DOWNLOAD PDF BEST PRACTICES FOR DESIGNING EFFICIENT TABLEAU WORKBOOKS

Chapter 1 : Tableau Software - ITKB - Confluence

The complete guide to designing perfect workbooks in Tableau that are efficient, performant, and effective. Developing charts and dashboards in Tableau is a new experience for many users and there are techniques and best practices they need to learn in order to create efficient workbooks.

Tables and crosstabs have their place and are supported and we will talk more on how to best use them later. Interactive â€” Tableau documents are primarily designed for interactive delivery to users, either on their desktops, over the web or on a mobile device. Unlike other BI tools that primarily produce print-focused output either to actual paper or to a document such as a PDF , the focus is on creating rich, interactive experiences that allow users to explore data and be guided through business questions. Iterative â€” discovery is an inherently cyclical process. Tableau is designed to speed the cycle from question to insight to question so that users can quickly develop a hypothesis, test it with available data, revise that hypothesis, test it again, and so on. Fast â€” historically the BI process has been slow. Slow to install and configure software, slow to make data available for analysis and slow to design and implement documents, reports, dashboards, etc. Tableau allows users to install, connect and develop documents faster than ever before â€” in many cases reducing the time to produce an answer from months or weeks to hours or minutes. Simple â€— traditional enterprise BI tools are often beyond the capability of most business users, either through cost or complexity. In many cases, users need the assistance of IT or a power user to help create the queries and documents they want. Tableau provides an intuitive interface for non-technical users to query and analyse complex data without needing them to become database or spreadsheet experts. Beautiful â€” they say beauty is in the eye of the beholder, but when it comes to visual communication there are best practices to be followed. Ubiquitous â€” increasingly, users are no longer creating documents for a single delivery platform. Users need to view and interact with data on their desktops, over the web, on mobile devices, embedded in other applications and documents, and more. Tableau allows a single document to be published and then used across all these platforms without any porting or redesign. What is Tableau not really good for? We suggest you consider revisiting your requirements or consider another approach if: Tableau can produce multi-page reports but they lack the level of format control that is available in dedicated, banded-style reporting tools. Tableau can be used to create push-delivery systems but this is not a native feature of Tableau. Tableau Server includes the concept of report subscriptions but this is a per-user pull model vs. The primary use case for the reader is to export the data to another format often a CSV or Excel file. This often means a tabular report with many rows of detailed data. To be clear, Tableau does allow users to export data from a view or dashboard to Excel â€” either at a summary or detail level. However, when the primary use case is to export it means this is an ersatz extracttransform-load ETL process. There are much more efficient solutions than a reporting tool to achieve this. You need highly complex, crosstab-style documents that perhaps mirror existing spreadsheet reports with complex sub-totalling, cross-referencing, etc. Additionally, there may be the need for scenario modelling, what-if analysis and even write-back of assumption data. Push Intelligence for Tableau - http: One of the powerful features of Tableau is its ability to connect to data across many different platforms. Broadly speaking these platforms can be characterised as one of the following: Each type of data source has its own set of advantages and disadvantages, and is treated uniquely. Minimising the differences between the platforms is something Tableau will work towards but today there are some data sources that are only supported on one platform. General advice Use native drivers As of version 9, Tableau supports native connectivity to over 40 different data sources. This means Tableau has implemented techniques, capabilities and optimizations specific to these data sources. Engineering and testing activities for these connections ensure they are the most robust Tableau has to offer. Tableau also supports general-purpose ODBC for accessing data sources beyond the list of native connectors. There can be differences in how each database vendor interprets or implements capabilities of the ODBC standard. In some cases Tableau will recommend or require you to create a data extract to continue working with a particular driver. If there is a native driver for the data source you are querying you should use this over the ODBC connections as it will generally provide better performance. Test as close to the data as possible As stated earlier, a general principal is that if a data source performs queries slowly then the experience in Tableau will be slow. A good way to test the raw performance of the data source is to if possible install Tableau Desktop on the machine where the data source resides and to run some queries. This will eliminate factors such as network bandwidth and latency from the performance and allow you to better understand the raw performance of the query in the data source. Additionally, using the localhost name for the data source instead of the DNS name can help determine if environmental factors such as slow name resolution or proxy servers are adding to the poor performance. In general it is best practice to import file-based data sources into the Tableau fast data engine. This will make queries perform much faster and also results in a much smaller file to store the data values. However, if the file is small or if you need a live connection to the file to reflect changing data you can connect live. This is called a shadow extract and it makes working with the data much faster than if you were to directly query the file. You may notice that the first time you use a large file it can take several seconds to load the data preview pane. This is because Tableau is extracting the data from the file and writing it to a shadow extract file. By default these files are created in C: Tableau keeps shadow extracts for the five most-recently used file data sources in this directory, deleting the least recently used file when a new one is created. If you subsequently reuse a file that has a shadow extract Tableau simply opens the extract file and the data preview appears almost instantly. Although shadow extract files contain underlying data and other information similar to the standard Tableau extract, shadow extract files are saved in a different format with a. Legacy connectors for Excel and text files Prior to Tableau 8. However there are some situations in which you might prefer to use the legacy drivers â€” e. In these scenarios users have the option to revert back to the legacy JET driver. A detailed listing of the differences between the two drivers can be found here: These can be row or column based, personal or enterprise, and accessed via native drivers or generic ODBC. Changing or tuning these will usually require assistance from your DBA, but can yield significant performance improvements. Row-based storage layouts are well-suited for OLTP-like workloads which are more heavily loaded with interactive transactions. Column-based storage layouts are well-suited for analytic workloads e. Today, many high performing analytic solutions are based on column-based RDBMS and you may find your queries perform faster if you use such a solution. Indexes Correct indexing on your database is essential for good query performance: Make certain you have indexes on columns used in filters. Be aware that using discrete date filters in some databases can cause queries to not use indexes on date and datetime columns. Ensure you have statistics enabled on your data to allow the query optimiser to create highquality query plans. Many DBMS environments have management tools that will look at a query and recommend indexes that would help. Consider an example where you have a dimension with incomplete data and some records contain NULL values. Join culling allows us to query only the relevant tables instead of all tables defined in your join. Consider the following scenario where we have joined multiple tables in a small star schema: With join culling, double-clicking on the Sales measure generates the following query: For example, if our fact table contained data for but the time dimension table only had values for , the result SUM [Sales] would potentially change depending on whether the time table is included. Note that while Tableau can use either hard or soft referential integrity it is often much better to use hard referential integrity because the database can do join culling too. For more information, see the following series of articles by Russell Christopher on his Tableau Love blog: Partitioning is a recommended strategy for large data volumes and is transparent to Tableau. Partitioning works well for Tableau if it is done across a dimension â€” e. Temp Tables There are many operations in Tableau that can result in the use of temp tables â€” e. It is recommended that you grant your users permission to create and drop temp tables, and ensure your environment has sufficient spool space for their queries they are running. The key points to keep in mind are that both have the same user interface in Tableau, the same visualisations, and the same expression language for calculated measures. The differences are mostly to do with metadata how and where it is defined , filtering, how totals and aggregations work and how the data source can be used in data blending. More details on the differences using Tableau over relational data sources vs. OLAP data sources can be found in the following knowledge base article: Tableau retrieves leaf level nodes not drill-through level data and makes them into a relational data source. Since multidimensional to relational transformation does not preserve all cube structures, switching back and forth between extract and live connection freely without impacting the state of your visualisation is not supported for cube extracts. You will need to make your choice before you start building up your viz. You can switch between alias options key, long name etc. Tableau also supports Spark SQL, an open source processing engine for big data that can perform up to x faster than MapReduce by running in-memory rather than on-disk. Impala is generally much faster than Hive and Spark is proving to be faster still. Even with these additional components, Hadoop is often not sufficiently responsive for analytical queries like Tableau creates. Further details for improving performance against Hadoop data sources can be found here: Using Tableau Server, this update process can be automated and scheduled. We will not cover them further here refer to the relational data source section above other than to point out you will generally want to keep these as live connections to avoid transferring large volumes of data from the cloud. Salesforce When connecting to Salesforce the following limitations of the connector need to be considered: API quotas - Hitting Salesforce live too often can cause an account to be suspended if the daily quota is hit. To maintain optimum performance and ensure that the Force. The initial extract can be very slow The first time you extract the data out of Salesforce, it could take a while based on table size, force. This is because objects are downloaded in their entirety. You cannot pre-filter data There is no ability to pre-filter the data through the Salesforce connector. If this requirement is critical for you, you could use a third party Salesforce ODBC driver for example from Simba, or DataDirect that supports live connections.

Chapter 2 : Best Practices for Published Data Sources - Tableau

Best Practices for Designing Efficient Tableau Workbooks, the V edition Posted on June 16, by Alan Eldridge With the impending release of Tableau 10, I have been refreshing my whitepaper on how to author efficient Tableau workbooks.

Here are the basic steps: Test the workbook in Tableau Desktop. Does it perform well? Test the workbook in Tableau Desktop on the Tableau Server machine. Does it perform the same as it did on the previous machine? Publish the workbook to Tableau Server, and find a time when there is low-to-no usage on the Tableau Server. Go to the published workbook. Did it perform relatively the same as the test in Step 2 within seconds? Test the workbook during a time of high usage on the Tableau Server either natural or do load testing using TabJolt. This means that the first testing stage is always in Tableau Desktop. Isolating to Desktop If you are using a workbook that has Published Data Sources, your queries will still be going through Tableau Server. To get all the processing moved to Desktop, follow these steps: Download the workbook file from Tableau Server to the local machine. You will need the database credentials to you can log into the database once it the source is local. These should be the same credentials used on Tableau Server. Some databases have resource limits in place for certain users, so at this point we want to be testing with the exact same credentials. The best instructions are from the Tableau Online Help. If you already have the workbook open, you can go up to the Data menu at the top, find the data source name and then choose Refresh. Once it has reloaded, click around or change filters whatever actions have been slow on the Server. Now end the performance recording. The Performance Recording will be a new workbook which will show which steps are taking the most time. They should be able to make changes or recommendations that will optimize the query at the database level. If they have reached a state where they feel like things are as optimized as possible, but Tableau is still doing some things that will necessarily take a long time, consider altering the worksheet where the inefficient query is running to see if Tableau will write a simpler query. Run the Tableau query in your direct querying tool. How many results come back you want a full row count, usually displayed somewhere. Most SQL querying tools limit the actual number of results they return back. If the limited amount returns quickly, then that indicates the dashboards and sheets need to be optimized to return less data until a user has filtered down. Have your DBA run the exact same query from Tableau directly on the database or as close to directly as the can. This tool should be using the same driver as Tableau Desktop it should be using the ODBC driver installed on the desktop machine. If the results are the relatively the same from the direct query on the database and using the SQL querying tool, but much slower in Tableau Desktop, open up a Support case and provide all of the testing steps, the results you are seeing, and any logs from Tableau Desktop so that the issue can be investigated. This step helps determine if there is something amiss in the actual configuration of the Tableau Server machine. Copy the workbook file over to the Tableau Server machine and install the exact same version of Tableau Desktop that you tested on previously. Open up the workbook, and run a Performance Recording just like you did earlier. If your querying is running significantly different than previously, then you have the 3 situation from before. Take the same steps to isolate if the issue is specific to Tableau or just to that machine. If it is Tableau specific, open a Support case. If machine specific, have your local IT team investigate what is causing the difference on that machine. You will need to publish the workbook in two variations: Tableau Server has Performance Recorder as well, and the Server Guide describes exactly how to use it. You are basically looking for any differences from Desktop to a single load of the workbook in Server. If the performance is relatively the same as it was in Desktop, then the issue most likely occurs during heavy load. If everything seems to be taking longer, or the queries are the same length but everything else is taking longer, then there may be an issue with the Tableau Server software. If this workbook performs poorly as well, with no other load on the, then there is something wrong with the Tableau Server software or its interaction with the underlying machine directly. One hidden place to look for performance issues is anti-virus actively scanning on the machine. Once that has been ruled out, you may want to look at Windows Performance Monitor to see what is happening.

You can also use TabMon to bring all of the various aspects of the Tableau Server software in view at once. Testing Tableau Server Under Load If the workbook only seems to be performing poorly under load, it is time to look at a performance recording while the server is busy. You can also investigate this from the other side, by having your DBA look at the number of incoming queries at a time. If you need to simulate a lot of load on a testing database , then you should use TabJolt. Tableau sends queries in parallel to a database. If your DBA says that the number of simultaneous queries is too much for the database to handle, you can place limits on the number of connections per process using this KB article. If query times are not the issue, then it is time to look at the TabMon results to see what part of the system is getting overloaded.

Chapter 3 : Best Practices for Designing Efficient Tableau Workbooks - PDF documents

An efficient workbook is: A workbook that takes advantage of the "principles of visual analysis" to effectively communicate the message of the author and the data, possibly by engaging the user in an interactive experience.

As a result, we do not attempt to deliver the same kind of experience as traditional enterprise BI platforms. Tableau is at its best when used to create workbooks that are: Visual â€" there is a mountain of evidence that shows the most effective way for humans to understand large, complex sets of data is through visual representation. Tables and crosstabs have their place and are supported and we will talk more on how to best use them later. Interactive â€" Tableau documents are designed for interactive delivery to users, either on their desktops, over the web or on a mobile device. Unlike other BI tools that primarily produce print-focused output either to actual paper or to a document such as a PDF , the focus is on creating rich, interactive experiences that allow users to explore data and be guided through business questions. Iterative â€" discovery is an inherently cyclical process. Tableau is designed to speed the cycle from question to insight to question so that users can quickly develop a hypothesis, test it with available data, revise that hypothesis, test it again, and so on. Fast â€" historically the BI process has been slow. Slow to install and configure software, slow to make data available for analysis and slow to design and implement documents, reports, dashboards, etc. Tableau allows users to install, connect and develop documents faster than ever before â€" in many cases reducing the time to produce an answer from months or weeks to hours or minutes. Simple â€" traditional enterprise BI tools are often beyond the capability of most business users, either through cost or complexity. In many cases, users need the assistance of IT or a power user to help create the queries and documents they want. Tableau provides an intuitive interface for non-technical users to query and analyse complex data without needing them to become database or spreadsheet experts. Beautiful â€" they say beauty is in the eye of the beholder, but when it comes to visual communication there are best practices to be followed. Ubiquitous â€" increasingly, users are no longer creating documents for a single delivery platform. Users need to view and interact with data on their desktops, over the web, on mobile devices, embedded in other applications and documents, and more. Tableau allows a single document to be published and then used across all these platforms without any porting or redesign. Cookies are required to access some content on our site. It only takes 15 seconds to fill out.

Chapter 4 : Isolating Tableau Server Performance Issues | Tableau and Behold!

is at its best when used to create workbooks that are: Visual - there is a mountain of evidence that shows the most effective way for humans to understand large, complex sets of data is through visual representation.

How do you get all that information into a dashboard? Making a dashboard in Tableau can be so technically easy just drag and drop , but then you discover that it can also be very difficult to make a meaningful dashboard. To begin with, the layout of your dashboard is dependent on what its purpose is, who will be using it and how they will be using it. This basic table illustrates just a few of the possible combinations of factors that need to be considered when building your dashboard. Juice Analytics provides a great guide for dashboard design that details requirements for interactive dashboards. Bad dashboards are cases where good information is being ignored because the user is being confused or abused. This is where beauty comes in. Not pretty, not cool, but beautiful. Beauty is meaningful design. Are there hard fast rules to follow? Do this, then this, then thisâ€¦? Its simple design is not off putting even though it is a complex puzzle. Each piece should have a good reason for being on the dashboard. The grid or columnar layout is particularly useful for exploratory and scorecard dashboards that display a lot of measures and provide the user with many filter options. To maintain simplicity there is only one font type Arial , a limited color palette, no data labels or mark shapes, and the filters are arranged in a column. The detailed information is available to the user in the tool-tip. A dashboard should be laid out in a balanced way. The most important items should have prominence. The order, sizing and grouping of elements goes from overview big picture to detail. While grid or columnar presentations are most logical and easy to balance, freeform dashboards can be much better for storytelling. The golden ratio is most helpful in achieving balance on this kind of dashboard. Use it as a general guide. This freeform dashboard presents much more detailed information at a glance, but the large map lets users know where to start and the detailed information is clutter free and not overwhelming. The same reaction can happen with an overwhelming dashboard. Aim for minimal complexity and maximum ease of use. I particularly like the hover filter action in Tableau. People are likely to accidentally filter or filter without having to make a decision while they are getting the hang of the dashboard. This dashboard has a fairly busy looking rank chart as the prominent piece of information. To make it easier to use, there is only one hover filter and two columns of information. The second column displays the text results from the chart making it easy for the user to instantly read the information. The bubble chart presents at a glance results through the use of color and size. Further down, the reader can see more detailed information, which also uses color to highlight negative values. Balancing simplicity, clarity and efficiency when building a dashboard is not an easy task and often requires many iterations. This is where an editor is invaluable. I have a few that will review for me, but one that I can count on to attack my dashboards. And yes, sometimes I do things for design that are not necessarily best practices for displays of analysis. I have added a green chart for no good reason other than it provided balance to the other colors on the dashboard. The web is full of bad dashboards. I would estimate that Stephen and Alberto have taught me to pay attention to not only the analysis, but the principles of perception and storytelling. Tableau Public authors too many to mention here and Web designers are also a great source inspiration, particularly for dashboard design and interactivity. Organize your books by color, take an online drawing class , doodle, sing, or daydream. Allow yourself to be playful and even a little goofy. Click here to view the Dashboard Design Layout workbook.

Chapter 5 : Performance | Tableau and Behold!

Best Practices for Designing Efficient Tableau Workbooks Kommentare: 2 This is the Link to our third best practices document about how to design efficient Tableau workbooks from MR High Efficient itself - Alan Eldridge.

Tweaking the server for better performance Alerts: Get notifications for server events Alerts are email notifications that you receive when something happens on Tableau Server. You can set up alerts for when the server is running out of disk space and for when server processes stop or start. These conditions often mean that there is an immediate problem. We discuss alerts in this section as a tool for getting information about server health. But as an entirely separate benefit, users can also make use of alerts. After you set up alerts, your users can subscribe to views to periodically receive a snapshot of views they are interested in on a recurring basis. An SMTP server is a service that you can send outbound email messages to. This is often something like smtp. This is 25 for most servers. Click Notificationson the Configuration tab and click Email Server. Click Apply Changes and Restart. Set up notifications Open TSM in a browser: Click Notificationson the Configuration tab and click Events. Configure notification settings for your organization: If you do select all the check boxes, here are the alerts that get activated. This can be useful if your users want to see information about views on a recurring basis. For example, users can get a view in their inboxes every week. See the Additional resources section at the end to read more about how users can set up subscriptions. Server component events For installations of Tableau Server on a single computer as described in this guide , you can receive a notification when Tableau Server processes stop or start. Because part of the server must be running to send an alert that processes have stopped, you only see notifications when the data engine, repository, and gateway processes stop. However, you see notifications for all Tableau Server processes that start. Anytime that server processes stop or that the server restarts unexpectedly, you should investigate the cause of the restart. For example, you may discover that the Windows Server computer is configured to restart automatically after Windows updatesâ€”in which case you may want to schedule updates for off-peak hours. Low disk space You can receive a notification when the disk space on the server computer falls below a threshold that you specify. Eventually, the server may even stop responding. Administrative views Administrative views are views that are built into Tableau Server and that provide an easy way for you to start understanding activity on Tableau Serverâ€”whether the activity comes from users or from server tasks like extracts. Sign in to Tableau Server as a server administrator. Tableau Server displays a list of administrative views. Here are the most important administrative views that you should look at when monitoring Tableau Server: Traffic to views Use this view to identify peak hours for user traffic. This is easier if you use the Time Range filter. For example, in the screenshot below, the filter is set to Last 3 months. If you know when users need your server most for their work, you can make sure that you schedule things like extract refreshes which can take up server resources for when usage is lightest. For example, in the screenshot, you can see that peak hours are between 1 PM and 3 PM. You can also tell that there is less traffic in the mornings than in the afternoons, and that traffic drops sharply outside of business hours. Background tasks for extracts Use this view to identify times of the day when extract refresh tasks take longer than usual and to identify extract refresh tasks that did not complete. If there are peak times for extract refresh tasks, distribute the extract refresh schedule so that fewer extracts run at the same time. Alternatively, if individual extracts are taking a long time, see if you can use Tableau Desktop to reduce the amount of data pulled into the extractâ€”for example, you can sample rows, hide unused fields, and aggregate data for visible dimensions. If you notice that there are particular extract refresh tasks that do not complete, try to connect to the data source from outside of Tableau Server to confirm that the connection information is correct and that the data source is available. For example, in the screenshot you can see that many extracts are failing, but you can also see that the failed extracts had very short runtimes. This might indicate that the data source for those extracts is unavailable. Stats for load times Use this view to identify which views are loading slowly. This helps you pinpoint workbooks that are very processing-intensive for the serverâ€”inefficient workbooks are one of the most common reasons for poor server performance. We list some resources later that can help you optimize and troubleshoot workbooks. Alternatively, it may be that a view is not processor-intensive but takes a long time to load because of data source constraints. You can also use extracts to avoid making live requests to a slow data source. For example, in the administrative view below, you can use the exact load times for views to see that the top views load significantly more slowly than the rest. However, you can also use the Windows Performance Monitor PerfMon included with Windows Server to monitor server processes and resource utilization. You can use PerfMon to gather very detailed information about every part of a server installation, including how often the computer is using the CPU, how much memory is being used, and even information about each Tableau Server process. For those who are new to PerfMon, it works by taking a snapshot of selected processes at regular intervals. The first step in using PerfMon is to set up a data collector set, which is how PerfMon stores the data that it collects. To collect information about Tableau Server processes with PerfMon, Tableau Server must be running when you create the data collector set. Create a new data collector set Click the Windows Start menu and search for "performance". Right-click Performance Monitor and then click Run as administrator. In the left pane, click Data Collector Sets. In the Create new Data Collector Set wizard, enter a name for the data collector set. For example, you might enter Tableau Server Performance. Select Create manually Advanced and then click Next. Under Create data logs, select Performance counter, and click Next. Select performance counters Set the sample interval to 30 seconds. Select performance counters from the list. The following table lists some performance counters that we recommend for tracking Tableau Server performance.

Chapter 6 : Designing efficient tableau workbooks 9 0 1 by Fabio Krauss Stabel - Issuu

Best Practices for Designing Efficient Workbooks We've had the benefit of Alan's internal version of this white paper for months. Now YOU can enjoy it too. This is good stuff.

The Tableau Performance Checklist: Each post expands upon one item listed in the master Tableau Performance Checklist. Work with a subset of your data. Extract a sample if needed. Take these examples for Sales by Item: Which viz above do you think communicates the data more effectively? We need to go deeper. It Starts with Good Preparation Just as important to your visualization is how you prepare your data for your worksheets. It can have a tremendous impact on their performance. There are a few things to consider when trying to keep your data as trim and streamlined as possible. Dashboard Performance Your visualizations can only perform as fast as your underlying data sources. That is the great performance inhibitor that makes great dashboards into headaches. The key to improving performance for slow data is to extract only the data that is needed for the worksheet to perform its analysis. Report Portability When you create a data extract, you create a TDE that can serve a single workbook or multiple workbooks. The good news is that a TDE can be packaged with a workbook as a self-contained reporting suite that is not reliant on a connection back to the data source. As such, it can be shared and disseminated far more easily than a workbook that relies on a live data connector. Data Source Efficiency Not only can a data extract improve the performance of your dashboard, but you can also improve the efficiency and performance of the data source itself. A data extract does not need to communicate with the data source from which it was drawn. It is a self-contained data subset. Aggregated Extracts When you create a data extract, Tableau also gives you the option to aggregate your data for all visible dimensions. Rather, it contains only the aggregated data. An aggregated extract is smaller than a standard data extract, and it creates another level of efficiency in generating fast performance with your dashboards and worksheets. Calculated Fields When you create a data extract, all calculated fields in the extract are converted to static fields. This is a great way to reduce the heavy lifting of the data source whenever data is accessed via a live connection. Calculated fields that are converted into static values can be accessed and aggregated as easily as any other field. These are particularly effective when dealing with string calculations, which are far more resource-heavy than numeric or date calculations. For more insight on calculated fields, check out this blog. Check back frequently as we add new posts and dive deeper into each point in the Tableau Performance Checklist. As always, feel free to get in touch with us if you have any questions regarding performance or anything Tableau related!

Chapter 7 : Notifications, Monitoring, and Tuning - Tableau

p 2 Best Practices for Designing Efficient Tableau Workbooks Foreword I would like to acknowledge that this document is a distillation of materials written by many authors. All I have done is to bring it together into a single document and try to apply some structure.

What makes up a published data source A Tableau data source consists of the following: The data connection information that describes what data you want to bring in to Tableau for analysis. When you connect to the data in Tableau Desktop, you can create joins, including joins between tables from different data types. You can rename fields on the Data Source page to be more descriptive for the people who work with your published data source. An extract, if you decide to create one. Guidelines for when to create an extract are included below, as well as in the additional resources. Information about how to access or refresh the data. The connection also includes access information. Examples of this type of information include: The path to an original Excel file. Embedded credentials or OAuth access tokens for accessing the data directly. Customization and cleanup that helps you and others use the data source efficiently. All of these refinements become part of the metadata contained in the data source that you publish and maintain. Preparing a data source for publishing When you publish a data source, consider these best practices: Create the connection for the information you want to bring into Tableau and do any customization and cleanup that will help you and others use the data source efficiently. If appropriate, create an extract of the data you want to publish. For more information, see the following section, When to use an extract. Develop a data source naming convention. After publishing a data source you cannot rename it directly. Instead, you need to publish a new copy with the new name, and then update all workbook connections. A well-considered naming convention can also help other users of the data deduce which data source to connect to. Consider designating the following roles among your Tableau users: A site administrator who manages published content, extract refreshes, and permissions on the server you publish to Tableau Server or Tableau Online. Central management helps to avoid data source proliferation. Authors who connect to managed data can be confident that the answers they find in it reflect the current state of the business. When to use an extract Under the following conditions you might be required or choose to publish an extract instead of connecting live. Publishing data to Tableau Online that it cannot reach directly Tableau Online in the cloud cannot reach data sources that you maintain on your local network. Depending on the connection, you might be required to publish an extract and set up a refresh schedule using Tableau Bridge. Some cloud-hosted data sources always require extracts. These include Google Analytics, Salesforce. You can set up refresh schedules for some of these data sources directly on Tableau Online; for others you use Tableau Bridge. Web data connector data sources always require extracts. If you connect to the data source using standard user name and password authentication, you can refresh it using Tableau Bridge. Improving performance Even if the server supports live connections to your data, an extract might make more sense. For example, if the database is large or the connection slow, you can extract a subset that includes only the pertinent information. The extract can be easier and faster to work with than connecting live. To learn more about creating data extracts, see Extract Your Data. Publishing data separately or embedded in workbooks You can publish data sources as standalone resources that workbooks connect to, or you can publish workbooks with the data sources included within them. When data is embedded in a workbook: Access to the data source is limited to the workbook in which you published it. Neither you nor other users can connect to that data from another workbook. You can set up extract refresh schedules as you do for data sources that you publish separately. Each way of publishing has its advantages. The table below shows a few common points of comparison. It is not a comprehensive list, and these are generalizations. How these and other factors apply to you are specific to your environment. Published separately Embedded in workbook Publishing data sources is a step toward centralizing data management. You can create policies geared toward minimizing data source proliferation and helping people find the right data for the work they do. Each embedded data source has a separate connection to the data. Each has the potential to show something different than the other at any given time and data source proliferation is common. Meant to be shared; becomes available for other Tableau users to connect to. Data is available only inside the workbook; it is not available for other Tableau Desktop users to connect to. Without content management and self-service guidelines, seeing a long list of data sources to connect to can be confusing to users who rely on the data to do their work, and is more difficult to manage on the server. Someone who changes a shared data source might be uncertain or unaware of the effects that those changes have on connected workbooks. Changing the data requires opening the workbook, where you can see the result of the change. Even if effects of data source changes on connected workbooks are planned, updating those connected workbooks is cumbersome. Same as above; however, if multiple workbooks use similar data and need to be updated, it might be worth connecting to a published data source instead. Extracts can be refreshed on a schedule. You set up one refresh schedule for the extract, and all workbooks that connect to it always show the most current data. If you want to keep the data fresh, each workbook must have its own refresh schedule. Generally helps you to optimize performance on the server or site. Performance might be affected when the server contains multiple workbooks that connect to the same original data, and each workbook has its own refresh schedule. Keeping extracts up-to-date When you publish a data source with an extract, you can refresh it on a schedule. For more information, see the following topics:

Chapter 8 : Best practice to share 6 different dashboards t |Tableau Community Forums

Designing Efficient Workbooks V9 Thanks to all the people whose hard work has contributed to this document, and to all the reviewers whose input has helped make this approachable. This will be available via the Tableau blog soon, but this is for the early adopters.

Best practices for Tableau 1. Publishing dashboards and worksheets In Tableau Desktop you have a workbook that is composed of dashboards and worksheets. A dashboard is like a canvas where you drop in one or more worksheets. You can also add other graphical elements and controls to the dashboard. Within a given workbook in Tableau Desktop, both dashboards and worksheets can be published to Tableau Server, where they then become views in Tableau Server. Pulling the data from a published dashboard does not work very well. Best practices You will probably want to publish dashboards for your users to see and use. But you will want to make the underlying data available so that Metric Insights can pull. So you must also publish your worksheets. Publish all worksheets in same workbook: See section below for instructions. Publish all worksheets in separate workbook: Another approach is to publish your one workbook from Tableau Desktop to two workbooks in Tableau Server. OPTION 1 - Publish all worksheets in same workbook For the one approach described above, you publish the worksheets with the dashboard, but you make the worksheet views on Tableau Server only available to Metric Insights, and not visible to your users. This section describes how to do that. Tableau Desktop - Publish all worksheets in same workbook as dashboards First, in Tableau Desktop, publish all worksheets in addition to the dashboards in the same workbook on Tableau Server. Note, if you choose "hide sheet" option in Tableau Desktop, then the worksheet is not an option under Publish to Server to publish as a view. So you must "unhide sheet" before you can publish the worksheet. Tableau Server - Deny view access permissions 1. Next, un-check the view button to remove access to all the users. This approach works if you are signed in with the credentials that Metric Insights uses for pulling data. An alternate approach is to select the edit link to the right of your workbook. Tableau Server - Deny view access permissions If you chose to Edit the permissions instead of Removing the permissions in the previous step, then follow these instructions. Select Deny for View permissions for any or all of your users. Do not deny view access for the Tableau user account that Metric Insights will use. In this example, we deny view for All Users. We do that because we are currently signed in with the Tableau user account that we use in Metric Insights for pulling the data from the views. OPTION 2 - Publish all worksheets in separate workbook As a different approach, you publish your dashboards in one workbook on Tableau Server, and your worksheets in a separate workbook. And you make the worksheet views on Tableau Server available to Metric Insights but not visible to your users. Tableau Desktop - Publish dashboards to workbook First, on Tableau Desktop publish your dashboards to the workbook on Tableau Server as you would normally do. Tableau Desktop - Publish all worksheets to separate workbook Zoom: Tableau Desktop - Publish all worksheets to separate workbook Then, on Tableau Desktop choose to publish your worksheets to a separate workbook on Tableau Server. If you are not signed in as the same Tableau user account that Metric Insights uses, then make sure that user has view permission too. Tableau Desktop - Publish all worksheets to separate workbook Finally, publish this workbook. Now only the user that published this workbook with the worksheets can access, and any additional users you gave View permission to.

Chapter 9 : Best Practices for Designing Efficient Workbooks | Tableau Software

Best Practices for Designing Efficient Tableau Workbooks Identifying Tableau query performance issues through the log files | Tableau and Behold! Like Show 0 Likes (0).