# Aim: Exploratory Data Analysis (EDA) using Python

**IDE:** Tableau

# Theory:

Exploratory Data Analysis (EDA) is a critical phase in the data analysis process that involves examining and understanding data to gain insights, detect patterns, and identify relationships. As an essential precursor to formal statistical modeling, EDA helps analysts familiarize themselves with the data's characteristics, uncover potential anomalies or errors, and inform subsequent analytical decisions. At its core, EDA emphasizes visualization techniques, statistical summaries, and intuitive exploration to extract meaningful information from raw datasets.

One of the fundamental principles of EDA is data visualization, which enables analysts to visually inspect the distribution, structure, and relationships within the dataset. Graphical representations such as histograms, scatter plots, box plots, and heatmaps provide intuitive insights into the data's central tendencies, variability, and distributions across different variables. Through visual exploration, analysts can identify outliers, understand data patterns, and detect potential relationships between variables, laying the groundwork for further analysis.

Statistical summaries are another integral aspect of EDA, providing quantitative insights into the dataset's key characteristics. Measures such as mean, median, mode, variance, and standard deviation offer valuable summaries of central tendency, dispersion, and shape for numerical variables. For categorical variables, frequency tables, proportions, and percentages illuminate the distribution and prevalence of different categories. These statistical summaries complement visualizations by providing numerical context and facilitating comparisons across variables and subsets of data.

EDA also involves exploring relationships and dependencies between variables to uncover underlying patterns and associations. Correlation analysis examines the strength and direction of relationships between pairs of numerical variables, highlighting potential dependencies and predictive insights. Additionally, cross-tabulations and contingency tables assess the association between categorical variables, revealing patterns and dependencies within the data. Exploring these relationships is crucial for understanding the underlying structure of the data and informing subsequent modeling decisions.

Another key aspect of EDA is outlier detection, which involves identifying observations that deviate significantly from the overall pattern or distribution of the data. Outliers can arise due to measurement errors, data entry mistakes, or genuine anomalies in the underlying process being studied. Through visual inspection, statistical tests, and domain knowledge, analysts can identify and assess the impact of outliers on the data analysis process. Handling outliers appropriately is essential to ensure the robustness and validity of subsequent analyses and modeling efforts.

In addition to these techniques, EDA encompasses data transformation and feature engineering to prepare the data for further analysis. This may involve handling missing values, transforming variables to meet modeling assumptions, and creating new variables or features to capture additional information. By iteratively exploring, visualizing, and transforming the data, analysts can gain a comprehensive understanding of its structure, patterns, and characteristics, laying the foundation for informed decision-making and hypothesis generation.

Exploratory Data Analysis is a critical step in the data analysis process, enabling analysts to explore, visualize, and understand data to extract meaningful insights and inform subsequent analytical decisions. Through a combination of visualization techniques, statistical summaries, and intuitive exploration, analysts can uncover patterns, detect anomalies, and identify relationships within the data, laying the groundwork for further analysis and modeling. EDA serves as a powerful tool for hypothesis generation, data preparation, and decision-making, driving value and actionable insights from raw datasets.

# Pre Lab Exercise:

Write the use-case/applications for following graphs:

1. Histogram:
2. Barplot:
3. Pie Chart:
4. Scatterplot:
5. BoxPlot:

# Tasks:

Perform the following tasks:

1. Perform the EDA analysis using Python over Superstore dataset.

* **Import Necessary Library and Read The Dataset :-**

**Code :-**

import matplotlib.pyplot as plt

import numpy as np

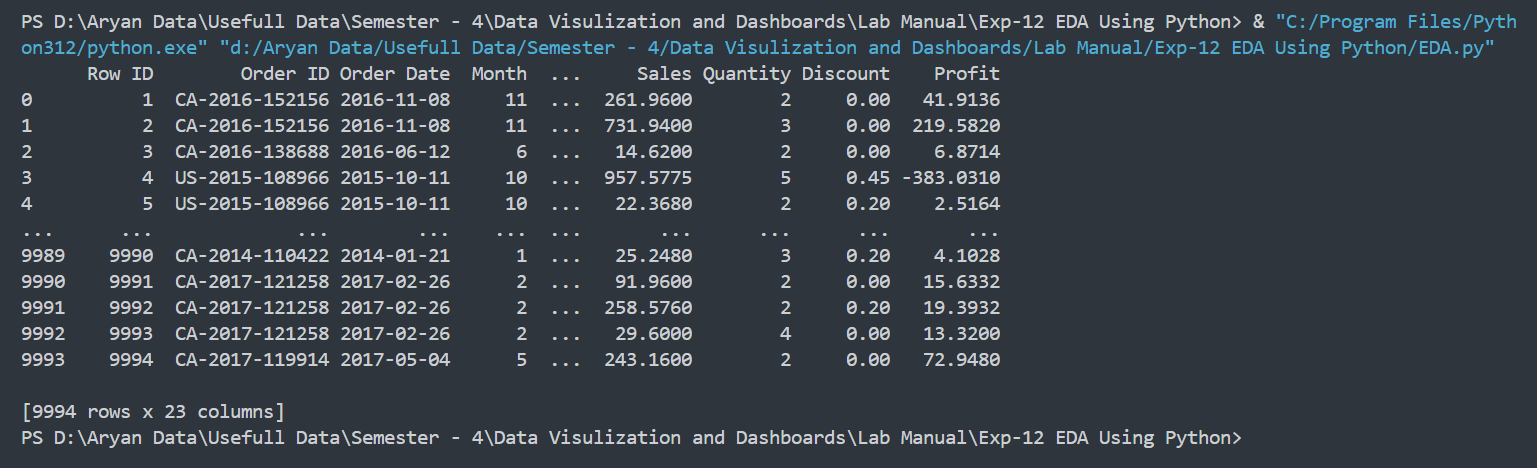
import pandas as pd

import seaborn as sns

Dataset = pd.read\_excel("./Sample - Superstore.xlsx", "Orders")

print(Dataset)

**Results :-**

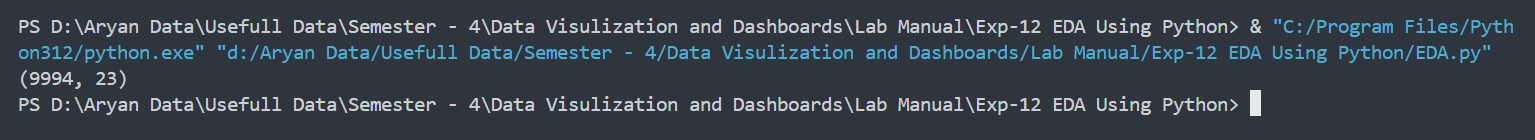
****

* **To get the shape of your Dataset :-**

**Code :-**

print(Dataset.shape)

**Result :-**

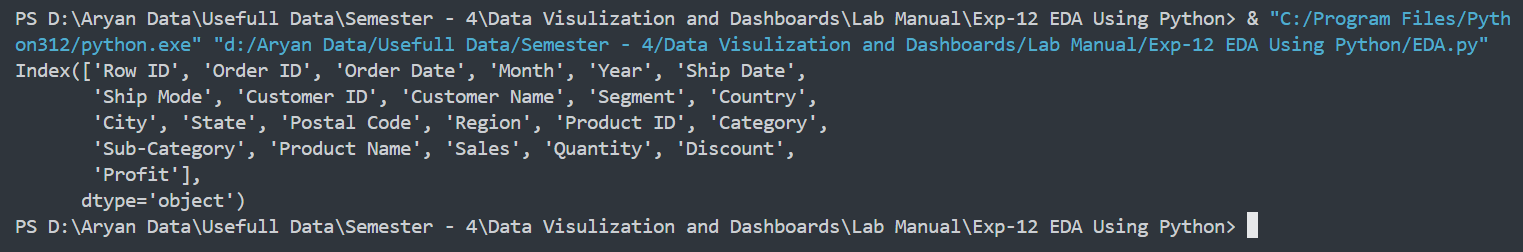


* **To get the columns of your dataset :-**

**Code :-**

print(Dataset.columns)

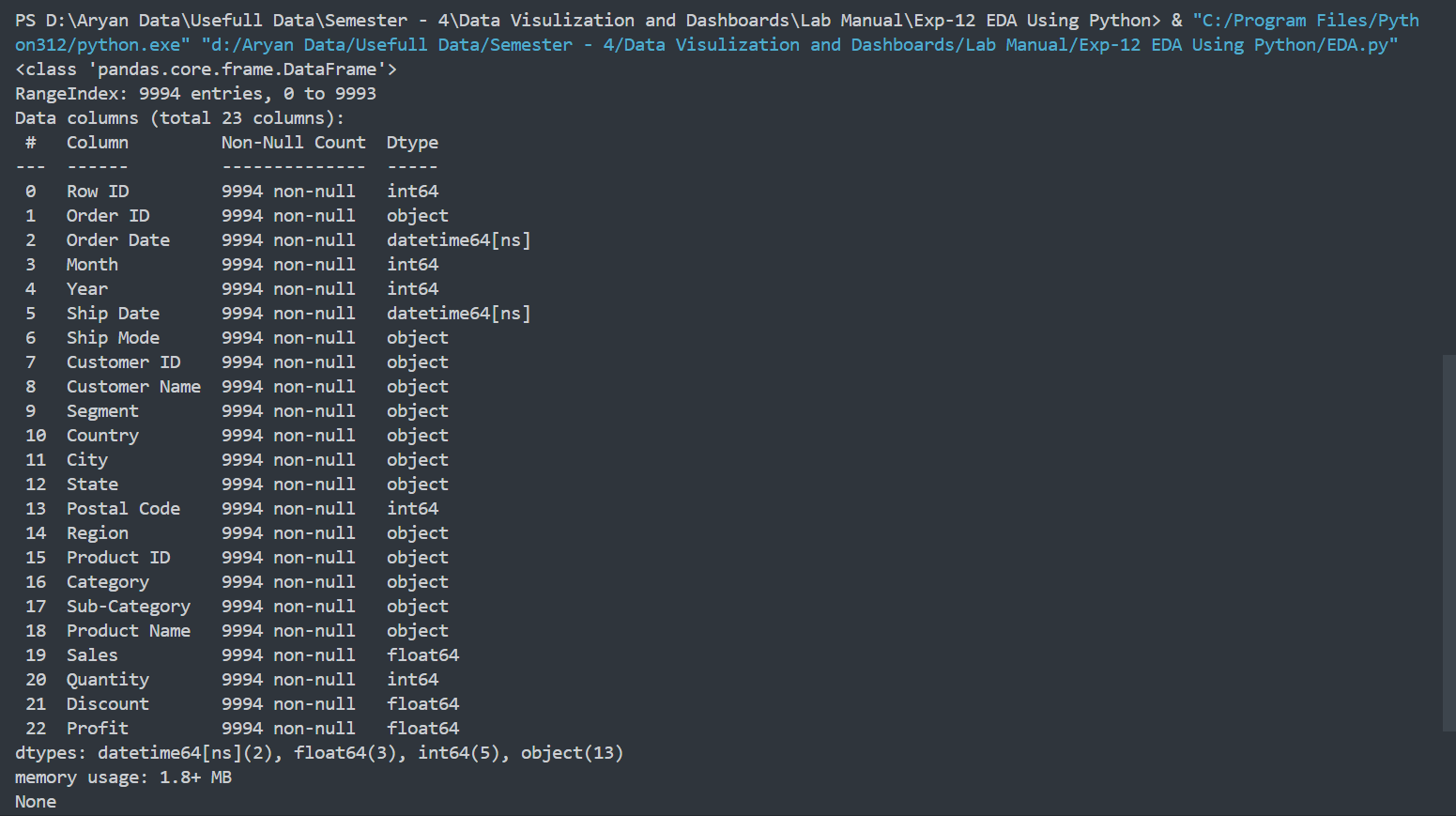
**Output :-**



* **To Get the Information of all the columns of your dataset :-**

**Code :-**

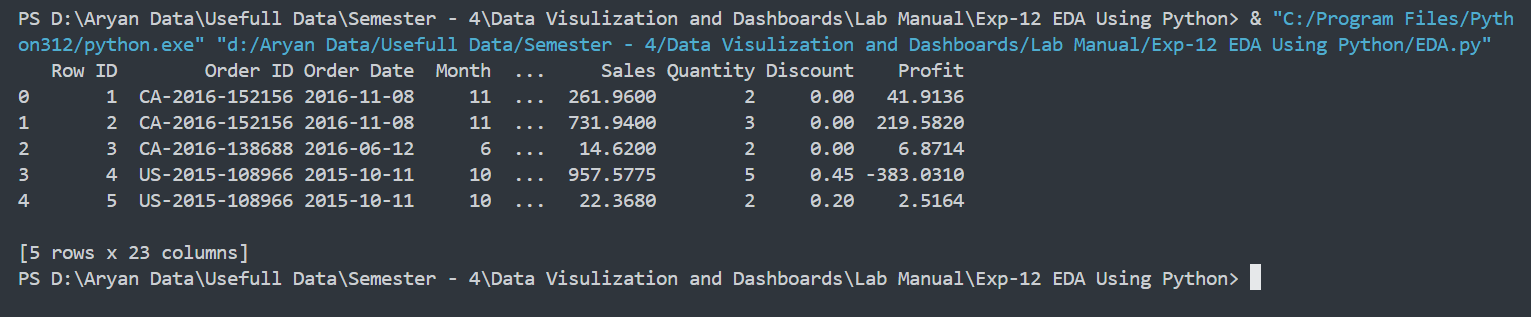
print(Dataset.info())

**Output :-**

# To Get the top rows of Datset

**Code :-**

# print(Dataset.head())

**Output :-**

* **To Get the Bottom rows of the Dataset**

**Code :-**

# print(Dataset.tail())

# Output :-

* **To Get Know if there is any null values**

**Code :-**

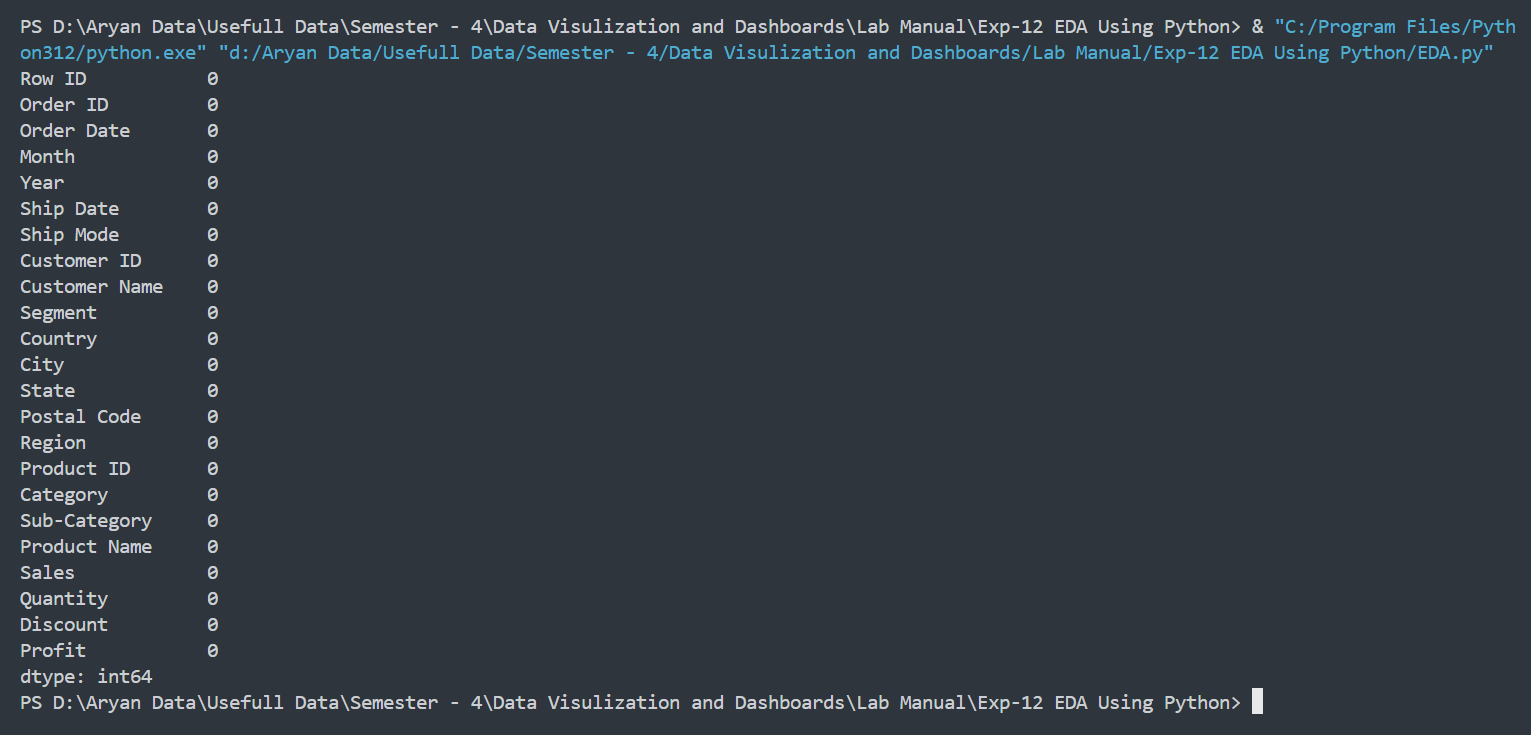
# print(Dataset.isnull())

# Output :-

# To Get the No of Null Values in the Columns

# Code :-

print(Dataset.isnull().sum())

**Output :-**

# To get the Total No of Null Values in the Dataset

# Code :-

# print(Dataset.isnull().sum().sum())

# Output :-

# To Get the No of Unique Values in each Columns

# Code :-

# print(Dataset.nunique())

# Output :-

# 

# To Drop The Column from a Dataset.

# Code :-

# Dataset = Dataset.drop(columns=["Postal Code"], axis=1)

# print(Dataset.columns)

# Output :-

# 

# To Get the State Count for Entire Dataset :-

# Code :-

# State\_Wise\_Counts = Dataset["State"].value\_counts(ascending=True)

# print(State\_Wise\_Counts)

# Output :-

# To Plot The Graph For the Same

# sns.countplot(x=Dataset["Region"], hue=Dataset["Region"])

# plt.show()

# To Plot The Count Plot for the Segment :-

# sns.countplot(x=Dataset["Segment"], hue=Dataset["Segment"])

# plt.show()

# 

# plt.figure(figsize=(50, 30))

# sns.countplot(

# x=Dataset["State"],

# hue=Dataset["State"],

# order=Dataset["State"].value\_counts(ascending=True).index,

# )

# plt.show()

# 

plt.figure(figsize=(50, 30))

sns.countplot(

data=Dataset,

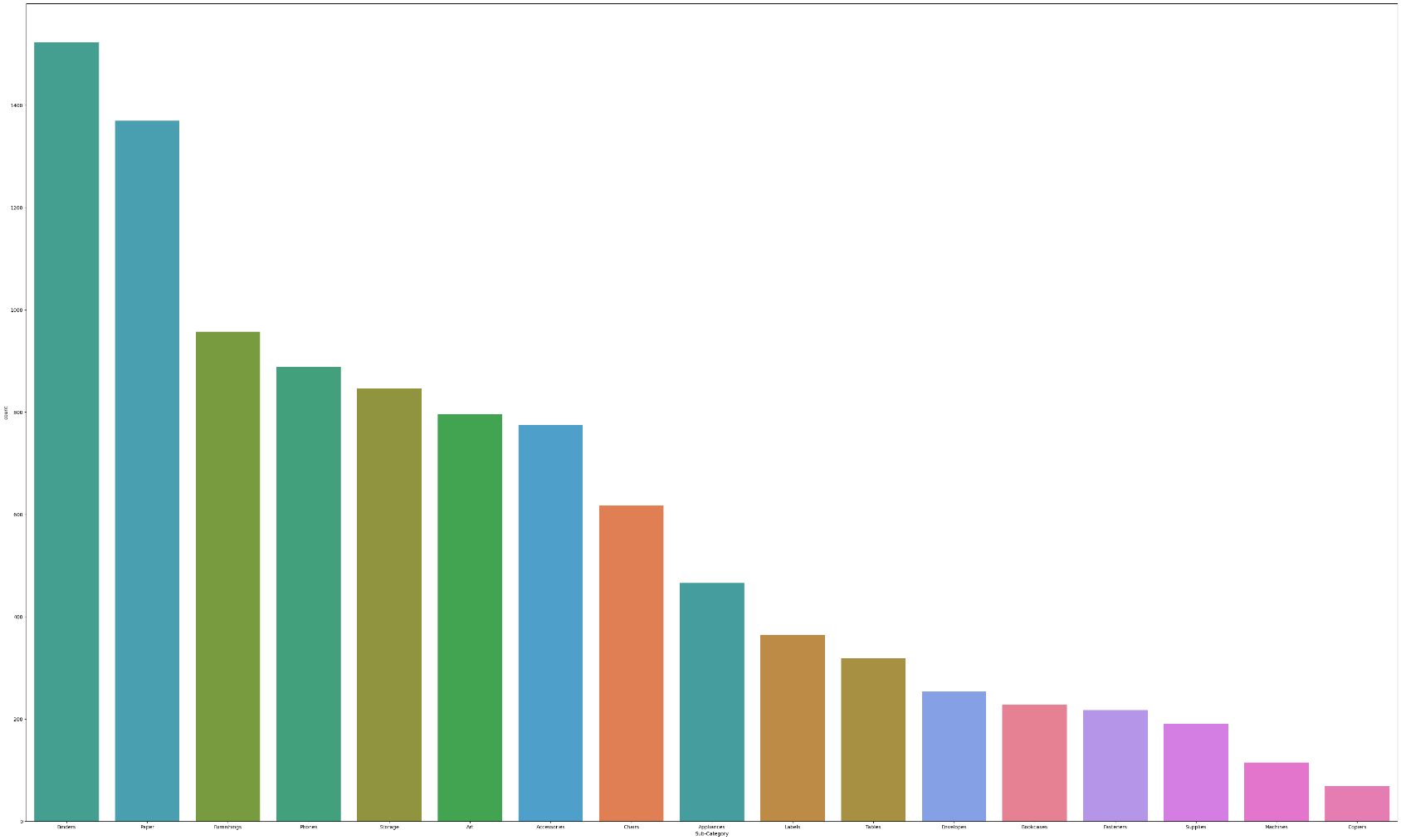
x="Sub-Category",

hue="Sub-Category",

order=Dataset["Sub-Category"].value\_counts().index,

)

plt.show()



# fig = plt.figure(figsize=(10, 4))

# plt.subplot(1, 2, 1)

# Dataset.groupby("Region")["Sales"].sum().plot.pie(autopct="%1.01f%%")

# plt.title("Region vs Sales")

# plt.subplot(1, 2, 2)

# Dataset.groupby("Region")["Profit"].sum().plot.pie(autopct="%1.01f%%")

# plt.title("Region vs Profit")

# plt.show()

# 

# fig = plt.figure(figsize=(10, 4))

# plt.subplot(1, 2, 1)

# Dataset.groupby("Segment")["Sales"].sum().plot.pie(autopct="%1.01f%%")

# plt.title("Segment vs Sales")

# plt.subplot(1, 2, 2)

# Dataset.groupby("Segment")["Profit"].sum().plot.pie(autopct="%1.01f%%")

# plt.title("Segment vs Profit")

# plt.show()

# 

# non\_numeric\_columns = Dataset.select\_dtypes(exclude=["number"]).columns

# Dataset\_numeric = Dataset.drop(columns=non\_numeric\_columns)

# sns.heatmap(Dataset\_numeric.corr(), annot=True)

# 

# fig = plt.figure(figsize=(10, 4))

# sns.pairplot(Dataset)

# plt.show()

# 

# Observation and Result Analysis:

Write your inference corresponding to each of the analysis

# Post Lab Exercise:

**Exercise 1:** Perform the following EDA analysis using Python over the Book Shop dataset.

**Pre-Requisites:-**

**Import Dataset :-**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

Book = pd.read\_excel('./Bookshop.xlsx' , 'Book')

Author = pd.read\_excel("./Bookshop.xlsx", "Author")

Author["Full\_Name"] = Author["First Name"] + " " + Author["Last Name"]

Info = pd.read\_excel('./Bookshop.xlsx' , 'Info')

Info['BookID'] = Info['BookID1'].astype(str) + Info['BookID2'].astype(str)

Award = pd.read\_excel('./Bookshop.xlsx' , 'Award')

Checkouts = pd.read\_excel('./Bookshop.xlsx' , 'Checkouts')

Edition = pd.read\_excel('./Bookshop.xlsx' , 'Edition')

Publisher = pd.read\_excel('./Bookshop.xlsx' , 'Publisher')

Ratings = pd.read\_excel('./Bookshop.xlsx' , 'Ratings')

Series = pd.read\_excel('./Bookshop.xlsx' , 'Series')

Sales\_Q1 = pd.read\_excel('./Bookshop.xlsx' , 'Sales Q1')

Sales\_Q2 = pd.read\_excel('./Bookshop.xlsx' , 'Sales Q2')

Sales\_Q3 = pd.read\_excel('./Bookshop.xlsx' , 'Sales Q3')

Sales\_Q4 = pd.read\_excel("./Bookshop.xlsx", "Sales Q4")

1. Who is the most popular or famous author (a. based on ratings; b. based on sales)

**Code :-**

# Based on Ratings

DS1 = pd.merge(Author, Book, on="AuthID")

DS1 = pd.merge(DS1, Ratings, on="BookID")

DS1["Full\_Name"] = DS1["First Name"] + " " + DS1["Last Name"]

df1 = (DS1.groupby("Full\_Name")["Rating"].mean().reset\_index().sort\_values(by="Rating", ascending=False))[:10]

plt.figure(figsize=(30, 15))

sns.barplot(x=df1["Full\_Name"], y=df1["Rating"] , hue = df1['Full\_Name'])

for i, value in enumerate(df1['Rating']):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

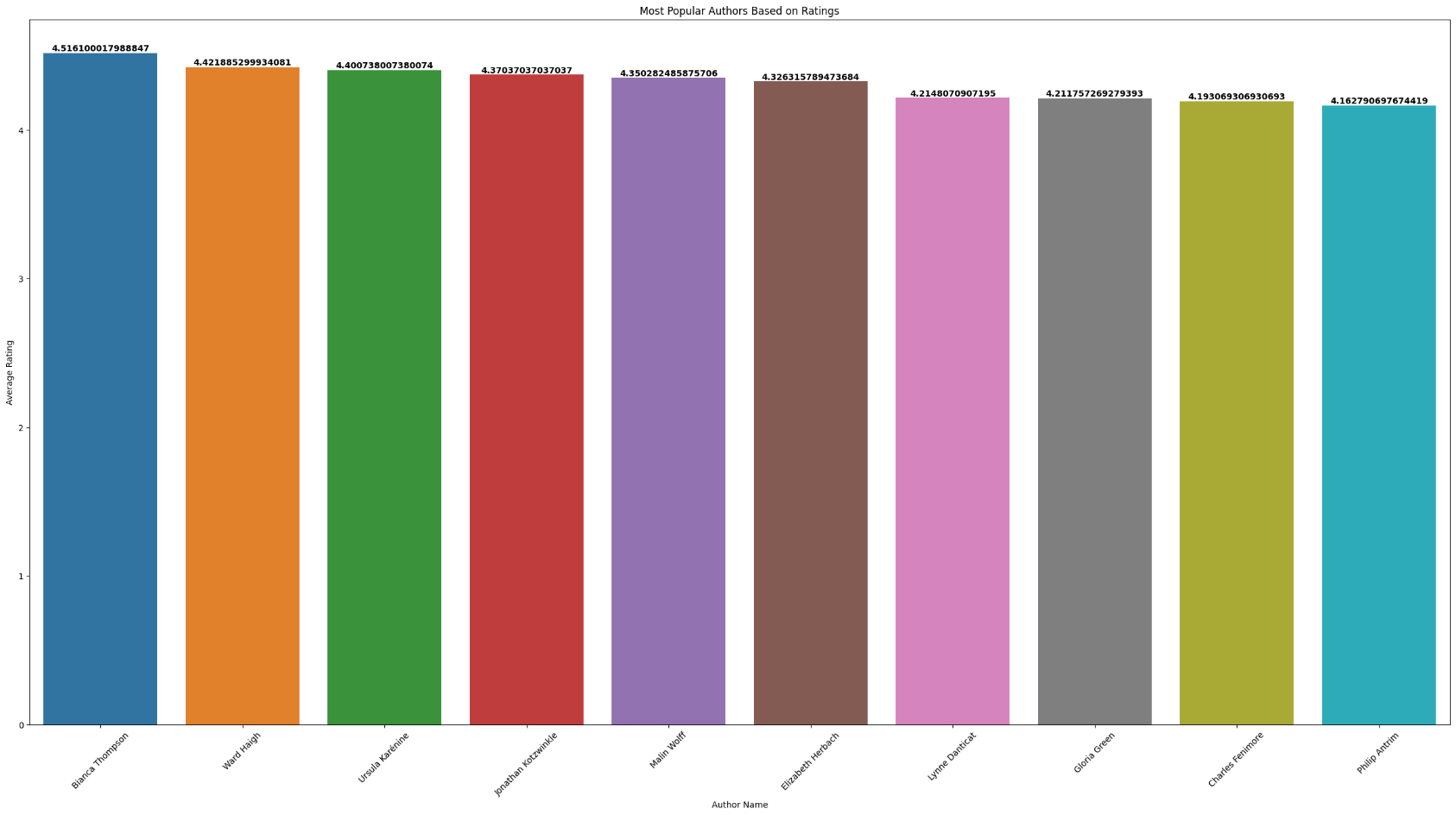
plt.xlabel("Author Name")

plt.ylabel("Average Rating")

plt.title("Most Popular Authors Based on Ratings")

plt.xticks(rotation=45)

plt.show()

**Output :-**

**Code :-**

Sales = pd.concat([Sales\_Q1.value\_counts("ISBN").reset\_index(),Sales\_Q2.value\_counts("ISBN").reset\_index(),Sales\_Q3.value\_counts("ISBN").reset\_index(),Sales\_Q4.value\_counts('ISBN').reset\_index(),],ignore\_index=True,)

Sales = pd.merge(Sales , Edition , on="ISBN")

Sales['Total Price'] = Sales['count'] \* Sales['Price']

Sales = Sales[['ISBN' , 'Total Price']].sort\_values(by='Total Price' , ascending=False)

DS1A = pd.merge(Book, Edition, on="BookID")

DS1A = pd.merge(DS1A, Author, on="AuthID")

DS1A = pd.merge(DS1A, Sales , on="ISBN")

DS1A = DS1A.groupby('Full\_Name')['Total Price'].sum().reset\_index().sort\_values(by="Total Price", ascending=False)[:10]

df1a = DS1A

plt.figure(figsize=(30, 15))

sns.barplot(x=df1a["Full\_Name"], y=df1a["Total Price"], hue=df1a["Full\_Name"])

for i, value in enumerate(df1a["Total Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Author Name")

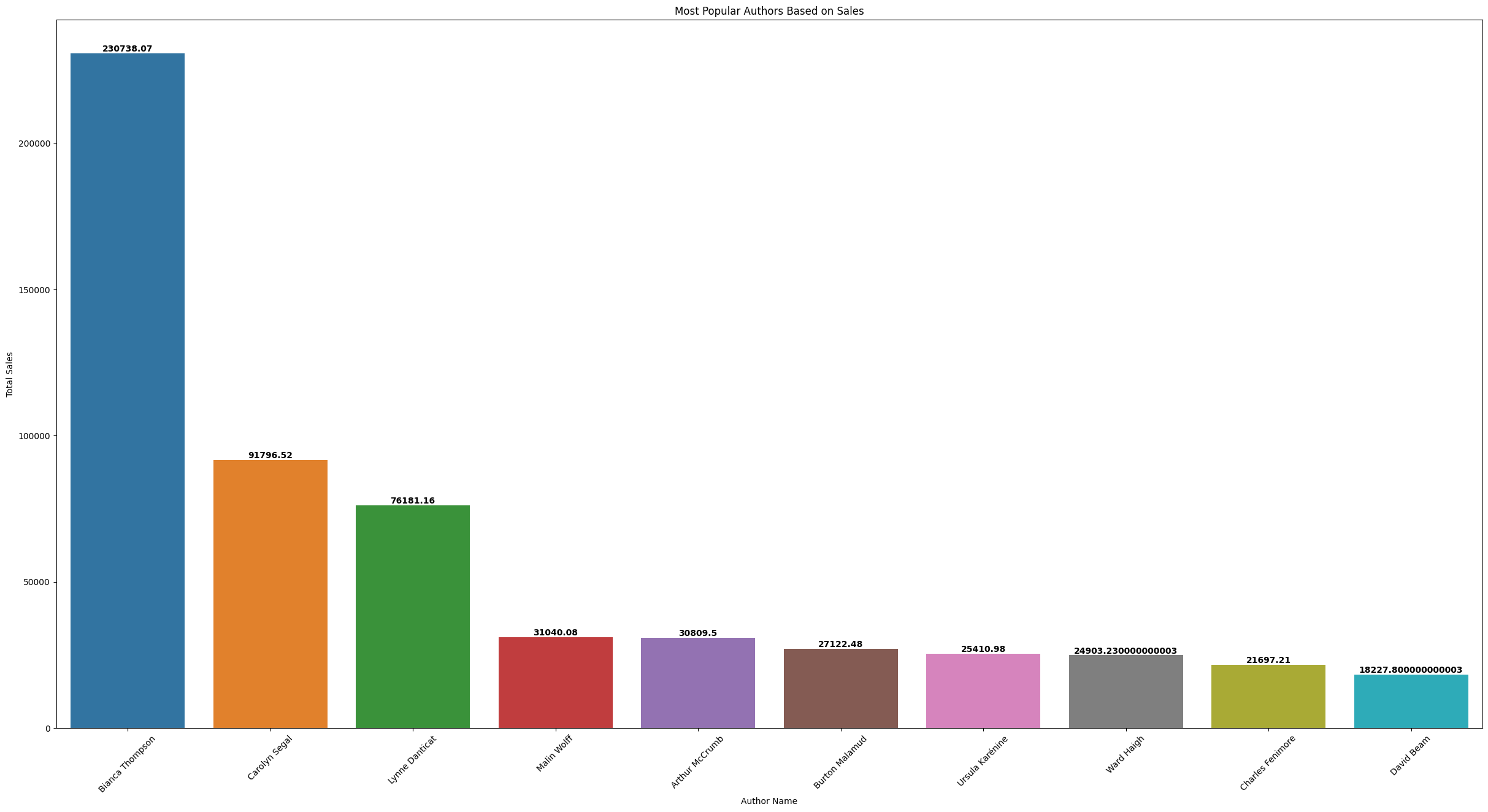
plt.ylabel("Total Sales")

plt.title("Most Popular Authors Based on Sales")

plt.xticks(rotation=45)

plt.show()

**Output :-**



1. Top-3 countries with the highest number of authors

**Code :-**

df2 = Author['Country of Residence'].value\_counts().reset\_index()

plt.figure(figsize=(30, 25))

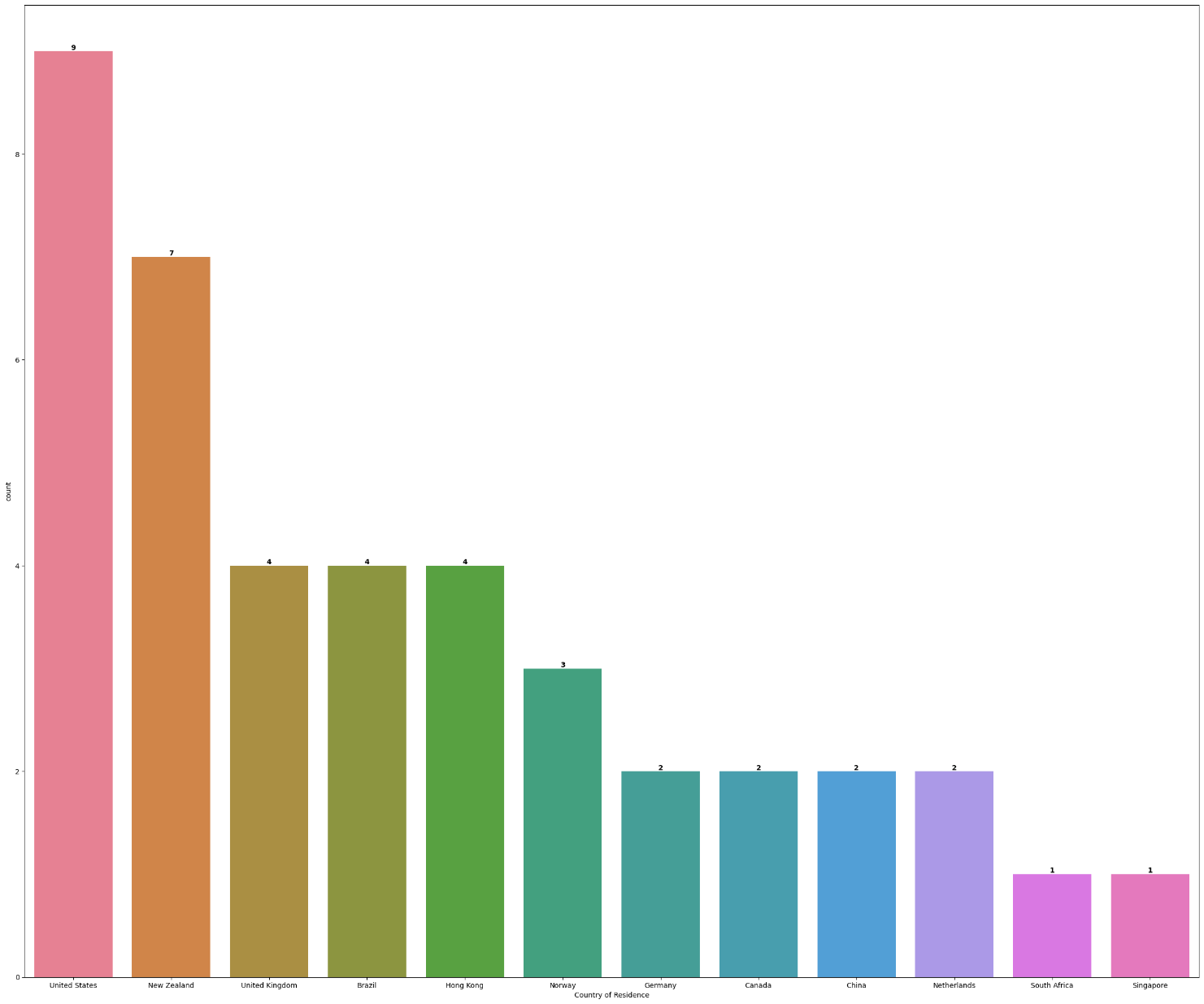
sns.barplot(data=df2, x="Country of Residence", y = "count" , hue="Country of Residence")

for i, value in enumerate(Author["Country of Residence"].value\_counts()):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.show()

**Output :-**

****

1. Who is the hardworking author in terms of working per day

**Code :-**

df3 = Author.sort\_values(by="Hrs Writing per Day" , ascending=False)

df3 = df3[['Full\_Name' , "Hrs Writing per Day"]]

plt.figure(figsize=(30, 25))

sns.barplot(data=df3, x="Full\_Name", y="Hrs Writing per Day", hue="Full\_Name")

for i, value in enumerate(df3["Hrs Writing per Day"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Author Name")

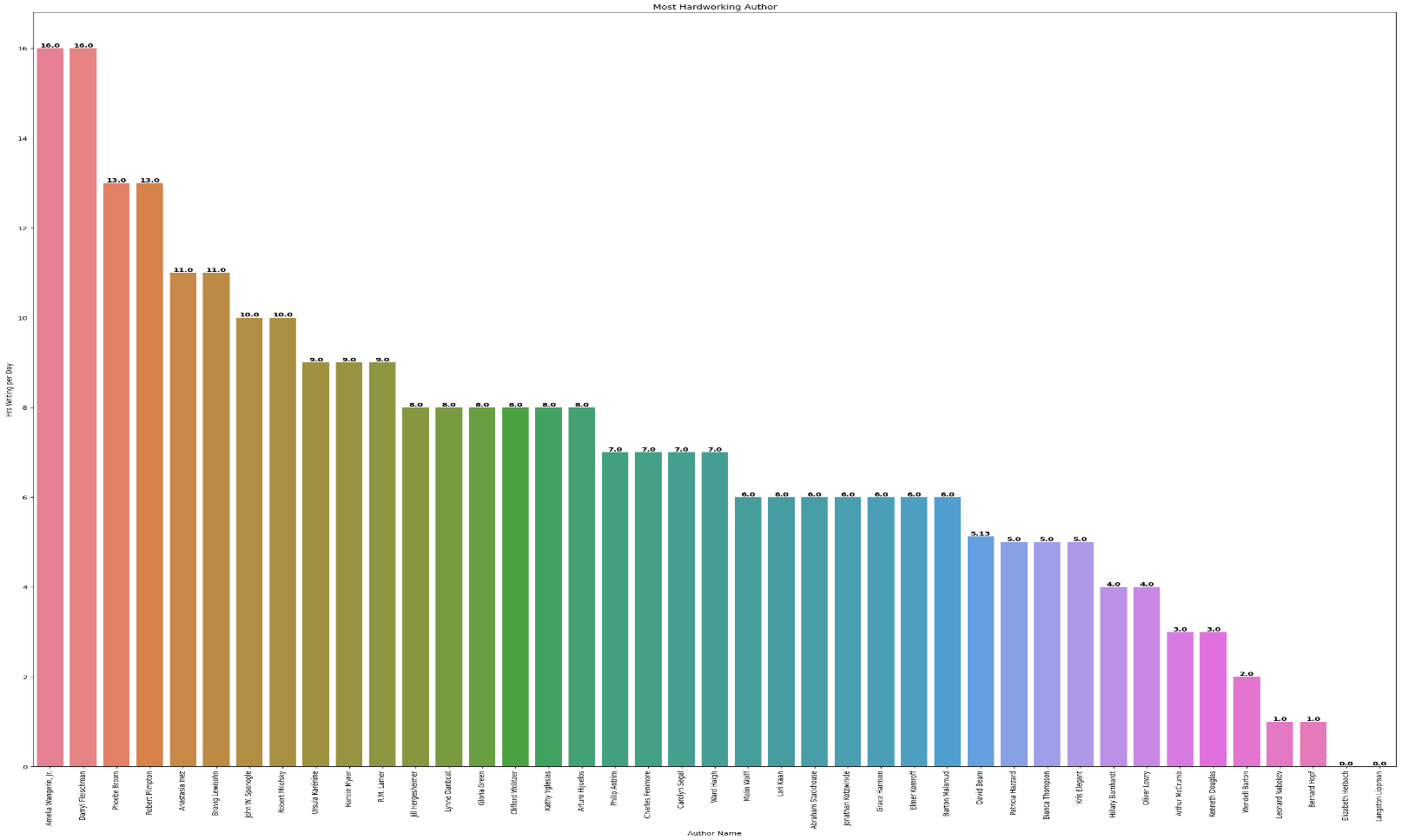
plt.ylabel("Hrs Writing per Day")

plt.title("Most Hardworking Author")

plt.xticks(rotation=90)

plt.show()

**Output :-**

****

1. Top 5 authors having the highest average ratings

**Code :-**

DS4 = pd.merge(Book , Ratings , on = "BookID")

DS4 = pd.merge(DS4 , Author , on = "AuthID")

df\_4 = DS4.groupby('Full\_Name')['Rating'].mean().sort\_values(ascending=False).reset\_index()[:5]

plt.figure(figsize=(30, 15))

sns.barplot(x=df\_4["Full\_Name"], y=df\_4["Rating"] , hue = df\_4['Full\_Name'])

for i, value in enumerate(df\_4['Rating']):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

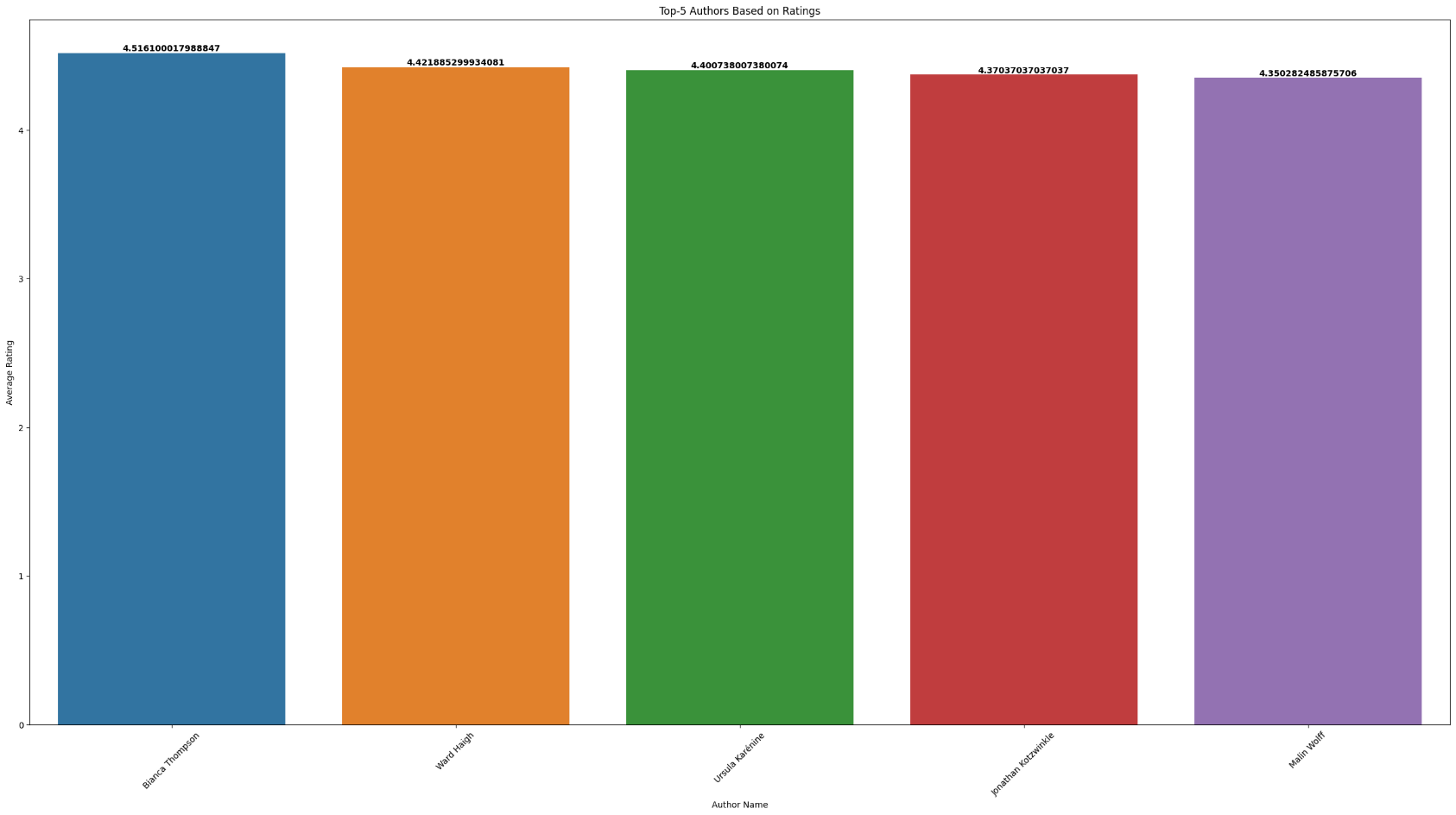
plt.xlabel("Author Name")

plt.ylabel("Average Rating")

plt.title("Top-5 Authors Based on Ratings")

plt.xticks(rotation=45)

plt.show()

**Output :-**

1. Top 5 Books having the highest average ratings

**Code :-**

DS5 = pd.merge(Book , Ratings , on='BookID')

df5 = pd.DataFrame(DS5.groupby('Title')['Rating'].mean().reset\_index()).sort\_values(by = 'Rating' , ascending=False)[:5]

plt.figure(figsize=(30, 15))

sns.barplot(x=df5["Title"], y=df5["Rating"] , hue = df5['Title'])

for i, value in enumerate(df5["Rating"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Book Title")

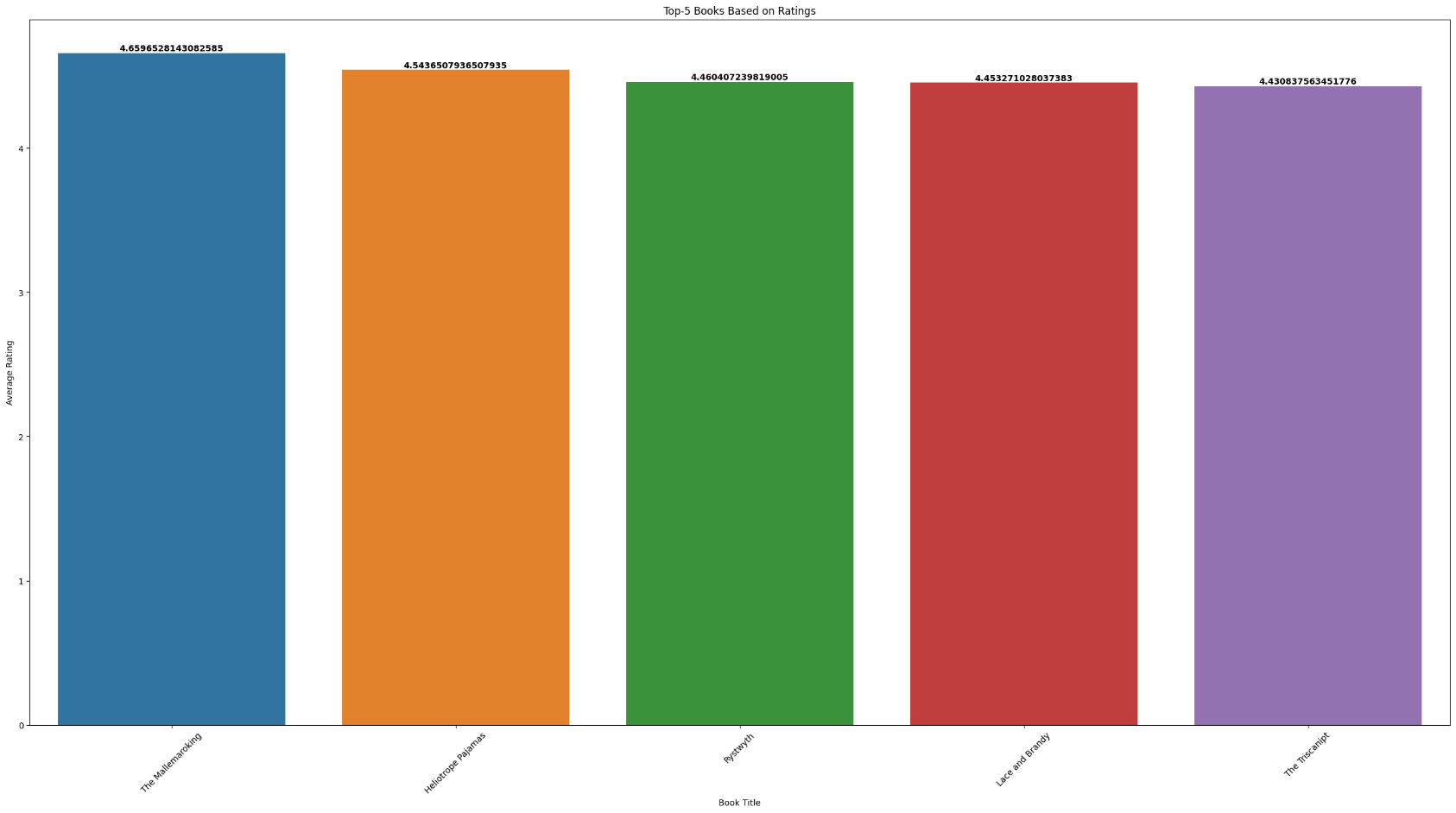
plt.ylabel("Average Rating")

plt.title("Top-5 Books Based on Ratings")

plt.xticks(rotation=45)

plt.show()

**Output :-**

****

1. Top-5 authors spending maximum amount on marketing

**Code :-**

DS6 = pd.merge(Book, Author, on="AuthID")

DS6 = pd.merge(DS6, Edition, on="BookID")

DS6 = pd.merge(DS6, Publisher, on="PubID")

df6 = pd.DataFrame(DS6.groupby("Full\_Name")["Marketing Spend"].sum().reset\_index()).sort\_values(by="Marketing Spend", ascending=False)[:5]

plt.figure(figsize=(30, 15))

sns.barplot(x=df6["Full\_Name"], y=df6["Marketing Spend"], hue=df6["Full\_Name"])

for i, value in enumerate(df6["Marketing Spend"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

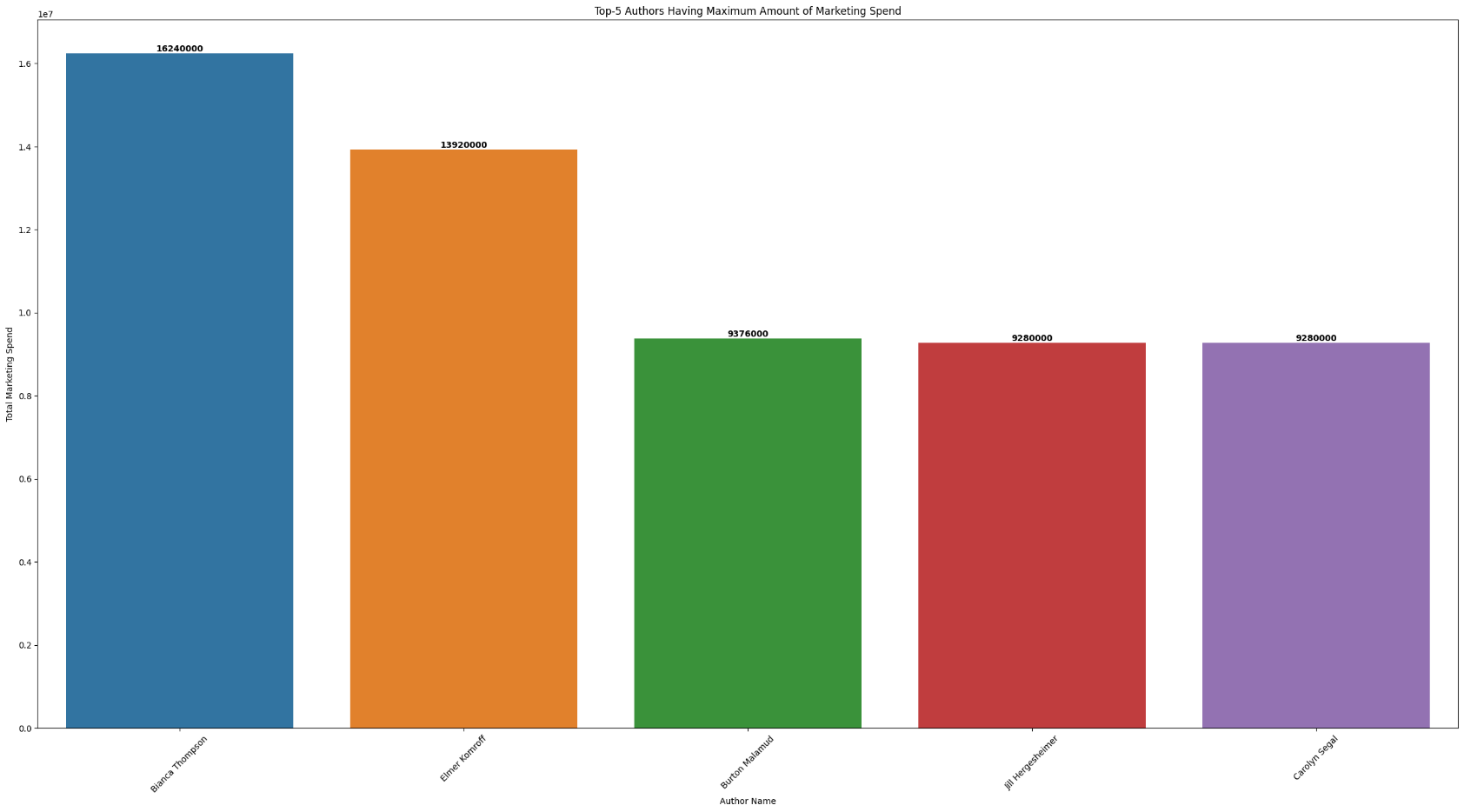
plt.xlabel("Author Name")

plt.ylabel("Total Marketing Spend")

plt.title("Top-5 Authors Having Maximum Amount of Marketing Spend")

plt.xticks(rotation=45)

plt.show()

**Output :-**

1. Top 5 authors writing the highest total number of pages

**Code :-**

DS7 = pd.merge(Book, Author, on="AuthID")

DS7 = pd.merge(DS7, Edition, on="BookID")

df7 = pd.DataFrame(DS7.groupby("Full\_Name")["Pages"].sum().reset\_index()).sort\_values(

by="Pages", ascending=False

)[:5]

plt.figure(figsize=(30, 15))

sns.barplot(x=df7["Full\_Name"], y=df7["Pages"], hue=df7["Full\_Name"])

for i, value in enumerate(df7["Pages"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

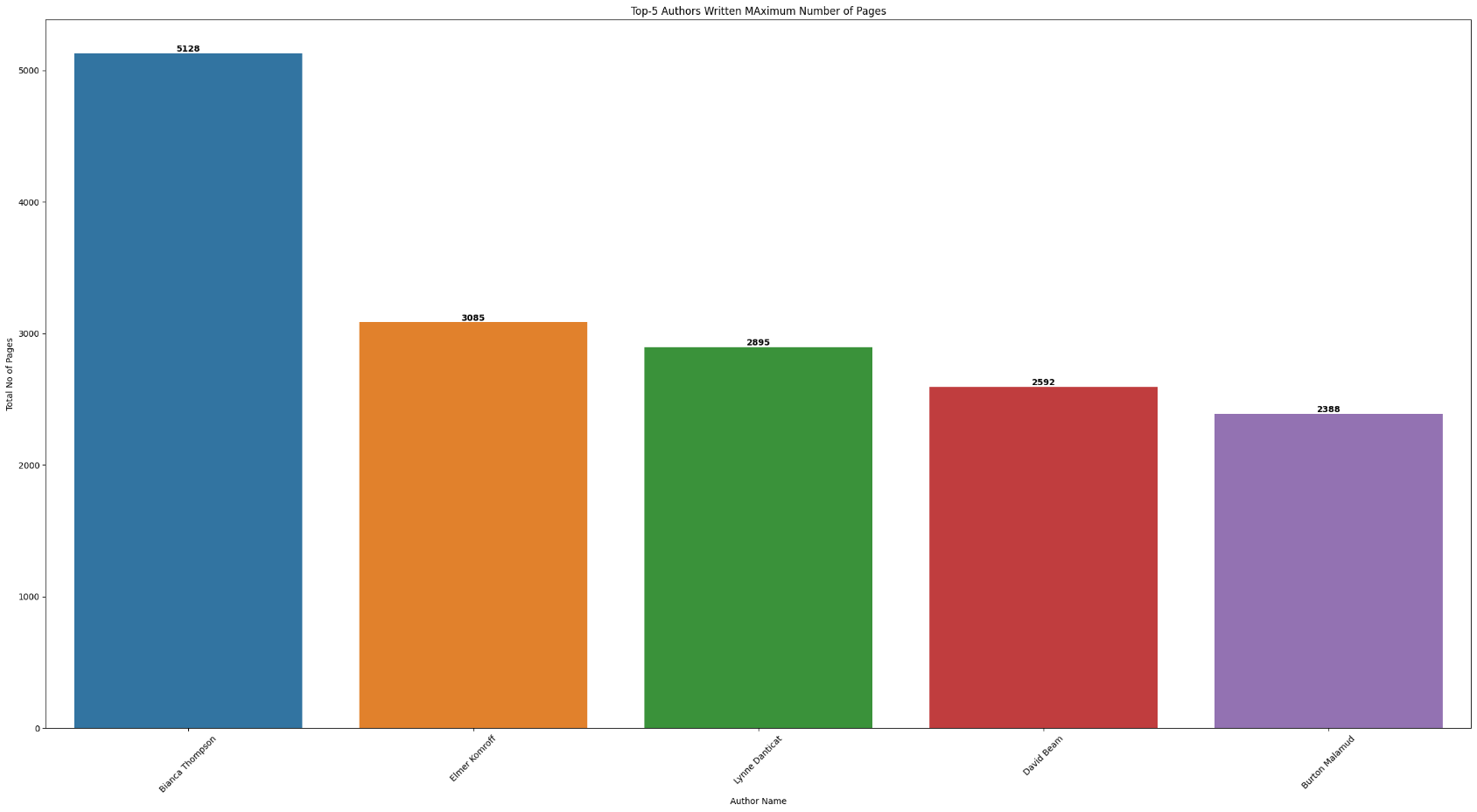
plt.xlabel("Author Name")

plt.ylabel("Total No of Pages")

plt.title("Top-5 Authors Written MAximum Number of Pages")

plt.xticks(rotation=45)

plt.show()

**Output :-**

1. Top-5 Publication House (in terms of count of books published)

**Code :-**

DS8 = pd.merge(Edition, Publisher, on="PubID")

df8 = DS8["Publishing House"].value\_counts().reset\_index()

plt.figure(figsize=(30, 25))

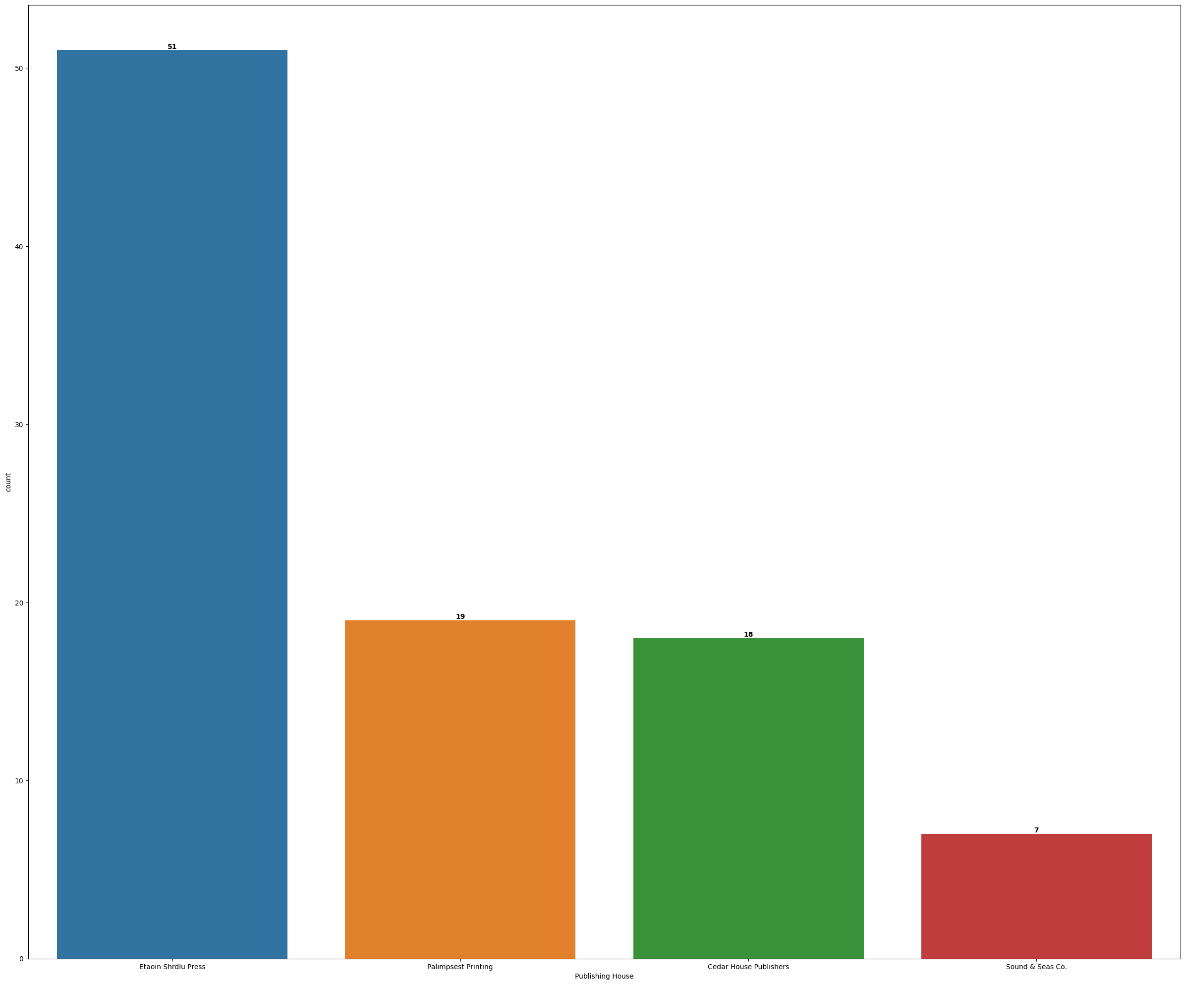
sns.barplot(data=df8, x="Publishing House", y="count", hue="Publishing House")

for i, value in enumerate(DS8["Publishing House"].value\_counts()):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.show()

**Output :-**

****

1. Top-5 publication house with highest-priced books

**Code :-**

DS9 = pd.merge(Edition, Publisher, on="PubID")

df9 = pd.DataFrame(

DS9.groupby("Publishing House")["Price"].max().reset\_index()

).sort\_values(by="Price" , ascending= False)

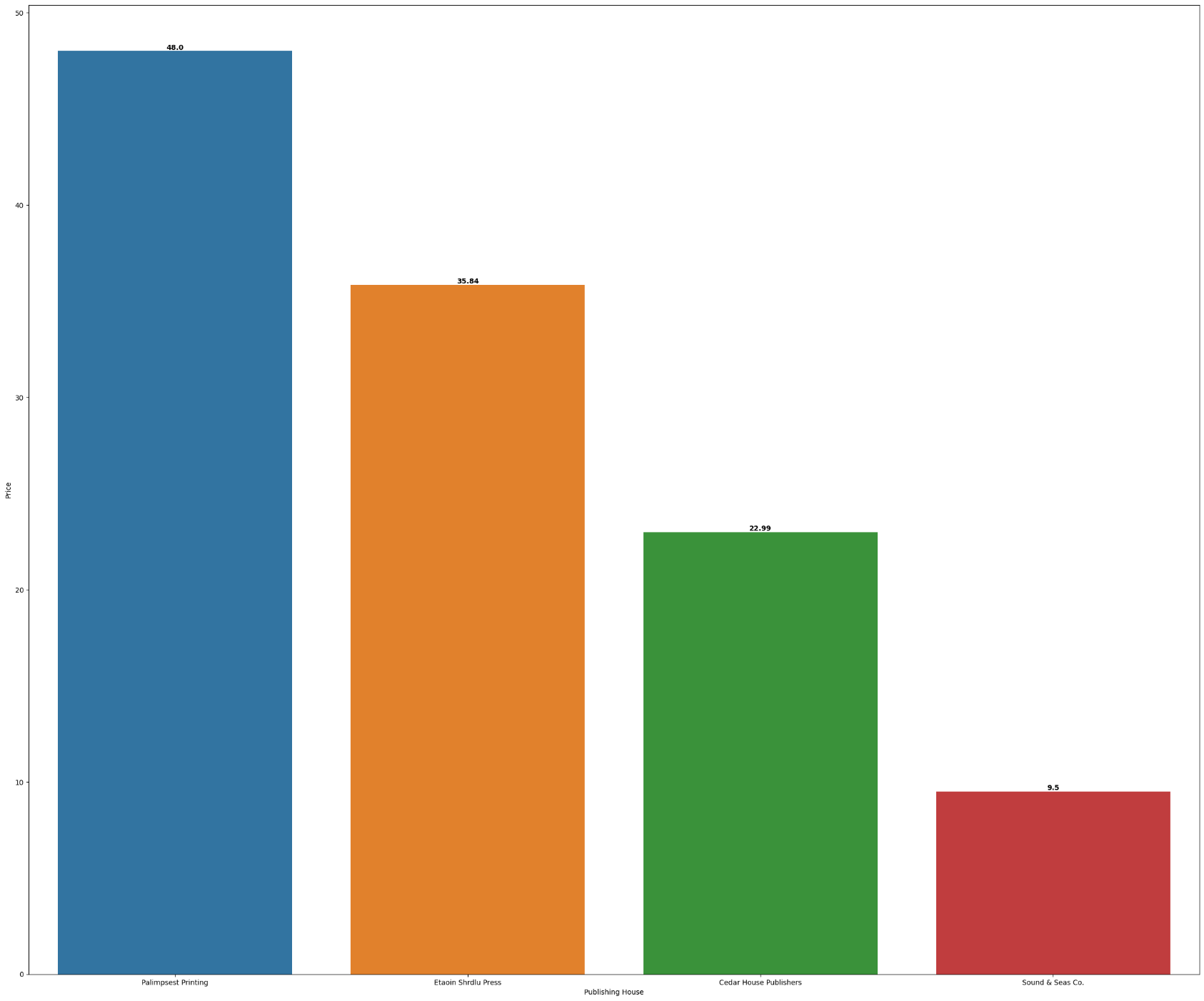
plt.figure(figsize=(30, 25))

sns.barplot(data= df9, x="Publishing House", y="Price", hue="Publishing House")

for i, value in enumerate(df9["Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.show()

**Output :-**

1. Total number of books in each genre

**Code :-**

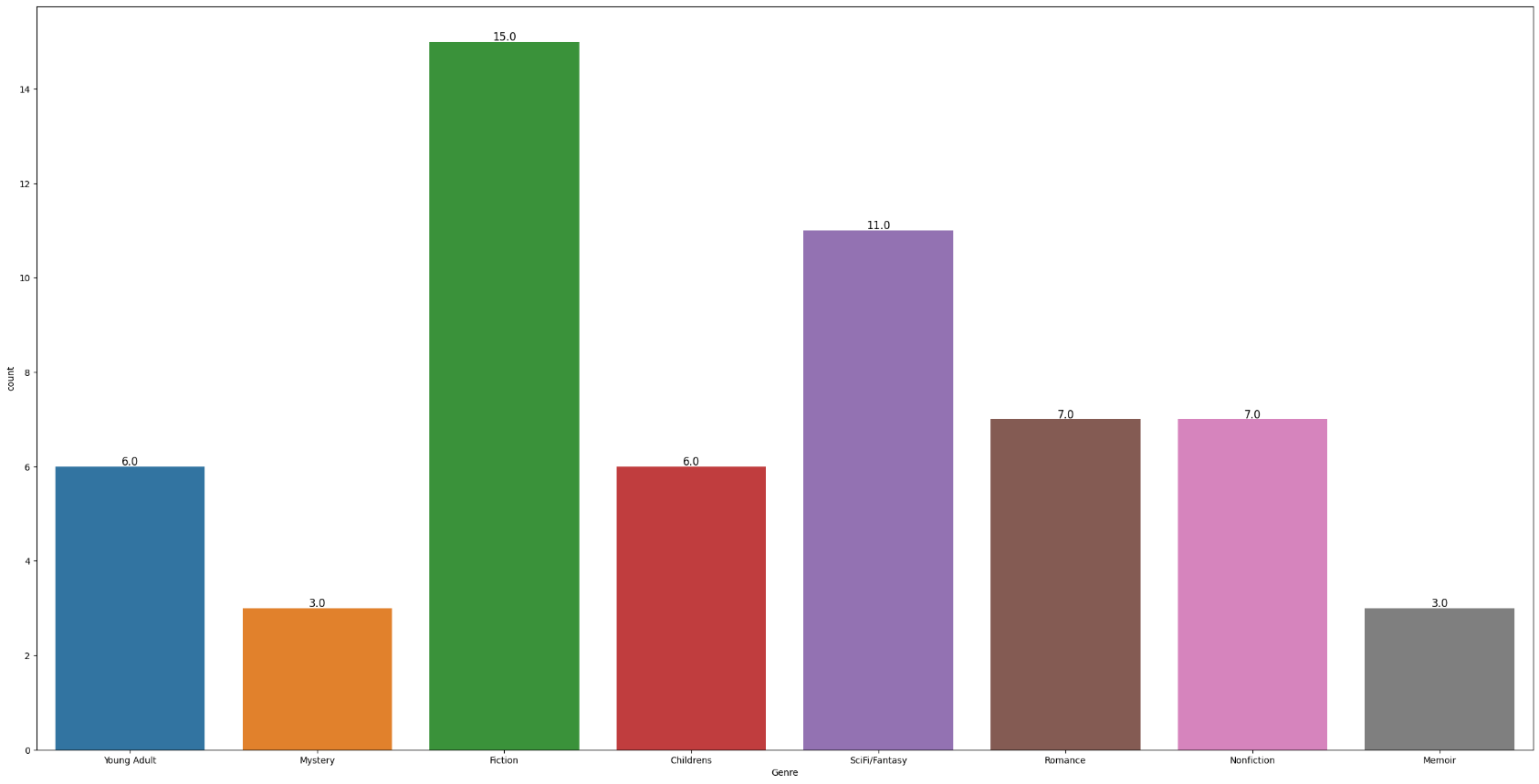
plt.figure(figsize=(30, 15))

ax = sns.countplot(data=Info, x="Genre", hue="Genre")

for p in ax.patches:

ax.annotate(f"{p.get\_height()}",(p.get\_x() + p.get\_width() / 2.0, p.get\_height()),ha="center",va="center",fontsize=12,color="black",xytext=(0, 5),textcoords="offset points",)

plt.show()

**Output:-**

1. Top-5 publication house with the highest sales in each quarter

Code:-

DS11 = pd.concat([Sales\_Q1.value\_counts("ISBN").reset\_index(),Sales\_Q2.value\_counts("ISBN").reset\_index(),Sales\_Q3.value\_counts("ISBN").reset\_index(),Sales\_Q4.value\_counts("ISBN").reset\_index(),],ignore\_index=True)

DS11 = pd.merge(DS11, Edition, on="ISBN")

DS11["Total Price"] = DS11["count"] \* DS11["Price"]

DS11 = pd.merge(DS11, Publisher, on="PubID")

df11 = pd.DataFrame(DS11.groupby("Publishing House")["Total Price"].sum()).reset\_index().sort\_values(by = "Total Price" , ascending= False)

plt.figure(figsize=(30, 15))

sns.barplot(

x=df11["Publishing House"], y=df11["Total Price"], hue=df11["Publishing House"]

)

for i, value in enumerate(df11["Total Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

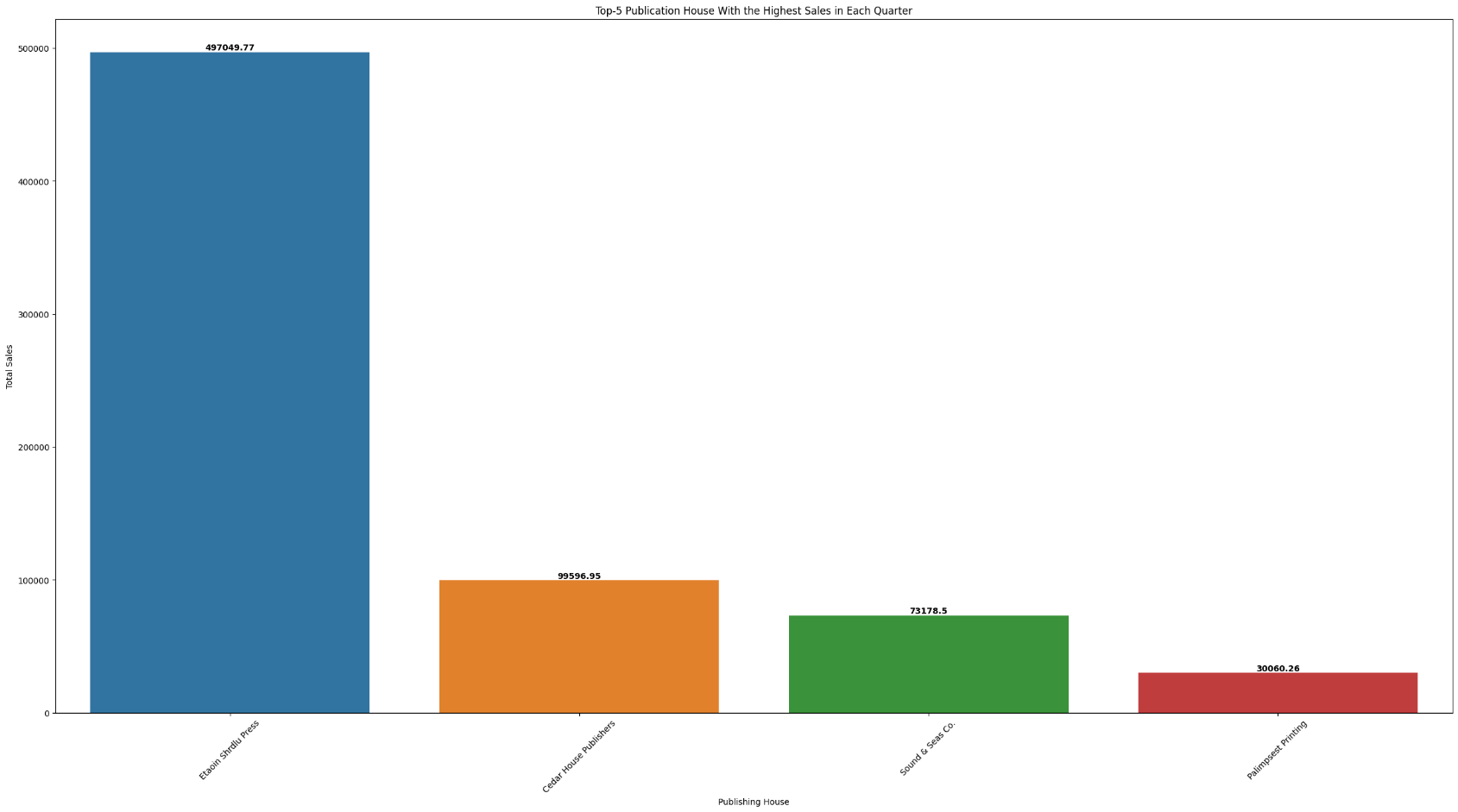
plt.xlabel("Publishing House")

plt.ylabel("Total Sales")

plt.title("Top-5 Publication House With the Highest Sales in Each Quarter")

plt.xticks(rotation=45)

plt.show()

Output :-

1. Top-3 youngest authors

Code :-

from datetime import datetime

df12 = Author

def calculate\_age(birthdate):

current\_date = datetime.now()

age = (current\_date.year- birthdate.year- ((current\_date.month, current\_date.day) < (birthdate.month,birthdate.day)))

return age

df12["Age"] = (df12["Birthday"].apply(calculate\_age))\*-1

df12 = df12[["Full\_Name", "Age"]]

df12 = df12.sort\_values(by="Age" , ascending=True)[:3]

plt.figure(figsize=(30, 25))

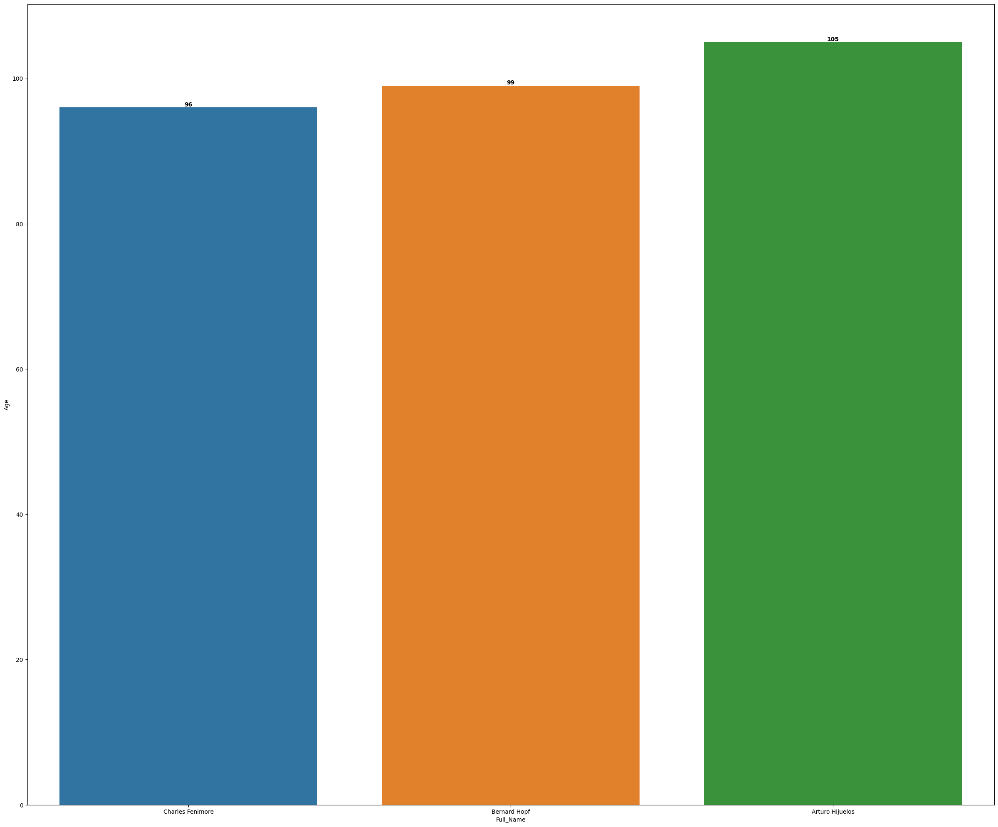
sns.barplot(data=df12, x="Full\_Name", y="Age", hue="Full\_Name")

for i, value in enumerate(df12["Age"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.show()

Output :-

****

1. Whose (Author) book is least read by the readers

Code :-

Sales = pd.concat([Sales\_Q1.value\_counts("ISBN").reset\_index(),Sales\_Q2.value\_counts("ISBN").reset\_index(),Sales\_Q3.value\_counts("ISBN").reset\_index(),Sales\_Q4.value\_counts('ISBN').reset\_index(),],ignore\_index=True,)

Sales = pd.merge(Sales , Edition , on="ISBN")

Sales['Total Price'] = Sales['count'] \* Sales['Price']

Sales = Sales[['ISBN' , 'Total Price']].sort\_values(by='Total Price' , ascending=False)

DS1A = pd.merge(Book, Edition, on="BookID")

DS1A = pd.merge(DS1A, Author, on="AuthID")

DS1A = pd.merge(DS1A, Sales , on="ISBN")

DS1A = DS1A.groupby('Full\_Name')['Total Price'].sum().reset\_index().sort\_values(by="Total Price", ascending=True)[:10]

df1a = DS1A

plt.figure(figsize=(30, 15))

sns.barplot(x=df1a["Full\_Name"], y=df1a["Total Price"], hue=df1a["Full\_Name"])

for i, value in enumerate(df1a["Total Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Author Name")

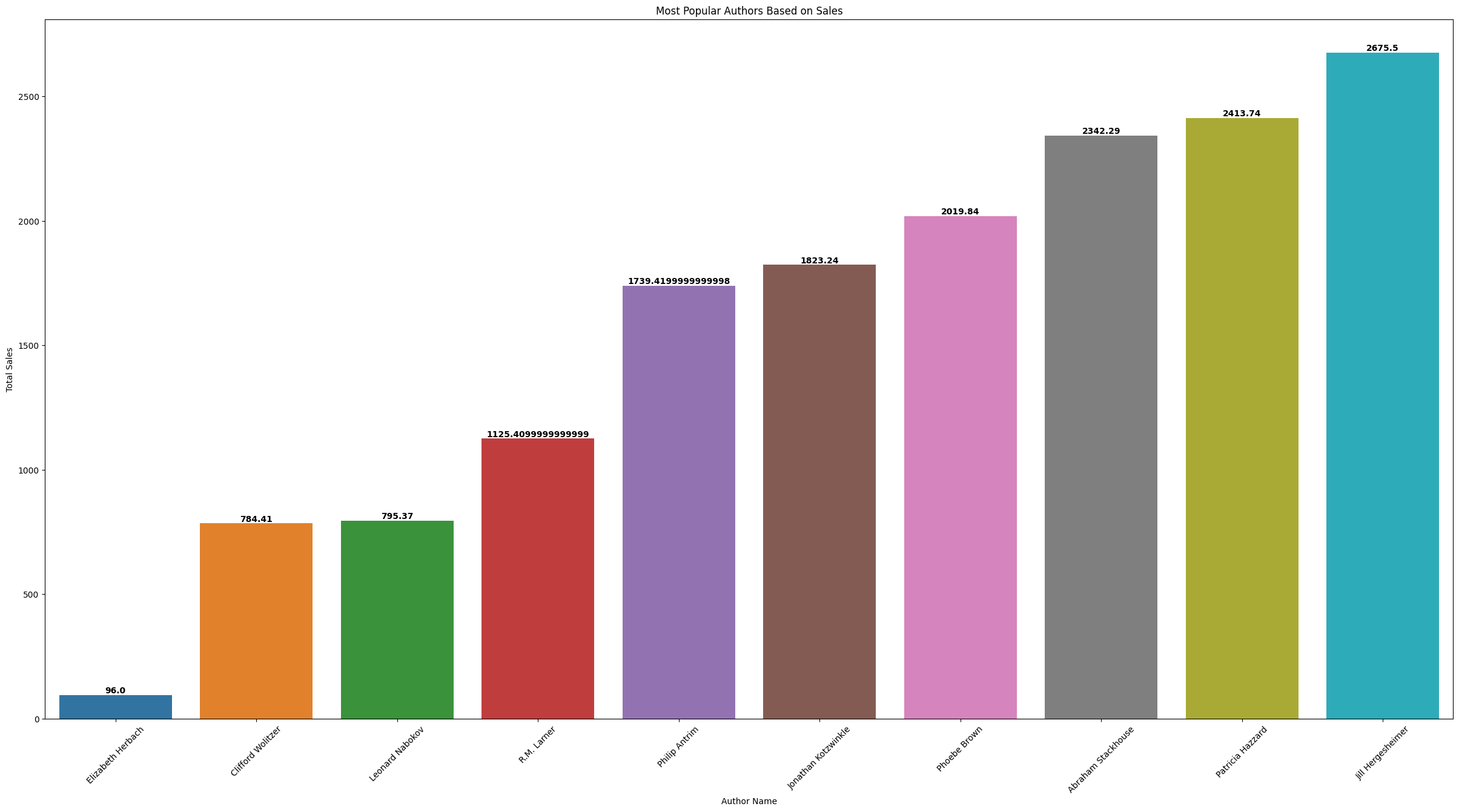
plt.ylabel("Total Sales")

plt.title("Most Popular Authors Based on Sales")

plt.xticks(rotation=45)

plt.show()

Output :-



1. Average price of the books published by top-5 authors (in terms of their published books)

Code :-

DS14 = pd.merge(Author, Book, on="AuthID")

DS14 = pd.merge(DS14, Ratings, on="BookID")

DS14 = pd.merge(DS14, Edition, on="BookID")

DS14["Full\_Name"] = DS14["First Name"] + " " + DS14["Last Name"]

Top\_Authors = list(

(

DS14.groupby("Full\_Name")["Rating"]

.mean()

.reset\_index()

.sort\_values(by="Rating", ascending=False)

)["Full\_Name"][:5]

)

DS14 = DS14[DS14["Full\_Name"].isin(Top\_Authors)]

df14 = pd.DataFrame(DS14.groupby("Full\_Name")['Price'].mean()).reset\_index().sort\_values(by = 'Price' , ascending=False)

plt.figure(figsize=(30, 15))

sns.barplot(x=df14["Full\_Name"], y=df14["Price"], hue=df14["Full\_Name"])

for i, value in enumerate(df14["Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

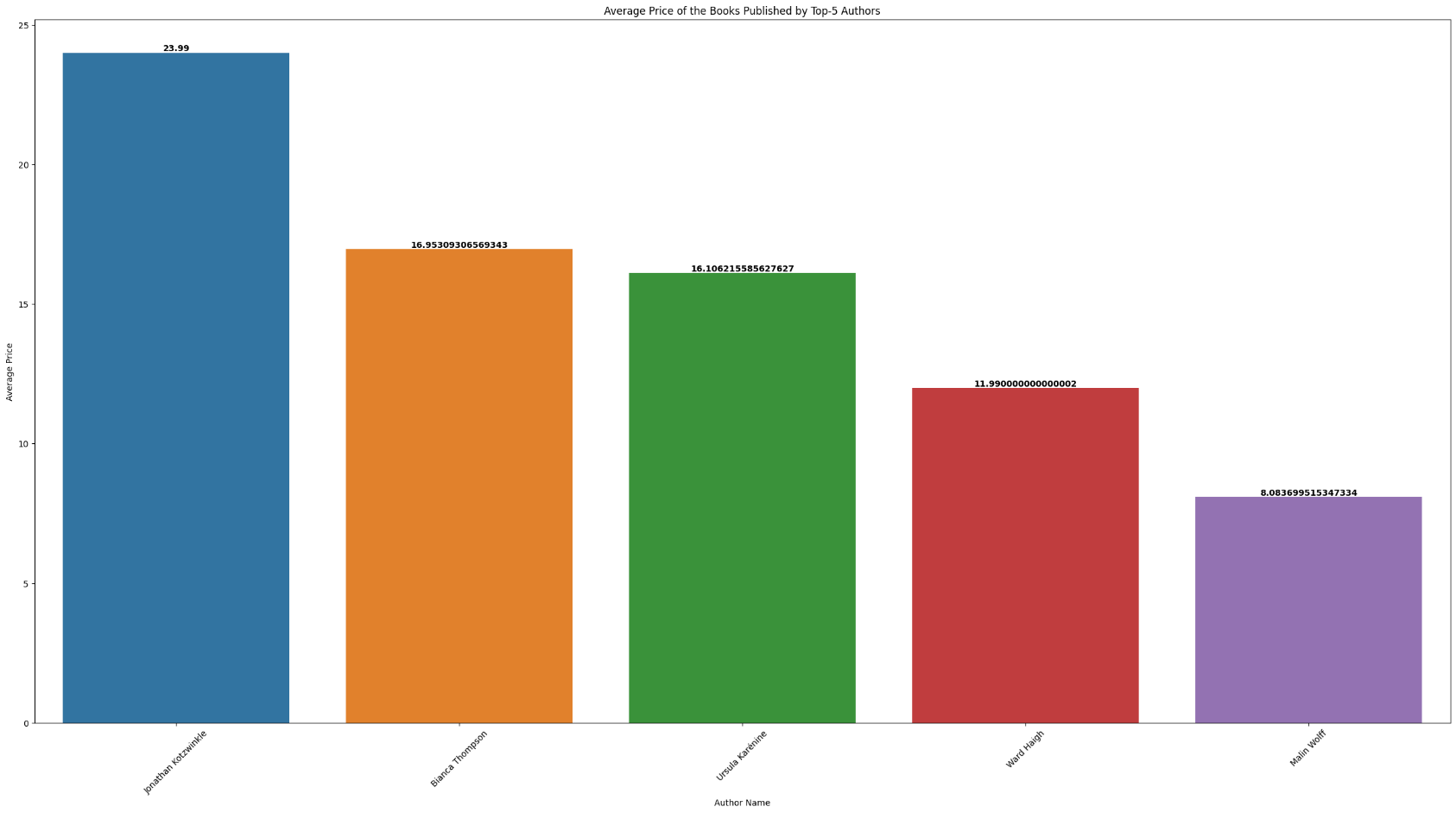
plt.xlabel("Author Name")

plt.ylabel("Average Price")

plt.title("Average Price of the Books Published by Top-5 Authors")

plt.xticks(rotation=45)

plt.show()

Output :-

1. Average price of the books for each publication house

Code :-

DS15 = pd.merge(Book ,Edition , on = "BookID")

DS15 = pd.merge(DS15, Publisher, on="PubID")

df15 = (pd.DataFrame(DS15.groupby("Publishing House")["Price"].mean()).reset\_index().sort\_values(by="Price", ascending=False))

plt.figure(figsize=(30, 15))

sns.barplot(x=df14["Full\_Name"], y=df14["Price"], hue=df14["Full\_Name"])

for i, value in enumerate(df14["Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Publishing House")

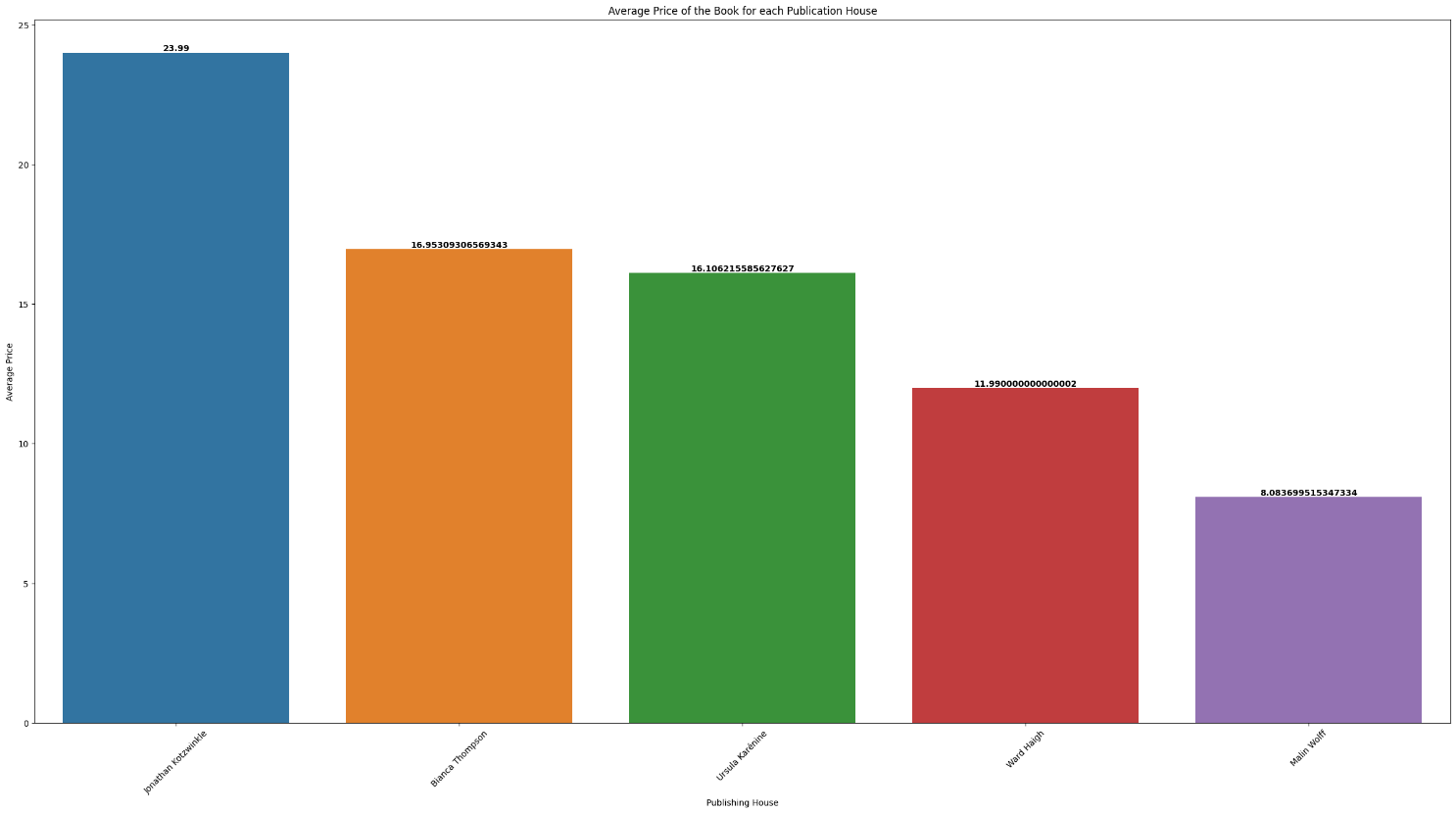
plt.ylabel("Average Price")

plt.title("Average Price of the Book for each Publication House")

plt.xticks(rotation=45)

plt.show()

Output :-



1. Which genre of the book has the highest sales

Code :-

DS16 = pd.merge(Info, Edition, on="BookID")

df16 = (pd.DataFrame(DS16.groupby("Genre")["Price"].sum()).reset\_index().sort\_values(by="Price" , ascending= False))

plt.figure(figsize=(30, 15))

sns.barplot(x=df16["Genre"], y=df16["Price"], hue=df16["Genre"])

for i, value in enumerate(df16["Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

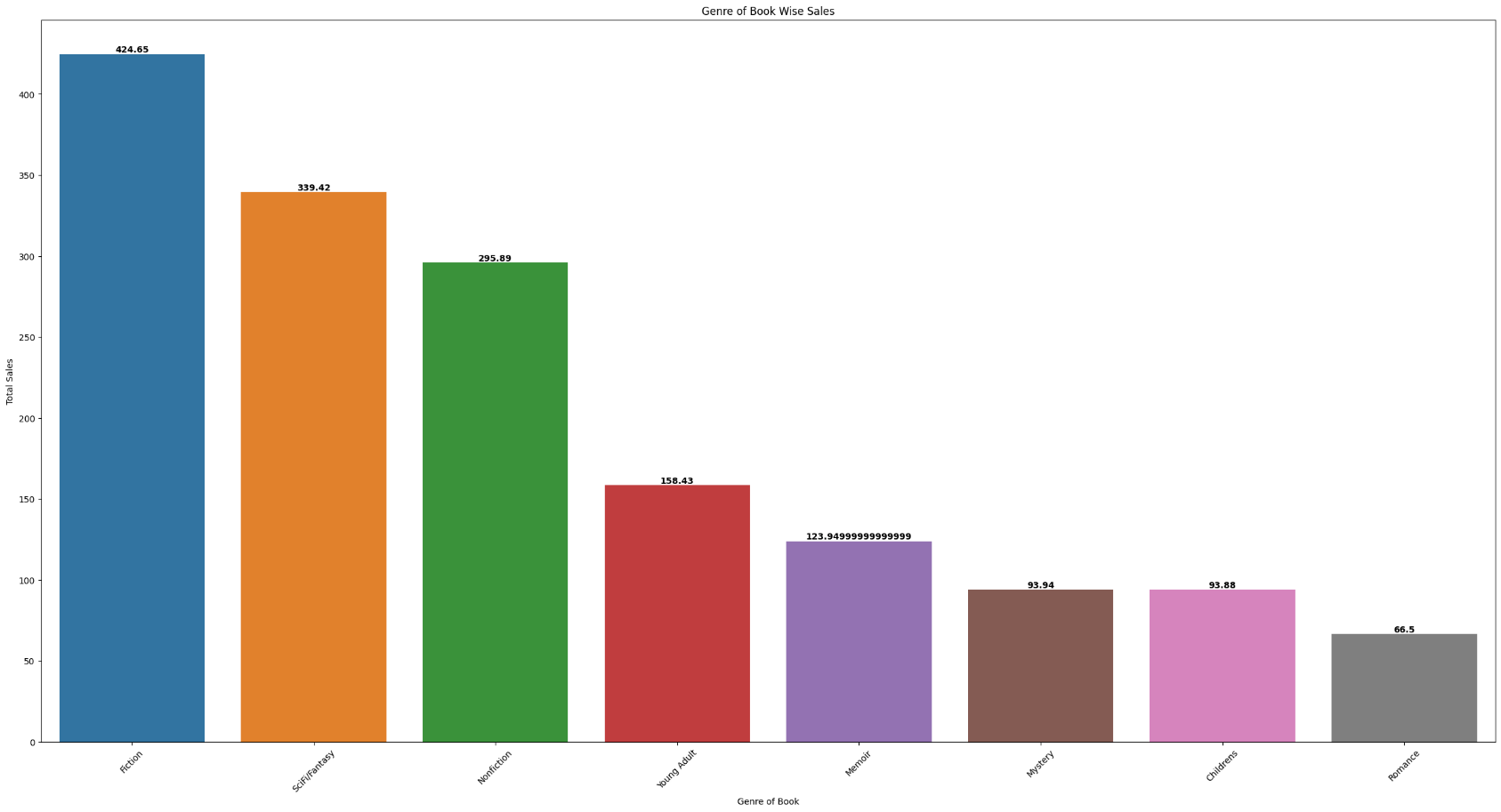
plt.xlabel("Genre of Book")

plt.ylabel("Total Sales")

plt.title("Genre of Book Wise Sales")

plt.xticks(rotation=45)

plt.show()

Output :-

1. Top 5 books having won the maximum number of awards

**Code :-**

DS17 = Award["Title"].value\_counts().reset\_index()

plt.figure(figsize=(30, 15))

sns.barplot(x=DS17["Title"], y=DS17["count"], hue=DS17["Title"])

for i, value in enumerate(DS17["count"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

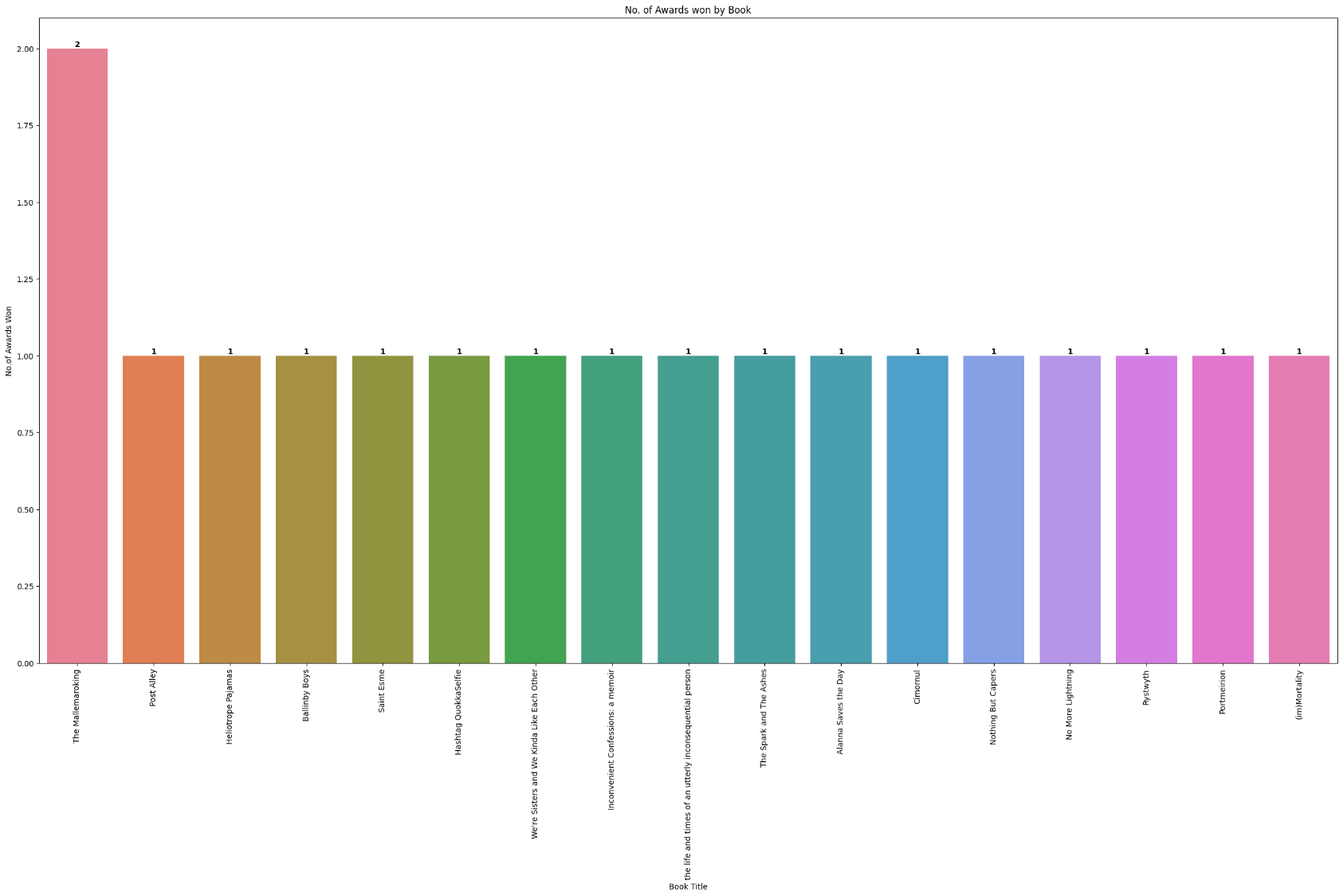
plt.xlabel("Book Title")

plt.ylabel("No.of Awards Won")

plt.title("No. of Awards won by Book")

plt.xticks(rotation=90)

plt.show()

**Output :-**

1. Top-3 publication houses having won the maximum number of awards

Code :-

DS18 = pd.merge(Book , Award , on="Title")

DS18 = pd.merge(DS18 , Edition , on="BookID")

DS18 = pd.merge(DS18, Publisher, on="PubID")

df18 = (DS18["Publishing House"].value\_counts()).reset\_index().sort\_values(by = "count" ,ascending= False)[:3]

plt.figure(figsize=(30, 15))

sns.barplot(x=df18["Publishing House"], y=df18["count"], hue=df18["Publishing House"])

for i, value in enumerate(df18["count"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

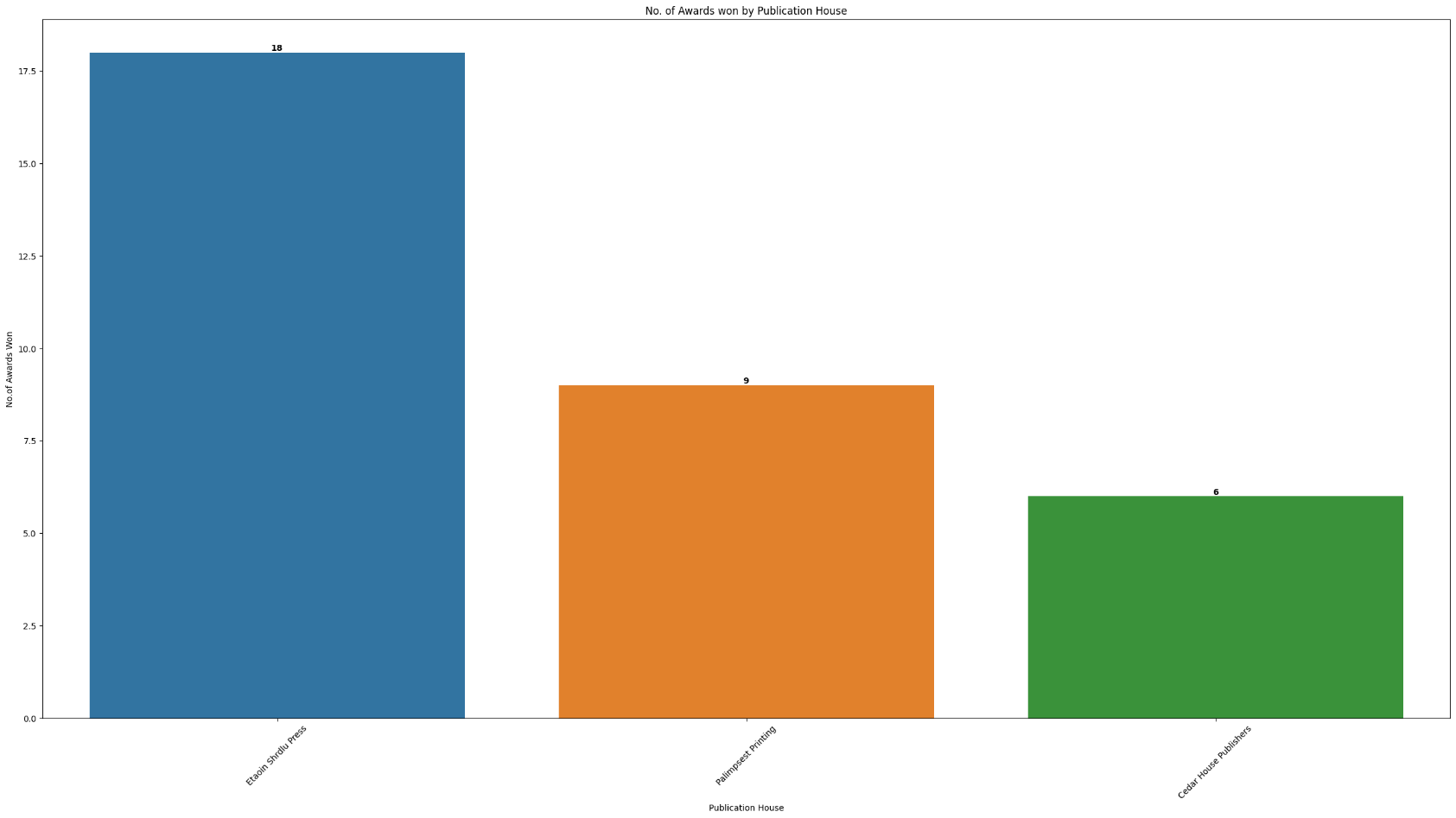
plt.xlabel("Publication House")

plt.ylabel("No.of Awards Won")

plt.title("No. of Awards won by Publication House")

plt.xticks(rotation=45)

plt.show()

Output :-

1. Which genre of the books has the highest price

Code :-

DS19 = pd.merge(Info, Edition, on="BookID")

df19 = pd.DataFrame(DS19.groupby("Genre")["Price"].max()).sort\_values(by="Price" , ascending= False).reset\_index()

plt.figure(figsize=(30, 15))

sns.barplot(x=df19["Genre"], y=df19["Price"], hue=df19["Genre"])

for i, value in enumerate(df19["Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

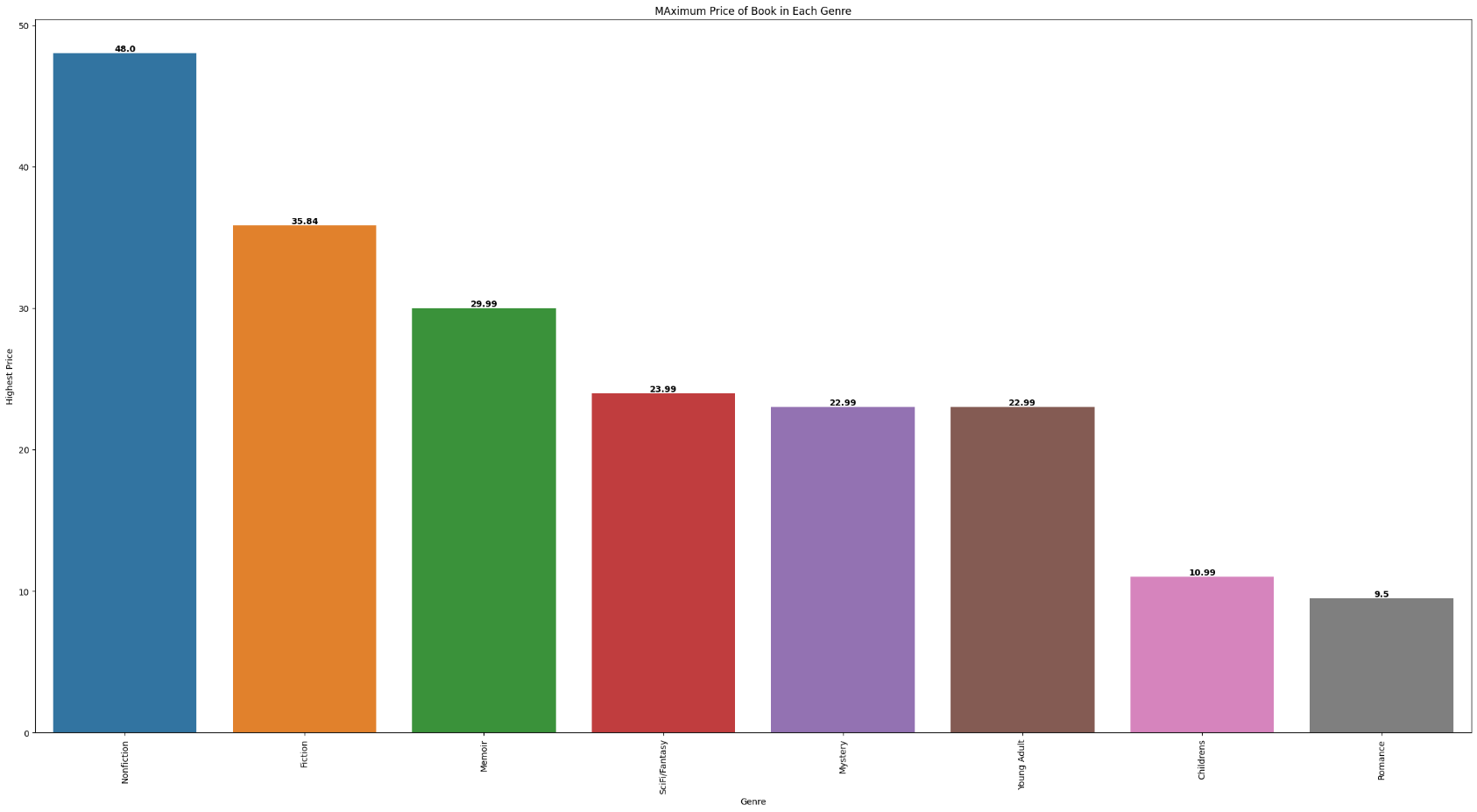
plt.xlabel("Genre")

plt.ylabel("Highest Price")

plt.title("MAximum Price of Book in Each Genre")

plt.xticks(rotation=90)

plt.show()

Output :-

1. Total number of books published by any publication house having a price >20 USD

Code :-

DS20 = pd.merge(Edition, Publisher, on="PubID")

DS20 = DS20[DS20['Price'] > 20.00]

df20 = DS20["Publishing House"].value\_counts().reset\_index()

plt.figure(figsize=(30, 15))

sns.barplot(x=df20["Publishing House"], y=df20["count"], hue=df20["Publishing House"])

for i, value in enumerate(df20["count"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Publishing House")

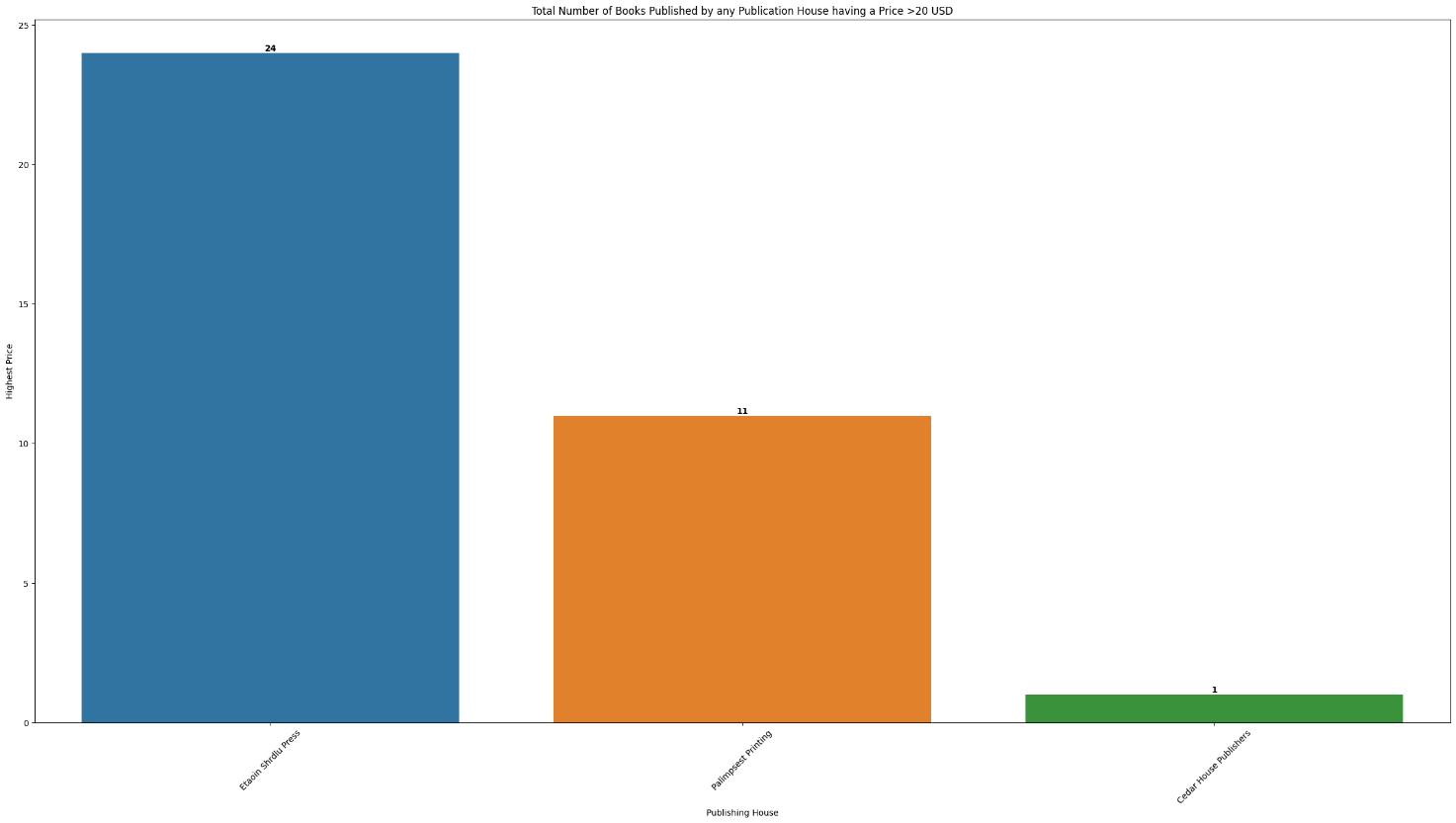
plt.ylabel("Highest Price")

plt.title("Total Number of Books Published by any Publication House having a Price >20 USD")

plt.xticks(rotation=45)

plt.show()

Output :-



1. Top-5 books with the highest worth (worth=price/number of pages)

Code :-

DS21 = pd.merge(Book, Edition, on="BookID")

DS21["Worth"] = DS21["Price"] / DS21["Pages"]

DS21 = DS21.drop\_duplicates(subset=["Title"])

DS21 = DS21.sort\_values(by="Worth", ascending=False)

df21 = DS21.head(5)

plt.figure(figsize=(30, 15))

sns.barplot(x=df21["Title"], y=df21["Worth"], hue=df21["Title"])

for index, value in enumerate(df21["Worth"]):

plt.text(index, value, str(round(value, 2)), ha="center", va="bottom", fontsize=10)

plt.xlabel("Book Title")

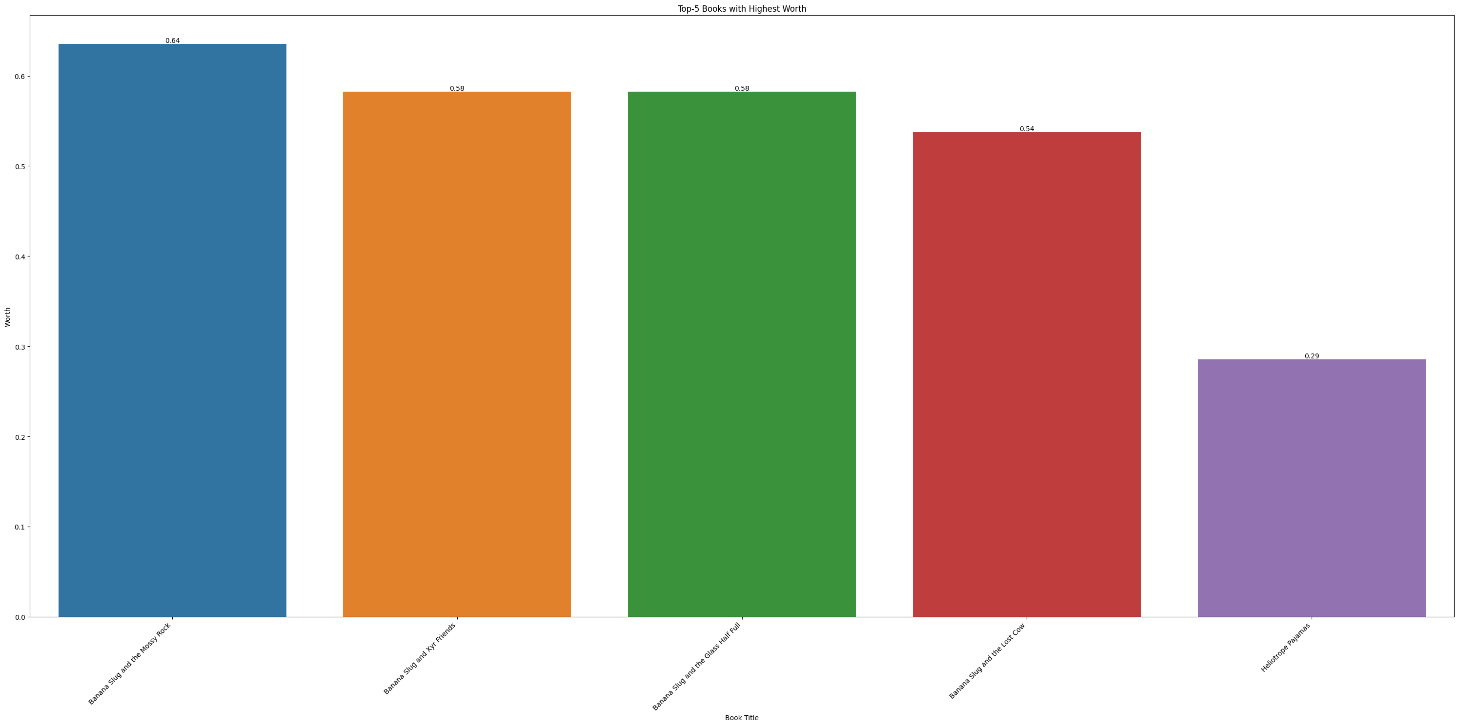
plt.ylabel("Worth")

plt.title("Top-5 Books with Highest Worth")

plt.xticks(rotation=45, ha="right")

plt.tight\_layout()

plt.show()

Output :-

1. Top-5 authors having the highest sales of their books

Code :-

DS22 = pd.concat([Sales\_Q1.value\_counts("ISBN").reset\_index(),Sales\_Q2.value\_counts("ISBN").reset\_index(),Sales\_Q3.value\_counts("ISBN").reset\_index(),Sales\_Q4.value\_counts("ISBN").reset\_index(),],ignore\_index=True,)

DS22 = pd.merge(DS22, Edition, on="ISBN")

DS22["Total Price"] = DS22["count"] \* DS22["Price"]

DS22 = pd.merge(DS22, Book, on="BookID")

DS22 = pd.merge(DS22, Author, on="AuthID")

DS22 = DS22[["Full\_Name", "Total Price"]]

df22 = (

pd.DataFrame(DS22.groupby("Full\_Name")["Total Price"].sum())

.reset\_index()

.sort\_values(by="Total Price" , ascending=False)

)

df22 = df22.head(5)

plt.figure(figsize=(30, 15))

sns.barplot(x=df22["Full\_Name"], y=df22["Total Price"], hue=df22["Full\_Name"])

for i, value in enumerate(df22["Total Price"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

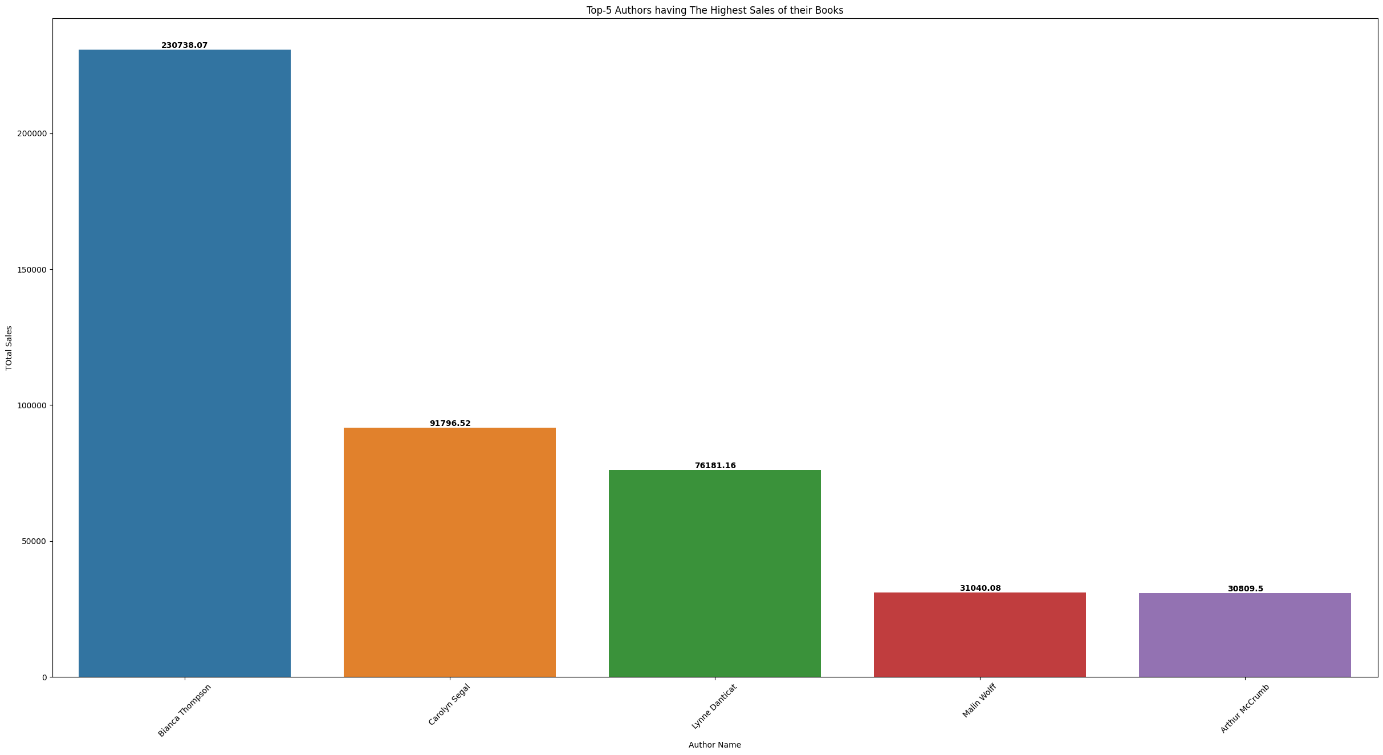
plt.xlabel("Author Name")

plt.ylabel("TOtal Sales")

plt.title("Top-5 Authors having The Highest Sales of their Books")

plt.xticks(rotation=45)

plt.show()

Output :-

1. Top-5 books with the highest number of pages

Code :-

DS23 = pd.merge(Edition , Book , on = "BookID")

DS23 = pd.merge(DS23, Author, on="AuthID")

DS23 = DS23.drop\_duplicates(subset=["Title"])

DS23 = DS23.sort\_values(by="Pages" , ascending= False)

df23 = DS23[["Title", "Pages"]].head(5)

plt.figure(figsize=(30, 15))

sns.barplot(x=df23["Title"], y=df23["Pages"], hue=df23["Title"])

for i, value in enumerate(df23["Pages"]):

plt.text(i, value, str(value), ha="center", va="bottom", weight="bold")

plt.xlabel("Book Title")

plt.ylabel("No.of Pages")

plt.title("Top-5 Books with the Highest Number of Pages")

plt.xticks(rotation=45)

plt.show()

Output :-

