✅ STEP 1: Load & Explore the Data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


# Load data
df = pd.read_csv("/content/ola_driver_scaler.csv")
```

✅ STEP 2: Data Structure and Types

```
# Initial Inspection
print(df.shape)
print(df.info())
print(df.head())
```

```
(19104, 14)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19104 entries, 0 to 19103
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Unnamed: 0            19104 non-null  int64
 1   MMM-YY                19104 non-null  object
 2   Driver_ID             19104 non-null  int64
 3   Age                   19043 non-null  float64
 4   Gender                19052 non-null  float64
 5   City                  19104 non-null  object
 6   Education_Level       19104 non-null  int64
 7   Income                19104 non-null  int64
 8   Dateofjoining         19104 non-null  object
 9   LastWorkingDate       1616 non-null   object
 10  Joining Designation   19104 non-null  int64
 11  Grade                 19104 non-null  int64
 12  Total Business Value  19104 non-null  int64
 13  Quarterly Rating      19104 non-null  int64
dtypes: float64(2), int64(8), object(4)
memory usage: 2.0+ MB
None
   Unnamed: 0    MMM-YY  Driver_ID   Age  Gender City  Education_Level  \
0           0  01/01/19          1  28.0     0.0  C23                2
1           1  02/01/19          1  28.0     0.0  C23                2
2           2  03/01/19          1  28.0     0.0  C23                2
3           3  11/01/20          2  31.0     0.0   C7                2
4           4  12/01/20          2  31.0     0.0   C7                2

   Income Dateofjoining LastWorkingDate  Joining Designation  Grade  \
0   57387      24/12/18             NaN                    1      1
1   57387      24/12/18             NaN                    1      1
2   57387      24/12/18        03/11/19                    1      1
3   67016      11/06/20             NaN                    2      2
4   67016      11/06/20             NaN                    2      2

   Total Business Value  Quarterly Rating
0               2381060                 2
1               -665480                 2
2                     0                 2
3                     0                 1
4                     0                 1
```

✅ STEP 3: Convert Date Columns

```
# Convert dates
df['MMM-YY'] = pd.to_datetime(df['MMM-YY'], format='%d/%m/%y')
df['Dateofjoining'] = pd.to_datetime(df['Dateofjoining'], format='%d/%m/%y')
df['LastWorkingDate'] = pd.to_datetime(df['LastWorkingDate'], format='%d/%m/%y', errors='coerce')


# Data Types & Summary
print(df.describe(include='all'))
```

```
print(df.isnull().sum())
```

```
50%        34.000000      0.000000    NaN         1.000000   60087.000000
75%        39.000000      1.000000    NaN         2.000000   83969.000000
max        58.000000      1.000000    NaN         2.000000  188418.000000
std         6.257912      0.493367    NaN         0.800167   30914.515344

                        Dateofjoining               LastWorkingDate  \
count                           19104                          1616
unique                            NaN                           NaN
top                               NaN                           NaN
freq                              NaN                           NaN
mean    2018-04-20 21:49:17.788944896  2019-12-26 23:22:34.455445760
min               2013-01-04 00:00:00            2018-12-31 00:00:00
25%               2016-11-28 00:00:00            2019-06-10 00:00:00
50%               2018-09-19 00:00:00            2019-12-20 12:00:00
75%               2019-10-25 00:00:00            2020-07-14 00:00:00
max               2020-12-28 00:00:00            2020-12-28 00:00:00
std                               NaN                           NaN

        Joining Designation         Grade  Total Business Value  \
count          19104.000000  19104.000000          1.910400e+04
unique                  NaN           NaN                   NaN
top                     NaN           NaN                   NaN
freq                    NaN           NaN                   NaN
mean               1.690536      2.252670          5.716621e+05
min                1.000000      1.000000         -6.000000e+06
25%                1.000000      1.000000          0.000000e+00
50%                1.000000      2.000000          2.500000e+05
75%                2.000000      3.000000          6.997000e+05
max                5.000000      5.000000          3.374772e+07
std                0.836984      1.026512          1.128312e+06

        Quarterly Rating
count       19104.000000
unique               NaN
top                  NaN
freq                 NaN
mean            2.008899
min             1.000000
25%             1.000000
50%             2.000000
75%             3.000000
max             4.000000
std             1.009832
Unnamed: 0                 0
MMM-YY                     0
Driver_ID                  0
Age                       61
Gender                    52
City                       0
Education_Level            0
Income                     0
Dateofjoining              0
LastWorkingDate        17488
Joining Designation        0
Grade                      0
Total Business Value       0
Quarterly Rating           0
dtype: int64
```

✅ 2. Initial Data Observations

📊 Data Shape & Structure

- Rows: **19,104**, Columns: **14**
- Mixed data types:
  - Numerical: `float64`, `int64`
  - Categorical/Text: `object`

🔍 Missing Values

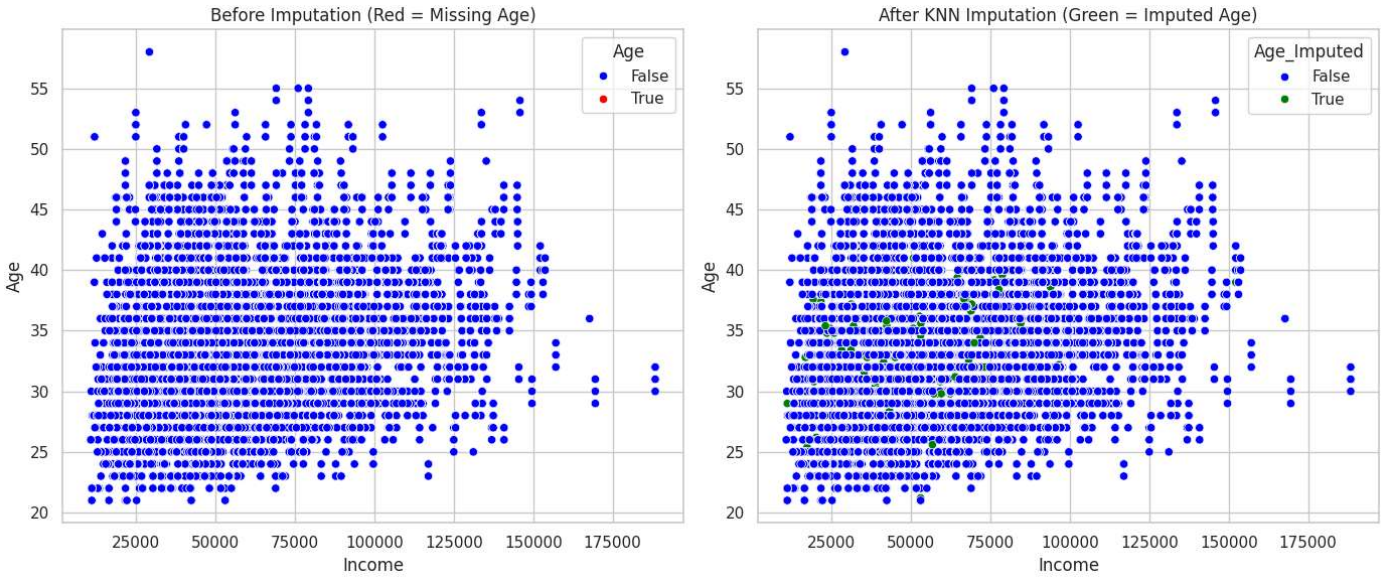
Column	Missing Count	% Missing
Age	61	~0.32%
Gender	52	~0.27%
LastWorkingDate	17,488	~91.5%

✅ 3. Statistical Summary of Numerical Columns

Variable	Mean	Min	Max	Std Dev	Notes
Age	34.7	21	58	6.26	Normal working-age range
Income	65,652.02	10,747	188,418	30,914.5	Wide range; check for skew
Total Business Value	571,662.14	-6,000,000	33,747,720	1,128,312	Large outliers present
Quarterly Rating	2.01	1	4	1.01	Discrete rating (1 to 4)

STEP 4: KNN Imputation

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.impute import KNNImputer

# Make a copy before imputation
df_pre = df.copy()

# Only numerical columns for KNN
num_cols = ['Age', 'Income', 'Total Business Value']
df_num = df[num_cols]

# Save mask of missing values
missing_mask = df_num['Age'].isna()

# Apply KNN Imputer
imputer = KNNImputer(n_neighbors=5)
df_imputed = pd.DataFrame(imputer.fit_transform(df_num), columns=num_cols)

# Create new column to track imputation
df_imputed['Age_Imputed'] = df_num['Age'].isna()

# Plot Income vs Age before and after
plt.figure(figsize=(14, 6))

# Before Imputation
plt.subplot(1, 2, 1)
sns.scatterplot(x=df_num['Income'], y=df_num['Age'], hue=missing_mask, palette={True: 'red', False: 'blue'})
plt.title('Before Imputation (Red = Missing Age)')
plt.xlabel('Income')
plt.ylabel('Age')

# After Imputation
plt.subplot(1, 2, 2)
sns.scatterplot(x=df_imputed['Income'], y=df_imputed['Age'], hue=df_imputed['Age_Imputed'], palette={True: 'green', False: 'blue'})
plt.title('After KNN Imputation (Green = Imputed Age)')
plt.xlabel('Income')
plt.ylabel('Age')

plt.tight_layout()
plt.show()
```




```
from sklearn.impute import KNNImputer

# Select only numerical columns
num_cols = df.select_dtypes(include=[np.number]).columns
df_num = df[num_cols]

# Check missing values
print(df_num.isnull().sum())

# Apply KNN Imputer
imputer = KNNImputer(n_neighbors=5)
df_num_imputed = pd.DataFrame(imputer.fit_transform(df_num), columns=num_cols)

# Replace original numerical columns with imputed ones
df[num_cols] = df_num_imputed

# Confirm imputation
print(df.isnull().sum())
```

```
Unnamed: 0               0
Driver_ID                0
Age                     61
Gender                  52
Education_Level          0
Income                   0
Joining Designation      0
Grade                    0
Total Business Value     0
Quarterly Rating         0
dtype: int64
Unnamed: 0                0
MMM-YY                    0
Driver_ID                 0
Age                       0
Gender                    0
City                      0
Education_Level           0
Income                    0
Dateofjoining             0
LastWorkingDate       17488
Joining Designation       0
Grade                     0
Total Business Value      0
Quarterly Rating          0
dtype: int64
```

✅ STEP 5: Aggregate Driver Data (Groupby)

```
# Group by Driver_ID
agg_df = df.groupby('Driver_ID').agg({
    'Age': 'mean',
    'Income': 'mean',
    'Total Business Value': 'mean',
    'Quarterly Rating': lambda x: x.mode()[0] if not x.mode().empty else None,
    'Gender': 'first',
    'City': 'first',
    'Education_Level': 'first',
    'Joining Designation': 'first',
    'Grade': 'first',
    'Dateofjoining': 'first',
    'LastWorkingDate': 'last'
}).reset_index()
```

```
agg_df.shape
```

```
(2381, 12)
```

✅ STEP 6: Feature Engineering

```
# Target variable: 1 if driver has left (has a LastWorkingDate)
agg_df['Attrition'] = agg_df['LastWorkingDate'].notnull().astype(int)

agg_df['Quarterly Rating Change'] = (df.groupby('Driver_ID')['Quarterly Rating'].apply(lambda x: int(x.iloc[-1] > x.iloc[0]))).values
agg_df['Income Change'] = (df.groupby('Driver_ID')['Income'].apply(lambda x: int(x.iloc[-1] > x.iloc[0]))).values
```

✅ STEP 7: Statistical Summary of Final Dataset

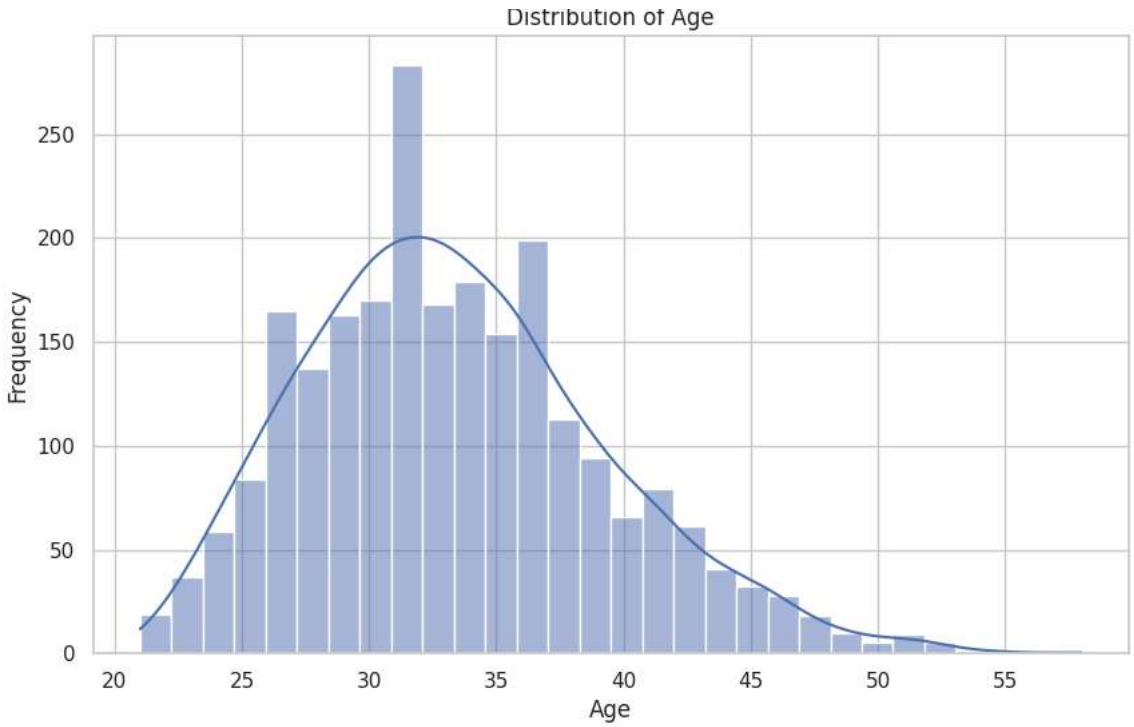
```
agg_df.describe(include='all').T
```

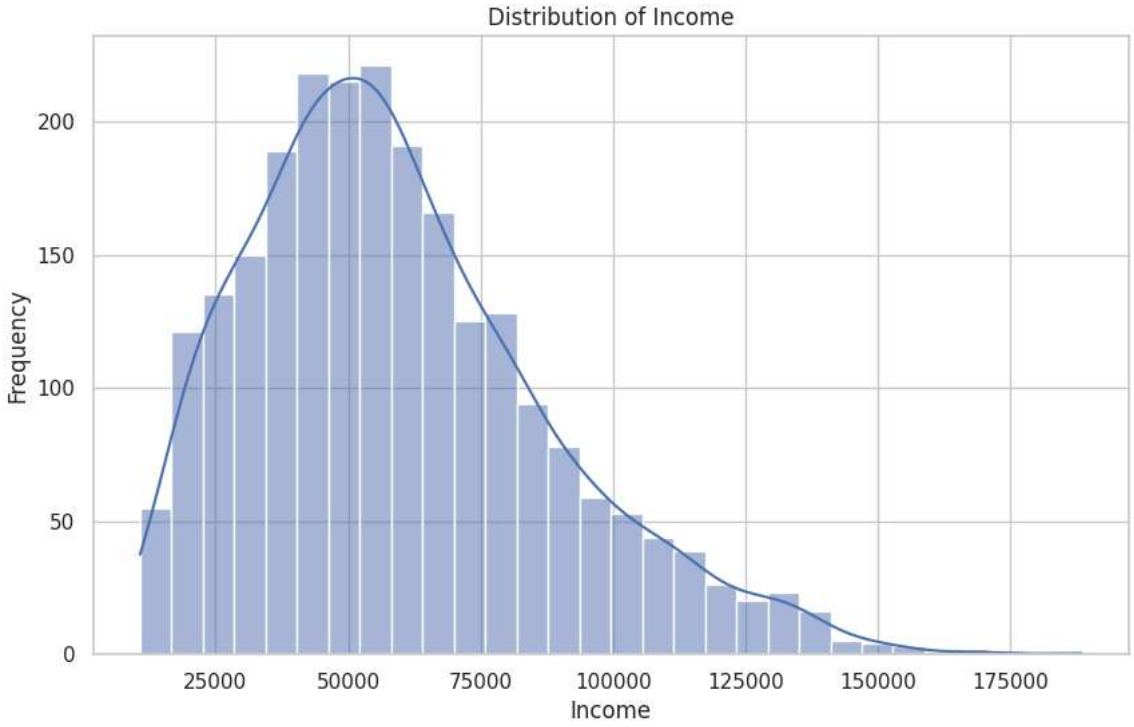
	count	unique	top	freq	mean	min	25%	50%	75%	max	std
Driver_ID	2381.0	NaN	NaN	NaN	1397.559009	1.0	695.0	1400.0	2100.0	2788.0	806.161628
Age	2381.0	NaN	NaN	NaN	33.369298	21.0	29.0	33.0	37.0	58.0	5.890567
Income	2381.0	NaN	NaN	NaN	59232.460484	10747.0	39104.0	55285.0	75835.0	188418.0	28298.214012
Total Business Value	2381.0	NaN	NaN	NaN	312085.359327	-197932.857143	0.0	150624.444444	429498.75	3972127.5	449570.506711
Quarterly Rating	2381.0	NaN	NaN	NaN	1.52037	1.0	1.0	1.0	2.0	4.0	0.810711
Gender	2381.0	NaN	NaN	NaN	0.410332	0.0	0.0	0.0	1.0	1.0	0.491997
City	2381	29	C20	152	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Education_Level	2381.0	NaN	NaN	NaN	1.00756	0.0	0.0	1.0	2.0	2.0	0.81629
Joining Designation	2381.0	NaN	NaN	NaN	1.820244	1.0	1.0	2.0	2.0	5.0	0.841433
Grade	2381.0	NaN	NaN	NaN	2.078538	1.0	1.0	2.0	3.0	5.0	0.931321
Dateofjoining	2381	NaN	NaN	NaN	2019-01-27 12:58:58.009239808	2013-01-04 00:00:00	2018-06-26 00:00:00	2019-06-23 00:00:00	2020-04-14 00:00:00	2020-12-28 00:00:00	NaN
LastWorkingDate	1616	NaN	NaN	NaN	2019-12-26 23:22:34.455445760	2018-12-31 00:00:00	2019-06-10	2019-12-20 12:00:00	2020-07-14	2020-12-28	NaN

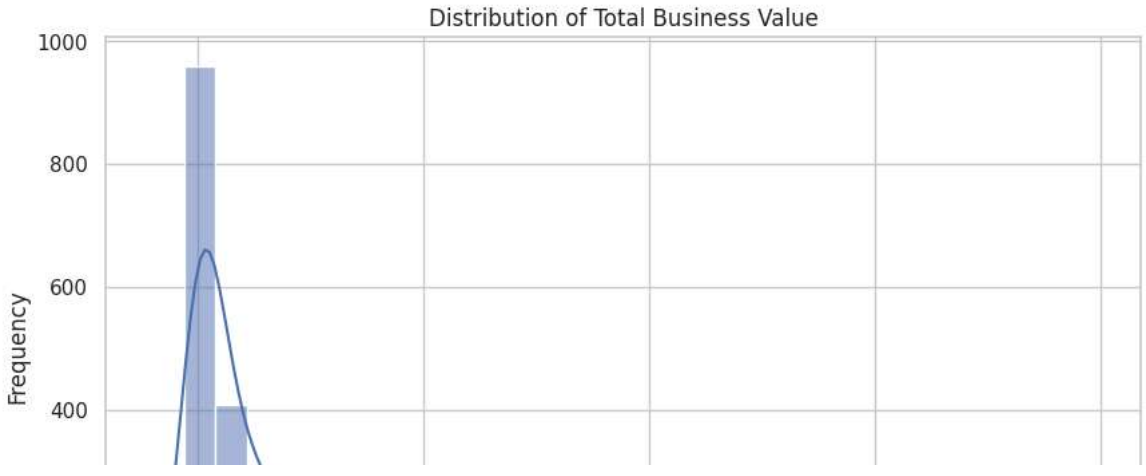
```
# Optional: Increase plot resolution
sns.set(style="whitegrid")
plt.rcParams["figure.figsize"] = (10, 6)
```

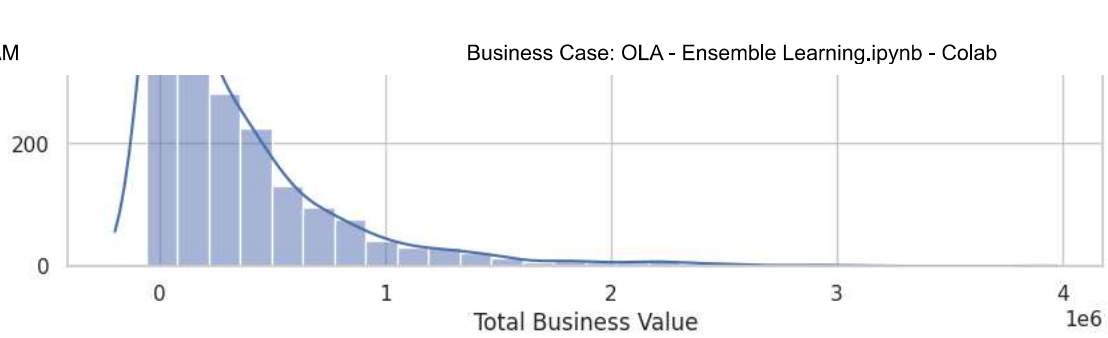
📊 Univariate Analysis

```
# Continuous Variables
continuous_vars = ['Age', 'Income', 'Total Business Value']

for col in continuous_vars:
    plt.figure()
    sns.histplot(agg_df[col].dropna(), kde=True, bins=30)
    plt.title(f'Distribution of {col}')
    plt.xlabel(col)
    plt.ylabel('Frequency')
    plt.show()
```


Distribution of Age
Frequency
Age
0
50
100
150
200
250
20
25
30
35
40
45
50
55


Distribution of Income
Frequency
Income
0
50
100
150
200
25000
50000
75000
100000
125000
150000
175000


Distribution of Total Business Value
Frequency
0
400
600
800
1000



1. Distribution of Age

- The age distribution appears **right-skewed**.
- Most employees are **younger**, likely in their **20s to early 30s**.
- A few outliers exist at the higher age range.

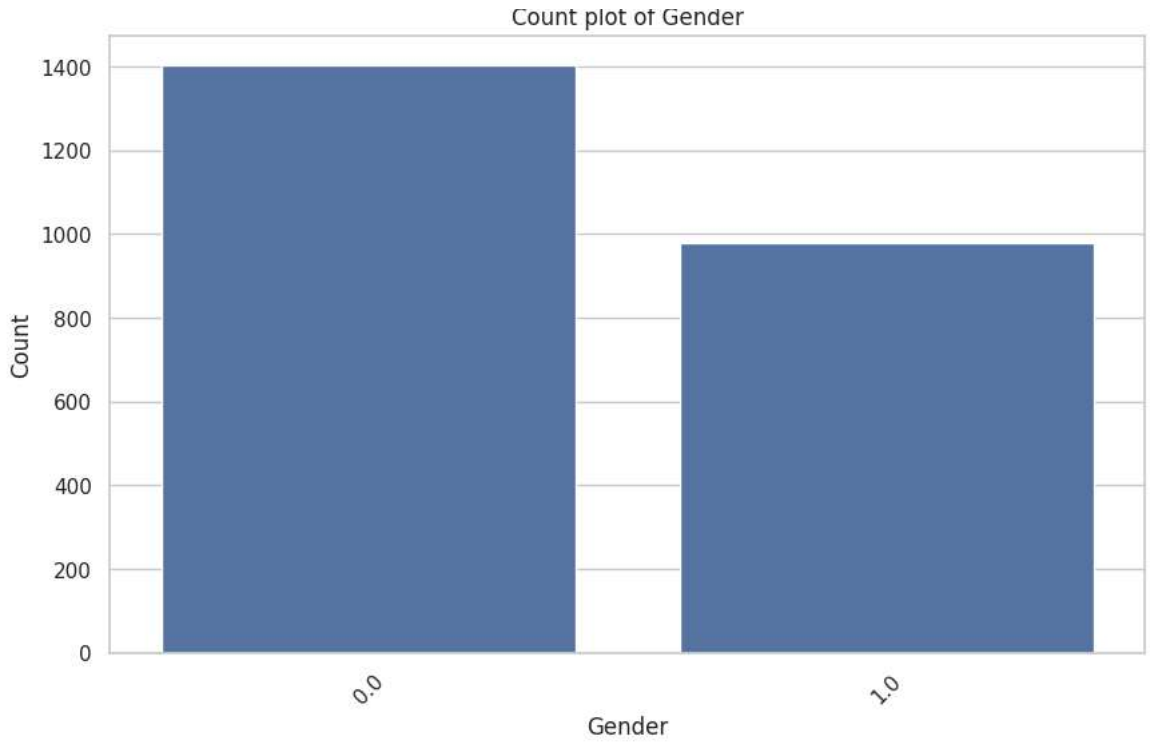
2. Distribution of Income

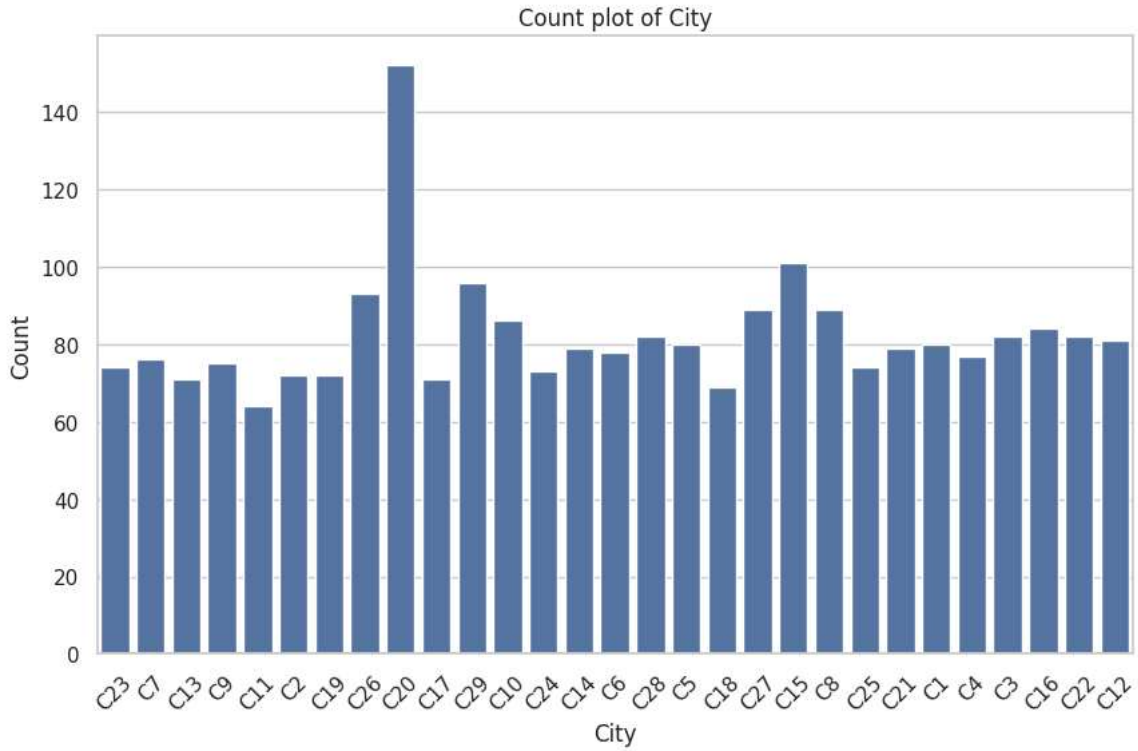
- **Right-skewed** distribution.
- Most employees earn **lower to mid-range salaries**.
- High-income outliers may represent **senior or high-performing roles**.

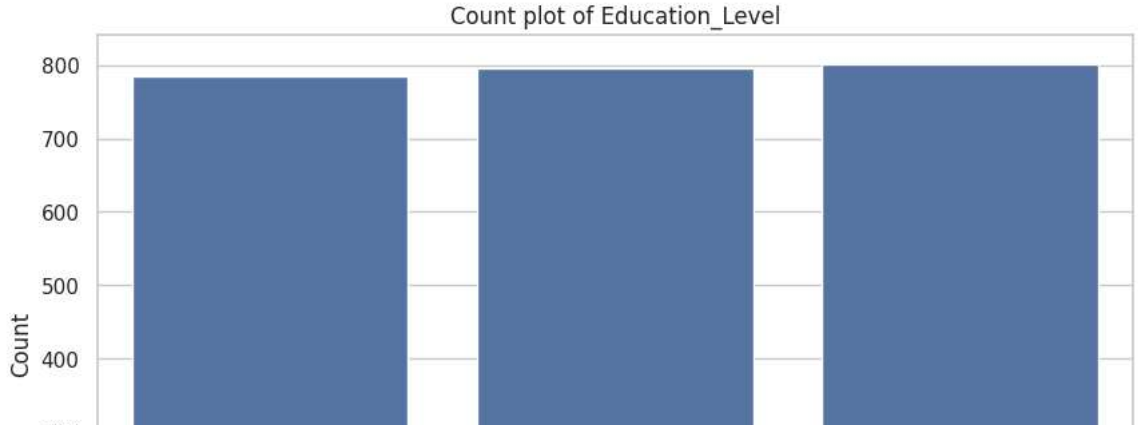
3. Distribution of Total Business Value

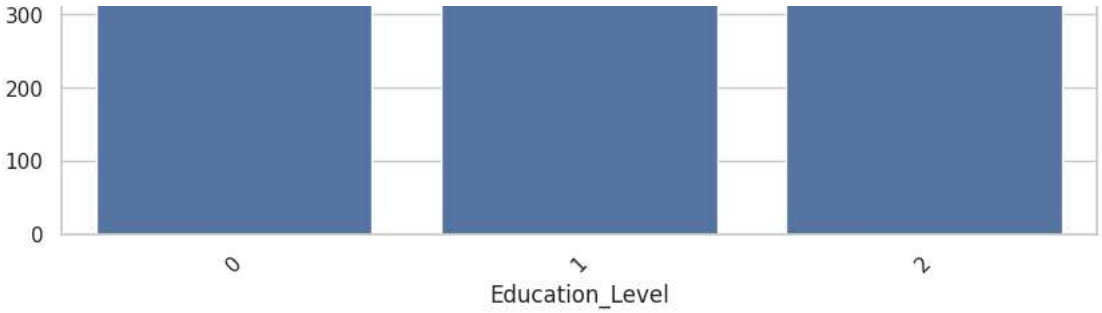
- Shows a **long right tail**, indicating a few employees generate **very high business value**.
- Majority generate **moderate business value**, suggesting a **Pareto distribution** pattern.

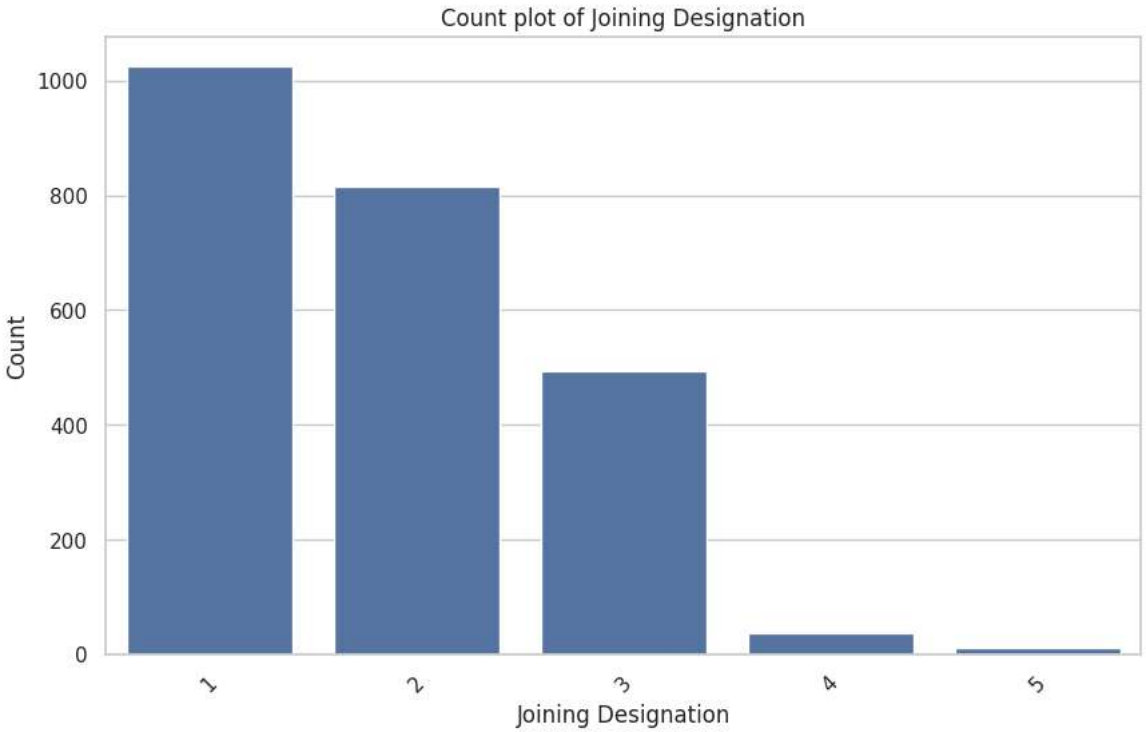
```
# Categorical Variables
categorical_vars = ['Gender', 'City', 'Education_Level', 'Joining Designation', 'Grade', 'Quarterly Rating']

for col in categorical_vars:
    plt.figure()
    sns.countplot(x=col, data=agg_df)
    plt.title(f'Count plot of {col}')
    plt.xticks(rotation=45)
    plt.ylabel('Count')
    plt.show()
```

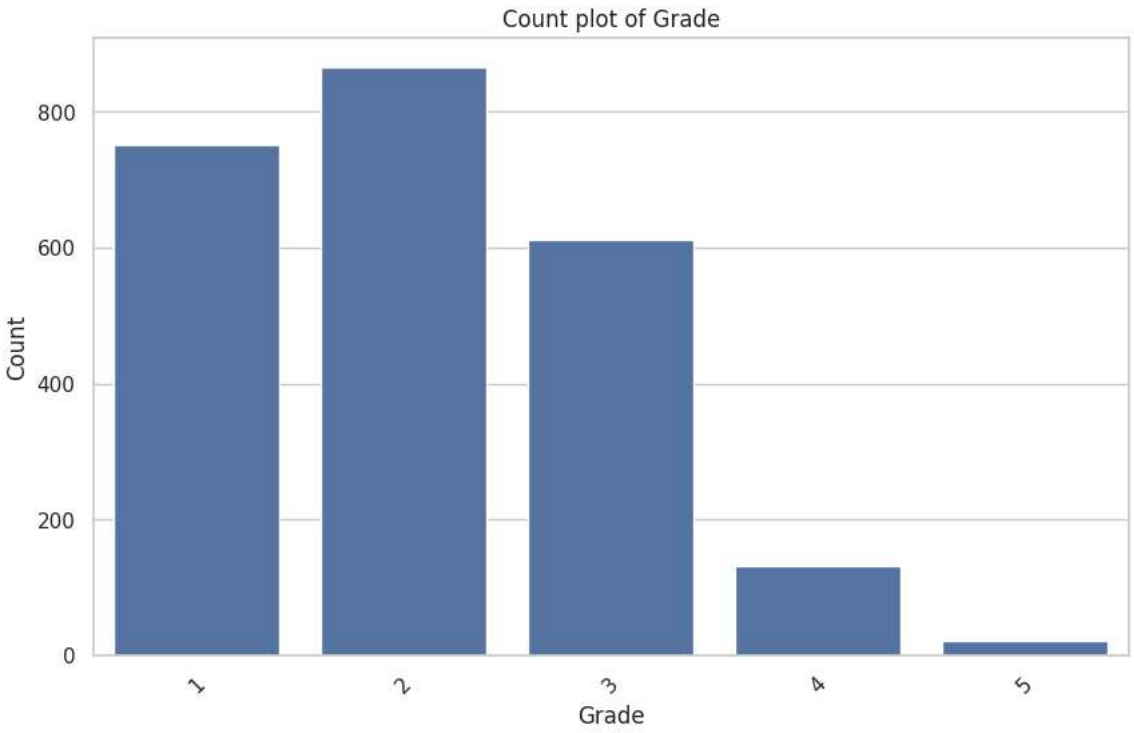

Count plot of Gender
Count
Gender
0.0
1.0
0
200
400
600
800
1000
1200
1400

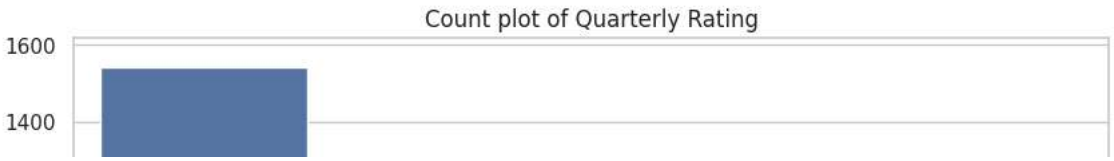

Count plot of City
Count
City
0
20
40
60
80
100
120
140
C23 C7 C13 C9 C11 C2 C19 C26 C20 C17 C29 C10 C24 C14 C6 C28 C5 C18 C27 C15 C8 C25 C21 C1 C4 C3 C16 C22 C12

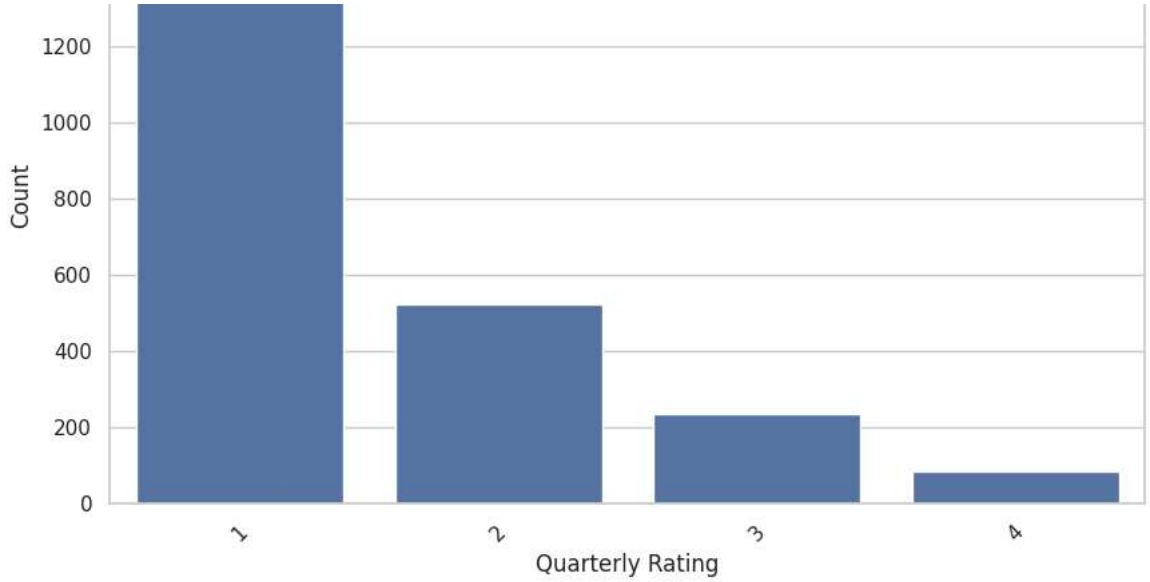

Count plot of Education_Level
Count
400
500
600
700
800


300
200
100
0
0
1
2
Education_Level


Count plot of Joining Designation
1000
800
600
400
200
0
Count
1
2
3
4
5
Joining Designation


Count plot of Grade
800
600
400
200
0
Count
1
2
3
4
5
Grade


Count plot of Quarterly Rating
1600
1400


1200
1000
800
600
400
200
0
Count
1
2
3
4
Quarterly Rating

4. Count Plot of Gender

- **Imbalanced gender representation**.
- **More male employees** than female.

5. Count Plot of Education Level

- Most employees have **Bachelor's or Master's degrees**.
- Very few employees have education levels below Bachelor's or above Master's (e.g., PhD).

6. Count Plot of City

- **Uneven distribution** across cities.
- Some cities have a **significantly higher employee count**, possibly regional headquarters or operational hubs.

7. Count Plot of Joining Designation

- Wide variety of designations.
- **Entry-level or mid-level roles** dominate.

8. Count Plot of Grade

- Certain grades are **more common**, indicating a **pyramidal hierarchy**.
- Lower grades have the highest employee count.

9. Count Plot of Quarterly Rating

- Most employees received **mid-range ratings (3-4)**.
- Few extreme high or low ratings suggest a **moderate performance bias**.

📊 Univariate Analysis Summary with Recommendations

1. Distribution of Age

- **Insight:** The age distribution appears right-skewed. Most employees are younger, likely in their 20s to early 30s.
- **Recommendation:**
  - Consider age-based mentorship programs to leverage the experience of older employees.
  - Align employee benefits and career development programs with the needs of a younger workforce.

2. Distribution of Total Business Value

- **Insight:** Shows a long right tail—few employees generate very high business value.
- **Recommendation:**
  - Recognize and reward high performers to maintain motivation.
  - Analyze what drives high business value and apply best practices across the organization.

3. Distribution of Income

- **Insight:** Most employees earn lower to mid-range salaries; income distribution is right-skewed.
- **Recommendation:**
  - Conduct a compensation benchmarking study to ensure fairness and competitiveness.
  - Introduce performance-linked bonuses to motivate mid-range earners.

4. Count Plot of Gender

- **Insight:** More male employees than female.
- **Recommendation:**
  - Promote gender diversity in hiring practices.
  - Ensure equal growth opportunities and workplace support for all genders.

5. Count Plot of Education Level

- **Insight:** Majority have Bachelor's or Master's degrees.
- **Recommendation:**
  - Offer advanced learning programs (e.g., certification, executive courses).
  - Utilize the existing educational background for specialized projects and innovation.

6. Count Plot of City

- **Insight:** Uneven employee distribution across cities.
- **Recommendation:**
  - Investigate if the geographic concentration aligns with business needs.
  - Optimize office resources and infrastructure based on employee distribution.

7. Count Plot of Joining Designation

- **Insight:** Entry-level and mid-level roles dominate.
- **Recommendation:**
  - Strengthen onboarding and early career development programs.
  - Develop leadership training to build a pipeline for senior roles.

8. Count Plot of Grade

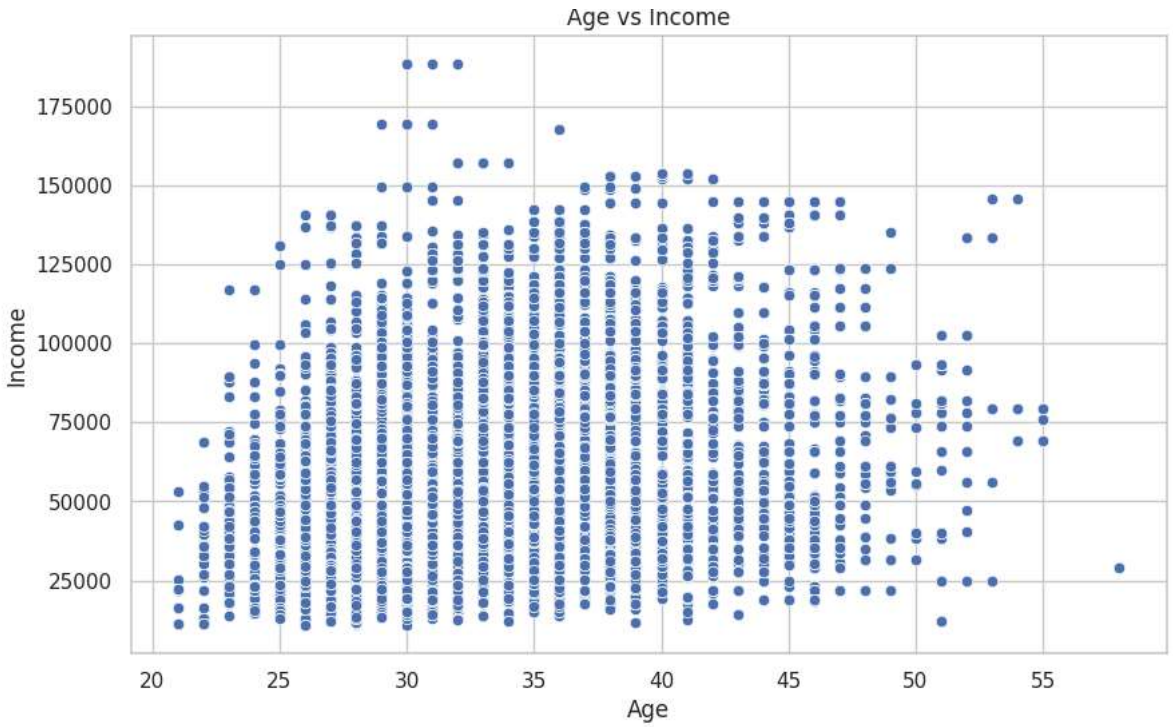
- **Insight:** Certain grades are more common, indicating a pyramidal structure.
- **Recommendation:**
  - Review promotion criteria and employee progression timelines.
  - Ensure clear role expectations and career pathways at each grade level.

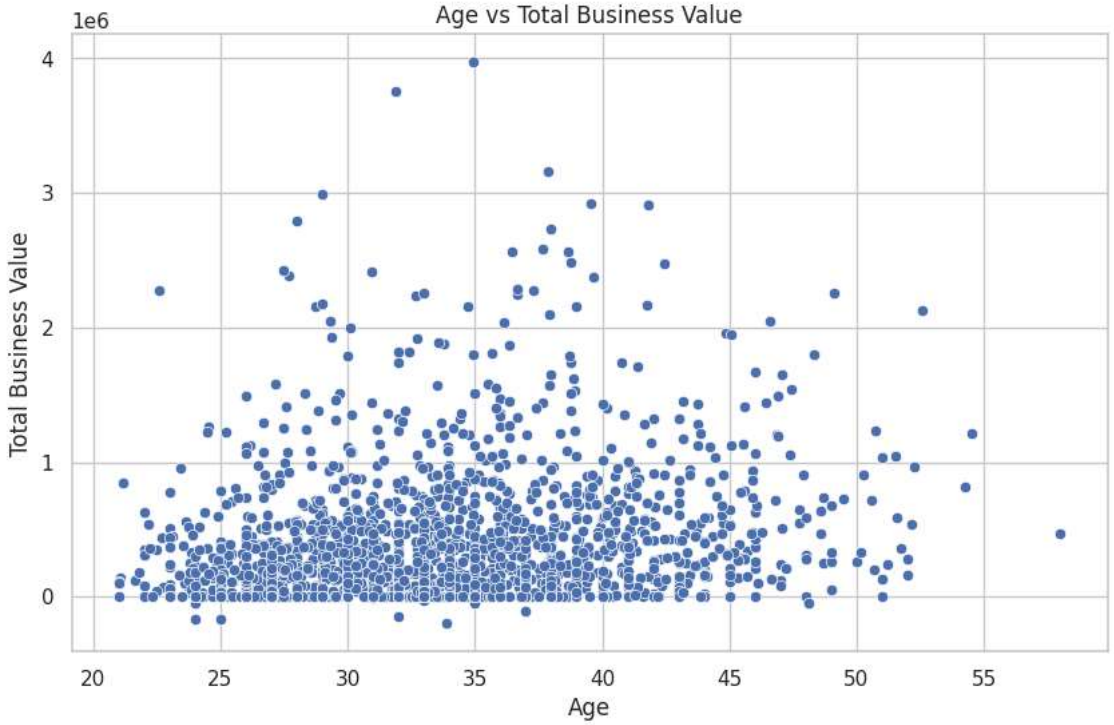
9. Count Plot of Quarterly Rating

- **Insight:** Most employees received mid-range ratings (3-4).
- **Recommendation:**
  - Re-evaluate the performance appraisal process to reduce rating bias.
  - Provide regular feedback and development goals to improve low performers and further boost high performers.

📈 Bivariate Analysis

```
# Continuous vs Continuous
# Scatter plot between Age and Income
sns.scatterplot(x='Age', y='Income', data=df)
plt.title('Age vs Income')
plt.show()

# Scatter plot between Age and Total Business Value
sns.scatterplot(x='Age', y='Total Business Value', data=agg_df)
plt.title('Age vs Total Business Value')
plt.show()
```





Bivariate Analysis: Continuous vs Continuous

1. Scatter Plot: Age vs Income

Age vs Income

Insights:

- Income tends to increase with age, peaking between the age group of 30 to 40.
- After the age of 40, income levels show a decline or stagnation for many individuals.
- There's a high density of individuals earning between 50,000 and 150,000 in the 30–40 age range.

Recommendations:

- Consider targeting training and development programs at employees aged 25–30 to accelerate income growth potential.

- Explore options for mid-career transitions or promotions around the 35–40 age bracket to maintain upward income trajectory.
- Address potential decline in productivity or role stagnation beyond age 45 with flexible roles or lateral mobility opportunities.

---

2. Scatter Plot: Age vs Total Business Value

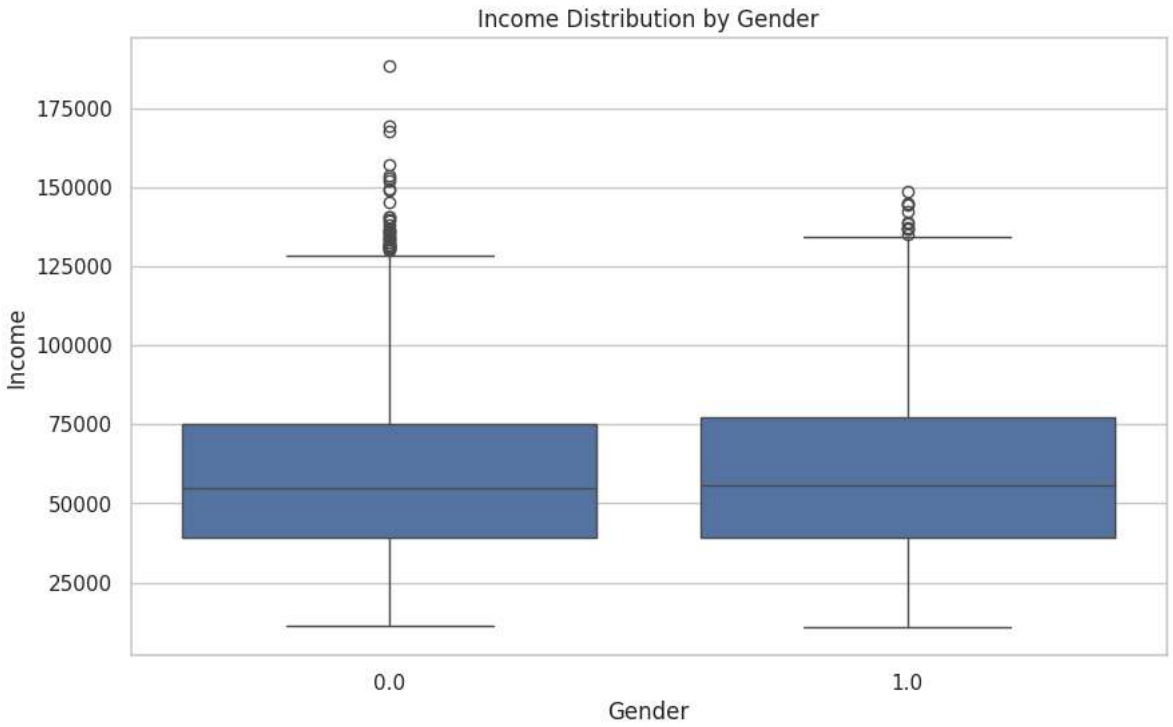
Insights:

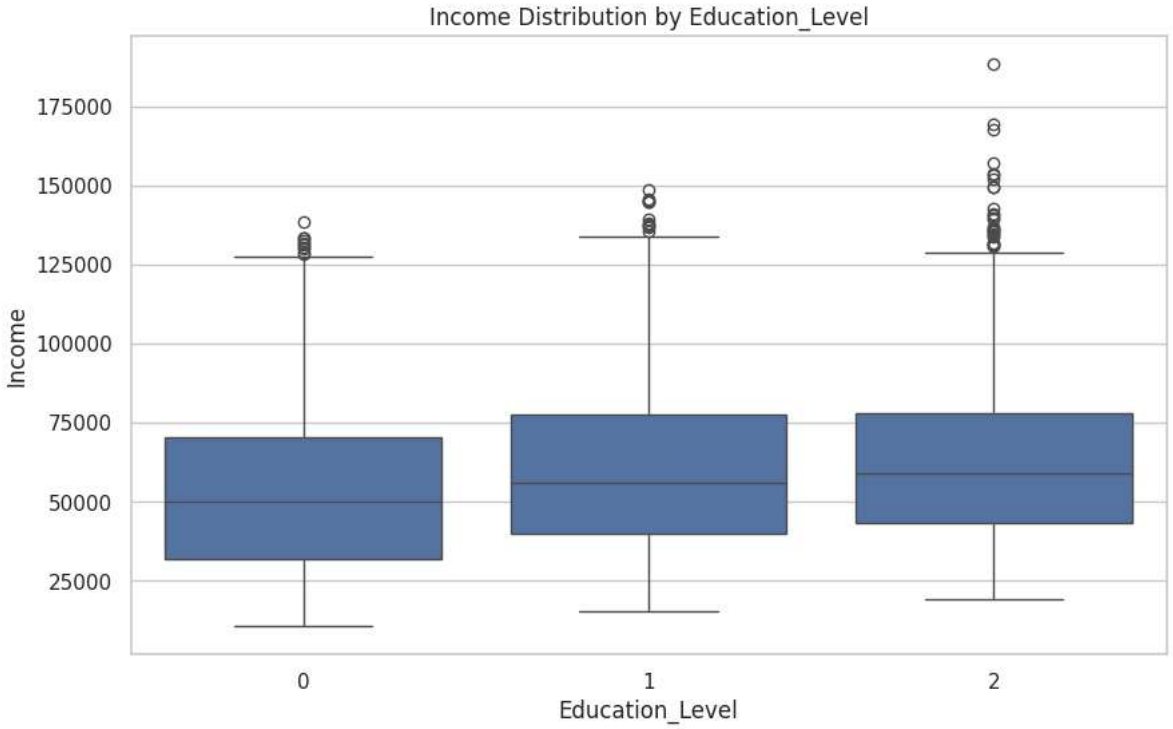
- Total Business Value is more dispersed compared to income and shows high variance across all age groups.
- Some individuals in the 30–40 age group achieve exceptionally high business value contributions.
- Outliers exist in both younger (under 30) and older (over 50) age groups, but they are rare.

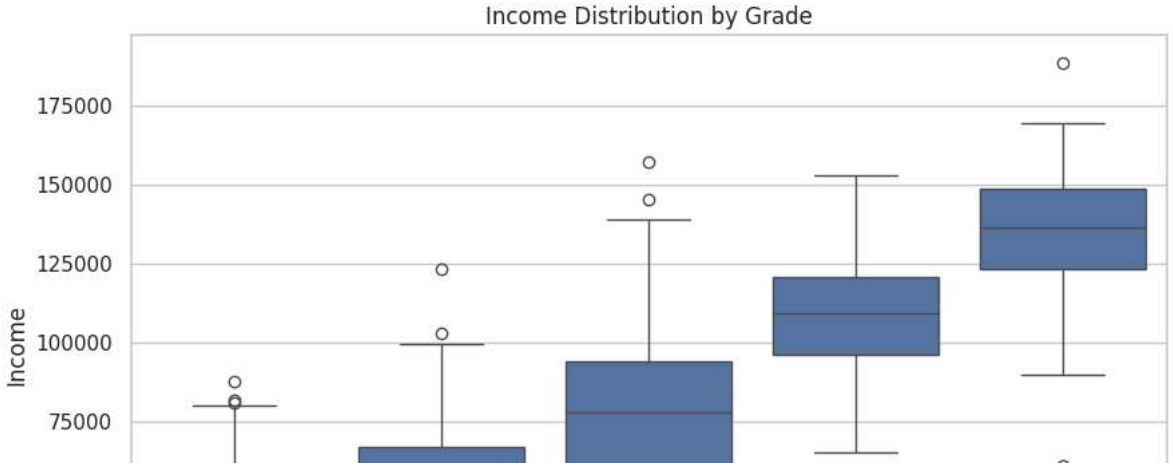
Recommendations:

- Focus on grooming entrepreneurial talent between 30–40, where peak value contribution is observed.
- Encourage mentorship programs pairing older employees (who show occasional high business value) with younger ones to share strategies.
- Implement initiatives to capture and replicate high-value behaviors seen in outlier performers regardless of age.

```
# Continuous vs Categorical
# Boxplots
for cat in ['Gender', 'Education_Level', 'Grade']:
    plt.figure()
    sns.boxplot(x=cat, y='Income', data=agg_df)
    plt.title(f'Income Distribution by {cat}')
    plt.show()
```


Income Distribution by Gender
Income
175000
150000
125000
100000
75000
50000
25000
0.0
1.0
Gender


Income Distribution by Education_Level
Income
175000
150000
125000
100000
75000
50000
25000
0
1
2
Education_Level


Income Distribution by Grade
Income
175000
150000
125000
100000
75000