Answers to Dr Fuller's Recruitment Quiz - Submitted by Sumayyah Musa

In [1]:
```
#import important libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import statsmodels.api as sm
import seaborn as sns
sns.set()
```

Exploratory Data Analysis

In [2]:
```
#Read in the data
data = pd.read_csv('cchs.csv', index_col = 0)
data.head() #to show the first five rows of the data
```

Out[2]:

	CASEID	height_m	weight_kg
1	1	1.575	65.25
2	2	1.905	99.00
3	3	1.803	77.40
4	4	1.727	85.50
5	5	1.803	81.00

In [3]:
```
data.shape #number of rows and columns in the data
```

Out[3]: (5000, 3)

In [4]:
```
df = data.copy() #to produce a copy of the data before making any permanent ch
ange
df.head()
```

Out[4]:

	CASEID	height_m	weight_kg
1	1	1.575	65.25
2	2	1.905	99.00
3	3	1.803	77.40
4	4	1.727	85.50
5	5	1.803	81.00

In [5]:
```
df.drop('CASEID',axis=1, inplace=True) #to drop unnecessary columns such as CASEID
df.head()
```

Out[5]:

	height_m	weight_kg
1	1.575	65.25
2	1.905	99.00
3	1.803	77.40
4	1.727	85.50
5	1.803	81.00

In [6]:
```
df.info() #to view information about the data including missing values
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5000 entries, 1 to 5000
Data columns (total 2 columns):
height_m     4805 non-null float64
weight_kg    4710 non-null float64
dtypes: float64(2)
memory usage: 117.2 KB
```

In [7]:
```
df.isnull().sum(axis = 0) #to also check for missing values
```

Out[7]:
```
height_m     195
weight_kg    290
dtype: int64
```

In [8]:
```
df.head(10)
```

Out[8]:

	height_m	weight_kg
1	1.575	65.25
2	1.905	99.00
3	1.803	77.40
4	1.727	85.50
5	1.803	81.00
6	1.727	78.75
7	1.626	62.10
8	NaN	NaN
9	1.905	105.75
10	1.854	85.50

In [9]:
```
#To permanently drop all rows with missing values
df.dropna(inplace=True)
print(df.shape)
df.isnull().sum(axis = 0)
```

```
(4681, 2)
```

Out[9]:
```
height_m     0
weight_kg    0
dtype: int64
```

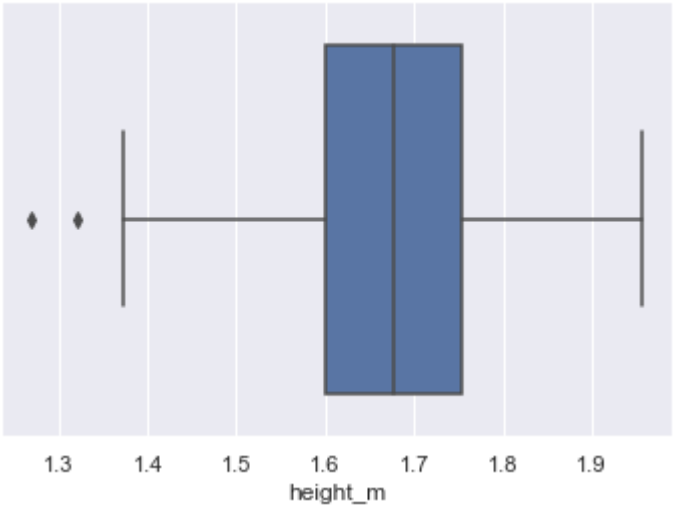
In [10]:
```
df.head(10)
```

Out[10]:

	height_m	weight_kg
1	1.575	65.25
2	1.905	99.00
3	1.803	77.40
4	1.727	85.50
5	1.803	81.00
6	1.727	78.75
7	1.626	62.10
9	1.905	105.75
10	1.854	85.50
11	1.651	61.20

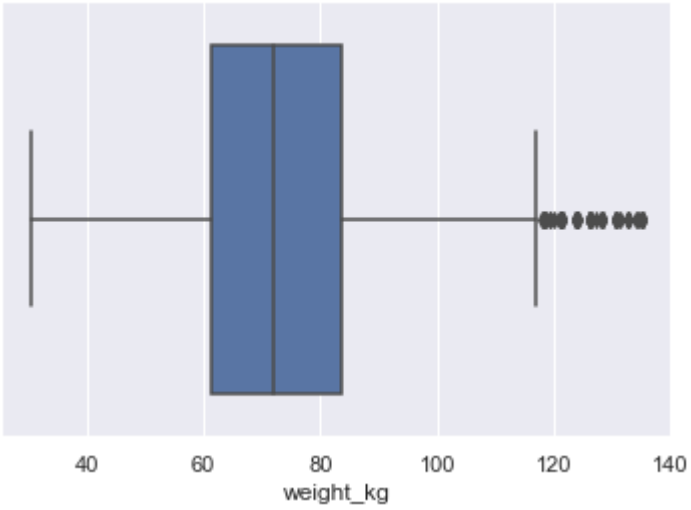
In [11]:
```
#to check for outliers in each column of the dataset
sns.boxplot(x = df['height_m'])
```

Out[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x23a299dda20>
```



In [12]:
```
sns.boxplot(x = df['weight_kg'])
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x23a293f9080>



In [13]:
```
#using Z score to show outliers
from scipy import stats
z = np.abs(stats.zscore(df))
print(z)

threshold = 3
print(np.where(z > 3))
```

```
[[1.08615205 0.51736149]
 [2.18397063 1.43287138]
 [1.17320544 0.18472234]
 ...
 [0.92546887 1.42747017]
 [0.42008628 0.77739254]
 [1.43085147 0.25733044]]
(array([  13,   61,   73,  218,  331,  523,  794, 1056, 1073, 1361, 1399,
       1489, 1570, 1660, 1712, 1920, 1955, 1982, 2419, 2470, 2506, 2611,
       2657, 2734, 2843, 2933, 3025, 3054, 3091, 3259, 3664, 3713, 3742,
       3828, 4042], dtype=int64), array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1], dtype=int64))
```

In [14]:
```
print(z[13][1])
```

```
3.513119785086425
```

In [15]:
```
#using z score to remove outliers
df_new = df[(z < 3).all(axis=1)]
```

In [16]:
```
df.shape
```

Out[16]: (4681, 2)

In [17]:
```
df_new.shape
```

Out[17]: (4646, 2)

In [18]:
```
df_new.describe() #to describe the statistical attributes of the data
```

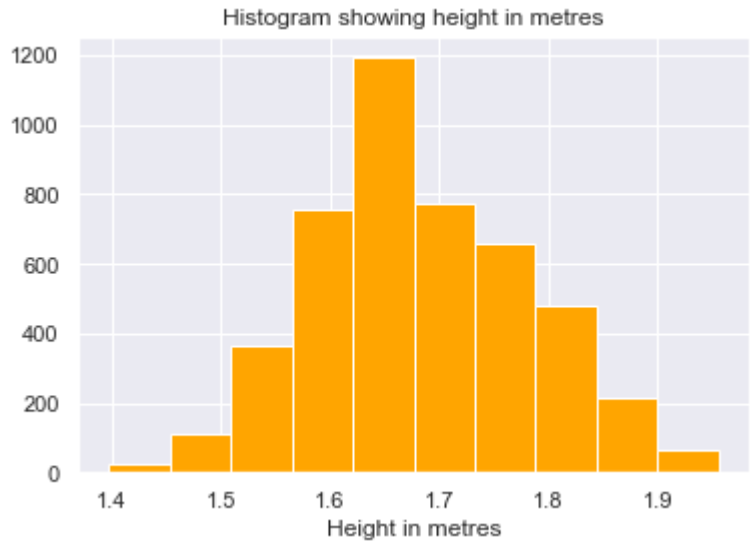
Out[18]:

	height_m	weight_kg
count	4646.000000	4646.000000
mean	1.684204	73.832475
std	0.100346	16.676402
min	1.397000	30.600000
25%	1.600000	60.770000
50%	1.676000	72.000000
75%	1.753000	83.250000
max	1.956000	126.000000

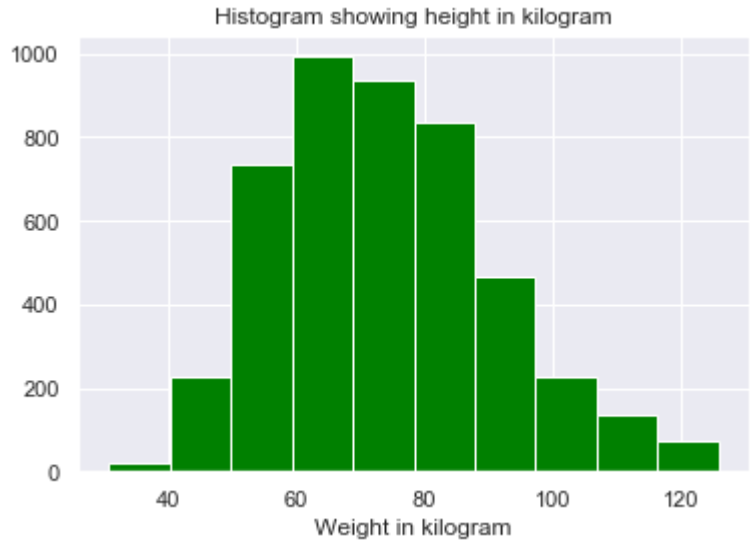
The above table shows that the mean height with standard deviation is 1.68+/-0.10m, and the mean weight is 73.83+/-16.68kg

Data Visualization - Histogram of the Height and Weight

In [19]:
```
#histogram using matplotlib
height = df_new['height_m']
plt.hist(x = height, color = 'orange', bins = 10)
plt.xlabel("Height in metres")
plt.title('Histogram showing height in metres')
plt.show()
```



In [20]:
```
#Histogram of weight using matplotlib
weight = df_new['weight_kg']
plt.hist(x = weight, color = 'green', bins = 10)
plt.xlabel("Weight in kilogram")
plt.title('Histogram showing height in kilogram')
plt.show()
```



Data Visualization - Scatter Plot

In [21]:
```
#Scatterplot using matplotlib
x = df_new['weight_kg']
y = df_new['height_m']
plt.scatter(x, y, c = 'purple')
plt.xlabel('Weight in kilogram')
plt.ylabel('Height in metres')
plt.title('Scatterplot of Weight and Height')
plt.show()
```

The above scatterplot shows overplotting due to the too much data points (4646) being plotted.

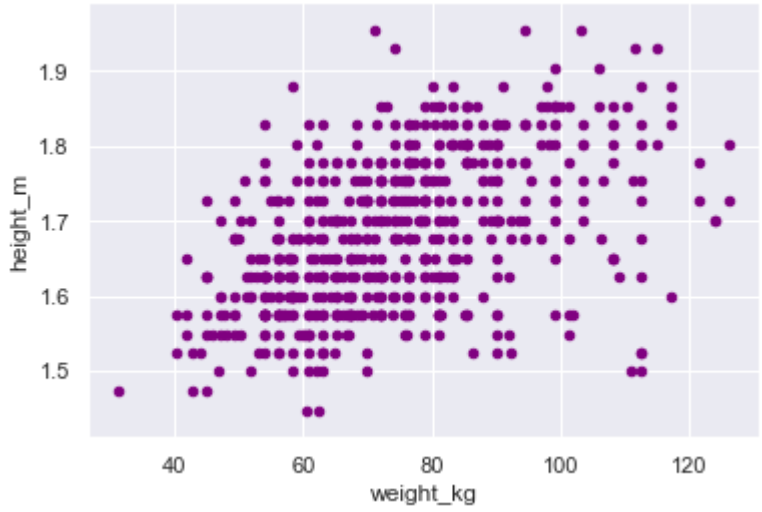
In [22]:
```
#To prevent overplotting, plot a subset of the data
data_plot = df_new[:600]
data_plot.head()
```

Out[22]:

	height_m	weight_kg
1	1.575	65.25
2	1.905	99.00
3	1.803	77.40
4	1.727	85.50
5	1.803	81.00

In [23]:
```
#Scatterplot of the subsetted data using pandas
data_plot.plot(x = 'weight_kg', y = 'height_m', kind = 'scatter', color = 'pur
ple')
```

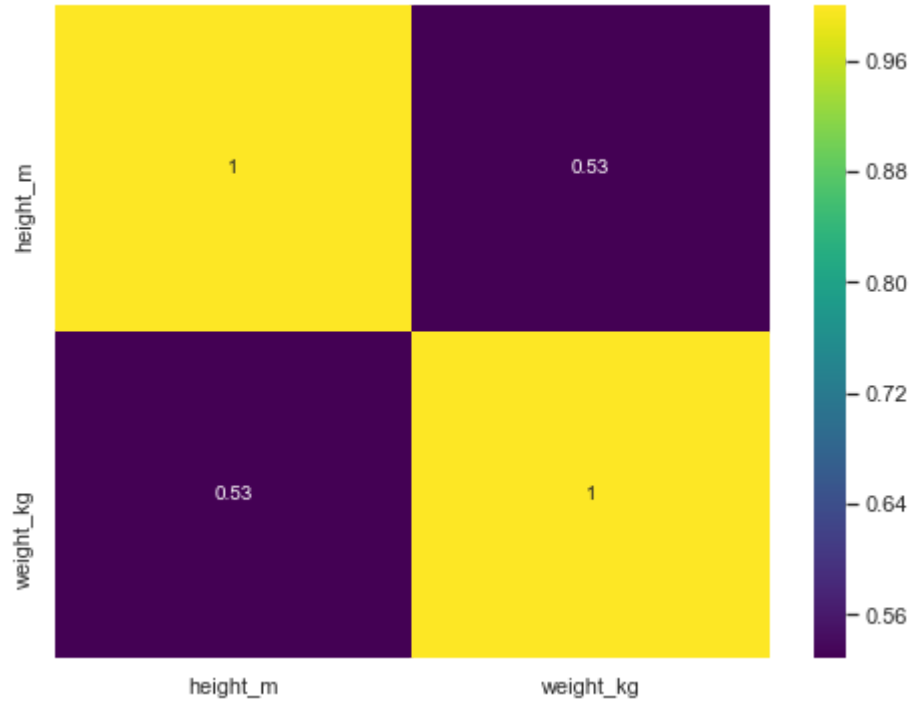
Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x23a295b7780>



Data Visualization - Correlation Matrix

In [24]:
```
#check for correlation and relationship between height and weight using correl
ation matrix
plt.figure(figsize=(8,6))
sns.heatmap(df_new.corr(),cmap='viridis',annot=True)
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x23a29611e10>



The scatterplot and correlation matrix show that there is a fair (neither strong nor weak) positive correlation between height and weight, with a correlation coefficient of 0.53. In other words, as weight increases fairly, so does height. To further examine this trend, an ordinary least squares regression need to be performed on the height and weight. Note that, there is no causal relationship between height and weight. The increase in height does not necessarily cause the increase in weight, or vice-versa.