Problem Statement:

Build a model that takes a natural language question and a table schema as input, and produces a structured SQL query as output. This generated SQL query, when executed on the corresponding table, should yield the correct answer to the question.

NOTE: We will be focusing on lightweight models (like DistilBERT or LLMs less than 2B) that can run for free on Google Colab. The goal isn't just to get the right answer, but to understand why it's so hard for a machine to connect natural language to structured table cells.

Key Objectives:

1. **Build a Model:** The Main task is to implement a model that takes a question and a table, and then generates a SQL query to find the answer (semantic parsing).
2. **Run it on Colab:** The entire project from start to finish must run in a standard Google Colab notebook.
3. **Analysis:** deeply analyze the linguistic challenges. Why does your model fail? When it succeeds, what linguistic cues did it likely pick up on?

Datasets:

You can choose one of the following:

- **WikiSQL:** A great starting point. The questions are simpler and map cleanly to SQL queries. It's perfect for a semantic parsing approach. [Link]
- **FeTaQA:** A much more challenging dataset! The questions are "free-form" and often require reasoning or pulling information from multiple cells. This is a good choice if you want to tackle a harder problem.

Literature Survey:

Expected to do a literature survey, but here are two key papers to get started.

1. **DIN-SQL**: Decomposing In-Context Learning for Text-to-SQL with Large Language Models (2023) - This paper details the semantic challenges of Text-to-SQL.
2. **PICARD**: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models
3. **FeTaQA:** Free-form Table Question Answering.

Analysis:

Analyze your model's failures and successes through an NLP lens. Consider these questions in your report:

- **Ambiguity:** How does your model handle ambiguous questions? For example, if a question asks for "the highest score" and the table has columns for "Exam 1 Score" and "Final Score," how does it know which one to pick?
- **Entity Extraction & Linking:** How well does the model link entities in the question (e.g., "New York") to the values in the table cells (e.g., "NYC", "New York City")? This is a classic entity linking problem.
- **Implicit Needs:** What about questions that require common-sense reasoning not explicitly stated? For instance, asking "who is the oldest person?" requires the model to know it needs to find the MAX of the "Age" column.
- **Compositionality:** How does the model handle complex questions that combine multiple conditions, like "Which players from the USA are over 25 years old?"

Novelty

- A clever way to encode the table schema and question together.
- A data augmentation technique to create more training examples.
- A unique method for handling numerical reasoning or date comparisons.
- Trying to combine the ideas from TAPAS and a Text-to-SQL approach.