1. Write a Pandas program to create a dataframe from a dictionary and display it.

Sample data:score={'Math': [78,85,96,80,86], 'English': [84,94,89,83,86], 'Hindi':

[86,97,96,72,83]}

# Importing the pandas library

import pandas as pd

# Creating a dictionary with sample data

score = {

    'Math': [78, 85, 96, 80, 86],

    'English': [84, 94, 89, 83, 86],

    'Hindi': [86, 97, 96, 72, 83]

}

# Converting the dictionary to a DataFrame

df = pd.DataFrame(score)

# Displaying the DataFrame

print(df)

Output:

Math English Hindi

0 78 84 86

1 85 94 97

2 96 89 96

3 80 83 72

4 86 86 83

1. Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels.

Sample Python dictionary data and list labels:

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James',

'Emily','Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8,19],

‘attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

# Importing the pandas library

import pandas as pd

# Importing numpy library to use np.nan for missing values

import numpy as np

# Creating a dictionary with the specified data

exam\_data = {

    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew',

'Laura', 'Kevin', 'Jonas'],

    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']

}

# Specifying the index labels

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

# Converting the dictionary to a DataFrame with the specified index labels

df = pd.DataFrame(exam\_data, index=labels)

# Displaying the DataFrame

print(df)

Output:

Name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

d James NaN 3 no

e Emily 9.0 2 no

f Michael 20.0 3 yes

g Matthew 14.5 1 yes

h Laura NaN 1 no

i Kevin 8.0 2 no

j Jonas 19.0 1 yes

ChatGPT Exercise

Using ChatGPT generate the python code to solve the same problem

Scenario: Analyzing Sales Data

Suppose you work for a retail company, and you have a dummy dataset containing sales data for the past year. The data includes information such as customer names, product names, sales quantities, prices, and dates. You want to perform various data analysis tasks like Total revenue for the year, Average revenue per sale, Best-selling product, Date with the highest total revenue also wants to generate product and total sales wise barchart using Pandas DataFrames.

Further, you need to get some inference out of the chart.

Create a ChatGPT prompt to generate the code for this scenario. Based on the code generated, ask ChatGPT to give the conclusion/inference.

Note. You can provide the data to ChatGPT in the form of a list or dictionary or ask it to use sample data.

import pandas as pd

import matplotlib.pyplot as plt

# Sample sales data

sales\_data = {

    'customer\_name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva', 'Frank', 'Grace', 'Hannah', 'Ivy', 'Jack'],

    'product\_name': ['Widget', 'Gadget', 'Widget', 'Gizmo', 'Widget', 'Gadget', 'Gizmo', 'Widget', 'Gadget', 'Widget'],

    'quantity': [5, 3, 2, 4, 1, 2, 1, 3, 2, 5],

    'price': [20, 35, 20, 40, 20, 35, 40, 20, 35, 20],

    'date': ['2023-01-15', '2023-02-20', '2023-03-10', '2023-04-05', '2023-05-17', '2023-06-22', '2023-07-11', '2023-08-13', '2023-09-29', '2023-10-25']

}

# Create DataFrame

df = pd.DataFrame(sales\_data)

# Convert date column to datetime

df['date'] = pd.to\_datetime(df['date'])

# Calculate total revenue

df['revenue'] = df['quantity'] \* df['price']

total\_revenue = df['revenue'].sum()

# Calculate average revenue per sale

average\_revenue\_per\_sale = df['revenue'].mean()

# Determine best-selling product

best\_selling\_product = df.groupby('product\_name')['quantity'].sum().idxmax()

# Identify the date with the highest total revenue

date\_with\_highest\_revenue = df.groupby('date')['revenue'].sum().idxmax()

# Generate product-wise total sales bar chart

product\_sales = df.groupby('product\_name')['quantity'].sum()

product\_sales.plot(kind='bar', title='Product-wise Total Sales')

plt.xlabel('Product')

plt.ylabel('Total Quantity Sold')

plt.show()

# Generate date-wise total sales bar chart

date\_sales = df.groupby('date')['revenue'].sum()

date\_sales.plot(kind='bar', title='Date-wise Total Sales')

plt.xlabel('Date')

plt.ylabel('Total Revenue')

plt.show()

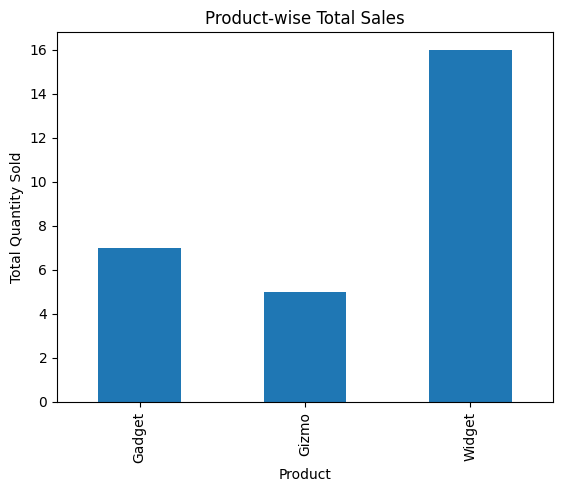
# Print results

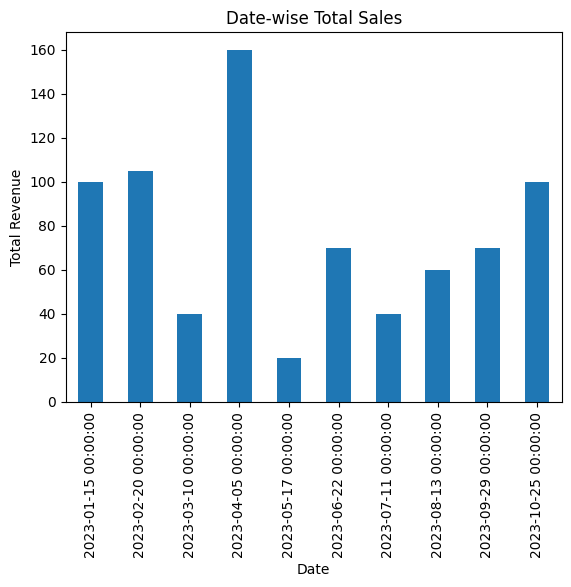
print("Total Revenue for the Year:", total\_revenue)

print("Average Revenue per Sale:", average\_revenue\_per\_sale)

print("Best-selling Product:", best\_selling\_product)

print("Date with the Highest Total Revenue:", date\_with\_highest\_revenue)





Total Revenue for the Year: 765

Average Revenue per Sale: 76.5

Best-selling Product: Widget

Date with the Highest Total Revenue: 2023-04-05 00:00:00