DBenVis: A Visual Analytics System for Comparing DBMS Performance via Benchmark Programs

Yoojin Choi
Seoul National University

Juhee Han
Seoul National University

Daehyun Kim
Seoul National University

Figure 1: DBenVis system. **A.** Users can choose benchmarks and upload, delete, or rename their result files. **B.** Query Plan Visualization shows query execution plans as collapsible trees, where the size of each node is proportional to a selected metric. **C.** TPC-H Duration provides a comparison of the performance between DBMSs. **D.** The TPC-H Specific Query Duration provides a detailed performance analysis of the selected query. **E.** The sysbench Result visualizes time series data of selected metrics. **F.** The Compare View supports the comparison of average values for the selected metrics.

Abstract

Database benchmarking is an essential method for evaluating and comparing the performance characteristics of a database management system (DBMS). It helps researchers and developers to evaluate the efficacy of their optimizations or newly developed DBMS solutions. Also, companies can benefit by analyzing the performance of DBMS under specific workloads and leveraging the result to select the most suitable system for their needs. The proper interpretation of raw benchmark results requires effective visualization, which helps users gain meaningful insights. However, visualization of the results requires prior knowledge, and existing approaches often involve time-consuming manual tasks. This is due to the absence of a unified visual analytics system for benchmark results across diverse DBMSs. To address these challenges, we present DBenVis, an interactive visual analytics system that provides efficient and versatile benchmark results visualization. DBenVis is designed to support both online transaction processing (OLTP) and online analytic processing (OLAP) benchmarks. DBenVis provides an interactive comparison view, which enables users to perform an in-depth analysis of performance characteristics across various metrics among different DBMSs. Notably, we devise an interactive visual encoding idiom for the OLAP benchmark to represent a query execution plan as a tree. In the process of building a system, we propose novel techniques for parsing meaningful data from raw benchmark results and converting the query plan to a D3 hierarchical format. Through case studies conducted with domain experts, we demonstrate the efficacy and usability of DBenVis.

1 Introduction

In the field of database management systems, performance analysis is one of the most important aspects of data management [5]. Hence, database benchmarking is essential for evaluating and comparing the performance characteristics of diverse DBMSs across varying workloads. DBMS researchers and developers benefit from database benchmarking, using it to compare multiple DBMSs in terms of performance, query plan, and resource utilization. It also helps evaluate the performance of their optimization or newly-developed DBMS compared to existing ones. Companies can also benefit by gaining valuable insights from the results and making an informed decision when selecting the most suitable DBMS for their needs. In order to fully leverage the benchmarking tools, such as in the cases described above, adequate visualization of benchmark results is necessary.

However, the process of visualizing the benchmark results is often cumbersome, involving the extraction of data, selection of suitable visualization tools, and the actual execution of visualization. This time-consuming procedure requires specialized expertise in benchmarks and visualization tools. Furthermore, there lacks a unified system that allows users to automatically compare benchmark results across multiple DBMSs. Another major challenge is the visualization of query execution plans, which are crucial for performance analysis. While existing tools such as PostgreSQL Explain Visualizer [3] and MySQL Visual Explain Plan [2] provide visualization of query plans, the problem is that they are limited to specific DBMS and use distinct visualization designs. This results in the absence of a comprehensive, cross-DBMS visual analytics solution.

To address these challenges, we introduce DBenVis, an interactive visual analytics system designed to provide user-friendly and effective visualizations of benchmark results and query plans. DBenVis supports both OLTP and OLAP workload and provides handy

interaction features. The system employs a line chart to depict time series data of various performance metrics and a bar chart to enable comparative analysis for OLTP benchmarks. For OLAP benchmarks, the system uses various bar chart types, such as grouped and stacked bar charts, for a thorough performance analysis. It also visualizes query plans as trees. We also evaluate our system through case studies with domain experts to confirm the system's efficacy and usability.

The major contributions of this paper are as follows:

- We develop techniques for extracting essential data from raw benchmark results and parsing and rearranging `EXPLAIN` results of various DBMSs into a D3 hierarchical format.
- We implement novel and unified visualization designs tailored to OLAP and OLTP benchmark results and query plans.
- We provide a user-friendly application designed to visualize diverse metrics comprehensively, providing detailed analysis.

The rest of the paper is organized as follows. Section 2 covers related works, and Section 3 defines requirements and tasks. Section 4 presents a comprehensive explanation of DBenVis and its contributions. In Section 5, we present the evaluation results. Lastly, the conclusion with discussions and future works is covered in Section 6.

2 Related Work

We introduce the research areas related to DBenVis, primarily focusing on two domains: benchmark result visualization and query plan visualization. Our work gets inspiration from existing studies and attempts to overcome some of the limitations in these areas.

2.1 Benchmark Result Visualization

Benchmark result visualization is a key aspect in the analysis of database performance. Several popular libraries are used in the visualization process, each with unique features and requirements. Matplotlib [6] is among the most popular Python libraries aimed at creating interactive visualizations. D3 [4] is one of the most famous web-based JavaScript libraries for visualization. Numerous other visualization libraries exist, including Seaborn [10] and FusionCharts. Although these libraries serve as a powerful tool for visualization, the challenge is that they require users to preprocess the raw benchmark results. This requires users to have prior knowledge of result data and the library, making it time-consuming.

HammerDB [1] is an open-source database benchmarking tool that allows loading and evaluating DBMS performance using TPC benchmarks and facilitates result visualization. However, its utility is constrained since it supports only a limited range of benchmarks and database systems. These limitations highlight the need for more versatile and user-friendly tools in database benchmark visualization.

2.2 Query Plan Visualization

The query planner in DBMS generates possible plans for submitted queries and examines each of these possible execution plans, ultimately selecting the execution plan that is expected to run the fastest. The selection of the right plan to match the query structure and the properties of the data is critical for good performance, so the system includes a complex planner that tries to choose the good plan. `EXPLAIN` command is used to see what query plan is chosen. The query plan generated from `EXPLAIN` structure follows a tree format, with scan nodes at the bottom level of the tree. For queries that require joining, aggregation, sorting, or other operations, additional nodes are present to execute these operations.

There are only a few query plan visualizers available. The PostgreSQL Explain Visualizer (PEV) [3] provides an interactive interface for visualizing query plans in a tree format. Users can engage with the visualization through clicking, zooming, and panning interactions. Although not as comprehensive as PEV, MySQL does provide its own visualization tool for explaining plans. This tool facilitates a basic understanding of query execution plans within the MySQL environment. Additionally, as far as we surveyed, no query plan visualizer is available for MariaDB. This notable absence highlights the need for a unified visualizer to compare and analyze query plans across various DBMSs.

3 Requirements and Tasks

We conducted a survey to identify limitations in existing works related to benchmark results and query plan visualization. Based on the challenges defined, we organized them into three analytics tasks and formulated corresponding requirements.

3.1 Task abstraction

We formulated three significant tasks and corresponding questions for each task as follows:

T1. **Comprehensive DBMS evaluation.** This task involves an evaluation of DBMS performance across various benchmark programs, including resource usage and query plans. *What are the average duration or TPS for each DBMS?*

T2. **In-depth DBMS comparison.** This task aims to compare DBMSs extensively by considering various metrics (*e.g.*, performance and query plan). *Which operation acts as a bottleneck? What are the major differences between two DBMSs in terms of performance? How do they vary in handling rows and costs?*

T3. **Explore and analyze details.** This task contains exploring additional details (*e.g.*, expected cost of an operator in query plans). *What are the cost implications of specific query plan operators?*

3.2 Requirement Analysis

Based on the above tasks, we formulated the requirements that DBenVis must satisfy as follows:

R1. **Visualize benchmark results.** The system should provide a comprehensive view of DBMS performance on benchmark programs. OLTP benchmarks include visualizing the TPS in time series and the average TPS. For OLAP benchmarks, it should visualize the duration of each query and query plans during the query execution. (T1 and T2)

R2. **Support comparison.** The system should support comparisons between DBMSs regarding performance and query plans. Users should be able to add or delete DBMSs, and the view should respond interactively. (T2)

R3. **Provide interactive dashboard.** The system should provide interactive features to support in-depth analysis. Users can upload result files and define the DBMS name or use predefined names as legends in plots. For the OLTP workload, users can select the specific range of time they want to explore. For OLAP, users can select the specific queries they want to explore. (T3)

4 The DBenVis System

We introduce DBenVis, a visual analytics system for DBMS researchers and developers seeking to visualize benchmark results in an easy and efficient way. This section describes data reconstruction techniques developed to extract meaningful data from benchmark results and rearrange query plans into hierarchical format. Also, novel visualization designs within DBenVis are presented.

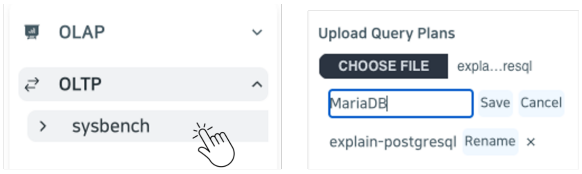


Figure 2: User interface for benchmark selection and file management.

```
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 4
Report intermediate results every 1 second(s)

Threads started!

[ 1s ] thds: 4 tps: 564.18 qps: 11345.53 (r/w/o: 7954.45/2258.72/1132.36) lat (ms,99%): 16.71 err/s: 0.00 reconn/s: 0.00
[ 2s ] thds: 4 tps: 486.91 qps: 9726.23 (r/w/o: 6803.77/1948.64/973.82) lat (ms,99%): 36.89 err/s: 0.00 reconn/s: 0.00
[ 3s ] thds: 4 tps: 525.66 qps: 10519.20 (r/w/o: 7368.25/2099.64/1051.32) lat (ms,99%): 35.59 err/s: 0.00 reconn/s: 0.00
...
[ 1799s ] thds: 4 tps: 514.82 qps: 10310.45 (r/w/o: 7221.51/2059.30/1029.65) lat (ms,99%): 51.94 err/s: 0.00 reconn/s: 0.00
[ 1800s ] thds: 4 tps: 520.25 qps: 10413.10 (r/w/o: 7289.57/2083.02/1040.51) lat (ms,99%): 47.47 err/s: 0.00 reconn/s: 0.00
Latency histogram (values are in milliseconds)
       value  ------------ distribution ------------ count
       1.608 |                                       3
       1.637 |                                       2
       1.667 |                                       1
...
SQL statistics:
    queries performed:
        read:                           12761826
        write:                          3646236
...
```

Figure 3: Partial snapshot of the result obtained from sysbench execution.

4.1 Interactive File Managements

This section introduces the interactive file management features in DBenVis. The process involves selecting a benchmark, uploading raw result files, and interactively modifying the view. DBenVis provides users with the option to choose between benchmarks. It currently supports two benchmarks: sysbench [7] and TPC-H [9]. Users can proceed to upload the corresponding result files obtained from benchmark execution. Multiple files can be uploaded for comparison purposes, with the interface adjusting responsively. Also, the interface allows the deletion of unwanted files and the renaming of files to apply customized names in the charts. Figure 2 illustrates the benchmark selection and file upload view of DBenVis.

4.2 Online Transaction Processing Benchmark

Our system supports sysbench, an open-source multi-threaded benchmarking tool designed to execute OLTP workloads. Figure 3 illustrates a partial snapshot of the result obtained from sysbench execution. It outputs a snapshot of workload statistics every second, including metrics such as transactions per second (TPS), queries per second (QPS), and latency per transaction in milliseconds. The final report provides users with average metrics values.

One challenge arises from the extensive set of intermediate results presented on a second-by-second basis, which makes it difficult for users to analyze the overall performance. The problem worsens when there are multiple result files to compare. Also, the final report lacks flexibility, as users cannot specify the time range for the average calculation. This can be a critical issue when the time range that exhibits unstable performance (*e.g.*, early phase of benchmark execution) should be excluded from the average calculation.

4.2.1 Data Processing for sysbench Results

We outline the data preprocessing techniques implemented to address the challenges presented above. Our approach involves parsing the raw result file, organizing each statistical metric data into an array format arranged chronologically, and subsequently passing the processed data to the module responsible for chart generation. The detailed process proceeds as follows: Upon the user submission of a file, DBenVis parses the results and stores TPS, QPS, and latency. Additionally, the system calculates the average values for each metric. When a user brushing event is dispatched, the system recalculates values exclusively based on the selected data points. The updated averages are then applied to visualization components.

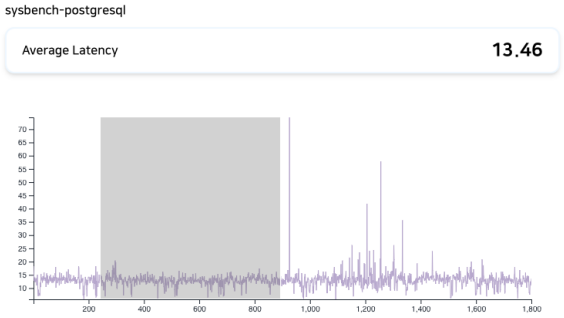


Figure 4: The card component and line chart with brush interaction for detailed analysis.

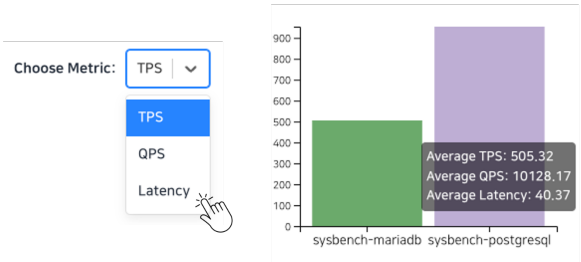


Figure 5: Interaction features of metric selection and tooltips.

4.2.2 sysbench Result Visualization Designs

This section presents the design of visualizations and interaction features aimed to ensure an effective and user-friendly interface. The processed metric data are visualized as a line chart, an effective chart type for visualizing trends of time series data over time intervals. Each result file is visualized through an individual line chart, with a corresponding card component above, which provides the average value of the selected metric. The bar chart positioned below enables an intuitive comparison of average values. We employ a bar chart because of its excellence in depicting the relationship between numeric values.

As shown in Figure 4, the line chart contains brush and zoom features, allowing users to focus on desired time ranges. When users incur a brushing event, the chart zooms in on the selected range, and average values are dynamically updated in the card component and bar chart. Figure 5 demonstrates how users can select a specific metric to visualize, which provokes responsive adjustments in the line chart and bar chart. Also, tooltips are implemented in the bar chart to facilitate users viewing all three metrics at once.

4.3 Online Analytic Processing Benchmark

DBenVis supports TPC-H benchmark, a decision support benchmark consisting of a suite of 21 business-oriented ad-hoc queries. The result file of TPC-H consists of a retrieved table as a response to the query and the corresponding processing time as shown in Figure 6. The system also provides visualization of query execution plans obtained using the `EXPLAIN` command. Currently, the system supports query plans of three high-ranking relational DBMSs: PostgreSQL, MariaDB, and MySQL [8].

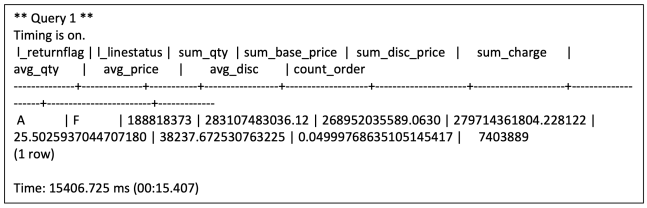


Figure 6: Results of TPC-H Query 1 execution.

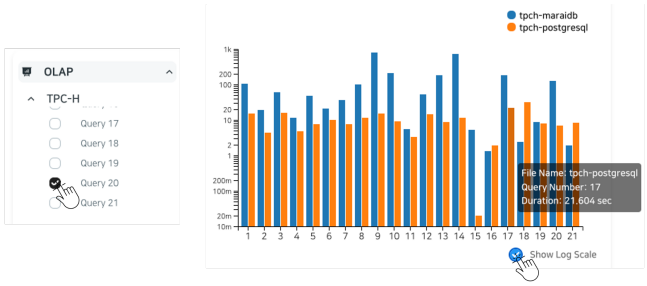


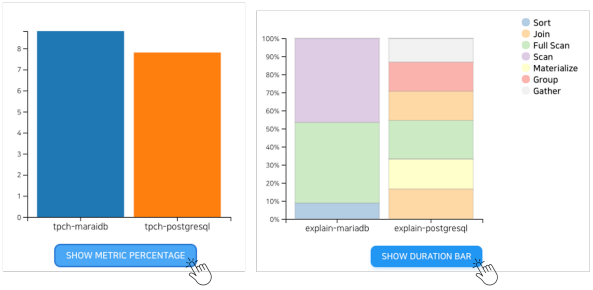
Figure 7: Interaction features of grouped bar chart.

4.3.1 TPC-H Result Visualization

This section introduces the data processing techniques and visualization designs employed for the comparative view on the right side of the interface. Similar to the data processing approach used in sysbench, DBenVis parses the duration from the raw result files and then stores the data in an array format based on query number sequence.

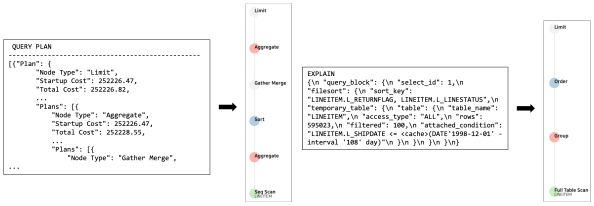
The processed data are visualized as a grouped bar chart, providing a broad overview of DBMS performance comparison. The interactive grouped bar chart is illustrated in Figure 7. The chart divides groups by file and maps unique colors for each group to enable easy identification. We implement a showing log scale feature to address the limitation of representing substantial differences in duration among queries and DBMSs. Users can turn the feature on or off using the 'Show log scale' button to see the chart in the log scale or linear scale respectively. Additionally, tooltips are implemented, offering detailed information such as file name, query number, and duration. Users can click bars within the grouped bar chart to select the desired comparison criteria.

Through the interaction of clicking bars within a grouped bar chart or checkbox in the sidebar, users can perform an in-depth comparative analysis within the selected bar chart positioned at the bottom (Figure 8a). The selected bar chart displays detailed comparisons, enabling users to perform a more comprehensive examination of the selected query.



(a) Duration bar chart (b) Percentage Distribution

Figure 8: Bar chart for selected query with bars representing duration and percentage distribution of selected metric.



(a) PostgreSQL (b) MariaDB

Figure 9: Query execution plans and generated trees for TPC-H Query 1.

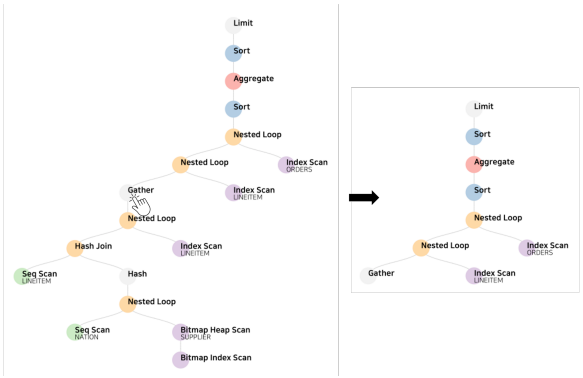


Figure 10: Collapsible tree.

4.3.2 Data Processing for `EXPLAIN` Results

This section describes the techniques developed for processing the `EXPLAIN` results. Figure 9 illustrates the query plans obtained from PostgreSQL and MariaDB for TPC-H Query 1. Notably, the challenge lies in addressing the unique format of `EXPLAIN` results and the variation in terminology used for operations, making direct comparisons of query plans complicated. Furthermore, the original result format does not adhere precisely to JSON standards, preventing the data from directly using the D3 hierarchy or `stratify`.

To overcome these challenges, we first introduce terminology mapping functions that facilitate easier comparison. The function ensures that even when the terminologies differ for the same operators, the system can identify operators as the same operators. A more significant contribution lies in developing techniques to parse and rearrange `EXPLAIN` result files for all the DBMSs DBenVis supports. We employ distinct techniques tailored to each unique result format, resulting in a uniform hierarchy format for the query plan data. This standardized format allows data to be passed to the D3 hierarchy, enabling the construction of root nodes and visualization as a tree. All additional properties of the operator are stored within the nodes and used for in-depth analysis.

4.3.3 `EXPLAIN` Result Visualization Designs

We describe the query plan visualization designs and user interactions. The goal is to build an interactive and visually intuitive analysis tool for users to better understand query plans and examine various operations' impact on performance. The design choice of the tree is motivated by the inherent tree format of query plans, which has a root node and child nodes connected by edges. Figure 10 illustrates the interactive features of the tree, where users can expand or collapse the tree by clicking on nodes. This enables users to focus on specific segments of the query plan they want to analyze. The

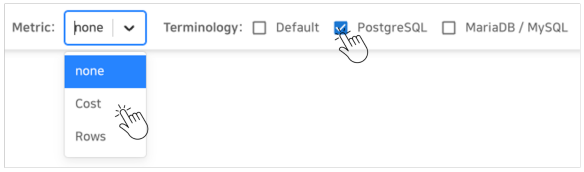


Figure 11: Interaction features of metric and terminology selection.

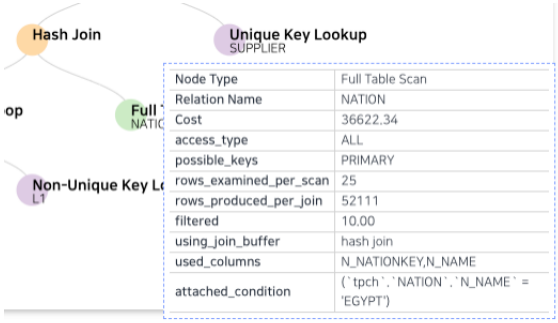


Figure 12: Tooltips with node that contains additional properties of operator.

distinct node colors correspond to different operators, enhancing the comparative analysis between various DBMSs. Each node represents an operator, with the node size proportional to the selected metric.

As shown in Figure 8b, a stacked bar chart appears by clicking on the 'Show metric percentage' button. The chart is designed to show each operator's relative chosen metric (e.g., cost or accessed rows estimated by planner) proportions, assuming a total cost of 100 percent. In Figure 8, users can select a metric to visualize. Furthermore, to reduce the mismatch arising from diverse terminologies, users can standardize the terminology used throughout the tree by selecting the terminology they want to use. For detailed analysis, the tooltip illustrated in Figure 12 appears when the user hovers over a specific node. Tooltips include additional properties such as cost, rows, and attached condition.

4.4 System Architecture and Implementation

The architecture of DBenVis consists of two primary modules: preprocessing and visualization. The preprocessing module serves two functions. First, it extracts meaningful data from the uploaded files. Second, it parses and reorganizes `EXPLAIN` result files into a hierarchical format. The visualization module is dedicated to presenting processed data through an interactive interface. This module is developed as a single application in JavaScript, utilizing D3.js for data-driven visualizations and the React library for building user interfaces. The system layout is optimized for full-screen viewing, particularly at a resolution of 1920 × 1080. This ensures that users can easily access and interact with the comprehensive data visualizations provided by the system.

5 Evaluation

To evaluate the efficacy and user-friendliness of our visual analytics tool, we engaged two experts in the field for interviews. Prior to these interviews, the experts were provided with a concise overview and detailed explanations of the tool's functionalities and the data visualizations implemented. During the informal interviews, we gathered valuable insights, including suggestions for additional features, opinions on the visual design, and feedback on user interaction with our system. Both experts concurred that our tool has the potential to significantly enhance analytics for its target users, particularly emphasizing the need for further refinement in user interaction to ensure a smoother, more intuitive experience.

For a more structured assessment, we formulated three key questions, each linked to specific requirements of DBenVis:

R1. **Visualize benchmark results.** Does the DBenVis effectively visualize the benchmark results?

R2. **Support comparison.** Does the DBenVis support easy comparison of benchmark results?

R3. **Provide interactive dashboard.** Are the interactive features of DBenVis useful and enhance user experience?

5.1 Fulfillment of Requirements

We asked participants to respond to a series of statements to determine if DBenVis met its intended requirements. This was done using a 5-point Likert scale survey. Analysis of the survey results revealed a general consensus among participants that DBenVis effectively meets the set requirements. This was reflected in the average scores: 4.5 for R1, 4 for R2, and a perfect score of 5 for R3. These scores indicate that there is relatively little variation in the satisfaction levels across different requirements. Additionally, the participants' responses reflected a favorable opinion of the tool's usability, with an average rating of 4.5.

5.2 Discussions

Future iterations of DBenVis offer possibilities for refinement and expansion. One key area of enhancement involves integrating the actual execution times of database operators into the system. This feature will allow users to more accurately assess the effectiveness of a query optimizer's selected plan, going beyond simple cost estimations. Furthermore, based on practical suggestions received during our case studies with experts, we plan to refine the query plan visualization aspect of the system. By enabling the aggregation of costs from child nodes in a collapsed tree view, DBenVis will offer a more comprehensive and nuanced analysis of costs, improving both user interaction and analytical depth.

Another significant addition under consideration is the capability to visualize and compare a variety of potential query plans generated by a DBMS optimizer. This enhancement would shed light on the spectrum of query strategies evaluated by the optimizer, providing users with deeper insights. Also, these planned improvements and new features are poised to take DBenVis to the next level in terms of functionality and user experience. We are committed to exploring and implementing these ideas in our forthcoming developments of DBenVis.

6 Conclusion

We presented DBenVis, an interactive visual analytics system designed for the comparative analysis of DBMSs performance. The contribution of this paper is threefold. First, we developed techniques to extract meaningful data from raw result files and transform it into the required data format for visualizations. Second, we implement novel and unified visualization designs to explore results obtained from OLAP and OLTP benchmark executions, as well as `EXPLAIN`. Lastly, we developed a user-friendly application with interactive features to enhance the user experience. Through case studies conducted with domain experts, we demonstrated the efficacy and usability of DBenVis. The implementation of our system is publicly available on GitHub at github.com/ratdbs/dbenchvis-front.

References

[1] Hammerdb. https://www.hammerdb.com/.
[2] Mysql visual explain plan. https://dev.mysql.com/doc/workbench/en/wb-performance-explain.html.
[3] Postgres explian visualzer. https://www.pgexplain.dev/.

[4] M. Bostock, V. Ogievetsky, and J. Heer. D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. doi: 10.1109/TVCG.2011.185

[5] D. E. Difallah, A. Pavlo, C. Curino, and P. Cudre-Mauroux. Oltp-bench: An extensible testbed for benchmarking relational databases. 7(4):277–288, dec 2013. doi: 10.14778/2732240.2732246

[6] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55

[7] A. Kopytov. sysbench. https://github.com/akopytov/sysbench.

[8] solid IT. DB-Engines. https://db-engines.com/.

[9] TPC Benchmarks. TPC-H. https://www.tpc.org/tpch/.

[10] M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021