# Position Statement: The Case for a Visualization Performance Benchmark

Leilani Battle
MIT

Remco Chang
Tufts University

Jeffrey Heer
University of Washington

Mike Stonebraker
MIT

## 1 Introduction

Visualizations are an invaluable tool in the data analysis process, as they enable scientists to explore and interpret billions of datapoints quickly, and with just a few rendered images. However, many visualization systems are unable to keep up with the unprecedented accumulation of data through remote sensors, field sensors, medical and personal devices, social networks, and more. This is due to certain assumptions that many of these tools rely on, such as the assumption that these systems can store entire datasets directly in main memory. With so many datasets massive datasets available, ranging from the NASA MODIS satellite imagery dataset[3] to the Internet Movie Database [4] to Twitter streams [1], this assumption no longer matches reality.

In response, new breed of visualization system has become the norm, where the data resides in a database management system (or DBMS) running on a remote server, and a visualization front-end running on a client machine (e.g., a laptop) issues database queries (e.g., SQL queries) to retrieve data from the DBMS (henceforth referred to as **visual data systems**). In this context, recent visual data systems have focused on enabling interactive exploration of large datasets, where the user observes only very low latencies (i.e. 500ms of latency or less) when performing interactions within the visualization front-end. To support this level of interactivity, recent visual data systems have employed a variety of strategies, including precomputation [22, 6, 21, 24], sampling [25, 16, 18, 15], and predictive DBMS query execution [19, 6, 9].

However, given the diversity of techniques, domain problems, system architectures, software platforms, etc., it has been difficult for the visualization community to compare these algorithms and techniques and decide which one is most suited for their needs. For example, while sampling is intuitive and easy to use, the introduction of sampling to the data processing pipeline introduces uncertainty in the resulting visualization. Conversely, pre-computation and pre-fetching techniques can provide precise answers, but at the cost of high storage requirement and runtime data transfer.

In this paper, we propose a new benchmark that allows for systematic and repeatable measurement of the different visual data systems. Our benchmark is inspired by the different ways that the database community and the visualization (and visual analytics) community evaluate systems. In the database community, the standard approach to comparing system performance is by running a *benchmark* using each system in question, such as the TPC-H benchmark [13], and comparing the results (e.g., the average response time or average latency). Though several database benchmarks exist, these benchmarks are designed for specific use cases, like data warehousing (i.e., TPC-H), genomics [29], and transactional processing [14, 11], which do not include visual analytics as a high-priority use case. As such, they are a poor approximation for gauging visualization performance.

In contrast, visualization benchmarks, like the Visual Analytics Benchmark Repository [26], focus on user perception and productivity (i.e., how well and how thoroughly a user can analyze a dataset given a particular visualization tool). As such, they provide limited support for performance evaluations (i.e., how fast the system runs when a user is interacting with it), as well as direct comparisons of specific analytical operations (i.e., comparing performance for specific queries).

We propose that a new benchmark should combine the best of both worlds by blending methodology from the database and the visualization communities. In particular, we suggest the following:

1. ease of customization of the benchmark, to support a variety of visual data system architectures
2. ease of interpretation of benchmark results, to ensure that users can compares systems using fair and transparent measures of system performance
3. Realistic scenarios, to ensure that the benchmark accurately represents how users analyze data through visual data systems.

Given these design considerations, in the rest of the paper we discuss a plan for developing the new benchmark, including data, queries, and comparable measures for evaluating the performance of large-scale visual data systems.

## 2 Current Evaluation Techniques

To better understand how a unified visualization benchmark is helpful, we first examine the limitations of existing evaluation methods. In this section, we present the different evaluation methods used by recent visual data systems published in both the database and the visualization communities, including imMens [22], SeeDB [30], A-WARE [15], and ForeCache [6]. We summarize the commonalities between these evaluation methods, and propose the creation of a unified visualization benchmark based on the integration of these methods.

### 2.1 Evaluation Setups for Modern Visual Data Systems

Here, we focus on analyzing how recent visual data systems are evaluated. Specifically, we discuss how datasets and workflows are selected to evaluate these systems, and the specific measures that are used to compare their performance.

**Datasets**: Currently, most datasets are selected or generated based on three features:

1. number of records,
2. dataset complexity (i.e., number of columns and data distributions both within and across columns),
3. and relevance (i.e., whether users of the visual data system already have a vested interest in the dataset).

Real-world datasets appear to be selected mainly for number of records, to test how systems scale, as well as for relevance, to demonstrate how systems perform with real world use cases. For example, the NASA MODIS dataset is used to test the ForeCache visualization system both in terms of scale (e.g., terabytes of data are processed) and and in applicability to users (e.g., earth scientists are recruited for evaluation) [6]. Synthetic datasets are created primarily to test systems under various distribution-related conditions (e.g., with varying distributions within and across data columns [20, 22, 19, 30]).

- *Leilani Battle is with MIT. email: leilani@csail.mit.edu*
- *Remco Chang is with Tufts University. email: remco@cs.tufts.edu*
- *Jeffrey Heer is with the University of Washington. email: jheer@cs.washington.edu*
- *Mike Stonebraker is with MIT. email: stonebraker@csail.mit.edu*

*Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014; date of publication xx xxx 2014; date of current version xx xxx 2014.*
*For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.*

An ideal dataset merges the best features of real-world and synthetic datasets: it would have direct real-world applications, and it would have interesting size and distribution properties. Unfortunately, finding all of these features within a single dataset is very difficult. Instead, it may make more sense to find several real-world datasets that share some of these properties, and create data generators to mimic them. This approach has been successfully utilized for benchmarks within the database community [29, 13], and thus would lend itself well to the development of a centralized visualization benchmark.

**Workflows/Workloads**: In addition to evaluating the performance of a data visual system based on the size and complexity of the input data, another common evaluation criteria is to test its performance based on the different usage scenarios, or **workflows** or **workloads**. Two methods are generally used to produce a workflow or workload with which to test a visual data system:

1. interaction logs (or DBMS query logs) are collected (either through a user study [6, 5, 19] or retrieved from an existing system or organization [19]) and used to drive performance experiments
2. a potential user workflow is manually created and translated into a log of interactions (or DBMS queries) to evaluate for performance [15, 20, 22, 10, 30]

Note that the first method is restricted to the specific system and dataset used to generate the logs, and thus may have limited applicability. Furthermore, this technique requires significant effort (i.e., conducting a user study) to produce usable results. However the advantage of this method is that real user behaviors are captured in the data. As such, any major performance gains demonstrated with this evaluation method are strongly supported by real-world use cases.

We have found the second evaluation method (manually creating a workload) to be more popular when evaluating the performance visual data systems. The clear advantage of this method is its applicability: any system can be evaluated using any reasonable dataset, using this method. Furthermore, because a user study does not have to be conducted, visual data systems can be evaluated much faster. However, the obvious drawback of this approach is that no external users are involved in the evaluation, making it more challenging to argue for strong performance for real-world applications.

A better evaluation approach could work as follows: analyzing interaction logs from a study with real users, and then develop realistic (but synthetic) workflows from the logs [15]. This method is similar to how benchmarks are developed for DBMSs [29, 12, 13]. However, the challenge for a new visualization benchmark will be to ensure that the synthetic workflows accurately represent visual analytics tasks *in general*, opposed to tasks that are specific to a single system.

**Evaluation Measures**: Given a **dataset** and a **workload**, we have found time to be the standard measure used to evaluate performance for all visual data systems. However, the consideration of time can applied to different steps of the data analysis process:

1. System response time: similar to measuring query speeds for DMBSs, a system's response time to a user's interaction is commonly used to evaluate the performance of a visual data system.
2. "Cold-start" time: given how fast-paced and varied data analysis tasks can be, it is often important to consider how quickly a user can begin to explore a new dataset with a visual data system. This initialization process encompasses the pre-compuation time of the visualization system, or the time required to build any supplemental data structures that are needed to drive visualization optimizations (e.g., samples, machine learning models, indexes, and data cube structures).

While the measure of system response time is common place, surprisingly, we found cold-start time to be ignored in the vast majority of performance evaluations[1]. For data cube-like structures, the pre-computation time can be very slow (e.g., could take hours [21, 24, 5, 22]), which can clearly have a significant impact on how a user interacts with the system. Even sampling techniques can have a long pre-computation time when executed over massive datasets. As such, we need metrics that represent a holistic view of the visual analysis process, and thus a more realistic view. A visualization benchmark would act as a centralized point for discussion of new and relevant evaluation measures, as well as provide a clear and well-documented set of measures for evaluating visualization system performance.

[1]except for the evaluations of the Nanocube [21] and Hashedcube [24] structures, and the Sculpin/ForeCache system [5]

### 2.2 Methods for Comparing Visual Data Systems

Currently, visual data systems are typically compared for performance using two different methods. These comparison methods are as follows (for a new system named System A, and an existing competitor named System B):

1. the techniques or algorithms of interest from System B are re-implemented in System A to make comparisons (e.g., the comparison method used for the ForeCache [6], imMens [22], SeeDB [30], and DICE [19] visualization systems);
2. System A and System B are run directly with the same experimental settings, and their outputs are compared (e.g., the comparison method used for the A-WARE visualization system [15], and standard comparison method for DBMSs).

However, most recent systems favor the first comparison method (i.e., to re-implement system logic) over the second (i.e., directly running other systems). This could be due to the difficulty of acquiring and then running the code for competing systems, and as well as issues in acquiring the dataset(s) used to evaluate competing systems (e.g., loading the terabytes of NASA satellite imagery visualized by ForeCache). There are two concerning aspects to this trend. First, it may be the case that some of these new visualization systems are difficult to actually run and use by people other than the original developers. Ideally, our community should be developing tools and techniques that others can easily use and build upon. However, if other database and visualization experts are unable to run these systems, it is unlikely that non-expert users are running them.

Second, the visual data systems mentioned above compare to at most two other systems using via re-implementation, and it is unlikely that this method will scale. Given the growing interest in cross-area collaborations between the database and visualization communities, the number of visual data systems and modules within this space will only continue to increase, and rapidly. Expecting researchers to re-implement optimization logic for every new visualization system that comes out is simply unrealistic, and becomes more outrageous as the community continues to grow.

Ideally, systems would be compared directly using the same dataset, workload, and performance measures (i.e., using the first comparison method). But given the large number of possible datasets to use, the plethora of use cases supported by different visualization systems, and variability of evaluation methods, selecting a single experimental setup appears to be a daunting task.

However, we have seen successful implementations of this comparison method in the database community, in particular the TPC benchmarks [23]. Given that the performance evaluations done for visualization systems are similar to those utilized to evaluate DBMS's, a visualization benchmark appears to be a viable alternative to current visualization evaluation methods. Furthermore, by adhering to a widely-accepted benchmark, we can encourage our community to produce easy-to-use systems, in turn supporting wider-spread usage of existing systems and cleaner performance comparisons for new systems.

### 2.3 The Need for a Standardized Benchmark

With the variety of methods used to measure performance, it is extremely difficult to objectively compare one visualization system with another, based on reported performance results. We described 3 dataset selection/creation methods, two workload creation methods, and two system comparison methods that are currently used, resulting in 12 possible evaluations, none of which can be directly compared with another. This problem will only grow worse as our community

continues to grow. However, we also see that these methods can be merged, and that the resulting hybrid methods share strong similarities with the design of existing database benchmarks. We believe this provides strong evidence not only for the need but also the viability of a new data-visualization management benchmark.

Given the similarities between evaluating scalable visual data systems and DBMSs, a natural starting point is to see if any existing benchmarks could be re-purposed as a visualization performance benchmark. In the next section, we discuss the pros and cons of existing benchmarks in the database and visualization communities, and how design decisions from these benchmarks could be leveraged to develop a data-visualization management benchmark.

## 3 Past Benchmarks

Our goal is to develop a unified performance benchmark that enables systematic and repeatable measurement for visualization systems within a realistic evaluation environment. As a first step towards this goal, we review existing efforts in the database and visualization communities towards developing realistic benchmarks, and identify key properties that can be propagated to the design of the new visualization performance benchmark.

### 3.1 Database Benchmarks

The database community has a long tradition of publishing and utilizing performance benchmarks. For example, the Transaction Processing Performance Council [23] was founded in 1988 to develop benchmarks and to provide "objective, verifiable performance data to the industry" [23]. It has since developed benchmarks that are considered the gold standard for evaluating databases for transactional processing (TPC-C [14, 11], TPC-E [11]), online analytical processing (TPC-H [13]), and recently databases in virtual environments (TPC-V [27]).

Of these benchmarks, one of particular interest is the TPC-H benchmark, which is used to evaluate the performance of various database management systems in the context of data warehouses. The TPC-H benchmark simulates a data warehouse providing decision support for a retail company. The queries in the TPC-H benchmark are a mix of both analytical queries, for monitoring the warehouse(s), and update queries, for simulating real-time dataset maintenance.

The TPC-H benchmark is a popular evaluation tool for systems supporting Online Analytical Processing (or OLAP) queries. OLAP queries mainly feature aggregation operations to compute statistics, such as computing the count or mean for a given data attribute, and thus share significant overlap in the operations executed in visual analytics tasks (e.g., aggregation for bar charts, box plots, and heatmaps).

### 3.2 Visualization Benchmarks

From the visualization community, we focus on the Visual Analytics Benchmark Repository [26], which provides the data, submissions and solutions of past VAST and InfoVis Challenges. What makes this benchmark unique is the availability of ground truth for existing analysis tasks, across several different datasets (i.e., the *solutions* for each Challenge). From this information, one can calculate the accuracy of the answers submitted to the VAST Challenges (which are also part of the benchmark), and by extension evaluate the effectiveness of the visualization tools used to produce these answers.

The majority of VAST and InfoVis challenges, including the VAST Challenges of the last four years, involve analyzing hand-made and code-generated data. However, a small number of these competitions utilized real-world data instead. For example, the 200X InfoVis Challenge utilized the 2000 Census PUMS Dataset [8], a 1% subset of data from the 2000 US Census, and the InfoVis 2007 Challenge used a small subset of the Internet Movie Database Dataset [4].

Interestingly, this definition of benchmark in the visualization community (measures accuracy of analyses derived using a visualization tool) deviates from the definition used by the database community (measures database speed and throughput for a known set of queries). However, the data derived for the Visual Analytics Benchmark Repository is still applicable to a performance-driven benchmark.

### 3.3 Comparing the Benchmarks

Here, we discuss the positive and negative aspects of the provided benchmarks with respect to performance, as well as opportunities to bring the benchmarks together in an effort to develop a more effective performance evaluation for visualization tools.

We identify three major limitations to existing benchmarks that make then unsuitable for a visualization performance benchmark:

1. a lack of explicit analysis operations (or queries) for analysis tasks within the benchmark (Visual Analytics Benchmark Repository)
2. a lack of realistic use cases for visual analytics (TPC-H)
3. a lack of flexibility in the benchmark due to partial reliance on hand-made data (Visual Analytics Benchmark Repository)

**Lack of Explicit Analysis Operations**: While the Visual Analytics Benchmark Repository is an interesting candidate benchmark for evaluating visual data systems, it falls short because the Visual Analytics Benchmarks were designed to allow flexible analysis workflows. As such, the solutions provided in the Benchmark Repository often only contain vague descriptions of the analysis process, or the answer to the expected results (e.g. pointing out what the "outlier" is and why).

While this flexibility serves the visual analytics community well, as a candidate benchmark for visual data systems, it leaves the exact analysis steps needed to produce the appropriate results up to interpretation. Without a well-specified workflow, it is extremely difficult to compare the performance of two systems. This is akin to comparing the performance of two different DBMSs that are executing different queries. In contrast, the TPC benchmarks have published query sets, so database vendors and researchers know exactly what operations must be supported to run them. This also provides flexibility in how the TPC-H benchmark is utilized. For example, if certain operations are not supported, one can instead report on the *subset* of TPC-H queries their DBMS supports, which still provides valuable information about the performance of the DBMS, as well as its limitations.

**Lack of Realistic Use Cases**: A major drawback to the TPC-H benchmark for visual analytics is the fact that it simulates a data warehouse. Though this is certainly a valid visual analytics use-case (at least for industry), it is far from representative of the challenges and tasks that the visualization community aims to address. In addition, the TPC-H benchmark schema and queries are not representative of how a visualization tool issues queries to produce interactive visualizations. In comparison, the Visual Analytics Benchmark Repository is designed to simulate real-world visual analytics use cases.

**Lack of Flexibility**: Most benchmarks generate at least some of their input data through code (e.g., the Threat Stream Data Generator [31] and TPC-H benchmark [13]). However, most of the benchmarks in the Visual Analytics Benchmark Repository are small (a few GB, or less), and rely on handmade data, limiting their ability to scale. The TPC-H benchmark in comparison is fully code-generated, and includes scale factor parameter to increase the dataset size.

### 3.4 Useful Properties

Even though existing benchmarks are unsuitable for evaluating the performance of visual data systems, they have useful properties that could be propagated to a new data visualization management benchmark, including : 1) dataset customization (e.g., increasing or increasing dataset sizes), 2) an explicit workload (e.g., specific queries to be executed), and 3) realistic visual analytics tasks. We expand on these ideas in the following section with a set of high level design considerations for a future data visualization management benchmark.

## 4 Benchmark Design Considerations

Here, we highlight key features that should be incorporated in the new visualization performance benchmark. We combine methodology from the database and visualization communities into three high-level design goals: 1) ease of interpretation, 2) ease of customization, and 3) realistic scenarios. In the rest of this section, we explain each major design consideration. In the reminder of the paper, we refer to this proposed benchmark as the ***Visualization Performance Benchmark***..

### 4.1 Ease of Interpretation

A major challenge in evaluating the performance of different visual data systems is finding common ground on which to make a direct comparison. A single benchmark helps system designers to focus on a clear set of goals for improving system performance. The designer is also able to gauge the impact of their techniques by calculating how many queries (i.e., analysis operations, workflows) in the benchmark are made faster using the new techniques.

Furthermore, a designer should be able to easily identify explicit strengths and weaknesses in their visualization system within the context of the Visualization Performance Benchmark (e.g., which queries run faster, and which run slower compared to other systems). After running the benchmark, the performance results should also be straightforward to interpret, which necessitates providing thorough documentation for all queries in the benchmark, including for each query: 1) the user interface interactions that are covered by this particular query (and how this mapping is developed); and 2) the optimization areas that are covered by this query.

### 4.2 Ease of Customization

Each visualization system is designed to support a unique set of dataset types and user interface features. As such, the Visualization Performance Benchmark should be configurable, to suit different system and architecture needs. For example, visual data systems (and modules) range from operating entirely on the client (Voyager [32], Nanocubes [21], Hashedcubes [24]), to operating on both client and server machines (imMens [22], ForeCache [6, 5], SeeDB [30], Voyager2 [33]). There are clear differences in available storage and computing resources between these different environments, and the size of the benchmark dataset should be scaled accordingly.

Other data distribution factors also play a key role in system evaluation, such as dataset skew and correlations between data columns. For example, in the context of DBMSs, extremely skewed data can cause a huge slow-down in performance if not carefully distributed across multiple machines [28]. Several visualization systems, including imMens [22], DICE [19], and Profiler [20], are also evaluated under different data distribution conditions. As such, parameterizing these dataset conditions is a critical use case for the Visualization Performance Benchmark.

### 4.3 Realistic Scenarios

The most important feature of the Visualization Performance Benchmark will be its ability to simulate a broad range of real-world visual analytics use cases. The closer the approximation, the more one can rely on the results from the benchmark as being indicative of the performance observed when real users interact with the system. These use cases should also encompass a reasonable set of interactions within a user interface, dictated by the set of user interactions featured in existing large-scale visualization systems. While a complete set of interactions will be a future point of research, as a starting point, we propose that the benchmark should include the following set of common data interactions: panning, zooming, filtering/selections, changing of axes (i.e., pivoting), and brushing and linking.

A final consideration in developing the benchmark lies in the structure of an "analysis session". Specifically, users' analyses tend to occur in concentrated bursts, resulting in chains of queries (either written by the user, or triggered by the visualization system) that are often related to one another. User interaction logs from visualization systems are known to be a rich source for understanding and learning behavioral patterns [7, 17], and these patterns can be utilized in performance optimizations [6, 5]. As such, a data-visualization management benchmark will be more powerful if the queries created for the benchmark also follow a session-based structure (i.e., if it incorporates some consistent representation of actual user behavior).

## 5 Proposed Benchmark Implementation

Given our design considerations, we now describe a proposal for developing the Visualization Performance Benchmark. In the remainder of this section, we briefly touch on all aspects of the benchmark (dataset, queries and metrics), but focus the discussion on the most challenging aspect to developing the benchmark: creating realistic queries.

### 5.1 Dataset and Evaluation Metrics

Here, we briefly discuss our plans for the other components of the benchmark. We will leverage data generation techniques from existing benchmarks to create a fully code-generated dataset (see Section 3.3 for more details). To ensure that the data is supported by realistic scenarios, we can use data from the previous VAST Challenges as the initial input to our data generator. To establish clear evaluation measures, we will incorporate all standard measures utilized in existing evaluation techniques, as well as under-utilized measures such as cold-start time (see Section 2 for more details).

### 5.2 Creating the Queries

To generate queries, we propose the following implementation steps. Given a specific VAST Challenge dataset from the Visual Analytics Benchmark Repository (e.g., VAST Challenge 2012 [2]), we:

1. Select a subset of interactions to be evaluated by the benchmark (see Section 4.3 for an initial set of supported interactions).
2. Collect existing analysis workflows for each dataset (i.e., collect ground truth), ensuring that the workflows are consistent with the supported interactions from step one.
3. Given a list of consecutive analysis steps in the ground truth data (e.g., interaction logs, or the VAST Challenge submissions), translate the analysis steps to queries.

Because the VAST Challenge datasets come with existing ground truth data, we have access to workflows created by real users. Furthermore, we can extract a new workflow for each submission made to the competition. By incorporating multiple submissions as separate workflows, we have the opportunity to showcase a diverse set of high-level analysis strategies (and corresponding interaction patterns) within the benchmark. Each of these workflows represents a unique query set that emphasizes different interactions (e.g., relying more on filters and brushing and linking, less on panning and zooming), enabling benchmark users to more easily customize the benchmark to match the interactions supported by different visual data systems.

Thus, benchmark users have three general options for evaluating their systems using the benchmark: 1) evaluate across all queries and all workflows, 2) choose specific workflows and evaluate only the queries in these workflows, or 3) ignore the session-based notion of the benchmark and only choose queries that support specific interactions (e.g., only evaluate against the filtering queries). In this way, general-purpose systems (i.e., systems that support all interactions from step one) can be compared against the entire benchmark, and specialized systems can be compared against the subset of the benchmark that they support. Note that when evaluating two specialized systems that do not share complete overlap in supported interactions, only the workflows representing the *intersection* of supported interactions can be directly compared. Similarly, performance results from one workflow cannot be directly compared with the results from another, since each workflow emphasizes different sets of interactions.

## 6 Conclusion

In this paper, we discuss the need for a new visualization performance benchmark. Performance benchmarks have been developed by both the database and visualization communities, but for completely different purposes, and as a result these benchmarks fail to address the key factors involved in evaluating visual data systems. We describe a core set of 3 design goals in developing a new benchmark: ease of customization, to support different system designs and architectures; ease of interpretation of benchmark results, to ensure fair and transparent comparisons across visualization systems; and realistic scenarios, to ensure that the benchmark reflects real-world use cases in visual analytics. We then propose methods to develop the dataset, queries, and evaluation metrics for a new data-visualization management benchmark, guided by past benchmarks and our design considerations.

## REFERENCES


[1] Public streams – Twitter Developers. https://dev.twitter.com/streaming/public.

[2] VAST Challenge 2012. http://www.vacommunity.org/VAST+Challenge+2012.

[3] LAADS Web – Search for Data Products. https://ladsweb.nascom.nasa.gov/search/?si=Terra%20MODIS&si=Aqua%20MODIS, Nov. 2016.

[4] IMDb. http://www.imdb.com/interfaces, Aug. 2017.

[5] L. Battle. *Behavior-Driven Optimization Techniques for Scalable Data Exploration*. PhD thesis, Massachusetts Institute of Technology, 2017.

[6] L. Battle, R. Chang, and M. Stonebraker. Dynamic Prefetching of Data Tiles for Interactive Visualization. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, pages 1363–1375, New York, NY, USA, 2016. ACM.

[7] E. T. Brown, A. Ottley, H. Zhao, Q. Lin, R. Souvenir, A. Endert, and R. Chang. Finding waldo: Learning about users from their interactions. *IEEE Transactions on visualization and computer graphics*, 20(12):1663–1672, 2014.

[8] U. C. Bureau. PUMS Data. https://www.census.gov/programs-surveys/acs/data/pums.html, 2017.

[9] U. Cetintemel, M. Cherniack, J. DeBrabant, Y. Diao, K. Dimitriadou, A. Kalinin, and O. Papaemmanouil. Query Steering for Interactive Data Exploration. In *6th Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, USA, Jan. 2013.

[10] S.-M. Chan, L. Xiao, J. Gerth, and P. Hanrahan. Maintaining interactivity while exploring massive time series. In *IEEE Symposium on Visual Analytics Science and Technology, 2008. VAST '08*, pages 59–66, Oct. 2008.

[11] S. Chen, A. Ailamaki, M. Athanassoulis, P. B. Gibbons, R. Johnson, I. Pandis, and R. Stoica. Tpc-e vs. tpc-c: characterizing the new tpc-e benchmark via an i/o comparison study. *ACM SIGMOD Record*, 39(3):5–10, 2011.

[12] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears. Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 143–154, New York, NY, USA, 2010. ACM.

[13] T. P. P. Council. Tpc-h benchmark specification. *Published at http://www.tpc.org/tpch/default.asp*, 21:592–603, 2008.

[14] T. P. P. Council. Tpc-c benchmark specification. *Published at http://www.tpc.org/tpcc/default.asp*, 2010.

[15] A. Crotty, A. Galakatos, E. Zgraggen, C. Binnig, and T. Kraska. The Case for Interactive Data Exploration Accelerators (IDEAs). HILDA '16, pages 11:1–11:6, New York, NY, USA, 2016. ACM.

[16] D. Fisher, I. Popov, S. Drucker, and m. schraefel. Trust Me, I'M Partially Right: Incremental Visualization Lets Analysts Explore Large Datasets Faster. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1673–1682, New York, NY, USA, 2012. ACM.

[17] H. Guo, S. R. Gomez, C. Ziemkiewicz, and D. H. Laidlaw. A Case Study Using Visualization Interaction Logs and Insight Metrics to Understand How Analysts Arrive at Insights. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):51–60, Jan. 2016.

[18] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online Aggregation. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, SIGMOD '97, pages 171–182, New York, NY, USA, 1997. ACM.

[19] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi. Distributed and interactive cube exploration. In *2014 IEEE 30th International Conference on Data Engineering (ICDE)*, pages 472–483, Mar. 2014.

[20] S. Kandel, R. Parikh, A. Paepcke, J. M. Hellerstein, and J. Heer. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 547–554, New York, NY, USA, 2012. ACM.

[21] L. Lins, J. T. Klosowski, and C. Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.

[22] Z. Liu, B. Jiang, and J. Heer. imMens: Real-time Visual Querying of Big Data. In *Proceedings of the 15th Eurographics Conference on Visualization*, EuroVis '13, pages 421–430, Aire-la-Ville, Switzerland, Switzerland, 2013. Eurographics Association.

[23] R. Nambiar, N. Wakou, F. Carman, and M. Majdalany. Transaction Processing Performance Council (TPC): State of the Council 2010. In *Performance Evaluation, Measurement and Characterization of Complex Systems*, Lecture Notes in Computer Science, pages 1–9. Springer, Berlin, Heidelberg, Sept. 2010.

[24] C. A. Pahins, S. A. Stephens, C. Scheidegger, and J. L. Comba. Hashedcubes: Simple, Low Memory, Real-Time Visual Exploration of Big Data. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2016.

[25] Y. Park, M. Cafarella, and B. Mozafari. Visualization-aware sampling for very large databases. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 755–766, May 2016.

[26] C. Plaisant, J. D. Fekete, and G. Grinstein. Promoting Insight-Based Evaluation of Visualizations: From Contest to Benchmark Repository. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):120–134, Jan. 2008.

[27] P. Sethuraman and H. Reza Taheri. *TPC-V: A Benchmark for Evaluating the Performance of Database Applications in Virtual Environments*, pages 121–135. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[28] R. Taft, E. Mansour, M. Serafini, J. Duggan, A. J. Elmore, A. Aboulnaga, A. Pavlo, and M. Stonebraker. E-store: Fine-grained elastic partitioning for distributed transaction processing systems. *Proc. VLDB Endow.*, 8(3):245–256, Nov. 2014.

[29] R. Taft, M. Vartak, N. R. Satish, N. Sundaram, S. Madden, and M. Stonebraker. GenBase: A Complex Analytics Genomics Benchmark. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 177–188, New York, NY, USA, 2014. ACM.

[30] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis. SeeDB: Efficient Data-driven Visualization Recommendations to Support Visual Analytics. *Proc. VLDB Endow.*, 8(13):2182–2193, Sept. 2015.

[31] M. A. Whiting, W. Cowley, J. Haack, D. Love, S. Tratz, C. Varley, and K. Wiessner. Threat stream data generator: Creating the known unknowns for test and evaluation of visual analytics tools. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–3, New York, NY, USA, 2006. ACM.

[32] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2015.

[33] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 2648–2659, New York, NY, USA, 2017. ACM.