# Insurance analysis

February 2, 2026

## 1 Library importing and data loading

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df=pd.read_csv("medical_insurance.csv")
```

[3]:
```
# data Overview
df.head
```

[3]:
```
<bound method NDFrame.head of       age     sex     bmi  children smoker
region      charges
0      19  female  27.900         0    yes  southwest  16884.92400
1      18    male  33.770         1     no  southeast   1725.55230
2      28    male  33.000         3     no  southeast   4449.46200
3      33    male  22.705         0     no  northwest  21984.47061
4      32    male  28.880         0     no  northwest   3866.85520
…    …      …      …       …     …        …           …
2767   47  female  45.320         1     no  southeast   8569.86180
2768   21  female  34.600         0     no  southwest   2020.17700
2769   19    male  26.030         1    yes  northwest  16450.89470
2770   23    male  18.715         0     no  northwest  21595.38229
2771   54    male  31.600         0     no  southwest   9850.43200

[2772 rows x 7 columns]>
```

[4]:
```
# finding duplicate data from dataset
df.duplicated().sum()
```

[4]:
```
np.int64(1435)
```

[5]:
```
# deleting duplicate data from data frame
df = df.drop_duplicates()
```

[6]:
```
# Verifying duplicate data is removed
df.duplicated().sum()
```

[6]:
```
np.int64(0)
```

[7]:
```
# finding null values in dataframe
df.isnull().sum()
```

[7]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

[8]:
```
# Overviewing data types
df.dtypes
```

[8]:
```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

[9]:
```
# finding mean (average) insurance premium value
np.mean(df['charges'])
```

[9]:
```
np.float64(13279.121486655948)
```

[10]:
```
# finding maximum insurance premium value
np.max(df['charges'])
```

[10]:
```
63770.42801
```

[11]:
```
# finding minimum insurance premium value
np.min(df['charges'])
```

[11]:
```
1121.8739
```

[12]:
```
'''Finding outliers in charges column using NumPy percentile and IQR method␣
 ↪then making variable Outlier and store new dataframe in it which is
containing comparision lower and upper value with charges column for␣
 ↪calculating outlier (| = Symbol of or operator)'''
# Outliers are identified using the IQR method to understand high-cost␣
 ↪insurance customers.
Q1 = np.percentile(df['charges'],25)
```

```
Q3 = np.percentile(df['charges'],75)
IQR = Q3 - Q1

lower = Q1 - 1.5* IQR
upper = Q3 + 1.5* IQR

outliers = df[(df['charges'] < lower) | (df['charges']> upper)]

len(outliers)
```

[12]:
```
139
```

[13]:
```
# checking head 10 outlier charges values compared with their respective␣
 ↪age,bmi,smoker,children row values
```

[14]:
```
outliers[["age","bmi","smoker","children","charges"]].head(10)
```

[14]:

| | age | bmi | smoker | children | charges |
|---|---|---|---|---|---|
| 14 | 27 | 42.13 | yes | 0 | 39611.75770 |
| 19 | 30 | 35.30 | yes | 0 | 36837.46700 |
| 23 | 34 | 31.92 | yes | 1 | 37701.87680 |
| 29 | 31 | 36.30 | yes | 2 | 38711.00000 |
| 30 | 22 | 35.60 | yes | 0 | 35585.57600 |
| 34 | 28 | 36.40 | yes | 1 | 51194.55914 |
| 38 | 35 | 36.67 | yes | 1 | 39774.27630 |
| 39 | 60 | 39.90 | yes | 0 | 48173.36100 |
| 49 | 36 | 35.20 | yes | 1 | 38709.17600 |
| 53 | 36 | 34.43 | yes | 0 | 37742.57570 |

[15]:
```
# counting outlier smoker value as yes and no category wise
outliers["smoker"].value_counts()
```

[15]:
```
smoker
yes    136
no       3
Name: count, dtype: int64
```

[16]:
```
# statistiacal view of bmi outliers
outliers["bmi"].describe()
```

[16]:
```
count    139.000000
mean      35.564604
std        4.434917
min       22.895000
25%       32.667500
50%       35.200000
75%       37.660000
max       52.580000
```

```
Name: bmi, dtype: float64
```

[17]:
```
# making BMI category using numpy
df=df.copy()
df.loc[:,'bmi_category']= np.where(df['bmi']<18.5,'Underweight',
                    np.where(df['bmi']<25,'Normal',
                    np.where(df['bmi']<30,'overweight','obese')))
# cheacking bmi and bmi category head value comparison
df[['bmi','bmi_category']].head()
```

[17]:
```
      bmi bmi_category
0  27.900   overweight
1  33.770        obese
2  33.000        obese
3  22.705       Normal
4  28.880   overweight
```

[ ]:
```

```

[18]:
```
# making risk level category using numpy as high risk = 1 , low risk = 0
# High-risk customers are defined as smokers with BMI greater than 30,As this␣
 ↪group consistently shows extremely high insurance charges.


df["risk_level"] = np.where((df["smoker"]=="yes") & (df["bmi"]>30), 1, 0)

df["risk_level"].value_counts()
```

[18]:
```
risk_level
0    1193
1     144
Name: count, dtype: int64
```

[19]:
```
# calculating average charge values as per risk level
df.groupby("risk_level")["charges"].mean()
```

[19]:
```
risk_level
0     9849.472701
1    41692.808992
Name: charges, dtype: float64
```

[20]:
```
# calculating median charge values as per risk level
df.groupby("risk_level")["charges"].median()
```

[20]:
```
risk_level
0     8342.90875
1    40918.31450
Name: charges, dtype: float64
```

[21]:
```
# for better understanding and data visualization value replacing in column␣
↪risk level (0 = low risk, 1= High risk)
df['risk_level'] = df['risk_level'].replace({1: 'High Risk', 0: 'Low Risk'})
```

[22]:
```
# checking column names
df.columns
```

[22]:
```
Index(['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges',
       'bmi_category', 'risk_level'],
      dtype='object')
```

[23]:
```
df.rename(columns={'sex':'gender','smoker':'is_smoker'},inplace=True)
```

[24]:
```
df['gender'].unique()
```

[24]:
```
array(['female', 'male'], dtype=object)
```

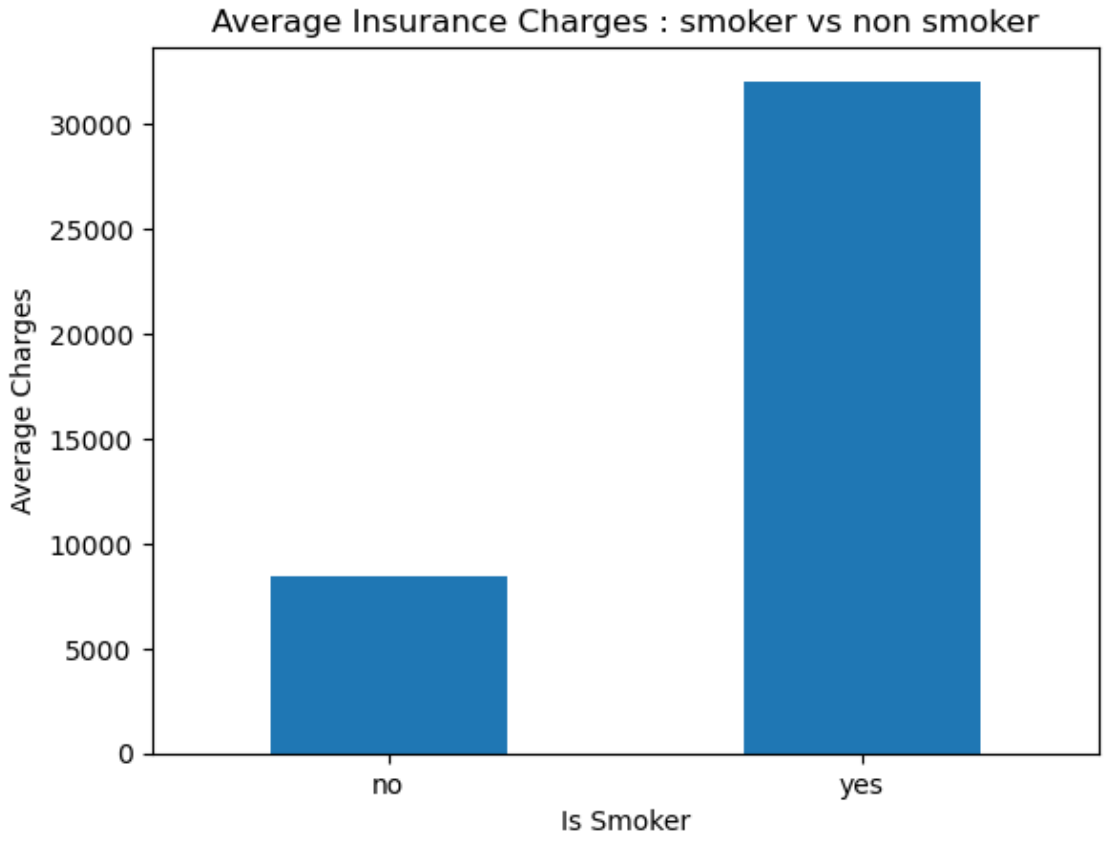
[25]:
```
df.head(10)
```

[25]:

| | age | gender | bmi | children | is_smoker | region | charges | bmi_category | risk_level |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 19 | female | 27.900 | 0 | yes | southwest | 16884.92400 | overweight | Low Risk |
| 1 | 18 | male | 33.770 | 1 | no | southeast | 1725.55230 | obese | Low Risk |
| 2 | 28 | male | 33.000 | 3 | no | southeast | 4449.46200 | obese | Low Risk |
| 3 | 33 | male | 22.705 | 0 | no | northwest | 21984.47061 | Normal | Low Risk |
| 4 | 32 | male | 28.880 | 0 | no | northwest | 3866.85520 | overweight | Low Risk |
| 5 | 31 | female | 25.740 | 0 | no | southeast | 3756.62160 | overweight | Low Risk |
| 6 | 46 | female | 33.440 | 1 | no | southeast | 8240.58960 | obese | Low Risk |
| 7 | 37 | female | 27.740 | 3 | no | northwest | 7281.50560 | overweight | Low Risk |
| 8 | 37 | male | 29.830 | 2 | no | northeast | 6406.41070 | overweight | Low Risk |
| 9 | 60 | female | 25.840 | 0 | no | northwest | 28923.13692 | overweight | Low Risk |

## 2 Data Visualisation

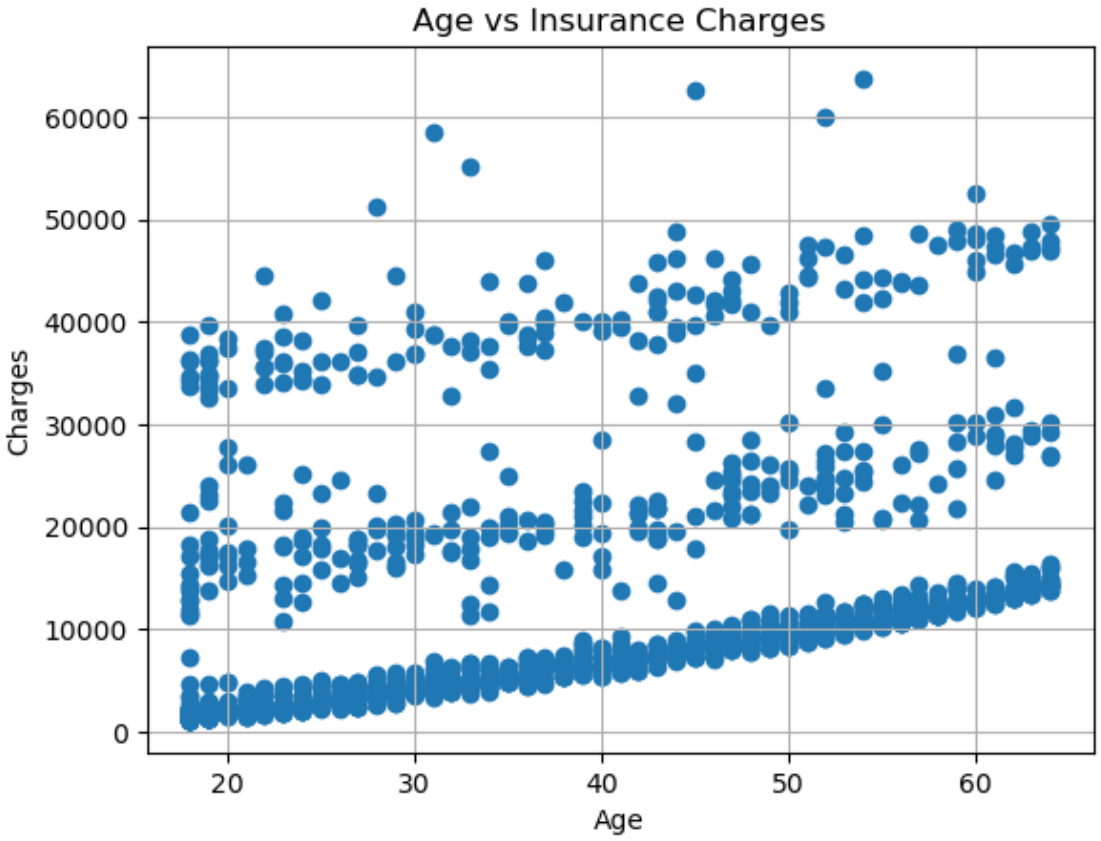
[26]:
```
# Insurance chages as per smoker or non smoker
avg_smoker = df.groupby('is_smoker')['charges'].mean()

plt.figure()
avg_smoker.plot(kind='bar')
plt.title('Average Insurance Charges : smoker vs non smoker')
plt.xlabel('Is Smoker')
plt.ylabel('Average Charges')
plt.xticks(rotation = 0)
plt.show()
```




[27]:
```
# Scatter plot comparision between Age vs Insurance charge
```

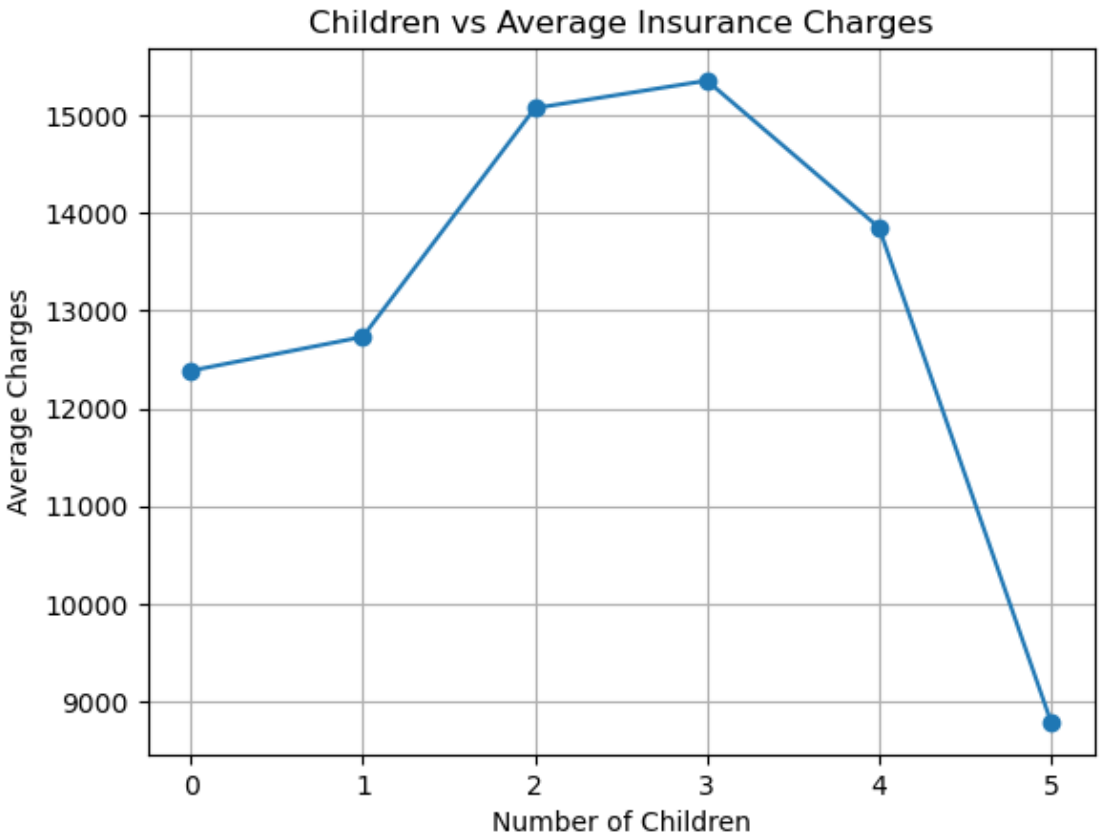
[28]:
```
plt.figure()
plt.scatter(df['age'], df['charges'])
plt.title("Age vs Insurance Charges")
plt.xlabel("Age")
plt.ylabel("Charges")
```

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



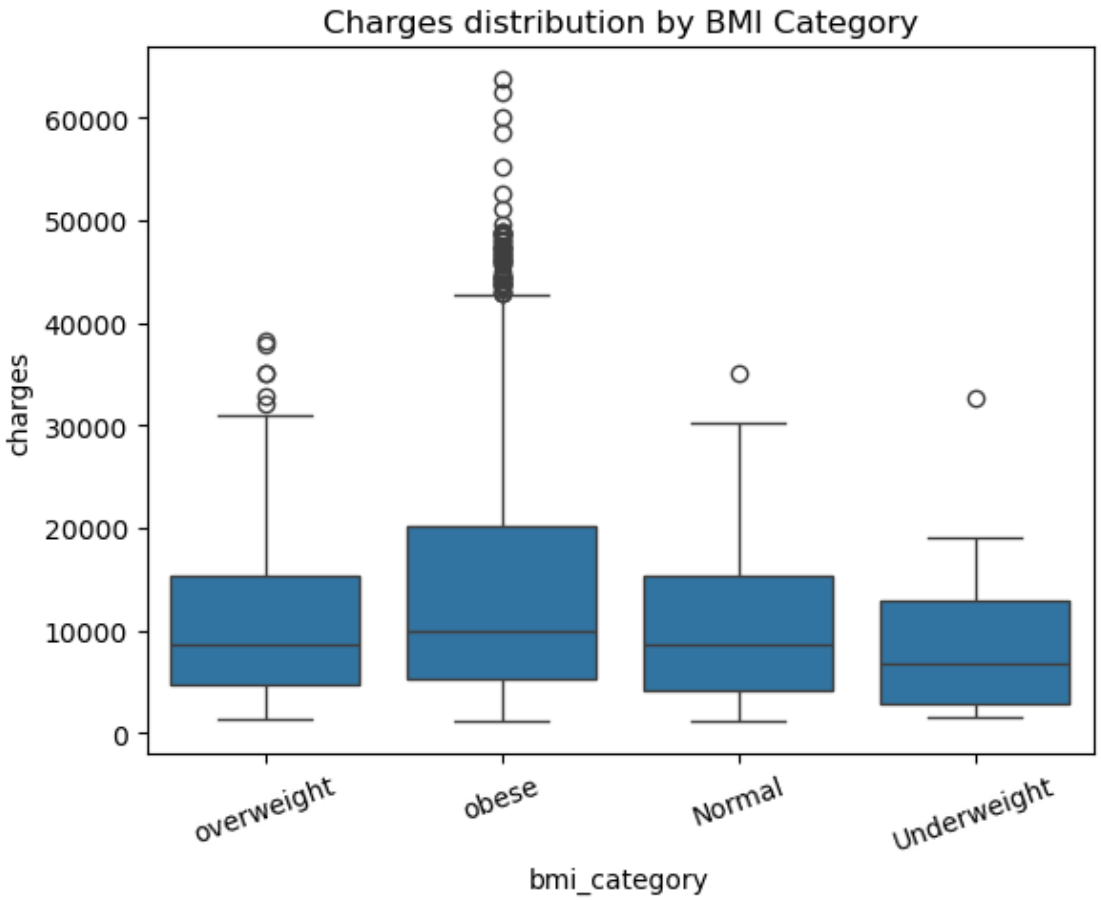

[29]:
```
# Line chart comparison between children vs average insurance charges
```

[30]:
```
children_avg = df.groupby('children')['charges'].mean()

plt.figure()
plt.plot(children_avg.index, children_avg.values, marker='o')
plt.title("Children vs Average Insurance Charges")
plt.xlabel("Number of Children")
plt.ylabel("Average Charges")
plt.grid(True)
plt.show()
```



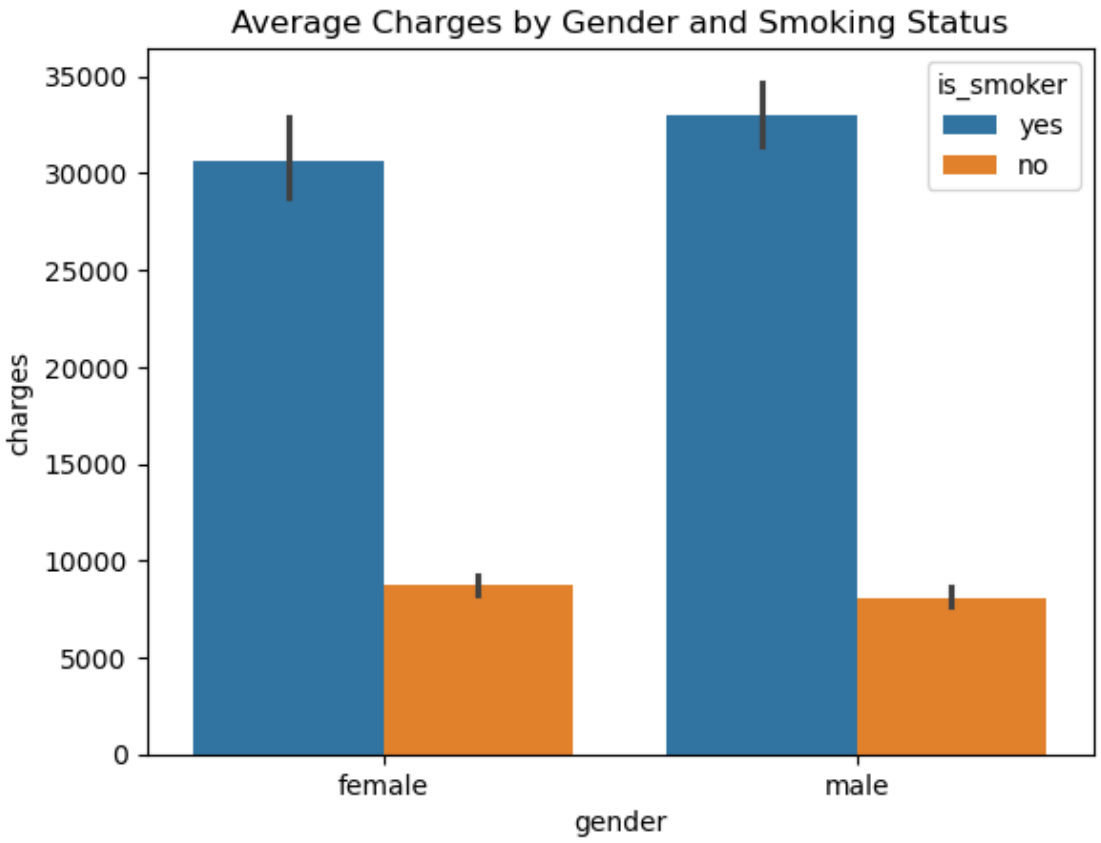

[ ]:

[31]:
```
# box plot comparison between Insurance charges vs BMI category
plt.figure()
sns.boxplot(x='bmi_category', y='charges', data=df)
plt.title("Charges distribution by BMI Category")
plt.xticks(rotation=20)
plt.show()
```




[32]:
```
# barplot comparison between Average charge vs gender and smoking status
```

[33]:
```
plt.figure()
sns.barplot(x='gender', y='charges', hue='is_smoker', data=df, estimator='mean')
plt.title("Average Charges by Gender and Smoking Status")
plt.show()
```




[ ]:

[ ]:

[36]:
```
df.to_csv("insurance_analysis_cleaned.csv", index=False)
```

[39]:
```
from sqlalchemy import create_engine

engine = create_engine("mysql+pymysql://root:root@localhost:3306/insurance_db")

df.to_sql("insurance_data", con=engine, if_exists="append", index=False)

print(" Data successfully loaded into MySQL")
```

```
Data successfully loaded into MySQL
```

[ ]: