BROADCOM SOFTWARE

WHITE PAPER

Best Practices and Design Principles for Building and Utilizing a DX Dashboard

Users' Browser Experience: Insights and Analysis

BPDP-DXDA-WP100
November 14, 2022

Table of Contents

Chapter 1: Executive Summary

This paper presents overall dashboard design, panel selection and query tips and best practices illustrated in the context of a prevalent customer use case. Dashboard design principles are presented and utilized.

The use case: Knowing the user experience even when users are never faced personally within your business is paramount to any business.

During these years of any-business-is-a-software-business digital transformation and data-driven value-ops, dev-ops, and ai-ops, this has become a well-known and appreciated truth.

User experience metrics can provide insight into the quality of digital services as well as progress and the effect of improvements added uptake of new services introduced, and thus, the success of your investments in your digital transformation and evolution - actionable and quantifiable.

DX APM with included sub-components of DX AXA and DX Dashboard can provide accurate overview of and insight into the user experience as well as calling to action and commencing analysis for source and root cause of a degradation.

Background: A customer invited us to demonstrate that DX Dashboards on AXA data automatically collected within their users' browsers could readily be utilized to produce overview and insight live dashboards (and PDF reporting) being on par with competitors' solutions - we met the challenge, and more.

In this paper, I seize the opportunity to share best practices as well as technical tips intended to ignite and inspire in the utilization of DX Dashboards as a very capable tool.

1.1 Understanding the Customer Need: Questions the Business Asked

A peer of mine reached out to me when a customer asked if it was possible to provide user experience insights based on data from the Browser Agent. Specifically, the customer wanted to provide APM users with statistics on page load time, apparently similar to what Google analytics can produce.

Page load time is the time that elapses from the user presses **Enter** until the next page is fully ready and is attributed to the page loaded as measured in the browser. Thus, page load time as captured by a browser reflects the user experience[1].

The customer asked two questions:

1. How many percent of customers experience a page load time of at most 1, 2, 3, … 10 seconds for any given time range (hours, days, weeks, months)?

   In other words, this question asks for the percent distribution of users relative to response times 1 through 10 seconds, and above? - within a selectable time range.

2. What is the maximum page load time of 25%, 50%, 75%, 99%, and 100% of the user experience for any given time range?

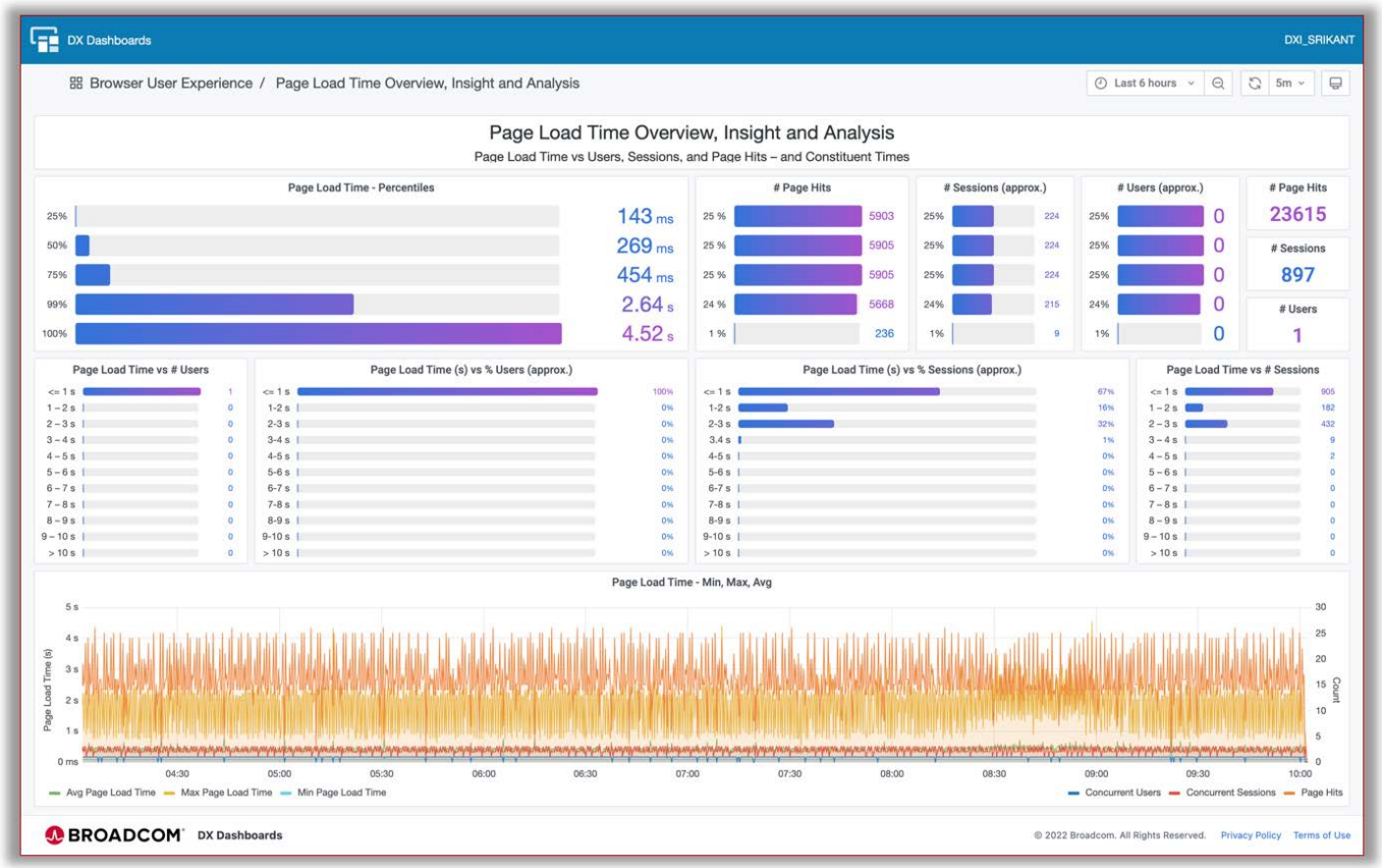
   In other words, this question asks for the five percentiles, 25, 50, 75, 99, 100, of page load time relative to number of users? - within a selectable time range.

Can APM visualize this? If so, how?

1. For PAGE events event.response.time = event.metric.page.page_load_time. There is no need to considerer the former for PAGE events (it exists to provide the same field for event response time naming across all even types).

Spoiler alert: Yes, this can readily be visualized within a DX Dashboard. Although the following figure shows the simple proof, it does not reflect my final suggestion for a dashboard. In the following sections, you will see how I refine this dashboard using easy to follow design principles and best practices.

Figure 1: Page Load Time Overview, Insight and Analysis



I will provide my reasoning at each step which you can apply to your own constructions and bring you to my final dashboard.

Chapter 2: Dashboard Design Principles

When designing dashboards, the following design principles can help guide decisions.

1. Prefer panels which include only the precise information requested or needed.
   - Help users focus. Avoid *noise* since more information compels users to interpret more data and its importance.
   - Don't repeat information. Adapt and optimize visualizations. Ensure data precisely meet their purpose by consulting use cases and personas.
   - Use the litmus test towards dashboard information: “Is it district?”, “Related?”, and “Relevant and applicable?”
   - Utilize hidden expandable rows for revealing detail panels only when and if necessary.
   - Hyperlink dashboards when expandable rows become excessive, by turning them into separate hyperlinked dashboards (use the inter-dashboard copy feature for moving them).
2. Prefer panels that visualize information in the most direct and clearest way.
   - Dashboards should be readily understandable for intended, non-APM, personas.
   - Avoid misinterpretation of data by making dashboards clear and intuitive.
   - Be careful with statistics upon statistics (eventually mark as *approximation).*
3. Prefer panels that highlight violated thresholds for calls to action or SLAs.
   - Draw attention to violations or degradations using consistent design elements such as color schemes or appropriate techniques for users with disabilities.
   - Key status information and calls to action should be prominently indicated on dashboards, enabling immediate calls to action when warranted - analyzing deeper, informing users and the business, invoking specialist peers, and so on.
4. Prefer panels that dynamically adapt to data levels or repeating data.
   - Invest in well-designed queries which consider multiple scenarios and edge cases (such as, no activity, null values in denominators).
   - Keep your dashboard maintenance manageable; adapt as-needed in smaller increments based on use case refinements and feedback from personas.
   - Use saved time to develop more dashboards!
5. Prefer actionable dashboards that fully support established workflows and collaboration by clearly guiding or linking to analysis and actionable next steps.
   - The dashboard developer should think like an architect: bridging data understanding, visualization, and business requirement acumen.
   - Do the thinking and apply that knowledge beforehand so information needed (as per use cases) is available to decide on next steps.
   - The dashboard developer should consider using expandable rows for simpler workflows and linking workflows between dashboards, providing focus, highlight and gradual narrow-in supporting stepwise analysis for more complex workflows. Utilize expandable rows to begin with, and gradually move towards hyperlinking multiple dashboards.
6. Prefer dashboards that provide needed context for correct and informed interpretation.
   - The dashboard developer should do the thinking and apply their knowledge beforehand to ensure that the information needed to draw conclusions is present.

NOTE: These design principles are intended to be applied as decision points litmus tests in the collaborative process of fleshing out dashboards to accommodate use cases and personas for the aim of arriving at a dashboard design that optimally supports use cases, personas, and workflows. Adopting these principles will compel teams to clarify needs and refine designs in a productive, iterative process.

Chapter 3: How a DX Dashboard was Built to Answer the Questions

In my experience, these distinct best practices are fundamental to building dashboards which meet persona, business and technical needs:

- Understand the needs of each persona and the use cases involved.
- Working with the personas, decide on panel visualizations to include in the dashboard.
- Investigate what data is needed for each panel.
- Build precise queries to fetch the data.
- Evaluate the dashboard with personas during dashboard development.
- Plan to assess and improve the dashboard with personas after its initial release.

For this and similar initiatives which involve multiple diverse personas, these elements should serve as guidelines; teams should work in short iterations with agility in mind to frequently gather suggestions and evaluate options and adapt as needed to increase value delivered for the personas.

That being said, my primary focus in this paper will be on *Know: sharing knowledge and considerations to master building the queries needed*. And as a prequel, establish use cases; as a sequel, refine the panels and dashboard as an evaluation iteration. Leading the all important practitioner's *How: Apply best practices dashboard design principles to arrive at a final dashboard design that optimally supports use cases, personas, and workflows*.

3.1 Uncover and State the Use Cases and their Personas

It is beneficial to consider the use cases and nuances behind the needs and questions expressed by each persona. Active reflection and team discussions will spawn creative solutions and help you reach complete, concise, and applicable answers.

In this case, the questions originated from business users needing an accurate overview of the real user experience (*accurate* and *real* meaning captured as close the user as possible).

The user forwarding the question to my peer was both an APM user at the customer, and the go to person for his business peers for monitoring user experience and for resolving issues. Thus, I decided to augment my *dashboard answer* with dashboard panels that would additionally break down the page load time into its constituent parts. This would enable the user to answer the obvious follow-up question to a degraded user experience: “Why?”, “What action is needed?”, “How do I proceed with analysis and resolution?”, and “When is it resolved?”

As the APM user was using AXA, I decided also to include panels that would enable correlation of data visualized in the dashboard to AXA data for him to eventually continue deeper root cause analysis within AXA.

The following points summarize the use case for the business user.

- Provide an accurate overview and insight into the user experience. Consider the following questions:
  - What is the user experience, and how are users being served?
  - When did degradation begin and end?
  - How severe, how many users, and how much activity?

The following points summarize the use case for the APM user.

- Provide panels to reveal correlation between degradation with activity and potential sources of degradation as well as correlation with AXA for deeper roots cause analysis. Consider the following questions:
  - Which users and sessions are affected and when?
  - What are the sources to a degraded user experience?
  - Are the degradation sources inside or outside the business?

Agreeing on use cases across personas brings an extremely helpful clarity that ensures high-value dashboards, and a shared understanding of priorities, needs and focus.

While there is overlap between the two questions and therefore between the use cases, this is not a problem. Rather it illustrates the challenge of building a dashboard which delivers without repetition and includes individual panels with distinct, relevant, and applicable information. I will consider these criteria in the following sections.

3.2 A Single Pane of Glass

Based on use cases uncovered, in my experience, it is worth exploring building common dashboards for the involved personas for collaboration on common and shared data and visualizations. The *single pane of glass* concept substantially reduces discussions about data, data sources and their validity. This saves time, improves the quality of communication, fosters collaboration, and directs action. These benefits are especially relevant when business critical needs arise, as it is obviously the case when severe user experience degradation is detected.

When working to uncover and document the use cases and needs of each persona, it is critical to agree on the data visualizations needed. As the team works toward consensus regarding the data visualizations, continually check and validate against use cases to clarify the reasoning and purpose of each element.

This approach will help solidify your data visualization decisions and speed agreement from the team as the dashboard design progresses. It leads to cleaner and crisper dashboard design and structure, so this approach provides value at a glance for each persona. The approach also streamlines data query decisions later in the process.

I invite you to see for yourself at the conclusion of this paper when I present my final dashboard, its panels, and its expandable rows.

3.3 Dashboard Structure: Panels

A dashboard is solely composed from panels. Each panel is one visual gadget: a graph, a stat, a table, a gauge, a pie chart, and so on.

Panels may be grouped into an expandable row, which through toggling, may be visible. In other words, all its panels are visible or hidden, the expandable row is collapsed into a visible place holder row. Thus, expandable rows lend themselves to digging into details, only expanding them when information of their detail panels are needed.

A panel is single specific graphical visualization of specific data: a line or bar chart, a table, a pie chart, and so on. Each panel is supported by the following components:

1. One or more queries for fetching data from the selected data source
2. Options to adapt the visualization

All panels include some statistical computation on data (average, min, max, percentiles, period or interval calculations, and so on). To be able to correctly interpret data visualized, it is important to appreciate that these computations can be performed in one of the following three distinct places:

- Within the data source as part of the query
- Within the query builder
- Within the panel

These options provide helpful flexibility in dashboard creation for massaging and visualizing data.

NOTE: Advanced use of DX Dashboards that utilize these options in combination may incur doing statistical computations on top of statistical computations. This scenario may make data and visualizations hard to interpret towards reality. I will share some considerations regarding this scenario. A very good first step is to be in the know about this caveat. Now you are.

DX Dashboards use Grafana that contains many visualizations. In the case described here, I use bar chart, graph, table, and stat. I also use expandable rows to view or hide groups of panels.

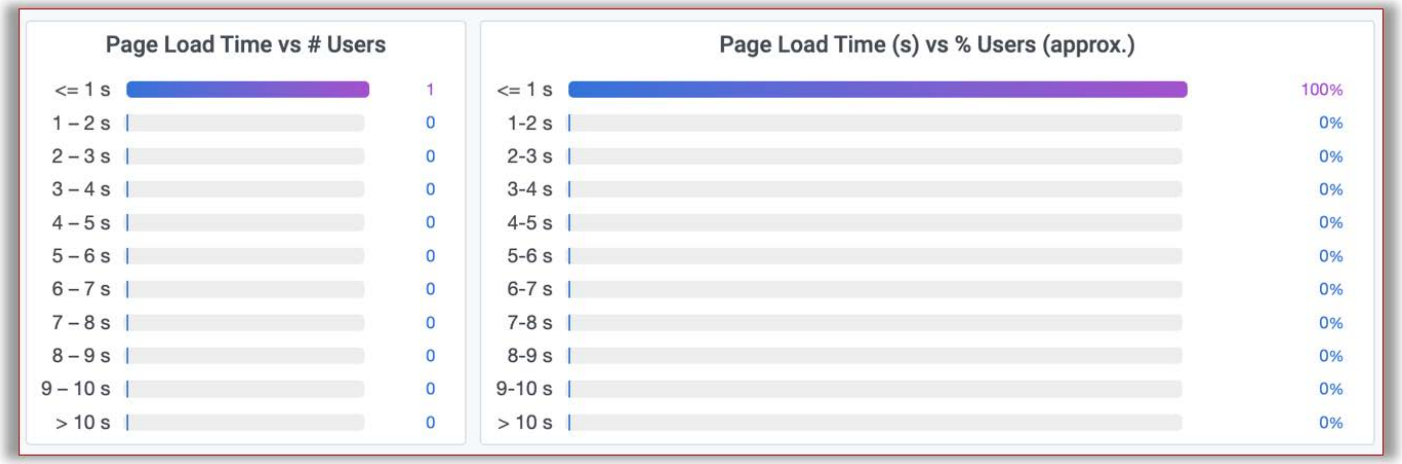
3.3.1 Panels for the Business User

For the two essential questions posed by the customer, I will first visualize the answer for each in two panels. Then, I reconsider the use cases to adapt and augment design decisions.

Question 1: Users by Page Load Time by Seconds

Where to begin? My first rough suggestion ID shown in the following figure:

Figure 2: Example Users by Page Load Time by Seconds



The bar chart in the left panel visualizes the number of users segmented by seconds of page load time. On the right, the panel visualizes percentage of users segmented by seconds of page load time. Unfortunately, in the tenant available to me, only page load times less than 1 second were present. By using queries, I was able to adapt to intervals below 1 second to address this limitation appropriately. You would be correct to say panels are rather static as the segmentation by seconds is fixed within the query.

The right panel more directly answers the customer's exact question. It provides at-a-glance value through visualization of page loads time distribution that could be most relevant for prioritizing work to meet business SLAs.

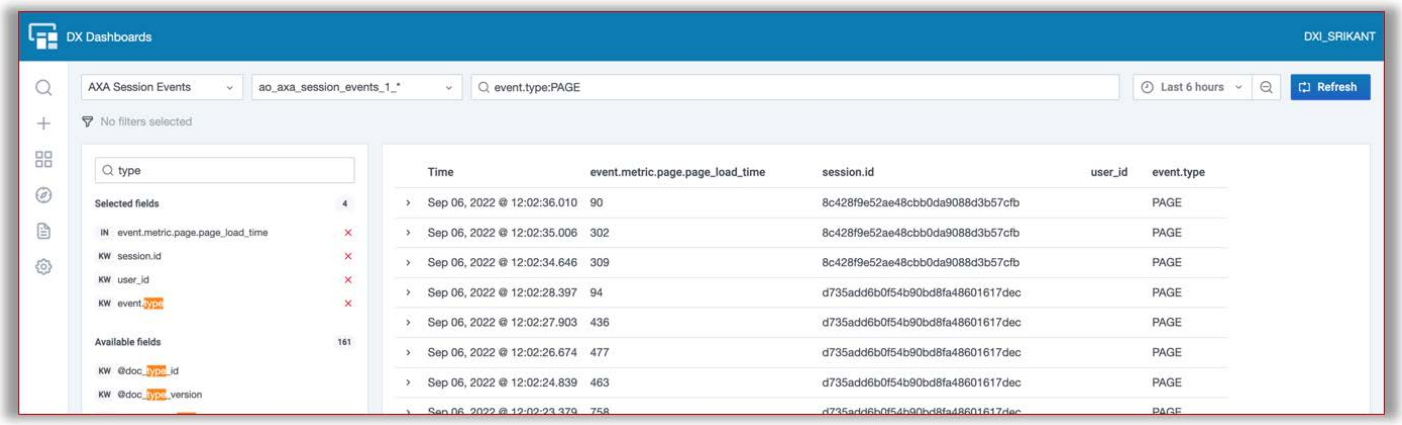
In addition to the percentage view, it would be helpful to display the number of users affected by degraded load times to help quantify the magnitude of impacted users. This is why I chose to retain and adapt the panel on the right.

3.3.1.1 Data from the Browser Agent and Its Availability in DX Dashboards

Browser Agent data is available within DX Dashboards as an elastic data source *AXA Session Events*. Elastic data sources, because they are flexibly structured as opposed to statically structured, use different terminology. A data record is a document. Thus, AXA Session Events has one document for every browser event that the Browser Agent captures. Page load time is captured for PAGE events on page load completion together with constituent timings of DNS lookup time, time to first byte, and so on. See more details in the following sections[1]. There are several other event types. Some examples of event types include, AJAX and SESSION. I will briefly discuss these event types later in the document.

You can easily inspect the contents of AXA Session Events using the Discover feature of DX Dashboards. I recommend you check the availability of browser data and test your (Lucene) queries to confirm you are able to fetch your desired data.

Figure 3: Example AXA Session Events



3.3.1.2 Left Panel: Page Load Time Versus Number of Users

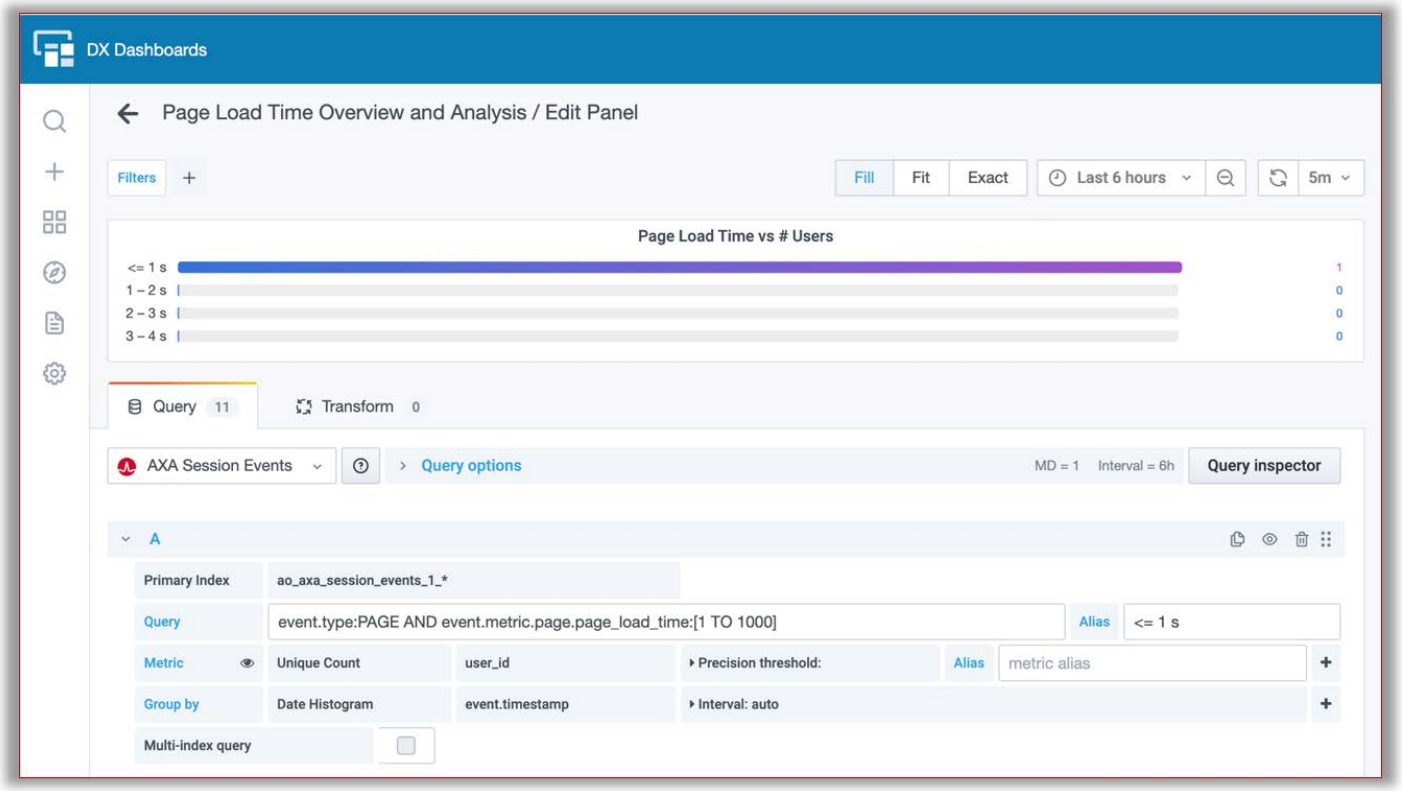
Action: Initially, create folder, dashboard, and the first panel

1. Create a folder called **Browser User Experience**.
2. In that folder, create a new dashboard named: **Page Load Time Overview, Insight, Analysis, and SLA Compliance**
3. Click the **Add Panel** icon. It is the left-most icon among the top-left set of action icons.
4. To edit the panel, click near the new panel's header and select **Edit** in the drop-down box that appears.
5. Select the panel's visualization as **Bar gauge**; which is Grafana lingo for bar chart.
6. In the **Panel Settings** expansion set **Panel Title** to: **Page Load Time vs # Users**

1. As per the navigation timing model and API of W3: www.w3.org/TR/navigation-timing/

The following figure shows the query A (auto-named, editable) for the left panel:

Figure 4: Example Query A



Action: Create a query to fetch user counts for intervals of page load time

1. In the drop-down box below the **Query** tab, select the data source for all queries to be: **AXA Session Events**
2. Enter the query[2] as: **event.type:PAGE AND event.metric.page.page_load_time:[1 TO 1000]**
   This query will fetch all documents with PAGE events with page load times between 0 and 1000 ms (numerically and inclusive boundary values).
3. Enter the alias as: **<= 1 s**
   This alias will label the resulting data series appropriately.
4. Select the metric (computation) from the drop-down box as: **Unique Count**
5. Select the data source field as: **Count user_id**[3]
   This data source will count the number of distinct user_id values within the fetched data

2. An error in the Lucene query syntax or in the field name will render no result and therefore no row in the bar chart and no resulting data series. You may use the Discover feature to develop and test your queries - know that Lucene queries are case sensitive.
3. You may use regular expressions in the metric field name for searching fields, such as, **ajax.*error**, to search for a field name with **ajax** and **error** - and know that selection is only applied when you click a field name in the drop-down box (only typing in a field name doesn't suffice).

6. Leave the **Group By** field as: **Date Histogram**

   This value is the default setting to obtain a data series result applicable for visualization with a x-axis of time. Here we are not utilizing that; we use a bar chart without a time dimension.)

This query A provides the count of distinct users for load times 0 – 1000 ms. This query provides data for the first row of the bar chart.

Before continuing with the remainder of the rows, there are two things we must consider to obtain a correct result:

1. DX Dashboards work on time series data for a given time window or time range. This time range is selectable in the upper right corner of a dashboard.

   Within that window, data is divided into sub-intervals[4] by default. These sub-intervals are either given with the query or defaulted to *auto,* in which case the query builder and the data source agrees on an interval.

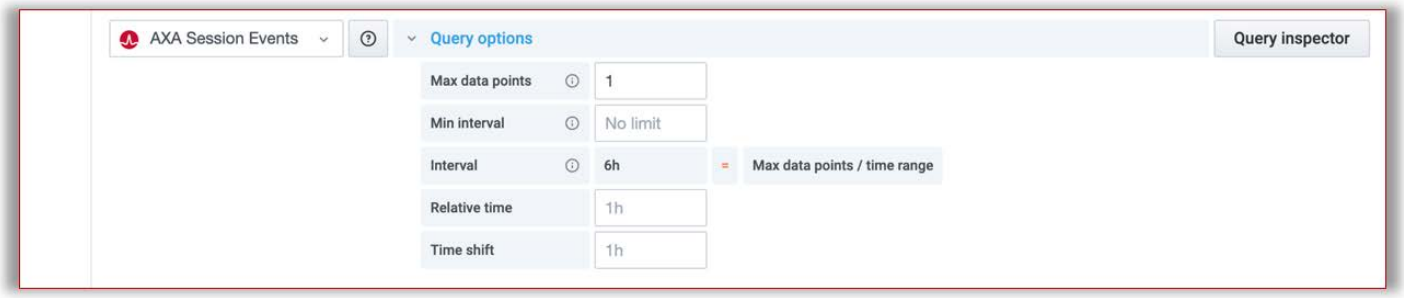
   For the current query, we want the number of distinct users for the entire window, and so we want a single interval only. The interval is selected as shown in the following action:

   Action: Set the query options to have all the panel's queries apply a single interval

   a. Expand **Query options** directly to the right of the AXA Session Events data source. This step indicates that just as the selected data source applies to all queries, so do query options. Setting the **Max data points** option to **1** sets to the number of intervals to **1** for all queries.

   b. This option makes the interval range adjust automatically so that its range matches the window time range.

Figure 5: Example Query Options



   c. *The setting of **Max data points** to* **1** *applies to all panels in this paper (except where noted).*

2. As DX Dashboards works on data series, a query will always return at least two data points[5]. Setting max data points to *1* will actually return two data points: the value for the current window and the value for the previous window.

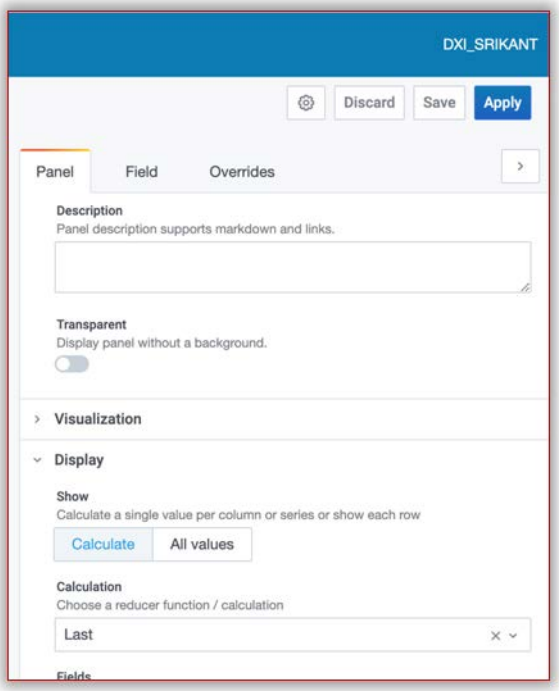
4. Thus, in this. paper, I carefully use (time) window and (time) interval to clearly differentiate the two.
5. DX Dashboards generally adjusts the number of data points returned to be agreeable with the data source resolution-wise as intervals can never be smaller than what the data source provides. Nor can intervals be split although the **Max data points** does not reflect that adjustment to make it apparent.

Action: Set Panel options to use Last (= *current*) value of all data series fetched by queries

a. To visualize the value for the current window, select: **Last**

This selection applies to all results of queries.

Figure 6: Example of Last Calculation



Action: Create queries for the remainder of bar chart rows

b. Duplicate the first query for every query for the remainder of the bar chart rows by clicking the query duplicate button available (double-document icon to the right in like of the query *A* name). Nine duplicates all together.

c. Change these duplicated queries to use the following respective load time ranges and aliases:

[1001 TO 2000] - with alias **1 - 2 s**

[2001 TO 3000] - with alias **2 - 3 s**

…

[9001 TO 1000] - with alias **9 - 10 s**

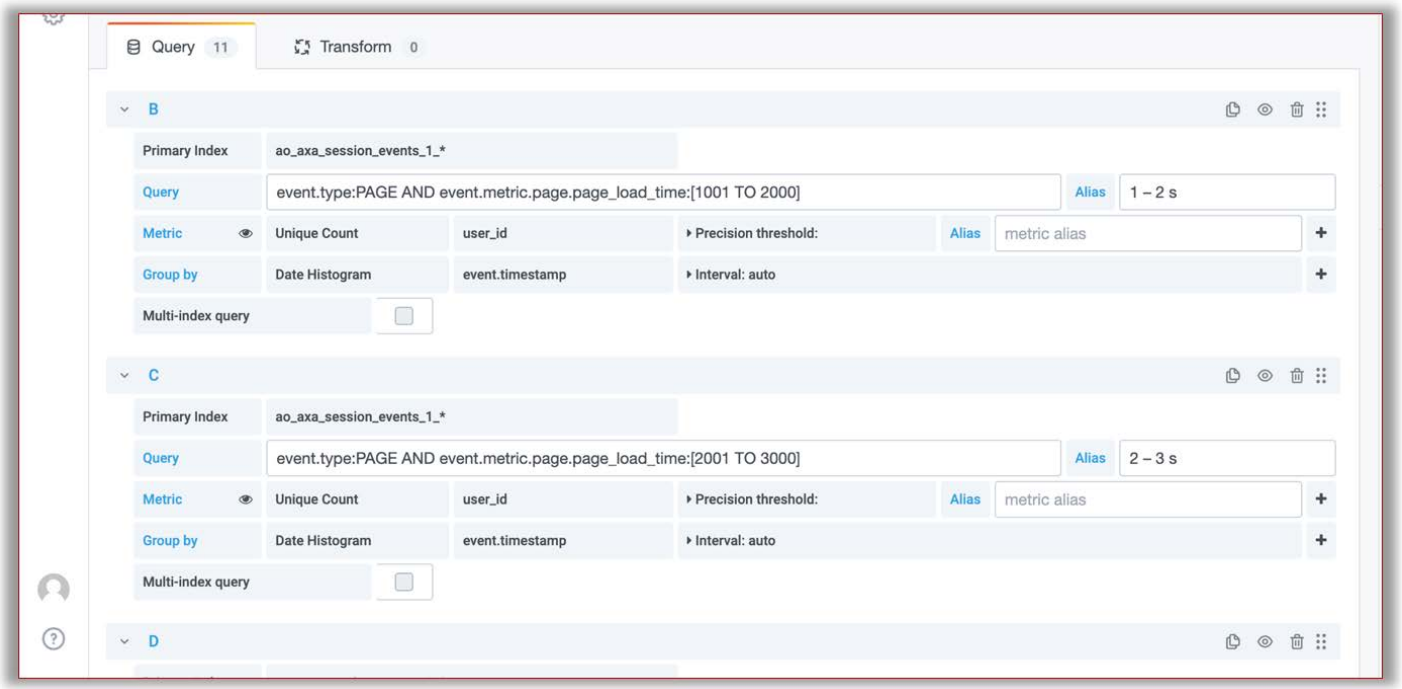
[10001 to *] - with alias **> 10 s**

This step provides the remainder of bar chart rows for the panel.

Yes, bar chart rows do visualize data series ~ a query result, per bar chart row, also known as a bucket.

The aliases given here will be used in transformations later to refer to individual query results.

Figure 7: Example Query for Bar Chart Row



This results in 11 queries, A through K. Each query produces a data series with two data points as per max data points. Yes, that was set to **1** and the DX Dashboards properly returned two. This resulted in the page load time buckets for intervals of one second for page load time as per the user's question.

As mentioned in the panel calculation, we set **Last** as the calculation to use the last value which is the value for the current time window.

Done: This finishes the left panel.

3.3.1.3 Right Panel: Page Load Time Versus Percent Users

To save time creating the right panel, I used the panel just constructed as a template.

TIP: Constructing panels in an order that lends itself to duplication is a general technique that you should always consider when building dashboards. When you want to visualize different metrics in similar manners, construct one dashboard fully, duplicate it, and then edit it. Use generic naming and features to lessen or even eliminate editing after duplication. This practice will greatly speed up dashboard creation and result in a consistent look.

TIP: You can use any existing panel as a template. To do so, click near the header of a panel you want to use as a template. Under **More**, select **Copy** in the expandable drop-down box.

The copied panel can now be pasted into any dashboard, not just the current dashboard, by opening the desired target dashboard. Click the **Add Panel** icon and select **Paste Copied Panel**. The panel will be added to the top of the opened dashboard and can then be moved to its desired position. The **Get Started** folder contains numerous examples which can be used as customizable templates. To help you get started, all OOTB dashboards are candidates for customization as well.

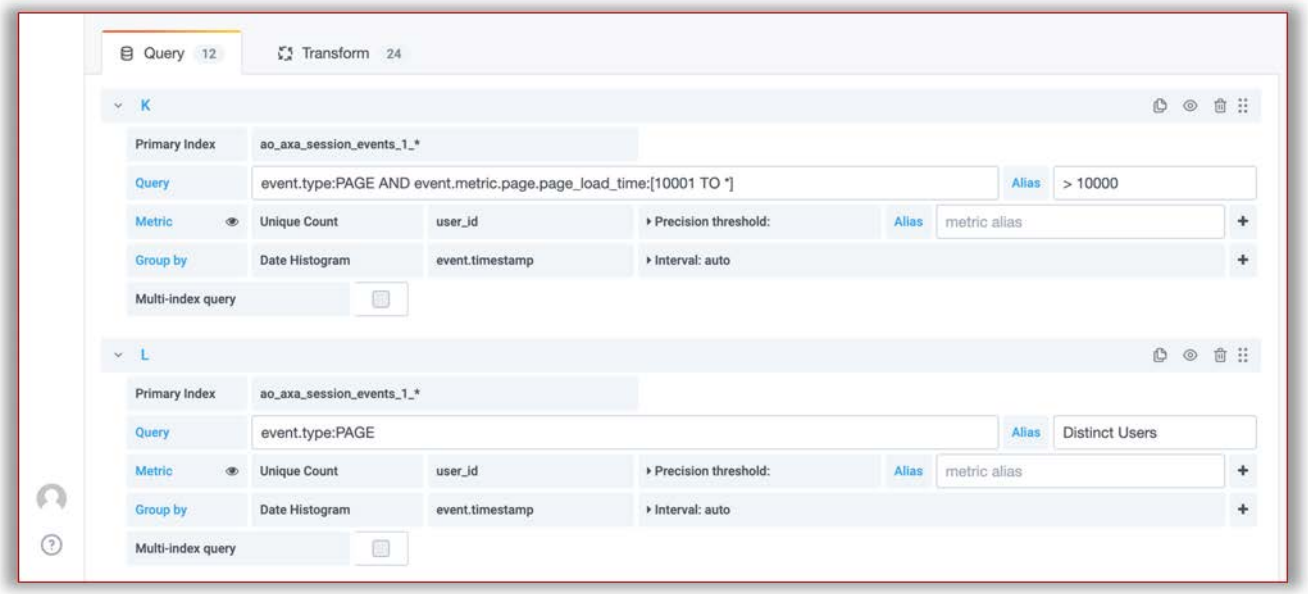
Action: Fetch count of distinct users

1. Duplicate the left panel by clicking near the panel header, and select **Duplicate** under **More** in the appearing, expandable drop-down box.
2. In the **Panel Settings** set the **Panel Title** to: **Page Load Time vs % of Users**

   The queries for right panel are the same as for the left panel. There is no need to modify the queries.

   Transformation will be used to add percentage calculations. However, for transformations to calculate percentages of users, we need the number of distinct users.
3. Add a12th *L* query, click: **+ Query**

Figure 8: Example Query Tab



4. Select all documents with PAGE events. Set **Query** to: **event.type:PAGE**
5. Set **Alias** to: **Distinct Users**
6. Set **Metric** to: **Unique Count - user_id**
7. Count the number of distinct users within the selected page events. Set **Group By** to: **Date Histogram**

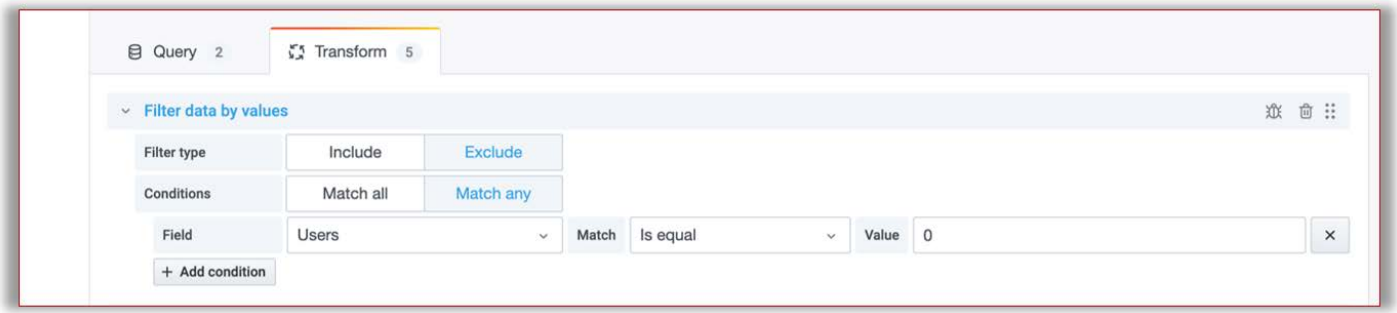
The calculation formula for user percentages per bucket generally is:

User% ~ users in bucket/total distinct users[6]

6. Percentages 0-1 may be shown 0-100% by selecting Unit as Percent(0-1) in panel visualization options

Potentially there might be time windows without activity or there may be page load events without a user_id. For these widows, total distinct users would be zero (the denominator) which would invalidate the calculation. To avoid this issue, we first apply a filter transformation to exclude fetched documents with zero users:

Figure 9: Example Filter Transform

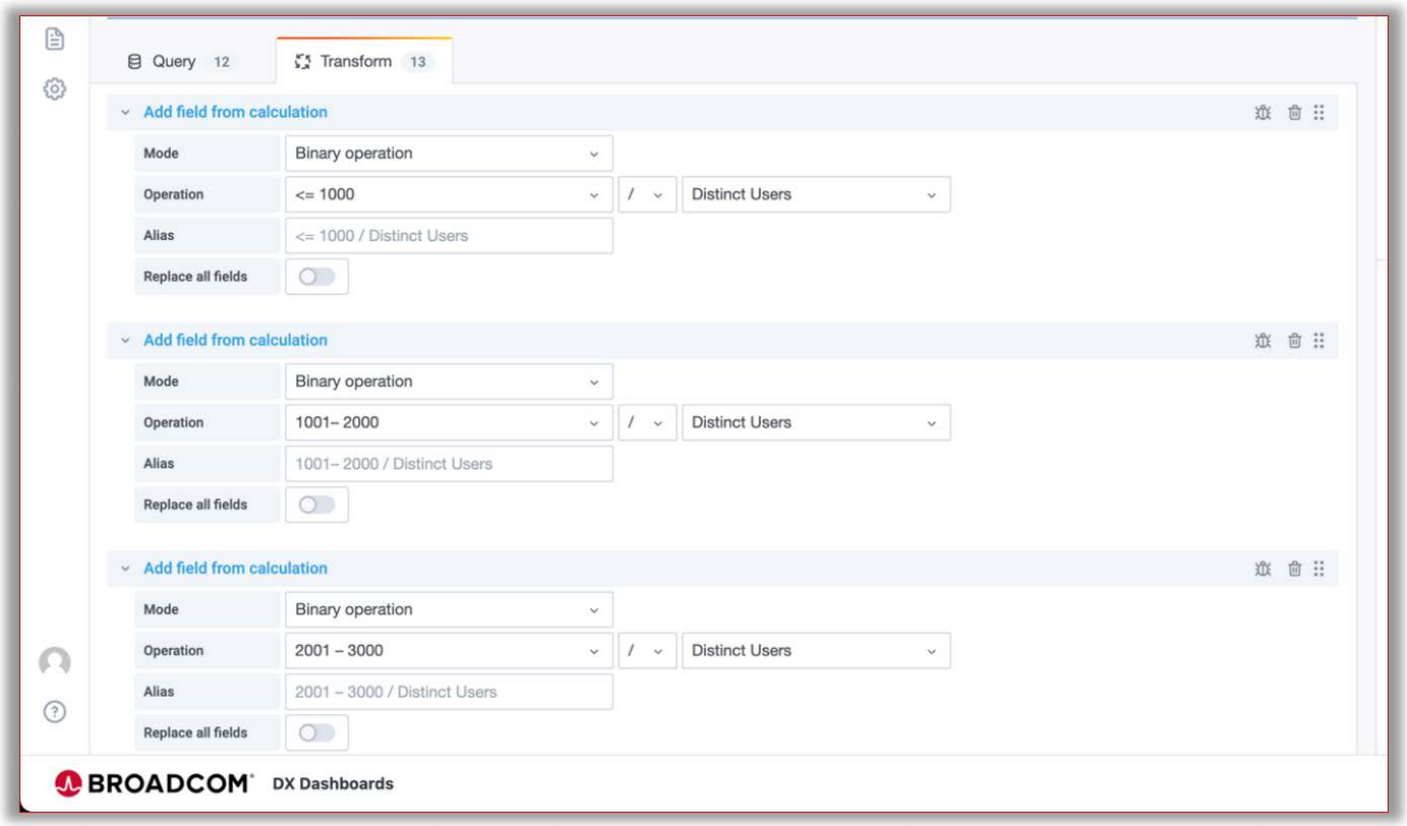


Action: Add a filter transform to exclude intervals without users

1. To the right of the **Query** tab, click the **Transform** tab.
2. Click **Add Transformation** and select a filter transform.
3. **Filter type**, click: **Exclude**
4. **Conditions**, click: **Match Any**
   It does not matter here, as there is one condition (any condition or all conditions met).
5. Select **Field** as: **Distinct Users**
6. Select **Match** as: **is Equal**
7. Set **Value** to: **0**

Next, to do the calculation for every row of user counts (~ every data series bucket) we use **Add field from calculations** transformations to divide user counts by the count of distinct users:

Figure 10: Example Add Field From Calculations



Action: Create percent of users for bar chart rows using transformations

1. Click **Add Transformation** and select **Add field from calculation.**
2. Select in the drop-down box its **Mode**: **Binary Operation**
3. Select the data series from the drop-down box of aliases used for queries: **< 1000**
4. Select its operation as: */*(division)
   Division is selected in the drop-down list of available operators.
5. Select from the drop-down box of available data series its operand: **Distinct User**
6. Do not set its alias.
   Do not fill in **Alias**. You will use default naming (as implied by the grayed-out visible text).

Once you have added the first transformation, it adds the new data series that you use as the *operand* for the second transformation.

This process is repeated for all 11 data series, referring to series using aliases from queries, as seen in the previous figure:

1000 - 2000

2000 - 3000

…

> 10000

Lastly, adapt the visualization of data series as rows of the bar chart.

Action: Use an *Organize Fields* transformation to hide intermediate results

1. Hide all intermediate results by disabling the eye icon for the rows of query results.
2. Hide the **Users** field.
3. Data series reflecting the percentages are renamed appropriately:

   <= 1 s

   1 - 2 s

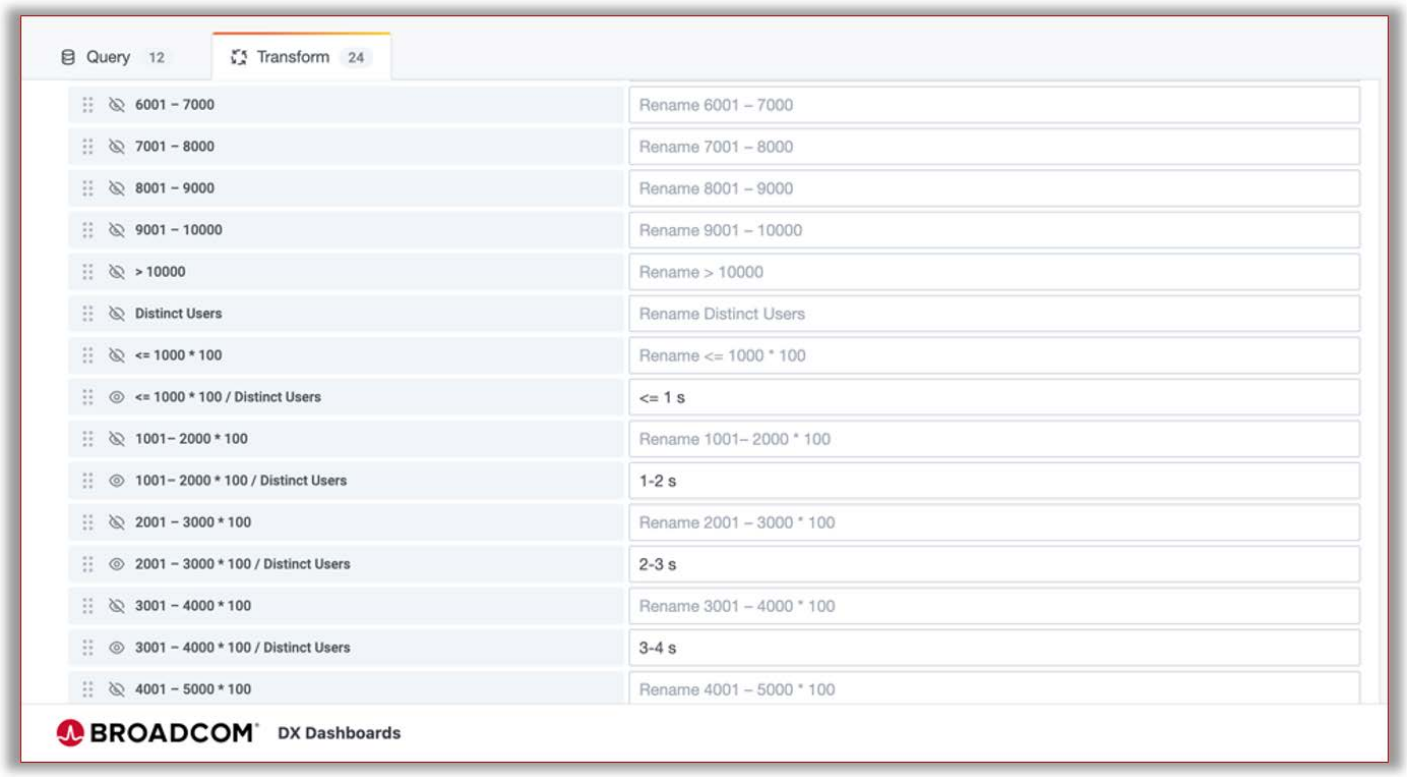
   …

   > 10 s

   You can see renaming take effect in the bar chart text as you go along.

After adjusting panel decimals to 0 (as greater precision does not make sense), the right panel is complete.

Figure 11: Example Hide Intermediate Results



Done: This finishes the right panel.

3.3.2 Question 2: Page Load Time by Percentiles

DX Dashboards has a generic percentiles feature that allows you to simply specify the desired percentiles of 25, 50, 75, 99, 100:

Figure 12: Example Page Load Time by Percentiles

This panel visualizes the five requested percentiles producing five buckets of sizes 25%, 25%, 25%, 24%, and 1% of page load times per bucket respectively. The 25% percentile contains 25% of users, the 50% percentile contains an additionally 25%, the 75% additionally 25%, the 99% additionally 24%, and 100% additionally 1% (25% + 25% + 25% + 24% + 1% = 100%).

3.3.2.1 Page Events and Calculating Percentiles

At this point you must consider the PAGE event documents of the AXA Session Events data source. A page event happens every time a page is fully loaded. At that event, collected timings (page load time, DNS time, DOM time, time to first byte, and so on) captured within the browser (as per the W3 spec previously referenced) and written as fields to a document created and appended to the data AXA Session Events data source. The document created always has a session ID and may have a user ID.

Thus, when calculating page load time percentiles neither sessions nor users are considered; only page load time is considered. Therefore, this panel cannot answer the user's questions without visually correlating the percentiles with counts of users and sessions.

We can easily calculate 25%, 24%, and 1% of distinct sessions and users within the time window. The technique is similar to the one used to calculate user percentages in the previous section.

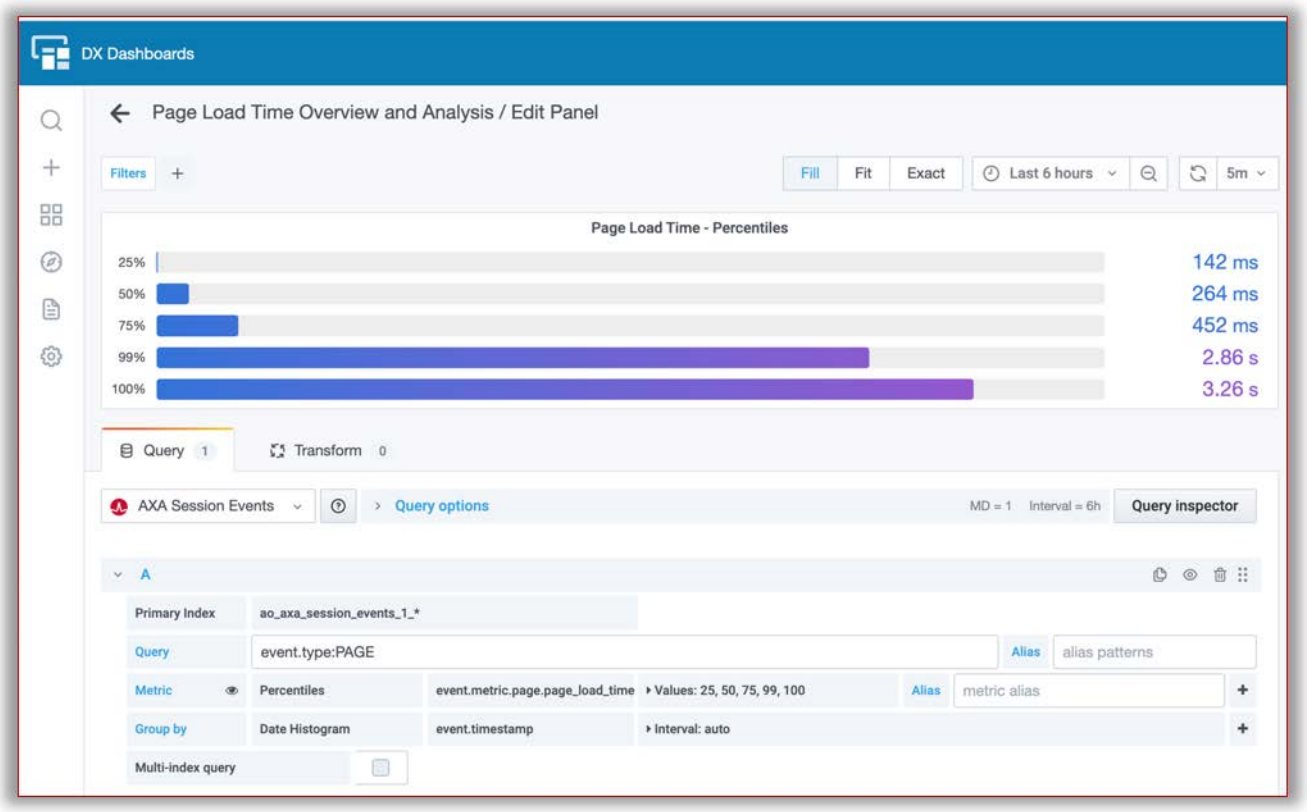
The additional steps needed are as follows:

1. Create percentiles panel, and 2a/b/c).
2. Create bucket counts for page hits, sessions, and users.
3. Visually correlate these buckets with percentages by juxtaposing the calculated values.

3.3.2.2 Page Load Time Percentiles Panel

Action: Building at the query

Figure 13: Example Page Load Time Percentiles Panel



1. The data source is still **AXA Session Events**.
2. Select all documents with PAGE events. Set Query to: **event.type:PAGE**
3. Do not set an Alias set as five data series will result (one per percentile).
4. Select **Metric** as: **Percentiles**
5. Select **Field** as: **event.metric.page.page_load_time**
6. Set **Values** to: **25, 50, 75, 99, 100**

   This setting obtains the five desired percentiles. Separate data series will result. Calculations are always per interval, even with 1 (one per percentile).

   To add or modify percentiles, say 95 and 97 for SLAs, you simply add them to the values list.
7. Group By: **Date Histogram**

As we duplicated the panel, Max Data Points are set to **1**, and panel calculation to **Last**.

By utilizing percentiles as metric aggregation, no further explicit calculations are necessary.

One advantage of using percentiles is that the resulting percentile values inherently adjusts to data values, following the definition of percentiles. In this respect, you can say the panel is dynamic whereas the initial panel design was static (making percentiles preferable regarding maintainability since adapting query ranges for every query is quite tedious and also prone to error).

The resulting data series are named after the specified metric with a prefix designating the percentile:

p25.0 event.metric.page.page_load_time

p50.0 event.metric.page.page_load_time

…

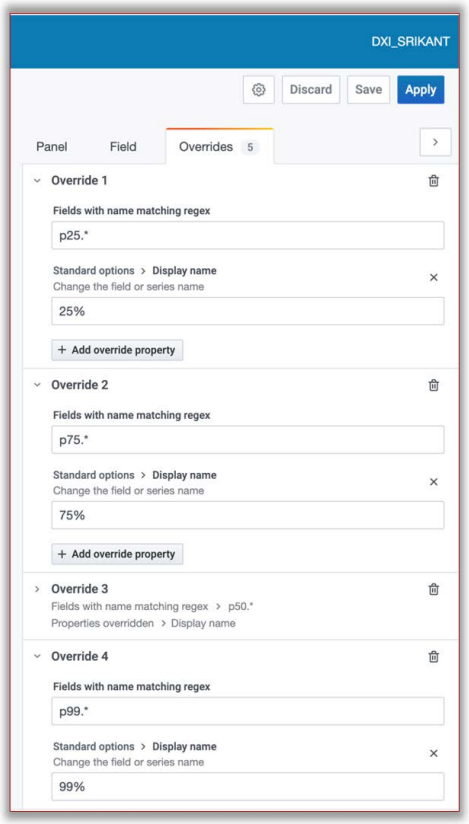
p100.0 event.metric.page.page_load_time

To make percentile texts appear to the left of the bars, we use bar chart panel overrides.

Action: Set 25%, 50%, 75%, 99%, and 100% to appear as texts for bar chart rows

1. In the panel click the **Overrides** tab.
2. Click **Add Override.**
3. Click **Add Override Property** and select **fields with name matching regex**: **pp25.***
   This override will match any field starting with p25
   The reason for using a regex will become clear below.
4. Repeat by creating another four overrides for **p50.*, p75.*, p99.*, p100.***.

Figure 14: Example Bar Chart Panel Overrides



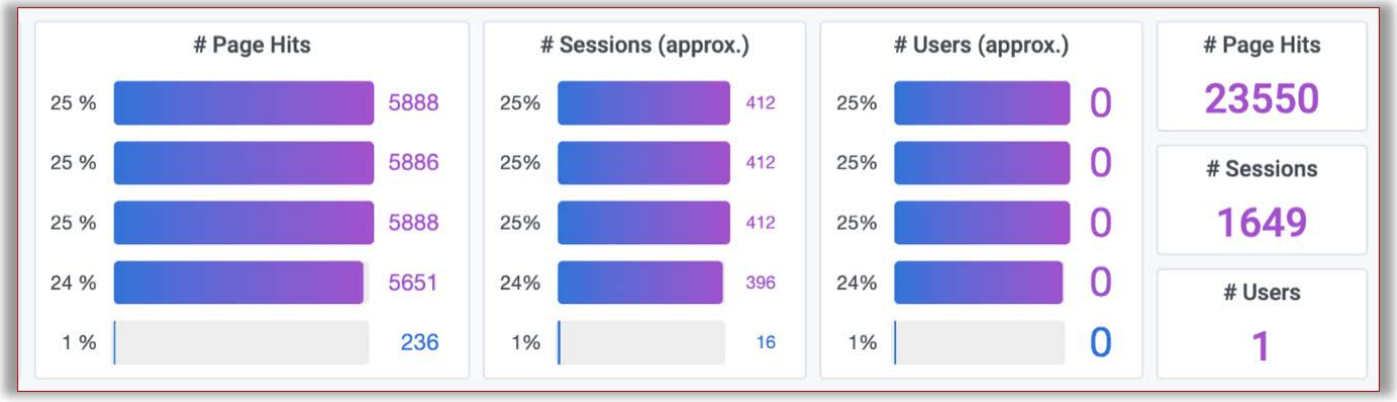
Done: This panel is now done.

3.3.3 Additional Panels for Counts of Page Hits, Sessions, and Users

The panel only visualizes the page load time percentile buckets. It does not yet show the number of page loads (also known as page hits), the number sessions, or the number of users which was a requirement.

I have positioned the additional panels needed to the right of the percentile panels:

Figure 15: Example Panels for Counts of Page Hits, Sessions, and Users

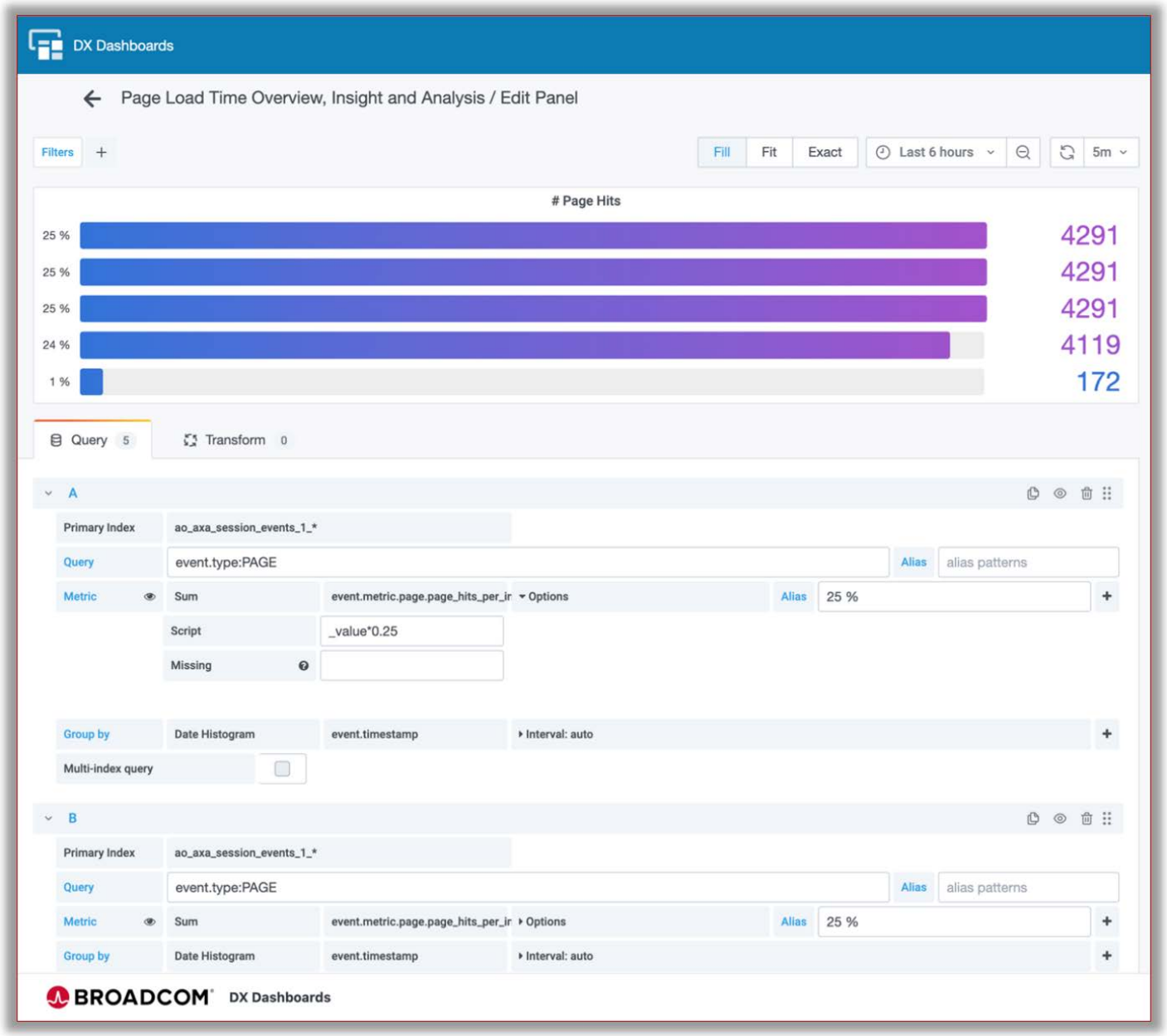


Since calculations for these additional panels are similar, create the # **Page Hits** panel first and use it as a template (duplicate and modify) for panels showing **# of Sessions** and **# of Users**.

The three smaller rightmost vertically stacked panels I will detail in the following section.

3.3.3.1 Page Hits Bucket Panel

Figure 16: Example Page Hits Bucket Panel



The query needed is similar to the query described above. Additionally, to obtain a 25% bucket, an Options script is used:

Action: Create percentile buckets for page hits, sessions, and users

1. Set **Query** to: **Event.type:PAGE**
2. Set **Script** by expanding **Options**: **_value*0.25**
   As **_value** refers to the value produced by the calculation, this option yields 25% of page hits.

3. Set **Alias** to: **25%**
   As three buckets of 25% are needed, this query is repeated identically three times.
4. Two additional queries are added using script snippets for **24% and 1%** respectively.

Done: The page hits panel is now created. Position it to the right of the percentile panels.

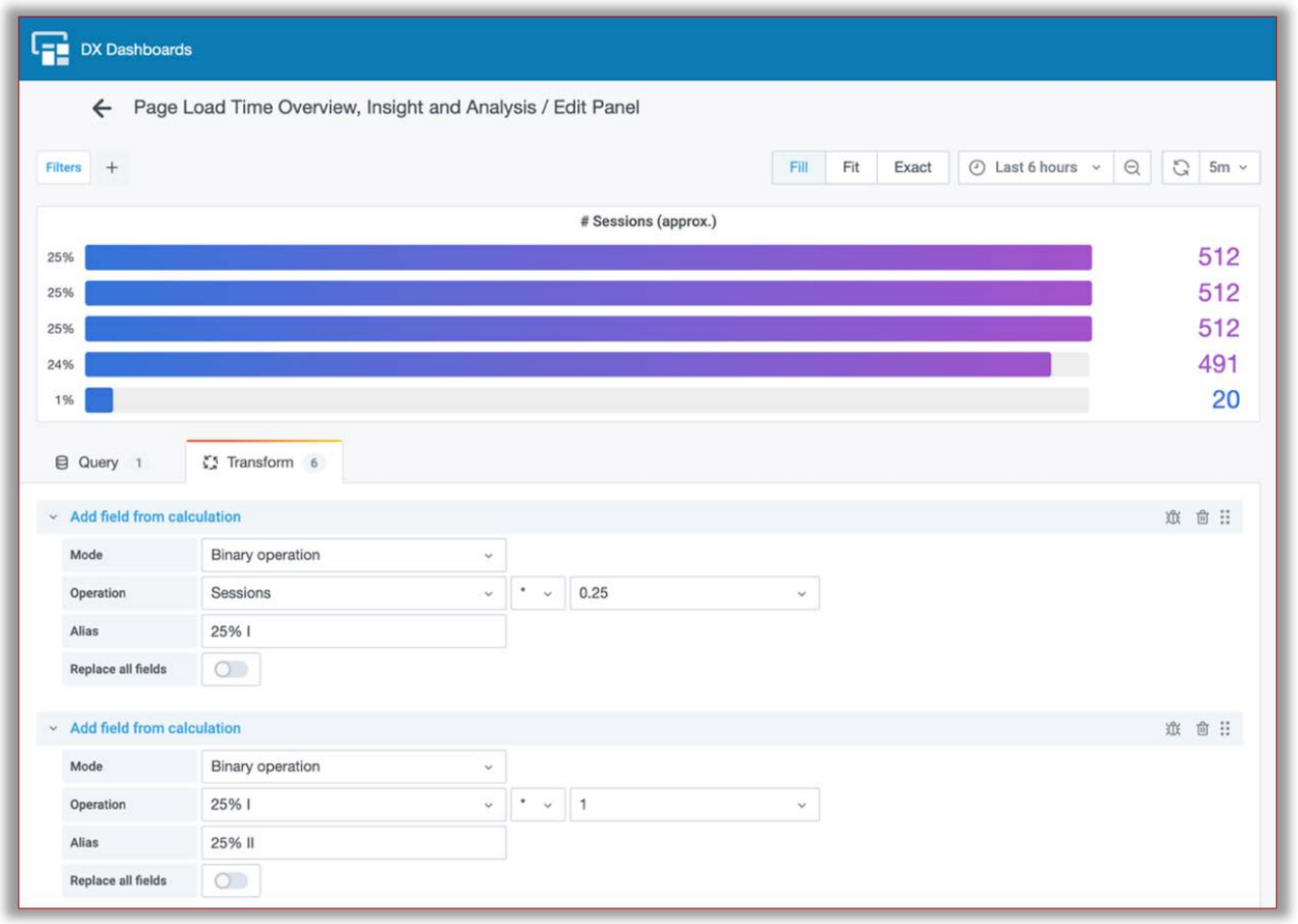
3.3.3.2 Number of Sessions and Number of Users Bucket Panels

The queries are similar but must use **unique count** to aggregate number of sessions and users respectively within the time window. Unfortunately, as option scripts are not available for **unique count**, transformations must be applied instead.

Action: Create query for user count within time window

1. Create a new panel.
2. Set **Panel Settings**, and **Panel Title** to: **Sessions**
3. Set **Query**: **event.type:PAGE**
4. Set **Metric**: **Unique Count, session_id**

Figure 17: Example Number of Sessions and Number of Users Bucket Panels



As per the screen shot above, calculations are similar to those explained previously with the one exception: Aliases must now be used differently to avoid duplicate calculations and aliases. Aliases 25% I, 25% II, and 25% III are used as shown in the following section.

Action: Transformations to calculate percentage buckets

1. Add an **Add field from calculation** transformation.
2. Set its fields as follows:
   Mode: **Binary operation**
   Field: **sessions**
   Operator: *
   Operand: **0.25**
   Alias: **25% I**
3. Add an **Add field from calculation** transformation.
4. Set its fields as follows:
   Mode: **Binary operation**
   Field: **25% I**
   Operator: ***
   Operand: **0.25**
   Alias: **25% II**
5. Repeat this, adjusted to create **25% III.**
6. Repeat the top one, adjusted to create **24%.**
7. Repeat the top one, adjusted to create **1%.**
8. Add an **Organize fields** transformation.
9. Set **Aliases**: **25%, 25%, 24%, 24%, and 1%** - respectively (duplicates are OK here).
10. Hide **Sessions**.

Done: The **Sessions Panel** is now created. Position it to the right of the **Hits** panel.

Action: Create user count buckets panel

1. Duplicate the sessions buckets panel just created.
2. Exchange **session.id** for **user_id** in the query.
3. Change **Alias** to **Users.**
4. Change **Transformations** to use **Users** instead of **Sessions.**
5. Rename fields as before in the **Organize fields** transformation.

Done: This creates the user count bucket panel. Position this panel to the right of the **Session Count** buckets panel.

3.3.3.3 Panels for User Counts, Session Counts, and Page Hits

The three smaller stacked panels on the right all have queries like their bucket counterparts. Instead of visualizing using a bar chart, a **Stat** panel is used for the single values.

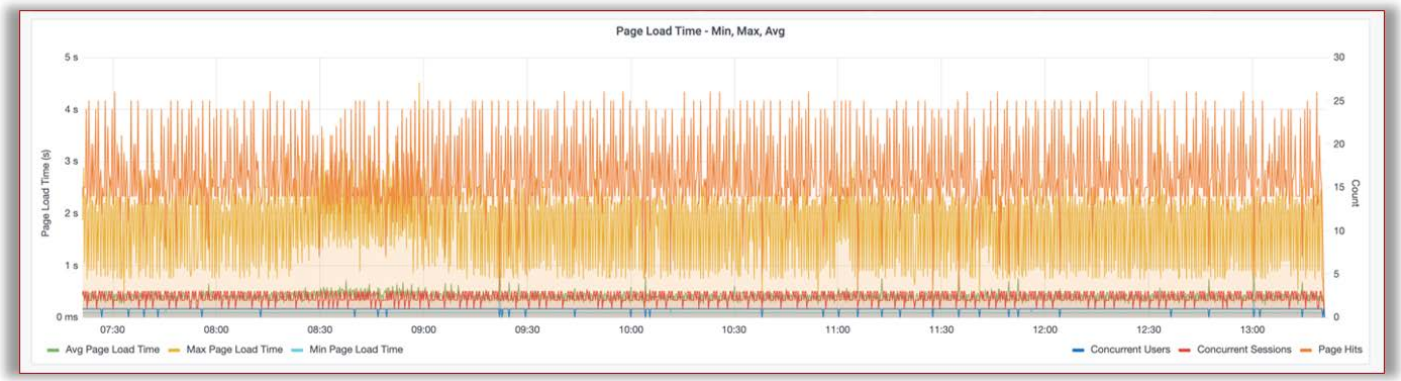
Action: Create page hit, session, and user count panels

1. For the page hits count panel duplicate the page hits bucket panel.
2. For the session count panel duplicate the session counts bucket panel.
3. For the user count panel duplicate the user count buckets panel.
4. Delete all transformation in the three panels.
5. Resize and position the panels.

3.3.4 Panel for Page Load Time: Minimum, Average, and Maximum

To address the use case requirements regarding timing of a degradation, a graph panel showing key data over time is needed.

Figure 18: Example Page Load Time - Minimum, Average, and Maximum



The panel contains six queries, constructed similarly to those detailed previously.

Action: Create Page Load Time, Avg, Min, and Max panel

1. Query, six times: **Event.type:PAGE**
2. Metrics settings respectively:
   Average, event.metric.page.page_load_time, "Avg page load time"
   Minimum, event.metric.page.page_load_time, "Min page load time"
   Maximum, event.metric.page.page_load_time, "Max page load time"
   Count unique, user_id, "Users"
   Count unique, session.id, "Sessions"
   Count, -, "Page Hits"
3. Clear **Max Data Points** in the query options.
   This panel is intended to visualize metrics over time.
4. Use the panel overrides to set the three last counts to the right Y-axis.

With its six key data visualized correlated together, this panel will reveal rising trends for page load time including maximum. Also, like all panels, it allows zooming in on a time range of the panel to determine the beginning and ending of a degradation. Exactly as requested by business users.

This panel is also a good example of the single-pane-of glass concept mentioned previously. It will also assist APM users by clearly showing them when or if activity levels and trends are related to degradation. The panel gives valuable indications for further analysis. Exactly as requested by APM users.

TIP: A graph panel may be zoomed time-wise by mouse click-drag-release within the graph, to reset the window time frame for the entire dashboard. Thus, using the graphed values to guide visually zooming in on a period of degradation for details for analysis.

Expandable dashboard rows will be added in the following section to further support the APM users' use case.

3.3.5 Additional Panel for Sessions by Page Load Time by Seconds

Above the two left panels, the user count and the user percentages buckets panels of the following figure were detailed.

Figure 19: Example Page Load Time: Minimum, Average, and Maximum

Now, given that AXA Session Data may not contain a user ID for all session data page events[7]. I advise to reveal the entire user experience, that additional sessions panels visualizing absolute and relative distributions of sessions are added as they are shown in the two left panels in the previous figure.

NOTE: Obviously, understanding your data is crucial in interpreting it correctly. As said, all page events may not contain a user_id. With an Elastic data source this means that whenever a query references a field of a document that does not exist in that document or is null, that field is ignored for the calculation. As every document reflects an event, in this case PAGE events, this means that PAGE events without user_id are effectively ignored (and the option to set an actual value for null is not viable). Thus, not all sessions are considered and therefore the panel's resulting visualization does not reveal the entire user experience. Not addressing this issue, by including further data, could lead to misinterpretations. Please refer to Dashboard Design Principles for more information.

Action: Create sessions panels for user experience

1. Duplicate the user counts and percent panels to sessions count and percent
2. Exchange everything **user** to **session**:
   - **Query**: Change Session.id to **user_id**
   - **Alias**: Change Users to **Sessions**
   - **Transforms**: Change Users to **Sessions**

7. Use the investigator mentioned above to readily investigate this for certainty.

3.3.6 Deciding between Percentiles and Second Distribution Panels

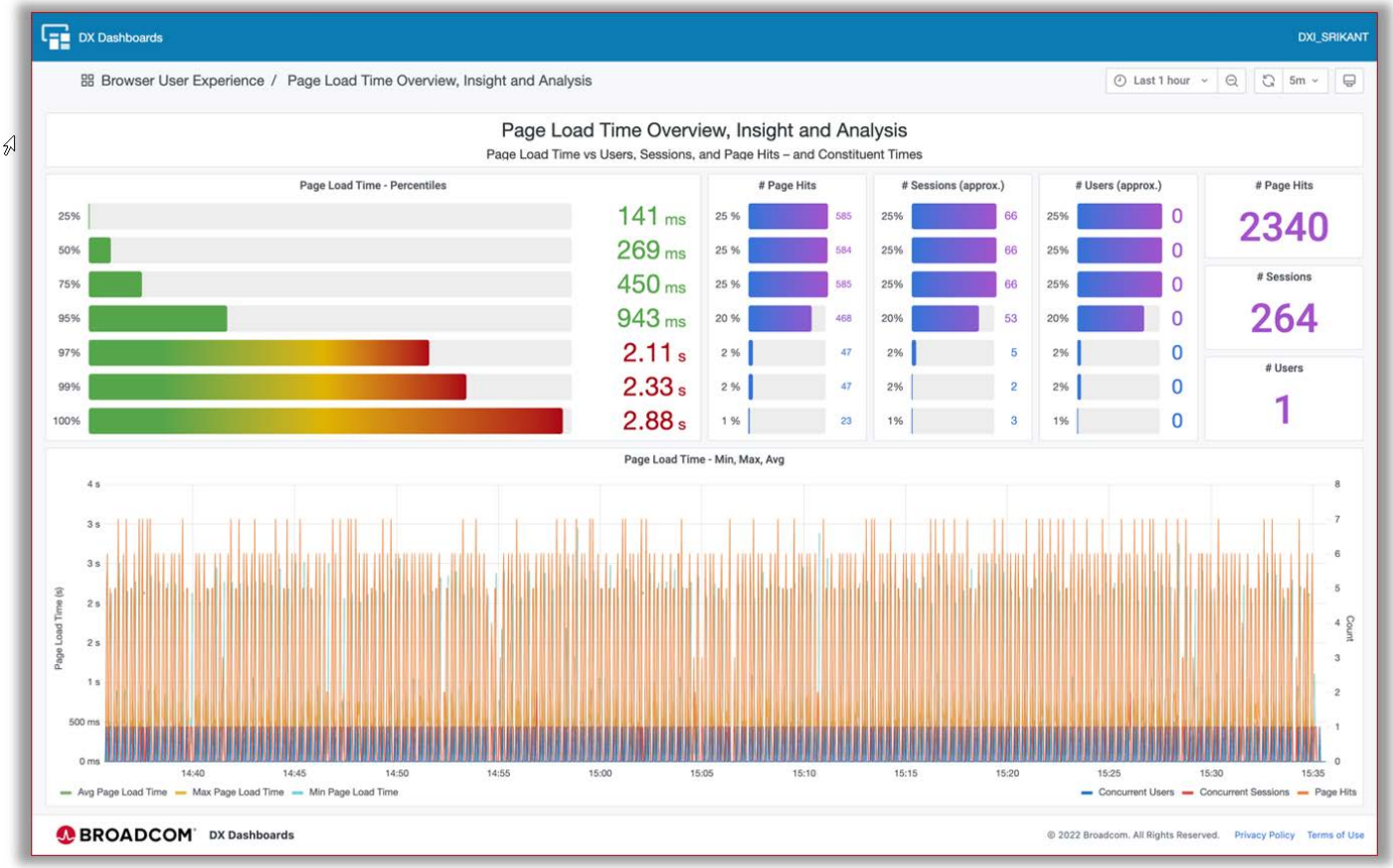
As you have possibly noticed, the panel's design so far visualizes almost the same information, page load time percentiles by percentiles respectively by second distribution. Is this relevant? To decide on that I offer by a few design principles. For more information, see Dashboard Design Principles.

These principles are especially and increasingly important when dashboards support less experienced APM users. Designing dashboards for these users will force superior dashboard design. Pay attention to and actively seek feedback from less experienced users and from non-APM personas.

Let us apply these principles to test our design. I would argue that the percentiles panel can be enhanced to be preferable to the percentiles' distribution panel for the business personas:

- Percentiles by their nature adapt to data (page load time - levels).
- Percentiles by their nature reveal data (page load time - densities).
- Adding 95 and 97 percentiles makes it emphasize experience versus activity quite ideally together with the bucket panels for page hits, # sessions, and # users.
- It readily highlights threshold violations and thus draws attention to any degradation.
- In real time or historically, it reveals violations of user experience percentile SLAs.

Figure 20: Example Page Load Time Overview, Insight and Analysis



This panel still does not answer the question of distribution by second intervals. Thus second distribution panels will be retained but moved to an expandable row. Thus, this detail will be visible on toggling and further adapted and optimized as shown in the following section.

3.3.6.1 Adaption and Optimization of Seconds Distribution Information

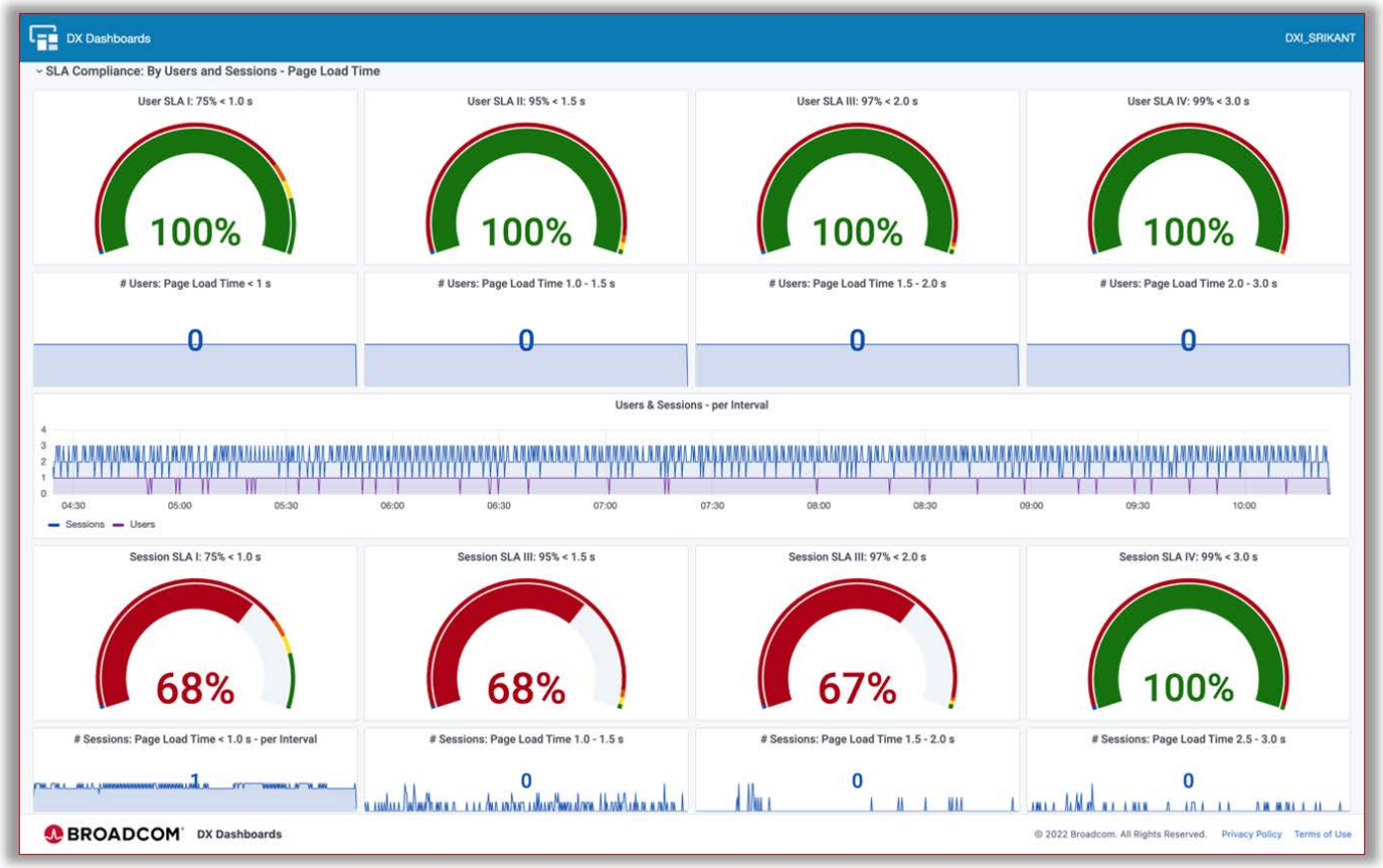
Seconds distribution for live monitoring could appropriately be used for highlighting SLA status. The SLA boundaries for user and session experience requirements can for example be reflected as follows:

- 75% < 1.0 s
- 95% < 1.5 s
- 97% < 2.0 s
- 99% < 3.0 s

Their visualizations could be enhanced and violation made prominent by showing the achieved percentages for example with color coding such as green, yellow, orange, red, and purple. The colors appear as per seriousness for the call to action as deemed by use cases. For example, the agreed color scheme might be yellow is be alert, orange is root cause analysis, orange is engage specialist and manager, red is decisive action required, purple is inform the business and customers. Note that the color scheme could be associated with actions and procedures and therefore included in use cases.

Below, **Gauge** and **Stat** panels are used instead of **Bar Chart** panels with the degradation highlighted by this color-coding. The **Gauge** panel dials visualize SLA compliance and the **Stat** panels quantify the number of users impacted.

Figure 21: Example Color Visualizations

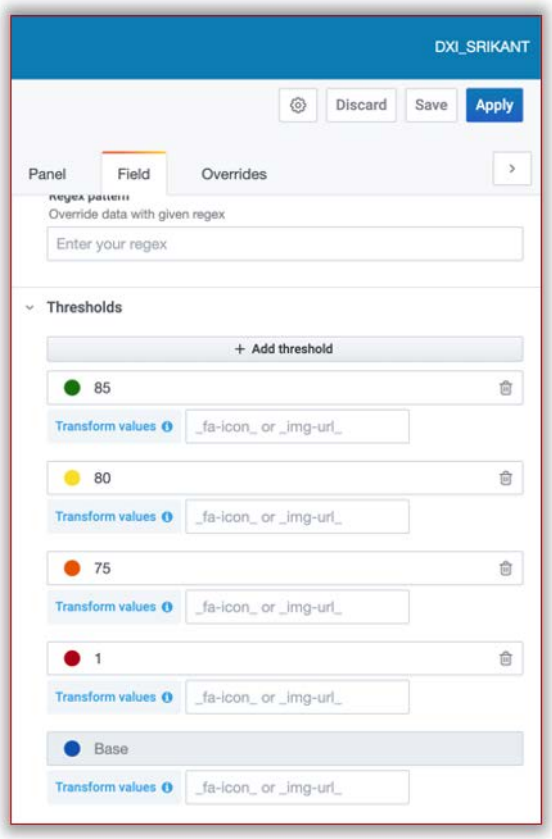


Action: Create User SLA Gauge by seconds panels

SLAs assumed are: 75% < 1.0 s, 95% < 1.5 s, 97% < 2 s, and 99% < 3 s

1. Duplicate the **User%** bucket panel, select its visualization as **Gauge**, and set its title **User SLA I: 75% < 1.0 s.**
2. Delete all queries except two: the one covering [1 TO 1000] and the Users.
3. Rename the [1 to 1000] query to **SLA** making it easy to duplicate as-is and use as a template.
4. Change **Transform** to use **SLA.**
5. Set color-code thresholds at: **85 - green, 80: yellow, 75 orange, 1 red, base - blue**
   The reason to use blue for base is that a zero value is neutral (no activity).

Figure 22: Example User SLA Gauge by Seconds



6. Duplicate this **Gauge** panel three times and adapt it for the remaining 95%, 97%, and 99% SLAs.

Action: Create User buckets Stat by seconds panels

1. Duplicate the **User Count** panel, select visualization as **Stat** and set its title **User with Page Load Time < 1.0 s.**
2. Clear **Query Options Max Data Points.**
3. Set **Panels Calculation** to *Last* (not null).
4. Delete all queries except the [1 TO 1000] and set its **Alias**: **SLA**

5. Delete all transforms.
6. Set the threshold base color to **blue** (any number of users is neutral).
7. Delete all overrides.
8. Duplicate this panel three times:
   – Adapt query ranges: **[1000 TO 1500], [1500 TO 200], [2000 TO 3000]**
   – Adapt titles

The Stat panels are used without Max Data points to graph over time and quantify the number of users in the respective SLA range throughout the time window such as the number of users with that experience (that is, users impacted when there is a degradation).

TIP: It is a feature of the **Stat** panel to also graph values when Graph Mode is Area. This adds context as to any trend of the shown value. This reduces the risk of misinterpretation.

The stat number itself shows the last non-null value.

Similarly, the bottom half of Gauges and Stats panels visualize the SLA status regarding sessions (as per previous discussion of missing user_ids).

Action: Create Session SLA Gauge and Stat by seconds panels

1. For each panel duplicate the corresponding **User** panel.
2. Exchange **User** for **Session** in title and queries.

Adding the middle Graph panel provides interval context by showing users and sessions counts to help reduce risk of misinterpretation of Gauges and Stats by indicating magnitude of impact.

By adapting and optimizing how information is visualized, value is increased, and value and interpretation improved. In this case, the right design decision will depend on whether business user SLAs are based on percentile SLAs or second SLAs. The final design should reflect this.

3.3.7 Panels for the APM User

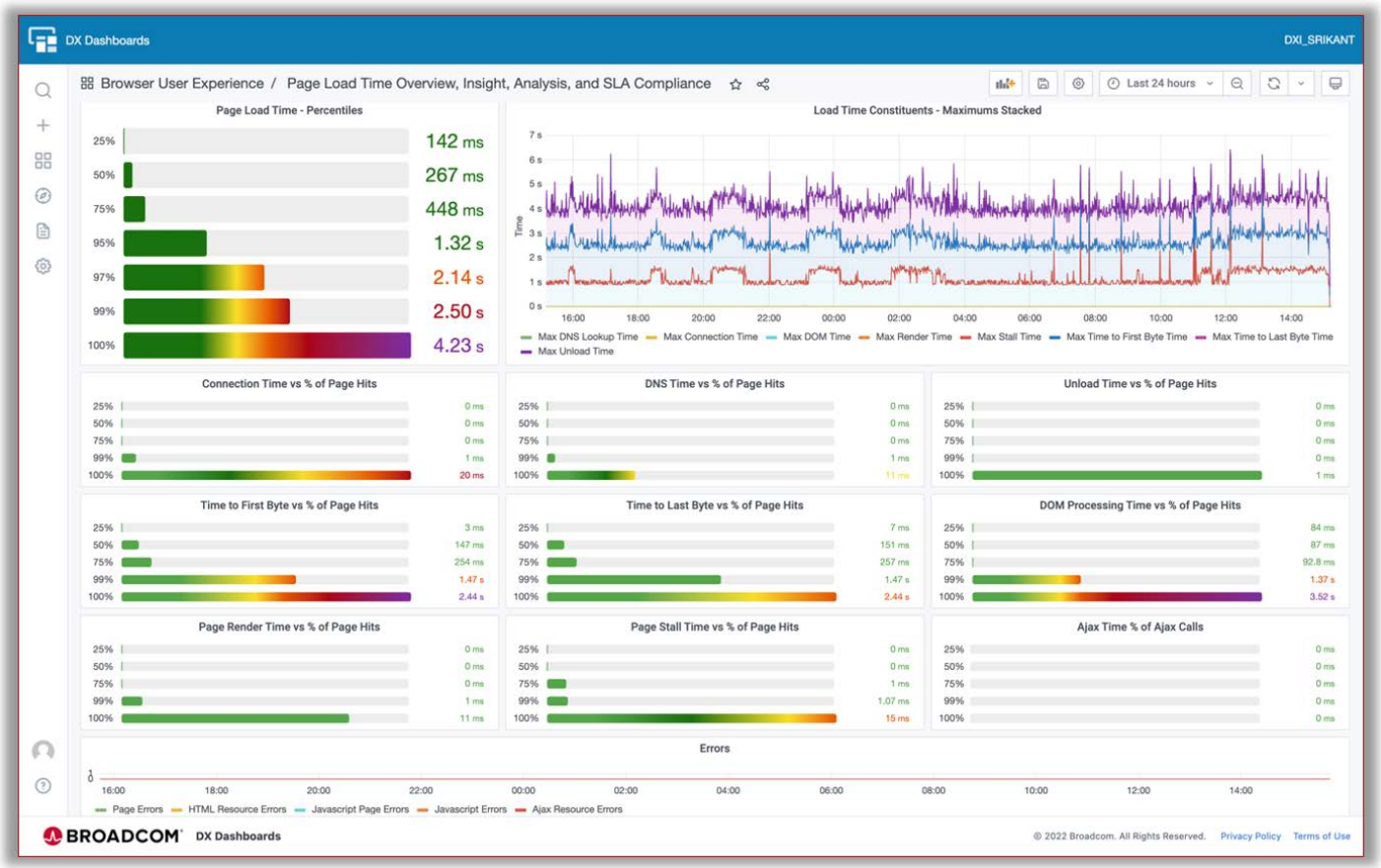
3.3.7.1 Adding Additional Expandable Rows for Insight, Analysis, and AXA Correlation

The panels detailed so far mainly cover the business user's use case. For the APM user's use cases, we can use expandable rows to ensure that the main panel is includes only the necessary information regarding a call to action: page load time degradation. Expandable rows are very well suited for that.

3.3.7.2 Expandable Row: Page Load Time Constituent Times – Percentiles

Possibly the first question the APM user will face when there is a degradation is, “Why?” The following panel is intended to provide context for users seeking the possible cause:

Figure 23: Example Page Load Time Constituent Times – Percentiles



In this dashboard, the percentages feature is used in the nine lower panels to show metrics that constitute the source of the page load time. For additional context, Ajax times are shown in the lower-right panel although the data reflects out-of-band asynchronous times.

A few key points:

- The calculated percentage values adjust to the level of each metric (such as, page load time in seconds, connection tome is milliseconds) is very advantageous.
- Because regex's were used for overrides, only the metric name of the query needs to be changed after duplicating panels.
- Color-coding highlights excessive contributions on a per metric basis.
- In the upper-right panel, constituent maximum times are stacked for highlighting main contributors. Note that the maximums metric is used to avoid hiding culprit outliers.
- For completeness, the bottom row of panels includes all types of errors visualized in a Graph panel.

This dashboard closely adheres to AXA's data interpretation for locating the source and root cause:

- Excessive unload time points to the Web application and heavy page design.
- Excessive DNS time points to DNS problems.
- Excessive connection time points to network routing and stability problems.
- Excessive time to first byte points to the serving back-end application.
- Excessive DOM processing points to too complex web page design.
- Excessive rendering time points to a resource constrained web client.
- Excessive Time to last byte points to network bandwidth constraints.

Thus, directly guiding to the next analysis step. Obviously, these points directly refer to use cases and indeed reveals when a degradation is resolved.

Actions: Create expandable row with panels for page load time constituents

1. Duplicate the percentiles panel previously created and use as-is positioned upper left.
2. Duplicate this nine times and adapt each for the respective metrics:

   Unload time, DNS time, connection time, time to first byte, DOM processing time, rendering time, and Time to last byte. Simply modify the metric name to reflect these values
3. Adapt panel color-coding threshold values to levels appropriate for each metric, adhering to call-to-action coloring: yellow, orange, red, and purple to highlight nearing, very near, and excessive contributions.

To add informative context to these constituent panels, a Graph panel visualizing maximum of all constituent times in a stacked manner per interval. This information is added to the right of the percentiles panel as shown in the following section.

Action: Create a Graph panel showing max constituent times per interval

1. Add a panel to the dashboard and set its visualization to **Graph**.
2. Add eight queries: **event.type:PAGE**
3. Select Maximum for each query and for all respective constituent times.
4. Set aliases as **max xxx.**

   Where **xxx** is the constituent time name.
5. For the **Panel** enable **Stack**.
6. Set **Sort** order **Decreasing**.

   For descending sort on values in the graph hover pop-up window.

This panel helps users quickly identify suspicious trends and outliers. If there are suspicious items, users can zoom in time-wise to view additional information.

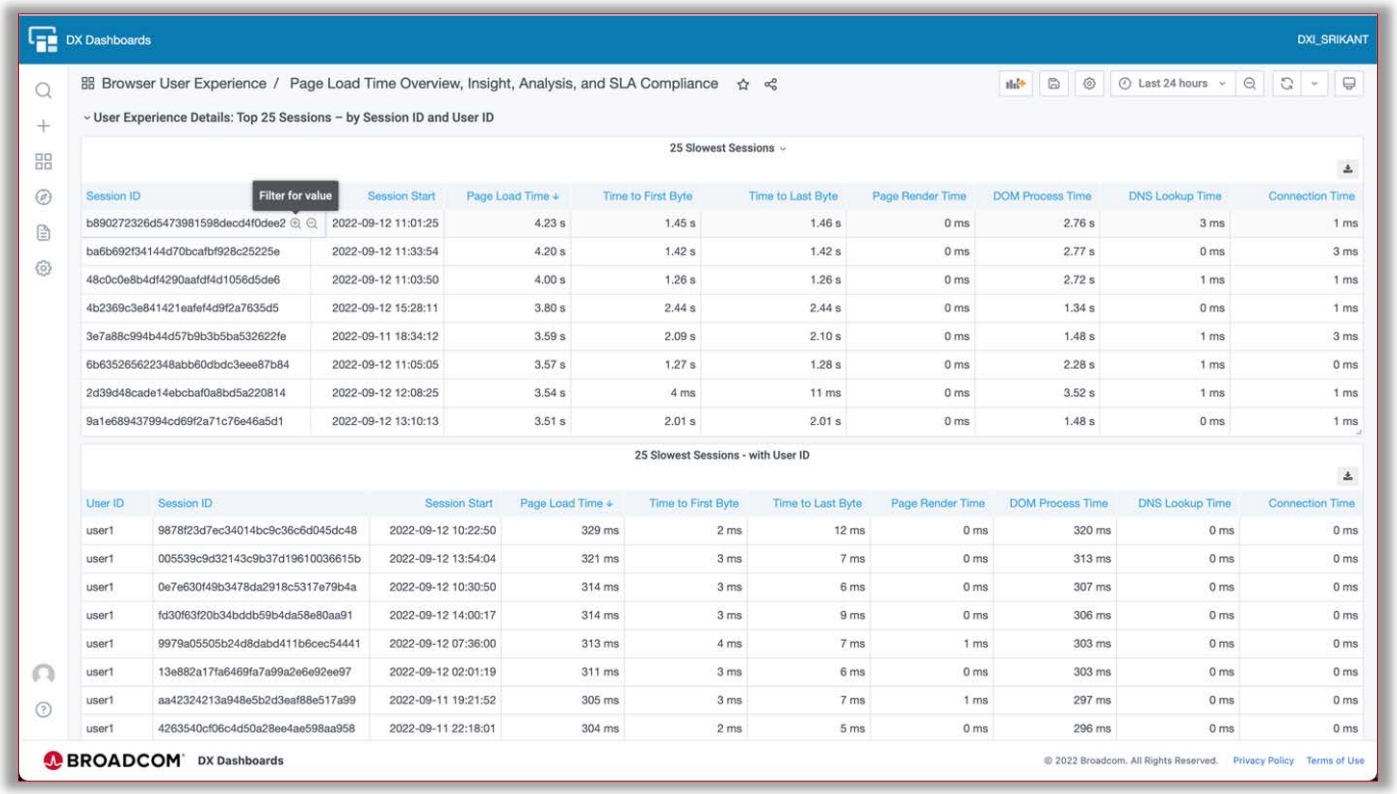
3.3.7.3 Expandable Row: Load Times for Top 25 Sessions and Users

The last expandable row contains two Table panels. These panels provide the following functions:

- Linkage to enhance AXA root cause analysis.
  - By providing a time frame with session and user ID, APM users can begin user session analysis by viewing waterfall diagrams for resource-loading and navigation into APM for back-end server application analysis utilizing hyperlinked transaction trace analysis.
- Filtering for focusing all panels' data on a single user or session.
  - Click the small circled **+** next to a session ID of a row.

- Filtering will activate at the top of the dashboard.
- All panels' queries will be augmented with a condition on the selected session ID resulting in all visible data to pertain to that selected session.

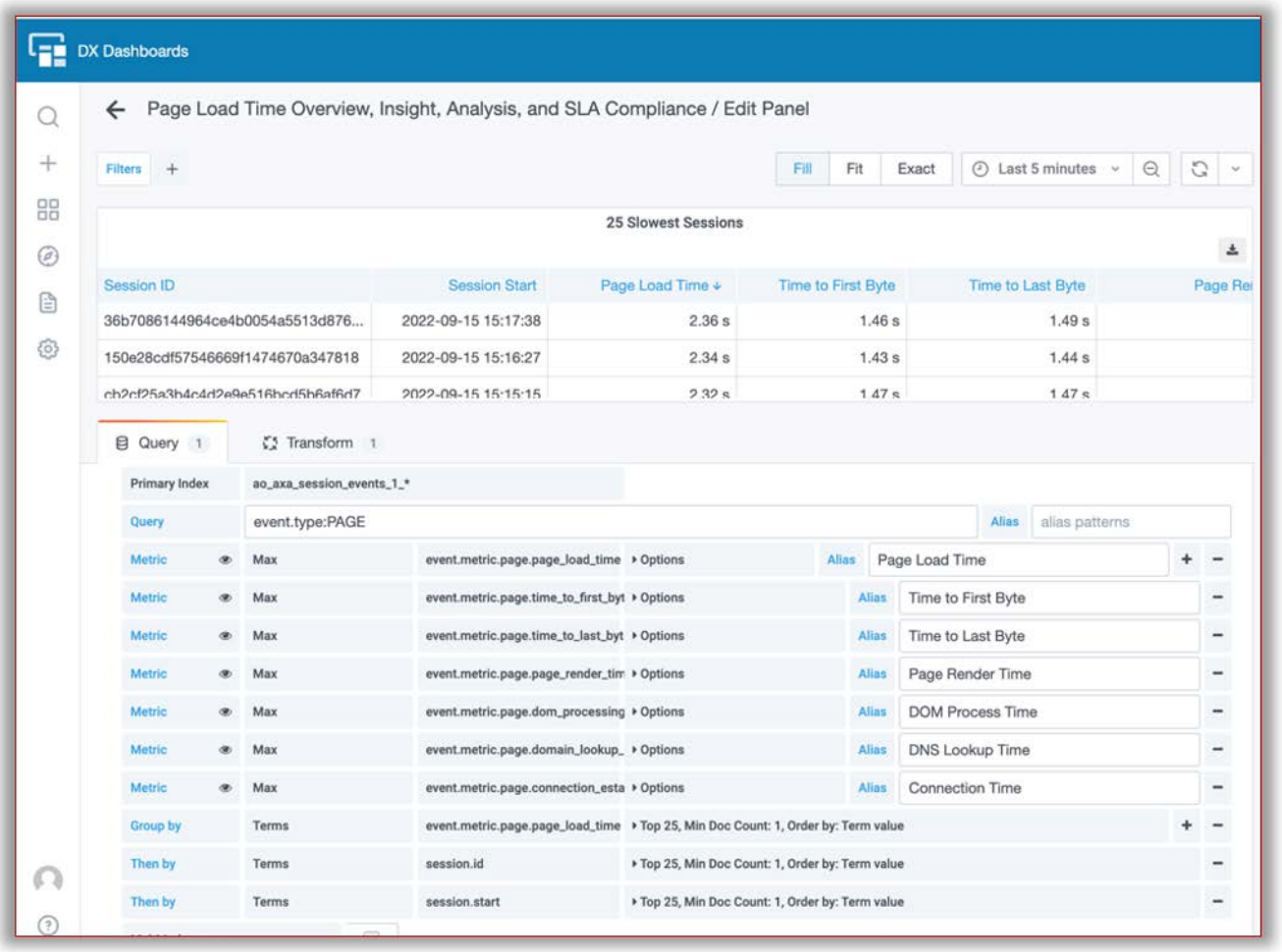
Figure 24: Example Load Times for Top 25 Sessions and Users



Action: Create top 25 table panels with session ID and start time

1. Add a new expandable row.
2. Set its title **User Experience Details: Top 25 Sessions - by Session ID**
3. Expand it.
4. Add a new panel and set its visualization to **Table**.
5. Position it underneath the just added expandable row.
6. Set query: **event.type:PAGE**
7. Add eight Metrics, one per constituent metric, as: **Max, event.metric.page.page_load_time, "Page Load Time"**
8. Add group-by terms to obtain sorted top 25 table on the page load time (first).
   - Group by: **Event.metric.page.page_load_time, Top 25**
   - Then by: **session_id, Top 25, Order by Term value**
   - Then by: **session_start, Top 25, Order by Term Value**

Figure 25: Example with Session ID and Start Time



Action: Create top 25 table panels with session ID, user ID and start time

1. Duplicate the panel just created and set title to include **User Id**.
2. Add **user_id** as a group by term to its query.

3.4 Know the Options

Lastly, here are two references to available out-of-the-box dashboards for analysis based on AXA Session Events in the DX Dashboards AXA Session Events folder:

- Performance Overview dashboard
- Session Events JS and AJAX Errors dashboard

3.5 Full Dashboard with Optimized Panels

Figure 26: Example of Final Browser User Experience, Overview, Insight and Analysis Dashboard

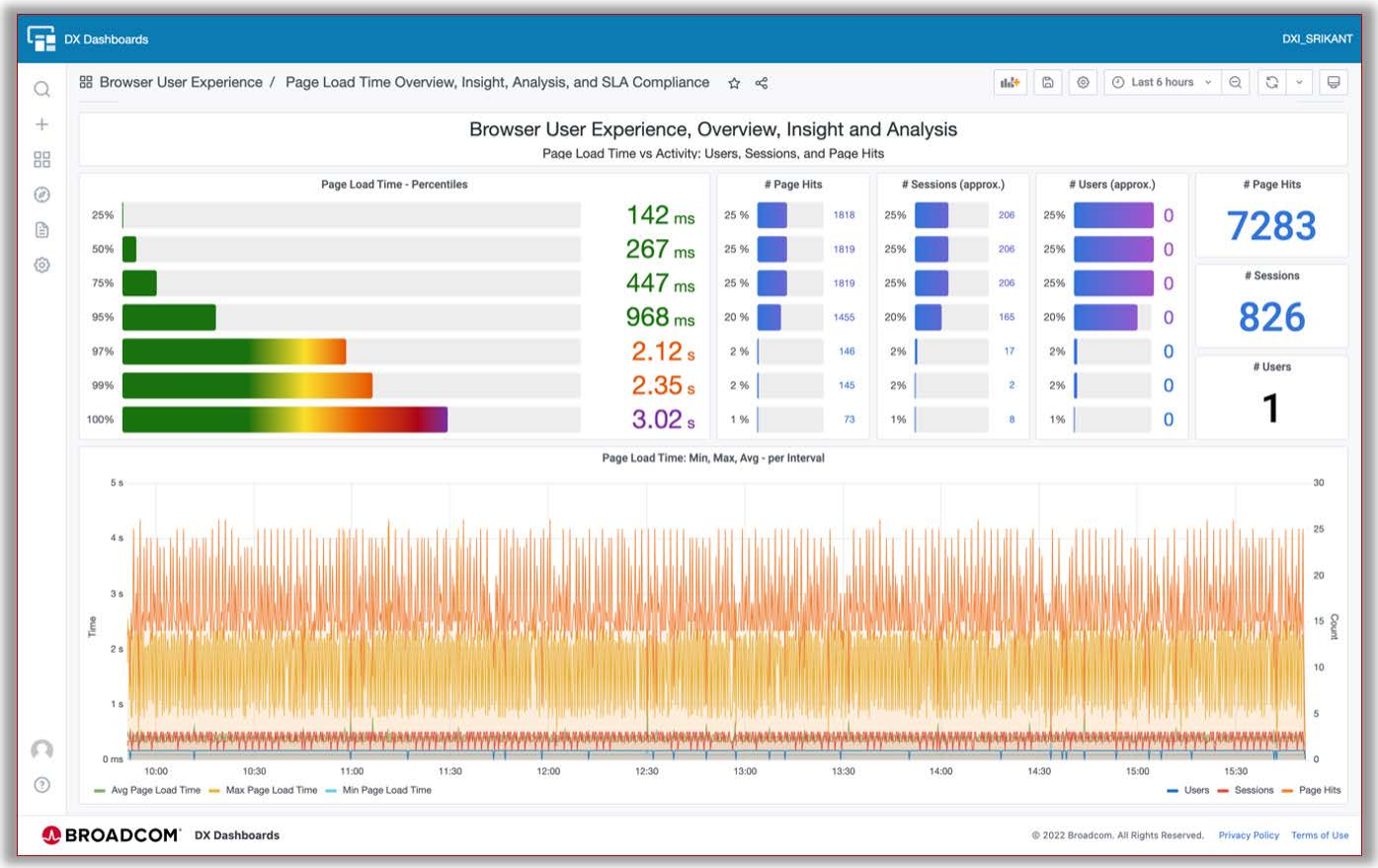


Figure 27: Example of Final SLA Compliance: By Users and Sessions - Page Load Time Dashboard

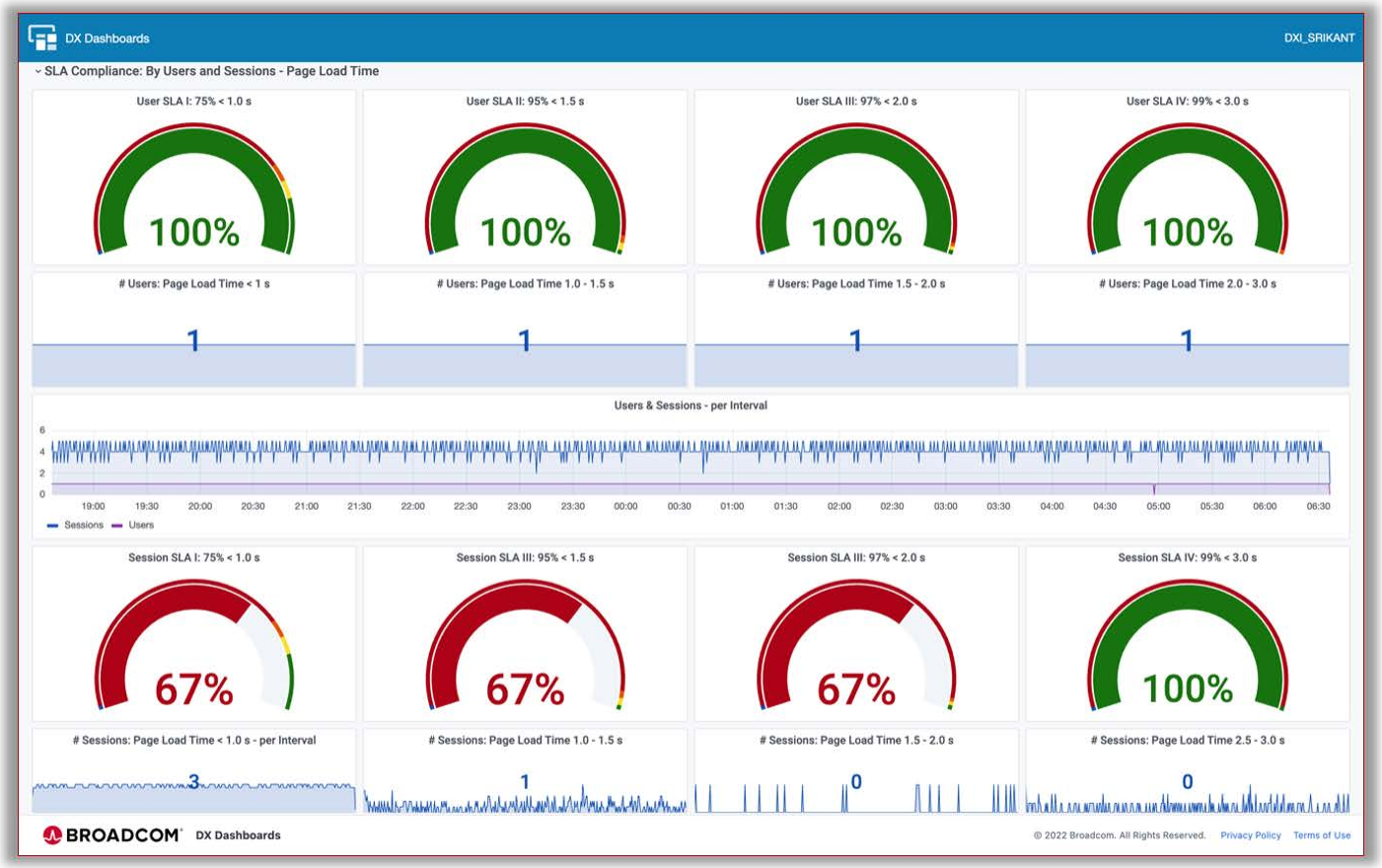


Figure 28: Example of Final User Experience Analysis: Page Load Time Constituent Times – Percentiles Dashboard

Now, I invite you to check the *single-pane of glass* concept against my final design as well as the design principles presented.

Author Profile

Henrik Nissen Ravn, Principal Engineering Solution Architect and APM Champion has worked with APM within Broadcom for the last 15 years. His 24 years of experience with our company spans DB2 for z/OS, data modeling, portal, security and APM software within Presales, Customer Lifecycle Solutions, SWAT, and Solution Engineering teams. An expert in helping customer teams master APM value, Henrik operates with a laser-like focus on increasing customer knowledge and satisfaction. He holds a master's degree in Computer Science from the University of Copenhagen.

About Broadcom® Software

Broadcom Software is a world leader in business-critical software that modernizes, optimizes, and protects the world's most complex hybrid environments. With its engineering-centered culture, Broadcom Software has an extensive portfolio of industry-leading infrastructure and security software, including AIOps, Cybersecurity, Value Stream Management, DevOps, Mainframe, and Payment Security. Our software portfolio enables scalability, agility, and security for the largest global companies in the world.

For more information, visit our website at: software.broadcom.com


BROADCOM
SOFTWARE