In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
data = pd.read_csv('marketing_data.csv')
```

In [3]:

```
data.head()
```

Out[3]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer
0	1826	1970	Graduation	Divorced	$84,835.00	0	0	6/16/14
1	1	1961	Graduation	Single	$57,091.00	0	0	6/15/14
2	10476	1958	Graduation	Married	$67,267.00	0	1	5/13/14
3	1386	1967	Graduation	Together	$32,474.00	1	1	05-11-2014
4	5371	1989	Graduation	Single	$21,474.00	1	0	04-08-2014

5 rows × 28 columns

In [4]:

```
data.tail()
```

Out[4]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Custor
2235	10142	1976	PhD	Divorced	$66,476.00	0	1	03-07-2(
2236	5263	1977	2n Cycle	Married	$31,056.00	1	0	1/22
2237	22	1976	Graduation	Divorced	$46,310.00	1	0	12-03-2(
2238	528	1978	Graduation	Married	$65,819.00	0	0	11/29
2239	4070	1969	PhD	Married	$94,871.00	0	2	09-01-2(

5 rows × 28 columns

In [5]:

```
data.shape
```

Out[5]:

```
(2240, 28)
```

In [6]:

```
data.columns
```

Out[6]:

```
Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', ' Income ',
       'Kidhome', 'Teenhome', 'Dt_Customer', 'Recency', 'MntWines',
       'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProduct
s',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Response', 'Complain', 'Country'],
      dtype='object')
```

In [7]:

```
data.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

```
data.isnull().sum()
```

Out[8]:

```
ID                     0
Year_Birth             0
Education              0
Marital_Status         0
 Income               24
Kidhome                0
Teenhome               0
Dt_Customer            0
Recency                0
MntWines               0
MntFruits              0
MntMeatProducts        0
MntFishProducts        0
MntSweetProducts       0
MntGoldProds           0
NumDealsPurchases      0
NumWebPurchases        0
NumCatalogPurchases    0
NumStorePurchases      0
NumWebVisitsMonth      0
AcceptedCmp3           0
AcceptedCmp4           0
AcceptedCmp5           0
AcceptedCmp1           0
AcceptedCmp2           0
Response               0
Complain               0
Country                0
dtype: int64
```

In [9]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 28 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   ID                   2240 non-null   int64 
 1   Year_Birth           2240 non-null   int64 
 2   Education            2240 non-null   object
 3   Marital_Status       2240 non-null   object
 4    Income              2216 non-null   object
 5   Kidhome              2240 non-null   int64 
 6   Teenhome             2240 non-null   int64 
 7   Dt_Customer          2240 non-null   object
 8   Recency              2240 non-null   int64 
 9   MntWines             2240 non-null   int64 
 10  MntFruits            2240 non-null   int64 
 11  MntMeatProducts      2240 non-null   int64 
 12  MntFishProducts      2240 non-null   int64 
 13  MntSweetProducts     2240 non-null   int64 
 14  MntGoldProds         2240 non-null   int64 
 15  NumDealsPurchases    2240 non-null   int64 
 16  NumWebPurchases      2240 non-null   int64 
 17  NumCatalogPurchases  2240 non-null   int64 
 18  NumStorePurchases    2240 non-null   int64 
 19  NumWebVisitsMonth    2240 non-null   int64 
 20  AcceptedCmp3         2240 non-null   int64 
 21  AcceptedCmp4         2240 non-null   int64 
 22  AcceptedCmp5         2240 non-null   int64 
 23  AcceptedCmp1         2240 non-null   int64 
 24  AcceptedCmp2         2240 non-null   int64 
 25  Response             2240 non-null   int64 
 26  Complain             2240 non-null   int64 
 27  Country              2240 non-null   object
dtypes: int64(23), object(5)
memory usage: 490.1+ KB
```

In [10]:

```
data.describe()
```

Out[10]:

	ID	Year_Birth	Kidhome	Teenhome	Recency	MntWines	M
count	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	224(
mean	5592.159821	1968.805804	0.444196	0.506250	49.109375	303.935714	26
std	3246.662198	11.984069	0.538398	0.544538	28.962453	336.597393	39
min	0.000000	1893.000000	0.000000	0.000000	0.000000	0.000000	(
25%	2828.250000	1959.000000	0.000000	0.000000	24.000000	23.750000	1
50%	5458.500000	1970.000000	0.000000	0.000000	49.000000	173.500000	8
75%	8427.750000	1977.000000	1.000000	1.000000	74.000000	504.250000	33
max	11191.000000	1996.000000	2.000000	2.000000	99.000000	1493.000000	199

8 rows × 23 columns

In [11]:

```
data.nunique()
```

Out[11]:

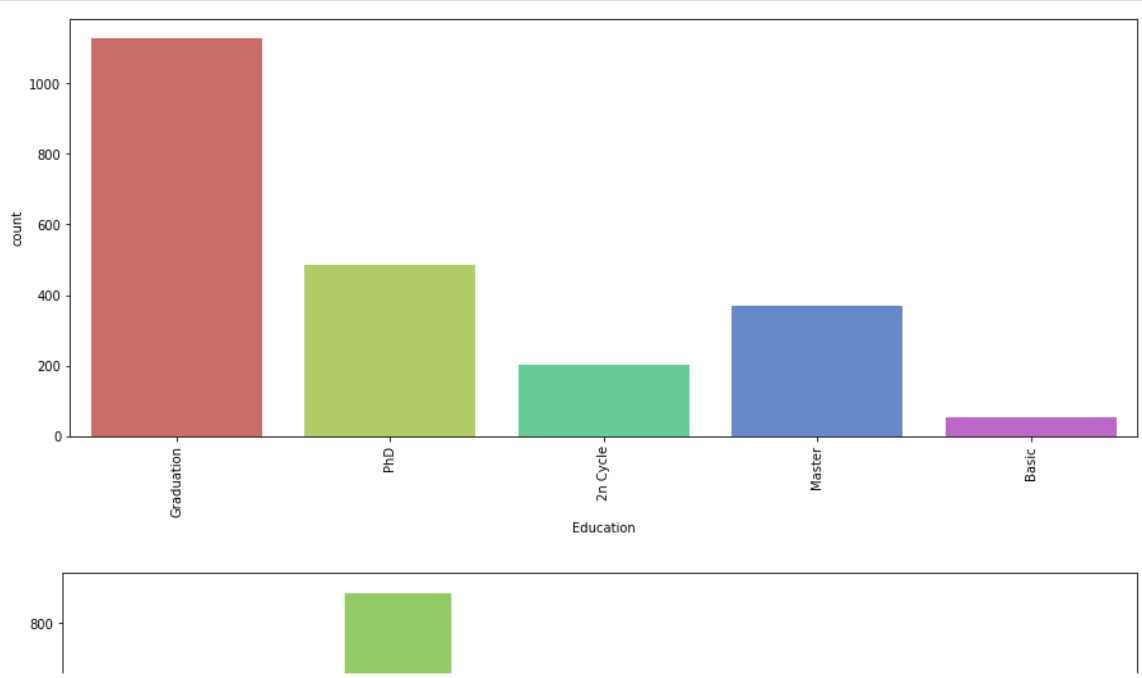
```
ID                     2240
Year_Birth               59
Education                 5
Marital_Status            8
 Income                1974
Kidhome                   3
Teenhome                  3
Dt_Customer             663
Recency                 100
MntWines                776
MntFruits               158
MntMeatProducts         558
MntFishProducts         182
MntSweetProducts        177
MntGoldProds            213
NumDealsPurchases        15
NumWebPurchases          15
NumCatalogPurchases      14
NumStorePurchases        14
NumWebVisitsMonth        16
AcceptedCmp3              2
AcceptedCmp4              2
AcceptedCmp5              2
AcceptedCmp1              2
AcceptedCmp2              2
Response                  2
Complain                  2
Country                   8
dtype: int64
```

In [12]:

```
data_cat = data[['Education', 'Marital_Status', 'Kidhome', 'Teenhome', 'AcceptedCmp3', 'Acc
                 'AcceptedCmp1', 'AcceptedCmp2', 'Response', 'Complain', 'Country']]
```

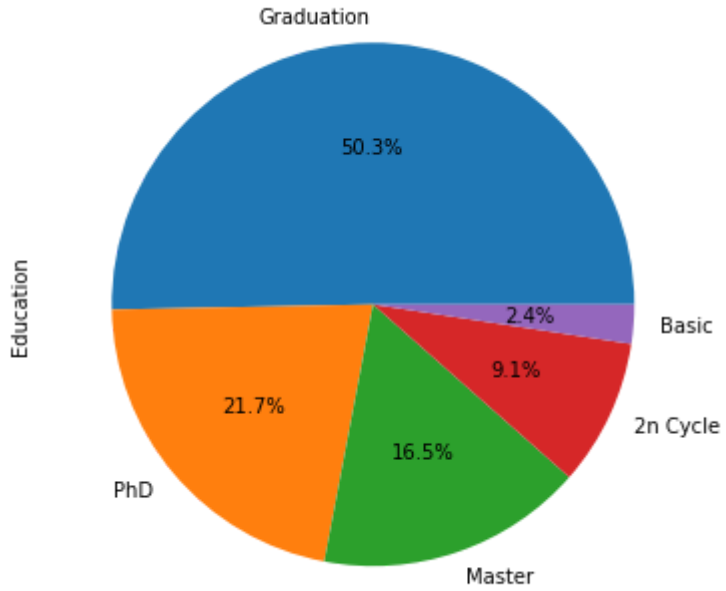
In [13]:

```
for i in data_cat.columns:
    plt.figure(figsize=(15,6))
    sns.countplot(data_cat[i], data = data_cat, palette = 'hls')
    plt.xticks(rotation = 90)
    plt.show()
```



In [14]:

```
for i in data_cat.columns:
    plt.figure(figsize=(15,6))
    data_cat[i].value_counts().plot(kind = 'pie', autopct = '%1.1f%%')
    plt.xticks(rotation = 90)
    plt.show()
```

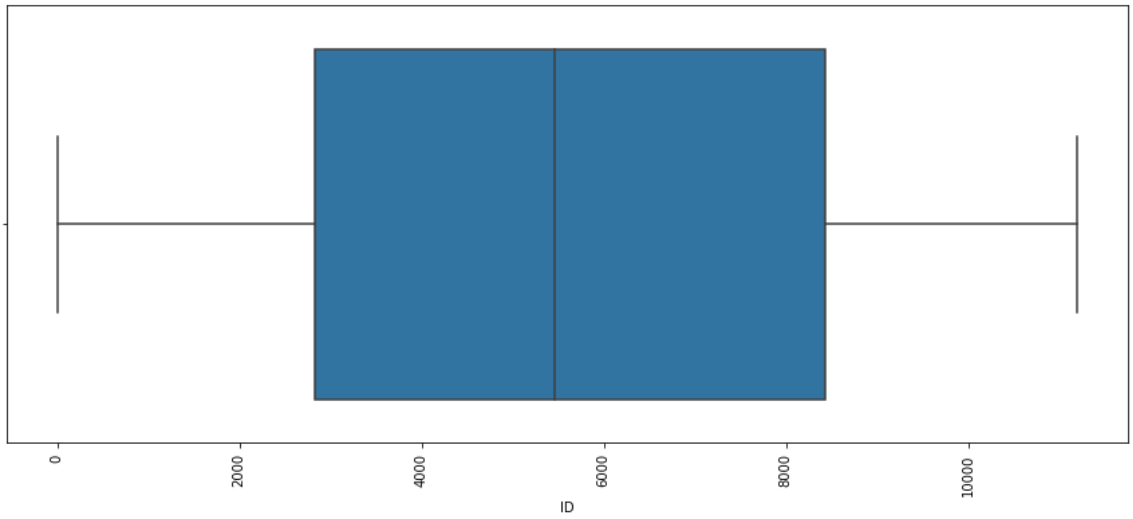


In [15]:

```
data_num = data[['ID', 'Year_Birth','Kidhome', 'Teenhome', 'Recency', 'MntWines','MntFruits
                 'MntFishProducts', 'MntSweetProducts','MntGoldProds', 'NumDealsPurchases',
                 'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth','AcceptedC
                 'AcceptedCmp5', 'AcceptedCmp1','AcceptedCmp2', 'Response', 'Complain']]
```

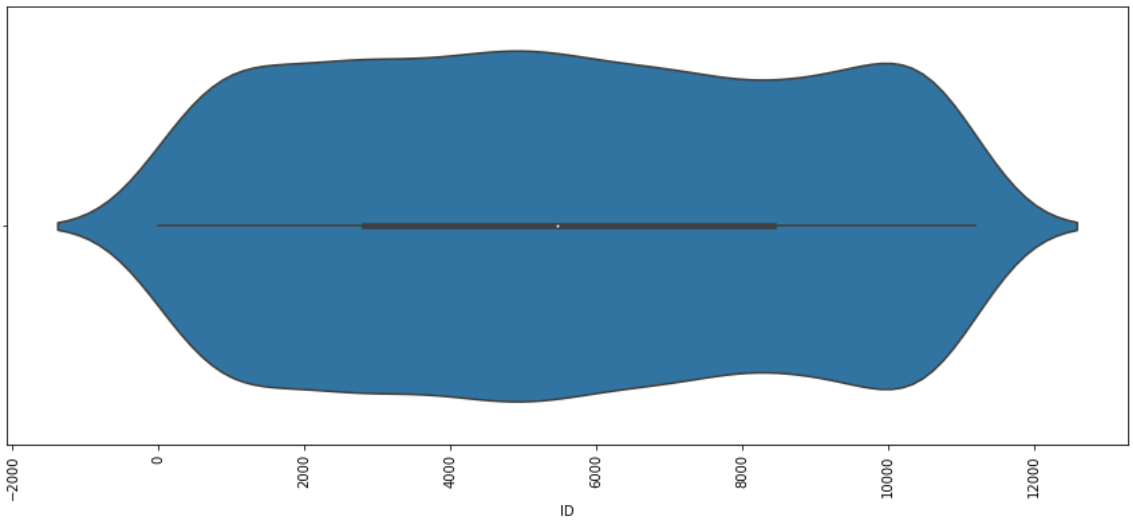
In [16]:

```
for i in data_num.columns:
    plt.figure(figsize=(15,6))
    sns.boxplot(data_num[i])
    plt.xticks(rotation = 90)
    plt.show()
```



In [21]:

```
for i in data_num.columns:
    plt.figure(figsize=(15,6))
    sns.violinplot(data_num[i])
    plt.xticks(rotation = 90)
    plt.show()
```



In [17]:

```
data.rename(columns={' Income ' : 'Income'}, inplace=True)
data['Income']=data["Income"].str.replace("$","")
data['Income']=data["Income"].str.replace(",","")
data['Income']=data["Income"].astype(float)
```

In [18]:

```
data['Income'].head()
```

Out[18]:

```
0    84835.0
1    57091.0
2    67267.0
3    32474.0
4    21474.0
Name: Income, dtype: float64
```

In [19]:

```
data['Income'].tail()
```

Out[19]:

```
2235    66476.0
2236    31056.0
2237    46310.0
2238    65819.0
2239    94871.0
Name: Income, dtype: float64
```

In [24]:

```
data['Income'].isnull().sum()/len(data)*100
```

Out[24]:

```
1.0714285714285714
```

In [25]:

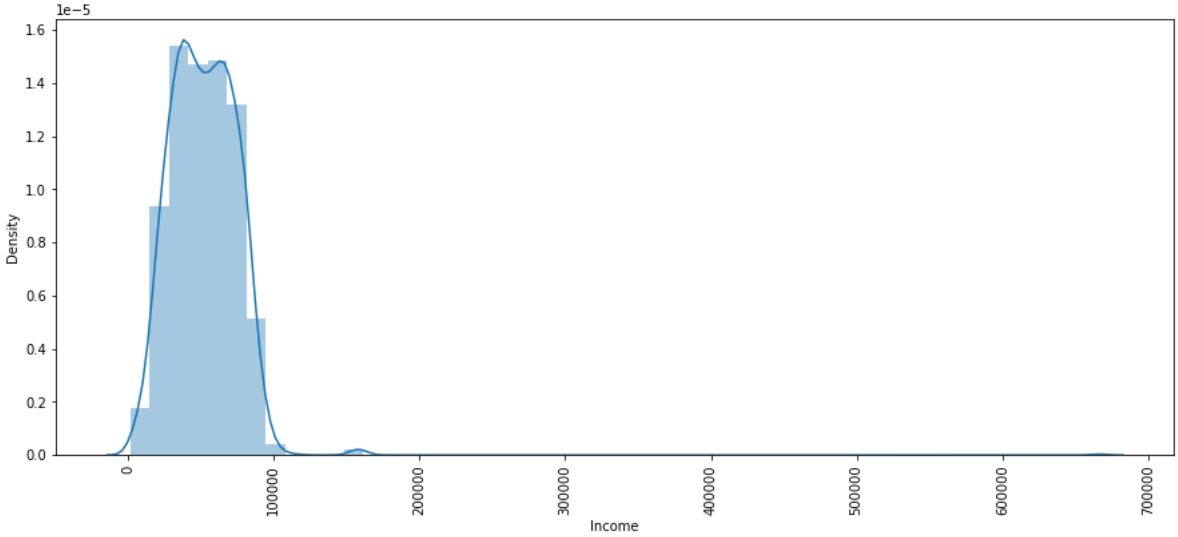
```
data['Income']
```

Out[25]:

```
0       84835.0
1       57091.0
2       67267.0
3       32474.0
4       21474.0
         ...
2235    66476.0
2236    31056.0
2237    46310.0
2238    65819.0
2239    94871.0
Name: Income, Length: 2240, dtype: float64
```

In [26]:

```
plt.figure(figsize=(15,6))
sns.distplot(data['Income'])
plt.xticks(rotation = 90)
plt.show()
```



In [27]:

```
data['Income']= data['Income'].fillna(data['Income'].median())
data['Income'].isna().sum()
```

Out[27]:

```
0
```

In [28]:

```
data['Dt_Customer'] = pd.to_datetime(data['Dt_Customer'])
```

In [29]:

```
data.info()
```

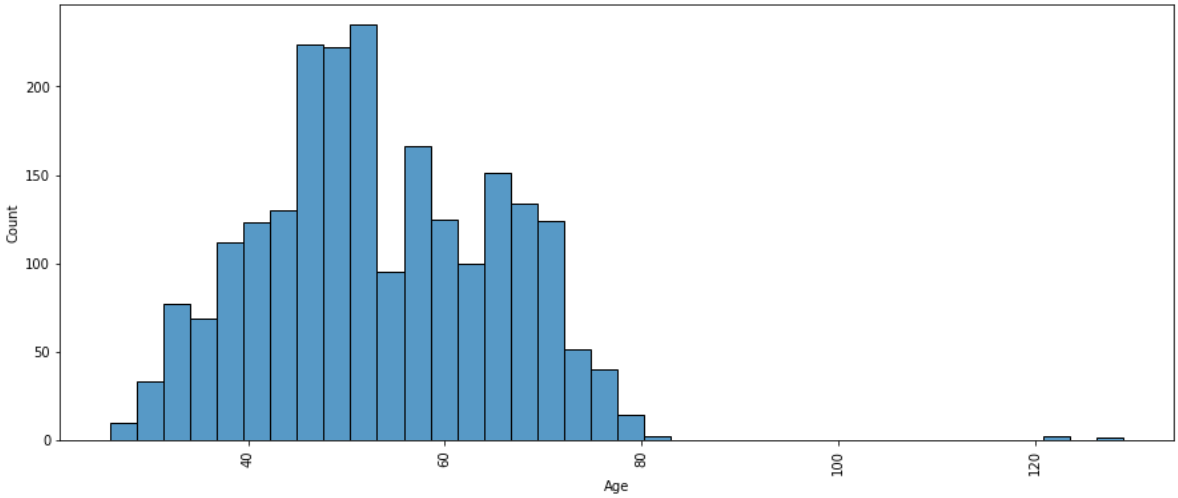
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 28 columns):
 #   Column               Non-Null Count  Dtype         
---  ------               --------------  -----         
 0   ID                   2240 non-null   int64         
 1   Year_Birth           2240 non-null   int64         
 2   Education            2240 non-null   object        
 3   Marital_Status       2240 non-null   object        
 4   Income               2240 non-null   float64       
 5   Kidhome              2240 non-null   int64         
 6   Teenhome             2240 non-null   int64         
 7   Dt_Customer          2240 non-null   datetime64[ns]
 8   Recency              2240 non-null   int64         
 9   MntWines             2240 non-null   int64         
 10  MntFruits            2240 non-null   int64         
 11  MntMeatProducts      2240 non-null   int64         
 12  MntFishProducts      2240 non-null   int64         
 13  MntSweetProducts     2240 non-null   int64         
 14  MntGoldProds         2240 non-null   int64         
 15  NumDealsPurchases    2240 non-null   int64         
 16  NumWebPurchases      2240 non-null   int64         
 17  NumCatalogPurchases  2240 non-null   int64         
 18  NumStorePurchases    2240 non-null   int64         
 19  NumWebVisitsMonth    2240 non-null   int64         
 20  AcceptedCmp3         2240 non-null   int64         
 21  AcceptedCmp4         2240 non-null   int64         
 22  AcceptedCmp5         2240 non-null   int64         
 23  AcceptedCmp1         2240 non-null   int64         
 24  AcceptedCmp2         2240 non-null   int64         
 25  Response             2240 non-null   int64         
 26  Complain             2240 non-null   int64         
 27  Country              2240 non-null   object        
dtypes: datetime64[ns](1), float64(1), int64(23), object(3)
memory usage: 490.1+ KB
```

In [30]:

```
from datetime import date
data['Age']  = date.today().year - data['Year_Birth']
```

In [31]:

```
plt.figure(figsize=(15,6))
sns.histplot(x=data['Age'])
plt.xticks(rotation = 90)
plt.show()
```



In [32]:

```
data2 = data[data['Age']>100]
```

In [33]:

```
data2
```

Out[33]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer
513	11004	1893	2n Cycle	Single	60182.0	0	1	2014-05-17
827	1150	1899	PhD	Together	83532.0	0	0	2013-09-26
2233	7829	1900	2n Cycle	Divorced	36640.0	1	0	2013-09-26

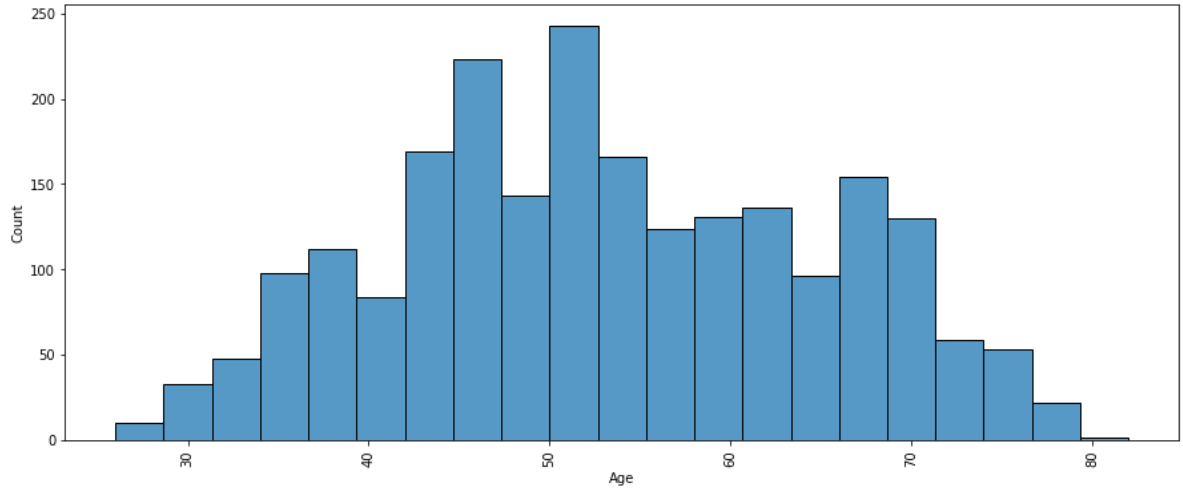
3 rows × 29 columns

In [34]:

```
data = data[data['Age']<100]
data = data[data['Income']<600000]
```

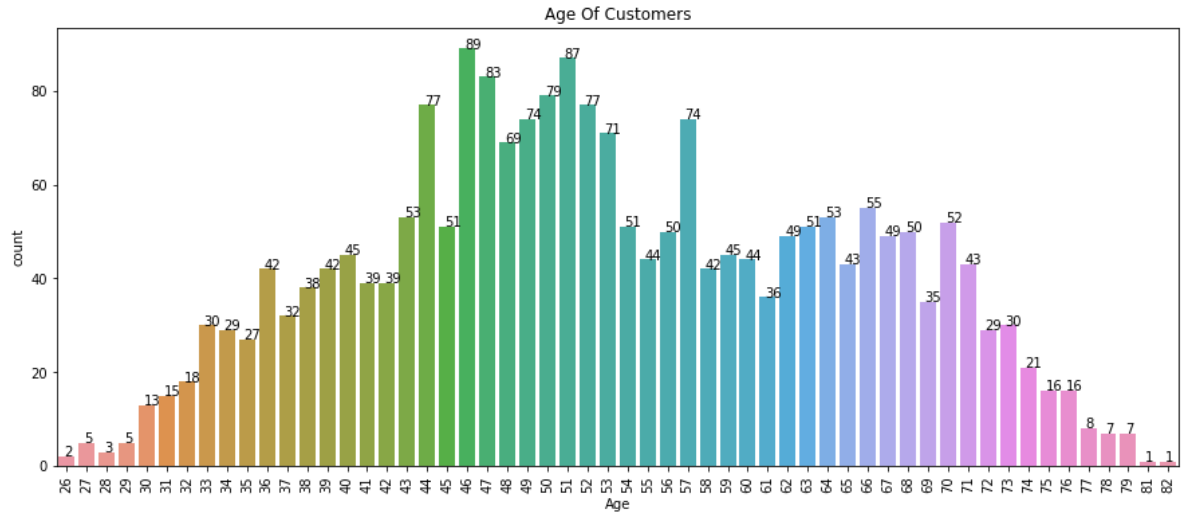
In [35]:

```
plt.figure(figsize=(15,6))
sns.histplot(x=data['Age'])
plt.xticks(rotation = 90)
plt.show()
```



In [37]:

```
plt.figure(figsize=(15, 6))
plt.title('Age Of Customers')
ax = sns.countplot(x=data['Age']);
for p in ax.patches:
    ax.annotate('{:}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
plt.xticks(rotation = 90)
plt.show()
```



In [38]:

```
data['Total Amount'] = data['MntWines']+data['MntFruits']+data['MntMeatProducts']+data['Mnt
```

In [39]:

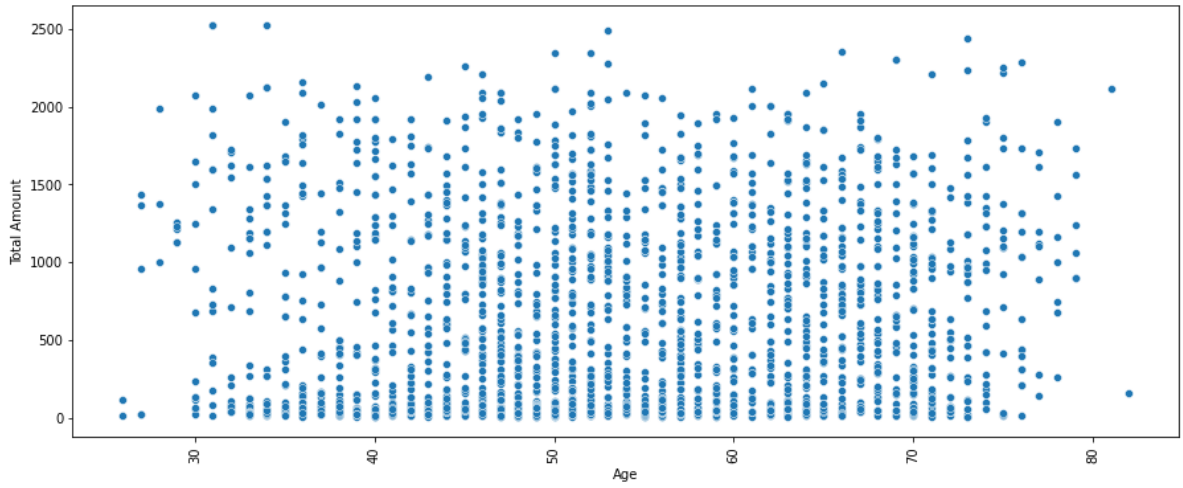
```
data['Total Amount']
```

Out[39]:

```
0       1190
1        577
2        251
3         11
4         91
        ...
2235     689
2236      55
2237     309
2238    1383
2239    1078
Name: Total Amount, Length: 2236, dtype: int64
```

In [40]:

```
plt.figure(figsize=(15,6))
sns.scatterplot(x=data['Age'],y=data['Total Amount'])
plt.xticks(rotation = 90)
plt.show()
```



In [41]:

```
data.filter(regex='Mnt',axis=1)
```

Out[41]:

	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProd
0	189	104	379	111	189	21
1	464	5	64	7	0	3
2	134	11	59	15	2	3
3	10	0	1	0	0	
4	6	16	24	11	0	3
...	...	...	...	...	...	
2235	372	18	126	47	48	7
2236	5	10	13	3	8	1
2237	185	2	88	15	5	1
2238	267	38	701	149	165	6
2239	169	24	553	188	0	14

2236 rows × 6 columns

In [42]:

```
data['PercentWine'] = (data['MntWines']/data['Total Amount'])*100
```

In [43]:

```
data['PercentOthers'] = ((data['MntFruits']+data['MntMeatProducts']+data['MntFishProducts']
```

In [44]:

```
data.filter(regex='Percent|Total|MntWines',axis=1)
```

Out[44]:

	MntWines	Total Amount	PercentWine	PercentOthers
0	189	1190	15.882353	84.117647
1	464	577	80.415945	19.584055
2	134	251	53.386454	46.613546
3	10	11	90.909091	9.090909
4	6	91	6.593407	93.406593
...	...	...	...	...
2235	372	689	53.991292	46.008708
2236	5	55	9.090909	90.909091
2237	185	309	59.870550	40.129450
2238	267	1383	19.305857	80.694143
2239	169	1078	15.677180	84.322820

2236 rows × 4 columns

In [45]:

```
data['Total_Purchase']= data['NumWebPurchases']+data['NumCatalogPurchases']+data['NumStoreP
```

In [46]:

```
data['Total_Purchase']
```

Out[46]:

```
0       14
1       17
2       10
3        3
4        6
        ..
2235    18
2236     4
2237    12
2238    19
2239    17
Name: Total_Purchase, Length: 2236, dtype: int64
```

In [47]:

```
data.filter(regex='WebP|Catalog|Store|Total_Purchase',axis=1)
```

Out[47]:

	NumWebPurchases	NumCatalogPurchases	NumStorePurchases	Total_Purchase
0	4	4	6	14
1	7	3	7	17
2	3	2	5	10
3	1	0	2	3
4	3	1	2	6
...	...	...	...	...
2235	5	2	11	18
2236	1	0	3	4
2237	6	1	5	12
2238	5	4	10	19
2239	8	5	4	17

2236 rows × 4 columns

In [48]:

```
data['PergStore'] = (data['NumStorePurchases']/data['Total_Purchase'])*100

data['PergOthers'] = ((data['NumCatalogPurchases']+data['NumWebPurchases'])/data['Total_Pur

data.filter(regex='Store|Total_P|Perg',axis=1)
```

Out[48]:

	NumStorePurchases	Total_Purchase	PergStore	PergOthers
0	6	14	42.857143	57.142857
1	7	17	41.176471	58.823529
2	5	10	50.000000	50.000000
3	2	3	66.666667	33.333333
4	2	6	33.333333	66.666667
...	...	...	...	...
2235	11	18	61.111111	38.888889
2236	3	4	75.000000	25.000000
2237	5	12	41.666667	58.333333
2238	10	19	52.631579	47.368421
2239	4	17	23.529412	76.470588

2236 rows × 4 columns

In [49]:

```
campaigns = pd.DataFrame(data.iloc[:,20:25].sum(), columns=['amount']).reset_index()
```

In [50]:

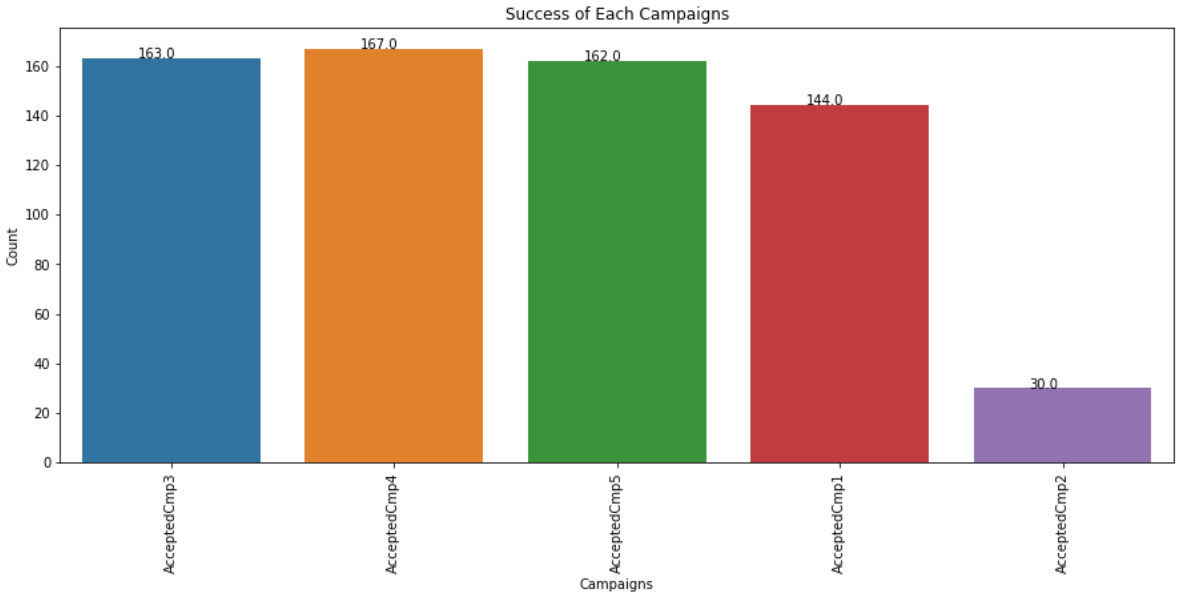
```
campaigns
```

Out[50]:

	index	amount
0	AcceptedCmp3	163
1	AcceptedCmp4	167
2	AcceptedCmp5	162
3	AcceptedCmp1	144
4	AcceptedCmp2	30

In [52]:

```
plt.figure(figsize=(15,6))
plt.title('Success of Each Campaigns')
yx = sns.barplot(x='index',y='amount',data=campaigns);
plt.xlabel('Campaigns')
plt.ylabel('Count')
for p in yx.patches:
    yx.annotate('{:}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
plt.xticks(rotation = 90)
plt.show()
```



In [53]:

```
revenue = pd.DataFrame(data.iloc[:,9:15].sum(),columns=['rev_generated']).reset_index()
```

In [54]:

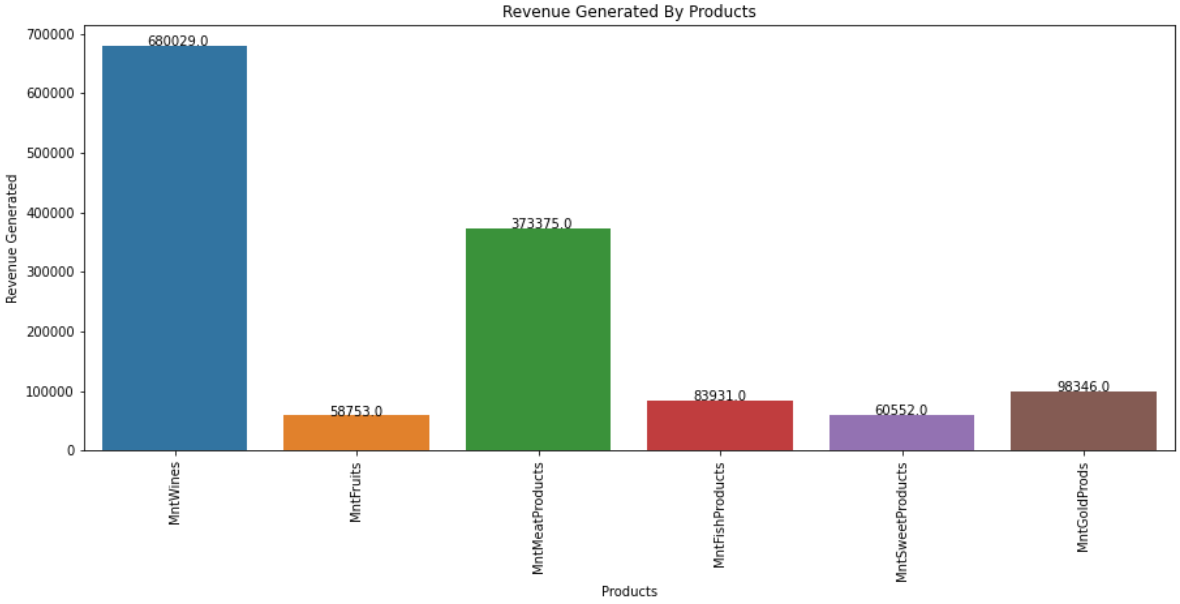
```
revenue
```

Out[54]:

	index	rev_generated
0	MntWines	680029
1	MntFruits	58753
2	MntMeatProducts	373375
3	MntFishProducts	83931
4	MntSweetProducts	60552
5	MntGoldProds	98346

In [55]:

```
plt.figure(figsize=(15,6))
plt.title("Revenue Generated By Products")
yyx= sns.barplot(x='index',y='rev_generated',data=revenue);
plt.xlabel('Products')
plt.ylabel('Revenue Generated')
for p in yyx.patches:
    yyx.annotate('{:}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
plt.xticks(rotation = 90)
plt.show()
```



In [56]:

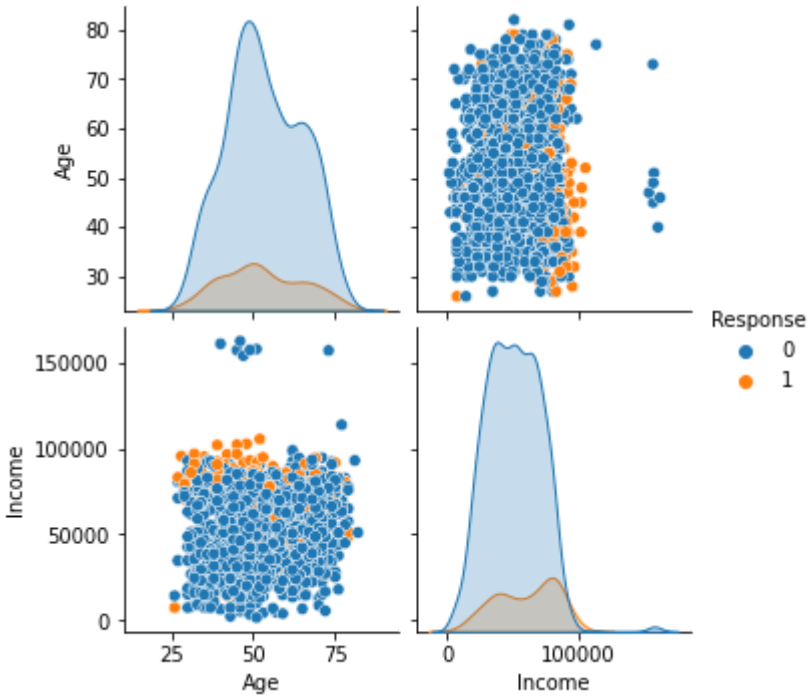
```
plt.figure(figsize=(15,6))
sns.relplot(x='Age',y='Income',hue='Response',data=data);
plt.title('Age v/s Income');
plt.xticks(rotation = 90)
plt.show()
```

```
<Figure size 1080x432 with 0 Axes>
```

In [58]:

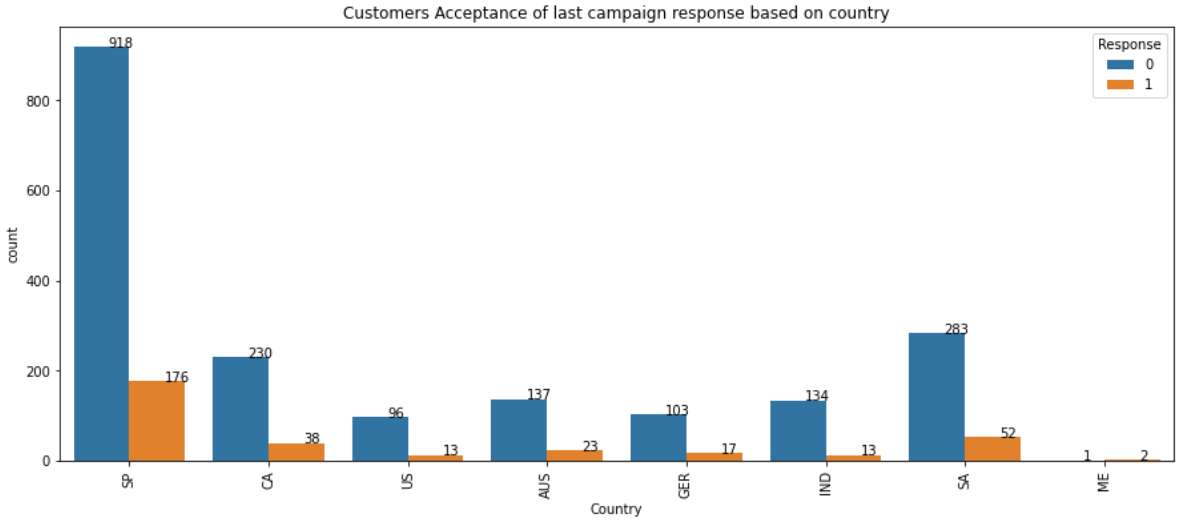
```
plt.figure(figsize=(15,6))
sns.pairplot(vars=['Age','Income'],hue='Response',data=data);
plt.xticks(rotation = 90)
plt.show()
```

```
<Figure size 1080x432 with 0 Axes>
```



In [59]:

```
plt.figure(figsize=(15,6))
cs = sns.countplot(x='Country',hue='Response',data=data)
plt.title('Customers Acceptance of last campaign response based on country');
for p in cs.patches:
    cs.annotate('{:}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
plt.xticks(rotation = 90)
plt.show()
```



In [60]:

```
data['Dependents']=data['Kidhome']+data['Teenhome']
```

In [61]:

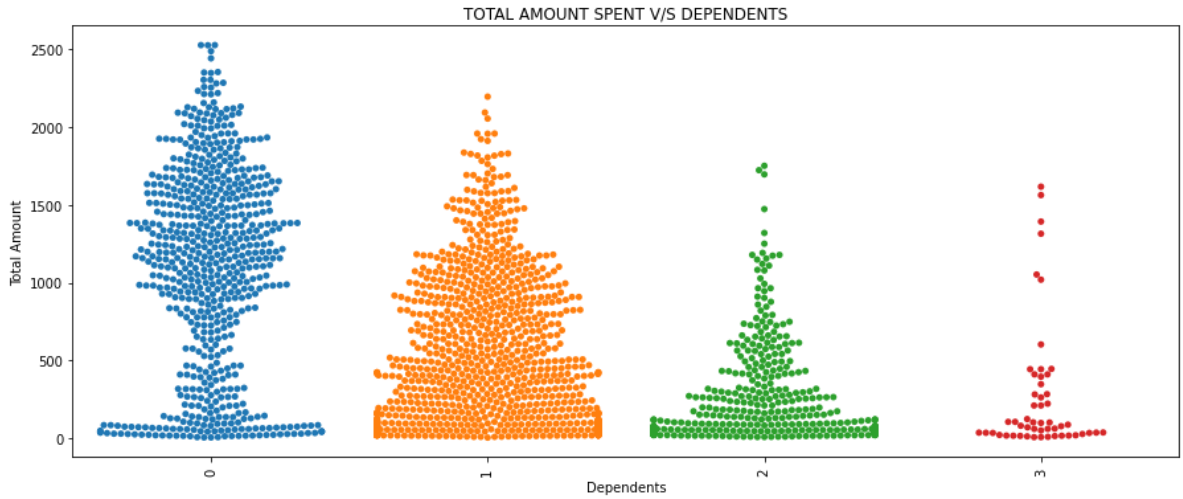
```
data['Dependents']
```

Out[61]:

```
0       0
1       0
2       1
3       2
4       1
       ..
2235    1
2236    1
2237    1
2238    0
2239    2
Name: Dependents, Length: 2236, dtype: int64
```

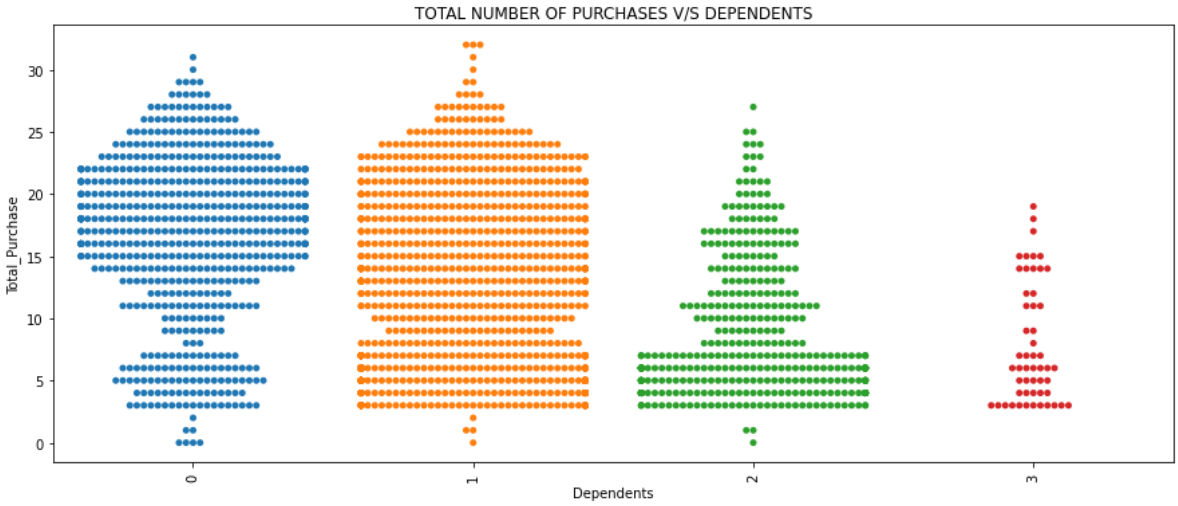
In [63]:

```
plt.figure(figsize=(15,6))
sns.swarmplot(x='Dependents',y='Total Amount',data=data);
plt.title("TOTAL AMOUNT SPENT V/S DEPENDENTS");
plt.xticks(rotation = 90)
plt.show()
```



In [64]:

```
plt.figure(figsize=(15, 6))
sns.swarmplot(x='Dependents',y='Total_Purchase',data=data);
plt.title('TOTAL NUMBER OF PURCHASES V/S DEPENDENTS');
plt.xticks(rotation = 90)
plt.show()
```



In [65]:

```
corrmat = data.corr()
corrmat
```

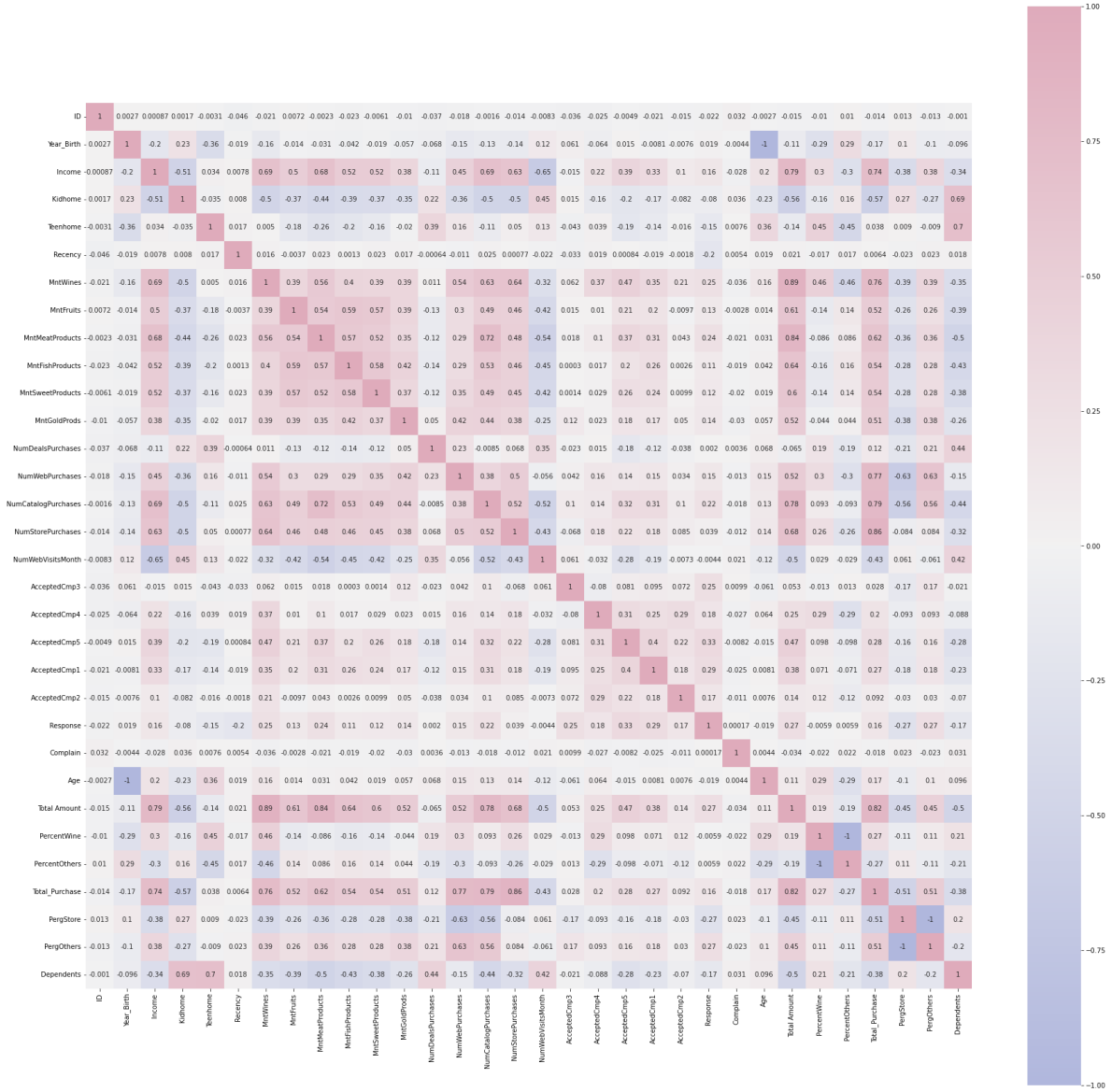
Out[65]:

	ID	Year_Birth	Income	Kidhome	Teenhome	Recency	MntWi
ID	1.000000	0.002659	0.000869	0.001656	-0.003052	-0.046300	-0.020
Year_Birth	0.002659	1.000000	-0.198835	0.233894	-0.363172	-0.019396	-0.162
Income	0.000869	-0.198835	1.000000	-0.510441	0.034315	0.007829	0.686
Kidhome	0.001656	0.233894	-0.510441	1.000000	-0.035339	0.007964	-0.496
Teenhome	-0.003052	-0.363172	0.034315	-0.035339	1.000000	0.016746	0.005
Recency	-0.046300	-0.019396	0.007829	0.007964	0.016746	1.000000	0.016
MntWines	-0.020726	-0.162809	0.686080	-0.496166	0.005047	0.016321	1.000
MntFruits	0.007247	-0.013657	0.504911	-0.372442	-0.176117	-0.003717	0.388
MntMeatProducts	-0.002274	-0.030729	0.684176	-0.436901	-0.261171	0.023446	0.561
MntFishProducts	-0.022903	-0.042359	0.518462	-0.387404	-0.204177	0.001314	0.398
MntSweetProducts	-0.006112	-0.019379	0.518126	-0.370486	-0.162527	0.022796	0.385
MntGoldProds	-0.010339	-0.057419	0.383548	-0.349462	-0.020447	0.017168	0.386
NumDealsPurchases	-0.037393	-0.068286	-0.107169	0.221489	0.388293	-0.000638	0.011
NumWebPurchases	-0.017712	-0.153873	0.450584	-0.361982	0.155649	-0.010776	0.542
NumCatalogPurchases	-0.001592	-0.125285	0.693781	-0.502331	-0.110551	0.025226	0.634
NumStorePurchases	-0.013613	-0.139237	0.628075	-0.500192	0.050177	0.000771	0.642
NumWebVisitsMonth	-0.008256	0.117498	-0.646382	0.447626	0.134636	-0.021850	-0.320
AcceptedCmp3	-0.035823	0.061107	-0.015064	0.014739	-0.042949	-0.033095	0.062
AcceptedCmp4	-0.025150	-0.064261	0.217659	-0.161686	0.038680	0.018779	0.373
AcceptedCmp5	-0.004916	0.015411	0.394308	-0.204918	-0.190383	0.000844	0.471
AcceptedCmp1	-0.021392	-0.008146	0.325254	-0.172435	-0.140426	-0.019367	0.354
AcceptedCmp2	-0.014970	-0.007621	0.104062	-0.081760	-0.015715	-0.001811	0.206
Response	-0.021596	0.018557	0.161121	-0.080005	-0.154941	-0.198781	0.247
Complain	0.031580	-0.004450	-0.027871	0.036336	0.007553	0.005361	-0.035
Age	-0.002659	-1.000000	0.198835	-0.233894	0.363172	0.019396	0.162
Total Amount	-0.015359	-0.113618	0.789375	-0.556719	-0.138197	0.020842	0.891
PercentWine	-0.010179	-0.294814	0.301223	-0.160375	0.452937	-0.017238	0.463
PercentOthers	0.010179	0.294814	-0.301223	0.160375	-0.452937	0.017238	-0.463
Total_Purchase	-0.013620	-0.173004	0.738647	-0.569099	0.037826	0.006426	0.756
PergStore	0.012832	0.104793	-0.376092	0.270237	0.009040	-0.022960	-0.392
PergOthers	-0.012832	-0.104793	0.376092	-0.270237	-0.009040	0.022960	0.392
Dependents	-0.001024	-0.095512	-0.340550	0.690256	0.698721	0.017826	-0.351

32 rows × 32 columns

In [73]:

```
cmap = sns.diverging_palette(260, -10, s = 50, l = 75, n = 6, as_cmap = True)
plt.figure(figsize=(30, 30))
sns.heatmap(corrmat, cmap = cmap, annot = True, square = True)
plt.show()
```



In [74]:

```
datacor = data[['Age','Education','Marital_Status','Dependents','Income','Total Amount','To
                'Recency','NumWebVisitsMonth','Response','Complain']]
datacor
```

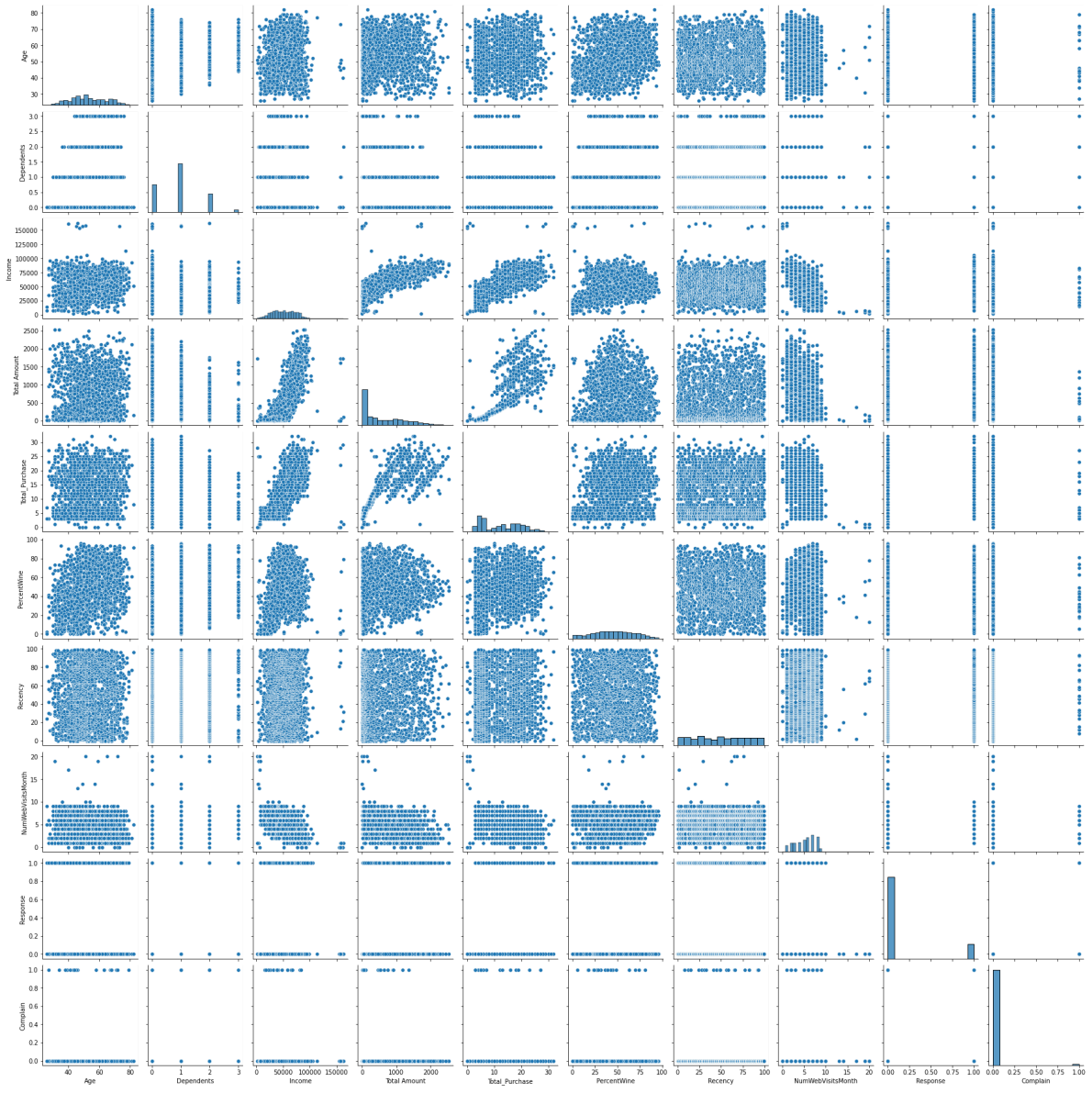
Out[74]:

	Age	Education	Marital_Status	Dependents	Income	Total Amount	Total_Purchase	PercentW
0	52	Graduation	Divorced	0	84835.0	1190	14	15.882
1	61	Graduation	Single	0	57091.0	577	17	80.415
2	64	Graduation	Married	1	67267.0	251	10	53.386
3	55	Graduation	Together	2	32474.0	11	3	90.909
4	33	Graduation	Single	1	21474.0	91	6	6.593
...	...	...	...	...	...	...	...	
2235	46	PhD	Divorced	1	66476.0	689	18	53.991
2236	45	2n Cycle	Married	1	31056.0	55	4	9.090
2237	46	Graduation	Divorced	1	46310.0	309	12	59.870
2238	44	Graduation	Married	0	65819.0	1383	19	19.305
2239	53	PhD	Married	2	94871.0	1078	17	15.677

2236 rows × 12 columns

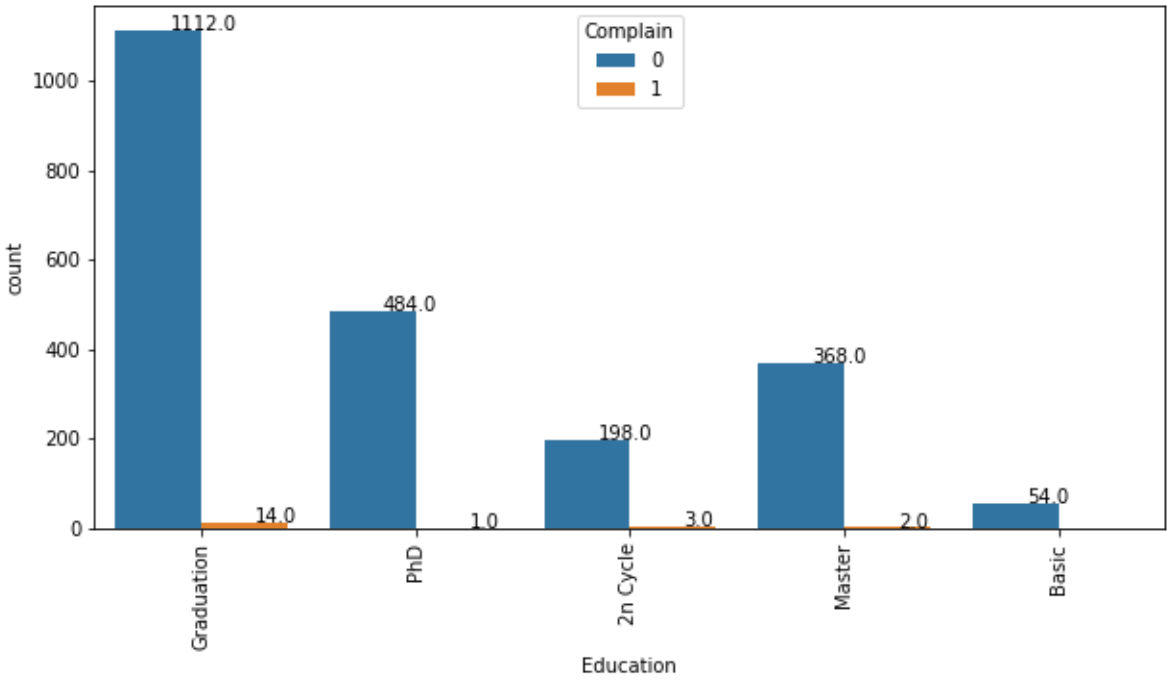
In [75]:

```
sns.pairplot(datacor);
plt.show()
```



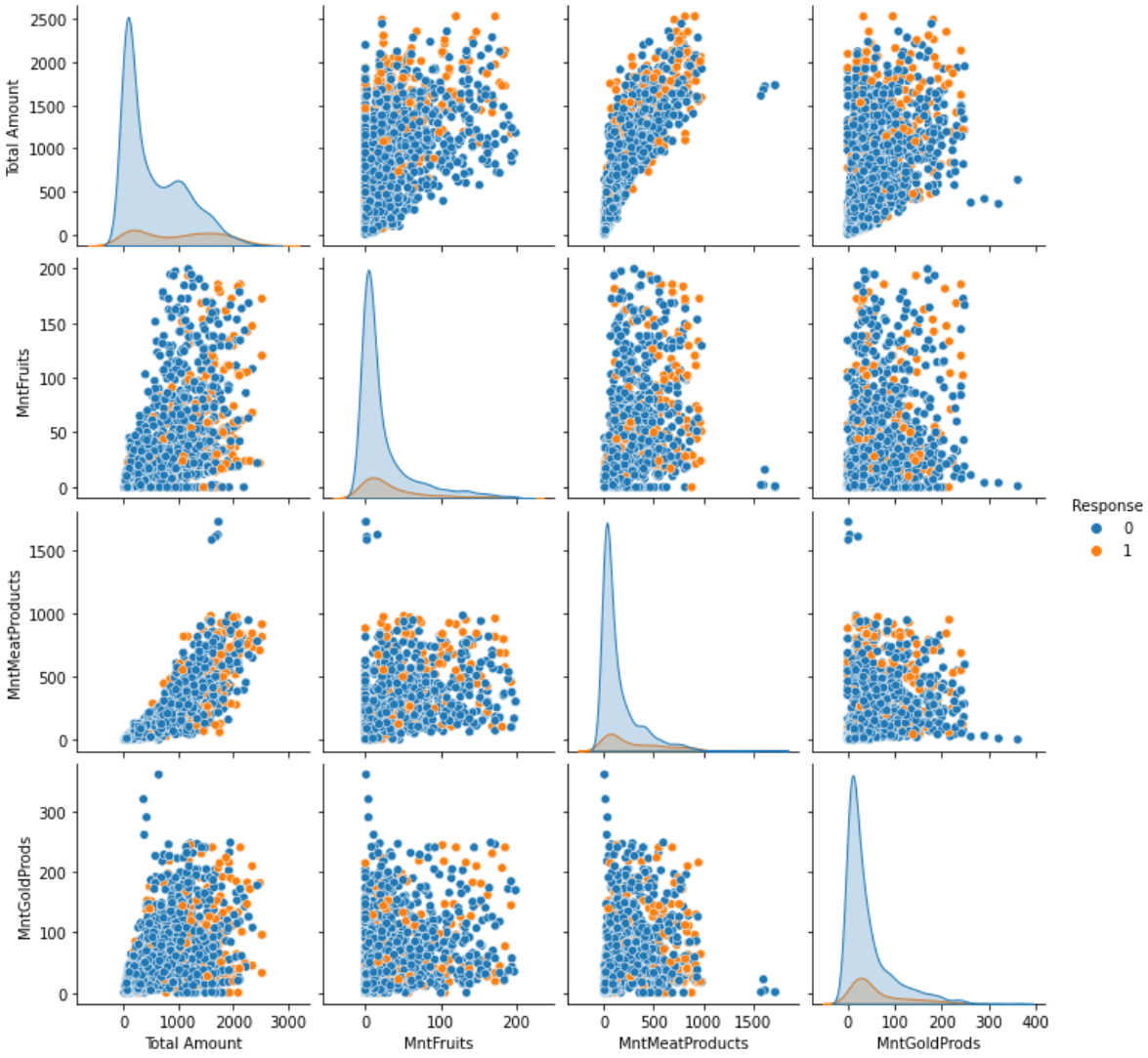
In [76]:

```
plt.figure(figsize=(10,5))
ec = sns.countplot(x='Education',hue='Complain',data=data)
for p in ec.patches:
    ec.annotate('{:}'.format(p.get_height()), (p.get_x()+0.25, p.get_height()+0.01))
plt.xticks(rotation = 90)
plt.show()
```



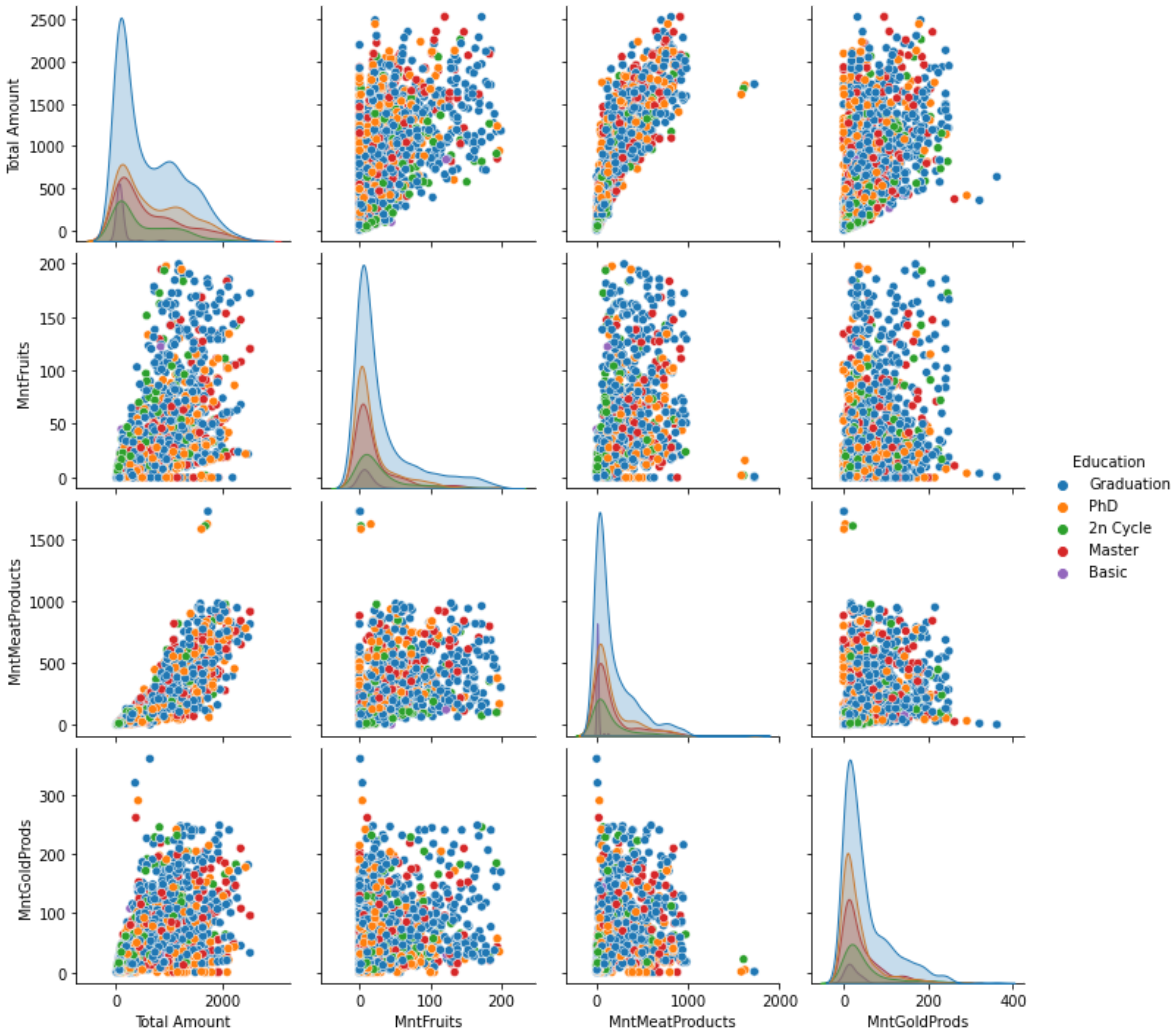
In [77]:

```
sns.pairplot(vars=['Total Amount','MntFruits','MntMeatProducts','MntGoldProds'], hue='Respo
plt.show()
```



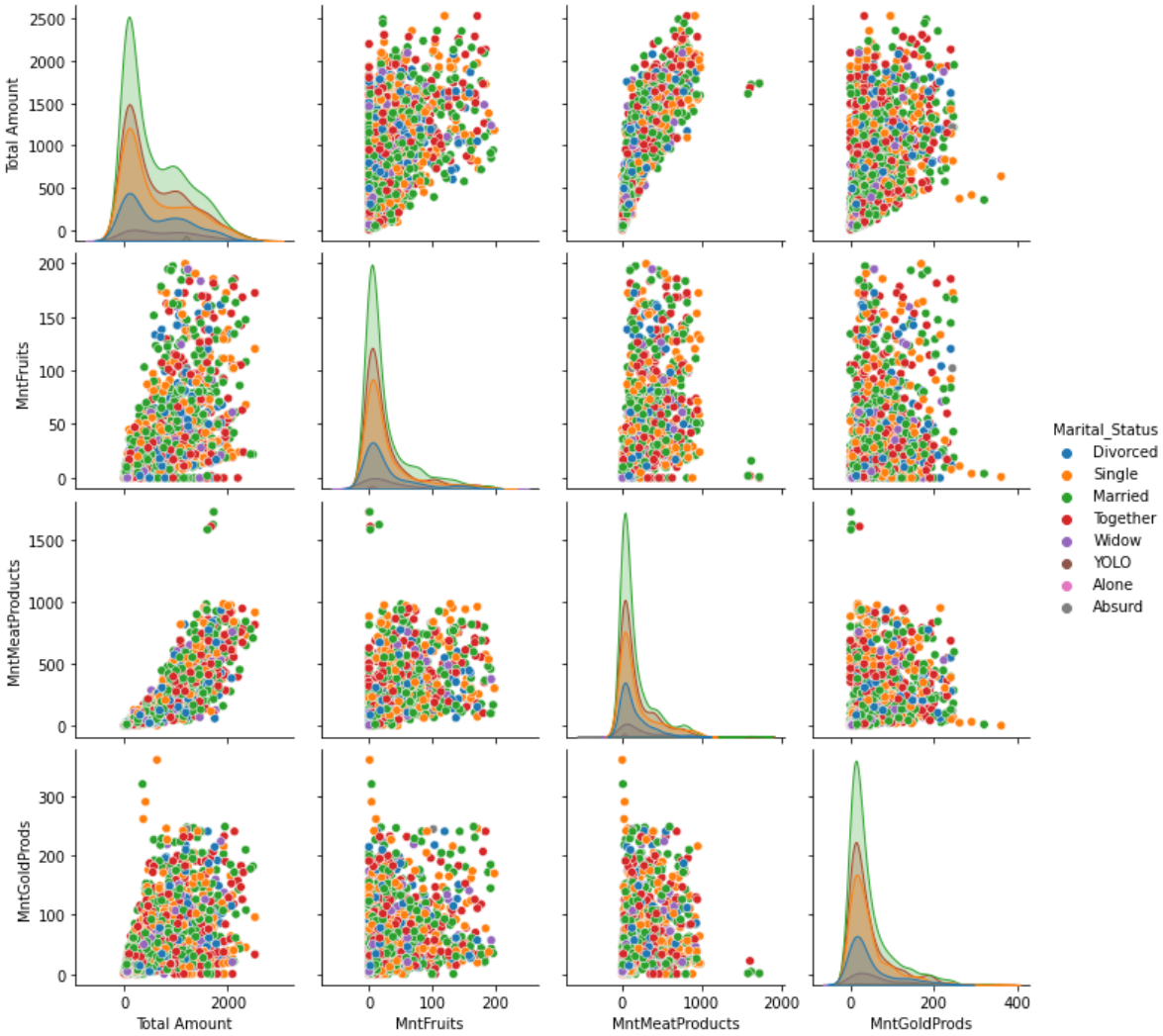
In [78]:

```
sns.pairplot(vars=['Total Amount','MntFruits','MntMeatProducts','MntGoldProds'], hue='Educa
plt.show()
```



In [79]:

```
sns.pairplot(vars=['Total Amount','MntFruits','MntMeatProducts','MntGoldProds'], hue='Marit
plt.show()
```



In [80]:

```
dates = []
for i in data["Dt_Customer"]:
    i = i.date()
    dates.append(i)
#Dates of the newest and oldest recorded customer
print("The newest customer's enrolment date in the records:",max(dates))
print("The oldest customer's enrolment date in the records:",min(dates))
```

```
The newest customer's enrolment date in the records: 2014-06-29
The oldest customer's enrolment date in the records: 2012-07-30
```

In [81]:

```
days = []
d1 = max(dates) #taking it to be the newest customer
for i in dates:
    delta = d1 - i
    days.append(delta)
data["Customer_For"] = days
data["Customer_For"] = pd.to_numeric(data["Customer_For"], errors="coerce")
```

In [84]:

```
print("Total categories in the feature Marital_Status:\n", data["Marital_Status"].value_cou
print("Total categories in the feature Education:\n", data["Education"].value_counts())
```

```
Total categories in the feature Marital_Status:
 Married     864
Together    578
Single      479
Divorced    231
Widow        77
Alone         3
YOLO          2
Absurd        2
Name: Marital_Status, dtype: int64

Total categories in the feature Education:
 Graduation    1126
PhD            485
Master         370
2n Cycle       201
Basic           54
Name: Education, dtype: int64
```

In [85]:

```
data["Age"] = 2022-data["Year_Birth"]
```

In [86]:

```
data["Spent"] = data["MntWines"]+ data["MntFruits"]+ data["MntMeatProducts"]+ data["MntFish
```

In [87]:

```
data["Children"]=data["Kidhome"]+data["Teenhome"]
```

In [88]:

```
data["Living_With"]=data["Marital_Status"].replace({"Married":"Partner", "Together":"Partne
```

In [89]:

```
data["Family_Size"] = data["Living_With"].replace({"Alone": 1, "Partner":2})+ data["Childre
```

In [90]:

```
data["Is_Parent"] = np.where(data.Children> 0, 1, 0)
```

In [91]:

```
data["Education"]=data["Education"].replace({"Basic":"Undergraduate","2n Cycle":"Undergradu
```

In [92]:

```
data=data.rename(columns={"MntWines": "Wines","MntFruits":"Fruits","MntMeatProducts":"Meat"
```

In [94]:

```
to_drop = ["Marital_Status", "Dt_Customer", "Year_Birth", "ID"]
data = data.drop(to_drop, axis=1)
```

In [99]:

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from yellowbrick.cluster import KElbowVisualizer
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt, numpy as np
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import AgglomerativeClustering
from matplotlib.colors import ListedColormap
from sklearn import metrics
import warnings
import sys
if not sys.warnoptions:
    warnings.simplefilter("ignore")
np.random.seed(42)
```

In [100]:

```
s = (data.dtypes == 'object')
object_cols = list(s[s].index)
print("Categorical variables in the dataset:", object_cols)
```

```
Categorical variables in the dataset: ['Education', 'Country', 'Living_Wit
h']
```

In [101]:

```
LE=LabelEncoder()
for i in object_cols:
    data[i]=data[[i]].apply(LE.fit_transform)
print("All features are now numerical")
```

```
All features are now numerical
```

In [102]:

```
ds = data.copy()
cols_del = ['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1','AcceptedCmp2',
ds = ds.drop(cols_del, axis=1)
scaler = StandardScaler()
scaler.fit(ds)
scaled_ds = pd.DataFrame(scaler.transform(ds),columns= ds.columns )
print("All features are now scaled")
```

```
All features are now scaled
```

In [103]:

```
print("Dataframe to be used for further modelling:")
scaled_ds.head()
```

```
Dataframe to be used for further modelling:
```

Out[103]:

	Education	Income	Kidhome	Teenhome	Recency	Wines	Fruits	Meat	
0	-0.89438	1.536081	-0.824939	-0.930615	-1.696543	-0.342115	1.957039	0.939627	1.34
1	-0.89438	0.240037	-0.824939	-0.930615	-1.696543	0.475081	-0.535714	-0.456408	-0.55
2	-0.89438	0.715402	-0.824939	0.905974	-1.696543	-0.505555	-0.384638	-0.478567	-0.41
3	-0.89438	-0.909932	1.032627	0.905974	-1.696543	-0.874036	-0.661610	-0.735615	-0.68
4	-0.89438	-1.423790	1.032627	-0.930615	-1.696543	-0.885922	-0.258741	-0.633682	-0.48

5 rows × 31 columns

In [107]:

```
scaled_ds = scaled_ds.dropna()
```

In [108]:

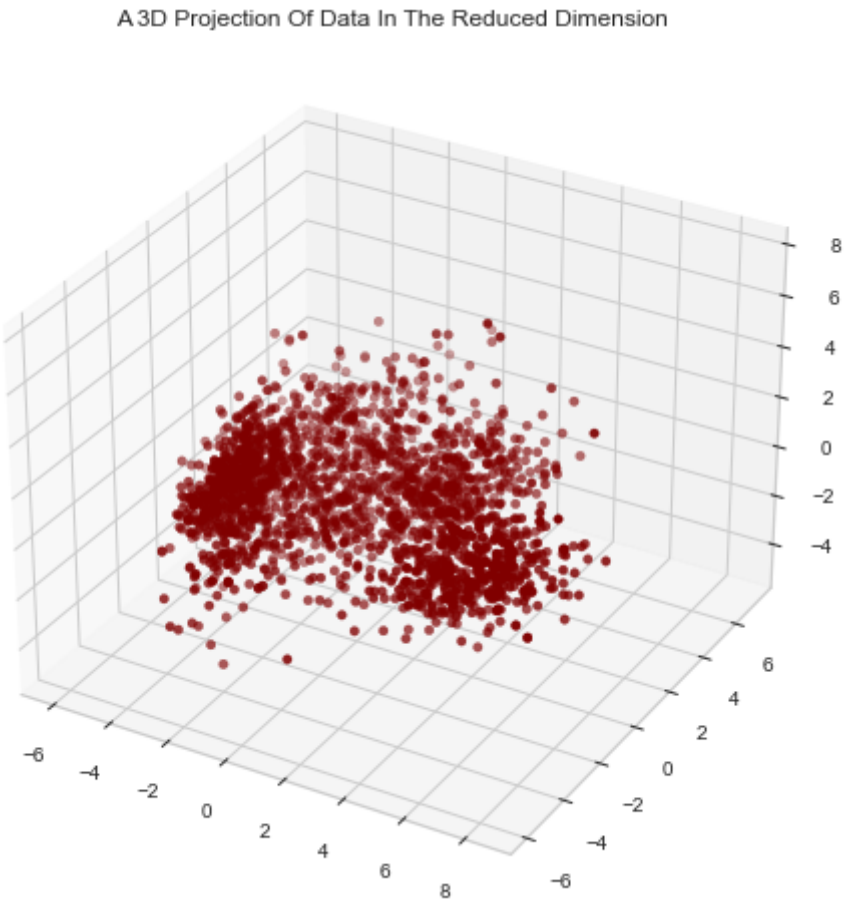
```
pca = PCA(n_components=3)
pca.fit(scaled_ds)
PCA_ds = pd.DataFrame(pca.transform(scaled_ds), columns=(["col1","col2", "col3"]))
PCA_ds.describe().T
```

Out[108]:

	count	mean	std	min	25%	50%	75%	max
col1	2230.0	-1.911774e-17	3.281491	-6.086858	-2.899652	-0.704452	2.729460	8.454845
col2	2230.0	-7.647097e-17	2.101889	-5.887682	-1.647021	-0.066964	1.552679	7.244360
col3	2230.0	-8.284355e-17	1.473437	-4.887482	-0.998291	-0.019217	0.915885	7.716834

In [109]:

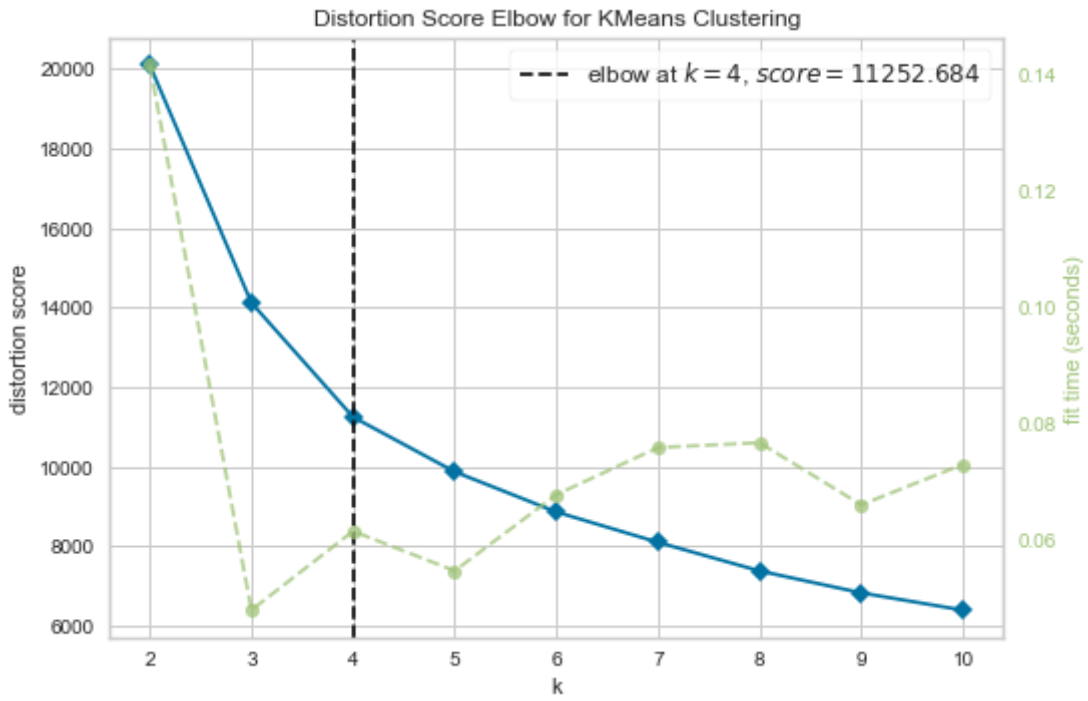
```
x =PCA_ds["col1"]
y =PCA_ds["col2"]
z =PCA_ds["col3"]
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x,y,z, c="maroon", marker="o" )
ax.set_title("A 3D Projection Of Data In The Reduced Dimension")
plt.show()
```



In [110]:

```
print('Elbow Method to determine the number of clusters to be formed:')
Elbow_M = KElbowVisualizer(KMeans(), k=10)
Elbow_M.fit(PCA_ds)
Elbow_M.show()
```

```
Elbow Method to determine the number of clusters to be formed:
```



Out[110]:

```
<AxesSubplot:title={'center':'Distortion Score Elbow for KMeans Clusterin
g'}, xlabel='k', ylabel='distortion score'>
```

In [112]:

```
AC = AgglomerativeClustering(n_clusters=5)
# fit model and predict clusters
yhat_AC = AC.fit_predict(PCA_ds)
PCA_ds["Clusters"] = yhat_AC
```

In [113]:

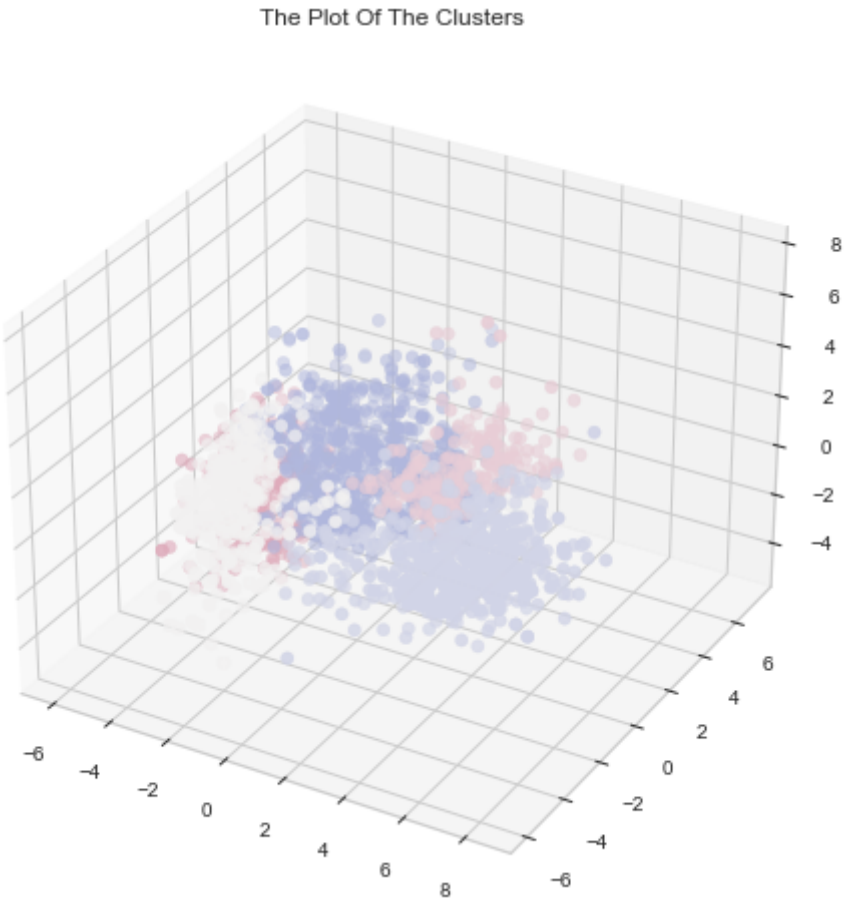
```
yhat_AC
```

Out[113]:

```
array([1, 1, 0, ..., 0, 1, 3], dtype=int64)
```

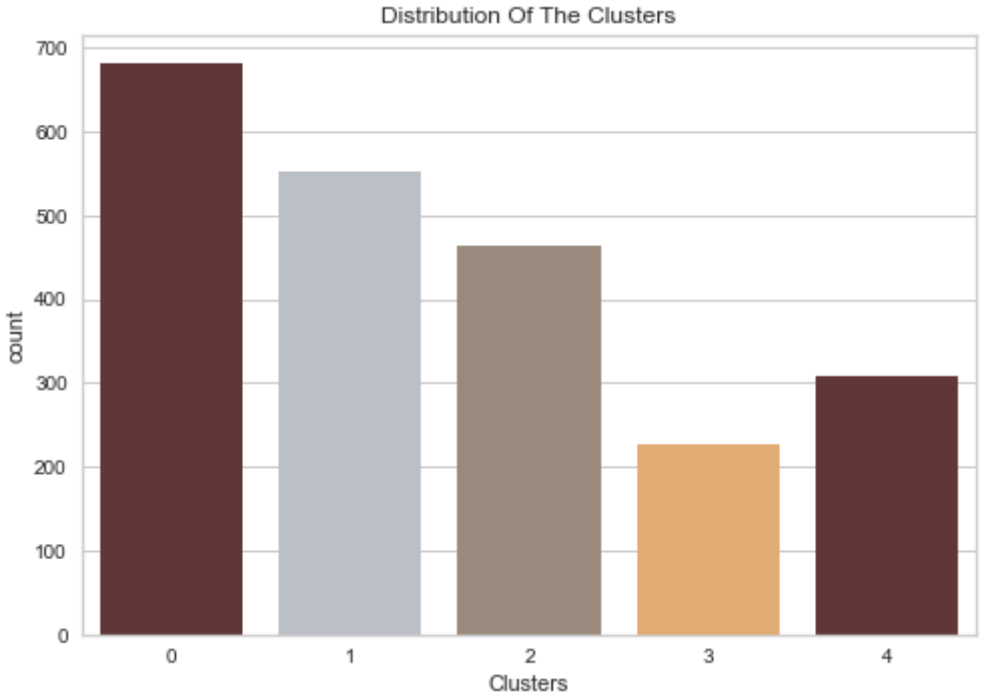
In [114]:

```
fig = plt.figure(figsize=(10,8))
ax = plt.subplot(111, projection='3d', label="bla")
ax.scatter(x, y, z, s=40, c=PCA_ds["Clusters"], marker='o', cmap = cmap )
ax.set_title("The Plot Of The Clusters")
plt.show()
```



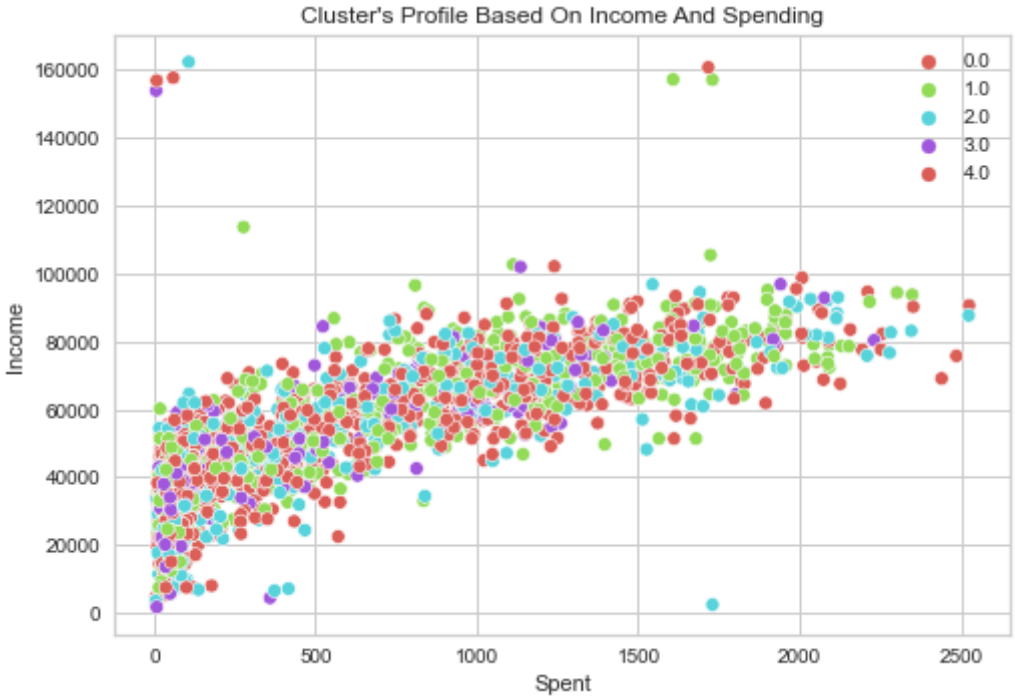
In [116]:

```
pal = ["#682F2F","#B9C0C9", "#9F8A78","#F3AB60"]
pl = sns.countplot(x=PCA_ds["Clusters"], palette= pal)
pl.set_title("Distribution Of The Clusters")
plt.show()
```



In [117]:

```
pl = sns.scatterplot(data = data,x=data["Spent"], y=data["Income"],hue=PCA_ds["Clusters"],
pl.set_title("Cluster's Profile Based On Income And Spending")
plt.legend()
plt.show()
```



In [119]:

```
plt.figure()
pl=sns.swarmplot(x=PCA_ds["Clusters"], y=data["Spent"], color= "#CBEDDD", alpha=0.5 )
pl=sns.boxenplot(x=PCA_ds["Clusters"], y=data["Spent"], palette=pal)
plt.show()
```