**Database Systems and Information Management Group**

**Fak. IV Electrical Engineering and Computer Science**

**Technische Universität Berlin**

**Thesis Proposal**

| **Thesis Type** | Bachelor’s | | **ECTS** | 12 |
| --- | --- | --- | --- | --- |
| **Student** | Sönke Fridtjof  Damm | | **Mat.**  **Number** | 403189 |
| **Email** | soenke.f.damm@campus.tu-berlin.de | | **Primary**  **Advisor** | Dr. Stefan Halfpap |
| **1st**  **Examiner** | Prof. Dr. Volker Markl | | **2nd**  **Examiner** | Prof. Dr. Matthias Böhm |
| **Academic Program** | | Industrial Engineering (Information and Communication Systems) | | |
| **Thesis Title** | | Robustness Benchmarking Using Parameterized Queries | | |
| **Duration Period** | | From 19/08/2024 to 08/11/2024 | | |

**1. Introduction / Scientific Background / Related Work**

* 1. **Specify the *obstacle* to be overcome**

In database systems the use of declarative query languages requires to analyze and select from a set of query execution plans. Performant execution plan choices of database systems are key properties to them, but in reality not easily achieved.

Robustness can be understood as the ability to choose performant query plans under changing conditions.

This thesis aims to find and implement a holistic approach to benchmark and visualize the robustness of a database management system.

* 1. **Specify the *motivation* (Why is this problem interesting?)**

Modern query processing is of high complexity and the differences in processing cost can vary significantly between an optimal plan or a random plan choice.

Extensive knowledge about a database's robustness therefore is a subject of interest in the industry as well as in the research field.

* 1. **Specify the *novelty* (Was this problem already solved?)**

Various approaches in the database field address robustness. Many propositions implement robustness improvement techniques through different query execution operators. Others do follow the goal to measure and visualize robustness. To our best knowledge they all worked with undisclosed or commercial database systems and their code is not open-source.

In contrast, we aim to build an own robustness benchmarking framework that is open-source and will be compatible to use with open-source database systems as PostgreSQL.

Further we aim to give propositions to help the future work towards finding a metric for evaluating robustness to enable practical comparison of systems.

* 1. **Specify the *anticipated impact* (how does solving this problem impact our world?)**

As mentioned before, unpredictable query optimizer robustness may lead to high cost and bad performance.

We assume therefore that an extensive overview of a given database engine's optimizer behavior brings benefit to the database community, as well as industry users.

Further this thesis aims to help to find robust database systems or to be starting point for future work on improving database robustness.

* 1. **Specify the *anticipated contributions* (*journal publication*? *open-source software*?)**

We anticipate upon the completion of this thesis and the system implementation to deliver an explicit Robustness evaluation of our PostgreSQL use case. Further, we anticipate publishing the robustness measurement tool as open-source software.

* 1. **Specify the *scientific background* (or foundational work) that your solution will build upon**

We will base scientific database background on chapters from textbooks e.g. [8] and database system specific background on further documentation[9], [10]. For the robustness part Haritsa [1] and Wiener et al.[5] give an overview of the ideas of database robustness research.

* 1. **Specify the *related work* (e.g., competing solutions corresponding to the research challenge)**

The robustness research so far brought up many approaches to improve query optimizer operations, as well as benchmarks to challenge optimizers with difficult workloads, for example the Join-Order Benchmark[6].

Other publications like the PICASSO Visualizer [2], [3] develop a query plan diagram tool that visualizes different query plans in a sophisticated way. Also Graefe et al.[4] found another way to visualize robustness and introduced the idea of a robustness metric.

Also, the work of several Dagstuhl Seminars on different strategies or sub-problems of the robustness as they address robustness improvement and robustness measurement strategies[7].

**2. Goal of the Thesis / Statement of the Research Problem**

**2.1 State the overarching goal that you aim to achieve in your thesis.**

Implementing a holistic approach to assess and visualize a database’s systems robustness properties.

**2.2 Provide a succinct, precise, and unambiguous *statement of the research problem(s)* or *question* to be solved, in order to attain the goal.**

How can benchmarking with parameterized queries be used to improve knowledge about a database system's robustness?

**2.3 Specification of the *scope of the thesis*.**

**2.3.1 State what is *in scope*, particularly, the *subproblems that will be explored.***

Within the scope of this thesis lies the performance measurement of parameterized query execution on the TPC-H benchmark dataset on a PostgreSQL database. Further, the visualization and the assessment of the benchmarking results and putting them into the context of the robustness characteristics of the database system.

**2.3.2 State what is *out of scope*.**

The benchmarking will follow a “black box” approach so that it is not within the scope to further search to reason the database's robustness for the found behavior, nor to work towards any modifying approaches to improve the database's robustness properties.

It is not the priority of this thesis to cover a bigger spectrum of database systems or to automatically work with other benchmark datasets.

**3. Thesis Approach**

**3.1 Describe the solution approach (e.g., algorithms, data, evaluation metrics, software, systems).**

As data we use the TPC-H decision benchmark on a PostgreSQL database as a choice of open-source database engine. A python script generates parameterized queries based on the existing TPC-H query templates and executes these on the database instance.

We measure several metrics and gather data of the query execution as follows: query execution times, cardinalities and query plans.

Further, we transform the query plans into a comparable format and visualize the optimizer’s execution plan choices in a plan diagram. The plan diagram associates the executed queries with a graphical representation of the respectively used query plan.

* 1. **How does the proposed solution differ from the state-of-the-art?**

Existing approaches focus on e.g. certain operators in the robustness assessment. We rather work on deriving a robustness metric for the system from our experiments. Also, commercial database systems are utilized mainly, whereas we focus on open-source engines and providing a open-source framework.

* 1. **How will you know if you have *succeeded* in attaining your goal, i.e., solved the research problem(s)?**

**3.3.1 How will you measure the *effectiveness* of your solution?**

We will use actual query execution plans containing different execution strategies to compare with the tool's output measure of query plan similarity to ensure the correctness and trustworthiness of our results. Other metrics as e.g. cardinalities and cardinality estimates will help to validate the correctness of the results.

**3.3.2 How and against which baseline(s) will you measure the *efficiency* of your solution?**

We will compare this thesis outcome to related work from e.g. PICASSO [2], [3].

Although it is not the main focus we will try to make the solution extendable to benchmark against more challenging workloads like the mentioned Join-order benchmark[6].

**4. Implementation Plan and Timeframe**

| **Dates** | **Mile stone** | **Tasks** | **Deliverable(s)** |
| --- | --- | --- | --- |
| 19.08 – 26.08 | **MS1** | 1. conduct literature review 2. read through identified papers 3. summarize key concepts, approaches and definitions | -list of relevant papers, tools  -reading notes |
| 26.08 – 09.09 | **MS2** | 1. study query plan *extraction* 2. explore visualization methods for use case 3. fully implement TPC-H query generation module and PostgreSQL benchmark integration | -functional benchmarking prototype on TPC-H dataset  - exemplary visualizations  -query parameterization module |
| 09.09 – 30.09 | **MS3** | 1. implement query runner/query plan extraction module 2. implement query plan comparison algorithm 3. validate prototype/system effectiveness | -functional query runner/extraction module  -comparison module |
| 30.09 – 21.10 | **MS4** | 1. run tool experiment on PostgreSQL db 2. collect and organize the results 3. build plan diagrams/heat map visualizations | -organized results  -visualizations |
| 21.10 – 04.11 | **MS5** | 1. analyze experimental results and interpret patterns 2. compare results with related approaches 3. analyze/interpret results (explain anomalies) | -results analysis/interpretation  -comparative analysis [2], [3]  - conclusion on robustness |
| 04.11 – 08.11 | **TC[[1]](#footnote-0)** | 1. complete thesis write-up 2. review and refine thesis 3. proofread & prepare appendices | -completed thesis document  -submit thesis by the due date[[2]](#footnote-1) |
| 08.11 – 22.11 | **SC[[3]](#footnote-2)** | 1. refine and finalize prototype/system 2. document code / README 3. prepare for open-source publication | -finalized tool prototype  -documentation  -publish code in an open repository |

**Preparing for the Thesis Defense**

| **Date** | **Tasks** | **General Instructions** |
| --- | --- | --- |
| ***post thesis submission*** | 1. prepare presentation slides | **send slides (as a .pdf file) to advisor(s) and Juan**, no later than 8:00 a.m. (GMT+1), the day of the scheduled defense |
| ***defense***  ***date*** | 1. deliver the presentation | Time limited: 15’ for B.Sc. and 20’ for M.Sc. |
| 1. answer raised questions | ca. 20’ for Q&A |



**Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_02.08.2024\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**5. Bibliography**

[1] Jayant R. Haritsa. 2019. Robust Query Processing: Mission Possible. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, April 2019. 2072–2075.<https://doi.org/10.1109/ICDE.2019.00242>

[2] Naveen Reddy and Jayant R Haritsa. 2005. Analyzing Plan Diagrams of Database Query Optimizers. *VLDB ’05: Proceedings of the 31st international conference on Very large data bases* (August 2005), 1228–1239.

[3] Jayant R. Haritsa. 2010. The Picasso database query optimizer visualizer. *Proc. VLDB Endow.* 3, 1–2 (September 2010), 1517–1520.<https://doi.org/10.14778/1920841.1921027>

[4] Goetz Graefe, Harumi Kuno, and Janet Wiener. 2009. Visualizing the robustness of query execution.<https://doi.org/10.48550/arXiv.0909.1772>

[5] Janet L. Wiener, Harumi Kuno, and Goetz Graefe. 2009. Benchmarking Query Execution Robustness. In *Performance Evaluation and Benchmarking*, 2009. Springer, Berlin, Heidelberg, 153–166.<https://doi.org/10.1007/978-3-642-10424-4_12>

[6] Viktor Leis, Bernhard Radke, Andrey Gubichev, Atanas Mirchev, Peter Boncz, Alfons Kemper, and Thomas Neumann. 2018. Query optimization through the looking glass, and what we found running the Join Order Benchmark. *The VLDB Journal* 27, 5 (October 2018), 643–668.<https://doi.org/10.1007/s00778-017-0480-7>

[7] Hannes Mühleisen, Danica Porobic, and Manuel Rigger. 2024. Ensuring the Reliability and Robustness of Database Management Systems (Dagstuhl Seminar 23441). *DROPS-IDN/v2/document/10.4230/DagRep.13.10.139* (2024).<https://doi.org/10.4230/DagRep.13.10.139>

[8] Hector Garcia-Molina, Jeffrey Ullman, and Jennifer Widom. 2013. Database Systems. Retrieved August 2, 2024 from<https://elibrary.pearson.de/book/99.150005/9781292037301>

[9] PostgreSQL: Documentation. Retrieved August 2, 2024 from <https://www.postgresql.org/docs/>

[10] Egor Rogov. 2023. PostgreSQL 14 internals. (2023). Retrieved August 2, 2024 from<https://edu.postgrespro.com/postgresql_internals-14_en.pdf>

**6. Use of AI Tools for Course Assignments, Theses, and Research Papers**

Prof. Dr. Volker Markl

Chair of Database Systems and Information Management (DIMA)

Technische Universität Berlin

June 7, 2023

**Policy Statement.**

1. Effective immediately, as a general policy, we do not permit the use of ChatGPT or other large language models (LLMs) in DIMA for the preparation of research papers, technical reports, theses, PhD dissertations, or any other original work.
2. The use of LLMs shall be strictly prohibited, as it on the one hand might prevent a fair grading of student performance, and on the other hand present the very serious risk of plagiarism. Similar to copying code from *Stack Overflow*, there is a spectrum with respect to when plagiarism occurs. However, in the scientific world one should be particularly aware and be careful with respect to this.
3. Every DIMA member should be aware of this policy and also point this out to your students and mentees. For course/project reports and theses, we will require students to sign a form that they confirm that they did not use any LLM tools, such as ChatGPT for the writing of their theses. Of course, this also applies to research papers and PhD dissertations.
4. Please be mindful of this matter and bring any violations of this policy to my attention, so that DIMA will be an environment free of the risk of plagiarism.



**Please sign below to acknowledge that you have read this policy and will adhere to it.**

**Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_\_06.08.2024\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. thesis complete [↑](#footnote-ref-0)
2. add an approximate date, the precise due date will be provided by the Prüfungsamt in writing at a later date. [↑](#footnote-ref-1)
3. system complete [↑](#footnote-ref-2)