M Koushik

2211CS010343

Group - 4

CSE

Dataset Description

The dataset "class_marks.csv" consists of students' marks from various questions in an exam. The structure of the dataset includes columns representing different question numbers and their respective marks. Some questions are divided into sub-parts, such as Q1aM4, Q1bM6, Q2aM6, Q2bM4, and so on. These sub-parts indicate that the total score for a question is the sum of multiple components.

Upon initial inspection, the dataset contained some missing values, which were later handled during the cleaning process. The Total column represents the sum of all question scores for each student, indicating their final score in the exam. Several data transformation steps were necessary, including merging sub-questions into single columns (e.g., combining Q1aM4 and Q1bM6 into Q1). Additionally, redundant columns were dropped to streamline the dataset.

To ensure consistency, the dataset was converted to integer format (int64) after handling missing values. Some specific score values (like 39 and 36) were replaced with 40, possibly to correct inconsistencies in grading. These modifications helped refine the dataset for more effective analysis.

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
df = pd.read_csv("class_marks.csv")
```

In [ ]:
```
df
```

Class Marks Data

In [5]:
```
df[df.Total>40].count
df
```

Out[5]:

	Total	Q1aM4	Q1bM6	Q2aM6	Q2bM4	Q3aM5	Q3bM5	Q4aM3	Q4bM7	Q5M10	Q6aM4	Q6bM6
0	37	4.0	5.0	6.0	4.0	2.0	1.0	NaN	5.0	8.0	4.0	6.0
1	32	4.0	3.0	4.0	3.0	NaN	NaN	3.0	6.0	9.0	NaN	NaN
2	33	4.0	5.0	5.0	1.0	5.0	5.0	NaN	NaN	8.0	NaN	NaN
3	24	4.0	6.0	6.0	3.0	2.0	2.0	NaN	NaN	NaN	2.0	NaN
4	36	3.0	6.0	4.0	4.0	5.0	4.0	NaN	NaN	10.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
81	32	3.0	6.0	3.0	4.0	5.0	3.0	NaN	NaN	NaN	4.0	6.0
82	27	2.0	2.0	5.0	3.0	NaN	NaN	NaN	NaN	7.0	3.0	5.0
83	37	4.0	6.0	6.0	2.0	NaN	NaN	NaN	NaN	9.0	4.0	6.0
84	28	4.0	NaN	5.0	4.0	5.0	4.0	NaN	NaN	6.0	NaN	NaN
85	29	4.0	6.0	NaN	NaN	NaN	NaN	3.0	5.0	7.0	1.0	4.0

86 rows × 12 columns

Class Marks Total Greater than 40

In [6]:
```
df.Total.value_counts()
```

Out[6]:
```
36    7
32    6
34    5
40    5
38    5
37    4
27    4
29    4
25    4
20    4
24    4
33    4
31    3
30    3
26    3
28    3
22    3
35    3
17    2
21    2
39    2
19    1
9     1
14    1
8     1
18    1
3     1
Name: Total, dtype: int64
```

In [7]:
```
df.replace(39, 40)
```

Out[7]:

	Total	Q1aM4	Q1bM6	Q2aM6	Q2bM4	Q3aM5	Q3bM5	Q4aM3	Q4bM7	Q5M10	Q6aM4	Q6bM6
0	37	4.0	5.0	6.0	4.0	2.0	1.0	NaN	5.0	8.0	4.0	6.0
1	32	4.0	3.0	4.0	3.0	NaN	NaN	3.0	6.0	9.0	NaN	NaN
2	33	4.0	5.0	5.0	1.0	5.0	5.0	NaN	NaN	8.0	NaN	NaN
3	24	4.0	6.0	6.0	3.0	2.0	2.0	NaN	NaN	NaN	2.0	NaN
4	36	3.0	6.0	4.0	4.0	5.0	4.0	NaN	NaN	10.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
81	32	3.0	6.0	3.0	4.0	5.0	3.0	NaN	NaN	NaN	4.0	6.0
82	27	2.0	2.0	5.0	3.0	NaN	NaN	NaN	NaN	7.0	3.0	5.0
83	37	4.0	6.0	6.0	2.0	NaN	NaN	NaN	NaN	9.0	4.0	6.0
84	28	4.0	NaN	5.0	4.0	5.0	4.0	NaN	NaN	6.0	NaN	NaN
85	29	4.0	6.0	NaN	NaN	NaN	NaN	3.0	5.0	7.0	1.0	4.0

86 rows × 12 columns

Replacing the Total Marks Value 39 to 40 in Entire data set

In [8]:
```
df.Total.value_counts()
```

Out[8]:
```
36    7
32    6
34    5
40    5
38    5
37    4
27    4
29    4
25    4
20    4
24    4
33    4
31    3
30    3
26    3
28    3
22    3
35    3
17    2
21    2
39    2
19    1
9     1
14    1
8     1
18    1
3     1
Name: Total, dtype: int64
```

In [9]:
```
df.replace(36, 40)
```

Out[9]:

	Total	Q1aM4	Q1bM6	Q2aM6	Q2bM4	Q3aM5	Q3bM5	Q4aM3	Q4bM7	Q5M10	Q6aM4	Q6bM6
0	37	4.0	5.0	6.0	4.0	2.0	1.0	NaN	5.0	8.0	4.0	6.0
1	32	4.0	3.0	4.0	3.0	NaN	NaN	3.0	6.0	9.0	NaN	NaN
2	33	4.0	5.0	5.0	1.0	5.0	5.0	NaN	NaN	8.0	NaN	NaN
3	24	4.0	6.0	6.0	3.0	2.0	2.0	NaN	NaN	NaN	2.0	NaN
4	40	3.0	6.0	4.0	4.0	5.0	4.0	NaN	NaN	10.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
81	32	3.0	6.0	3.0	4.0	5.0	3.0	NaN	NaN	NaN	4.0	6.0
82	27	2.0	2.0	5.0	3.0	NaN	NaN	NaN	NaN	7.0	3.0	5.0
83	37	4.0	6.0	6.0	2.0	NaN	NaN	NaN	NaN	9.0	4.0	6.0
84	28	4.0	NaN	5.0	4.0	5.0	4.0	NaN	NaN	6.0	NaN	NaN
85	29	4.0	6.0	NaN	NaN	NaN	NaN	3.0	5.0	7.0	1.0	4.0

86 rows × 12 columns

Replacing the Marks 36 to 40

In [10]:
```
df.replace(36, 40).Total.value_counts()
```

Out[10]:
```
40    12
32     6
34     5
38     5
37     4
27     4
29     4
25     4
24     4
33     4
20     4
28     3
31     3
22     3
26     3
30     3
35     3
39     2
21     2
17     2
14     1
9      1
19     1
8      1
18     1
3      1
Name: Total, dtype: int64
```

In [11]:
```
df
```

Out[11]:

	Total	Q1aM4	Q1bM6	Q2aM6	Q2bM4	Q3aM5	Q3bM5	Q4aM3	Q4bM7	Q5M10	Q6aM4	Q6bM6
0	37	4.0	5.0	6.0	4.0	2.0	1.0	NaN	5.0	8.0	4.0	6.0
1	32	4.0	3.0	4.0	3.0	NaN	NaN	3.0	6.0	9.0	NaN	NaN
2	33	4.0	5.0	5.0	1.0	5.0	5.0	NaN	NaN	8.0	NaN	NaN
3	24	4.0	6.0	6.0	3.0	2.0	2.0	NaN	NaN	NaN	2.0	NaN
4	36	3.0	6.0	4.0	4.0	5.0	4.0	NaN	NaN	10.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
81	32	3.0	6.0	3.0	4.0	5.0	3.0	NaN	NaN	NaN	4.0	6.0
82	27	2.0	2.0	5.0	3.0	NaN	NaN	NaN	NaN	7.0	3.0	5.0
83	37	4.0	6.0	6.0	2.0	NaN	NaN	NaN	NaN	9.0	4.0	6.0
84	28	4.0	NaN	5.0	4.0	5.0	4.0	NaN	NaN	6.0	NaN	NaN
85	29	4.0	6.0	NaN	NaN	NaN	NaN	3.0	5.0	7.0	1.0	4.0

86 rows × 12 columns

In [12]:
```
df["Q3"] = df["Q3aM5"] + df["Q3bM5"]
df["Q4"] = df["Q4aM3"] + df["Q4bM7"]
df.drop(["Q2aM6", "Q2bM4" , "Q3aM5" , "Q3bM5" , "Q4aM3" ,"Q4bM7"], axis=1, inplace=Tru
df
```

Out[12]:

	Total	Q1aM4	Q1bM6	Q5M10	Q6aM4	Q6bM6	Q3	Q4
0	37	4.0	5.0	8.0	4.0	6.0	3.0	NaN
1	32	4.0	3.0	9.0	NaN	NaN	NaN	9.0
2	33	4.0	5.0	8.0	NaN	NaN	10.0	NaN
3	24	4.0	6.0	NaN	2.0	NaN	4.0	NaN
4	36	3.0	6.0	10.0	NaN	NaN	9.0	NaN
...	...	...	...	...	...	...	...	...
81	32	3.0	6.0	NaN	4.0	6.0	8.0	NaN
82	27	2.0	2.0	7.0	3.0	5.0	NaN	NaN
83	37	4.0	6.0	9.0	4.0	6.0	NaN	NaN
84	28	4.0	NaN	6.0	NaN	NaN	9.0	NaN
85	29	4.0	6.0	7.0	1.0	4.0	NaN	8.0

86 rows × 8 columns

Merging the Two columns And Naming as One Column and Dropping the columns merged

In [13]:
```
df["Q5"] = df["Q5M10"]
df["Q6"] = df["Q6aM4"] +df["Q6bM6"]
df.drop(["Q5M10", "Q6aM4","Q6bM6"],axis=1,inplace=True)
df
```

Out[13]:

	Total	Q1aM4	Q1bM6	Q3	Q4	Q5	Q6
0	37	4.0	5.0	3.0	NaN	8.0	10.0
1	32	4.0	3.0	NaN	9.0	9.0	NaN
2	33	4.0	5.0	10.0	NaN	8.0	NaN
3	24	4.0	6.0	4.0	NaN	NaN	NaN
4	36	3.0	6.0	9.0	NaN	10.0	NaN
...	...	...	...	...	...	...	...
81	32	3.0	6.0	8.0	NaN	NaN	10.0
82	27	2.0	2.0	NaN	NaN	7.0	8.0
83	37	4.0	6.0	NaN	NaN	9.0	10.0
84	28	4.0	NaN	9.0	NaN	6.0	NaN
85	29	4.0	6.0	NaN	8.0	7.0	5.0

86 rows × 7 columns

In [14]:
```
df.Q6==10
```

Out[14]:
```
0      True
1     False
2     False
3     False
4     False
      ...
81     True
82    False
83     True
84    False
85    False
Name: Q6, Length: 86, dtype: bool
```

The Question Q6 who got 10 marks returns True else False

In [15]:
```
df.Total==40
```

Out[15]:
```
0     False
1     False
2     False
3     False
4     False
      ...
81    False
82    False
83    False
84    False
85    False
Name: Total, Length: 86, dtype: bool
```

In [16]:
```
df.loc[(df.Total == 40)]
```

Out[16]:

	Total	Q1aM4	Q1bM6	Q3	Q4	Q5	Q6
33	40	NaN	NaN	10.0	10.0	NaN	10.0
51	40	0.0	NaN	NaN	10.0	10.0	NaN
53	40	4.0	6.0	10.0	NaN	10.0	NaN
65	40	4.0	6.0	10.0	NaN	10.0	NaN
73	40	4.0	6.0	10.0	NaN	10.0	10.0

Specifies the Specific location where the Marks who got 40

In [17]:
```
df.loc[(df.Total == 40) & (df.Q6 == 10)]
```

Out[17]:

	Total	Q1aM4	Q1bM6	Q3	Q4	Q5	Q6
33	40	NaN	NaN	10.0	10.0	NaN	10.0
73	40	4.0	6.0	10.0	NaN	10.0	10.0

Specifies the specific location who got total 40 marks and also 10 marks in Q6

In [18]:
```
df
```

Out[18]:

	Total	Q1aM4	Q1bM6	Q3	Q4	Q5	Q6
0	37	4.0	5.0	3.0	NaN	8.0	10.0
1	32	4.0	3.0	NaN	9.0	9.0	NaN
2	33	4.0	5.0	10.0	NaN	8.0	NaN
3	24	4.0	6.0	4.0	NaN	NaN	NaN
4	36	3.0	6.0	9.0	NaN	10.0	NaN
...	...	...	...	...	...	...	...
81	32	3.0	6.0	8.0	NaN	NaN	10.0
82	27	2.0	2.0	NaN	NaN	7.0	8.0
83	37	4.0	6.0	NaN	NaN	9.0	10.0
84	28	4.0	NaN	9.0	NaN	6.0	NaN
85	29	4.0	6.0	NaN	8.0	7.0	5.0

86 rows × 7 columns

In [19]:
```
df["Q1"] = df["Q1aM4"] +df["Q1bM6"]
df.drop(["Q1aM4","Q1bM6"],axis=1,inplace=True)
df
```

Out[19]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3.0	NaN	8.0	10.0	9.0
1	32	NaN	9.0	9.0	NaN	7.0
2	33	10.0	NaN	8.0	NaN	9.0
3	24	4.0	NaN	NaN	NaN	10.0
4	36	9.0	NaN	10.0	NaN	9.0
...	...	...	...	...	...	...
81	32	8.0	NaN	NaN	10.0	9.0
82	27	NaN	NaN	7.0	8.0	4.0
83	37	NaN	NaN	9.0	10.0	10.0
84	28	9.0	NaN	6.0	NaN	NaN
85	29	NaN	8.0	7.0	5.0	10.0

86 rows × 6 columns

In [20]:
```
df
```

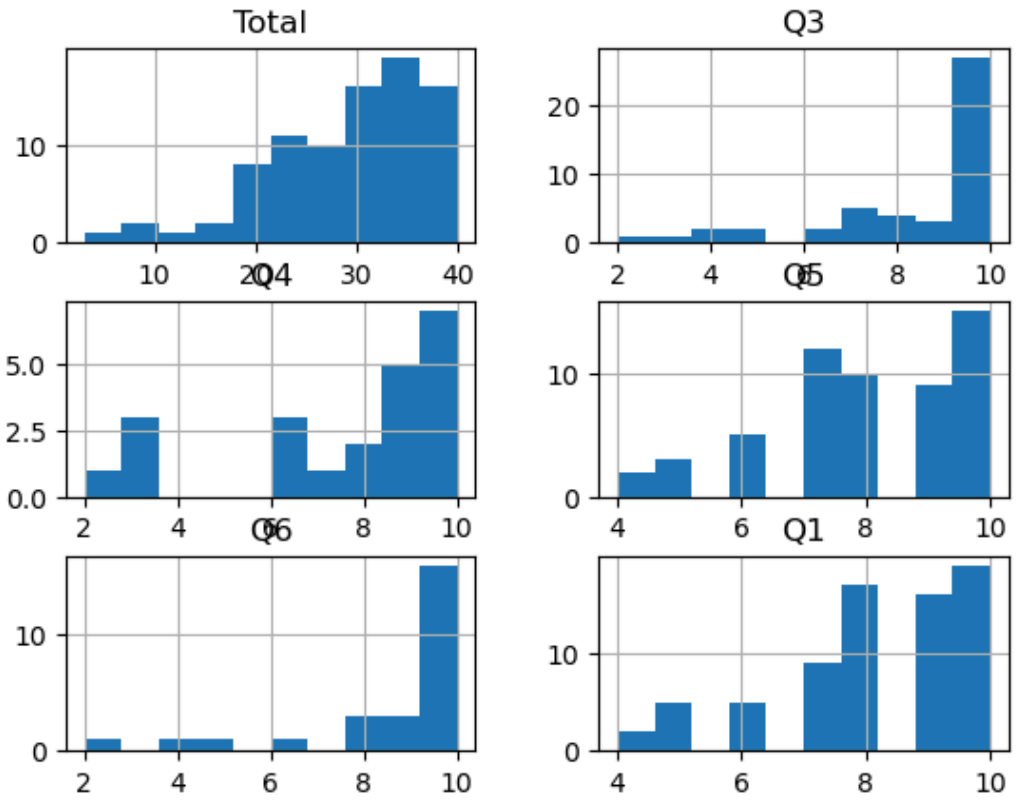
Out[20]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3.0	NaN	8.0	10.0	9.0
1	32	NaN	9.0	9.0	NaN	7.0
2	33	10.0	NaN	8.0	NaN	9.0
3	24	4.0	NaN	NaN	NaN	10.0
4	36	9.0	NaN	10.0	NaN	9.0
...	...	...	...	...	...	...
81	32	8.0	NaN	NaN	10.0	9.0
82	27	NaN	NaN	7.0	8.0	4.0
83	37	NaN	NaN	9.0	10.0	10.0
84	28	9.0	NaN	6.0	NaN	NaN
85	29	NaN	8.0	7.0	5.0	10.0

86 rows × 6 columns

In [21]:
```
df.hist()
```

Out[21]:
```
array([[<AxesSubplot:title={'center':'Total'}>,
        <AxesSubplot:title={'center':'Q3'}>],
       [<AxesSubplot:title={'center':'Q4'}>,
        <AxesSubplot:title={'center':'Q5'}>],
       [<AxesSubplot:title={'center':'Q6'}>,
        <AxesSubplot:title={'center':'Q1'}>]], dtype=object)
```



Histogram of all columns in the Dataset

In [22]:
```
df = df.fillna(0)
```

Filling 0 to the all null values in the dataset

In [23]:
```
df
```

Out[23]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3.0	0.0	8.0	10.0	9.0
1	32	0.0	9.0	9.0	0.0	7.0
2	33	10.0	0.0	8.0	0.0	9.0
3	24	4.0	0.0	0.0	0.0	10.0
4	36	9.0	0.0	10.0	0.0	9.0
...	...	...	...	...	...	...
81	32	8.0	0.0	0.0	10.0	9.0
82	27	0.0	0.0	7.0	8.0	4.0
83	37	0.0	0.0	9.0	10.0	10.0
84	28	9.0	0.0	6.0	0.0	0.0
85	29	0.0	8.0	7.0	5.0	10.0

86 rows × 6 columns

In [24]:
```
df = df.astype("int64")
```

Converting the datatype float to int64

In [25]:
```
df
```

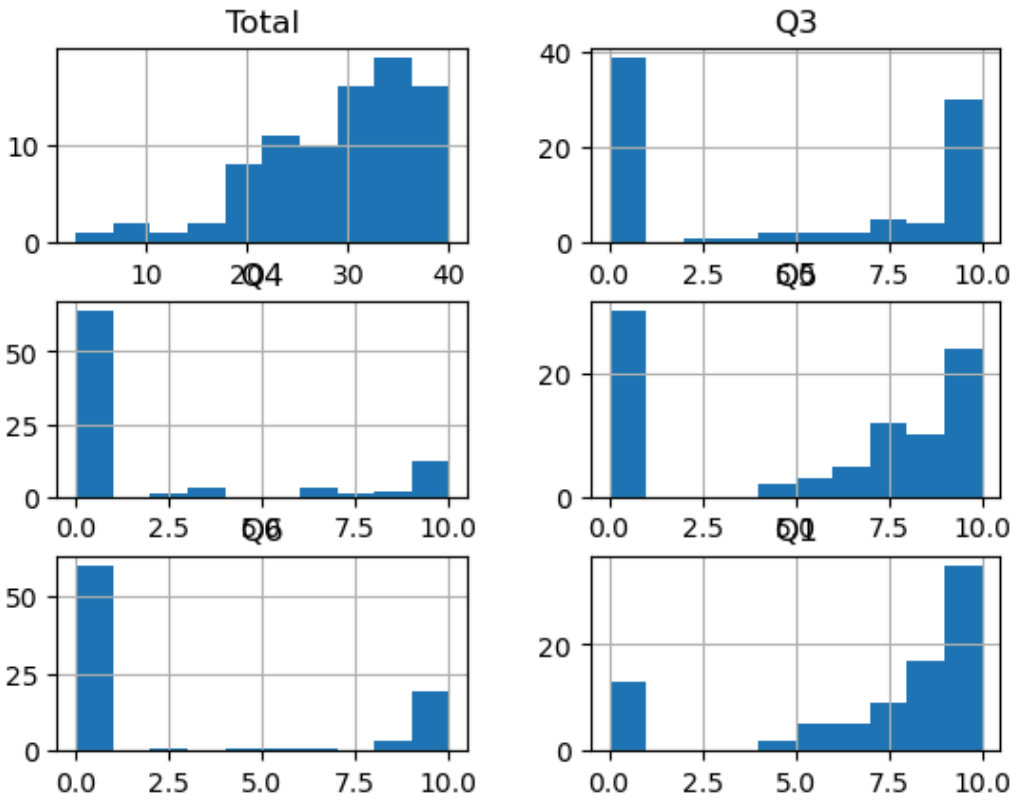
Out[25]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3	0	8	10	9
1	32	0	9	9	0	7
2	33	10	0	8	0	9
3	24	4	0	0	0	10
4	36	9	0	10	0	9
...	...	...	...	...	...	...
81	32	8	0	0	10	9
82	27	0	0	7	8	4
83	37	0	0	9	10	10
84	28	9	0	6	0	0
85	29	0	8	7	5	10

86 rows × 6 columns

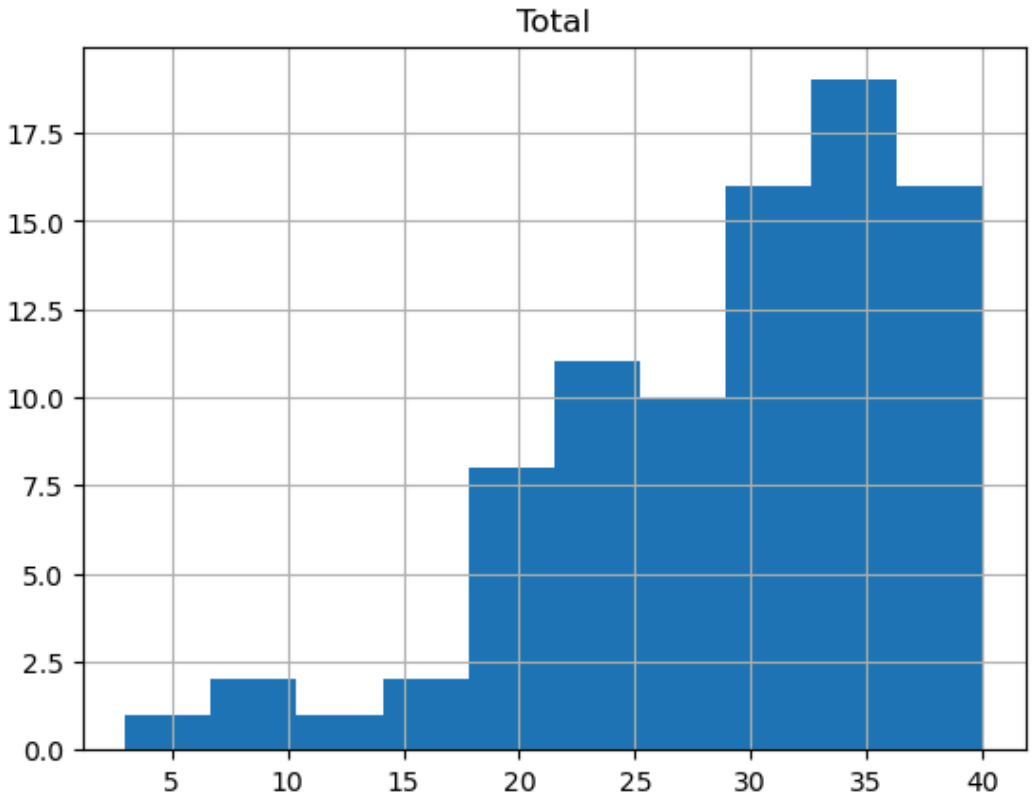
In [26]:
```
df.hist()
```

Out[26]:
```
array([[<AxesSubplot:title={'center':'Total'}>,
        <AxesSubplot:title={'center':'Q3'}>],
       [<AxesSubplot:title={'center':'Q4'}>,
        <AxesSubplot:title={'center':'Q5'}>],
       [<AxesSubplot:title={'center':'Q6'}>,
        <AxesSubplot:title={'center':'Q1'}>]], dtype=object)
```



In [27]:
```
df.hist("Total")
```

Out[27]:
```
array([[<AxesSubplot:title={'center':'Total'}>]], dtype=object)
```

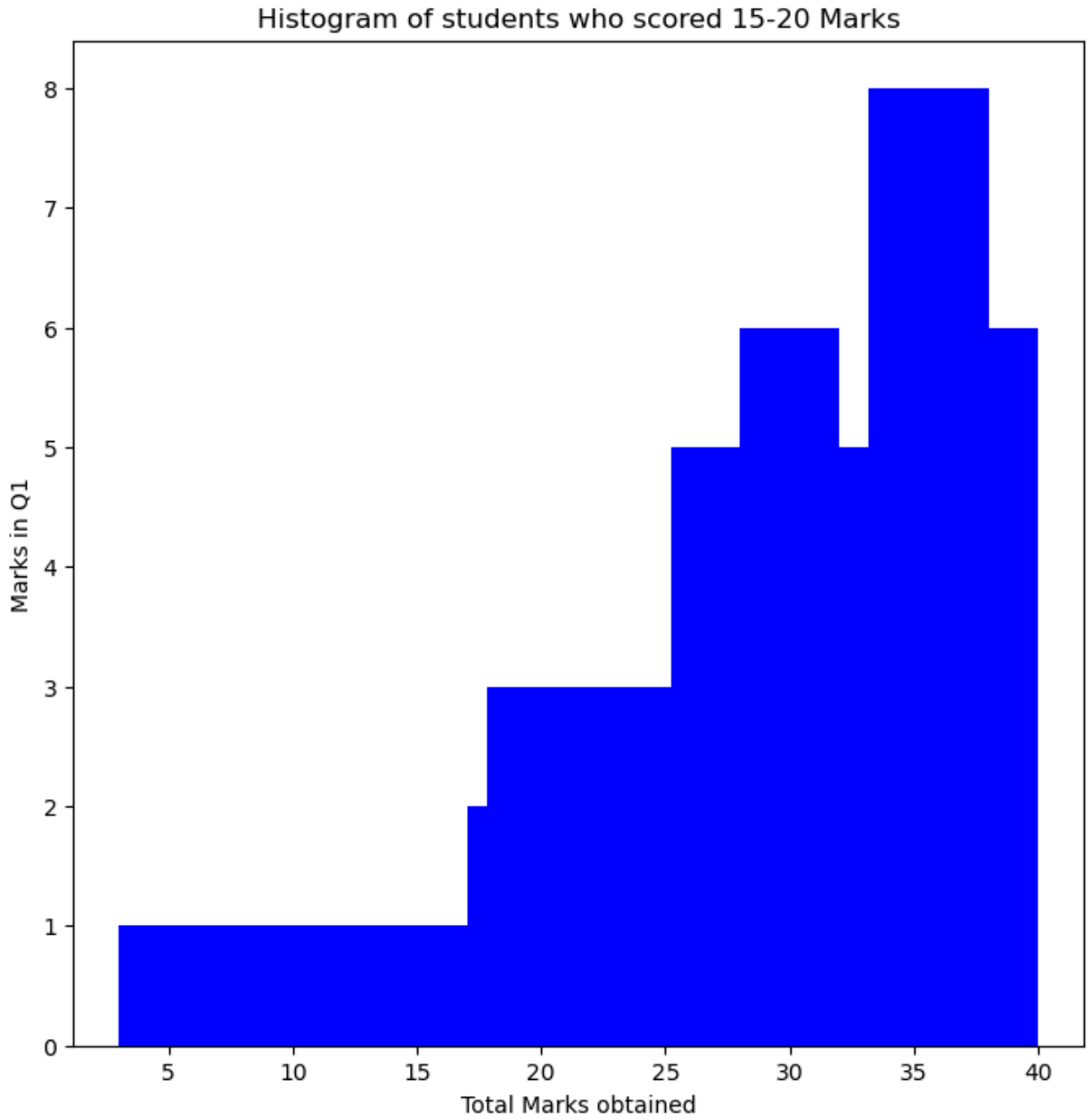


Histogram of Total Marks column

Most of the above 20

Students are highest at the 35 marks

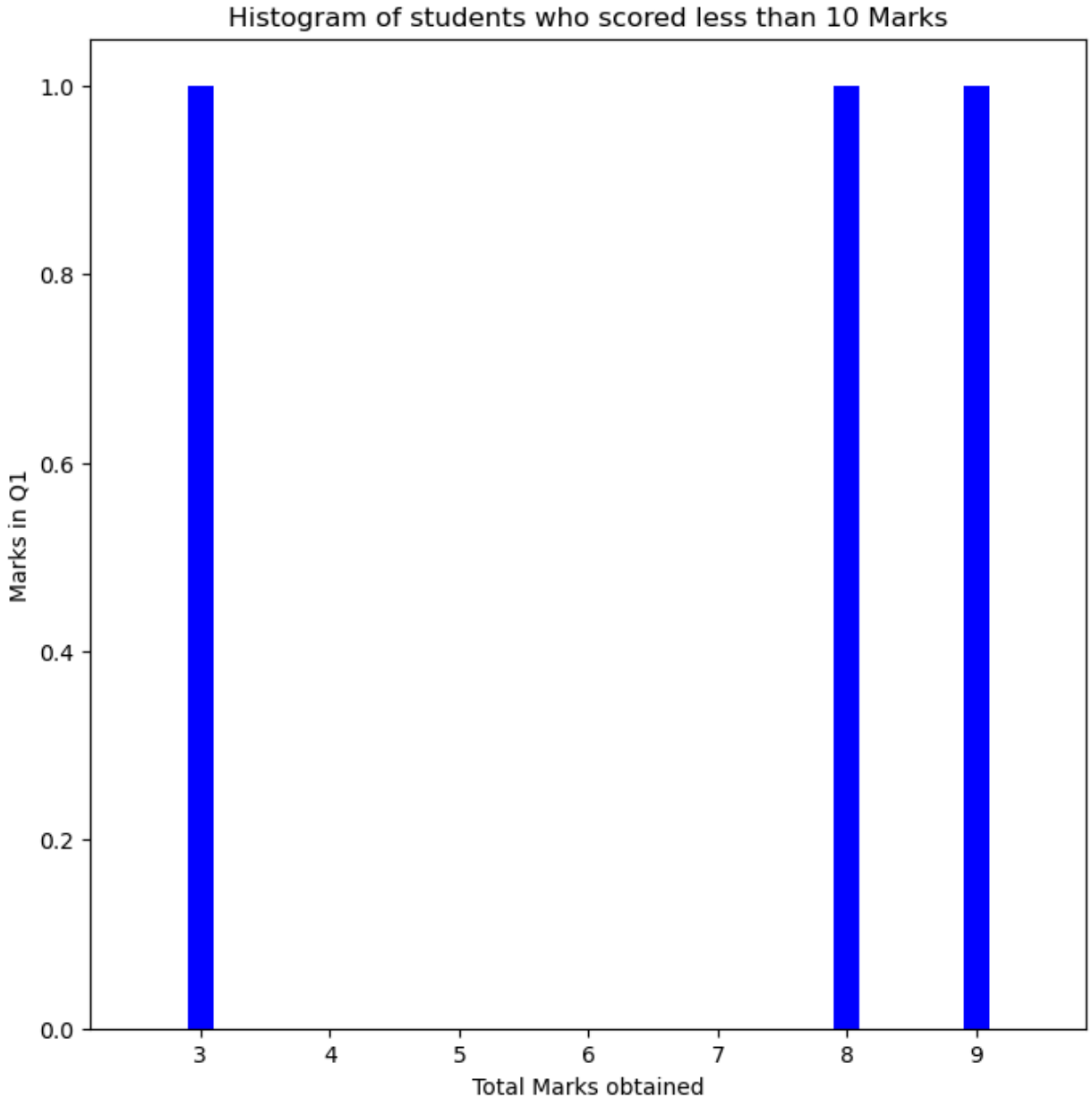
In [28]:
```
k = df.groupby('Q1')['Total']
k.hist(color='blue', figsize=[8,8], grid=False, bins=5)
plt.title("Histogram of students who scored 15-20 Marks")
plt.xlabel("Total Marks obtained")
plt.ylabel("Marks in Q1")
plt.show()
```



In [ ]:

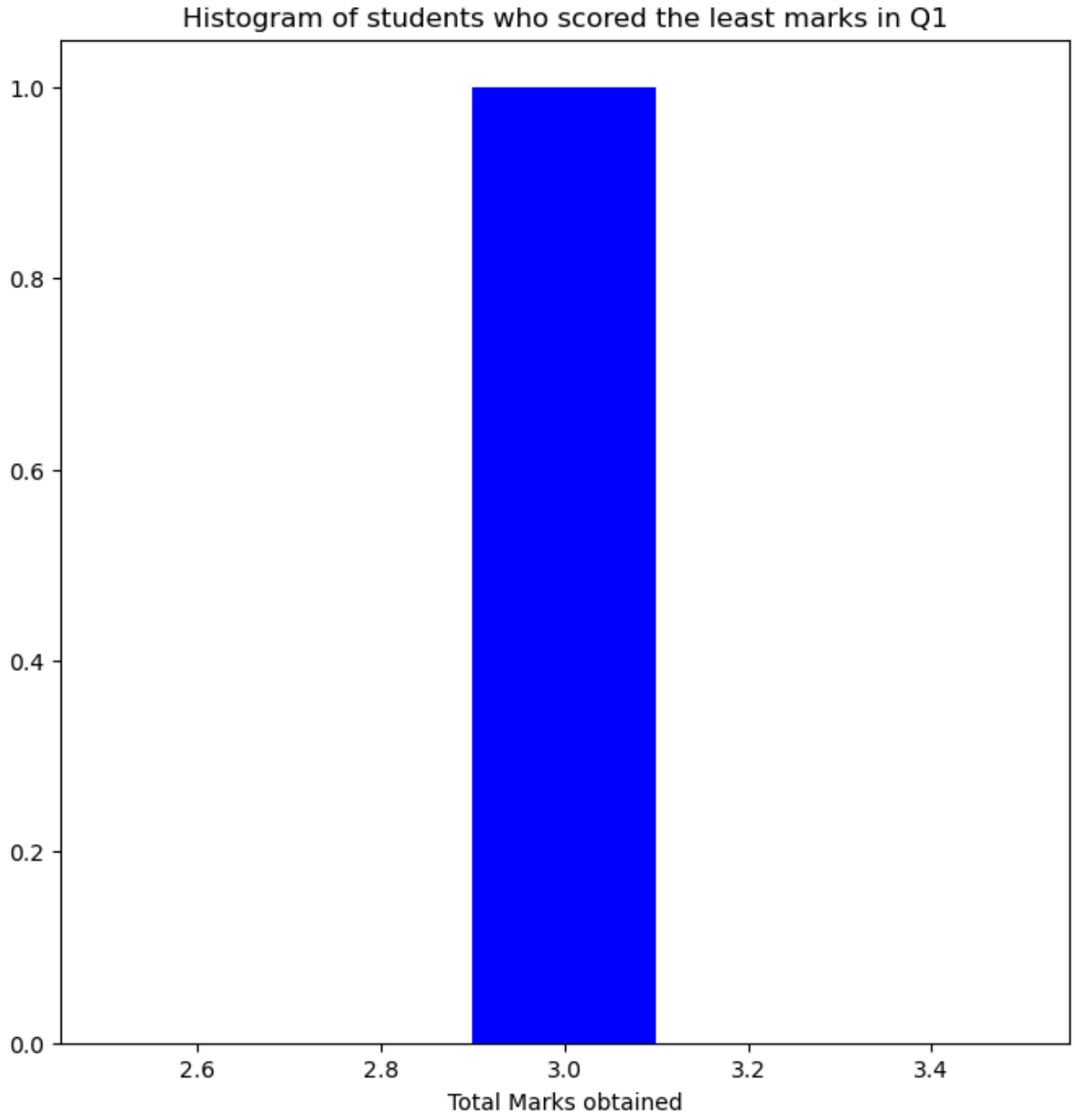
Students scored 35 marks are more in Q1 Answer

In [29]:
```
filtered_df = df[df['Total'] < 10]
k = filtered_df.groupby('Q1')['Total']
k.hist(color='blue', figsize=[8,8], grid=False, bins=5)
plt.title("Histogram of students who scored less than 10 Marks")
plt.xlabel("Total Marks obtained")
plt.ylabel("Marks in Q1")
plt.show()
```



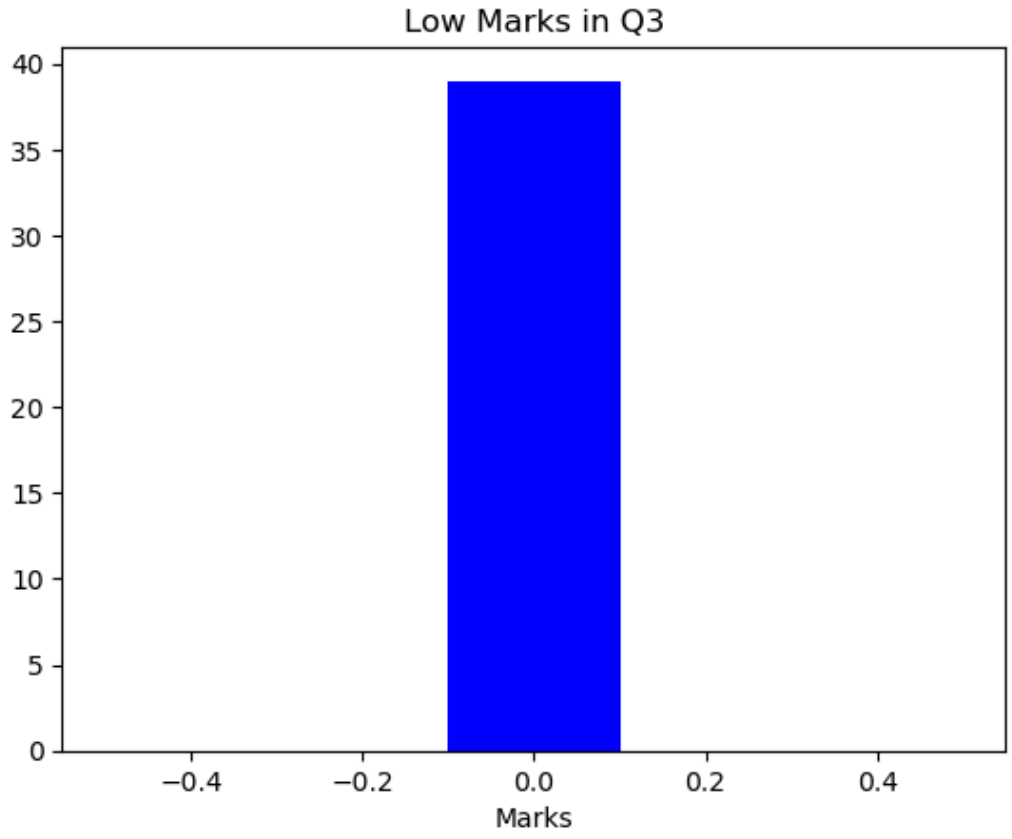
Students below 10 marks majorly got 3 , 8 and 9 marks

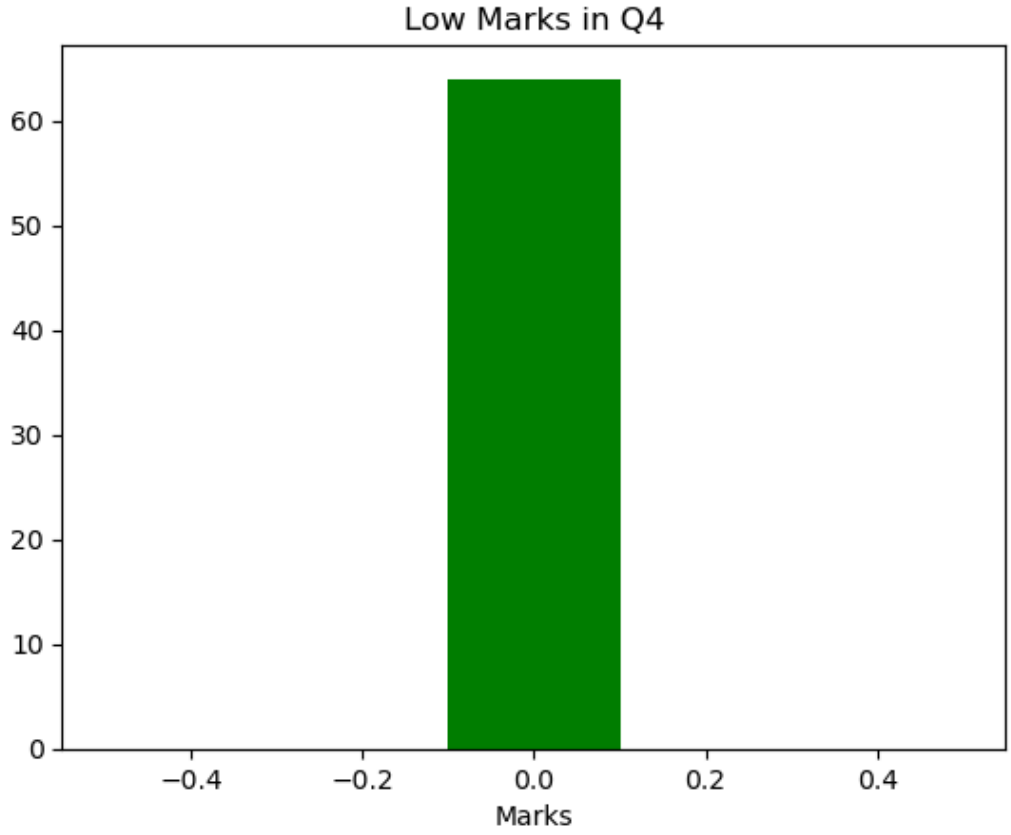
In [30]:
```
min_marks_q1 = df['Total'].min()
low_marks = df[df['Total'] == min_marks_q1]
low_marks['Total'].hist(color='blue', figsize=[8,8], grid=False, bins=5)
plt.title("Histogram of students who scored the least marks in Q1")
plt.xlabel("Total Marks obtained")
plt.ylabel("")
plt.show()
```

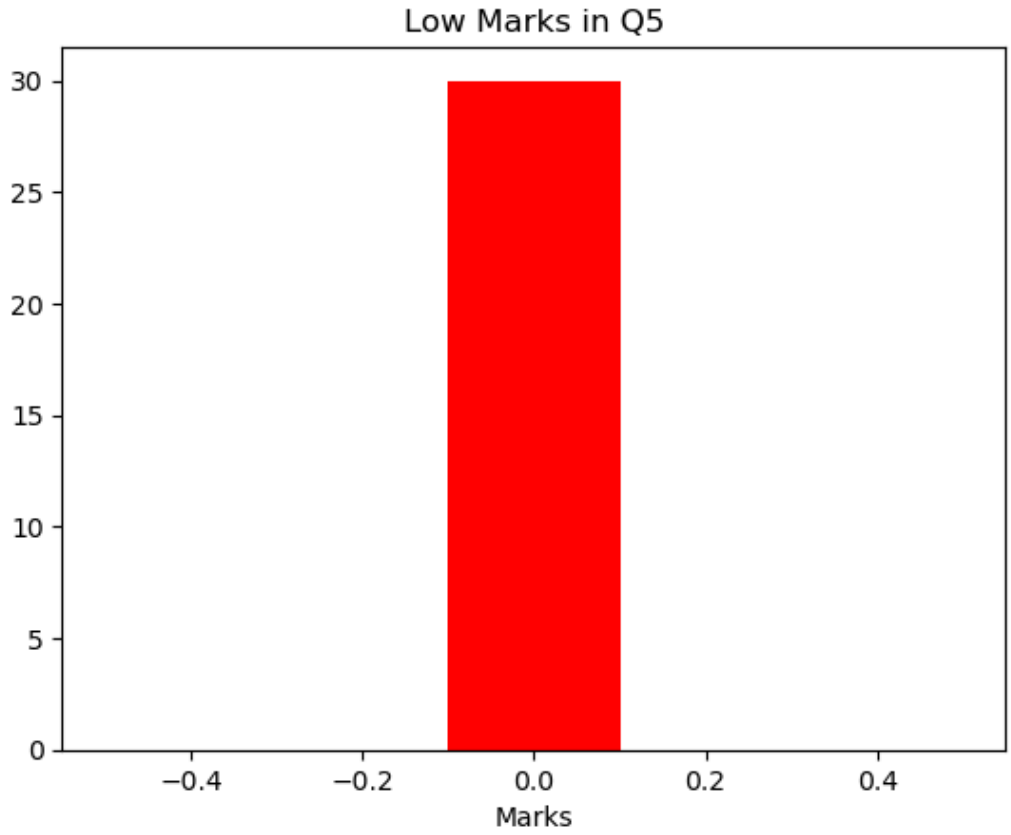


least marks in the Q1 is 3 maximum of students scored 3 Marks

In [31]:
```
marks_Q3 = df['Q3'].min()
marks_Q4 = df['Q4'].min()
marks_Q5 = df['Q5'].min()

low_marks_Q3 = df[df['Q3'] == marks_Q3]
low_marks_Q4 = df[df['Q4'] == marks_Q4]
low_marks_Q5 = df[df['Q5'] == marks_Q5]

low_marks_Q3['Q3'].hist(color='blue', bins=5, grid=False)
plt.title("Low Marks in Q3")
plt.xlabel("Marks")
plt.show()

low_marks_Q4['Q4'].hist(color='green', bins=5, grid=False)
plt.title("Low Marks in Q4")
plt.xlabel("Marks")
plt.show()

low_marks_Q5['Q5'].hist(color='red', bins=5, grid=False)
plt.title("Low Marks in Q5")
plt.xlabel("Marks")
plt.show()
```

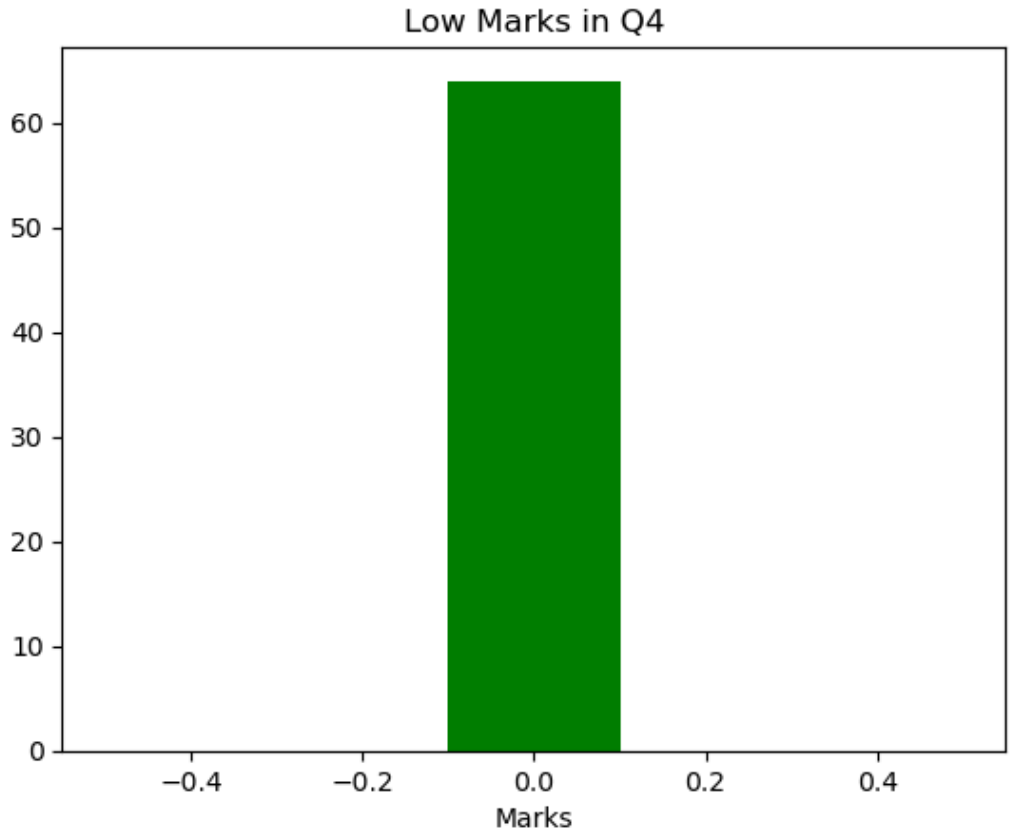


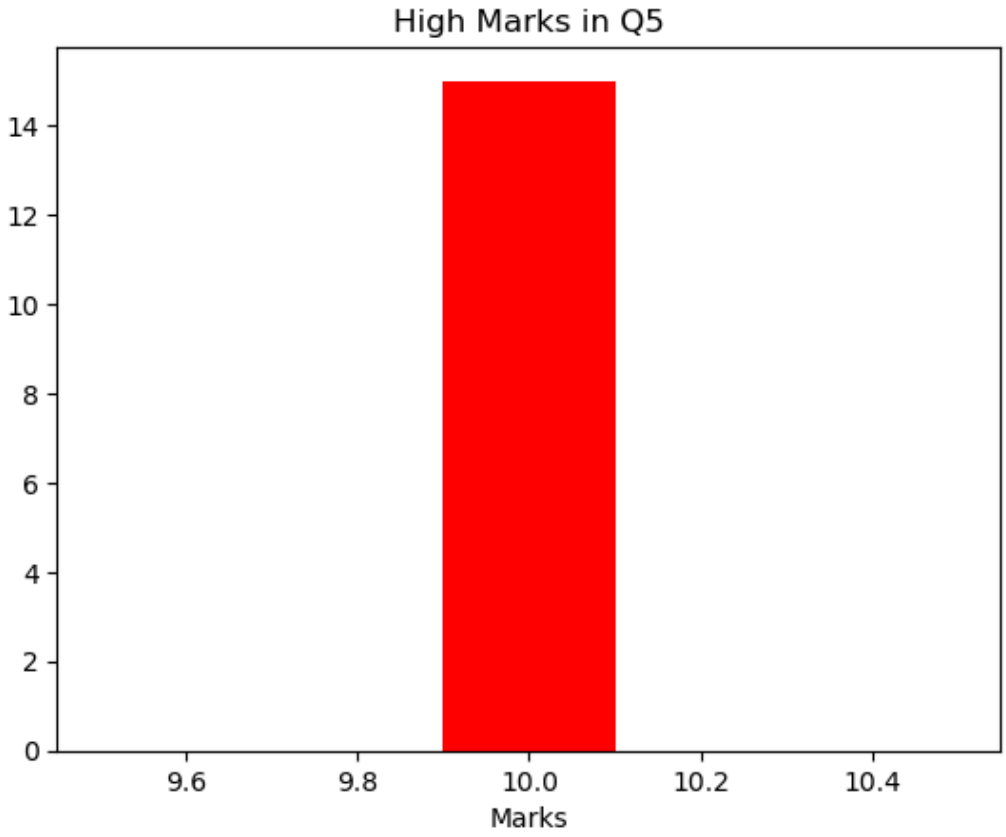




In Q3, Q4, Q5 Question students scored least marks is 0

In [32]:
```
marks_Q4 = df['Q4'].min()
marks_Q5 = df['Q5'].max()

low_marks_Q4 = df[df['Q4'] == marks_Q4]
high_marks_Q5 = df[df['Q5'] == marks_Q5]

low_marks_Q4['Q4'].hist(color='green', bins=5, grid=False)
plt.title("Low Marks in Q4")
plt.xlabel("Marks")
plt.show()

high_marks_Q5['Q5'].hist(color='red', bins=5, grid=False)
plt.title("High Marks in Q5")
plt.xlabel("Marks")
plt.show()
```

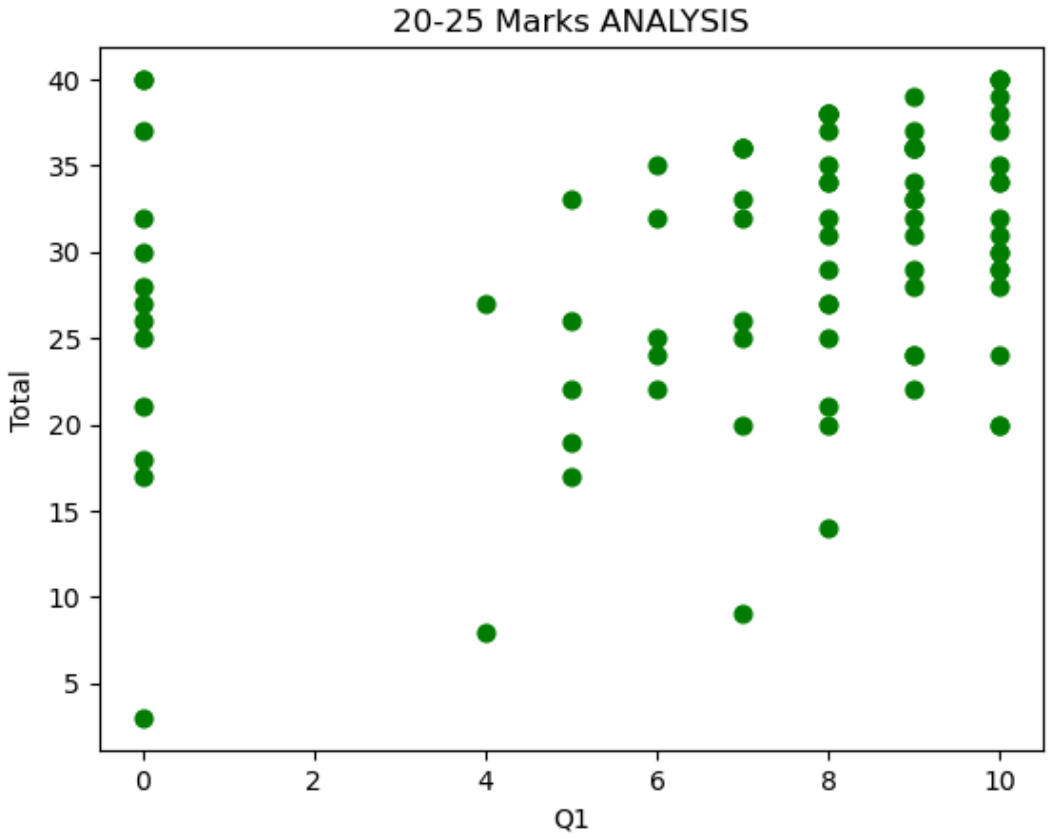




Low marks in , Q4 is 0 scored by students

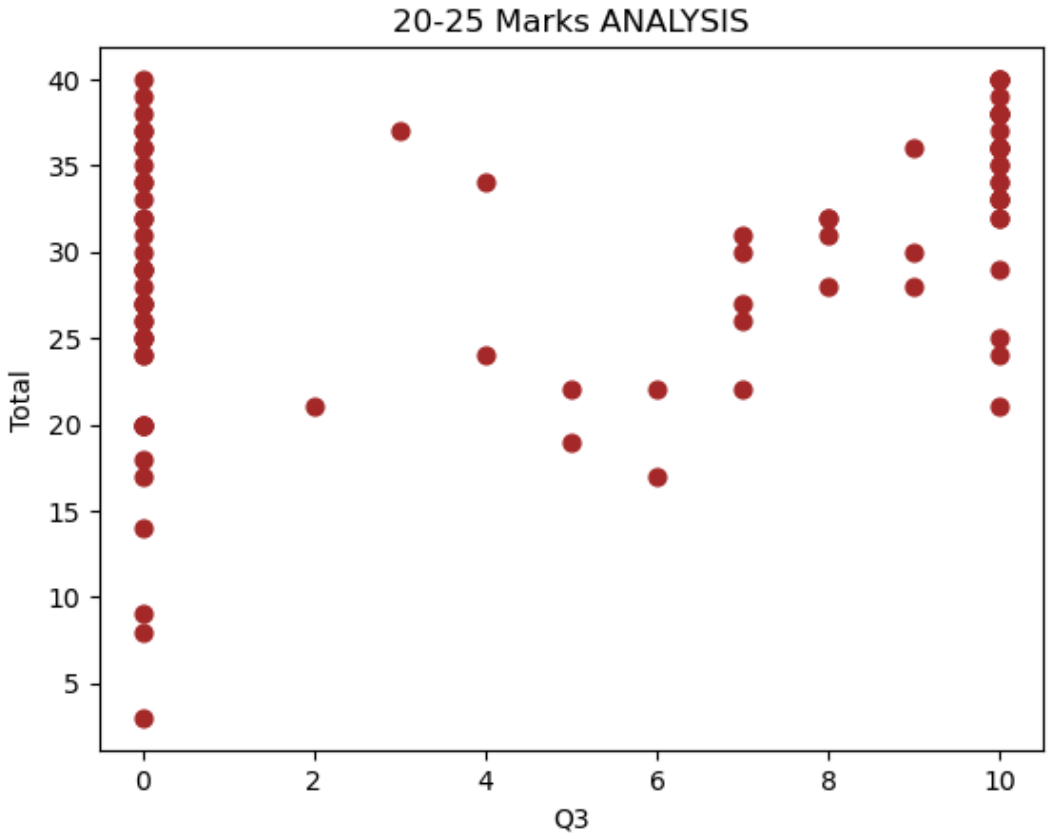
More than 14 number of students scored above 10 marks in the Q5

In [33]:
```
df.plot.scatter(x='Q1',y='Total',color='green',s=40)
plt.title("20-25 Marks ANALYSIS")
plt.show()
```



Students scored mostly 6 to 10 members

In [34]:
```
df.plot.scatter(x='Q3',y='Total',color='brown',s=40)
plt.title("20-25 Marks ANALYSIS")
plt.show()
```



students scored 0 marks maximum

In [37]:
```
c = df.loc[(df['Total'] >= 30) & (df['Total'] <= 40)]
c = c.reset_index(drop=True)
c
```

Out[37]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3	0	8	10	9
1	32	0	9	9	0	7
2	33	10	0	8	0	9
3	36	9	0	10	0	9
4	34	0	0	0	0	10
5	35	10	0	0	10	6
6	37	0	9	0	10	8
7	34	4	3	9	4	8
8	32	8	0	9	0	6
9	30	9	0	0	0	10
10	32	10	10	0	10	0
11	30	7	0	8	0	0
12	36	0	0	9	10	7
13	34	10	0	0	0	10
14	33	10	6	7	0	7
15	39	0	0	0	10	10
16	32	10	6	0	0	8
17	38	10	0	10	0	8
18	32	0	0	10	0	10
19	40	10	10	0	10	0
20	30	0	0	8	0	10
21	37	10	0	10	9	0
22	31	0	0	10	0	8
23	38	10	8	0	0	10
24	33	0	7	8	9	9
25	36	0	9	10	0	9
26	34	10	0	6	0	8
27	36	10	0	7	0	9
28	38	10	10	10	0	8
29	39	10	0	10	0	9
30	40	0	10	10	0	0
31	40	10	0	10	0	10
32	38	0	0	10	10	8
33	35	0	10	7	10	8
34	34	0	0	6	0	9
35	38	10	0	10	10	8
36	36	10	0	7	0	7
37	36	10	0	9	0	7

	Total	Q3	Q4	Q5	Q6	Q1
38	40	10	0	10	0	10
39	31	8	6	7	0	9
40	35	10	0	5	0	10
41	36	10	0	7	0	9
42	40	10	0	10	10	10
43	33	10	0	8	0	5
44	31	7	0	6	0	10
45	32	8	0	0	10	9
46	37	0	0	9	10	10

Total marks 30-40 is filtered from the data set

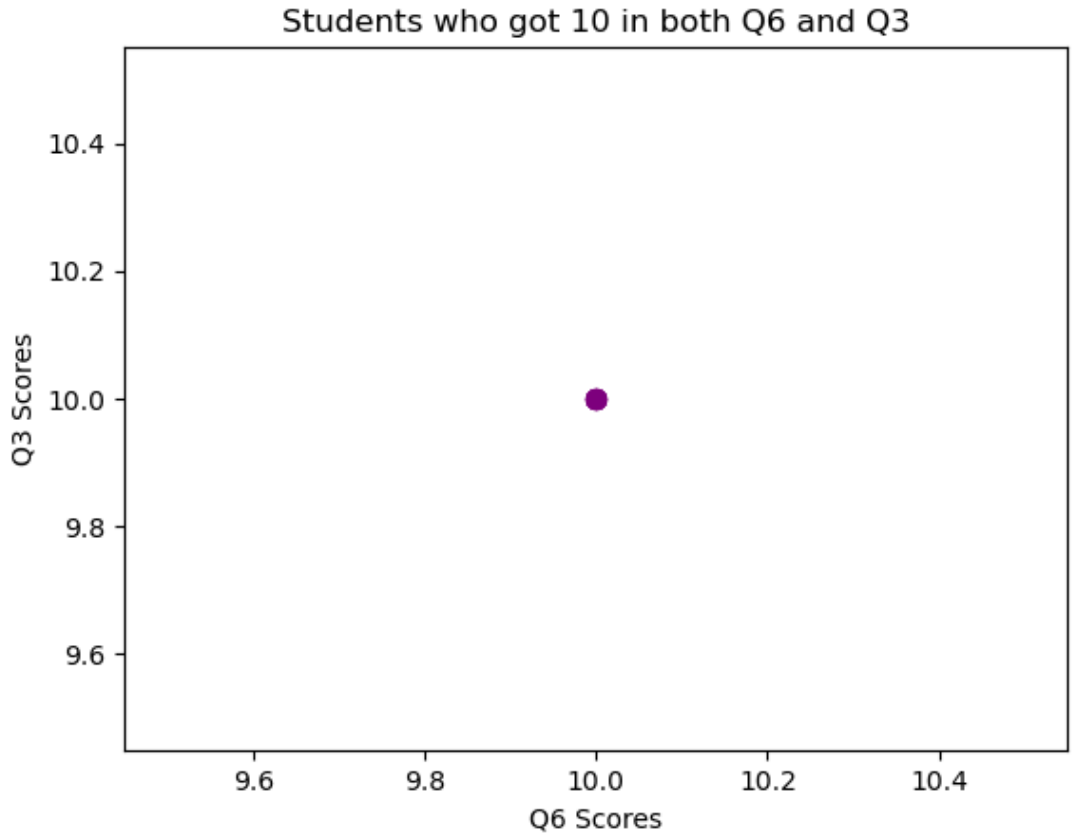
In [38]:
```
c = df.loc[(df['Total'] >= 30) & (df['Total'] <= 40)].head()
c = c.reset_index(drop=True)
c
```

Out[38]:

	Total	Q3	Q4	Q5	Q6	Q1
0	37	3	0	8	10	9
1	32	0	9	9	0	7
2	33	10	0	8	0	9
3	36	9	0	10	0	9
4	34	0	0	0	0	10

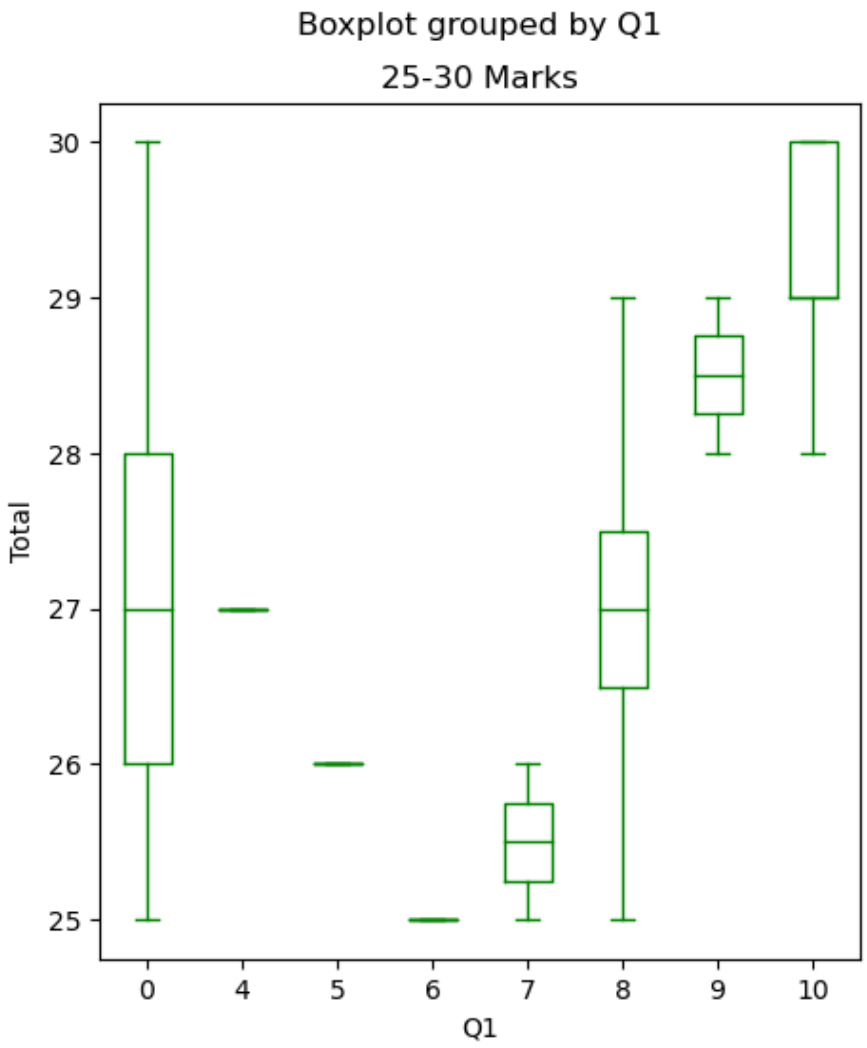
Head of 5 students who got above 30-40 marks

In [39]:
```
filtered_students = df[(df['Q6'] == 10) & (df['Q3'] == 10)]
plt.scatter(filtered_students['Q6'], filtered_students['Q3'], color='purple', s=50)
plt.title("Students who got 10 in both Q6 and Q3")
plt.xlabel("Q6 Scores")
plt.ylabel("Q3 Scores")
plt.show()
```



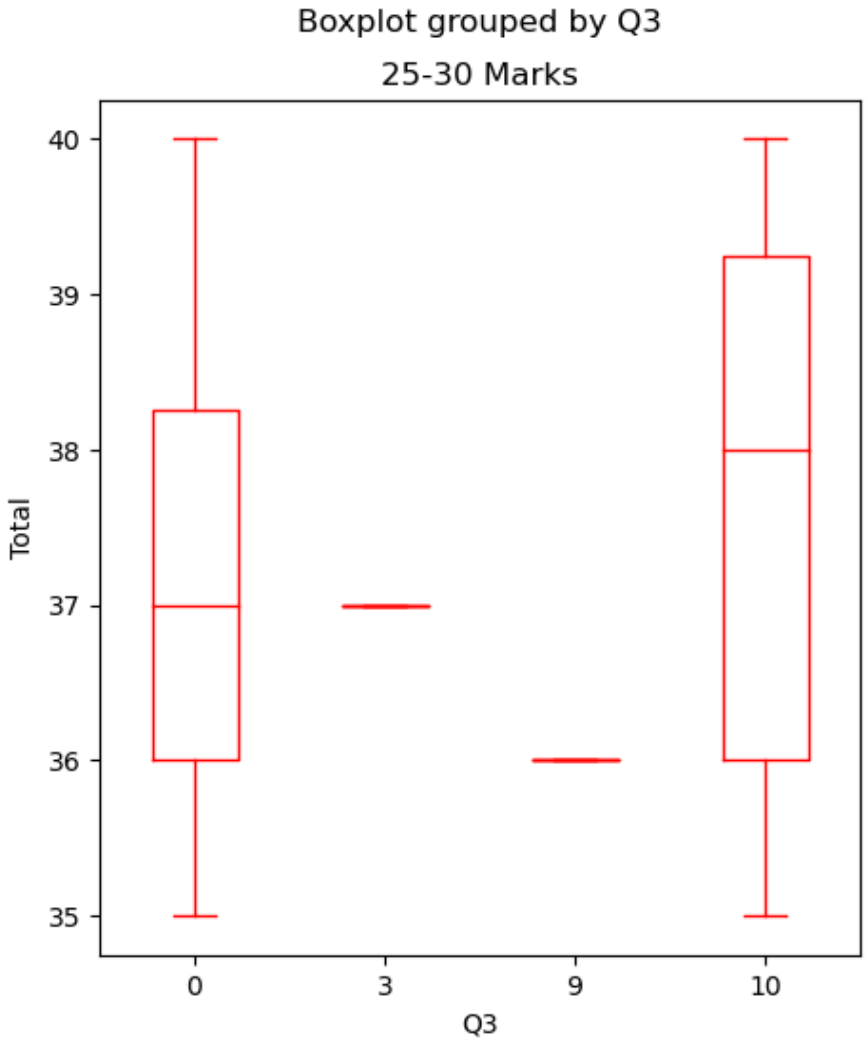
Students who got 10 marks in Q3 and Q6 are plotted

In [40]:
```
c = df[(df['Total'] >= 25) & (df['Total'] <= 30)]
c.boxplot(by='Q1', column =['Total'], grid = False,color='Green',figsize=[5,6])
plt.title("25-30 Marks")
plt.ylabel("Total")
plt.show()
```



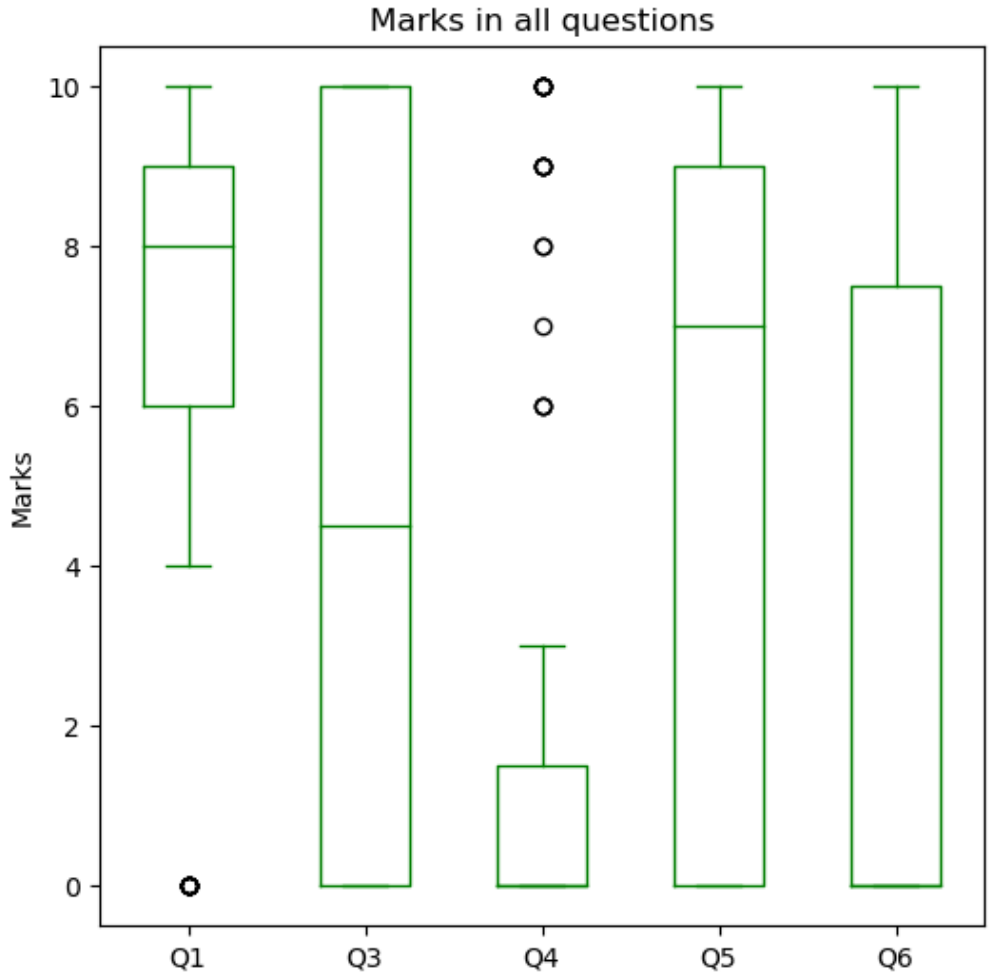
scores between 25 and 30 and scores are distributed across different Q1 groups. 26 to 28 are more

In [41]:
```
The code filters scores between 25 and 30 and uses a green boxplot to show how these s
c.boxplot(by='Q3', column =['Total'], grid = False,color='red',figsize=[5,6])
plt.title("25-30 Marks")
plt.ylabel("Total")
plt.show()
```



Marks in Q3 35-40 and the maximum marks are 36 to 39

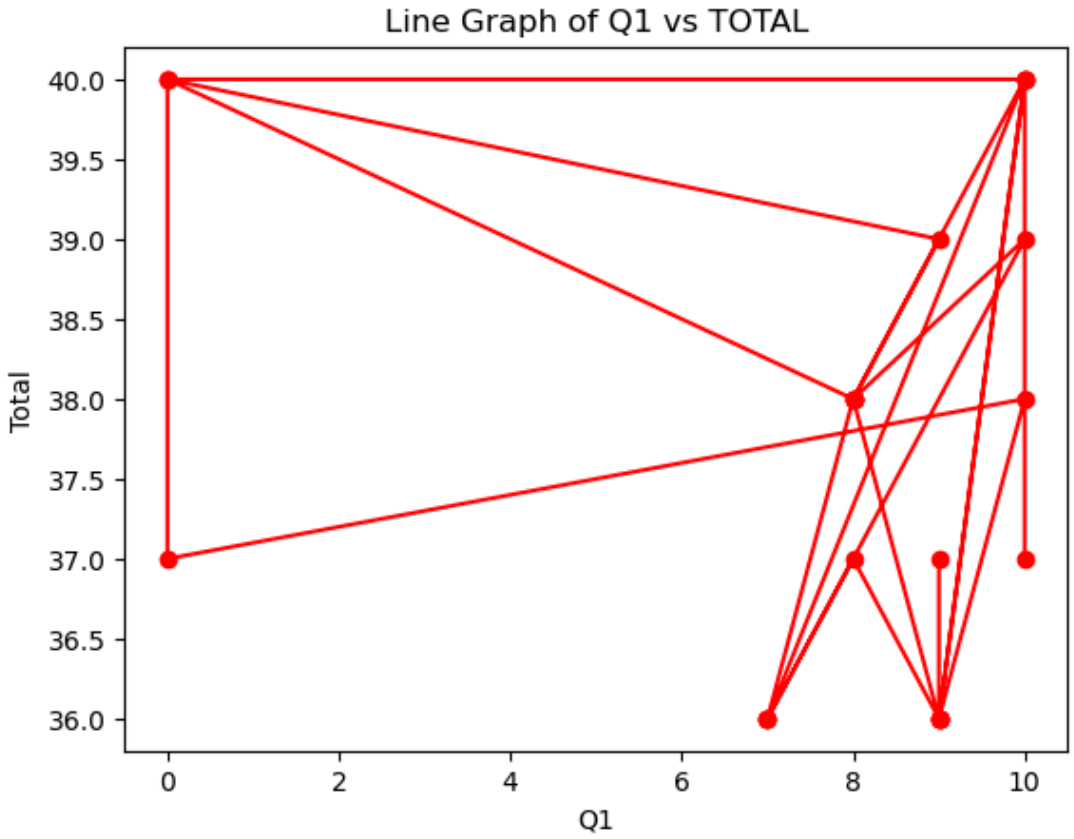
In [44]:
```
import matplotlib.pyplot as plt
df[['Q1', 'Q3', 'Q4', 'Q5', 'Q6']].boxplot(grid=False, color='Green', figsize=[6,6])
plt.title("Marks in all questions")
plt.ylabel("Marks")
plt.show()
```



Marks in all questions the Q4 question students got less marks compared to all answers

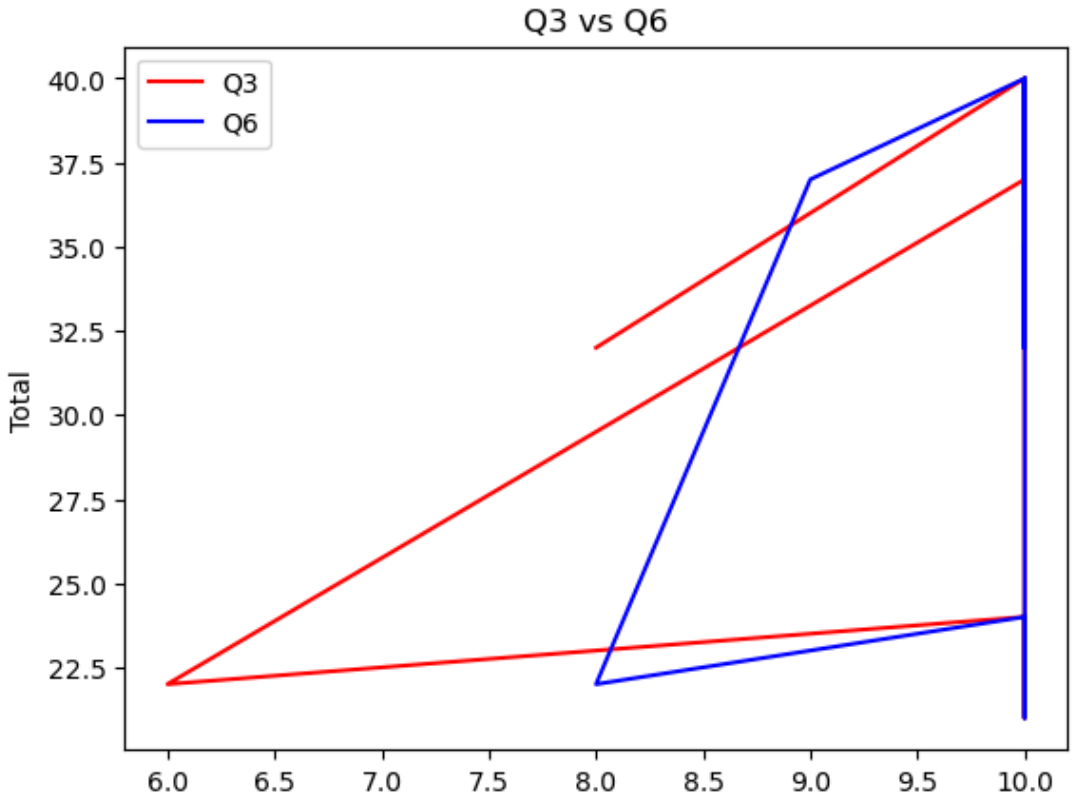
In [51]:

```
filtered_data = df[df['Total'] > 35]
plt.plot(filtered_data['Q1'], filtered_data['Total'], color='red', marker='o')
plt.title("Line Graph of Q1 vs TOTAL")
plt.xlabel("Q1")
plt.ylabel("Total")
plt.show()
```



Graphs shows that who scored above 35 marks students scored 37 to 40 marks majorly

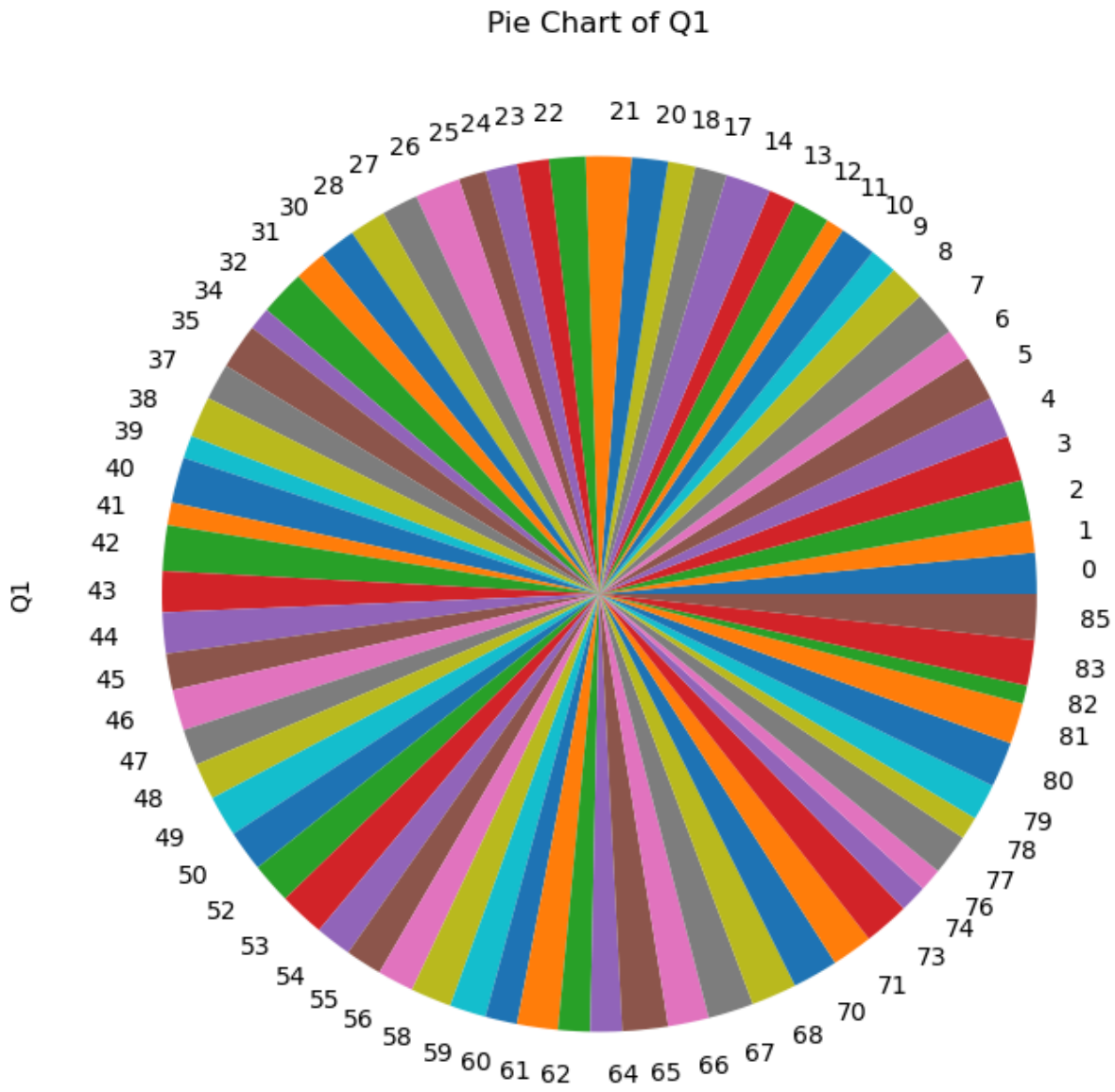
In [46]:
```
filtered_data = df[(df['Q3'] > 5) & (df['Q6'] > 5)]
plt.plot(filtered_data['Q3'], filtered_data['Total'], color='red', label='Q3')
plt.plot(filtered_data['Q6'], filtered_data['Total'], color='blue', label='Q6')
plt.title("Q3 vs Q6")
plt.ylabel("Total")
plt.legend()
plt.show()
```



Marks who scored in more than 5 marks in the Q3 and Q6 8 members got the same marks

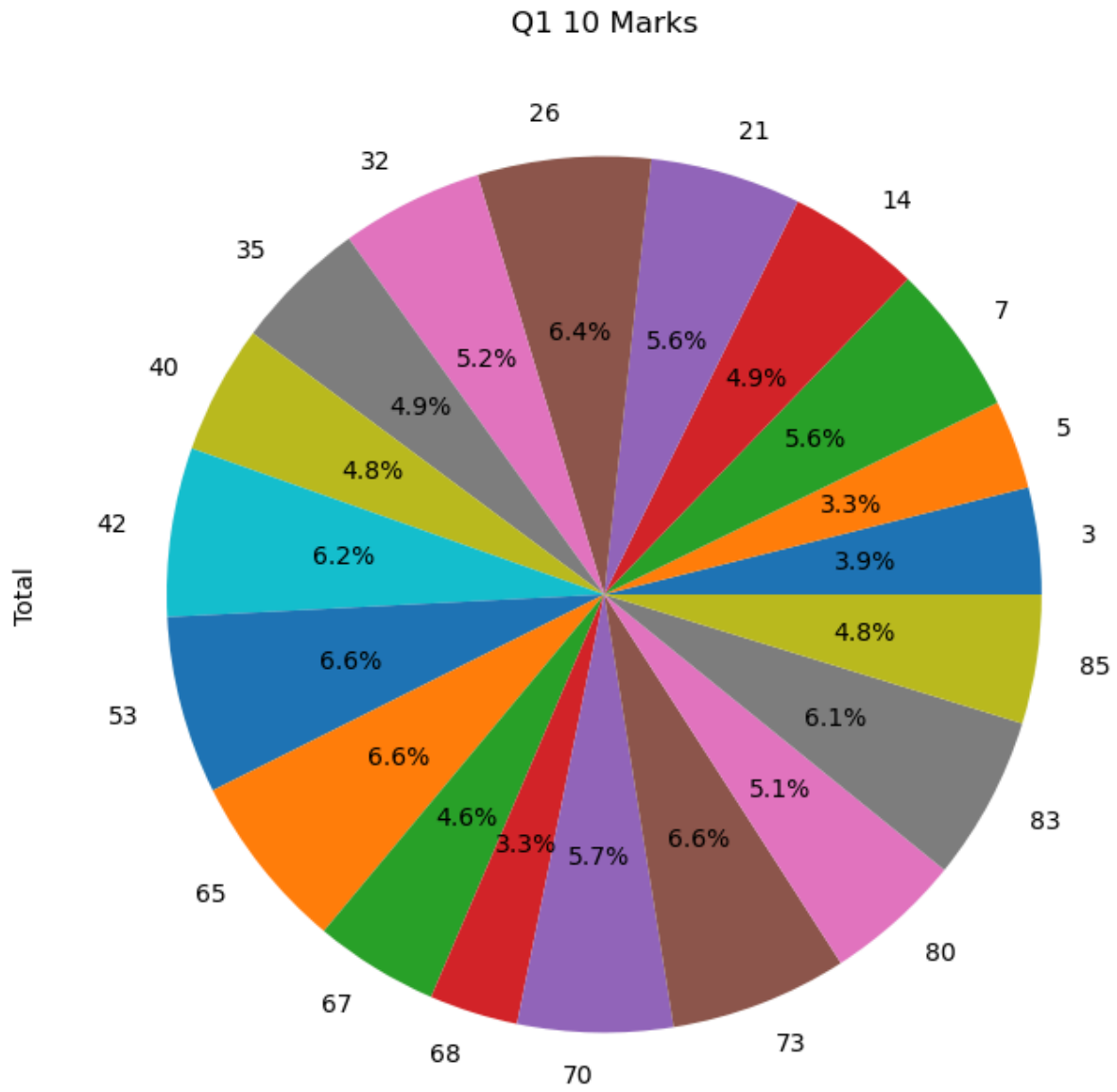
In [48]:
```
df['Q1'].plot(kind='pie',subplots=True,figsize=(8,8))
plt.title("Pie Chart of Q1")
```

Out[48]: Text(0.5, 1.0, 'Pie Chart of Q1')



Q1 Marks distribution

In [49]:
```
df[df['Q1'] == 10]['Total'].plot(kind='pie', figsize=(8,8), autopct='%1.1f%%', legend=
plt.title("Q1 10 Marks")
plt.show()
```



Students who scored 10 marks in the Q1 and percentage

DATASET OBSERVATION

After the dataset was cleaned and organized, several analyses were conducted. The initial step was to visualize the distribution of marks through histograms. The Total marks histogram gave information on how the students performed overall, whether the scores were normally distributed or skewed.

A targeted analysis was done on students who obtained a total score of 40. Filtering methods were used to determine and study their responses, specifically analyzing if there was a consistent pattern of full marks in some questions. A particular emphasis was given to Q6, determining instances where students obtained 10 marks in that section as well as a total score of 40.

Also, the dataset was converted completely by replacing missing values with 0 to avoid gaps in the records. This facilitated easier application of statistical calculations and visualizations. The final dataset was clean and well-formatted, with pertinent data without inconsistencies.

In general, the analysis produced a better visualization of student performance trends, grade patterns, and potential score adjustment. Additional procedures might involve correlation analysis among

In [ ]: