 18  mental_health_consequence  1259 non-null   object        
 19  phys_health_consequence    1259 non-null   object        
 20  coworkers                  1259 non-null   object        
 21  supervisor                 1259 non-null   object        
 22  mental_health_interview    1259 non-null   object        
 23  phys_health_interview      1259 non-null   object        
 24  mental_vs_physical         1259 non-null   object        
 25  obs_consequence            1259 non-null   object        
dtypes: datetime64[ns](1), int64(1), object(24)
memory usage: 255.9+ KB
```

checking for any inconsistencies in categorical columns

In [25]:

```
# display all the column names
cols = data.select_dtypes(include=['object']).columns.to_list()
print("Columns are:", cols,"\n")

for c in cols:
    print("Column:",c)     #display column name
    print("No.of unique values:",len(data[c].unique()))     #display number of uniq
    print(data[c].unique(),"\n\n")    #display unique values
```

```
Columns are: ['Gender', 'Country', 'state', 'self_employed', 'family_history', 'tr
eatment', 'work_interfere', 'no_employees', 'remote_work', 'tech_company', 'benefi
ts', 'care_options', 'wellness_program', 'seek_help', 'anonymity', 'leave', 'menta
l_health_consequence', 'phys_health_consequence', 'coworkers', 'supervisor', 'ment
al_health_interview', 'phys_health_interview', 'mental_vs_physical', 'obs_conseque
nce']

Column: Gender
No.of unique values: 49
['Female' 'M' 'Male' 'male' 'female' 'm' 'Male-ish' 'maile' 'Trans-female'
 'Cis Female' 'F' 'something kinda male?' 'Cis Male' 'Woman' 'f' 'Mal'
 'Male (CIS)' 'queer/she/they' 'non-binary' 'Femake' 'woman' 'Make' 'Nah'
 'All' 'Enby' 'fluid' 'Genderqueer' 'Female ' 'Androgyne' 'Agender'
 'cis-female/femme' 'Guy (-ish) ^_^' 'male leaning androgynous' 'Male '
 'Man' 'Trans woman' 'msle' 'Neuter' 'Female (trans)' 'queer'
 'Female (cis)' 'Mail' 'cis male' 'A little about you' 'Malr' 'p' 'femail'
 'Cis Man' 'ostensibly male, unsure what that really means']


Column: Country
No.of unique values: 48
['United States' 'Canada' 'United Kingdom' 'Bulgaria' 'France' 'Portugal'
 'Netherlands' 'Switzerland' 'Poland' 'Australia' 'Germany' 'Russia'
 'Mexico' 'Brazil' 'Slovenia' 'Costa Rica' 'Austria' 'Ireland' 'India'
 'South Africa' 'Italy' 'Sweden' 'Colombia' 'Latvia' 'Romania' 'Belgium'
 'New Zealand' 'Zimbabwe' 'Spain' 'Finland' 'Uruguay' 'Israel'
 'Bosnia and Herzegovina' 'Hungary' 'Singapore' 'Japan' 'Nigeria'
 'Croatia' 'Norway' 'Thailand' 'Denmark' 'Bahamas, The' 'Greece' 'Moldova'
 'Georgia' 'China' 'Czech Republic' 'Philippines']


Column: state
No.of unique values: 45
['IL' 'IN' 'CA' 'TX' 'TN' 'MI' 'OH' 'CT' 'MD' 'NY' 'NC' 'MA' 'IA' 'PA'
 'WA' 'WI' 'UT' 'NM' 'OR' 'FL' 'MN' 'MO' 'AZ' 'CO' 'GA' 'DC' 'NE' 'WV'
 'OK' 'KS' 'VA' 'NH' 'KY' 'AL' 'NV' 'NJ' 'SC' 'VT' 'SD' 'ID' 'MS' 'RI'
 'WY' 'LA' 'ME']


Column: self_employed
No.of unique values: 2
['No' 'Yes']


Column: family_history
No.of unique values: 2
['No' 'Yes']


Column: treatment
No.of unique values: 2
['Yes' 'No']


Column: work_interfere
No.of unique values: 4
['Often' 'Rarely' 'Never' 'Sometimes']


Column: no_employees
No.of unique values: 6
['6-25' 'More than 1000' '26-100' '100-500' '1-5' '500-1000']
```

```
Column: remote_work
No.of unique values: 2
['No' 'Yes']


Column: tech_company
No.of unique values: 2
['Yes' 'No']


Column: benefits
No.of unique values: 3
['Yes' "Don't know" 'No']


Column: care_options
No.of unique values: 3
['Not sure' 'No' 'Yes']


Column: wellness_program
No.of unique values: 3
['No' "Don't know" 'Yes']


Column: seek_help
No.of unique values: 3
['Yes' "Don't know" 'No']


Column: anonymity
No.of unique values: 3
['Yes' "Don't know" 'No']


Column: leave
No.of unique values: 5
['Somewhat easy' "Don't know" 'Somewhat difficult' 'Very difficult'
 'Very easy']


Column: mental_health_consequence
No.of unique values: 3
['No' 'Maybe' 'Yes']


Column: phys_health_consequence
No.of unique values: 3
['No' 'Yes' 'Maybe']


Column: coworkers
No.of unique values: 3
['Some of them' 'No' 'Yes']


Column: supervisor
No.of unique values: 3
['Yes' 'No' 'Some of them']


Column: mental_health_interview
No.of unique values: 3
['No' 'Yes' 'Maybe']
```

```
Column: phys_health_interview
No.of unique values: 3
['Maybe' 'No' 'Yes']


Column: mental_vs_physical
No.of unique values: 3
['Yes' "Don't know" 'No']


Column: obs_consequence
No.of unique values: 2
['No' 'Yes']
```

Observation:

Gender column is having data in different formats. There are 49 unique values in Gender column

In [26]:
```
# to display values in Gender column and it's count
data.Gender.value_counts()
```

Out[26]:
```
Male                                               615
male                                               206
Female                                             121
M                                                  116
female                                              62
F                                                   38
m                                                   34
f                                                   15
Make                                                 4
Male                                                 3
Woman                                                3
Cis Male                                             2
Man                                                  2
Female (trans)                                       2
Female                                               2
Trans woman                                          1
msle                                                 1
male leaning androgynous                             1
Neuter                                               1
cis male                                             1
queer                                                1
Female (cis)                                         1
Mail                                                 1
cis-female/femme                                     1
A little about you                                   1
Malr                                                 1
p                                                    1
femail                                               1
Cis Man                                              1
Guy (-ish) ^_^                                       1
Enby                                                 1
Agender                                              1
Androgyne                                            1
Male-ish                                             1
maile                                                1
Trans-female                                         1
Cis Female                                           1
something kinda male?                                1
Mal                                                  1
Male (CIS)                                           1
queer/she/they                                       1
non-binary                                           1
Femake                                               1
woman                                                1
Nah                                                  1
All                                                  1
fluid                                                1
Genderqueer                                          1
ostensibly male, unsure what that really means       1
Name: Gender, dtype: int64
```

In [27]:
```
# Remove the Undecisive
list = ['A little about you', 'p']
data = data[~data['Gender'].isin(list)]
```

verification

In [28]:
```
len(data.Gender.value_counts())
```

Out[28]:
```
47
```

In [29]:
```
# create separate lists for three categories
male_str = ["male", "m", "male-ish", "maile", "mal", "male (cis)", "make",
```

```
             "male ", "man", "msle", "mail", "malr", "cis man", "Cis Male",
             "cis male"]

female_str = ["cis female", "f", "female", "woman",  "femake", "female ",
              "cis-female/femme", "female (cis)", "femail"]

trans_str = ["trans-female", "something kinda male?", "queer/she/they",
             "non-binary", "nah", "all", "enby", "fluid", "genderqueer",
             "androgyne", "agender", "male leaning androgynous",
             "guy (-ish) ^_^",
             "trans woman", "neuter", "female (trans)", "queer",
             "ostensibly male, unsure what that really means"]
```

In [30]:

```
#storing the essential data in Gender column
def gender_discover(x):
    if x in male_str:
        return "male"
    elif x in female_str:
        return "female"
    else:
        return "trans"

data['Gender'] = data['Gender'].apply(gender_discover)
```

verification

In [31]:

```
len(data.Gender.value_counts())
```

Out[31]:

```
3
```

In [32]:

```
# to display values in Gender column and it's count
data.Gender.value_counts()
```

Out[32]:

```
trans     932
male      245
female     80
Name: Gender, dtype: int64
```

Handling outliers in Age column

All value above 60 will be capped to 60 (on average 60 is the retirement age for employment). All value below 18 will be capped to 18 (on average 18 is the minimum age for employment)

In [33]:

```
#removing the outliers
data["Age"] = np.where(data["Age"] <18, 18, data["Age"])
data["Age"] = np.where(data["Age"] >60, 60, data["Age"])
```

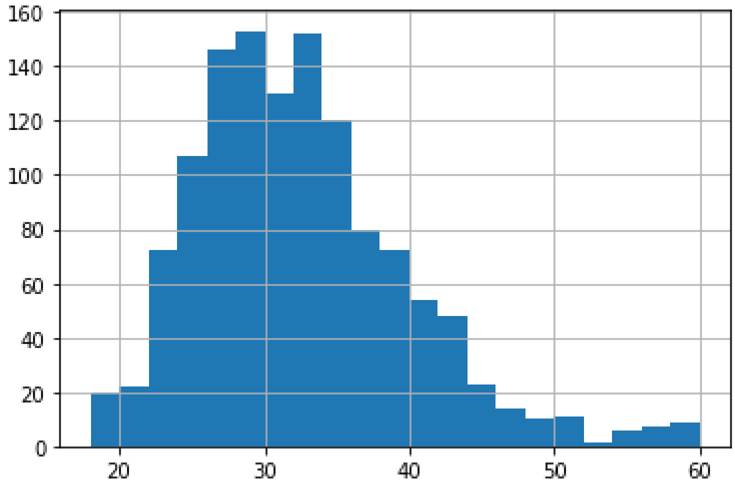
verification after handling outliers in Age column

In [34]:

```
# display distribution of data
data.Age.hist(bins = 21)
```

Out[34]:

```
<AxesSubplot:>
```



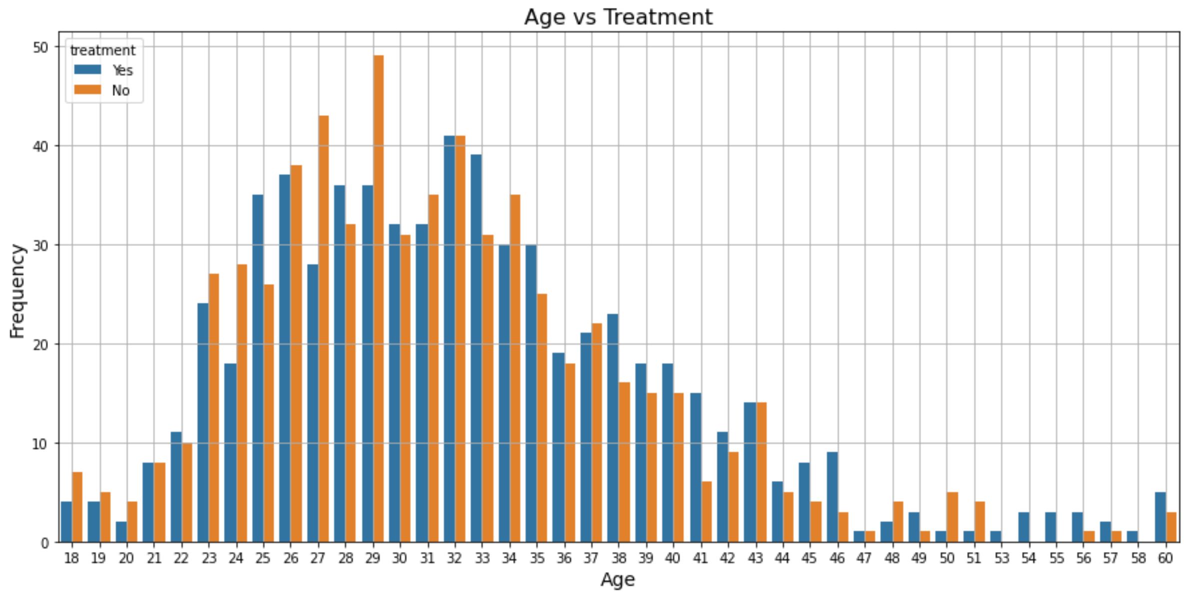
Observations:

Majority of people participated in the survey are from mid 20s to mid 30s

Exploratory Data Analysis

Question: How does age relate to their awareness toward mental health?

In [35]:
```
# plotting an empty figure of width 15 and height 7
fig = plt.figure(figsize=(15, 7))

# Plot countplot of age concerning treatment
sns.countplot(x='Age', hue='treatment', data=data)

# Add title name
plt.title(label='Age vs Treatment', size=16)

# Add x and y labels
plt.xlabel(xlabel='Age', size=14)
plt.ylabel(ylabel='Frequency', size=14)
plt.grid(b=True)

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



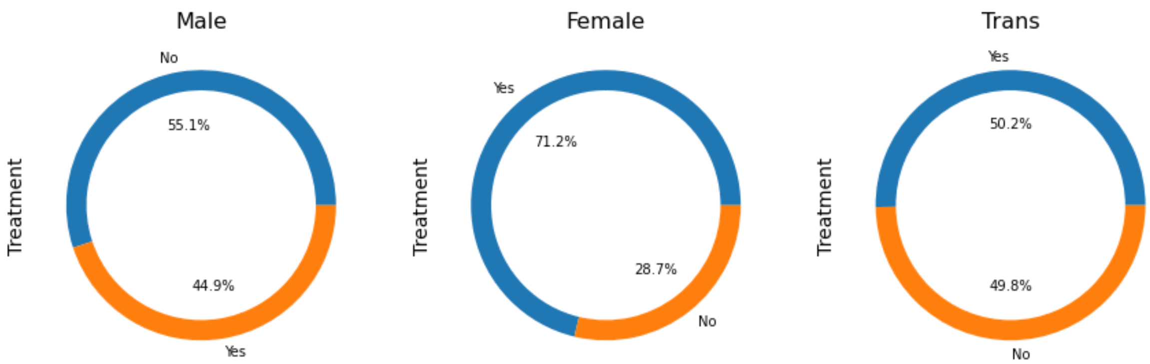
Observation:

People in the age group 18 - 26 are less conscious about treatment. People in the age group 27 - 29 are not seriously concerned about their mental health. People of age 33 and above are conscious and are up for treatment.

What is the association between treatment and gender in terms of ratio?

In [36]:

```
# Instantiate figure and axes object
fig, ax = plt.subplots(nrows=1, ncols=3, figsize=(15, 7))

# Initiate a list of gender values and index
gender = ['male','female','trans']
custom_index = [0, 1, 2]

# Plot donut chart for each gender concerning treatment
for i, j in zip(gender, custom_index):
  data['treatment'][data['Gender']==i].value_counts().plot.pie(
                                                  autopct='%1.1f%%',
                                                  wedgeprops=dict(width=0.15),
                                                  ax=ax[j])

  ax[j].set_title(label= i.capitalize(), size=16)
  ax[j].set_ylabel(ylabel='Treatment', size=14)


# Display the graph
plt.show()
```

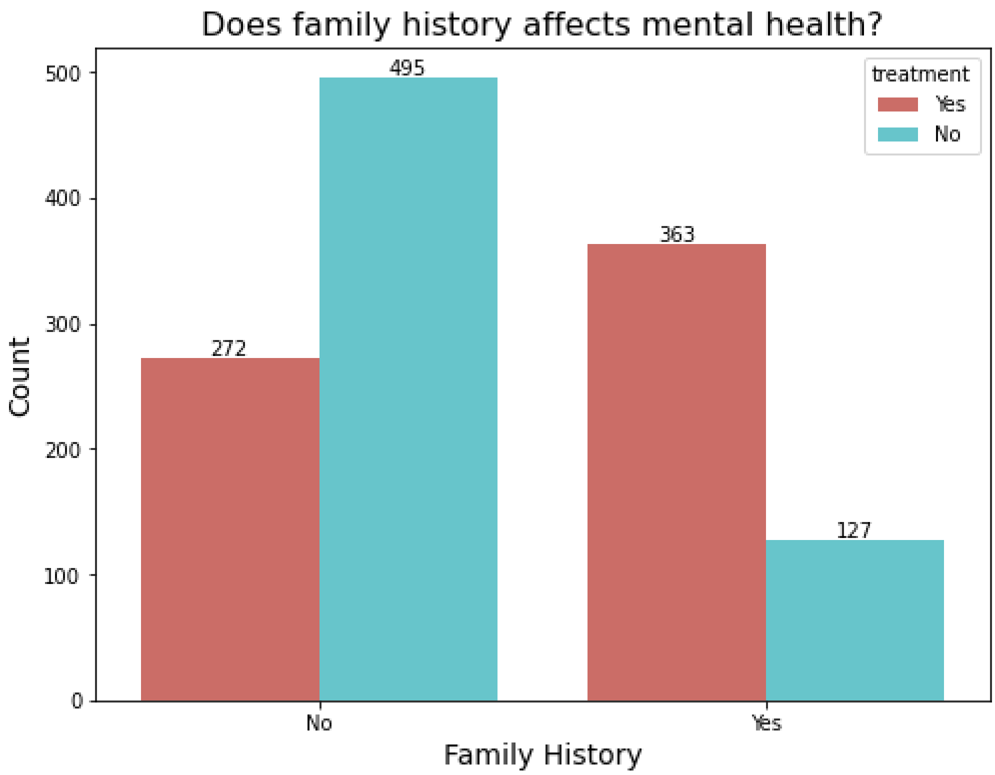


Observation:

Females are concerned about their mental health and 71.2% females are taking treatment

Question: What is the association between treatment and family history of the employee?

In [37]:
```
# plotting an empty figure of width 8 and height 6
fig = plt.figure(figsize=(8, 6))

# Plot the coutplot figure
ax = sns.countplot(x='family_history', hue='treatment', data=data, palette = 'hls'

# to label the values
for i in ax.containers:
    ax.bar_label(i,)

plt.title('Does family history affects mental health?', size=16)

plt.xlabel('Family History', size=14)
plt.ylabel('Count', size=14)

# Display the figure
plt.show()
```



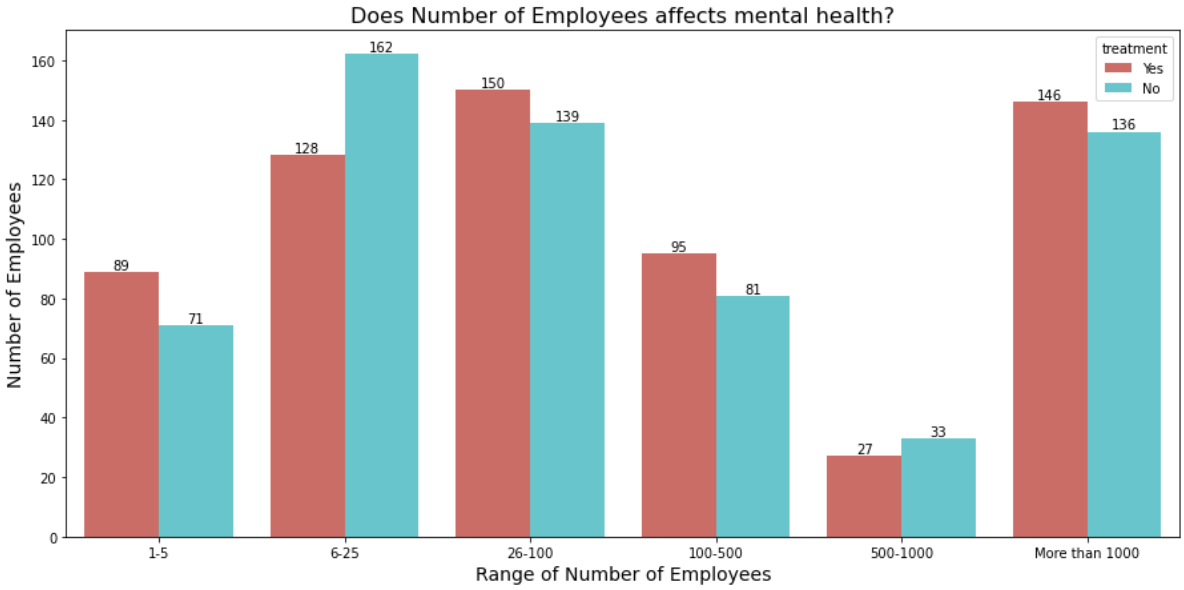
Observation:

Employees who have family history are very much likely to go for treatment

Question: What is the association between treatment and number of the employee in the company?

In [38]:
```
# plotting an empty figure of width 15 and height 7
fig = plt.figure(figsize=(15, 7))
```

```
# Plot the coutplot figure
order = ['1-5', '6-25', '26-100', '100-500', '500-1000', 'More than 1000']
ax = sns.countplot(x='no_employees', hue='treatment',data = data, order = order, p

# to label the values
for i in ax.containers:
    ax.bar_label(i,)

plt.title('Does Number of Employees affects mental health?', size=16)

plt.xlabel('Range of Number of Employees', size=14)
plt.ylabel('Number of Employees', size=14)

# Display the figure
plt.show()
```

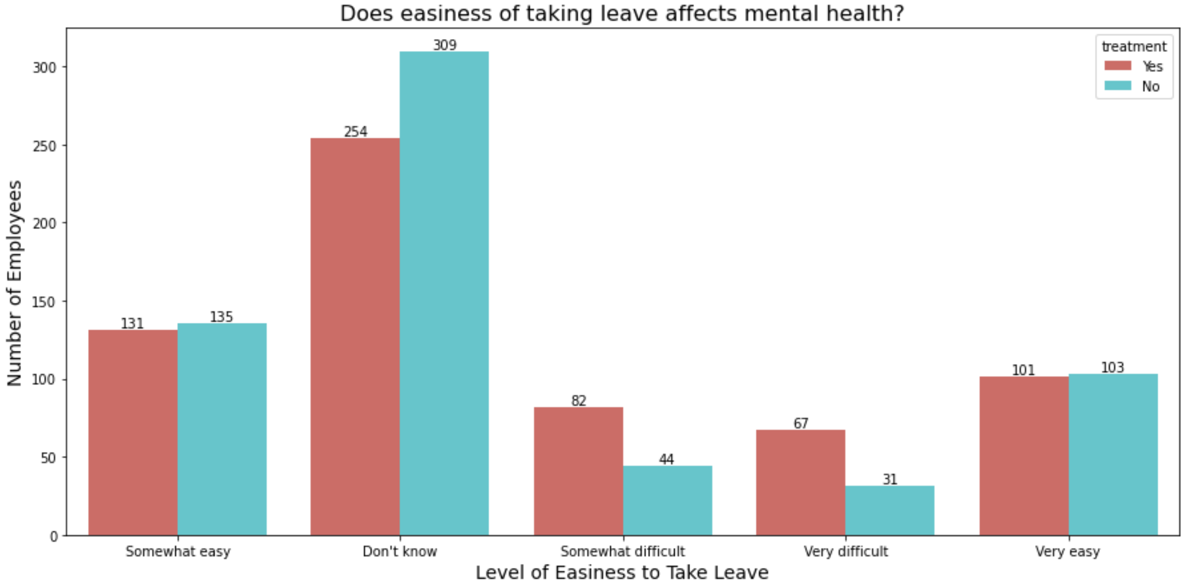


Observation:

More number of people are going for treatment when the company size is 26-100. It is also noticed that less number of people are going for treatment when the company size is 6-25

Question: What is the association between treatment and easiness of taking leave in the company?

In [38]:
```
# plotting an empty figure of width 15 and height 7
fig = plt.figure(figsize=(15, 7))

# Plot the coutplot figure
ax = sns.countplot(x='leave', hue='treatment',data = data, palette = 'hls')

# to label the values
for i in ax.containers:
    ax.bar_label(i,)

plt.title('Does easiness of taking leave affects mental health?', size=16)

plt.xlabel('Level of Easiness to Take Leave', size=14)
plt.ylabel('Number of Employees', size=14)

# Display the figure
plt.show()
```

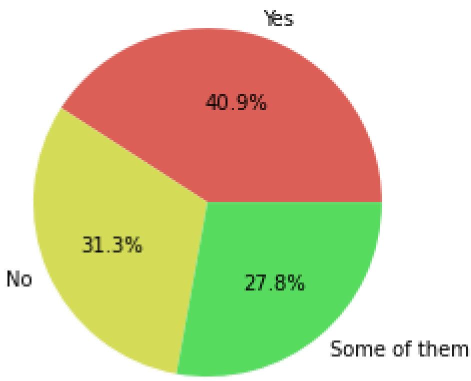


Observation:

Many people are not ready to open their mind in this case. But, we can see number of people going for treatment for the cases - 'Somewhat difficult' and 'Very difficult' is more.

Question: Are the employees willing to discuss their mental problems with their supervisors?

In [39]:
```
# declaring data
data = data.supervisor.value_counts().to_list()
keys = ['Yes', 'No', 'Some of them']

# plotting data on chart
plt.pie(data, labels = keys, autopct='%1.1f%%', colors=sns.color_palette('hls'))

# displaying chart
plt.show()
```



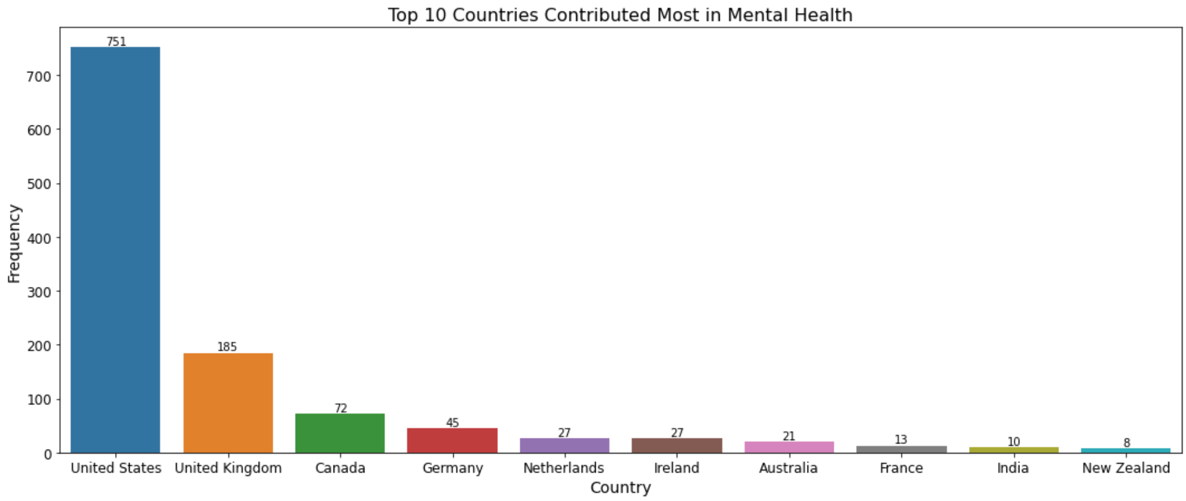
Observation:

41% of the employees are willing to discuss their mental problems with supervisors. Others are having mixed reactions.

Question: Which countries have contributed the most in terms of mental health?

In [43]:

```
from collections import Counter

# Get top 10 countries name and frequency
country_count = Counter(data['Country'].dropna().tolist()).most_common(10)
country_index = [country[0] for country in country_count]
country_val = [country[1] for country in country_count]

# plotting an empty figure of width 18 and height 7
fig = plt.figure(figsize=(18, 7))

# Plot the barplot figure
ax = sns.barplot(x = country_index,y= country_val)

# to label the values
for i in ax.containers:
    ax.bar_label(i,)

plt.title(label='Top 10 Countries Contributed Most in Mental Health', size=16)

plt.xlabel(xlabel = 'Country', size=14)
plt.ylabel(ylabel = 'Frequency', size=14)

plt.xticks(size=12)
plt.yticks(size=12)

# Output the figure
plt.show()
```



Observation:

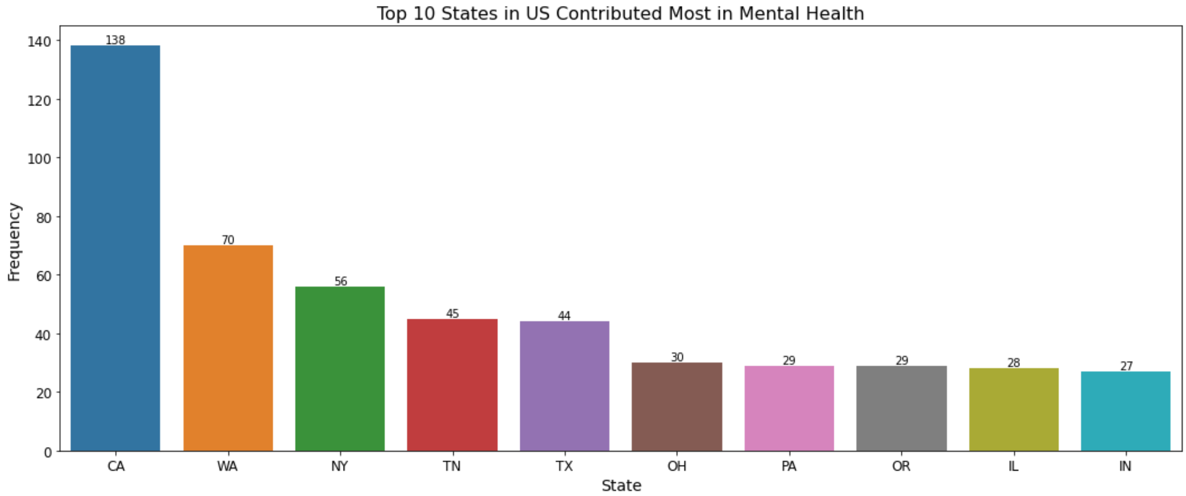
US contributed the most with 750 respondents

Question: Which states in US contributed the most in terms of mental health?

In [44]:

```
# Extract states data of US
usa_data = data[data['Country']=='United States']
frequency = usa_data['state'].value_counts()[0:10].values
labels = usa_data['state'].value_counts()[0:10].index

# plotting an empty figure of width 18 and height 7
fig = plt.figure(figsize=(18, 7))

# Plot the barplot figure
```

```
ax = sns.barplot(x = labels, y = frequency)

# to label the values
for i in ax.containers:
    ax.bar_label(i,)

plt.title(label='Top 10 States in US Contributed Most in Mental Health', size=16)

plt.xlabel(xlabel='State', size=14)
plt.ylabel(ylabel='Frequency', size=14)

plt.xticks(size=12)
plt.yticks(size=12)

# Output the figure
plt.show()
```



Observation:

CALIFORNIA is the state that contributed the most with 149 respondents

Question: What is the contribution of top 3 countries in terms of mental health?

In [45]:
```
# Create a new dataframe based on top 3 countries
countries = pd.concat([data.loc[data['Country'] == 'United States'],
                       data.loc[data['Country'] == 'United Kingdom'],
                       data.loc[data['Country'] == 'Canada']]).reset_index(drop=Tru

# Display the results
print('The number of contribution of top 3 countries in terms of mental health:',co
print('Their proportion from total people surveyed is ', np.round((countries.shape
```

```
The number of contribution of top 3 countries in terms of mental health: 1008
Their proportion from total people surveyed is  80.06 %
```

Observation:

Top 3 countries - US, UK, Canada contributed 80.11% to the survey in mental health

Summarization

The mental health survey helped to understand the mental condition of employees working in tech firms across countries. A total of 1259 entries were recorded during the survey out of which 1007 were recorded from the top 3 countries.The United States leads the chart in terms of participation in the survey followed by the United Kingdom and Canada. From a state point of view, California leads the chart when run down the analysis.

The following set of parameters are found to be affecting mental health the most and thus requires treatment: Age, Gender, Family history, Level of easiness to take leave, and Number of employees in a company.

There should be an awareness program about mental health and its effects in every company. Relationship Managers should be supportive with the right guidance towards their employees.