A Declarative Specification for Authoring Metrics Dashboards

Will Epperson, Kanit Wongsuphasawat, Allison Whilden, Fan Du, Justin Talbot

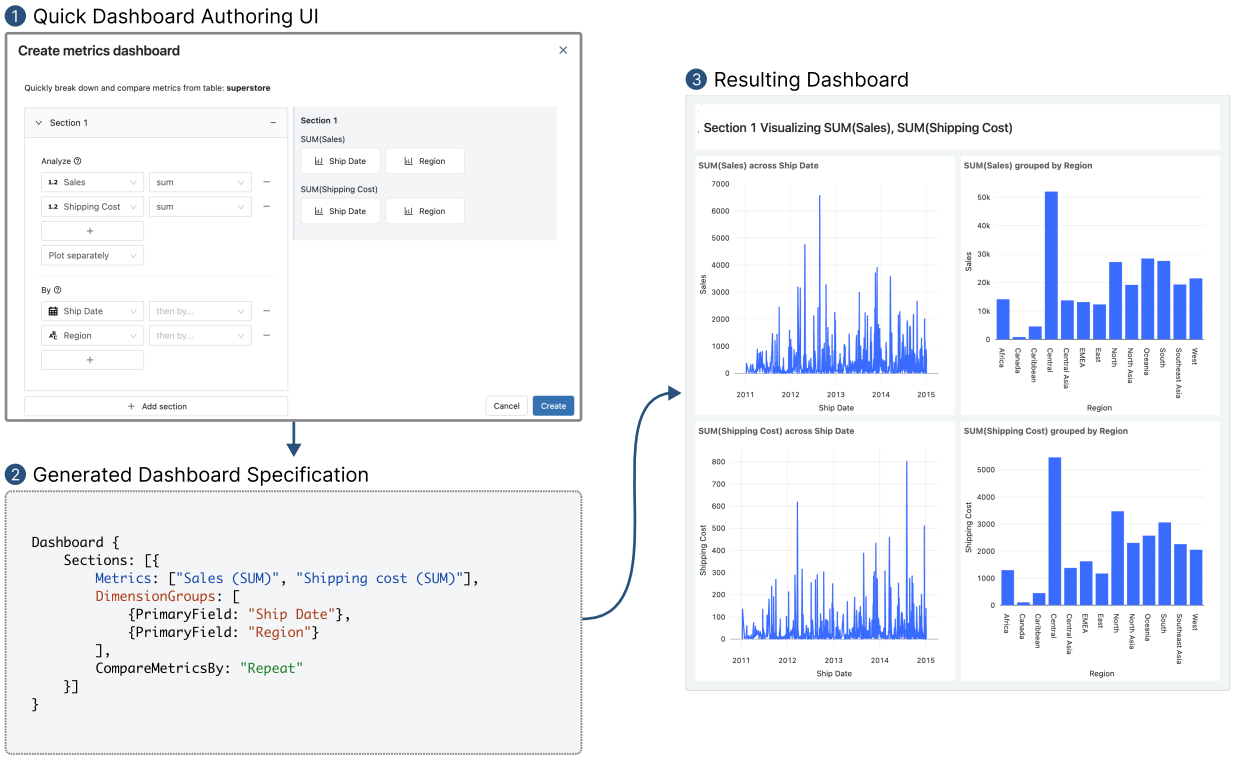

Fig. 1: The *Quick Dashboard* system allows users to rapidly specify dashboards by choosing metrics and dimensions of interest from a data table. **(1)** A user begins by using the *Quick Dashboard* UI to choose metrics and dimensions for each section of their dashboard. In this example, the user has selected two metrics (Sales and Shipping Cost) and two dimensions (Ship Date and Region). **(2)** The UI creates a corresponding dashboard specification for the user's input. **(3)** The system then generates the final dashboard by combining each metric and dimension according to the specification. This dashboard can be used immediately or further customized by changing the chart types or re-arranging components.

Abstract— Despite their ubiquity, authoring dashboards for metrics reporting in modern data analysis tools remains a manual, time-consuming process. Rather than focusing on interesting combinations of their data, users have to spend time creating each chart in a dashboard one by one. This makes dashboard creation slow and tedious. We conducted a review of production metrics dashboards and found that many dashboards contain a common structure: breaking down one or more metrics by different dimensions. In response, we developed a high-level specification for describing dashboards as sections of metrics repeated across the same dimensions and a graphical interface, *Quick Dashboard*, for authoring dashboards based on this specification. We present several usage examples that demonstrate the flexibility of this specification to create various kinds of dashboards and support a data-first approach to dashboard authoring.

Index Terms—Dashboards, visualization recommendation.

- *Will Epperson is with Carnegie Mellon University. E-mail: willepp@cmu.edu.*
- *Kanit Wongsuphasawat is with Databricks. E-mail: kanit.w@databricks.com.*
- *Allison Whilden is with Databricks. E-mail: allison.whilden@databricks.com.*
- *Fan Du is with Databricks. E-mail: fan.du@databricks.com.*
- *Justin Talbot is with Databricks. E-mail: justin.talbot@databricks.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

1 Introduction

Dashboards are used for a variety of purposes, including monitoring important metrics, providing an overview of data, or for communication [9]. In this paper, we focus on using dashboards to monitor and report on metrics of interest. Metrics are quantitative variables such as the total dollar amount of sales or number of successful surgeries. When analyzed across *temporal* or *categorical* dimensions, interesting trends can be found. For example, a dashboard might show metrics like the total number of sales or the ratio of successful to unsuccessful surgeries across dimensions like time and region. Putting together multiple visualizations in a dashboard allows users to see how their metrics change across these different slices.

Current dashboard authoring experiences such as Tableau, Looker, or PowerBI typically focus on the *bottom-up* creation of individual

charts, which are later combined into a dashboard [1, 5, 11]. Whether creating dashboards through code-first or GUI interfaces, the bottom-up approach is slow and repetitive. Each chart must be manually created, saved, and then combined into the final dashboard. Even for a simple dashboard with only a few metrics and dimensions, the time spent on specifying the charts of a dashboard could be better spent on understanding the results and considering meaningful data combinations. Recent research has discussed the steep learning curve for many dashboard tools that require users to have a deep knowledge of visualization design [9]. Like other recent work [6], we view this as an opportunity for simpler interfaces to dashboard creation.

To simplify and accelerate dashboard creation, we observed the fact that dashboards are typically not a combination of random charts, but have an underlying repetitive structure. Specifically, dashboards are often structured as sections of charts where the **same metrics are broken down by the same dimensions**. We used this repetitive structure to design a high-level specification for describing dashboards. To demonstrate the utility of this abstraction, we built a user interface, *Quick Dashboard*, that allows users to quickly author dashboards. Our system allows users to specify *which* parts of their data they are interested in visualizing rather than *how* to visualize each combination of columns. *Quick Dashboard* uses a chart recommender to create each of the charts in the dashboard. After creation, users can then customize each of the charts in the dashboard using a GUI chart editor and change the layout of the widgets. Developers of dashboard authoring tools can use our specification to build experiences similar to that demonstrated in *Quick Dashboard* that focus on data-first rather than chart-first authoring flows. In short, our paper makes the following contributions:

1. A high-level specification for describing dashboards as sections of metrics combined with dimensions. Each section contains the cross-product of all metrics and dimensions.
2. A demonstration of the flexibility and utility of the proposed dashboard specification through a no-code dashboard authoring tool, *Quick Dashboard*, and examples of three real-world dashboards built using the tool.

2 Related Work

We draw from two primary areas of related work: visualization recommendations for single and multi-view charts and dedicated tools for dashboard authoring.

2.1 Visualization Recommendation

Visualization recommendation helps analysts understand their data by automating visual presentation or suggesting interesting parts of the data [16]. At the individual chart level, this aids in faster exploration of visual designs that best communicate the data [4], and when looking at multiple charts can help analysts rapidly explore different aspects and combinations of data [18]. *Quick Dashboard* uses a rule-based chart recommender to generate individual charts that are composed into dashboards based on the data types of input fields, similar to [17]. Since the focus of this paper is on a specification for expressing dashboards, we omit the fine-grained details on how the individual charts are generated however discuss input format for our chart recommender in § 3.4. *Quick Dashboard* can be used with other chart recommendation modules that take similar inputs.

Dashboards are instances of multi-view visualizations, where multiple charts are used with common fields to break down trends [8]. Beginning with Tufte's early work on small multiples [13], recent research has explored guidelines and design principles for crafting effective multi-view visualizations [7, 15]. We incorporate such guidelines into our system, in particular focusing on consistent scales and axes when multiple recommended charts have the same field in a section (constraint C1 from [7]). Future iterations of our system plan on further incorporating the guidance on using consistent colors across all charts in a dashboard from previous work [7].

2.2 Dashboard Authoring Tools

Another line of relevant research focuses on improved tools for dashboard creation. Recent approaches have explored supporting faster dashboard authoring through natural language interfaces [10] or by providing example images to bootstrap dashboard creation [3]. Tools like VizDeck [2] focus on data *exploration* by showing users many ranked charts that can be saved and combined into a dashboard. Furthermore, multi-view recommendation approaches like MultiVision allow users to take a single visualization and augment it with other encodings and visualizations to create complementary views [19]. Our approach differs in that we assume the user has already explored their dataset and wants to focus on the presentation of specific columns compared to one another, namely metrics vs dimensions. Therefore we focus on recommending an encoding of user-selected data as a dashboard, rather than supporting data discovery.

Most relevant to our work is the Medley system that allows users to specify an analytic intent such as measure analysis and fields of interest and choose from recommended visualization groups [6]. Our work differs in that we do not require users to specify their analysis goal upfront and we focus on a top-down specification of a dashboard. This makes the specification simpler since users do not have to select a task in addition to the fields they wish to visualize in their dashboard sections. Additionally, the flexibility of our specification allows for the creation of dashboards with many of the same attribute combinations as Medley by describing sections of metrics and dimensions. Future work could use our dashboard specification for task-based recommendations to support authoring experiences similar to Medley.

3 Quick Dashboarding

To help users create dashboards by only specifying columns of interest from their data, we developed a novel dashboard specification and UI tool, *Quick Dashboard*. In this section, we describe our process of developing the underlying dashboard specification and the UI interface. We also discuss several example dashboards that can be created with this specification.

3.1 Goals & Requirements

In our discussions with current dashboard users at a large technology company, we found a common need was better support for quickly authoring a new dashboard without having to specify each component chart and query. In order to support this data-driven and fast dashboard creation we identified the following requirements of our specification and system:

R1: **Specify dashboard based on repeated metrics and dimensions:** Users should be able to create a dashboard by selecting which fields of a dataset they would like to present without having to specify *how* to visualize each combination of fields.

R2: **Allow customization afterward:** After we recommend an initial dashboard, users should be able to easily change chart types if they do not like the defaults.

R3: **Single recommendation:** For each input specification, only a single dashboard should be generated. Since verifying dashboard designs is slow, we only show a single design to a user that they can edit rather than having them browse recommendations.

3.2 Developing the specification

To generate a higher-level, data-focused, dashboard specification, we looked for repetitive structure among existing dashboards and considered the primary goals users have when creating dashboards. We reviewed dozens of internal and customer dashboards and found that the dominant goal of dashboards was to report and break down *metrics* of interest. Metrics are business meaningful numbers such as the total number of sales. To provide a more in-depth picture of metrics, dashboards often break them down by other aspects of the data. For example, looking at the total number of sales across time or by region. We refer to these fields used to break down metrics as dimensions. Dashboards often have a logical structure to help make visual interpretation easier by grouping charts that break down the same metric into *sections*. These observed patterns informed our dashboard specification that we present in the next section.

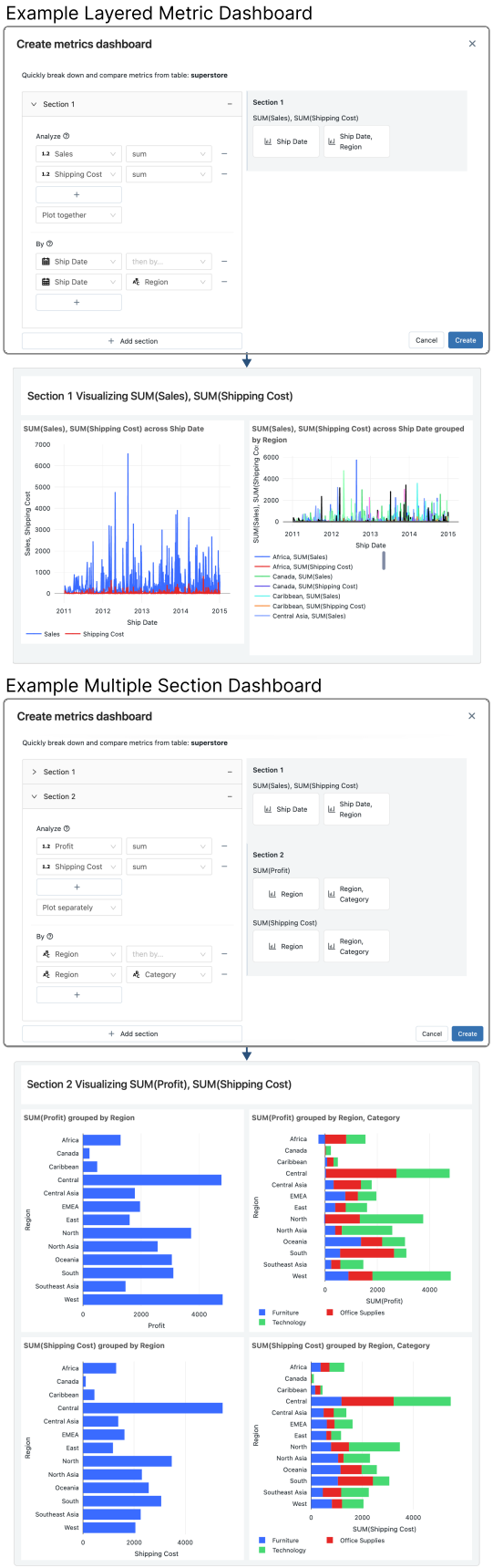

Fig. 2: **Top:** A dashboard where two metrics, Sales and Shipping Cost, are layered together across two different dimension groups. **Bottom:** A dashboard with multiple sections. The first section is the same as the top dashboard and thus not displayed in the preview.

3.3 Dashboard Specification

Our dashboard specification relies on two different types of fields (i.e. columns in a table): metrics and dimensions. This terminology is similar to many existing systems for data analysis and dashboard creation such as Tableau [11], although they refer to metrics as measures. In our case, we use the following definitions:

- *Metric:* Metrics are quantitative fields that include their preferred aggregation (such as sum or mean).
- *Dimension:* Dimensions are categorical or temporal fields that are used to break down metrics.

By selecting only metrics and dimensions of interest, users can fully specify their dashboard (R1). We use the following specification to describe a dashboard:

```
Dashboard {
    Sections: [{
        Metrics: Field[],
        DimensionGroup: {PrimaryField, SecondaryField},
        MetricLayout: "Layer" | "Repeat"
    }]
}
```

There are several important details in this specification and how it is translated to an actual dashboard. First, a dashboard is a series of sections, each with its own metrics and dimensions. A dashboard can have an arbitrary number of sections, and each section has an arbitrary number of metrics and dimensions. We require each section to have at least one metric. Users can use `COUNT(*)` as a default metric to plot the count of dimensions if no other fields are specified as metrics.

In each section, all metrics are combined with all dimensions. The `MetricLayout` option determines *how* metrics in the same section are combined. When this value is `"Layer"` then the metrics are all layered onto the same chart(s), which are then combined with the different dimension groups. For example, Figure 2 (top) shows a dashboard where two metrics are combined in each chart. When this value is `"Repeat"` then each metric is put in its own chart. Figure 1 and Figure 2 (bottom) show dashboards with each metric in its own chart. By default, metrics are *repeated* and plotted in separate charts.

Since each metric (or all metrics) are combined with each dimension, we use the concept of dimension *groups*. This allows us to create charts where a metric is broken down by multiple dimensions such as breaking down Sales by Date and Region in the same chart. We limit the number of dimensions in a group to 2 since it makes encoding easier and putting more than 3 variables onto a single chart starts to become crowded and less informative.

3.4 Chart Recommendation

In order to produce charts from metrics and dimensions, we use a chart recommendation module to create charts that best visualize these fields. Users do not have to choose the chart type or encoding channel, they simply specify their metrics and dimensions according to the specification above and we produce sensible charts for them (R1, R2). We chose to recommend a single chart for each field input so that users do not have to browse options (R3), but can make modifications to the chart type and encoding afterward (R2). Our chart recommendation module takes the following as inputs: each column and its datatype, along with a preferred axis. This preferred axis allows us to force consistent axes for the same metric plotted across multiple dimensions. The chart recommender has no notion of metric or dimension and uses a rule-based approach to choose effective visualizations based on the input column data types inspired by previous chart recommendation systems [16, 18].

3.5 *Quick Dashboard* UI

The *Quick Dashboard* UI demonstrates how systems can leverage our specification to build dashboard authoring experiences. *Quick Dashboard* exists within a larger data analytics product, however similar authoring experiences can be developed in platforms that have access to data tables and can output dashboards. In our authoring flow, users first

select a table and then use the *Quick Dashboard* authoring modal to select column combinations of interest. Therefore we show a dashboard preview on the right side of the interface so that users can understand how the metrics and dimensions they choose will be translated into a dashboard. Once a user hits create, a dashboard specification is created which is then translated to an actual dashboard with the chart recommender. In Figure 1 and Figure 2 we demonstrate a filled-in UI and the resulting dashboard for three different kinds of dashboards. By using the UI, users do not have to manually author the dashboard specification. This data-first approach to dashboard authoring also makes it easier for a broader set of users to create dashboards without the need to write queries to get their data or manually author each individual chart.

3.6 Example Dashboards

In this section, we describe several example dashboards and the specification that was used to create them. These dashboard layouts are inspired by sections from real production dashboards at a large technology company and demonstrate the flexibility of our specification in producing different kinds of sections.

3.6.1 Example 1: Breaking down metrics by dimensions

Our first example dashboard demonstrates our core abstraction of breaking down multiple metrics by multiple dimensions. The UI, spec, and produced dashboard are shown in Figure 1. We use the following specification in this example:

```
Dashboard {
    Sections: [{
        Metrics: ["Sales (SUM)", "Shipping Cost (SUM)"],
        DimensionGroups: [
            {PrimaryField: "Ship Date"},
            {PrimaryField: "Region"}
        ],
        MetricLayout: "Repeat"
    }]
}
```

We have two metrics, Sales and Shipping Cost, along with two dimension groups. Each dimension group only has one dimension, Ship Date or Region. Since we are using the *Repeat* metric layout, each metric will be plotted in its own chart. We combine every metric with every dimension group in a section so we end up with 4 charts total for this dashboard. Part 3 of Figure 1 shows the generated dashboard. We layout each metric on its own row; therefore we produce a grid of charts for this dashboard. This grid layout allows users to compare metrics as well, such as comparing how Sales and Shipping Cost compare across Ship Date in the first column.

3.6.2 Example 2: Comparing metrics in the same chart

If we take the specification from *Example 1* and change the `MetricLayout` value to `"Layer"` we can compare these two metrics on the same chart. This is useful when trying to directly compare two metrics on a dashboard such as in Figure 2 (top) where we compare Sales and Shipping cost over Ship date all in the same chart. In addition to changing the `MetricLayout`, we also add another dimension to the second dimension group. Since we have two layered metrics across two dimension groups, we end up with two total charts in this dashboard. The second chart in our dashboard has 4 columns total: the two metrics (layered on the same chart) of Sales and Shipping Cost along with the two dimensions of Ship Date and Region. We can see how the simple abstraction of metrics and dimensions allows us to create sophisticated layerings of metrics and dimensions in a dashboard.

3.6.3 Example 3: Putting it all together with multiple sections

Finally, we can combine the previous two examples to create a rich dashboard with multiple sections. Each section can have related or totally new metrics and dimensions. The following specification creates the dashboard in Figure 2 (bottom) and demonstrates the flexibility of this specification to create rich dashboards just by combining metrics and dimensions into sections of dashboards.

```
Dashboard {
    Sections: [{
        Metrics: ["Sales (SUM)", "Shipping Cost (SUM)"],
        DimensionGroups: [
            {PrimaryField: "Ship Date"},
            {PrimaryField: "Ship Date", SecondaryField: "Region"}
        ],
        MetricLayout: "Layer"
    },
    {
        Metrics: ["Sales (SUM)", "Shipping Cost (SUM)"],
        DimensionGroups: [
            {PrimaryField: "Region"},
            {PrimaryField: "Region", SecondaryField: "Category"}
        ],
        MetricLayout: "Repeat"
    }]
}
```

4 Discussion and Future Work

In this section, we discuss the implications of *Quick Dashboard* and the underlying specification and opportunities for future avenues of work.

4.1 Focusing on data rather than visual specification

Since the *Quick Dashboard* UI allows users to focus on the aspects of their data they are interested in displaying, rather than *how* to construct visualizations for this data, it speeds up the dashboard creation process. Even if several of the automatically generated visualizations need to be tweaked, *Quick Dashboard* can help users bootstrap their creation process. We focus automation at the chart level, rather than the entire dashboard level. This is different than some other recommendation approaches that focus on recommending entire dashboards or multi-view visualizations [14, 19]. Users have the best idea of what combinations of data will be interesting to them but should not have to worry about how to actually encode those combinations. By using a higher level of abstraction for dashboard creation, we view *Quick Dashboard* as a first step towards more fluid and rapid dashboard authoring flows. This makes it easier for a wider range of users to create dashboards, instead of just consuming them [12].

4.2 Interactive data-focused dashboard authoring

When there exists an underlying specification for describing dashboards, it makes future authoring experiences easier to implement. For example, in the current *Quick Dashboard* system, the authoring flow is one way: users select their metrics and dimensions while viewing a preview of this generated dashboard, and then the dashboard is created and potentially tweaked. Future authoring flows can support editing dashboards by selecting metrics and dimensions or creating dashboards in real-time from the specification, eliminating the need for a preview. Regardless of the specification UI and flow, our dashboard specification offers a useful formalism for how to think about dashboards and their specification from data.

4.3 Further incorporating best practices

We can further incorporate design constraints for multi-view visualizations to make it easier for dashboard authors to follow best practices. We obey constraints for axis consistency from [7] since this makes interpretation easier, and plan on incorporating consistent color constraints in the future. Our dashboard specification can be used independently of the chart recommender so improvements to chart recommendation will help make our generated dashboards better.

5 Conclusion

We present a data-first dashboard specification, where each section contains metrics combined with dimensions. To support authoring dashboards with this specification, we created the *Quick Dashboard* tool that provides a no-code user interface and chart recommendations. We demonstrated the flexibility of the proposed specification and tool through examples of authoring three real-world dashboards. Our specification of dashboards opens up new opportunities for making dashboard authoring experiences faster and easier.

ACKNOWLEDGMENTS

Thanks to the Visualization team at Databricks for their feedback on this work.

REFERENCES

[1] Google. Google looker. https://cloud.google.com/looker, 2022. 2

[2] A. Key, B. Howe, D. Perry, and C. Aragon. Vizdeck: Self-organizing dashboards for visual analytics. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, p. 681–684. Association for Computing Machinery, New York, NY, USA, 2012. doi: 10.1145/2213836.2213931 2

[3] R. Ma, H. Mei, H. Guan, W. Huang, F. Zhang, C. Xin, W. Dai, X. Wen, and W. Chen. Ladv: Deep learning assisted authoring of dashboard visualizations from images and sketches. *IEEE Transactions on Visualization and Computer Graphics*, 27(9):3717–3732, 2021. doi: 10.1109/TVCG.2020.2980227 2

[4] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6), 2007. doi: 10.1109/TVCG.2007.70594 2

[5] Microsoft. Microsoft powerbi. https://powerbi.microsoft.com/en-us/, 2022. 2

[6] A. Pandey, A. Srinivasan, and V. Setlur. Medley: Intent-based recommendations to support dashboard composition. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1135–1145, 2023. doi: 10.1109/TVCG.2022.3209421 2

[7] Z. Qu and J. Hullman. Keeping multiple views consistent: Constraints, validations, and exceptions in visualization authoring. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):468–477, 2018. doi: 10.1109/TVCG.2017.2744198 2, 4

[8] J. C. Roberts. On encouraging multiple views for visualization. *Proceedings. 1998 IEEE Conference on Information Visualization. An International Conference on Computer Visualization and Graphics (Cat. No.98TB100246)*, pp. 8–14, 1998. 2

[9] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher. What do we talk about when we talk about dashboards? *IEEE Transactions on Visualization and Computer Graphics*, 25(1):682–692, 2019. doi: 10.1109/TVCG.2018.2864903 1, 2

[10] A. Srinivasan and V. Setlur. BOLT: A Natural Language Interface for Dashboard Authoring. In T. Hoellt, W. Aigner, and B. Wang, eds., *EuroVis 2023 - Short Papers*. The Eurographics Association, 2023. doi: 10.2312/evs.20231035 2

[11] Tableau. Tableau dashboards. https://www.tableau.com/learn/get-started/dashboards, 2022. 2, 3

[12] M. Tory, L. Bartram, B. Fiore-Gartland, and A. Crisan. Finding their data voice: Practices and challenges of dashboard users. *IEEE Computer Graphics and Applications*, 43(1):22–36, 2023. doi: 10.1109/MCG.2021.3136545 4

[13] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, 2 ed., 2001. 2

[14] 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), sep 2015. doi: 10.14778/2831360.2831371 4

[15] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, p. 110–119. Association for Computing Machinery, New York, NY, USA, 2000. doi: 10.1145/345513.345271 2

[16] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Towards a general-purpose query language for visualization recommendation. In *ACM SIGMOD Human-in-the-Loop Data Analysis (HILDA)*, 2016. doi: 10.1145/2939502.2939506 2, 3

[17] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Towards a general-purpose query language for visualization recommendation. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, HILDA '16. Association for Computing Machinery, New York, NY, USA, 2016. doi: 10.1145/2939502.2939506 2

[18] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2016. doi: 10.1109/TVCG.2015.2467191 2, 3

[19] A. Wu, Y. Wang, M. Zhou, X. He, H. Zhang, H. Qu, and D. Zhang. Multivision: Designing analytical dashboards with deep learning based recommendation. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):162–172, 2022. doi: 10.1109/TVCG.2021.3114826 2, 4