CS 6422 Project Proposal: SQL Query Analysis in EvaDB

Krish Nathan

September 2023

1 Introduction and Objectives

Program analysis is an area of active development, with research spanning from formal methods to using large language models (LLMs) to explain snippets of code. Especially in dynamically typed languages such as Python, static analysis can prevent many classes of runtime errors which might not be caught by a programmer. Furthermore, LLMs are now capable of explaining how programs work. These tools can provide context to how sections of code work and make it much easier to debug and add new features.

The advantages of program analysis do not need to be restricted to general programming languages. Database queries are often complex and difficult to reason about. Perhaps LLMs could provide an assist by explaining what a SQL query does in plain English. This functionality can be baked into EvaDB, so that the programmer need not search elsewhere. Furthermore, EvaDB knows the tables and relationships, so it has greater context around potential queries. My project will focus on applying static analysis and LLMs to explain what a SQL query will do within EvaDB.

2 Goals / Supported Queries

I aim to support a function `EXPLAIN_SQL(query)` where query is a string containing the SQL query of interest. A straightforward way to start would be to leverage ChatGPT or another LLM to return a string explanation of the query. Additionally, EvaDB has context about the tables and relationships between them. I hope to inject this context into the prompt to produce more accurate explanation from the LLM.

Beyond explainability, database programmers care quite a bit about the performance of queries. However, the query plan might not always be transparent or easy to understand for them. As a stretch goal, I hope to support a `EXPLAIN_QUERYPLAN` function which explains the query plan in plain English. I'm aware that this depends on the particular database engine, so perhaps I could target only one engine to reduce the scope. A natural language explanation could help the programmer understand the performance characteristics of different queries concretely.

3 Anticipated Challenges

Determining the output of a program is known to reduce to the Halting problem, which is undecidable. It's impossible to determine how a program will behave given any arbitrary set of inputs. The same applies to SQL queries, so we will need to reduce the scope of this problem significantly. I aim to focus first on `SELECT` since it is often used and simpler than other queries. After this, `INSERT` and `GROUP` would be good challenges, and they are also quite popular. I will avoid nested queries for now, since they pose an additional layer of complexity.

4 Development Environment

I plan to create a fork of EvaDB where I will implement the two queries that I have proposed. Since EvaDB is written in Python, I will be developing my application in Python as well.

5 Progress so Far

I have looked through the EvaDB docs as well as the sentiment analysis use case and the example youtube question repository. My next step is to fork the EvaDB repository and investigate the EvaDB source code more.

6 Development Plans

As a minimimum viable product, I plan to implement the `EXPLAIN_SQL(query)` function within EvaDB and support at least `SELECT` queries, although ideally more including `INSERT` and `GROUP`.

As a version two, I plan to inject context about the database schema into the prompt to explain a query. The schema information would need to be gathered from EvaDB itself.

As a version three, I plan to implement the `EXPLAIN_QUERYPLAN` function targeting a single database engine. Producing the query plan may be slightly challenging since this will involve the internals of the database engine, especially the query planner and optimizer. However, some datbases such as MySQL provide an EXPLAIN statement which can produce information about the query plan or the runtime behavior of a given query. If this can be achieved, I would use an LLM to generate the plain English interpretation of the query plan.

References

[1] Query plan
https://en.wikipedia.org/wiki/Query_plan

[2] Query optimization
https://en.wikipedia.org/wiki/Query_optimization

[3] Static code analysis
https://en.wikipedia.org/wiki/Static_program_analysis

[4] MySQL explain
https://dev.mysql.com/doc/refman/8.0/en/explain.html

[5] Adrenaline, static code analysis for Github repos
https://github.com/shobrook/adrenaline