In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
data = pd.read_csv('Absenteeism_at_work.csv')
```

In [5]:
```
col_string = list(data.columns)[0]
columns = col_string.split(';')
```

In [6]:
```
dataFrame = {}
for i in range(data.shape[0]):
    val = list(data.loc[i])[0]
    values = val.split(';')
    values = list(map(float, values))

    for j in range(len(values)):
        if columns[j] in dataFrame:
            dataFrame[columns[j]].append(values[j])
        else:
            dataFrame[columns[j]] = [values[j]]

fData = pd.DataFrame(dataFrame)
fData.head()
```

Out[6]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work loa Average/da
0	11.0	26.0	7.0	3.0	1.0	289.0	36.0	13.0	33.0	239.55
1	36.0	0.0	7.0	3.0	1.0	118.0	13.0	18.0	50.0	239.55
2	3.0	23.0	7.0	4.0	1.0	179.0	51.0	18.0	38.0	239.55
3	7.0	7.0	7.0	5.0	1.0	279.0	5.0	14.0	39.0	239.55

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work loa Average/da
4	11.0	23.0	7.0	5.0	1.0	289.0	36.0	13.0	33.0	239.5

5 rows × 21 columns

In [18]:

```
ind = list(fData[fData['Absenteeism time in hours'] == 0].index)
ffData = fData.drop(ind, axis=0)
```

In [6]:

```
cat = ['Seasons', 'Disciplinary failure', 'Education', 'Son', 'Social d
rinker', 'Social smoker', 'Pet']
num = [
    'ID', 'Transportation expense', 'Distance from Residence to Work',
'Service time',
    'Age', 'Work load Average/day ', 'Hit target', 'Weight', 'Height',
'Body mass index',
    'Absenteeism time in hours'
]
```

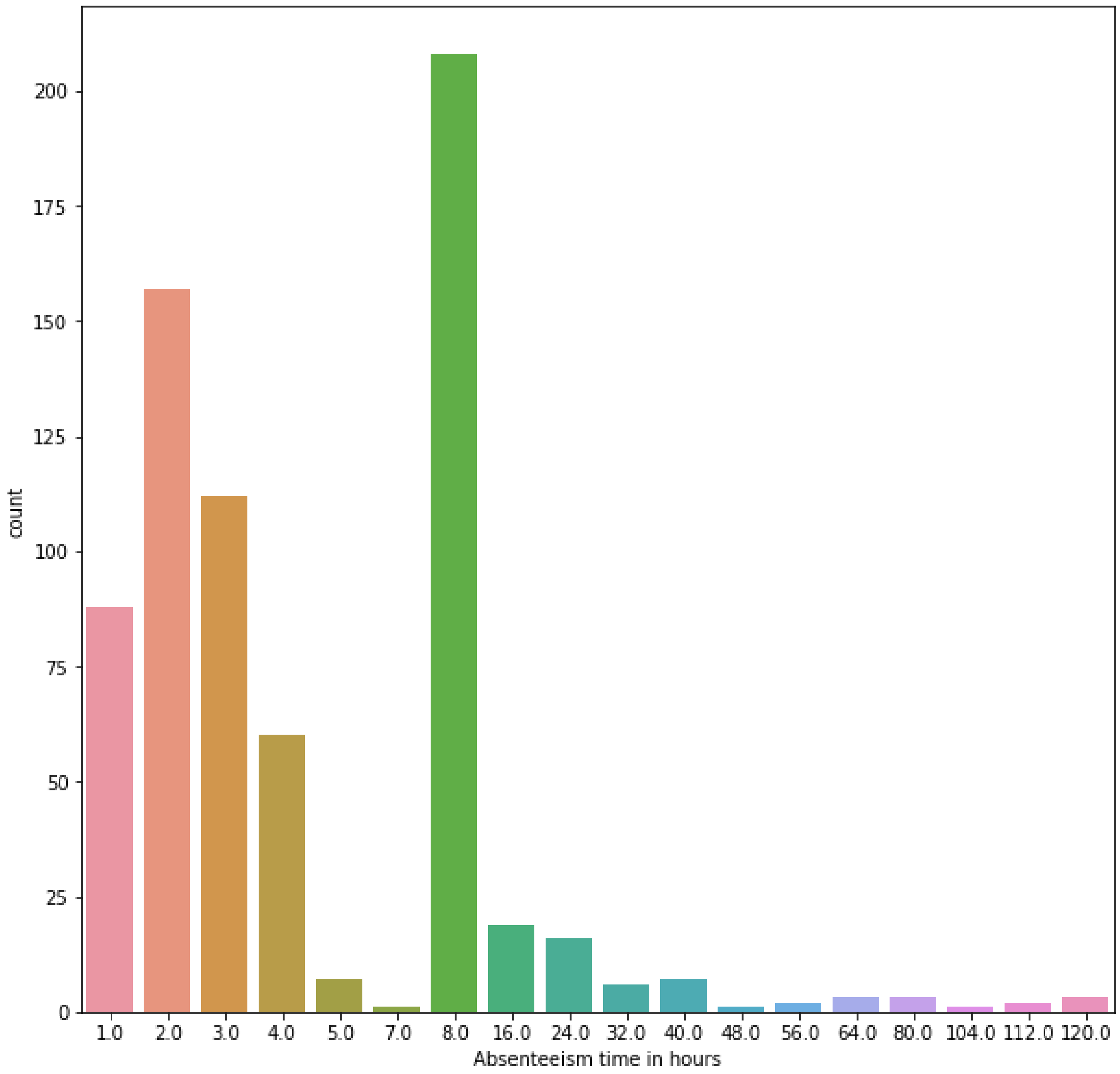
Hours count

In [27]:

```
plt.figure(figsize=(10, 10))
sns.countplot(ffData['Absenteeism time in hours'])
```

Out[27]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x151367c2160>
```

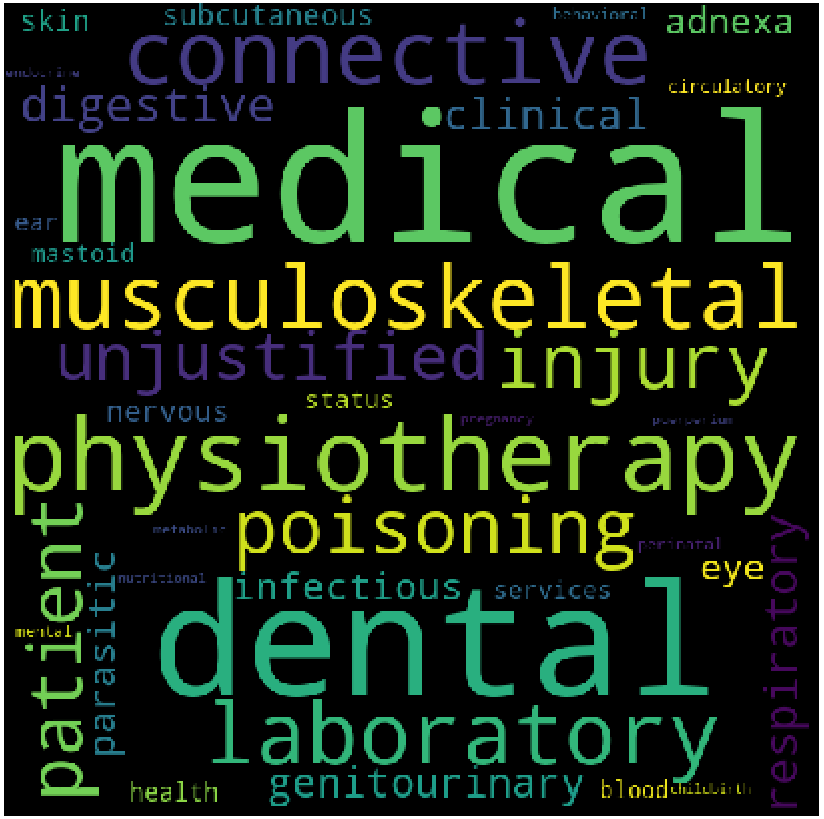


Majority of the employees were Absent for 8 hours and less. That is they were absent for one whole day if its a 9 to 5 job.

Most used Reasons for Absence.

In [88]:

```python
from wordcloud import WordCloud, STOPWORDS

'''
1 Certain infectious and parasitic diseases
2 Neoplasms
3 Diseases of the blood and blood-forming organs and certain disorders
 involving the immune
mechanism
4 Endocrine, nutritional and metabolic diseases
5 Mental and behavioral disorders
6 Diseases of the nervous system
7 Diseases of the eye and adnexa
8 Diseases of the ear and mastoid process
9 Diseases of the circulatory system
10 Diseases of the respiratory system
11 Diseases of the digestive system
12 Diseases of the skin and subcutaneous tissue
13 Diseases of the musculoskeletal system and connective tissue
14 Diseases of the genitourinary system
15 Pregnancy, childbirth and the puerperium
16 Certain conditions originating in the perinatal period
17 Congenital malformations, deformations and chromosomal abnormalities
18 Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified
19 Injury, poisoning and certain other consequences of external causes
20 External causes of morbidity and mortality
21 Factors influencing health status and contact with health services.

patient follow-up (22), medical consultation (23), blood donation (24),
laboratory examination (25), unjustified absence (26), physiotherapy (27), dental consultation (28).
'''


mapping = {
    1 : 'infectious parasitic',
    2 : 'neoplasms',
    3 : 'blood blood-forming immune',
    4 : 'endocrine nutritional metabolic',
    5 : 'mental behavioral',
    6 : 'nervous',
    7 : 'eye adnexa',
    8 : 'ear mastoid',
    9 : 'circulatory',
    10 : 'respiratory',
    11 : 'digestive',
    12 : 'skin subcutaneous',
    13 : 'musculoskeletal connective',
    14 : 'genitourinary',
    15 : 'pregnancy childbirth puerperium',
    16 : 'perinatal',
    17 : 'congenital malformations deformations chromosomal',
    18 : 'clinical laboratory',
    19 : 'injury poisoning',
    20 : 'morbidity mortality',
    21 : 'health status services',
    22 : 'patient',
    23 : 'medical',
    24 : 'blood',
    25 : 'laboratory',
    26 : 'unjustified',
    27 : 'physiotherapy',
    28 : 'dental'
}

reasons = list(ffData['Reason for absence'])
fString = ' '
for res in reasons:
    fString += mapping[int(res)] + ' '

wordcloud = WordCloud(width = 600, height = 600,
                background_color ='black',
                collocations = False,
                min_font_size = 10).generate(fString)

plt.figure(figsize = (6, 6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```



Took important medical terms from each reason and checked which medical term was used the most. It turned out to be Medical word was used most which corresponds to reason 23 and dental was used second most which corresponds to reason 28.

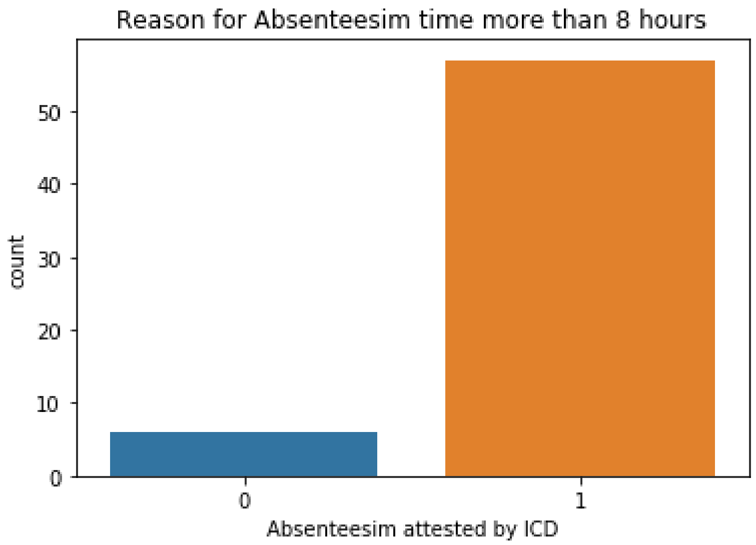
A look into Absentees with more than 8 hours.

```
In [20]: def change(col):
             if col[0] <= 21:
                 return 1
             else:
                 return 0

         ffData['icd_attested'] = ffData[['Reason for absence']].apply(change, a
         xis=1)
```

```
In [21]: temp = ffData[ffData['Absenteeism time in hours'] > 8.0]
```

```
In [22]: sns.countplot(temp['icd_attested'])
         plt.title('Reason for Absenteesim time more than 8 hours')
         plt.xlabel('Absenteesim attested by ICD')
```

```
Out[22]: Text(0.5, 0, 'Absenteesim attested by ICD')
```



57 of 63 with high absenteesim time have Absences attested by the International Code of Diseases (ICD). Hence it must have been due to some serious disease which made impossible for the employer to get to the office for many hours.

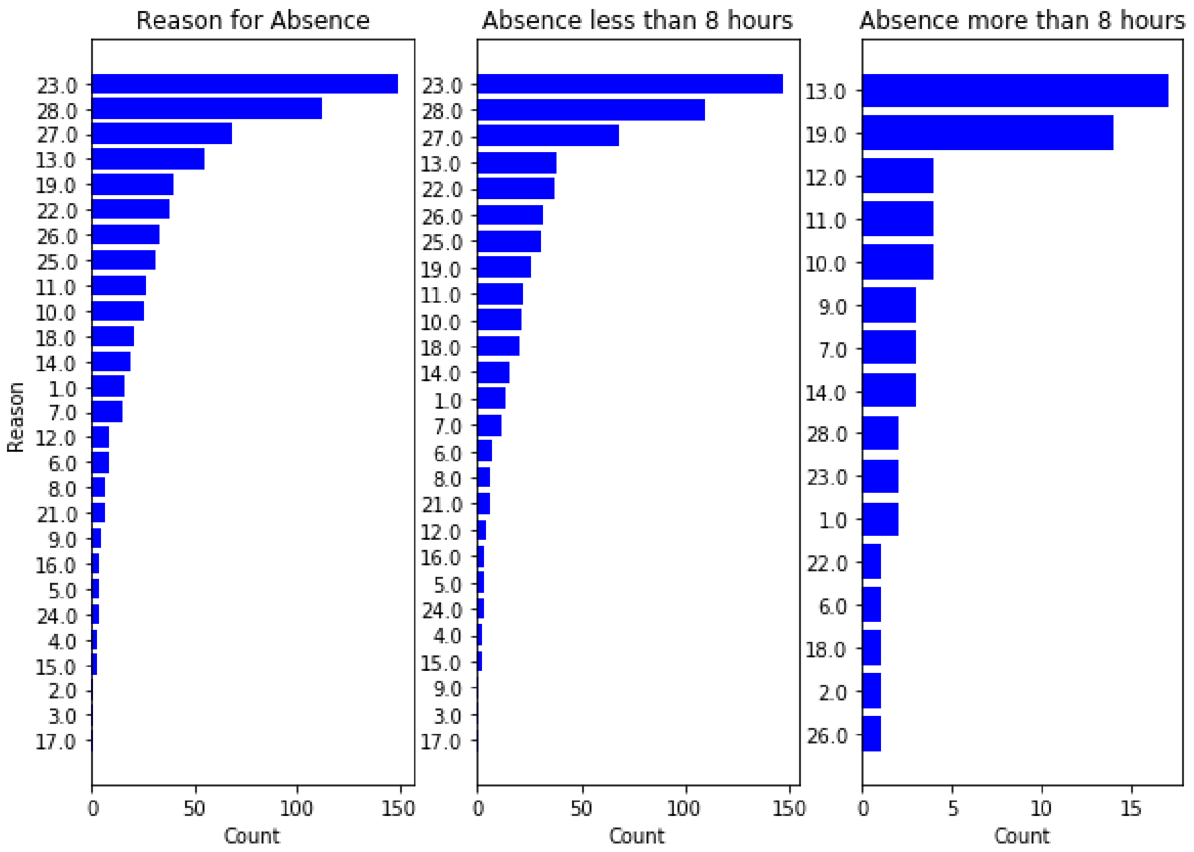
In [23]:

```
#temp[temp['icd_attested'] == 1]['Reason for absence']

fig, axes = plt.subplots(1, 3)
fig.set_figheight(7, 7)
fig.set_figwidth(10, 10)

freq = {}
val = list(ffData['Reason for absence'])
for v in val:
    if v in freq:
        freq[v] += 1
    else:
        freq[v] = 1
freq_s = sorted(freq.items(), key=lambda t : t[1])
x = [str(t[0]) for t in freq_s]
y = [t[1] for t in freq_s]
axes[0].barh(x, y, color='blue')
axes[0].set_title('Reason for Absence')
axes[0].set_xlabel('Count')
axes[0].set_ylabel('Reason')

t1 = ffData[ffData['Absenteeism time in hours'] <= 8.0]
freq2 = {}
val2 = list(t1['Reason for absence'])
for v in val2:
    if v in freq2:
        freq2[v] += 1
    else:
        freq2[v] = 1
freq_s = sorted(freq2.items(), key=lambda t : t[1])
x = [str(t[0]) for t in freq_s]
y = [t[1] for t in freq_s]
axes[1].barh(x, y, color='blue')
axes[1].set_title('Absence less than 8 hours')
axes[1].set_xlabel('Count')

t2 = ffData[ffData['Absenteeism time in hours'] > 8.0]
freq3 = {}
```

```
val3 = list(t2['Reason for absence'])
for v in val3:
    if v in freq3:
        freq3[v] += 1
    else:
        freq3[v] = 1
freq_s = sorted(freq3.items(), key=lambda t : t[1])
x = [str(t[0]) for t in freq_s]
y = [t[1] for t in freq_s]
axes[2].barh(x, y, color='blue')
axes[2].set_title('Absence more than 8 hours')
axes[2].set_xlabel('Count')
```

Out[23]:

```
Text(0.5, 0, 'Count')
```



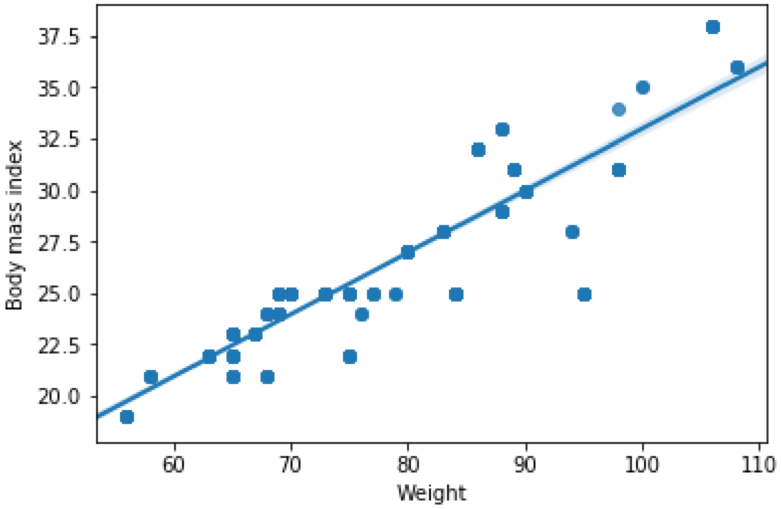
Weight vs BMI

In [24]:

```
sns.regplot(fData['Weight'], fData['Body mass index'])
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x151362bdcf8>
```



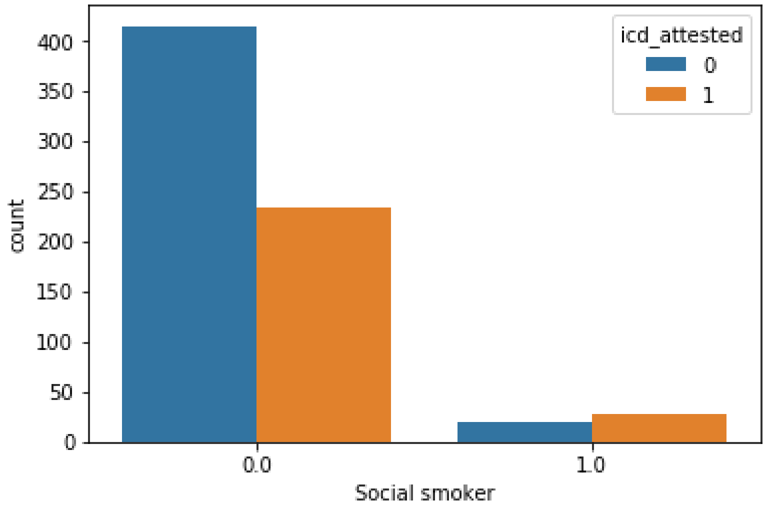
Smoking and Drinking

In [28]:

```
sns.countplot(ffData["Social smoker"], hue=ffData['icd_attested'])
```

Out[28]:

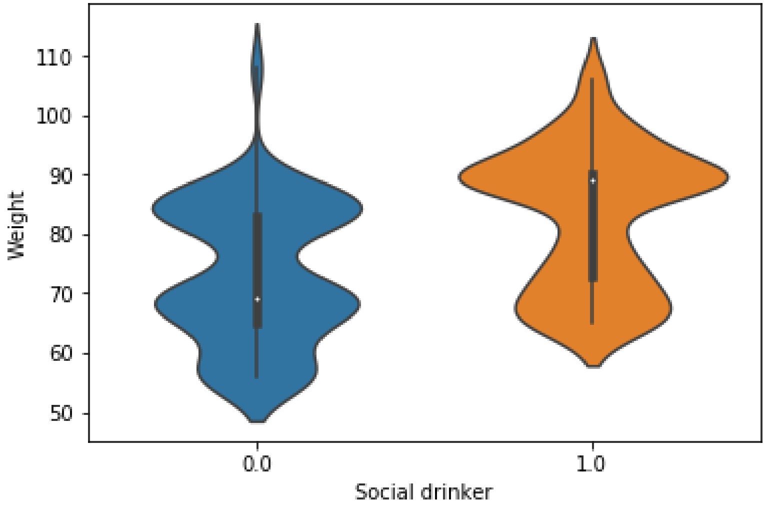
```
<matplotlib.axes._subplots.AxesSubplot at 0x1513684f8d0>
```



Smoking is bad for health.

```
In [29]: sns.violinplot(ffData["Social drinker"], ffData['Weight'])
```

```
Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x15136c37898>
```



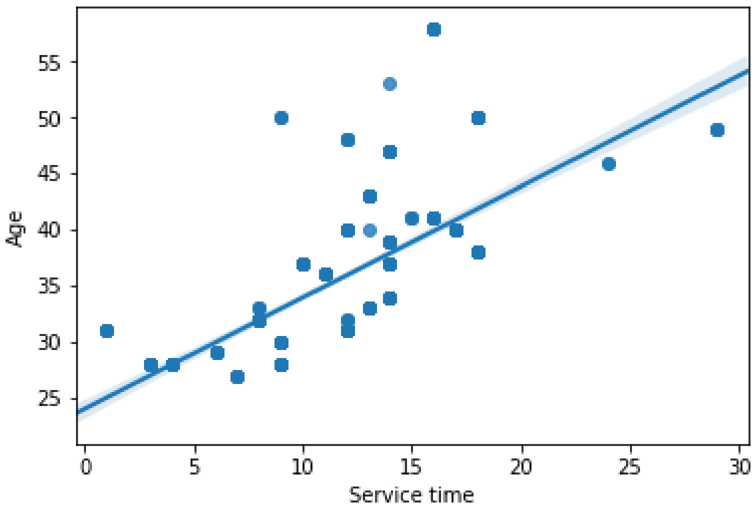
Alcohol can cause weight gain in four ways: it stops your body from burning fat, it's high in kilojoules, it can make you feel hungry , and it can lead to poor food choices.

Service Time

Service Time vs Age

```
In [30]: sns.regplot(fData['Service time'], fData['Age'])
```

```
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x15136c85cc0>
```



Service time indicates how many years he worked for that company.

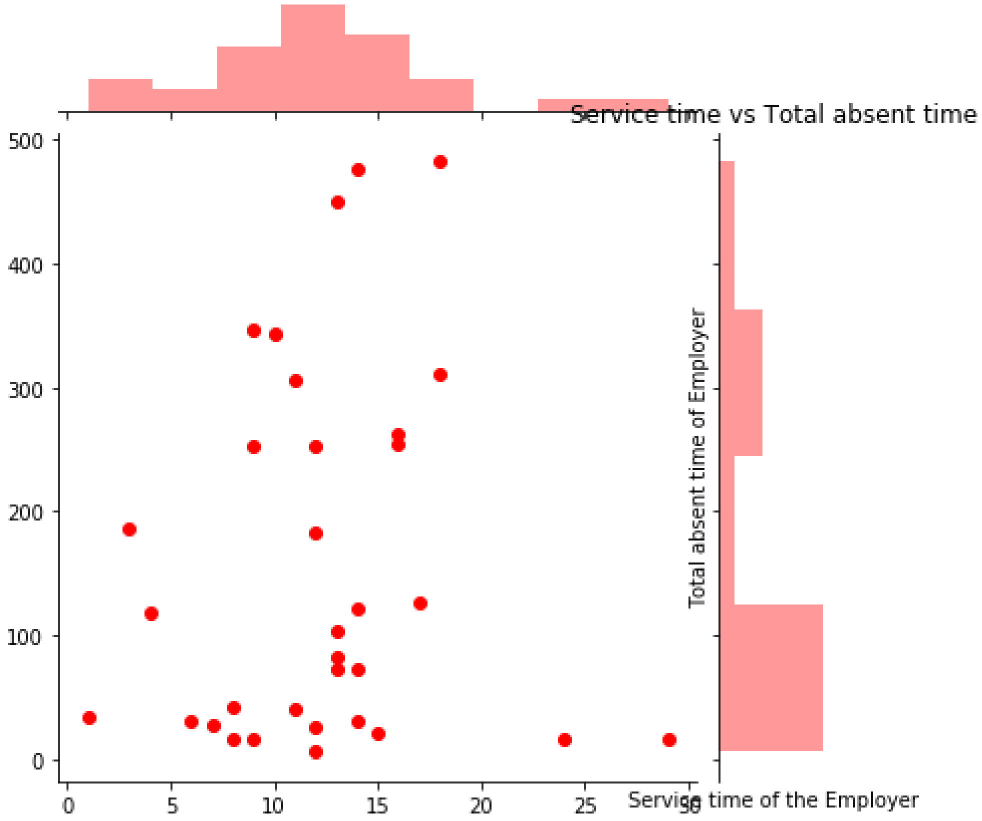
Service Time vs Total Absent Time

In [65]:

```
g_data = ffData.groupby('ID')

a = list(ffData['ID'])
b = list(ffData['Service time'])
store = {}
for i in range(len(a)):
    if a[i] not in store:
        store[a[i]] = b[i]

abs_total = dict(g_data['Absenteeism time in hours'].sum())

x = []
y = []
for k in abs_total:
    x.append(store[k])
    y.append(abs_total[k])

sns.jointplot(x=x,y=y, kind='scatter', color='red')
plt.xlabel("Service time of the Employer")
plt.ylabel("Total absent time of Employer")
plt.title("Service time vs Total absent time")
```

Out[65]:

```
Text(0.5, 1.0, 'Service time vs Total absent time')
```



Pet and Son vs Absteesim Time.

In [36]:

```
def binaryP(col):
    if col[0] == 1:
        return 'Have Pets'
    else:
        return 'No Pets'

def binaryC(col):
    if col[0] == 1:
        return 'Have Children'
    else:
        return 'No Children'

def ds(cols):
    d = cols[0]
    s = cols[1]

    if d == 0 and s == 0:
        return 'None'
    if d == 0 and s == 1:
        return 'Smoker'
    if d == 1 and s == 0:
        return 'Drinker'
    return 'Drinker & Smoker'

fData['Have Children?'] = fData[['Son']].apply(binaryC, axis=1)
fData['Have Pets?'] = fData[['Pet']].apply(binaryP, axis=1)
fData['Drinker or Smoker'] = fData[['Social drinker', 'Social smoker']]
.apply(ds, axis=1)
```

In [37]:

```
import numpy as np

def logT(col):
    return np.log(col[0]+1)

fData['Absent time log'] = fData[['Absenteeism time in hours']].apply(l
ogT, axis=1)
```

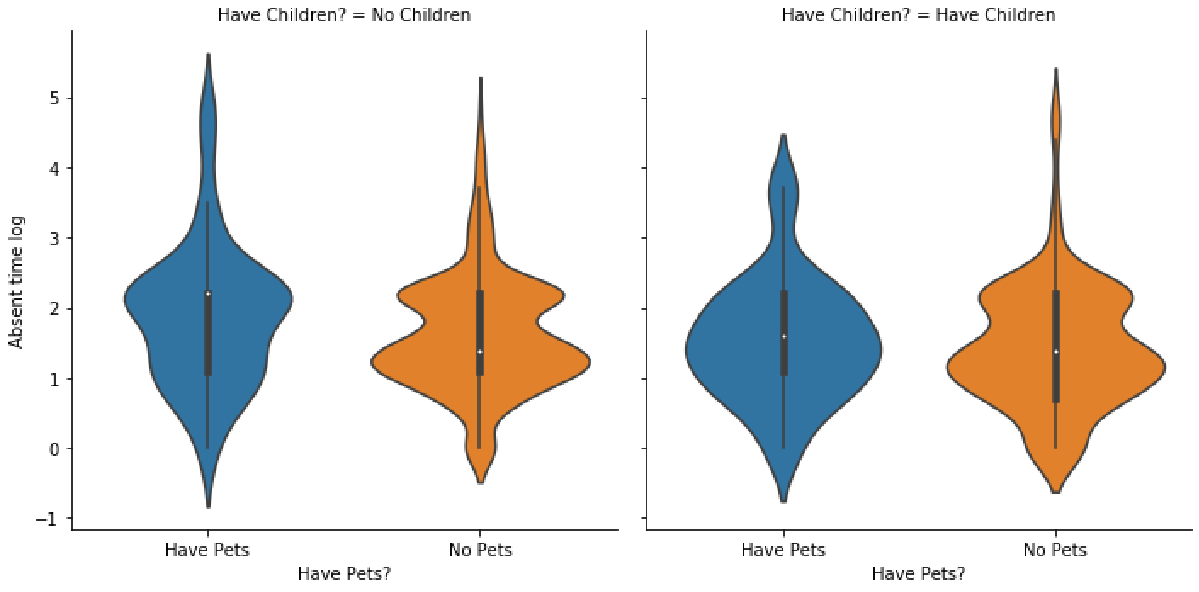
In [38]:

```
plt.figure(figsize=(15, 15))
sns.catplot(y="Absent time log",x="Have Pets?",kind='violin',data=fData
,col='Have Children?')
```

Out[38]:

```
<seaborn.axisgrid.FacetGrid at 0x151382b3b00>
```

```
<Figure size 1080x1080 with 0 Axes>
```



Seems like family is not the reason for Absent time.

Feature Importance Using Tree Based Models

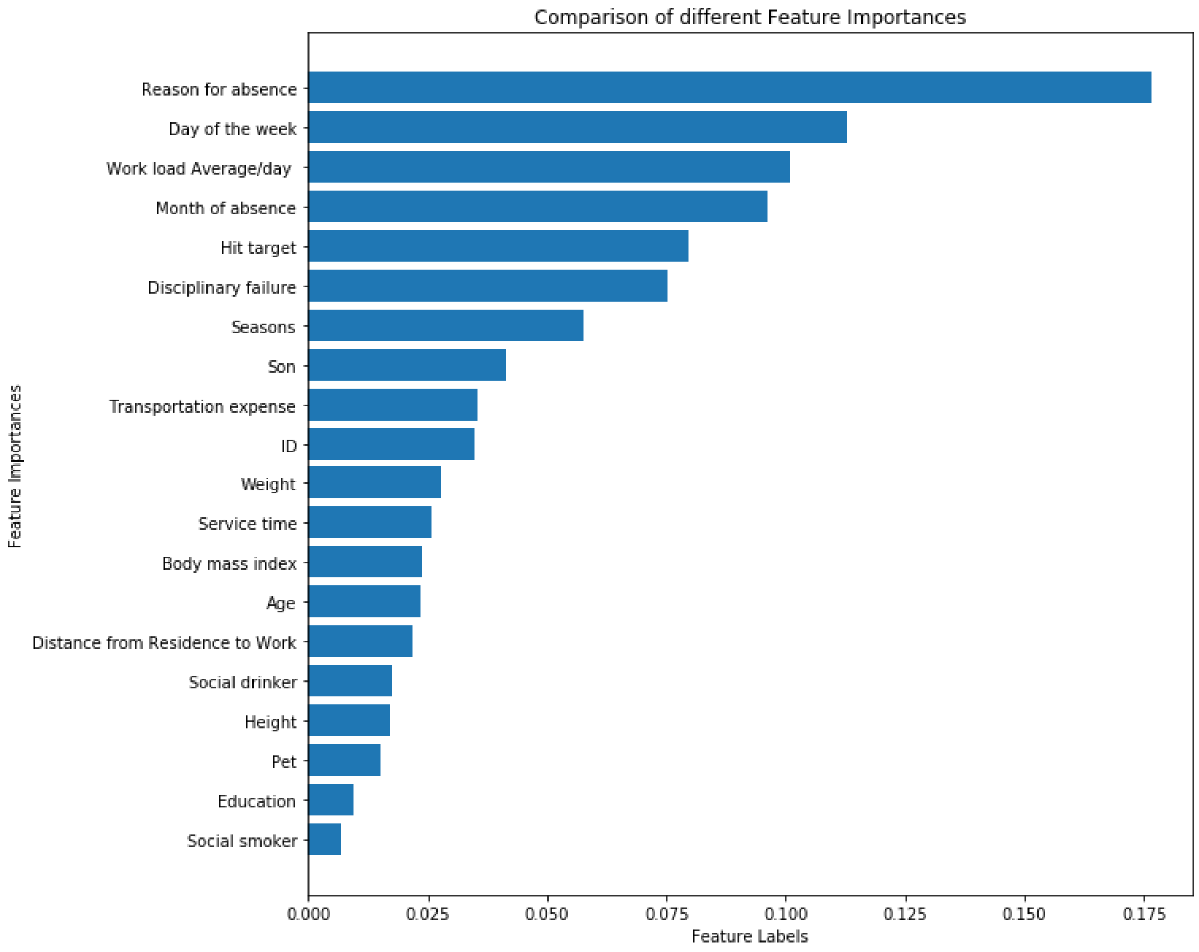
1. Extra Tree Classifier.

In [195]:

```
from sklearn.ensemble import ExtraTreesClassifier

train = fData.drop('Absenteeism time in hours', axis=1)
labels = pd.DataFrame(fData['Absenteeism time in hours'], columns=['Absenteeism time in hours'])

extra_tree_forest = ExtraTreesClassifier(n_estimators = 5, criterion = 'entropy')
extra_tree_forest.fit(train, labels)
feature_importance = extra_tree_forest.feature_importances_
```

```
feats = {}
imp = list(train.columns)
for i in range(len(imp)):
    feats[imp[i]] = feature_importance[i]
imps_s = sorted(feats.items(), key=lambda t : t[1])
x = [str(t[0]) for t in imps_s]
y = [t[1] for t in imps_s]
plt.figure(figsize=(10, 10))
plt.barh(x, y)
plt.xlabel('Feature Labels')
plt.ylabel('Feature Importances')
plt.title('Comparison of different Feature Importances')
plt.show()
```

```
C:\Users\HP\Anaconda3\lib\site-packages\ipykernel_launcher.py:7: DataCo
nversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples,), for example using ra
vel().
  import sys
```



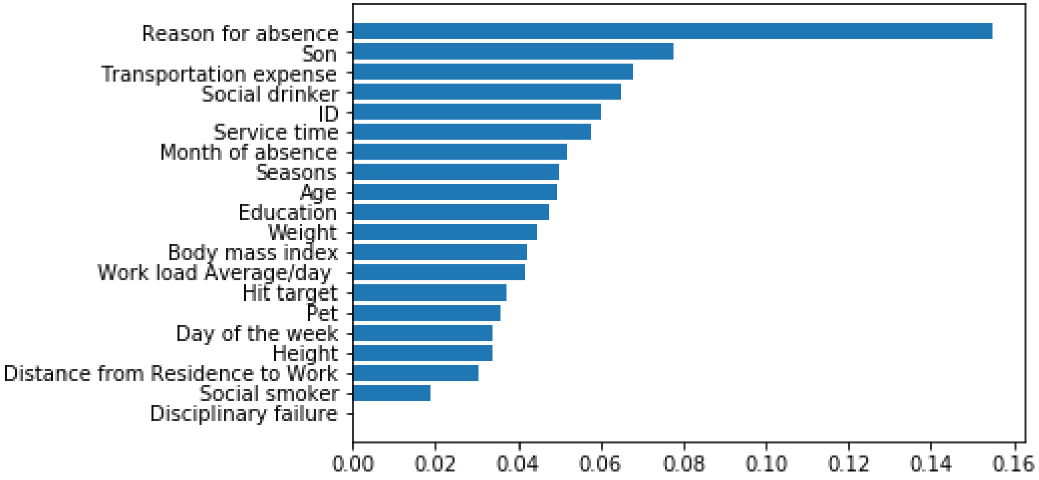
In [192]:

```
from xgboost import XGBClassifier

xgb = XGBClassifier()
xgb.fit(train, labels)

cols = list(train.columns)
imps = xgb.feature_importances_
```

```
col_imp = {}
for i in range(len(cols)):
    col_imp[cols[i]] = imps[i]

col_imp = sorted(col_imp.items(), key=lambda t : t[1])
x = [t[0] for t in col_imp]
y = [t[1] for t in col_imp]

plt.barh(x, y)
```

```
C:\Users\HP\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example us
ing ravel().
  return f(**kwargs)
C:\Users\HP\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example us
ing ravel().
  return f(**kwargs)
```

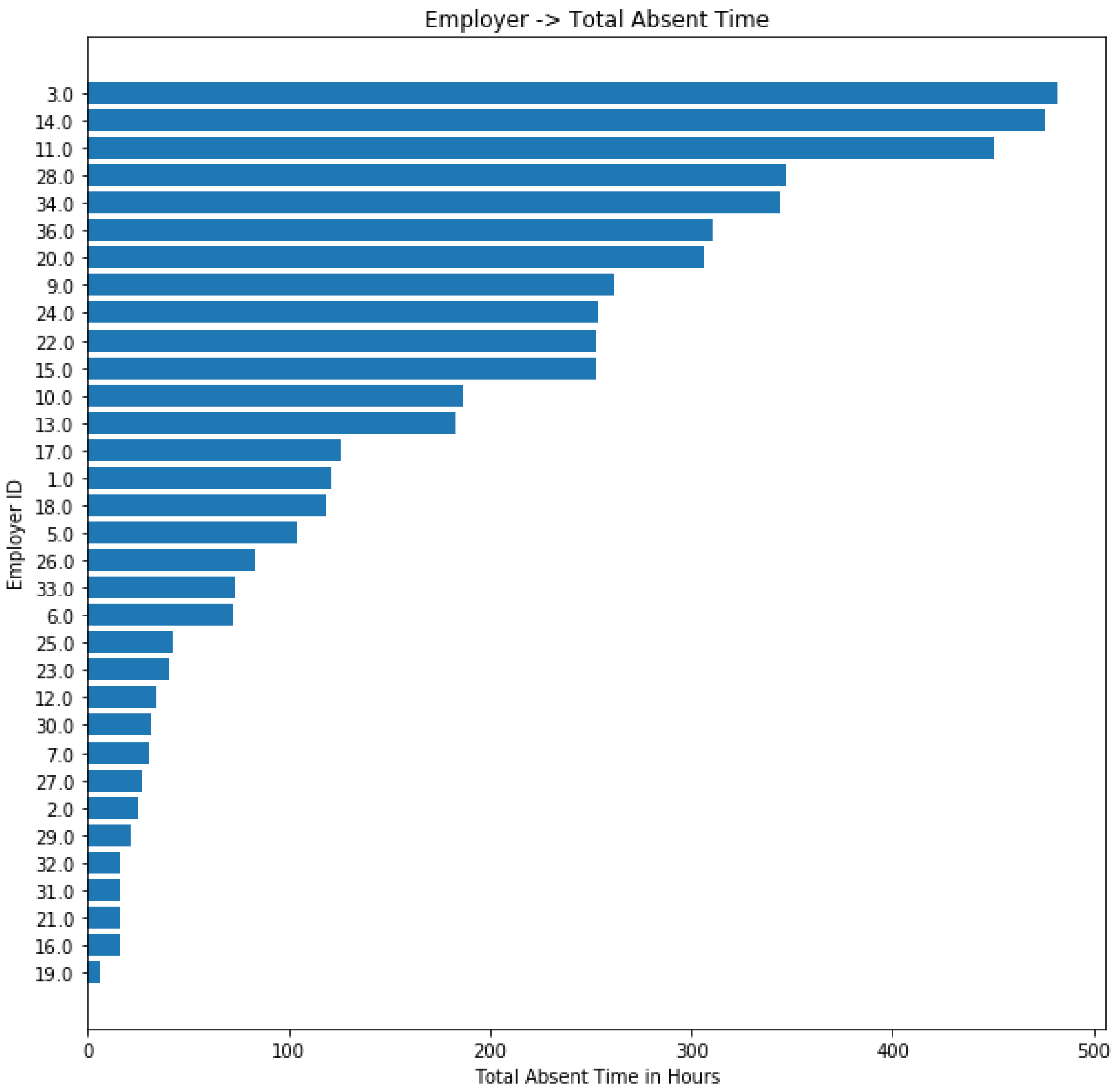
```
Out[192]: <BarContainer object of 20 artists>
```



Employeer total absteesim time (From highest to lowest)

In [39]:

```
ids_abs_time = dict(ffData.groupby('ID')['Absenteeism time in hours'].sum())

s = sorted(ids_abs_time.items(), key=lambda t : t[1])
x = [str(t[0]) for t in s]
y = [t[1] for t in s]
plt.figure(figsize=(10, 10))
plt.barh(x, y)
plt.xlabel('Total Absent Time in Hours')
plt.ylabel('Employer ID')
plt.title('Employer -> Total Absent Time')
plt.show()
```

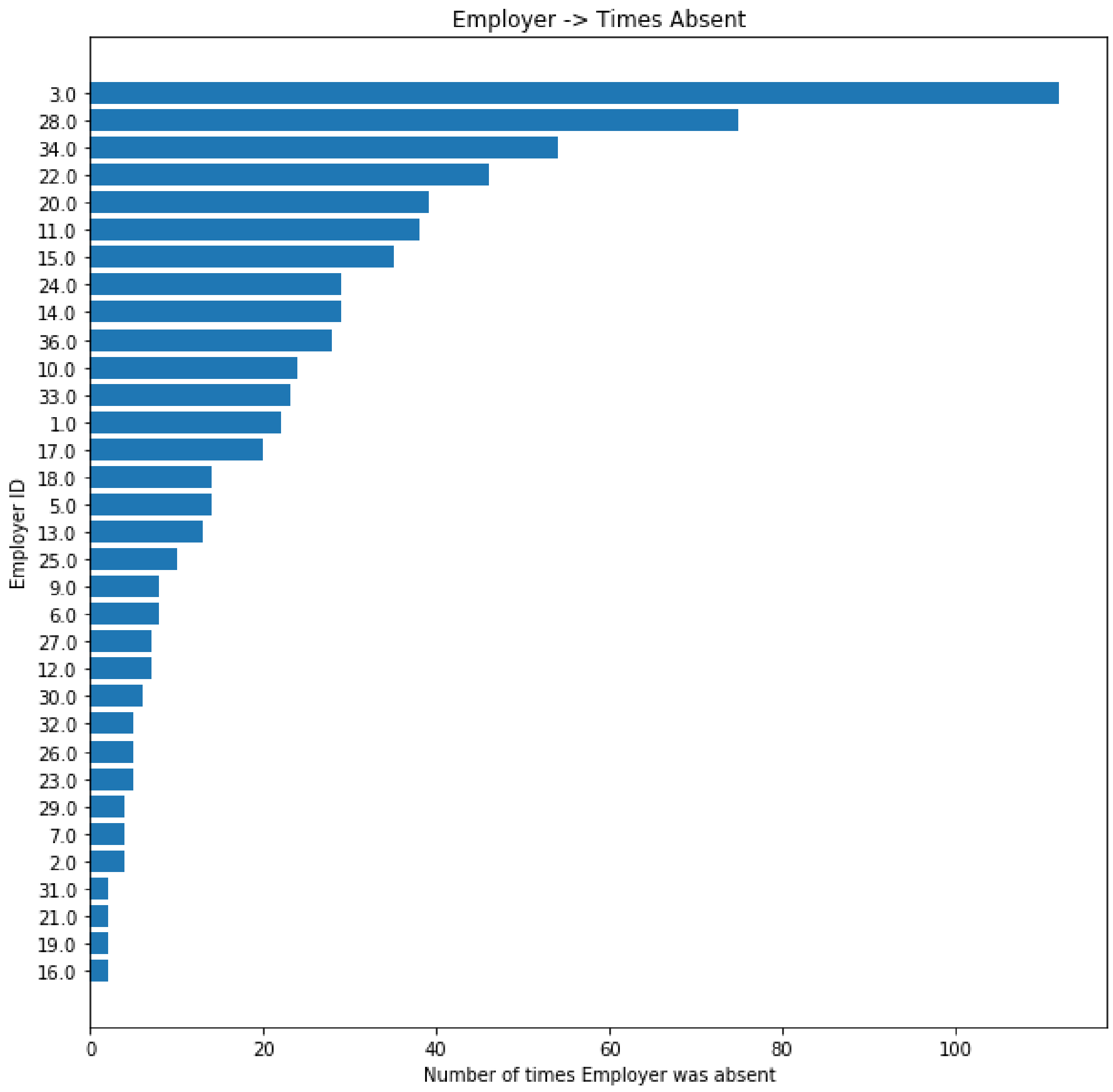


Employeers Absteesim count (From highest to

lowest)

In [40]:

```
ids_abs_time = dict(ffData.groupby('ID')['Absenteeism time in hours'].count())

s = sorted(ids_abs_time.items(), key=lambda t : t[1])
x = [str(t[0]) for t in s]
y = [t[1] for t in s]
plt.figure(figsize=(10, 10))
plt.barh(x, y)
plt.xlabel('Number of times Employer was absent')
plt.ylabel('Employer ID')
plt.title('Employer -> Times Absent')
plt.show()
```



How Seasons affect Transportation cost and

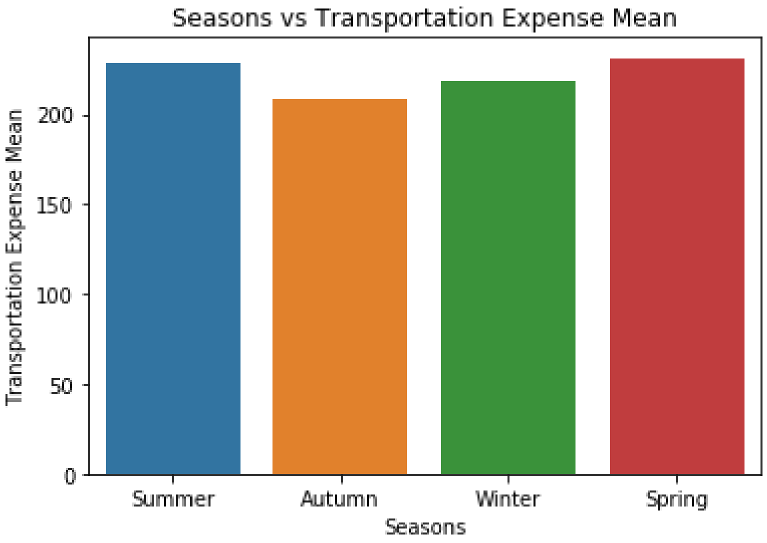
Absenteesim Time?

In [16]:

```
y = list(fData.groupby('Seasons')['Transportation expense'].mean())
x = ['Summer', 'Autumn', 'Winter', 'Spring']
sns.barplot(x, y)
plt.xlabel('Seasons')
plt.ylabel('Transportation Expense Mean')
plt.title('Seasons vs Transportation Expense Mean')
```

Out[16]:

```
Text(0.5, 1.0, 'Seasons vs Transportation Expense Mean')
```

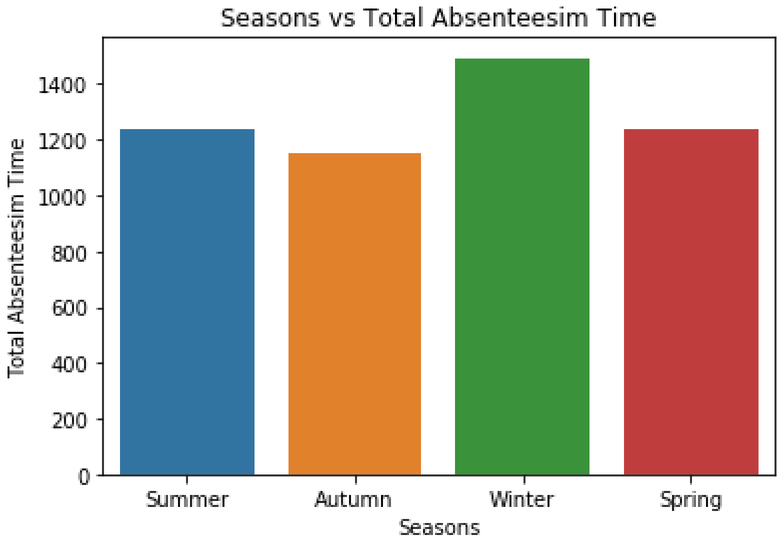


In [68]:

```
y = list(ffData.groupby('Seasons')['Absenteeism time in hours'].sum())
x = ['Summer', 'Autumn', 'Winter', 'Spring']
sns.barplot(x, y)
plt.xlabel('Seasons')
plt.ylabel('Total Absenteesim Time')
plt.title('Seasons vs Total Absenteesim Time')
```

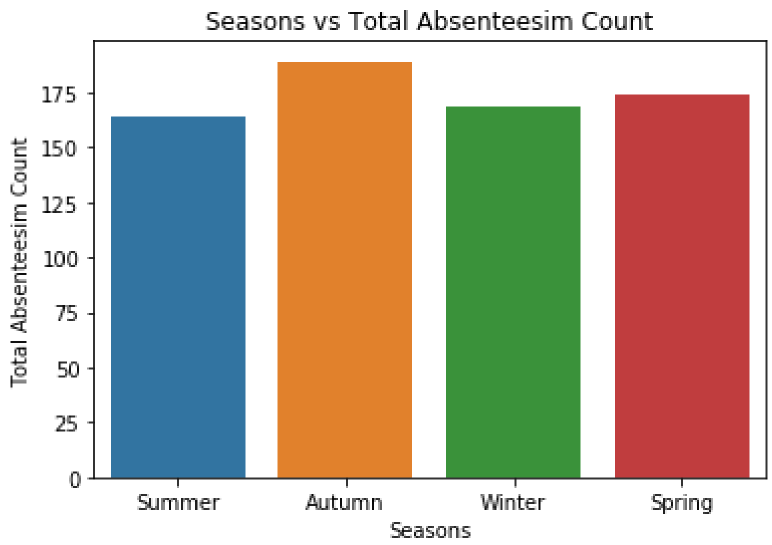
Out[68]:

```
Text(0.5, 1.0, 'Seasons vs Total Absenteesim Time')
```



In [69]:
```
y = list(ffData.groupby('Seasons')['Absenteeism time in hours'].count
())
x = ['Summer', 'Autumn', 'Winter', 'Spring']
sns.barplot(x, y)
plt.xlabel('Seasons')
plt.ylabel('Total Absenteesim Count')
plt.title('Seasons vs Total Absenteesim Count')
```

Out[69]:
```
Text(0.5, 1.0, 'Seasons vs Total Absenteesim Count')
```



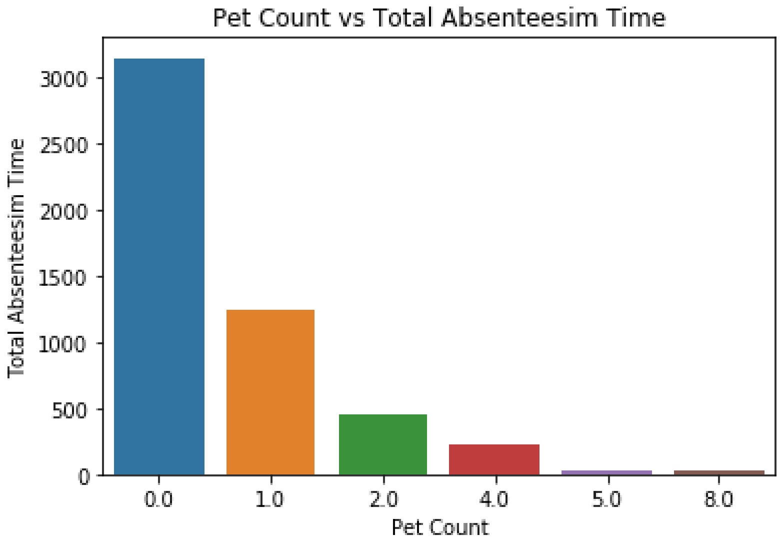
Pet vs Absenteesim Time

In [74]:

```
store = dict(ffData.groupby('Pet')['Absenteeism time in hours'].sum())
x = [k for k in store]
y = [store[k] for k in store]
sns.barplot(x, y)
plt.xlabel('Pet Count')
plt.ylabel('Total Absenteesim Time')
plt.title('Pet Count vs Total Absenteesim Time')
```

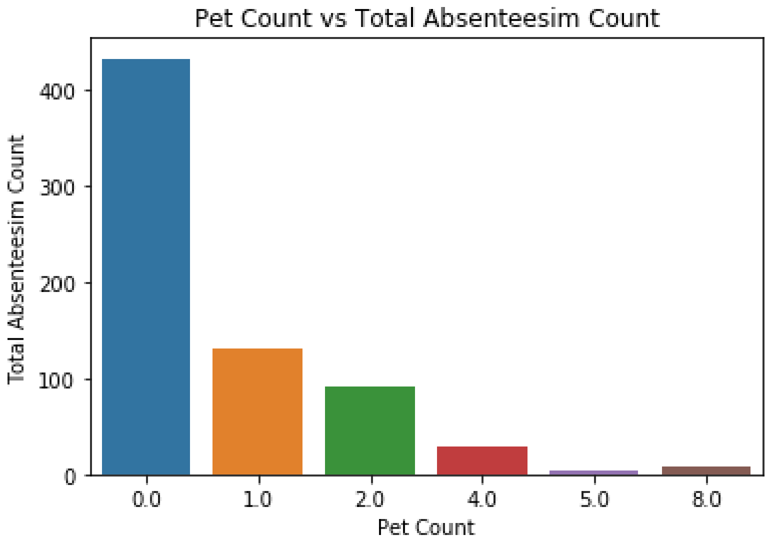
Out[74]:

```
Text(0.5, 1.0, 'Pet Count vs Total Absenteesim Time')
```



In [75]:
```
store = dict(ffData.groupby('Pet')['Absenteeism time in hours'].count
())
x = [k for k in store]
y = [store[k] for k in store]
sns.barplot(x, y)
plt.xlabel('Pet Count')
plt.ylabel('Total Absenteesim Count')
plt.title('Pet Count vs Total Absenteesim Count')
```

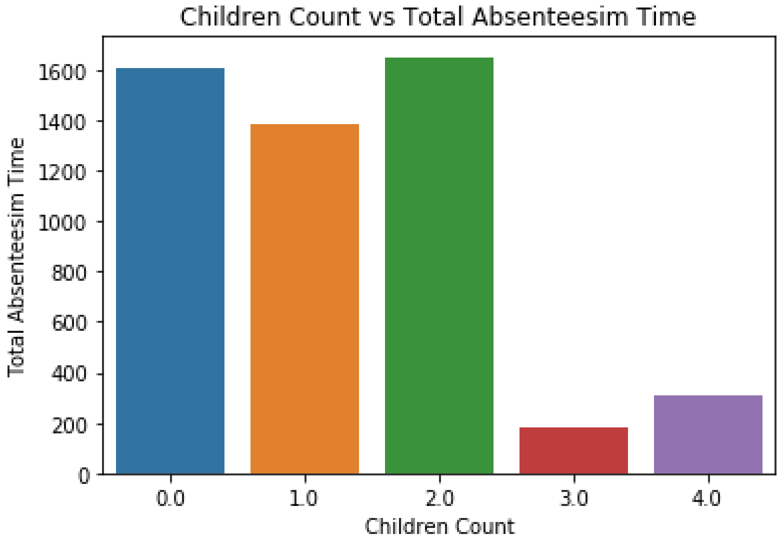
Out[75]:
```
Text(0.5, 1.0, 'Pet Count vs Total Absenteesim Count')
```



Son vs Abseentism Time

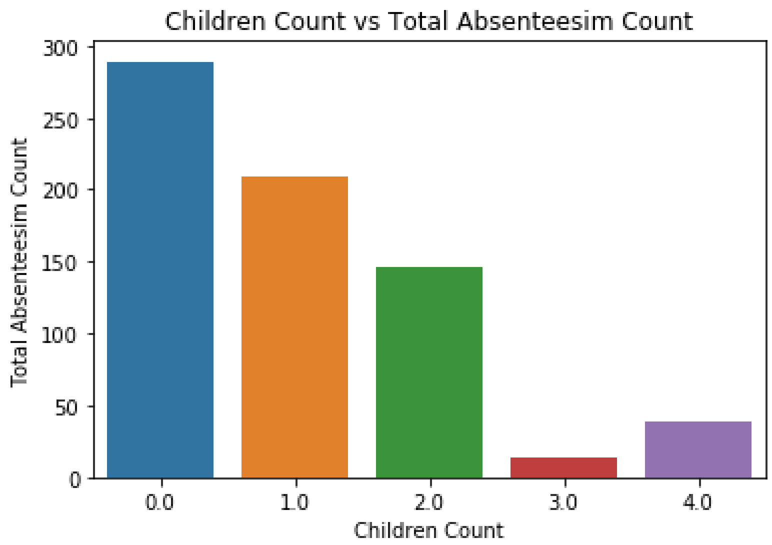
```
In [77]: store = dict(ffData.groupby('Son')['Absenteeism time in hours'].sum())
         x = [k for k in store]
         y = [store[k] for k in store]
         sns.barplot(x, y)
         plt.xlabel('Children Count')
         plt.ylabel('Total Absenteesim Time')
         plt.title('Children Count vs Total Absenteesim Time')
```

```
Out[77]: Text(0.5, 1.0, 'Children Count vs Total Absenteesim Time')
```



In [78]:
```
store = dict(ffData.groupby('Son')['Absenteeism time in hours'].count
())
x = [k for k in store]
y = [store[k] for k in store]
sns.barplot(x, y)
plt.xlabel('Children Count')
plt.ylabel('Total Absenteesim Count')
plt.title('Children Count vs Total Absenteesim Count')
```

Out[78]:
```
Text(0.5, 1.0, 'Children Count vs Total Absenteesim Count')
```



```
In [89]: !pip install nbconvert
```

```
Requirement already satisfied: nbconvert in c:\users\hp\anaconda3\lib\s
ite-packages (5.4.1)
Requirement already satisfied: mistune>=0.8.1 in c:\users\hp\anaconda3
\lib\site-packages (from nbconvert) (0.8.4)
Requirement already satisfied: jinja2 in c:\users\hp\anaconda3\lib\site
-packages (from nbconvert) (2.10)
Requirement already satisfied: pygments in c:\users\hp\anaconda3\lib\si
te-packages (from nbconvert) (2.3.1)
Requirement already satisfied: traitlets>=4.2 in c:\users\hp\anaconda3
\lib\site-packages (from nbconvert) (4.3.2)
Requirement already satisfied: jupyter_core in c:\users\hp\anaconda3\li
b\site-packages (from nbconvert) (4.4.0)
Requirement already satisfied: nbformat>=4.4 in c:\users\hp\anaconda3\l
ib\site-packages (from nbconvert) (4.4.0)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\hp\anacon
da3\lib\site-packages (from nbconvert) (0.3)
Requirement already satisfied: bleach in c:\users\hp\anaconda3\lib\site
-packages (from nbconvert) (3.1.0)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\hp\anac
```

```
onda3\lib\site-packages (from nbconvert) (1.4.2)
Requirement already satisfied: testpath in c:\users\hp\anaconda3\lib\si
te-packages (from nbconvert) (0.4.2)
Requirement already satisfied: defusedxml in c:\users\hp\anaconda3\lib
\site-packages (from nbconvert) (0.5.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\hp\anaconda
3\lib\site-packages (from jinja2->nbconvert) (1.1.1)
Requirement already satisfied: ipython-genutils in c:\users\hp\anaconda
3\lib\site-packages (from traitlets>=4.2->nbconvert) (0.2.0)
Requirement already satisfied: decorator in c:\users\hp\anaconda3\lib\s
ite-packages (from traitlets>=4.2->nbconvert) (4.4.0)
Requirement already satisfied: six in c:\users\hp\anaconda3\lib\site-pa
ckages (from traitlets>=4.2->nbconvert) (1.12.0)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in c:\users\hp\a
naconda3\lib\site-packages (from nbformat>=4.4->nbconvert) (3.0.1)
Requirement already satisfied: webencodings in c:\users\hp\anaconda3\li
b\site-packages (from bleach->nbconvert) (0.5.1)
Requirement already satisfied: attrs>=17.4.0 in c:\users\hp\anaconda3\l
ib\site-packages (from jsonschema!=2.5.0,>=2.4->nbformat>=4.4->nbconver
t) (19.1.0)
Requirement already satisfied: pyrsistent>=0.14.0 in c:\users\hp\anacon
da3\lib\site-packages (from jsonschema!=2.5.0,>=2.4->nbformat>=4.4->nbc
onvert) (0.14.11)
Requirement already satisfied: setuptools in c:\users\hp\anaconda3\lib
\site-packages (from jsonschema!=2.5.0,>=2.4->nbformat>=4.4->nbconvert)
(40.8.0)
```

In [91]:

```
!pip install pandoc
```

```
Collecting pandoc
  Downloading https://files.pythonhosted.org/packages/49/b1/d2d4b30ee81
ea5cb7aee5ba3591752a637fdc49d0a42fa9683874b60b9fb/pandoc-1.0.2.tar.gz
(488kB)
Requirement already satisfied: ply in c:\users\hp\anaconda3\lib\site-pa
ckages (from pandoc) (3.11)
Building wheels for collected packages: pandoc
  Building wheel for pandoc (setup.py): started
  Building wheel for pandoc (setup.py): finished with status 'done'
  Stored in directory: C:\Users\HP\AppData\Local\pip\Cache\wheels\d1\e8
```

```
\71\bc3242b3e8f119c62eebdb0dee519fd40ac293e4835839db7c
Successfully built pandoc
Installing collected packages: pandoc
Successfully installed pandoc-1.0.2
```

In [93]:

```
!pip install texlive-xetex
```

```
Collecting texlive-xetex
```

```
  ERROR: Could not find a version that satisfies the requirement texliv
e-xetex (from versions: none)
ERROR: No matching distribution found for texlive-xetex
```

In [ ]: