Problem Statement 1 :

1. API Data Retrieval and Storage: You are tasked with fetching data from an external REST API, storing it in a local SQLite database, and displaying the retrieved data. The API provides a list of books in JSON format with attributes like title, author, and publication year.

Assumption:

API returns data in json format ,each book record contains attributes like titles,Author,and publication year.

My approach would be :

- Send an HTTP GET request using beautifull soup to the external REST API to retrieve book data , we get a json format data through the GET request then turn it into a usable data structure.
- Establish connection to local sqlite database connection using magic% library from python.
- Create a Books table with column ,insert book records into the database using insert parameterized SQL quries,retrieve the stored records from the database and display them

2.Data Processing and Visualization: Given a dataset containing information about students' test scores, fetch the data from an API, calculate the average score, and create a bar chart to visualize the data.

Assumption:

API returns data from student table along with their scores to take the average ,some rows contain missing or invalid values have to take care of that thing when we get the data .

My approach would be :

- Fetch student data through API request using beautiful soup python package ,when we get the data check for invalid and missing values and take care of it if there is missing and invalid values either by filling by taking student average score or removing the missing values .
- Calculate the average score using SQL avg query by summing all valid scores.
- Use the bat chart to visualise the student score we can use python seaborn or matplotlib package to visualise it , give an appropriate title name and axes for better understandability and interpret the student score by comparing the student score.

3. CSV Data Import to a Database: Write a Python script that reads data from a CSV file containing user information (e.g., name, email) and inserts it into a SQLite database.

Assumption:

The csv contains many attributes such as name ,email and address like field.

My approach would be :

- Read the CSV file using python pd.read_csv file handling library the code would be:

Import pandas as pd

Data = pd.read_csv('dataset.csv')

- Check and perform basic validation
- Create a user table in SQLite using create table query

Create table user(id int primary key,

name varchar(20).

Email varchar(20));

- Insert each data from the csv file into the database using SQL insert query

Insert into user values(1,'abcd','abcd@gmail.com')

- Commit the changes to save it permanently.it ensures database consistency.

4.Send a link to the most complex Python code you have written

https://github.com/hannafarsin/FullStack-Automated-Review-Rating-System/blob/main/notebooks/fullstack_review_rating_with_logistic_regression.ipynb

5.Send a link to the most complex database code you have written

https://github.com/hannafarsin/IBM-Data-Science-Proffessional-Certificate-/tree/main/Databases%20and%20SQL%20for%20Data%20Science%20with%20Python

Problem statement- assignment 2

1. Where would you rate yourself on (LLM, Deep Learning, AI, ML). A, B, C [A = can code independently; B = can code under supervision; C = have little or no understanding]

ML – A=can code independently

DL – B=can code under supervision

AI- B = can code under supervision

LLM – B = can code under supervision ,limited experience training LLMs from scratch.

2. What are the key architectural components to create a chatbot based on LLM? Please explain the approach on a high-level.

An LLM based chatbot has multiple component to work with ie, understand user query,retrieve the rrelevent content based on the search and generate relevant output.

User interface(accept user input,generate user answer or chatbot response)→input processing and prompt management (user input is processed and formatted into a prompt that LLM can understand)→Embedding generation(to enable semantic understanding ,these embedding captures the semantic meaning of the text and these support similarity based retrieval)→vector database(store vector representation of knowledge,retrieve semantically similar content,and support RAG) →Retriever(select relevant context)→prompt construction with context(provide external knowledge to the LLM ,it will improve accuracy)→LLM(to generate natural language response).

User query → Embedding generation → Vector search → Context retrieval → Prompt construction → LLM response.

3. Please explain vector databases. If you were to select a vector database for a hypothetical problem (you may define the problem) which one will you choose, and why?

Vector databases are any database out there to store and retrieve info but in here the data it store vector embedding generated by machine learning model.it enables semantic similarity search allowing system to search for meaning wise rather than the exact text.

LLM are pretrained with millions of user text or documents and when a user input a query the system will semantically search for the most similar vectors and gives the result back to the user.

For eg a chatbot for a website for user help or support like amazon Rufus(an ai shopping assistant that answer product question etc), customer related queries can use a vector database to retrieve semantically relevant query instead of relying on keyword search .

To choose a vector database I would go with FAISS vector database because of its fast similarity search ,suitable for local or to test or to research based application.