Towards Democratizing Relational Data Visualization

Nan Tang
Qatar Computing Research Institute, HBKU
ntang@hbku.edu.qa

Eugene Wu
Columbia University
ewu@cs.columbia.edu

Guoliang Li
Tsinghua University
liguoliang@tsinghua.edu.cn

ABSTRACT

The problem of data visualization is to transform data into a visual context such that people can easily understand the significance of data. Nowadays, data visualization becomes especially important, because it is the *de facto* standard for modern business intelligence and successful data science. This tutorial will cover three specific topics: ***visualization languages*** define how the users can interact with various visualization systems; ***efficient data visualization*** processes the data and produces visualizations based on well-specified user queries; ***smart data visualization*** recommends data visualizations based on underspecified user queries. In this tutorial, we will go logically through these prior art, paying particular attentions on problems that may attract the interest from the database community.

CCS CONCEPTS

• **Human-centered computing** → **Visualization**; **Visualization techniques**;

KEYWORDS

relational data, data visualization

ACM Reference Format:
Nan Tang, Eugene Wu, and Guoliang Li. 2019. Towards Democratizing Relational Data Visualization. In *2019 International Conference on Management of Data (SIGMOD '19), June 30–July 5, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3299869.3314029

1 INTRODUCTION

Data visualization is a powerful means to present compelling stories of data to humans who are more visually oriented.

SIGMOD '19, June 30–July 5, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5643-5/19/06…$15.00
https://doi.org/10.1145/3299869.3314029

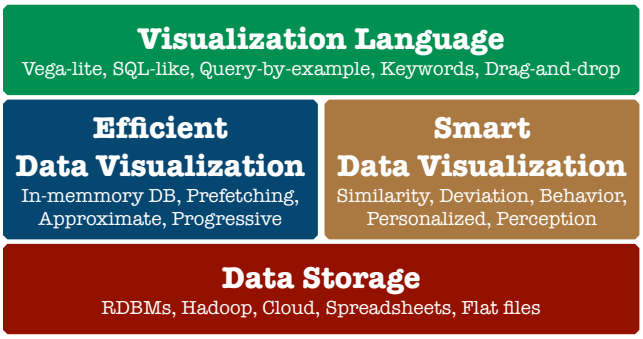

Figure 1: The Stack of Data Visualization

Nowadays, all organizations have more data than ever at their disposal. Consequently, more and more organizations use data and advanced analytics to inform operational decisions. Data visualization is a natural means to both provide an overview of massive datasets, and to help users interpret the results of data analytics. The success of leading vendors in data visualization, such as Tableau, Qlik and Power BI, has revolutionized the way that even non-technical people can understand and take action through data.

The considerable interest and efforts from industry and academia have gone a long way, however, the journey of *democratizing relational data visualization* – such that anyone can easily generate various visualizations to understand, analyze, and present massive volumes of data – is still in its early stages. The purpose of this tutorial is to provide an overview of where we are from the DB perspective.

Let us provide a small amount of formalism. *Given a relational database* $\mathbf{D}$*, a user specification* $\mathbf{S}$*, a* ***data visualization*** *can be thought of as a function* $\mathbf{F}(\cdot,\cdot)$ *such that* $\mathbf{F}(\mathbf{D},\mathbf{S})$ *computes a set* $\mathbf{V}$ *of visualizations.*

Based on the above definition, Figure 1 presents a simple architectural stack for relational data visualization.

Data storage describes the data storage layer containing the relational data $\mathbf{D}$. In practice, it may be stored in a DBMS, cloud storage, or flat files. It may be very large: it may contain thousands of tables (or databases), tables may be wide (*e.g.,* several hundred attributes), and the volumes may be large (*e.g.,* petabytes). Most commercial visualization systems can access and visualize across multiple sources, such as Tableau,

Qlik, Amazon QuickSight. In contrast, most research prototypes mainly read data from DBMSs, such as SeeDB [70], DeepEye [54], Polaris [68], zenvisage [66], and Voyager [74].

We mention storage for completeness, however leave the technical details out of this tutorial, because it is the focus of entire subfields of the data management community. We will focus on the three visualization-specific components below:

Visualization languages are the mechanism that users express their visualization goals. A language specification **S** must be precise enough that the visualization system can process it efficiently, yet easy to use for developers or end-users. Language specification vary from procedural to declarative, and can be expressed textually or interactively (e.g., drag-and-drop operations).

Efficient data visualization is to quickly generate visualizations **V** when the specification **S** is well-specified, *i.e.,* there is no ambiguity of what $\mathbf{F}(\mathbf{D}, \mathbf{S})$ will produce. A primary goal is responsiveness and scalability (*i.e.,* $\mathbf{F}(\mathbf{D}, \mathbf{S})$) can be evaluated in real time); to achieve this goal, systems may relax the result semantics to return approximate and progressive results if $\mathbf{F}(\mathbf{D}, \mathbf{S})$) cannot be evaluated in real time.

Smart data visualization is when the user underspecifies **S**. The system must first "smartly" complete the specification in a way that anticipates the user's intentions, before it can render a visualization. This is analogous to keyword search, in that the search terms are underspecified. Thus different search engines (e.g., Google) will output different results. Smart data visualization could be fully automatic, reference visualization based, user behavior based, personalized, etc.

Open Problems. The last part of this tutorial is dedicated to open problems, which include big data challenges due to massive datasets that still stymie the goal of real time interactions, the effects of (dirty) data whose resulting visualizations may mislead the users with false trends due to data errors, deep data visualization that leverages deep learning technologies towards smarter data visualization, a similarly large data visualization benchmark for smart data visualization recommendations. and other related problems.

2 TARGET AUDIENCE AND LENGTH

The primary audience is researchers, practitioners and students that are interested in data visualization, or using data visualization to solve problems. The tutorial will be self-contained, and we will include a broad introduction and motivating examples for non-specialists to follow.

Moreover, this tutorial should also be of interest to the data mining and machine learning communities, due both to the importance of finding compelling stories through data visualization for various data analytical tasks, and to its connections with smart data visualizations.

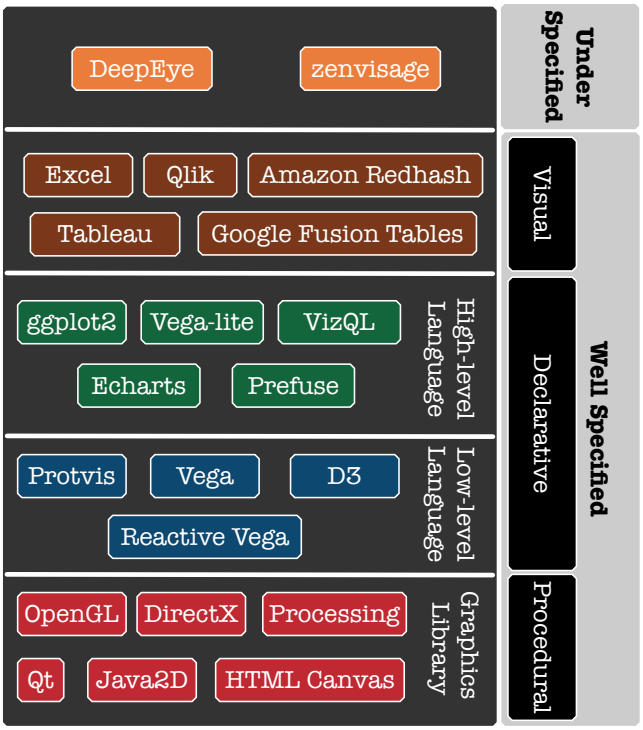

Figure 2: Data Visualization Languages

The intended length of this tutorial is 3 hours.

3 TUTORIAL OUTLINE

We will start with a brief overview of this tutorial, to give the audience a clear outline and talk goals. We will then present motivating examples from emerging applications to illustrate the importance of data visualization in multiple domains and tasks.

3.1 Data Visualization Languages

In this tutorial, we use the term "data visualization languages" to broadly mean any method that the users can specify *what* visualizations do they want to data visualization systems.

On one side, users can be either experts or non-experts, and may come from different backgrounds, such as mathematician or engineer. On the other side, data visualization systems are implemented based on different design purposes, personal expertise, or any other reason. Consequently, there are many data visualization languages. Figure 2 gives an overview about our classification of visualization languages.

Well-specified Languages. This class of languages provide unambiguous semantics to a given query **S**. We define three classes of well-specified languages.

(1) *Procedural language* requires the specification of a series of well-structured steps and procedures to compose a visualization. They are typically *Graphics Libraries*, which

specify the most basic graphical primitives (*e.g.,* vector in vector graphic, pixel in bitmap) in graphics, such as OpenGL, DirectX, Qt, Java2D, HTML Canvas, Raphaël [2], Processing (https://processing.org), Piccolo [10] and others.

(2) *Declarative language* requires the specification **S** of only *what* the users want; *how* to execute them is engine dependent (*i.e.,* the implementation of function **F**(**D**, **S**)). We further categorize them into two classes.

– *Low-level languages* [3, 12, 42, 62, 72] abstract graphical elements as visualization primitives. The visualization primitives include marks (*e.g.,* bar in histogram, line in line chart), axes, legends, scales (the mapping of data to visual attributes, usually as a function), transformations (*e.g.,* grouping and binning), signals (used in interaction), etc. We will mainly talk about D3 [42, 72], Vega [3] and Reactive Vega [62].

– *High-level languages* [25, 26, 35, 61, 72, 73, 75, 79] further abstract the details of low-level visualization construction. They provide concise specification interfaces that are easier for new users to learn and use. Users omit low-level details, which a compiler fills in with sensible defaults. In this context, we will focus on key ideas from ggplot2 [72] and Vega-Lite [61], and discuss their connections with SQL.

(3) *Visual languages* follow the "direct manipulation principle" [64], that lets end-users interact with a visual interface to specify visual operations and specifications through a mouse or touch screen. For instance, Polaris [68] lets users directly drag attributes from a list onto the X or Y axes to specify aggregation-based analyses. Interactions may also specify transformations, chart types, predicates, and more.

Visual languages are widely, and increasingly, used in commercial visualization systems such as Microsoft Excel, Apple Numbers, Amazon Redash, Qlik, Tableau (evolved from Polaris [68]), Spotfire [5], Google Spreadsheets, Google Fusion Tables [22] and more.

Underspecified Languages. Generally speaking, underspecified languages contain "gaps" in the specification **S**, and it is the task of the visualization system to interpret the underspecified input (*i.e.,* the implementation of the function **F**(**D**, **S**)), in a variety of ways. Users may leave "hints" in the gaps so the system can better interpret.

The first type of hint is "reference-based", where the users provide a reference visualization as a seed. zenvisage [65, 66] supports queries which return similar or dissimilar visualizations (*e.g.,* similar trends in line charts) with a user provided reference visualization. Similarly, Draco [44], Voyager2 [74] and others take as input incomplete declarative visualization specifications as reference.

The second type of hint is "keyword-based", similar to keyword search. Systems such as VizDeck [48] and DeepEye [39](http://deepeye.tech) take as input keywords and return recommended visualizations. For example, the user of the latter system may input "*show me line charts about electricity*", and the system will recommend line charts which also contain the column "electricity".

3.2 Efficient Data Visualization

When the visualization specification is well-defined, a key system goal is to generate the visualization quickly. We will present four important classes of techniques.

(1) Exact data visualization. The class describes systems that compute visualizations exactly. We further refine these based on their system design:

– *Apply existing systems and technique.* Many visualization systems (*e.g.,* DeepEye [38, 54], Polaris [68], SeeDB [70, 71], Vizdom [14], M4 [30]) perform computation via issuing SQL queries to a general DBMS. Others [20, 41, 49, 57] use hardware (such as multi-core and GPU), parallel processing [37, 49, 70, 71], or existing types of data structures to accelerate visualization.

– *Modify DBMS designs.* Industry and academia have modified DBMS functionality or system designs to optimize them for data visualization. These include industry systems such as Hyper [1, 46, 47] which is an efficient main-memory and hybrid OLTP and OLAP DBMS that is now customized for Tableau's data engine. In academia, systems such as the Data Visualization Management System [51–53, 75, 80] explore how relational query languages can be used to express interactive visualizations [80], and how relational DBMS designs can be adapted and extended – for instance with fast lineage support [51–53] – to speed up interactive visualization.

– *Predictive data visualization.* Users interact with visualizations throughout their exploration and analysis process. These interactions are typically informed from current and previous views—by changing the parameters of current visualization, adding a predicate, or zooming to see details or overviews. Thus many visualization systems [9, 11, 13, 18, 28, 31, 37, 69] seek to predict and speculatively execute/prefetch future visualizations that the user may request.

(2) Approximate data visualization. In some cases, it may be too difficult to compute exact visualizations quickly enough. In these cases, approximation is a pragmatic and effective mechanism to trade-off responsiveness and accuracy, and many systems [4, 6, 16, 17, 27, 34, 43, 50, 56] speed up data processing by leveraging approximate query processing (AQP) techniques. To inform how error bounds may be set, there has been recent work to quantify perceptual inaccuracies [58, 76], or *perceptual function*, in ways that may be employed by such approximation approaches.

(3) Progressive data visualization. A related, but distinct concept, is to quickly provide an overview (or approximation)

of the visualization, and then gradually refine the details over time [11, 16, 19, 28, 36, 43, 56, 67]. Users have the flexibility to choose to either wait for more interesting details, or be sufficiently satisfied to make a decision (e.g., perform another interaction). It is important to understand how progressive results affect the user's exploration process [81].

(4) Data reduction. When rendering visualizations, many systems map each data point to a visual element, which may result in a cluttered visualization that overloads the user. Data reduction methods help address this issue by summarizing the dataset to help users better identify patterns. We will review common data reduction techniques – including filtering and sampling [16, 34, 43, 60], aggregation (clustering) [24], and model fitting [21] – and when each is effective for particular tasks.

3.3 Smart Data Visualization

Precisely specifying **S** is hard, even for experts, especially in the common situation that the users may not even know what they precisely want. Not surprisingly, several systems [39, 55, 65, 66] allow users to provide an underspecified **S**, and smartly computes visualizations **V**, which is also referred to as *data visualization recommendation*. The practical need for such systems is that data visualization is typically used for data exploration – users try to find the compelling but unknown stories instead of just depicting the stories.

Apparently, the hardest part of smart data visualization is to quantify the "goodness" of visualizations **V** *w.r.t.* **D** and **S**, *i.e.,* to "guess" what the users want. Intuitively, the "one size does not fit all" principle applies here. Consequently, which visualizations to recommend have been approached from different angles, such as similarity-based methods [65, 66] that recommend visualizations which have the similar trends, patterns or statistical information with the reference visualization; deviation-based approaches [33, 65, 70, 71] that capture interesting visualizations as outliers; behavior-based solutions that infer users' intent by his/her present behavior [23], or a sequence of actions during interactions; personalized visualizations that recommend visualizations by leveraging historical data [48]; and perception-based methods that model the perceptual effectiveness by predefined rules [40, 54, 63, 74] or using machine learning models [15, 38, 44].

Furthermore, even if **S** is well-specified, recommending more interesting visualizations, similar to or different from **V**, to the users could still have plenty of rewarding reasons.

3.4 Conclusions and Open Problems

Although data visualization has been extensively studied, there are still many opportunities and challenges towards the goal of democratizing data visualization.

Interactivity under massive data volumes. Many existing approaches towards interactivity may be reduced to processing a single visualization faster, on more data. However, there is still tremendous opportunity to explicitly model and address interactivity directly – visualization specifications triggered by interactions are highly correlated. Modeling interactions as *sessions* of similar queries [78] can enable explicit state sharing across queries.

Dirty data. Real-life data is typically dirty and visualizing dirty data may mislead users. For example, a data that is integrated from multiple sources may contain many duplicates. Carefully cleaning every dataset before visualizing them is simply too expensive in practice. Hence, visualization-driven data cleaning that focuses on the user's specific analysis is in high demand, and early approaches such as Scorpion [77], Profiler [33], and others [32, 45] push towards this goal.

Deep learning for smarter data visualization. Deep learning based techniques have revolutionized many domain, such as image recognition, natural language understand, automatic car, and many others. An emerging topic is the use of deep learning to better recommend data visualizations [54, 59], in an analagous way as used in modern search engines.

Data Visualization Benchmarks. Like ImageNet or the classic TPC benchmarks, it is important to develop benchmarks for performance and recommendation. The benchmarks should be faithful to the visual analysis tasks, provide reusable traces and data, and in the case of recommendation, have high coverage and quality of its labels. There is an emerging focus on developing benchmarks for performance measures [7, 8, 29]. There is potential for a similarly large data visualization benchmark for smart data visualization recommendations.

4 ACKNOWLEDGEMENTS

We acknowledge 973 Program of China (2015CB358700), NSF of China (61632016, 61521002, 61661166012), NSF grants 1527765 and 1564049, Huawei and TAL education.

5 BIOGRAPHY

Nan Tang is a Senior Scientist at Qatar Computing Research Institute, HBKU, Qatar. He has received the best paper award of VLDB 2010, and several papers have been invited to VLDBJ and TKDE as the best papers series. Dr Nan's main research interests are data preparation and data visualization.

Eugene Wu is an assistant professor at Columbia University, New York, USA. Eugene's primary research interests focus on the systems and algorithmic challenges at the intersection of databases and interactive visual interfaces, and more broadly, for the future of human-data-interaction. He has received

the 2018 VLDB 10-year test of time award, best of conference citations at ICDE and VLDB, SIGMOD 2016 best demo award, and faculty awards from Google and Amazon. He is funded by NSF 1527765, NSF 1564049.

Guoliang Li is a professor at Tsinghua University, China. Guoliang's main research interests include data integration and cleaning, data visualization, and human-in-the-loop data management. He got VLDB 2017 early research contribution award, TCDE 2014 early career award, CIKM 2017 best paper award. His ICDE 2018 and KDD 2018 papers have been invited to TKDE and TKDD as the best papers series.

REFERENCES

[1] [n. d.]. HyPer: A Hybrid OLTP&OLAP High Performance DBMS. https://hyper-db.de.

[2] [n. d.]. Raphaël: JAVASCRIPT LIBRARY. http://raphaeljs.com.

[3] [n. d.]. Vega: A Visualization Grammar. https://vega.github.io/vega/.

[4] Sameer Agarwal, Barzan Mozafari, Aurojit Panda, Henry Milner, Samuel Madden, and Ion Stoica. 2013. BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data. In *ACM*. http://doi.acm.org/10.1145/2465351.2465355

[5] Christopher Ahlberg. 1996. Spotfire: an information exploration environment. In *SIGMOD Record*. ACM.

[6] Daniel Alabi and Eugene Wu. 2016. PFunk-H: approximate query processing using perceptual models. In *HILDA*. http://doi.acm.org/10.1145/2939502.2939512

[7] Leilani Battle, Marco Angelini, Carsten Binnig, Tiziana Catarci, Philipp Eichmann, Jean-Daniel Fekete, Giuseppe Santucci, Michael Sedlmair, and Wesley Willett. 2018. Evaluating Visual Data Analysis Systems: A Discussion Report. In *HILDA*.

[8] Leilani Battle, Remco Chang, Jeffrey Heer, and Mike Stonebraker. 2017. Position statement: The case for a visualization performance benchmark. In *DSIA Workshop*.

[9] Leilani Battle, Remco Chang, and Michael Stonebraker. 2015. Dynamic Prefetching of Data Tiles for Interactive Visualization. In *SIGMOD*.

[10] Benjamin B. Bederson, Jesse Grosjean, and Jon Meyer. 2004. Toolkit Design for Interactive Structured Graphics. In *IEEE TSE*.

[11] Nikos Bikakis, George Papastefanatos, Melina Skourla, and Timos Sellis. 2016. A Hierarchical Aggregation Framework for Efficient Multilevel Visual Exploration and Analysis. In *Computer Science*.

[12] Michael Bostock and Jeffrey Heer. 2009. Protovis: A Graphical Toolkit for Visualization. In *IEEE Transactions on Visualization & Computer Graphics*.

[13] Sye Min Chan, Ling Xiao, J Gerth, and P Hanrahan. 2008. Maintaining interactivity while exploring massive time series.. In *IEEE*.

[14] Andrew Crotty, Alex Galakatos, Emanuel Zgraggen, Carsten Binnig, and Tim Kraska. 2015. Vizdom: Interactive Analytics through Pen and Touch. In *PVLDB*.

[15] Victor Dibia and Çağatay Demiralp. 2018. Data2Vis: Automatic Generation of Data Visualizations Using Sequence-to-Sequence Recurrent Neural Networks. In *arXiv*.

[16] Bolin Ding, Silu Huang, Surajit Chaudhuri, Kaushik Chakrabarti, and Chi Wang. 2016. Sample + Seek: Approximating Aggregates with Distribution Precision Guarantee. In *SIGMOD*. http://doi.acm.org/10.1145/2882903.2915249

[17] Bolin Ding, Silu Huang, Surajit Chaudhuri, Kaushik Chakrabarti, and Chi Wang. 2016. Sample+ seek: Approximating aggregates with distribution precision guarantee. In *SIGMOD*.

[18] Punit R. Doshi, Elke A. Rundensteiner, and Matthew O. Ward. 2003. Prefetching for Visual Data Exploration. In *Masters Theses*.

[19] Niklas Elmqvist and Jean Daniel Fekete. 2010. Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines. In *IEEE Transactions on Visualization & Computer Graphics*.

[20] J Fekete and C Plaisant. 2003. Interactive information visualization of a million items. In *The Craft of Information Visualization*.

[21] Erich Fuchs, Thiemo Gruber, Jiri Nitschke, and Bernhard Sick. 2010. Online Segmentation of Time Series Based on Polynomial Least-Squares Approximations. In *IEEE Transactions on Pattern Analysis & Machine Intelligence*.

[22] Hector Gonzalez, Alon Y Halevy, Christian S Jensen, Anno Langen, Jayant Madhavan, Rebecca Shapley, Warren Shen, and Jonathan Goldberg-Kidon. 2010. Google fusion tables: web-centered data management and collaboration. In *SIGMOD*.

[23] David Gotz and Zhen Wen. 2009. Behavior-driven visualization recommendation. In *IUI*.

[24] Gregor Hackenbroich, Gregor Hackenbroich, Gregor Hackenbroich, and Volker Markl. 2014. Faster visual analytics through pixel-perfect aggregation. In *PVLDB*.

[25] Pat Hanrahan. 2006. VizQL:a language for query, analysis and visualization. In *SIGMOD*.

[26] Jeffrey Heer, Stuart K Card, and James A Landay. 2005. Prefuse: a toolkit for interactive information visualization. In *CHI*.

[27] Joseph M Hellerstein, Ron Avnur, Andy Chou, Christian Hidber, Chris Olston, Vijayshankar Raman, Tali Roth, and Peter J Haas. 1999. Interactive data analysis: The control project. In *Computer*. IEEE.

[28] Prasanth Jayachandran, Karthik Tunga, Niranjan Kamat, and Arnab Nandi. 2014. Combining user interaction, speculative query execution and sampling in the DICE system. In *PVLDB*.

[29] Lilong Jiang, Protiva Rahman, and Arnab Nandi. 2018. Evaluating Interactive Data Systems: Workloads, Metrics, and Guidelines. In *SIGMOD*.

[30] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. 2014. M4: a visualization-oriented time series data aggregation. In *PVLDB*.

[31] Alexander Kalinin, Ugur Cetintemel, and Stan Zdonik. 2014. Interactive data exploration using semantic windows. In *SIGMOD*.

[32] Sean Kandel, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. 2011. Wrangler: interactive visual specification of data transformation scripts. In *CHI*.

[33] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. 2012. Profiler: integrated statistical analysis and visualization for data quality assessment. In *AVI*.

[34] Albert Kim, Eric Blais, Aditya G. Parameswaran, Piotr Indyk, Samuel Madden, and Ronitt Rubinfeld. 2015. Rapid Sampling for Visualizations with Ordering Guarantees. In *PVLDB*.

[35] Deqing Li, Honghui Mei, Yi Shen, Shuang Su, Wenli Zhang, Junting Wang, Ming Zu, and Wei Chen. 2018. ECharts: A declarative framework for rapid construction of web-based visualization. In *Visual Informatics*.

[36] Lauro Lins, James T. Klosowski, and Carlos Scheidegger. 2013. Nanocubes for Real-Time Exploration of Spatiotemporal Datasets. In *IEEE Transactions on Visualization & Computer Graphics*.

[37] Zhicheng Liu, Biye Jiang, and Jeffrey Heer. 2013. imMens : real-time visual querying of big data. In *EuroVIS*.

[38] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. 2018. DeepEye: Towards Automatic Data Visualization. In *ICDE*. http://doi.ieeecomputersociety.org/10.1109/ICDE.2018.00019

[39] Yuyu Luo, Xuedi Qin, Nan Tang, Guoliang Li, and Xinran Wang. 2018. DeepEye: Creating Good Data Visualizations by Keyword Search. In *SIGMOD*. http://doi.acm.org/10.1145/3183713.3193545

[40] Jock Mackinlay, Pat Hanrahan, and Chris Stolte. 2007. Show me: Automatic presentation for visual analysis. In *TVCG*. IEEE.

[41] Bryan McDonnel and Niklas Elmqvist. 2009. Towards utilizing gpus in information visualization: A model and implementation of image-space operations. In *TVCG*. IEEE.

[42] Bostock Michael, Ogievetsky Vadim, and Heer Jeffrey. 2011. D3: Data-Driven Documents. In *IEEE Transactions on Visualization & Computer Graphics*.

[43] Dominik Moritz, Danyel Fisher, Bolin Ding, and Chi Wang. 2017. Trust, but Verify: Optimistic Visualizations of Approximate Queries for Exploring Big Data. In *CHI*. http://doi.acm.org/10.1145/3025453.3025456

[44] Dominik Moritz, Chenglong Wang, Greg L Nelson, Halden Lin, Adam M Smith, Bill Howe, and Jeffrey Heer. 2018. Formalizing Visualization Design Knowledge as Constraints: Actionable and Extensible Models in Draco. In *InfoVIS*.

[45] Kristi Morton, Hannaneh Hajishirzi, Magdalena Balazinska, and Dan Grossman. 2016. View-Driven Deduplication with Active Learning. In *arXiv*.

[46] Thomas Neumann. 2011. *Efficiently compiling efficient query plans for modern hardware*. VLDB Endowment.

[47] Thomas Neumann and Alfons Kemper. 2015. Fast Serializable Multi-Version Concurrency Control for Main-Memory Database Systems. In *SIGMOD*.

[48] Daniel B Perry, Bill Howe, Alicia MF Key, and Cecilia Aragon. 2013. VizDeck: Streamlining exploratory visual analytics of scientific data. In *iConference*.

[49] Harald Piringer, Christian Tominski, Philipp Muigg, and Wolfgang Berger. 2009. A Multi-Threading Architecture to Support Interactive Visual Exploration. In *TVCG*.

[50] Marianne Procopio, Carlos Scheidegger, Eugene Wu, and Remco Chang. 2017. Load-n-Go: Fast Approximate Join Visualizations That Improve Over Time. In *DSIA*.

[51] Fotis Psallidas and Eugene Wu. 2018. Demonstration of Smoke: A Deep Breath of Data-Intensive Lineage Applications. In *SIGMOD (demo)*.

[52] Fotis Psallidas and Eugene Wu. 2018. Provenance in Interactive Visualizations. In *HILDA*.

[53] Fotis Psallidas and Eugene Wu. 2018. Smoke: Fine-grained Lineage at Interactive Speeds. In *VLDB*.

[54] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2018. DeepEye: An automatic big data visualization framework. In *Big Data Mining & Analytics*.

[55] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2018. Deepeye: Visualizing your data by keyword search. In *EDBT Vision*.

[56] Sajjadur Rahman, Maryam Aliakbarpour, Ha Kyung Kong, Eric Blais, Karrie Karahalios, Aditya Parameswaran, Ronitt Rubinfield, Sajjadur Rahman, Maryam Aliakbarpour, and Ha Kyung Kong. 2017. I've seen "enough": incrementally improving visualizations to support rapid decision making. In *PVLDB*.

[57] Oliver Rübel, Kesheng Wu, Hank Childs, Jeremy Meredith, Cameron GR Geddes, Estelle Cormier-Michel, Sean Ahern, Gunther H Weber, Peter Messmer, Hans Hagen, et al. 2008. High performance multivariate visual data exploration for extremely large data. In *SC*.

[58] Gabriel Ryan, Abigail Mosca, Remco Chang, and Eugene Wu. 2018. At a Glance: Approximate Entropy as a Measure of Line Chart Visualization Complexity. In *InfoVIS*.

[59] Bahador Saket, Dominik Moritz, Halden Lin, Victor Dibia, Cagatay Demiralp, and Jeffrey Heer. 2018. Beyond heuristics: Learning visualization design. In *arXiv*.

[60] Anish Das Sarma, Hongrae Lee, Hector Gonzalez, Jayant Madhavan, and Alon Halevy. 2012. Efficient spatial sampling of large geographical tables. In *SIGMOD*.

[61] A Satyanarayan, D Moritz, K Wongsuphasawat, and J Heer. 2016. Vega-Lite: A Grammar of Interactive Graphics. In *IEEE Transactions on Visualization & Computer Graphics*.

[62] Arvind Satyanarayan, Ryan Russell, Jane Hoffswell, and Jeffrey Heer. 2015. Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization. In *IEEE Transactions on Visualization & Computer Graphics*.

[63] Jinwook Seo and Ben Shneiderman. 2005. A rank-by-feature framework for interactive exploration of multidimensional data. In *Information Visualization*.

[64] Ben Shneiderman. 1983. Direct Manipulation: A Step Beyond Programming Languages. In *IEEE Computer*.

[65] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya G. Parameswaran. 2016. Effortless Data Exploration with zenvisage: An Expressive and Interactive Visual Analytics System. In *PVLDB*.

[66] Tarique Siddiqui, John Lee, Albert Kim, Edward Xue, Xiaofo Yu, Sean Zou, Lijin Guo, Changfeng Liu, Chaoran Wang, Karrie Karahalios, and Aditya G. Parameswaran. 2017. Fast-Forwarding to Desired Visualizations with Zenvisage. In *CIDR*.

[67] C. D Stolper, A Perer, and D Gotz. 2014. Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics. In *IEEE Transactions on Visualization & Computer Graphics*.

[68] C. Stolte and P. Hanrahan. 2000. Polaris: a system for query, analysis and visualization of multi-dimensional relational databases. In *Information Visualization*.

[69] Farhan Tauheed, Thomas Heinis, Felix ShÃijrmann, Henry Markram, and Anastasia Ailamaki. 2012. SCOUT: Prefetching for Latent Feature Following Queries. In *PVLDB*.

[70] Manasi Vartak, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2014. SEEDB: automatically generating query visualizations. In *PVLDB*.

[71] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya G. Parameswaran, and Neoklis Polyzotis. 2015. SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics. In *PVLDB*.

[72] Hadley Wickham. 2010. A Layered Grammar of Graphics. In *Journal of Computational & Graphical Statistics*.

[73] Leland Wilkinson. 2005. *The grammar of graphics*. Springer. 673âĂŞ677 pages.

[74] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2017. Voyager 2: Augmenting visual analysis with partial view specifications. In *CHI*.

[75] Eugene Wu, Leilani Battle, and Samuel R. Madden. 2014. The case for data visualization management systems. In *Cancer Letters*.

[76] Eugene Wu, Lilong Jiang, Larry Xu, and Arnab Nandi. 2016. Graphical Perception in Animated Bar Charts. In *arXiv*.

[77] Eugene Wu and Samuel Madden. 2013. Scorpion: Explaining Away Outliers in Aggregate Queries. In *VLDB*.

[78] Eugene Wu and Arnab Nandi. 2015. Towards perception-aware interactive data visualization systems. In *DSIA Workshop*.

[79] Eugene Wu, Fotis Psallidas, Zhengjie Miao, Haoci Zhang, and Laura Rettig. 2017. Combining Design and Performance in a Data Visualization Management System. In *CIDR*.

[80] Eugene Wu, Fotis Psallidas, Zhengjie Miao, Haoci Zhang, Laura Rettig, Yifan Wu, and Thibault Sellam. 2017. Combining Design and Performance in a Data Visualization Management System.. In *CIDR*.

[81] Emanuel Zgraggen, Alex Galakatos, Andrew Crotty, Jean-Daniel Fekete, and Tim Kraska. 2017. How progressive visualizations affect exploratory analysis. In *IEEE Transactions on Visualization & Computer Graphics*. IEEE.