**Interview Questions and Answers:**

1. **How did you connect Python to a database?**

Ans. I have used

1️ import the SQLite library Python comes with a built-in sqlite3 module for working with SQLite databases.

import sqlite3

2️ Connect to the Database I have use sqlite3.connect() to either connect to an existing .db file or create a new one if it doesn’t exist.

conn = sqlite3.connect('sales\_data.db') # 'sales\_data.db' is the database file If you pass ':memory:' instead, it creates a temporary database in RAM.

3️ Create a Cursor A cursor is like a control handle that lets you execute SQL commands.

cur = conn.cursor()

4️ Execute SQL Commands You can use .execute() to run SQL commands — for creating tables, inserting data, querying, etc.

cur.execute("SELECT \* FROM sales") To fetch results:

results = cur.fetchall() # get all rows

5️ Close the Connection After I have done, always close the connection to free resources.

conn.close()

1. **What SQL query did you run**?

Ans. 1.Select \* from sales;

2.SELECT Product, sum(Quantity)as Total\_qty, sum(Quantity \* Revenue) AS Total\_Revenue

FROM sales

group by Product

These are the sql query I did.

1. **What does GROUP BY do?**

Ans. The GROUP BY clause groups rows that have the same values in specified columns into summary rows. It’s often used with aggregate functions like:

SUM()

AVG()

COUNT()

MAX()

MIN()

1. **How did you calculate revenue?**

Ans. Revenue = Price\_Per\_Box × Quantity

1. **How did you visualize the result?**

Ans. I have used matplotlib, a super popular Python library for creating charts and plots. Specifically, we made bar charts to compare total revenue by category.

import matplotlib.pyplot as plt

# Adding data labels to the bar chart

for i, value in enumerate(result\_df['Total\_Revenue']):

    plt.text(i, value, f'{value:.2f}', ha='center', va='bottom')

# Plotting the bar chart

plt.bar(result\_df['Product'], result\_df['Total\_Revenue'], color='skyblue')

plt.xlabel('Product')

plt.ylabel('Total Revenue')

plt.title('Product vs Total Revenue')

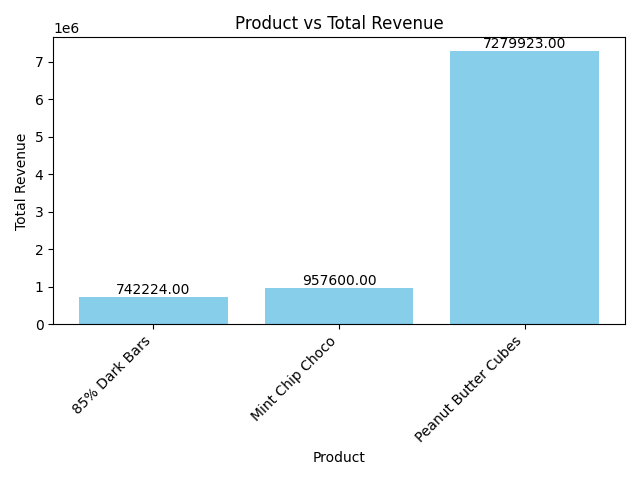
plt.xticks(rotation=45, ha='right')

plt.tight\_layout()

# Show the plot

plt.savefig('product\_vs\_total\_revenue.png')

plt.show()



1. **What does pandas do in your code?**

Ans. When we connected to the SQLite database, we used pandas.read\_sql\_query() to pull query results into a DataFrame — which is basically a Python version of a spreadsheet or SQL result table.

import pandas as pd

df = pd.read\_sql\_query("SELECT \* FROM sales", conn)

What it did:

Ran the SQL query

Loaded the result neatly into a DataFrame

Made it easy to view, analyze, and visualize the data.

1. **What’s the benefit of using SQL inside Python?**

Ans. SQL inside Python lets us:

* Run powerful, precise queries
* Analyze and visualize flexibly
* Automate and scale reporting
* Combine multiple data sources effortlessly

Example Use Case: “Get total revenue by country, visualize as a bar chart”

Use SQL inside Python to group and sum the data

Load it into a pandas DataFrame

Plot it with matplotlib

Quick, clean, and visual — no manual effort needed.

1. **Could you run the same SQL query directly in DB Browser for SQLite?**

Ans. Yes — absolutely! we can run the same SQL queries we used in Python directly in DB Browser for SQLite too.

It’s a free, visual, and user-friendly tool for working with SQLite databases. we can:

Open .db files (like your sales\_data.db)

Browse tables

Run SQL queries

View and export results