**Prompt to generate SQL query :**

You are a SQL expert. Use the table schema and metadata provided below to generate a SQL query. The data contains information about Insurance Claims, FNOL call, Adjustor notes and other details.

<table info>

Table: {table\_name}

Schema:

{schema}

Metadata:

{table\_metadata}

<table info/>

<Special Instructions>

- The SQL query must extract data semantically from text-based columns like 'adjuster\_notes' , 'fnol\_call'

without relying on exact pattern matching (e.g., LIKE or CONTAINS).

- Summarize or Explain means to read the entire Unstructured / Text Column requested

- claim no. , Claim Number , claim # - all mean the same primary key column

- Use LIMIT instead of TOP for restricting rows.

<Special Instructions/>

Convert the following natural language query into a SQL query:

{natural\_language\_query}

Only return the SQL query, no additional text.

"""

**LLM Response Prompt**

"""

    Generate a concise natural language response based on the standalone question,

    the SQL query result, and the chat history. Incorporate special instructions for uniform interpretation of key terms.

    """

    special\_instructions = """

Interpret the following terms consistently:

- "Summarize," "Describe," and "Explain" all mean "Provide a detailed breakdown with comprehensive coverage of the topic."

- Focus on clarity, detail, and coherence in the response.

- Avoid generic phrases or filler text (e.g., "Based on the information provided").

- Do not assume missing or unrelated information unless explicitly stated in the data.

"""

    if isinstance(dataframe, str) and "SQL error" in dataframe:

    # Handle SQL syntax errors

        error\_prompt = f"""

    The following request could not be processed due to a technical error in generating or executing the SQL query.

    Explain this to the user in simple terms without using technical jargon like 'SQL error' or 'query execution.'

    Focus on a helpful and user-friendly response.

    User Request: {standalone\_question}

    Chat History:

    {chat\_history}

    Error Details:

    {dataframe}

    """

        response = llm.invoke(error\_prompt)

        return response.content.strip()

    elif dataframe.empty:

    # Handle empty results

        no\_data\_prompt = f"""

    The following request was made, but the system could not find any relevant information.

    Provide a concise and direct response to the user, following these special instructions:

    {special\_instructions}

    User Request: {standalone\_question}

    Chat History:

    {chat\_history}

    """

        response = llm.invoke(no\_data\_prompt)

        return response.content.strip()

    else:

        # Convert the DataFrame to a comma-separated format including column names

        result\_text = ", ".join(dataframe.columns) + "\n"  # Add column names

        result\_text += "\n".join(dataframe.astype(str).apply(lambda row: ", ".join(row), axis=1).tolist())

        # Use the LLM to generate a response based on the query result and chat history

        response\_prompt = f"""

Provide a response based on the retrieved information. Follow these special instructions:

{special\_instructions}

User Request: {standalone\_question}

Information Retrieved:

{result\_text}

Chat History:

{chat\_history}

"""