Brandenburg University of Technology

IT Security
Computerscience and Media
Prof. Dr. Oleg Lobachev
Florian Eich

# Natural Language Queries using Large Language Models

Bachelor Thesis

Summer semester 2025

April 17, 2025

Mara Schulke – Matr-Nr. 20215853

**Abstract**

This thesis explores the integration of large language models (LLMs) into PostgreSQL database systems in order to make the database accessible via natural language instead of the postgres SQL dialect. The research focuses on implementation strategies, performance optimization, and practical applications of this concept.

## Contents

## List of Figures

## List of Abbreviations

| | |
|---|---|
| GPT | Generative Pretrained Transformer |
| SQL | Structured Query Language |
| API | Application Programming Interface |
| LLM | Large Language Model |
| DBMS | Database Management System |
| NL2SQL | Natural Language to SQL |

# 1 Introduction

## 1.1 Problem Statement and Motivation

Database systems represent a backbone of modern computer science, allowing for rapid advancements whilst shielding us from the problem categories that come along with managing and querying large amounts of, usually structured, data efficiently. However, most Database Management Systems (DBMS) have traditionally required specialized knowledge, usually of the Structured Query Language (SQL), in order to become useable. Whilst this barrier may be percieved differently across diverse usergroups it represents a fundamental misalignment between end-user goals (e.g. analysts, researchers, domain experts etc.) and the underlying DBMS, thus often requiring software engineering efforts in order to reduce this friction.

This barrier is the reason entire classes of software projects exists (for example, admin / support panels), data analytics tools etc. which therefore introduce significant churn and delay between the implementation of a database system and reaching the desired end user impact. Often these projects span multiple years, require costly staffing and yield little to no novel technical value.

Emerging technologies such as Large Language Models (LLMs) have proven themselves as a sensible tool for bridging fuzzy user provided input into discrete, machine readable formats. Prominent models in this field have demostrated outstanding capabilities that enable computer scientists to tackle new problem classes, that used to be challenging / yielded unsatisfying results with discrete programming approaches.

This thesis is exploring ways to overcome the above outlined barrier using natural language queries, so that domain experts, business owners, support staff etc. are able to seamlessly interact with their data, essentially eliminating the requirement of learning SQL (and its pitfalls). By translating natural language to SQL using Large Language Models this translation becomes very robust (e.g. against different kinds of phrasing) and enables novel applications in how businesses, researchers and professionals interact with their data — it represents a fundamental shift (ie. moving away from SQL) towards a more inclusive and data driven world.

## 1.2 Objectives of the Thesis

This thesis aims to address the aforementioned challanges when it comes to database accessibility. The following objectives are the core research area of this thesis:

1. Develop a database extension that can translate natural language queries into semantically accurate SQL queries using Large Language Models.

2. To evaluate the effectiveness and feasibility of different Models aswell as prompt engineering techniques in order to improve the performance of the system.

3. Identify and address issues when it comes to handling amibguous, complex and domain specific user input.

4. Benchmark the performance of the implementation against common natural language to SQL (NL2SQL) benchmarks.

5. Idenitfy potential use cases for real world scenarios that could deliver a noticable upsides to users.

6. Analyze the short commings and limitations of this approach and propose potential solutions to overcome them.

### 1.3 Research Questions

**RQ1 — Are natural language queries performant and accurate enough to find application in the real world?**

The primary research questions when it comes to natural language database interfaces evolves around their semantic accuracy and reliability. LLMs have notoriously been known for their ability to hallucinate / produce false, but promising outputs. This behaviour can be especially dangerous when opting for data driven decisions that rely on false data due to a mistranslation from natural language to SQL. LLMs could cause hard to understand/debug behaviour, like false computation of distributions when the intermediate format is not being shown to the user. This thesis tries to determine whether such hallucinations could be reasonably prevented (ie. through pre and post-processing if LLM input/output).

Additionally, this research eveluates the associated performance and hardware requirements, as a practical deployment of such a natural language database interface outside of research situations is guarded by the ability to:

1. Run on reasonable, mass available hardware (e.g. excluding high end research GPUs).
2. Run within a suitable timeframe that is not causing a significant performance degredation of the overall system / delay to the end user.

**RQ2 — Are there effective approaches against ambiguity in natural language input?**

To provide semantically correct results ambiguity in the user-provided natural language queries must be adequately addressed. This thesis investigates various approaches to ambiguity management and resolution. Natural language queries can demonstrate ambiguity even at low levels of complexity — e.g. there are two different types of "sales" in a database schema, and the user asks to retireve "all sales".

Such situations present the second major challenge associated with the practical implementation of natural language database interfaces. The success of this concept will significantly depend on whether suitable designs and mitigation techniques can be implemented without creating problems with regards to the aforementioned performance and hardware requirements. The research focus lies on both preventative measures through optimized pre-processing stages and prompt engineering techniques as well as reactive strategies that post process LLM output, either on the basis of further user input or context inference.

**RQ3 — What are possible approaches for increasing semantic accuracy of queries?**

In order to enhance the semantic accuracy a series of improvements may be applied to the pipeline. Potential optimizations include supplying (parts of) the schema during LLM prompting, implementation of interactive contextual reasoning through a conversational interface which would allow for user refinement, the implementation of a robust SQL parsing and validation mechanism and a hybrid approach partly relying on tradition NLP preprocessing techniques. This research will quantify semantic accuracy using popular NL2SQL benchmarks and empirically evaluate the impact each approach has on the benchmark performance. Furthermore this research will take a look at the optimal combination of the aforementioned solutions in order to develop a system that strikes the right balance between accuracy and performance.

### 1.4 Methodological Approach

### 1.5 Structure of the Thesis

# 2 Theoretical Foundations

## 2.1 Generative Pretrained Transformers (GPT)

### 2.1.1 Architecture and Functionality

### 2.1.2 Training and Fine-tuning

### 2.1.3 Application Areas

## 2.2 PostgreSQL as a Database System

### 2.2.1 Architecture of PostgreSQL

### 2.2.2 Extension Capabilities

### 2.2.3 PostGIS and Other Extensions as Examples

## 2.3 Embedding AI Models in Database Systems

### 2.3.1 Current Approaches and Solutions

### 2.3.2 Technical Challenges

### 2.3.3 Benefits of Integration

# 3 Conceptual Design of GPT Embedding in PostgreSQL

## 3.1 Requirements Analysis

### 3.1.1 Functional Requirements

### 3.1.2 Non-functional Requirements

## 3.2 Architecture Design

### 3.2.1 Interface Design

### 3.2.2 Data Model

### 3.2.3 Integration into PostgreSQL

## 3.3 Technical Implementation Strategies

### 3.3.1 Foreign Data Wrapper

### 3.3.2 Extension Using C/C++

### 3.3.3 PL/Python or Other Procedural Languages

### 3.3.4 Comparison of Approaches

# 4 Implementation

## 4.1 Development Environment and Tools

## 4.2 Integration of the GPT Model

### 4.2.1 Model Selection and Optimization

### 4.2.2 API Connection or Local Embedding

## 4.3 Development of the PostgreSQL Extension

### 4.3.1 SQL Functions for GPT Interactions

### 4.3.2 Data Type Conversion and Processing

### 4.3.3 Error Handling and Logging

## 4.4 Optimization

### 4.4.1 Performance Tuning

### 4.4.2 Memory Usage

### 4.4.3 Parallelization

# 5 Evaluation

## 5.1 Test Environment and Methodology

## 5.2 Performance Tests

### 5.2.1 Latency

### 5.2.2 Throughput

### 5.2.3 Scalability

## 5.3 Use Cases

### 5.3.1 Natural Language Queries

### 5.3.2 Text Generation Within the Database

### 5.3.3 Semantic Search and Text Classification

## 5.4 Comparison with Alternative Approaches

# 6 Discussion

## 6.1 Interpretation of Results

## 6.2 Limitations of the Implementation

## 6.3 Ethical and Data Privacy Considerations

## 6.4 Potential Future Developments

# 7 Summary and Outlook

## 7.1 Summary of Results

## 7.2 Addressing the Research Questions

## 7.3 Outlook for Future Research and Development

## References

[1] Author, A. (Year). Title of the reference. Journal/Publisher, Volume(Issue), Pages.

## Appendix

**Installation Guide**

**API Documentation**

**Code Examples**

**Test Data and Results**