1. Load the dataset “bank-additional-full” using pd.read_csv and complete the following tasks with appropriate interpretation:

i. Perform the basic analysis. What kind of insights do they provide?

```
import matplotlib.pyplot as plt
import pandas as pd
import os

os.getcwd()

'd:\\PYTHON\\DATA SCIENCE\\DS1 Record'

bank_df = pd.read_csv("D:/PYTHON/DATA SCIENCE/DATA/bank-additional-
full.csv",sep = ';')

bank_df

       age          job  marital            education  default housing
loan  \
0       56    housemaid  married             basic.4y       no      no
no
1       57     services  married          high.school  unknown      no
no
2       37     services  married          high.school       no     yes
no
3       40       admin.  married             basic.6y       no      no
no
4       56     services  married          high.school       no      no
yes
...    ...          ...      ...                  ...      ...     ...
...
41183   73      retired  married  professional.course       no     yes
no
41184   46  blue-collar  married  professional.course       no      no
no
41185   56      retired  married    university.degree       no     yes
no
41186   44   technician  married  professional.course       no      no
no
41187   74      retired  married  professional.course       no     yes
no

         contact month day_of_week  ...  campaign  pdays  previous  \
0      telephone   may         mon  ...         1    999         0
1      telephone   may         mon  ...         1    999         0
2      telephone   may         mon  ...         1    999         0
```

```
3      telephone   may         mon  ...        1    999        0
4      telephone   may         mon  ...        1    999        0
...          ...   ...         ...  ...      ...    ...      ...
41183   cellular   nov         fri  ...        1    999        0
41184   cellular   nov         fri  ...        1    999        0
41185   cellular   nov         fri  ...        2    999        0
41186   cellular   nov         fri  ...        1    999        0
41187   cellular   nov         fri  ...        3    999        1

          poutcome emp.var.rate  cons.price.idx  cons.conf.idx
euribor3m  \
0      nonexistent          1.1          93.994          -36.4
4.857
1      nonexistent          1.1          93.994          -36.4
4.857
2      nonexistent          1.1          93.994          -36.4
4.857
3      nonexistent          1.1          93.994          -36.4
4.857
4      nonexistent          1.1          93.994          -36.4
4.857
...            ...          ...             ...            ...
...
41183  nonexistent         -1.1          94.767          -50.8
1.028
41184  nonexistent         -1.1          94.767          -50.8
1.028
41185  nonexistent         -1.1          94.767          -50.8
1.028
41186  nonexistent         -1.1          94.767          -50.8
1.028
41187      failure         -1.1          94.767          -50.8
1.028

       nr.employed    y
0           5191.0   no
1           5191.0   no
2           5191.0   no
3           5191.0   no
4           5191.0   no
...            ...  ...
41183       4963.6  yes
41184       4963.6   no
41185       4963.6   no
41186       4963.6  yes
41187       4963.6   no

[41188 rows x 21 columns]

bank_df.shape
```

```
(41188, 21)

bank_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             41188 non-null  int64
 1   job             41188 non-null  object
 2   marital         41188 non-null  object
 3   education       41188 non-null  object
 4   default         41188 non-null  object
 5   housing         41188 non-null  object
 6   loan            41188 non-null  object
 7   contact         41188 non-null  object
 8   month           41188 non-null  object
 9   day_of_week     41188 non-null  object
 10  duration        41188 non-null  int64
 11  campaign        41188 non-null  int64
 12  pdays           41188 non-null  int64
 13  previous        41188 non-null  int64
 14  poutcome        41188 non-null  object
 15  emp.var.rate    41188 non-null  float64
 16  cons.price.idx  41188 non-null  float64
 17  cons.conf.idx   41188 non-null  float64
 18  euribor3m       41188 non-null  float64
 19  nr.employed     41188 non-null  float64
 20  y               41188 non-null  object
dtypes: float64(5), int64(5), object(11)
memory usage: 6.6+ MB

bank_df.head()

   age        job  marital    education  default housing loan
contact  \
0   56  housemaid  married     basic.4y       no      no   no
telephone
1   57   services  married  high.school  unknown      no   no
telephone
2   37   services  married  high.school       no     yes   no
telephone
3   40     admin.  married     basic.6y       no      no   no
telephone
4   56   services  married  high.school       no      no  yes
telephone

  month day_of_week  ...  campaign  pdays  previous     poutcome
emp.var.rate  \
```

```
0   may         mon  ...         1    999         0  nonexistent
1.1
1   may         mon  ...         1    999         0  nonexistent
1.1
2   may         mon  ...         1    999         0  nonexistent
1.1
3   may         mon  ...         1    999         0  nonexistent
1.1
4   may         mon  ...         1    999         0  nonexistent
1.1

   cons.price.idx  cons.conf.idx  euribor3m  nr.employed   y
0          93.994          -36.4      4.857       5191.0  no
1          93.994          -36.4      4.857       5191.0  no
2          93.994          -36.4      4.857       5191.0  no
3          93.994          -36.4      4.857       5191.0  no
4          93.994          -36.4      4.857       5191.0  no

[5 rows x 21 columns]
```

Interpretation:

Basic analysis provides a quick summary of the dataset, including the number of rows and columns, data types, and missing values. It also helps understand key feature distributions, giving an overview of the data's structure and quality

ii. Create a new column named “Conversion” by transforming categorical values in the variable “y” into numerical representations, and why is this transformation important in data analysis?

```
bank_df['conversion']=bank_df['y'].apply(lambda x: 1 if x=='yes'else
0)

bank_df

       age          job  marital            education  default housing
loan  \
0       56    housemaid  married             basic.4y       no      no
no
1       57     services  married          high.school  unknown      no
no
2       37     services  married          high.school       no     yes
no
3       40       admin.  married             basic.6y       no      no
no
4       56     services  married          high.school       no      no
yes
...    ...          ...      ...                  ...      ...     ...
...
```

```
41183   73      retired  married  professional.course      no     yes
no
41184   46  blue-collar  married  professional.course      no      no
no
41185   56      retired  married    university.degree      no     yes
no
41186   44   technician  married  professional.course      no      no
no
41187   74      retired  married  professional.course      no     yes
no

         contact month day_of_week  ...  pdays  previous     poutcome
\
0      telephone   may         mon  ...    999         0  nonexistent

1      telephone   may         mon  ...    999         0  nonexistent

2      telephone   may         mon  ...    999         0  nonexistent

3      telephone   may         mon  ...    999         0  nonexistent

4      telephone   may         mon  ...    999         0  nonexistent

...          ...   ...         ...  ...    ...       ...          ...

41183   cellular   nov         fri  ...    999         0  nonexistent

41184   cellular   nov         fri  ...    999         0  nonexistent

41185   cellular   nov         fri  ...    999         0  nonexistent

41186   cellular   nov         fri  ...    999         0  nonexistent

41187   cellular   nov         fri  ...    999         1      failure


       emp.var.rate cons.price.idx  cons.conf.idx  euribor3m
nr.employed  \
0               1.1         93.994          -36.4      4.857
5191.0
1               1.1         93.994          -36.4      4.857
5191.0
2               1.1         93.994          -36.4      4.857
5191.0
3               1.1         93.994          -36.4      4.857
5191.0
4               1.1         93.994          -36.4      4.857
5191.0
...             ...            ...            ...        ...
...
```

```
41183            -1.1          94.767           -50.8       1.028
4963.6
41184            -1.1          94.767           -50.8       1.028
4963.6
41185            -1.1          94.767           -50.8       1.028
4963.6
41186            -1.1          94.767           -50.8       1.028
4963.6
41187            -1.1          94.767           -50.8       1.028
4963.6

           y  conversion
0         no           0
1         no           0
2         no           0
3         no           0
4         no           0
...      ...         ...
41183    yes           1
41184     no           0
41185     no           0
41186    yes           1
41187     no           0

[41188 rows x 22 columns]
```

Interpretation

Convert "yes" to 1 and "no" to 0 in the y column. This makes it easier for analysis and modeling.

iii. Describe how the Aggregate Conversion Rate is calculated and interpret its significance in the context of the dataset.

Aggregate Conversion Rate

```
bank_df['conversion'].sum(),bank_df.shape[0]
```

```
(np.int64(4640), 41188)
```

```
(bank_df['conversion'].sum()/bank_df['conversion'].count())*100
```

```
np.float64(11.265417111780131)
```

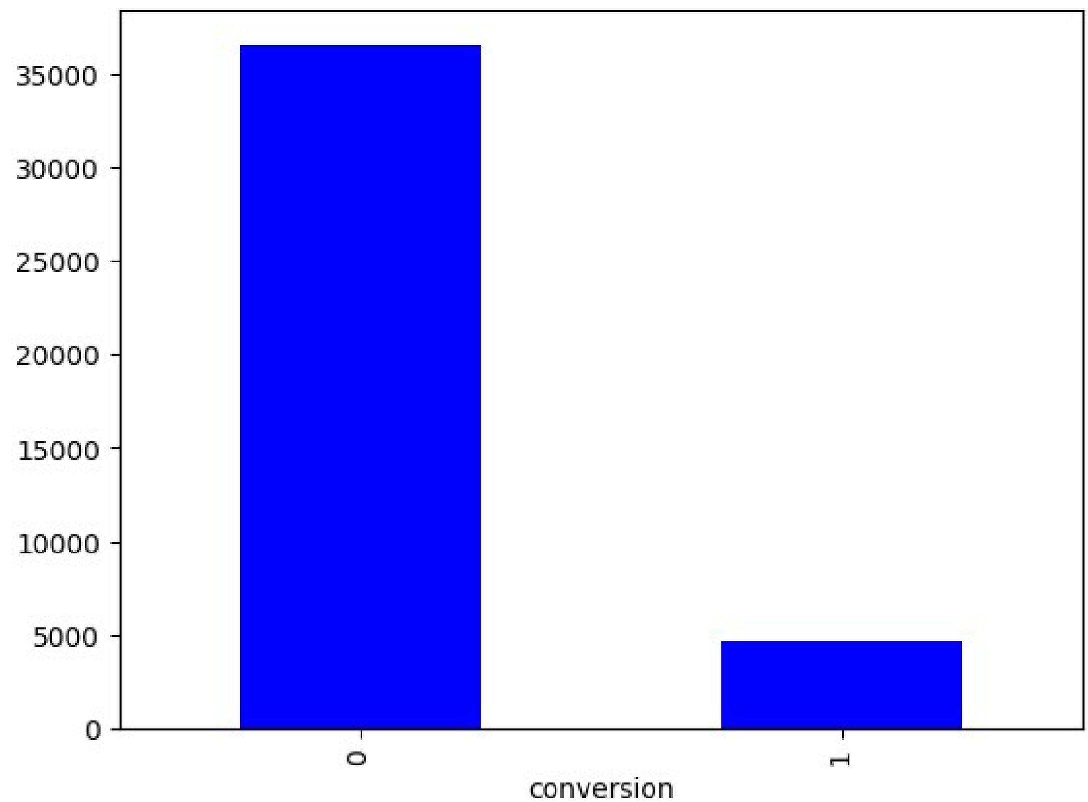
Interpretation

The Aggregate Conversion Rate is the percentage of "yes" responses out of the total entries. It shows the overall success of the campaign.

iv. What is the purpose of plotting the conversion data using a bar chart, and how does the code achieve this visualization?

```
bank_df.groupby('conversion').count()['age'].plot(kind='bar',
    color='blue',)
```

```
<Axes: xlabel='conversion'>
```



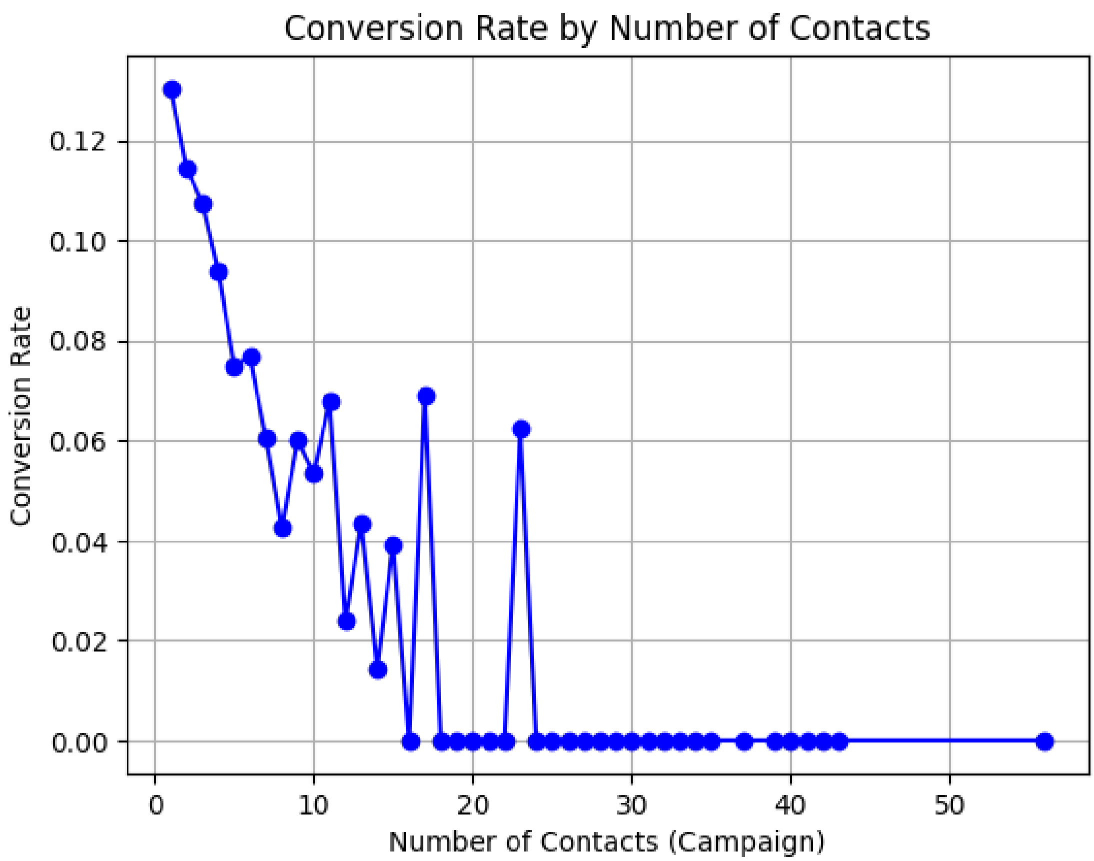
Interpretation:

Plotting conversion data with a bar chart helps compare the number of people in each category. The code groups data by "conversion," counts entries, and displays the results in a green bar chart.

v. How can conversion rates by the number of contacts be calculated and visualized in a dataset

```
conversion_by_contacts = bank_df.groupby('campaign')
['conversion'].mean()
conversion_by_contacts.plot(kind='line', marker='o', color='blue')
plt.title("Conversion Rate by Number of Contacts")
plt.xlabel("Number of Contacts (Campaign)")
plt.ylabel("Conversion Rate")
```

```
plt.grid(True)
plt.show()
```



Interpretation:

Conversion rates by the number of contacts are calculated by grouping the data by "campaign" and taking the average of "conversion" values. The code then visualizes this with a green line chart, adding markers, labels, a title, and a grid for clarity.

vi. How are age groups created using a lambda function in a dataset, and why is grouping data into age ranges beneficial for analysis?

```
bank_df['AgeGroup'] = bank_df['age'].apply(lambda x: '18-30' if 18 <=
x <= 30 else
                                            ('31-40' if 31 <= x <= 40
else
                                            ('41-50' if 41 <= x <= 50
else
                                            ('51-60' if 51 <= x <= 60
else '60+'))))

bank_df['AgeGroup']
```

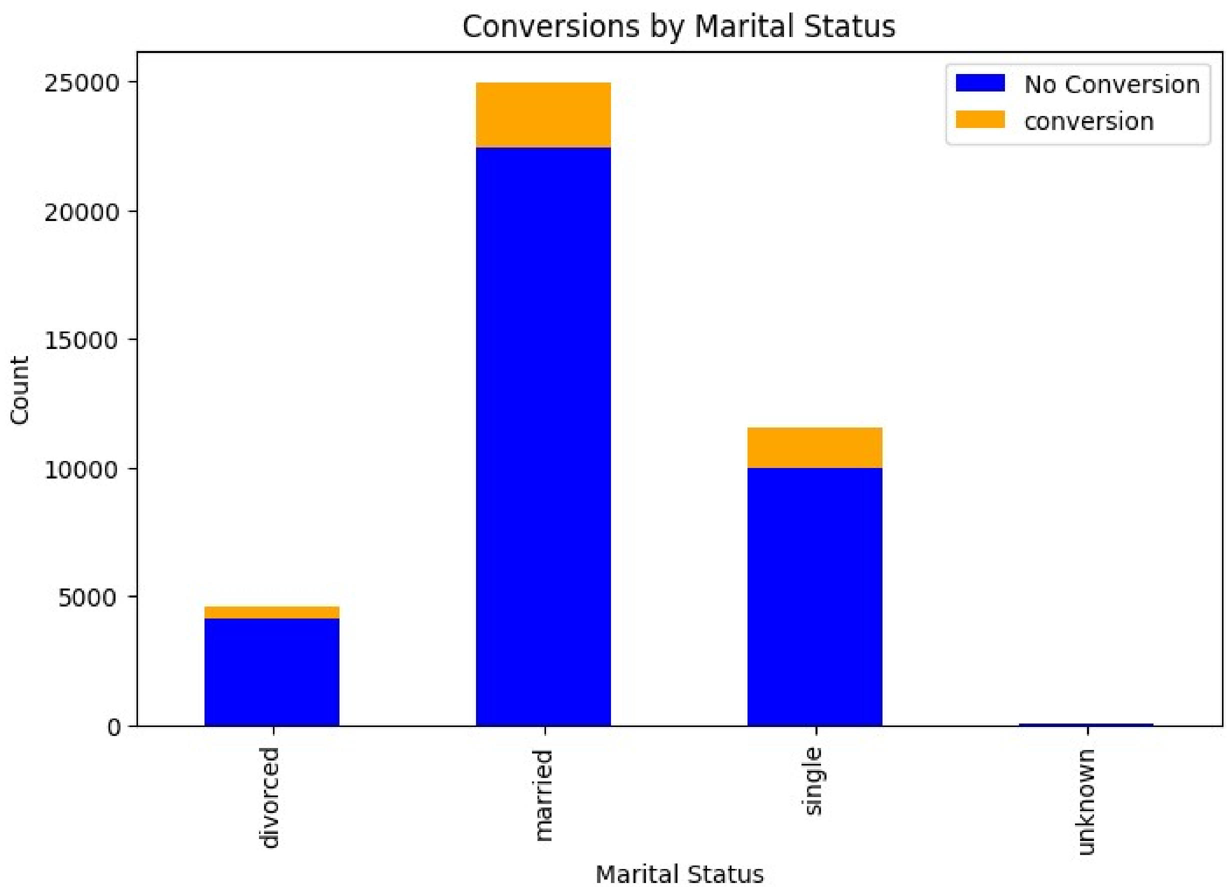
```
0        51-60
1        51-60
2        31-40
3        31-40
4        51-60
         ...
41183      60+
41184    41-50
41185    51-60
41186    41-50
41187      60+
Name: AgeGroup, Length: 41188, dtype: object
```

Interpretation:

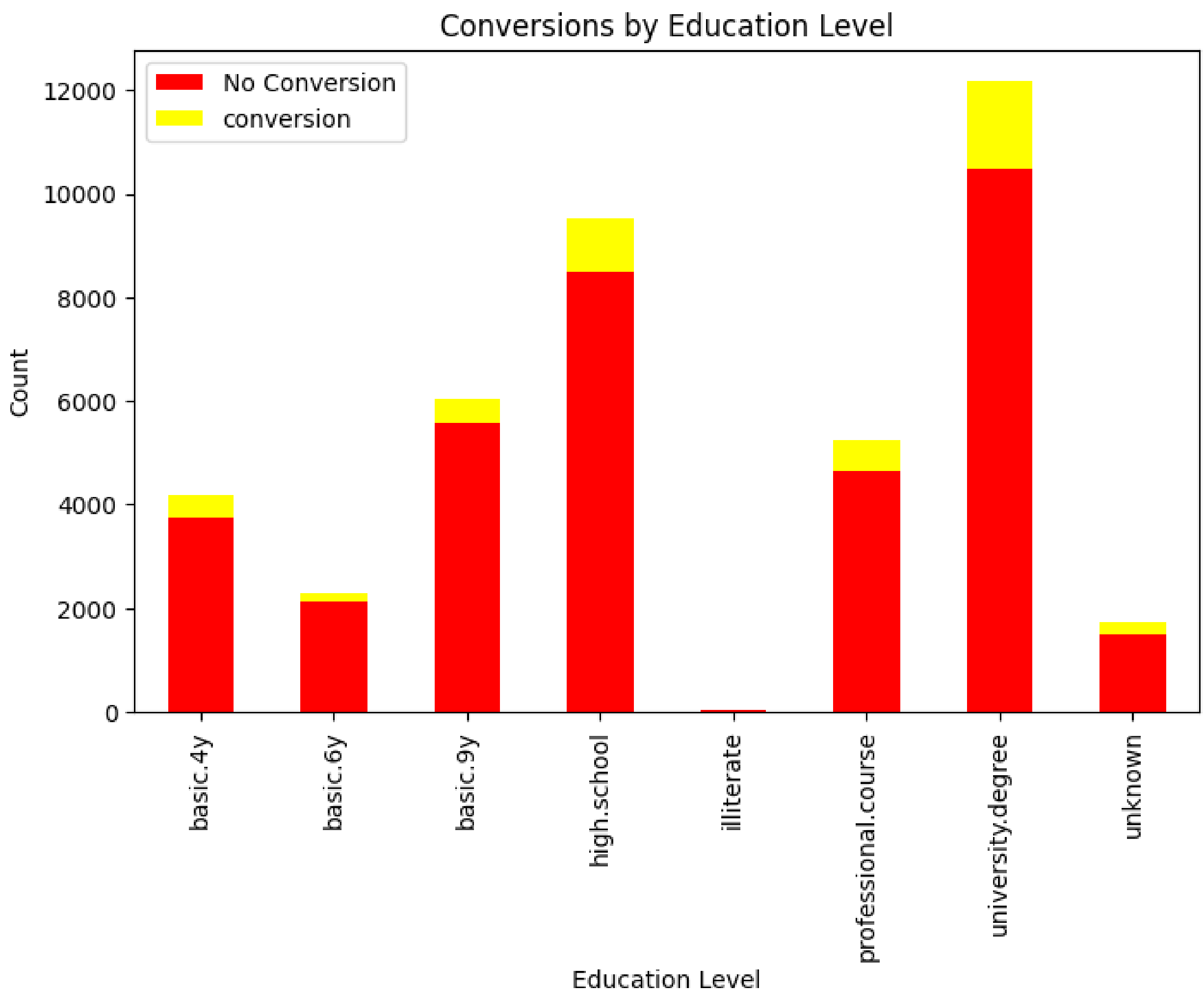
Age groups are created using a lambda function that assigns labels like '18-30' or '31-40' based on age. This helps analyze trends and behaviors within specific age ranges, making insights clearer and decisions easier.

vii. In an analysis comparing conversions and non-conversions by marital status, what additional insights could be explored and how would you extend the code to perform this analysis with the variable Education

```
marital_status_conversion = bank_df.groupby(['marital',
'conversion']).size().unstack()
education_conversion = bank_df.groupby(['education',
'conversion']).size().unstack()
marital_status_conversion.plot(kind='bar', stacked=True, figsize=(8,
5), color=['blue', 'orange'])
plt.title("Conversions by Marital Status")
plt.xlabel("Marital Status")
plt.ylabel("Count")
plt.legend(["No Conversion", "conversion"])
plt.show()
```




```
education_conversion.plot(kind='bar', stacked=True, figsize=(8, 5),
color=['red', 'yellow'])
plt.title("Conversions by Education Level")
plt.xlabel("Education Level")
plt.ylabel("Count")
plt.legend(["No Conversion", "conversion"])
plt.show()
```



Interpretation:

To analyze conversion rates by marital status and education level, follow these steps: Group the dataset by marital status and education level. Calculate the conversion rate as a percentage. Compare the results to observe proportional difference.