SQL codes

1. Objective

Objective of this project to do initial research about mental health of technology employees using survey data in years 2014, 2016, 2017, 2018 and 2019.The data is taken from Open Source Mental Illness (OSMI - Website: https://osmihelp.org/research (https://osmihelp.org/research)).

2. Data

2.1 Importing essential libraries

In [1]:
```
# Importing Essential Libraries

import sqlite3 # Importing SQL Lite3
import pandas as pd # Data manipulation and analysis
import matplotlib as pltm # Data vizualisation
import numpy as np # Multi array and matrix calculations
import seaborn as sb # Data vizualisation
import matplotlib.pyplot as plt # Data vizualisation

from pandas import Series, DataFrame #Data manipulation and analysis
from matplotlib import rcParams
from matplotlib import pyplot
pd.set_option("display.max_rows", None, "display.max_columns", None)
```

2.2 Connecting SQlite3

In [2]:
```
db = sqlite3.connect(

    '/Users/ruhidmirzayev/downloads/mental_health.sqlite'
                     )
# db stands for databas
```

In [3]:
```
data = pd.read_sql_query(
''' SELECT name from sqlite_master
    WHERE type= 'table';''', db)
data.head()
```

Out[3]:

	name
0	Answer
1	Question
2	Survey

In [4]:
```
table1 = pd.read_sql(

    '''SELECT * FROM QUESTION; '''

    ,db)
table1.head()# Questions asked in survey
```

Out[4]:

	questiontext	questionid
0	What is your age?	1
1	What is your gender?	2
2	What country do you live in?	3
3	If you live in the United States, which state ...	4
4	Are you self-employed?	5

In [5]:
```
print('The number of columns and rows are:', table1.shape )
```

```
The number of columns and rows are: (105, 2)
```

In [6]:
```
table2 = pd.read_sql(

''' SELECT * FROM Answer;'''

    ,db)

table2.head()
```

Out[6]:

	AnswerText	SurveyID	UserID	QuestionID
0	37	2014	1	1
1	44	2014	2	1
2	32	2014	3	1
3	31	2014	4	1
4	31	2014	5	1

In [7]:
```
print('The number of columns and rows are:', table2.shape )
```

```
The number of columns and rows are: (236898, 4)
```

In [8]:
```
table3 = pd.read_sql(

    '''SELECT * FROM Survey;'''

    , db)
table3.head() # Survey description
```

Out[8]:

	SurveyID	Description
0	2014	mental health survey for 2014
1	2016	mental health survey for 2016
2	2017	mental health survey for 2017
3	2018	mental health survey for 2018
4	2019	mental health survey for 2019

In [9]:
```
print('The number of columns and rows are:', table3.shape )
```

```
The number of columns and rows are: (5, 2)
```

3. Data Cleaning

3.1 Dublicates

UserID looks like has unique values, let's double check.

In [10]:
```
dublicates = pd.read_sql(

''' SELECT SurveyID, UserID, COUNT(UserID) FROM Answer
    GROUP BY UserID
    HAVING COUNT(UserID)>1 '''

    ,db)
dublicates.head()
```

Out[10]:

	SurveyID	UserID	COUNT(UserID)
0	2014	1	26
1	2014	2	26
2	2014	3	26
3	2014	4	26
4	2014	5	26

In [11]:
```
userid_unique1 = pd.read_sql(

''' SELECT COUNT(DISTINCT UserID) as Total FROM Answer '''

    ,db)
print ('The numer of users:')
userid_unique1.head()
```

```
The numer of users:
```

Out[11]:

	Total
0	4218

In both cases, the number of users equal to 4218.

3.2 Cleaning data

Age

In [12]:
```
age_count = pd.read_sql(

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 1 GROUP BY AnswerText;'''
    ,db)
age_count
```

Out[12]:

	AnswerText	count(AnswerText)
0	-1	5
1	-29	1
2	0	1
3	11	1
4	15	1
5	17	1
6	18	9
7	19	20
8	20	17
9	21	39
10	22	74

As we see from the above results, there are numbers like -1, -29, 0, 8, and 11 which do not look like the age for employees.

In [13]:
```
age = db.execute(

''' DELETE
    FROM Answer
    WHERE AnswerText NOT IN (SELECT AnswerText FROM Answer Where AnswerText Between 16 AND 80)
    AND QuestionID = 1;''')
```

In [14]:
```
# After cleaning
age_cleaning = pd.read_sql(

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 1 GROUP BY AnswerText
    ORDER BY AnswerText DESC;'''

    ,db)
age_cleaning
```

Out[14]:

	AnswerText	count(AnswerText)
0	8	1
1	74	1
2	72	1
3	70	1
4	67	2
5	66	2
6	65	3
7	64	3
8	63	5
9	62	3
10	61	7
11	60	5
12	59	6
13	58	5
14	57	14
15	56	13
16	55	22
17	54	17
18	53	15
19	52	17
20	51	22
21	50	30
22	5	1
23	49	36
24	48	29
25	47	38
26	46	58
27	45	74
28	44	68
29	43	82
30	42	100
31	41	88
32	40	122
33	39	137
34	38	160
35	37	184
36	36	147

37	35	201
38	34	202
39	33	201
40	329	1
41	323	1
42	32	227
43	31	223
44	30	250
45	3	1
46	29	229
47	28	220
48	27	197
49	26	194
50	25	147
51	24	128
52	23	107
53	22	74
54	21	39
55	20	17
56	19	20
57	18	9
58	17	1

We still have numbers like, 3 and 5.This is because our column AnswerText is not integer.

In [15]:
```
# We added 0 after column and after numers to make values integer.
age = db.execute(

''' DELETE
    FROM Answer
    WHERE AnswerText NOT IN (SELECT AnswerText FROM Answer Where (AnswerText + 0) Between 16 + 0 AND 80
    AND QuestionID = 1;''')
```

In [16]:
```
age_cleaned = pd.read_sql(

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 1 GROUP BY AnswerText
    ORDER BY AnswerText DESC;'''

    ,db)
age_cleaned
```

Out[16]:

	AnswerText	count(AnswerText)
0	74	1
1	72	1
2	70	1
3	67	2
4	66	2
5	65	3
6	64	3
7	63	5
8	62	3
9	61	7
10	60	5
11	59	6
12	58	5

13	57	14
14	56	13
15	55	22
16	54	17
17	53	15
18	52	17
19	51	22
20	50	30
21	49	36
22	48	29
23	47	38
24	46	58
25	45	74
26	44	68
27	43	82
28	42	100
29	41	88
30	40	122
31	39	137
32	38	160
33	37	184
34	36	147
35	35	201
36	34	202
37	33	201
38	32	227
39	31	223
40	30	250
41	29	229
42	28	220
43	27	197
44	26	194
45	25	147
46	24	128
47	23	107
48	22	74
49	21	39
50	20	17
51	19	20
52	18	9
53	17	1

Gender

In [17]:
```
gender_count = pd.read_sql(

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 2
    GROUP BY AnswerText ORDER BY count(AnswerText) DESC;'''

    ,db)
gender_count.head(20)
```

Out[17]:

	AnswerText	count(AnswerText)
0	Male	2830
1	Female	914
2	male	212
3	female	110
4	-1	24
5	Nonbinary	8
6	non-binary	6
7	Genderqueer	4
8	Agender	4
9	None	3
10	Non-binary	3
11	Genderfluid	3
12	agender	2
13	Trans woman	2
14	Trans man	2
15	Other	2
16	Non binary	2
17	Male-ish	2
18	Female (trans)	2
19	Enby	2

Some of answer to the gender question is not clear to define.

In [18]:
```
# Lowering all Capital letters and make them same.
Gender_lower = db.execute(

''' UPDATE Answer
    SET AnswerText = LOWER(AnswerText)
    WHERE QuestionID = 2;''')
```

In [19]:
```
# Let's delete rows where we can not identify the gender
gender_cleaning = db.execute(

''' DELETE FROM Answer
    WHERE QuestionID = 2  AND AnswerText
        NOT IN (SELECT AnswerText
        FROM Answer
        WHERE AnswerText IN ('male', 'female'));''')
```

In [20]:
```
# Gender values cleaned as we see from below.
gender_cleaned = pd.read_sql(

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 2
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)

gender_cleaned
```

Out[20]:

	AnswerText	count(AnswerText)
0	male	3043
1	female	1024

Country

In [21]:
```
country = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 3
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
country.head(100)
```

Out[21]:

	AnswerText	count(AnswerText)
0	United States of America	1853
1	United States	751
2	United Kingdom	482
3	Canada	199
4	Germany	136
5	Netherlands	98
6	Australia	73
7	Ireland	51
8	France	51
9	India	50
10	Brazil	37

United States of America and United States are the same coutry but different data entry. We can set all of them to Uneted States. There is country name -1 and 'Bahamas, The'. We can delet the and drop -1.

In [22]:
```
country_cleaning = db.execute(
''' Update Answer
    SET AnswerText =
        (CASE
            WHEN AnswerText = 'United States' THEN 'USA'
            WHEN AnswerText = 'United States of America' THEN 'USA'
            WHEN AnswerText = 'United Kingdom' THEN 'UK'
            WHEN AnswerText = 'Bahamas, The' THEN 'Bahamas'
            WHEN AnswerText = -1 THEN NULL
            ELSE AnswerText
        END)
    WHERE QuestionID = 3;''')
```

In [23]:
```
country_cleaned = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 3
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
country_cleaned
```

Out[23]:

	AnswerText	count(AnswerText)
0	USA	2604
1	UK	482
2	Canada	199
3	Germany	136
4	Netherlands	98
5	Australia	73
6	Ireland	51
7	France	51
8	India	50
9	Brazil	37
10	Sweden	29

Self-employed

In [24]:
```
SelfEmployed = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 5
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
SelfEmployed
```

Out[24]:

	AnswerText	count(AnswerText)
0	0	3550
1	1	650
2	-1	18

It looks lie 1 for yes, 0 for no. -1 does not make sense, we can delete them.

In [25]:
```
SE_cleaning = db.execute(
''' UPDATE Answer
    SET AnswerText = NULL
    WHERE AnswerText = -1
    AND QuestionID = 5;''')
```

In [26]:
```
SE_Cleaned = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 5
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
SelfEmployed
```

Out[26]:

	AnswerText	count(AnswerText)
0	0	3550
1	1	650
2	-1	18

Mental Health in Family History

In [27]:
```
FamMent = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 5
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
FamMent
```

Out[27]:

	AnswerText	count(AnswerText)
0	0	3550
1	1	650
2	None	0

Here we do not need to clean it.

Number of Employees in Company

In [28]:
```
EmpNum = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 8
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
EmpNum
```

Out[28]:

	AnswerText	count(AnswerText)
0	More than 1000	912
1	26-100	824
2	100-500	788
3	6-25	689
4	-1	504
5	1-5	254
6	500-1000	247

The number of self-employed people were 650, but here 1-5 is just 254. The data does not seem quite precise. We can delete rows consisting -1.

In [29]:
```
EmpNum_cleaning = db.execute(
''' DELETE FROM Answer
    WHERE QuestionID = 8
    AND AnswerText = -1''')
```

In [30]:
```
EmpNum_cleaned = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 8
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
EmpNum_cleaned
```

Out[30]:

	AnswerText	count(AnswerText)
0	More than 1000	912
1	26-100	824
2	100-500	788
3	6-25	689
4	1-5	254
5	500-1000	247

Still, this values are not numbers, rather strings

Mental Health Disorder

In [31]:
```
MentalHealth = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 32
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
MentalHealth
```

Out[31]:

	AnswerText	count(AnswerText)
0	Yes	1417
1	No	896
2	Possibly	275
3	Maybe	246
4	Don't Know	109
5	-1	15

In [32]:
```
MentalHealth_cleaned = db.execute(
''' DELETE FROM Answer
    WHERE QuestionID = 32
    AND AnswerText = -1''')
```

Mental Health Disorder Diagnozed

In [33]:
```
MentalHealth_diagnozed = pd.read_sql (

''' SELECT AnswerText, count(AnswerText) FROM Answer
    WHERE QuestionID = 34
    GROUP BY AnswerText
    ORDER BY count(AnswerText) DESC;'''

    ,db)
MentalHealth_diagnozed
```

Out[33]:

	AnswerText	count(AnswerText)
0	Yes	1363
1	-1	863
2	No	732

In [34]:
```
MentalHealth_diagnozed_cleaned = db.execute(

''' DELETE FROM Answer
    WHERE QuestionID = 34
    AND AnswerText = -1 ''')
```

In [35]:
```
race_cleaned = db.execute(

''' DELETE FROM Answer
    WHERE QuestionID = 89
    AND AnswerText = -1 ''')
```

4. Data Visualisation

4.1 Function for Pie Chart

In [36]:
```
# Inserting additional libraries
import plotly.express as px
import plotly.graph_objs as go
```

We need to join tables Answer and Question for data visualization.

In [37]:
```
dt_join = pd.read_sql(

''' SELECT * FROM Answer
    INNER JOIN Question on
    Answer.QuestionID=Question.questionid'''

    ,db)
dt_join.head()
```

Out[37]:

	AnswerText	SurveyID	UserID	QuestionID	questiontext	questionid
0	37	2014	1	1	What is your age?	1
1	44	2014	2	1	What is your age?	1
2	32	2014	3	1	What is your age?	1
3	31	2014	4	1	What is your age?	1
4	31	2014	5	1	What is your age?	1

In [38]:
```
dt = dt_join[['SurveyID','questiontext','AnswerText']]
dt.head()
```

Out[38]:

	SurveyID	questiontext	AnswerText
0	2014	What is your age?	37
1	2014	What is your age?	44
2	2014	What is your age?	32
3	2014	What is your age?	31
4	2014	What is your age?	31

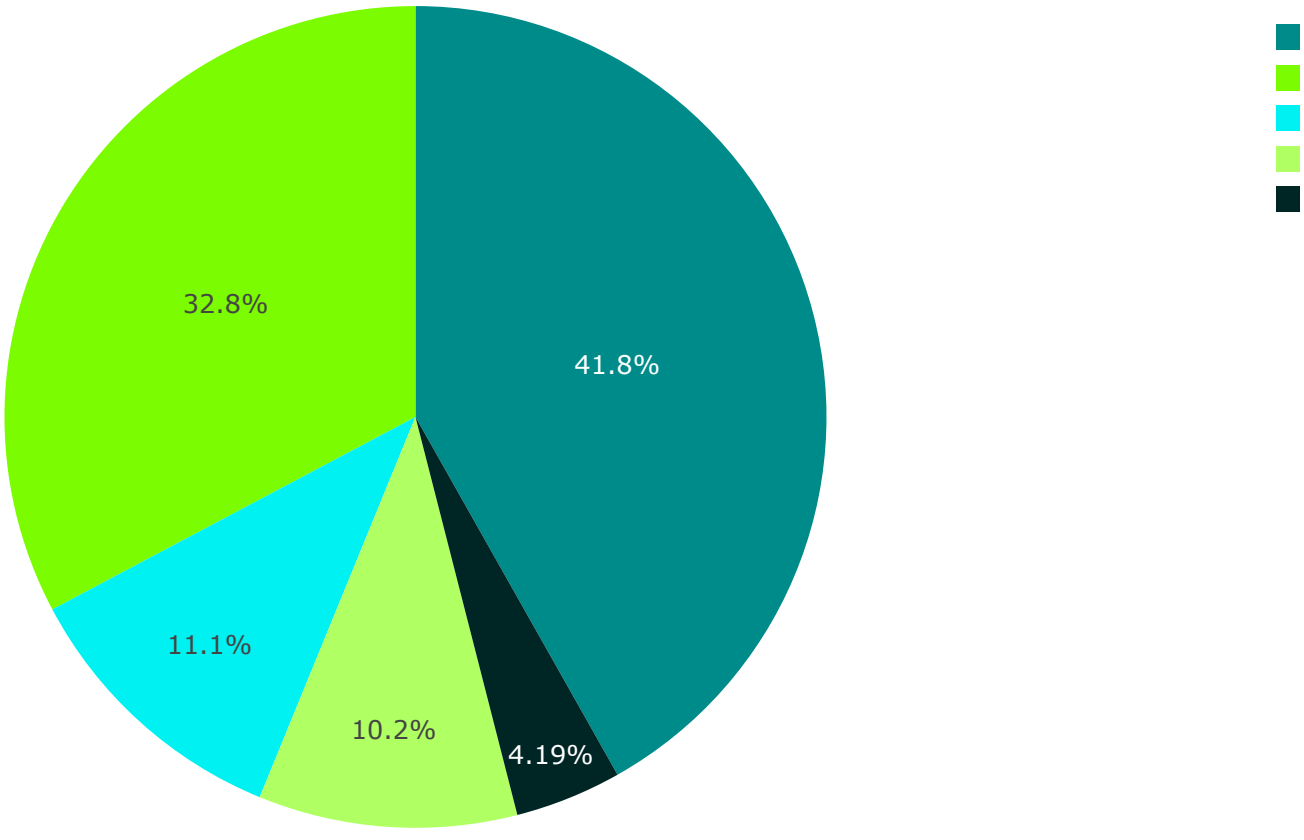
In [39]:
```
def find(dt):
    function = dt.value_counts()
    return function

def find_percent(dt):
    percent = dt.value_counts()*100/len(dt)
    return percent
```

In [40]:
```
# Defining our pie chart function, this is because we data visualization is easier while all observation
# one column
def pie_chart(dt):

    targets = list(find_percent(dt).index)
    values = list(find_percent(dt).values)

    fig = px.pie(
        values=values,
        names=targets,
        color_discrete_sequence=['darkcyan', 'lawngreen'])
    fig.show()
```

4.2 Visualizing questions and answers

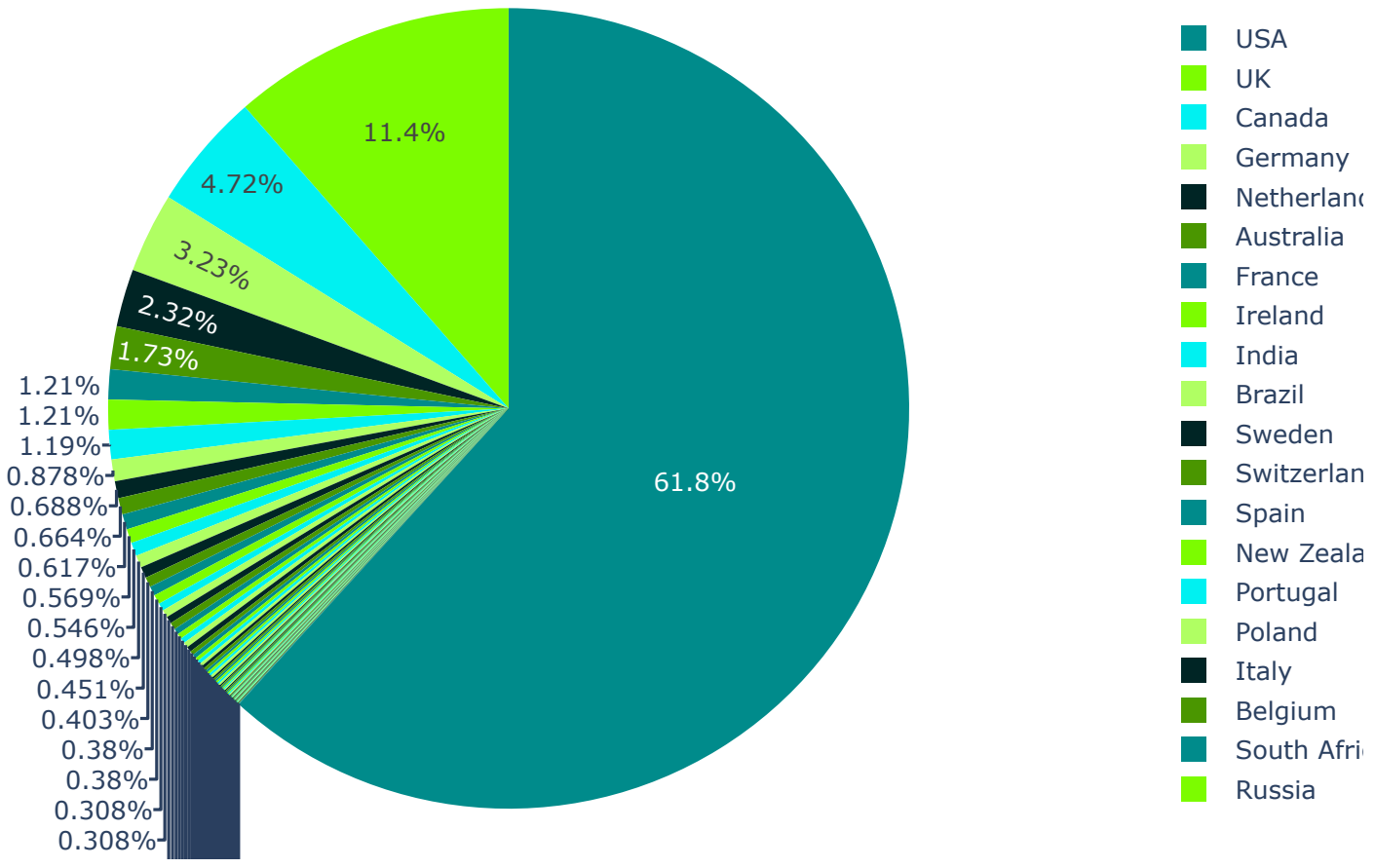
Do you currently have a mental health disorder?

In [41]:
```
pie_chart(dt[dt['questiontext'] == 'Do you currently have a mental health disorder?']['AnswerText'])
```



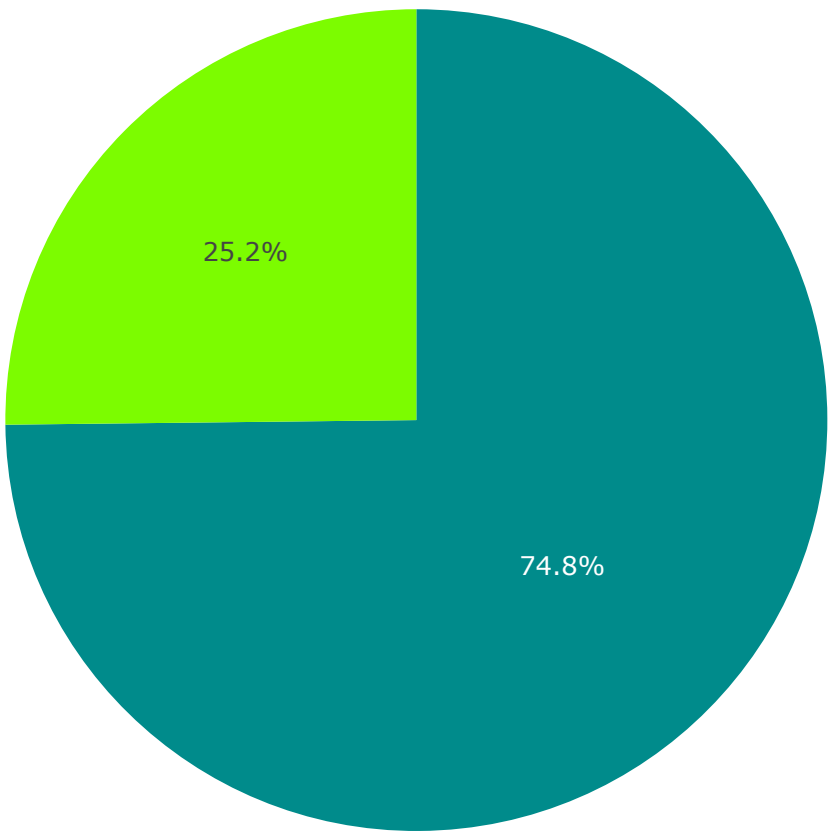
What country do you live in?

In [42]:
```
pie_chart(dt[dt['questiontext'] == 'What country do you live in?']['AnswerText'])
```



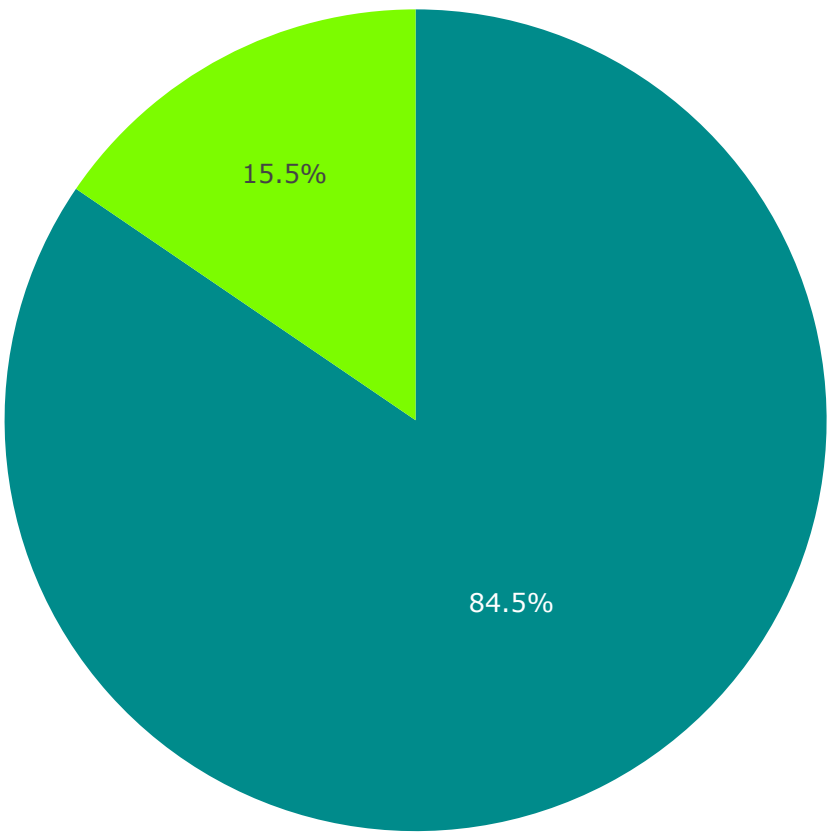
What is your gender?

In [43]:
```
pie_chart(dt[dt['questiontext'] == 'What is your gender?']['AnswerText'])
```



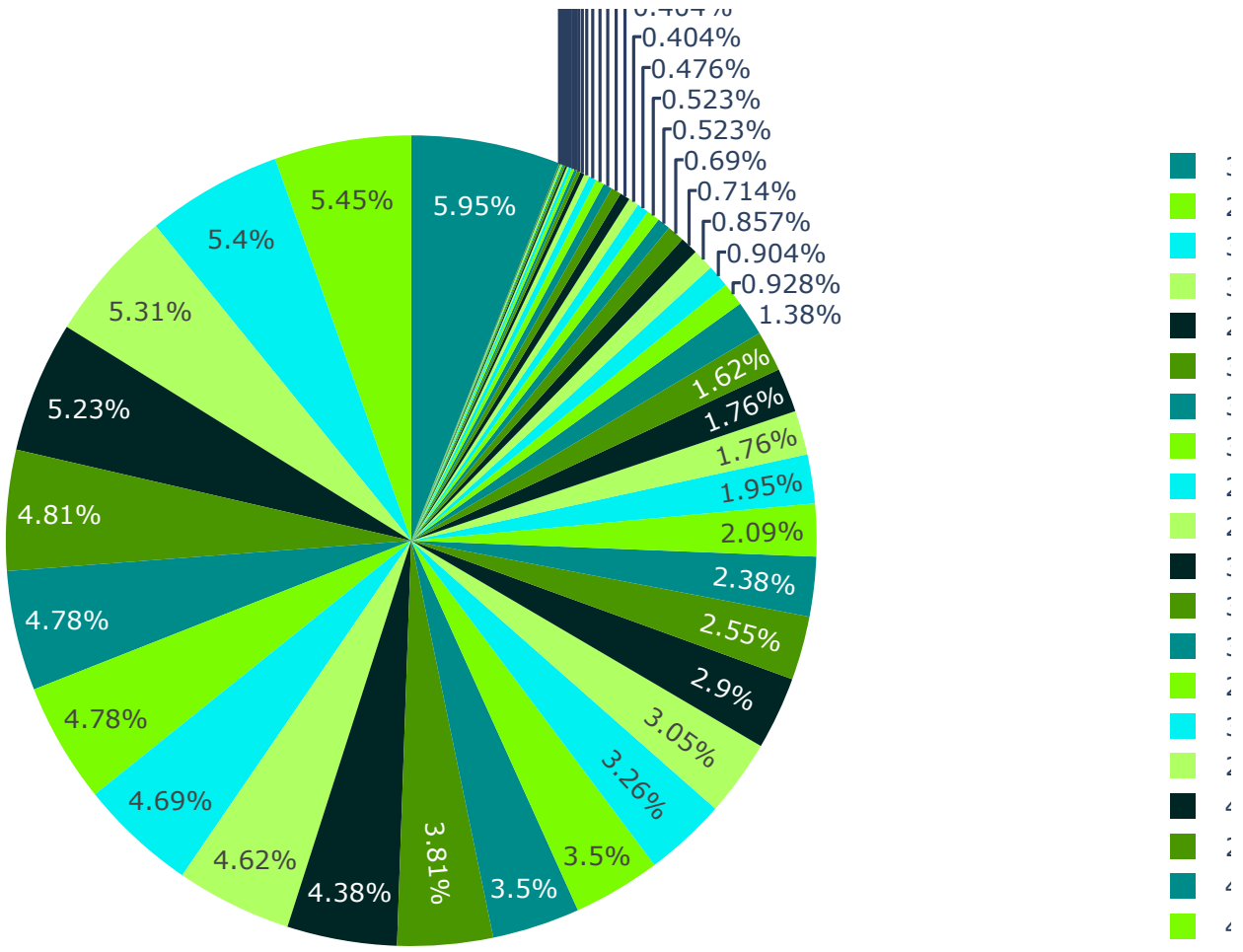
Are you self-employed?

In [44]:
```
pie_chart(dt[dt['questiontext'] == 'Are you self-employed?']['AnswerText'])
```



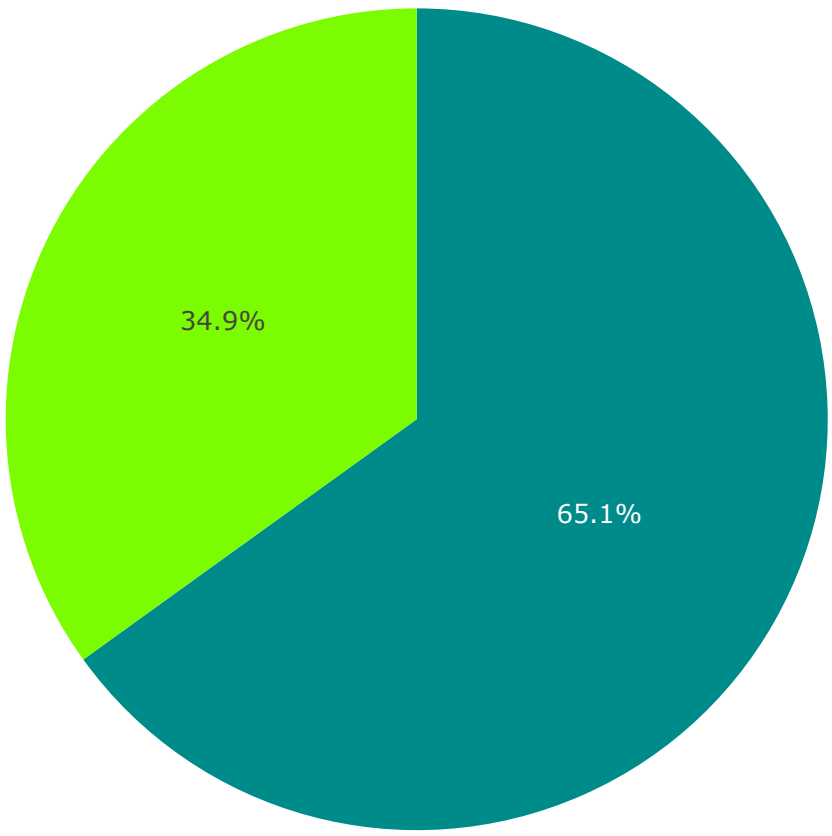
What is your age?

In [45]:
```
pie_chart(dt[dt['questiontext'] == 'What is your age?']['AnswerText'])
```



Have you ever been diagnosed Mental Health Disorder?

In [46]:
```
pie_chart(dt[dt['questiontext'] == 'Have you ever been diagnosed with a mental health disorder?']['Answe
```



What is your race?

In [47]:
```
pie_chart(dt[dt['questiontext'] == 'What is your race?']['AnswerText'])
```