Section 5: Bonus (Optional)

Write ~150 words on:

(1) How you'd support students struggling with concepts or deadlines.

Ans: Assisting students who find certain concepts or deadlines challenging requires understanding, flexibility, and a methodical approach. To start, I would foster a welcoming environment where students can freely share their confusion or concerns. For those struggling with understanding, I would decompose topics into smaller, manageable sections and enhance comprehension through examples, analogies, and engaging discussions. I would also suggest practical exercises to connect theoretical knowledge with real-world application. In cases where deadlines are a challenge, I would support them in organizing their tasks, dividing assignments into smaller goals, and introducing time management techniques like the Pomodoro method. I believe in tailored learning, so I would evaluate whether a student absorbs information better through visuals, written content, or hands-on experience and modify my assistance accordingly. Frequent check-ins and positive reinforcement can also greatly impact their progress. Ultimately, I would serve as both a mentor and a source of encouragement, ensuring that every student feels included and that learning remains an enjoyable and attainable objective.

(2) How you would break down a complex topic like "Gradient Descent" for beginners.

Ans: To help beginners understand a complex topic like Gradient Descent, I would start by using relatable analogies to build intuition. I often liken it to navigating down a hill while blindfolded: although you can't see the lowest point, you can sense the slope underfoot and take steps downward. Each step is guided by the steepness of the slope—this represents the gradient. After that, I would introduce the mathematical concept of minimizing a loss function and clarify how the algorithm makes movements proportional to the slope (the gradient) to attain the minimum. Next, I would explain the learning rate as the size of each step taken—if it's too small, progress is slow; if it's too large, you might overshoot. To reinforce understanding, I would provide visual demonstrations of gradient descent in action and gradually incorporate code snippets using Python or Jupyter Notebooks. By establishing the concept with intuition initially and then progressing to the mathematics and practical implementation, beginners can comprehend both the "how" and the "why."

Section 6: AI Tools & LLMs

Prompt Used:

"Write a query to get monthly sales trends in the year 2013 using the Chinook database."

Initial AI Response:

ChatGPT initially provided a query using the strftime() function:

```
SELECT
    strftime('%Y-%m', InvoiceDate) AS YearMonth,
    SUM(Total) AS MonthlySales
FROM
    Invoice
WHERE
    strftime('%Y', InvoiceDate) = '2013'
GROUP BY
    YearMonth
ORDER BY
    YearMonth;
```

However, this syntax is specific to SQLite. When I ran the query on MySQL, I received an error: ***"Error Code: 1305. FUNCTION chinook.strftime does not exist"***

I informed the model of the error and clarified that I was using MySQL.

ChatGPT’s Updated Response:

```
SELECT
    DATE_FORMAT(InvoiceDate, '%Y-%m') AS YearMonth,
    SUM(Total) AS MonthlySales
FROM
    Invoice
WHERE
    YEAR(InvoiceDate) = 2013
GROUP BY
    YearMonth
ORDER BY
```

```
    YearMonth;
```

Reflection:

What did the AI do well?

The initial response included a well-organized SQL query designed to analyze monthly sales trends. It also offered a clear explanation for each component of the query, which facilitated comprehension. Although the original version utilized functions specific to SQLite, the underlying logic of the query was sound. After I pointed out that I was working with MySQL, the AI promptly modified the syntax correctly without any misunderstandings.

What did you have to modify or improve?

I needed to specify the SQL engine I was working with since the original solution utilized SQLite's strftime() function. After I indicated this, ChatGPT substituted it with the DATE_FORMAT() and YEAR() functions appropriate for MySQL. This modification was crucial to prevent syntax errors and ensure the query was compatible with my database. Aside from that, there were no significant enhancements required as the revised query functioned flawlessly.