Univ.-Prof. Dr.-Ing. Matthias Boehm
Technische Universität Berlin
Faculty IV - Electrical Engineering and Computer Science
Berlin Institute for the Foundations of Learning and Data (BIFOLD)
Big Data Engineering (DAMS Lab) Group

1 PPDS SoSe2024: Exercise – Building a Query Processing Engine

Published: Apr 13, 2024 (last update: Apr 13, 2024)
Deadline: Jul 01, 2024, 11.59pm

The project aims to provide practical experience in solving a complex task, getting used to working in a team and coordinating the progress via a version control system. The task is to implement a query processing engine that reads a local file using the Open-Next-Close (ONC) principle and performs operations like selections, filtering, projections, joins, and aggregations on top of the data. The preferred programming languages to implement the project are C, C++, or Java, for which we also provide an API. The expected result is a zip archive named `PPDS_Exercise_<group_ID>.zip` (replace `<group_ID>` by your group ID), containing your source code and a design document in PDF format.

Grading: This exercise is pursued in groups of 4 people (one submission). The grading is a *pass/fail* for the entire team. Exercises with $\geq 50/100$ points are a pass, and the quality expectations increase with the team size.

1.1 Implementation (45/100 points)

Implement the functionality provided in the existing API. The API and benchmark are available for C/C++ and Java. If you decide to use a different programming language, you have to port the API and benchmark yourself. To ensure good collaboration within your group we encourage you to set up your own private GitHub/GitLab project. The goal of the project is to implement the functionality given in the API in an efficient manner. For further information about the principles used in the practical please look at the teaching material provided in the accompanying lectures. You are allowed to create as many additional files and classes as needed for your implementation, however, please do not change the existing code.

Expected Results: Code for your own query processing engine.

1.2 Tests (10/100 points)

In addition to the existing *basic_test*, you have to write your own unit tests and integration tests to ensure that your implementation is correct. Tipp: Also think about special cases that you might want to test.

Expected Results: Additional tests that should be located in the test directory of the project.

1.3 Performance Goal (15/100 points)

To test how the efficiency of your project we will run the *benchmark* against your implementation on our scale-up server:

- HW: Two Intel Xeon Gold 6338 CPUs@2.2-3.2 GHz (64 physical/128 virtual cores), 1 TB DDR4 RAM, 16x SATA SSDs (in RAID-0).

- OS: Ubuntu 20.04
- C/C++ Compiler: gcc/g++ 11
- Java-11: Openjdk 11.0.20

To receive points for this task your implementation has to reach the performance goal of

$$T(SUT) < T(ref\_impl)/4$$

where *SUT* refers to the provided C++ reference implementation. Please refer to the *README.md* of the project for further information on how to run the benchmark.

1.4 Design Document(15/100 points)

In addition to the implementation, you are required to write a design documentation where you describe the task you solved and how you solved it. You should also mention pitfalls you encountered and are encouraged to address the limitations of your implementation. The documentation can either be written in English or German language.

Expected Results: A PDF document of max. 5 pages.

1.5 Project Presentation (15/100 points)

The results of your work are presented on **Jul 08, 2024, 4 pm** in the lecture hall where all groups have to attend. Your group will give a presentation showing the results of your project work. The presentation should last 8 minutes followed by a 2-minute discussion.

Expected Results: PowerPoint slides or PDF file.