```
from io import IncrementalNewlineDecoder
##import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from matplotlib import style
```

```
## import the data
diabetes= pd.read_csv("/content/diabetes dataset.csv")
```

```
diabetes
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

Next steps: Generate code with diabetes | View recommended plots | New interactive sheet

```
diabetes.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Next steps: Generate code with diabetes | View recommended plots | New interactive sheet

```
##  columnname
diabetes.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
## count of outcome column
diabetes.groupby('Outcome').size()
```

Outcome	0
0	500
1	268

dtype: int64

```
##checking null value
diabetes.isnull().any()
##info
diabetes.info()
##glucose
diabetes['Glucose'].value_counts().head(10)
diabetes['Glucose']
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

	Glucose
0	148
1	85
2	183
3	89
4	137
...	...
763	101
764	122
765	121
766	126
767	93

768 rows × 1 columns

dtype: int64

```
##bloodpressure
diabetes['BloodPressure'].value_counts().head(10)
```

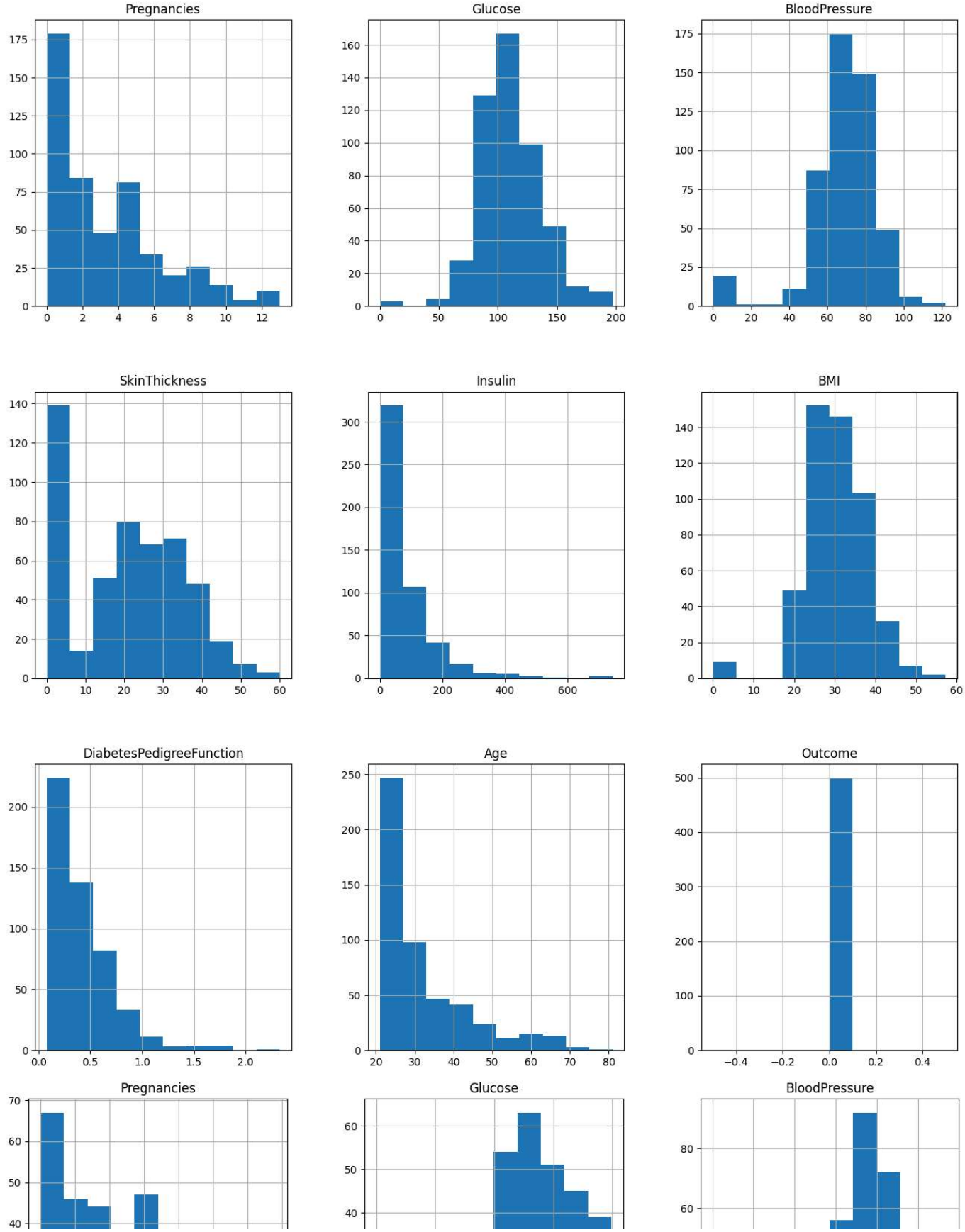
BloodPressure	count
70	57
74	52
78	45
68	45
72	44
64	43
80	40
76	39
60	37
0	35

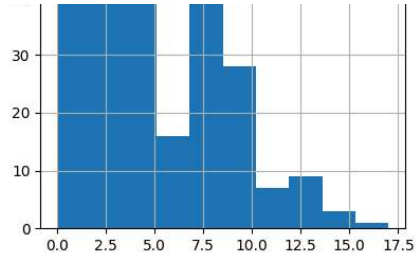
dtype: int64

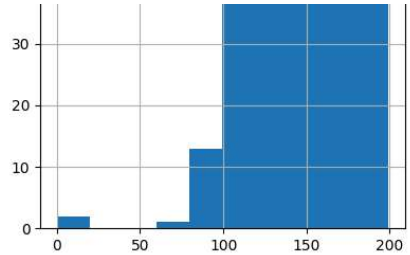
```
## the function will draw histogram by data column nameand title
def plot_histogram(data_val,title_name):
    plt.figure(figsize=[10,6])
    plt.hist(data_val,edgecolor="green")
    #plt.grid(axis='y', alpha=0.75)
    plt.title(title_name,fontsize=15)
    plt.show()
diabetes.groupby('Outcome').hist(figsize=(16, 18))
```

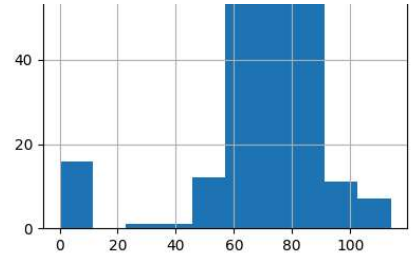
Outcome	0
0	[[Axes(0.125,0.666111;0.215278x0.213889), Axes...
1	[[Axes(0.125,0.666111;0.215278x0.213889), Axes...

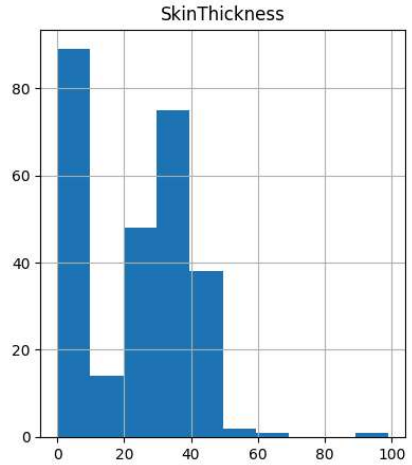
dtype: object

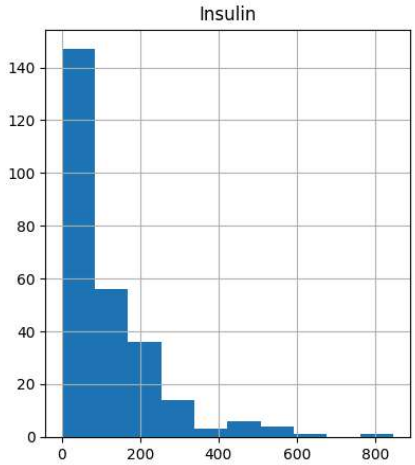


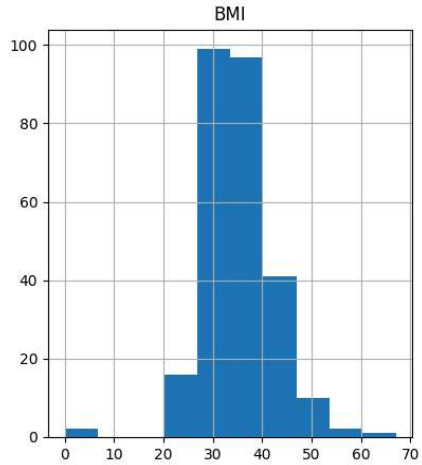

0
10
20
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

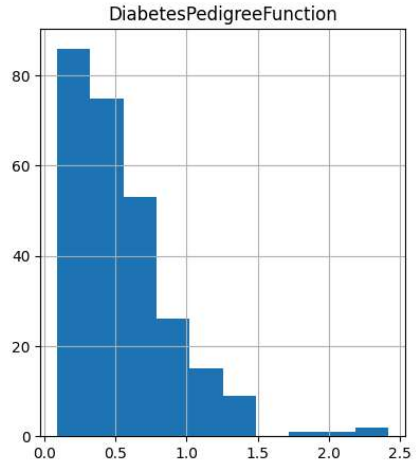

0
10
20
30
0
50
100
150
200

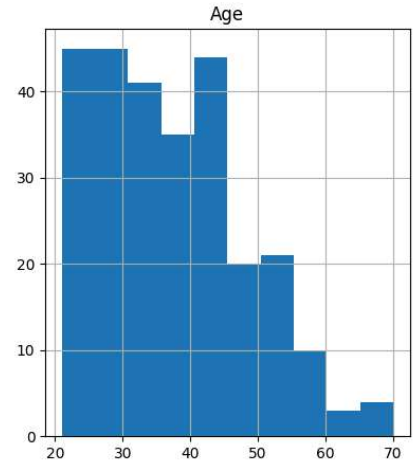

0
20
40
0
20
40
60
80
100

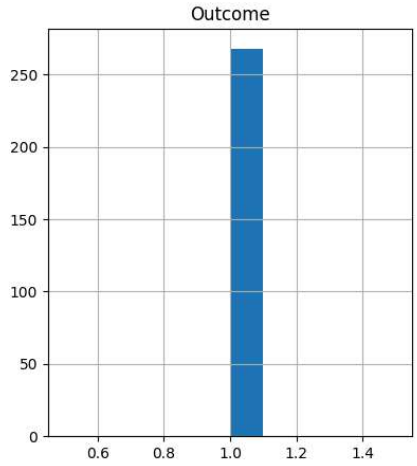

SkinThickness
0
20
40
60
80
0
20
40
60
80
100


Insulin
0
20
40
60
80
100
120
140
0
200
400
600
800


BMI
0
20
40
60
80
100
0
10
20
30
40
50
60
70


DiabetesPedigreeFunction
0
20
40
60
80
0.0
0.5
1.0
1.5
2.0
2.5


Age
0
10
20
30
40
20
30
40
50
60
70


Outcome
0
50
100
150
200
250
0.6
0.8
1.0
1.2
1.4

```
#function to get total count of zeros and outcome details together
def get_zeros_outcome_count(data,column_name):
    count = data[data[column_name] == 0].shape[0]
    print("Total No of zeros found in " + column_name + " : " + str(count))
    print(data[data[column_name] == 0].groupby('Outcome')['Age'].count())
#Checking count of zeros in blood pressure
get_zeros_outcome_count(diabetes,'BloodPressure')
```

```
Total No of zeros found in BloodPressure : 35
Outcome
0    19
1    16
Name: Age, dtype: int64
```

```
##checking count of zeros in glucose
get_zeros_outcome_count(diabetes,'Glucose')
##checking count of zeros in skinthickness
get_zeros_outcome_count(diabetes,'SkinThickness')
##checking count of zeros in BMI
get_zeros_outcome_count(diabetes,'BMI')
##checking count of zeros in insulin
get_zeros_outcome_count(diabetes,'Insulin')
```

```
Total No of zeros found in Glucose : 5
Outcome
0    3
1    2
Name: Age, dtype: int64
Total No of zeros found in SkinThickness : 227
Outcome
0    139
1     88
Name: Age, dtype: int64
Total No of zeros found in BMI : 11
Outcome
0    9
1    2
Name: Age, dtype: int64
Total No of zeros found in Insulin : 374
Outcome
0    236
1    138
Name: Age, dtype: int64
```

```
diabetes_mod = diabetes[(diabetes.BloodPressure != 0) & (diabetes.BMI != 0) & (diabetes.Glucose != 0)]
print(diabetes_mod.shape)
## the stats of data after removing bloodpressure,bmi,glucose 0 rows
diabetes_mod.describe().transpose()
```

```
(724, 9)
```

	count	mean	std	min	25%	50%	75%	max
Pregnancies	724.0	3.866022	3.362803	0.000	1.000	3.000	6.0000	17.00
Glucose	724.0	121.882597	30.750030	44.000	99.750	117.000	142.0000	199.00
BloodPressure	724.0	72.400552	12.379870	24.000	64.000	72.000	80.0000	122.00
SkinThickness	724.0	21.443370	15.732756	0.000	0.000	24.000	33.0000	99.00
Insulin	724.0	84.494475	117.016513	0.000	0.000	48.000	130.5000	846.00
BMI	724.0	32.467127	6.888941	18.200	27.500	32.400	36.6000	67.10
DiabetesPedigreeFunction	724.0	0.474765	0.332315	0.078	0.245	0.379	0.6275	2.42
Age	724.0	33.350829	11.765393	21.000	24.000	29.000	41.0000	81.00
Outcome	724.0	0.343923	0.475344	0.000	0.000	0.000	1.0000	1.00

```
#Lets create positive variable and store all 1 value Outcome data
Positive = diabetes_mod[diabetes_mod['Outcome']==1]
Positive.head(5)
```

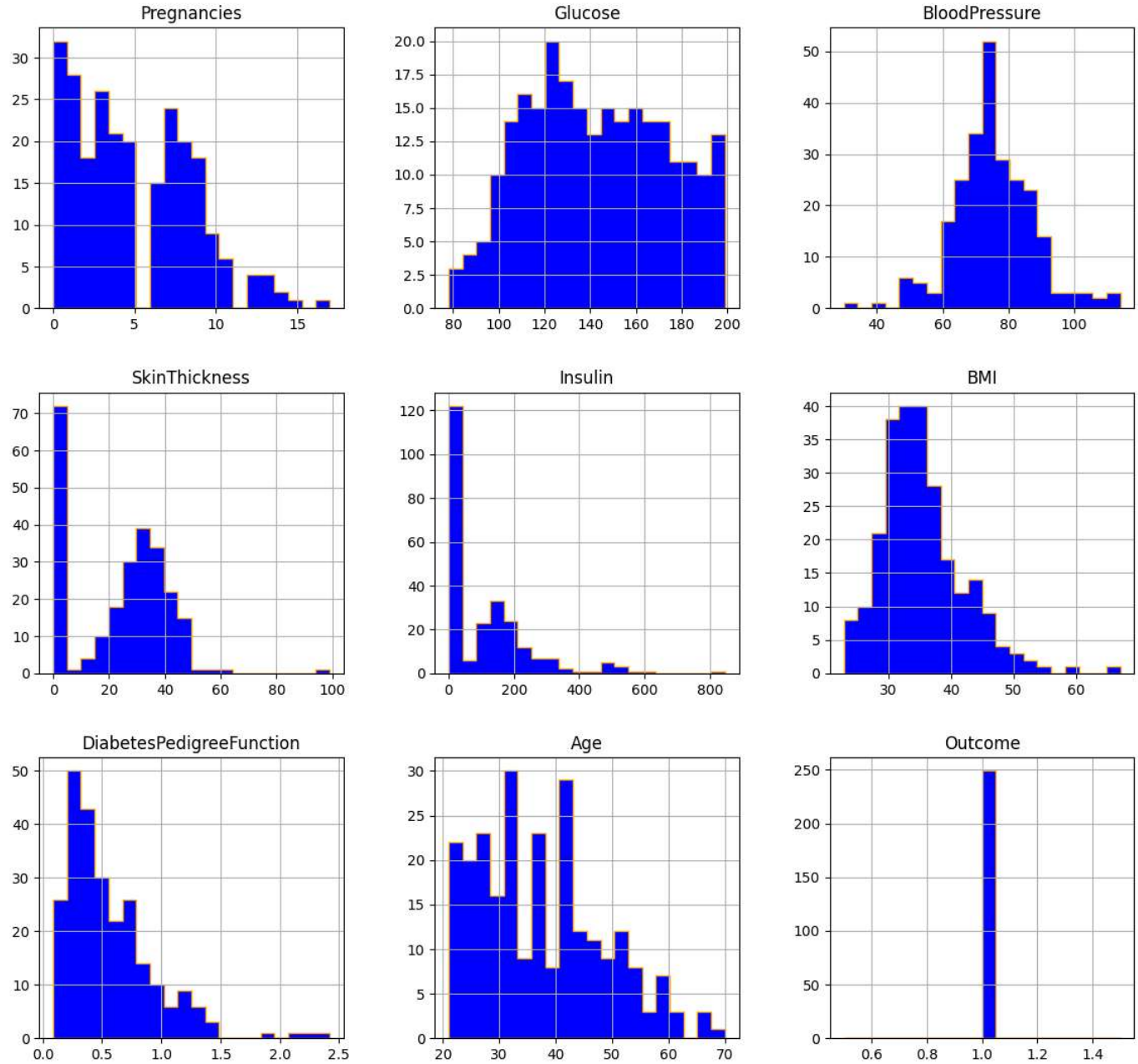
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
2	8	183	64	0	0	23.3	0.672	32	1
4	0	137	40	35	168	43.1	2.288	33	1
6	3	78	50	32	88	31.0	0.248	26	1
8	2	197	70	45	543	30.5	0.158	53	1

Next steps: Generate code with Positive | View recommended plots | New interactive sheet

```
Positive.groupby('Outcome').hist(figsize=(14, 13),histtype='stepfilled',bins=20,color="blue",edgecolor="orange")
```

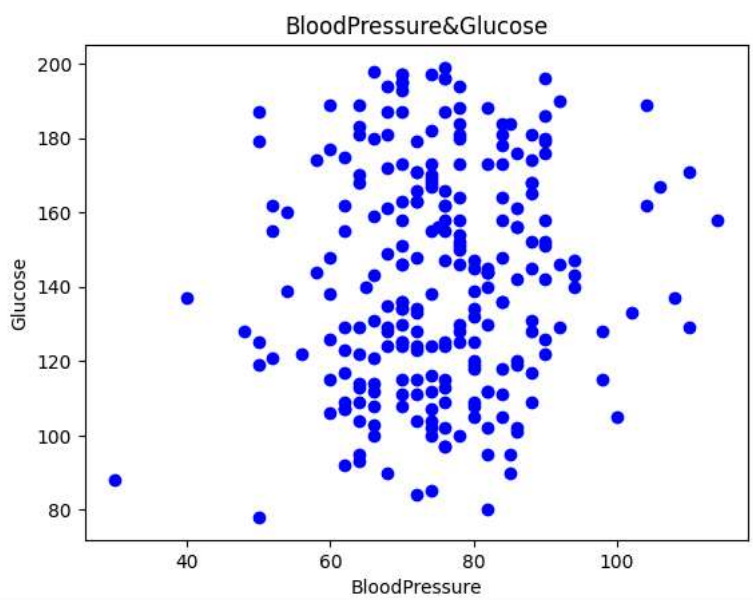
Outcome	0
1	[[Axes(0.125,0.666111;0.215278x0.213889), Axes...

dtype: object

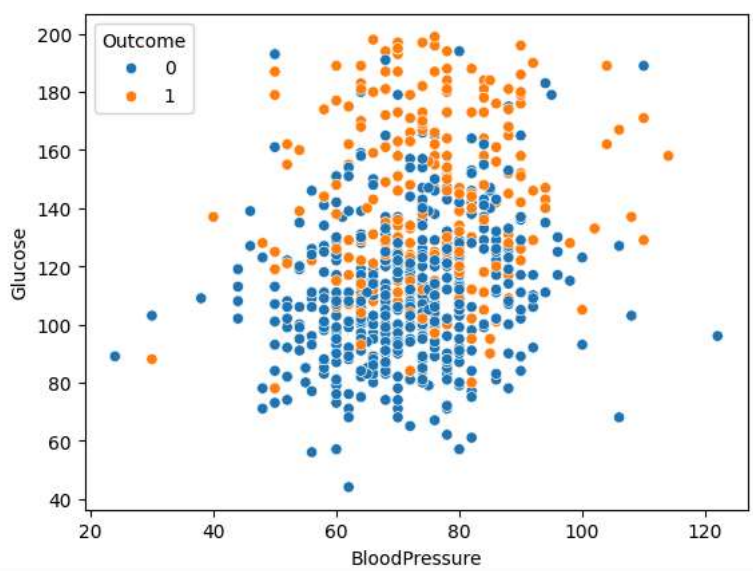



```
#function to create scatter plot
def create_scatter_plot(first_value,second_value,x_label,y_label,colour):
    plt.scatter(first_value,second_value, color=[colour])
    plt.xlabel(x_label)
    plt.ylabel(y_label)
    title_name = x_label + '&' + y_label
    plt.title(title_name)
    plt.show()
BloodPressure = Positive['BloodPressure']
Glucose = Positive['Glucose']
SkinThickness = Positive['SkinThickness']
Insulin = Positive['Insulin']
```

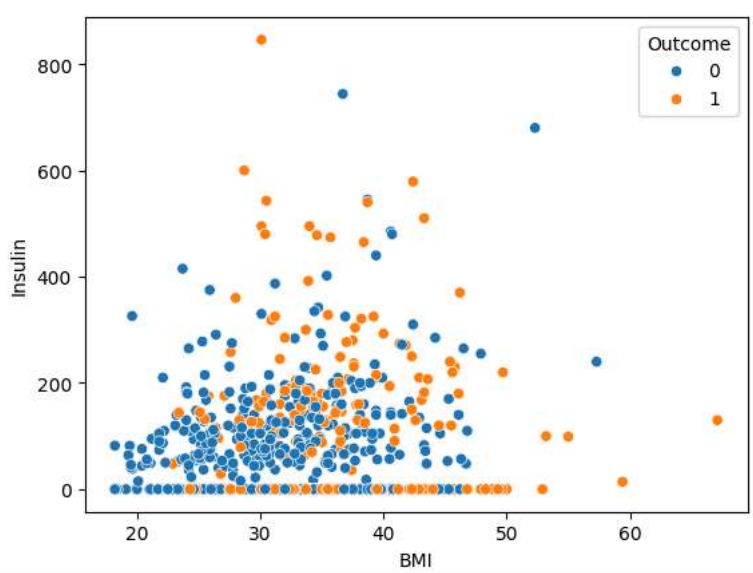
```
BMI = Positive['BMI']
create_scatter_plot(Positive['BloodPressure'],Positive['Glucose'],'BloodPressure','Glucose','blue')
```



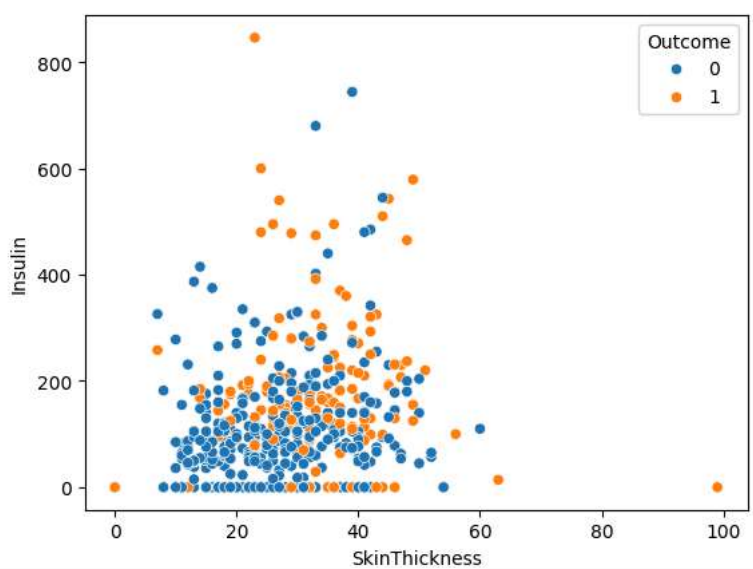

```
g =sns.scatterplot(x= "BloodPressure" ,y= "Glucose",
              hue="Outcome",
              data=diabetes_mod);
```




```
B=sns.scatterplot(x="BMI",y="Insulin",
                  hue="Outcome",data=diabetes_mod);
```




```
s=sns.scatterplot(x="SkinThickness",y="Insulin",hue="Outcome",data=diabetes_mod);
```




```
##correlation matrix
diabetes_mod.corr()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Ou
Pregnancies	1.000000	0.134915	0.209668	-0.095683	-0.080059	0.012342	-0.025996	0.557066	0.2
Glucose	0.134915	1.000000	0.223331	0.074381	0.337896	0.223276	0.136630	0.263560	0.4
BloodPressure	0.209668	0.223331	1.000000	0.011777	-0.046856	0.287403	-0.000075	0.324897	0.1
SkinThickness	-0.095683	0.074381	0.011777	1.000000	0.420874	0.401528	0.176253	-0.128908	0.0
Insulin	-0.080059	0.337896	-0.046856	0.420874	1.000000	0.191831	0.182656	-0.049412	0.1
BMI	0.012342	0.223276	0.287403	0.401528	0.191831	1.000000	0.154858	0.020835	0.2
DiabetesPedigreeFunction	-0.025996	0.136630	-0.000075	0.176253	0.182656	0.154858	1.000000	0.023098	0.1
Age	0.557066	0.263560	0.324897	-0.128908	-0.049412	0.020835	0.023098	1.000000	0.2
Outcome	0.224417	0.488384	0.166703	0.092030	0.145488	0.299375	0.184947	0.245741	1.0

```
feature_names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']
X = diabetes_mod[feature_names]
y = diabetes_mod.Outcome
X.head()
```