rittmanmead
A DATA AND ANALYTICS COMPANY

OBIEE Performance Analysis

ORACLE Gold Partner

OBIEE Performance Analysis

Company Name

Rittman Mead R&D

August 2015

Contents

Introduction

The performance of any user-facing computer system is a key aspect in its success and acceptance by users. In this age of technology, everyone is familiar with how fast systems *can* work ("Google Speed") and their expectations are set accordingly. This report has been created based on the actual usage data from your OBIEE system and identifies areas in which performance improvements could assist with user satisfaction and retention.

Method

OBIEE's **Usage Tracking** is the source for all data cited in this report, unless stated otherwise. Usage Tracking logs all queries executed by the BI Server including information about how long it took to run and the amount of data processed.

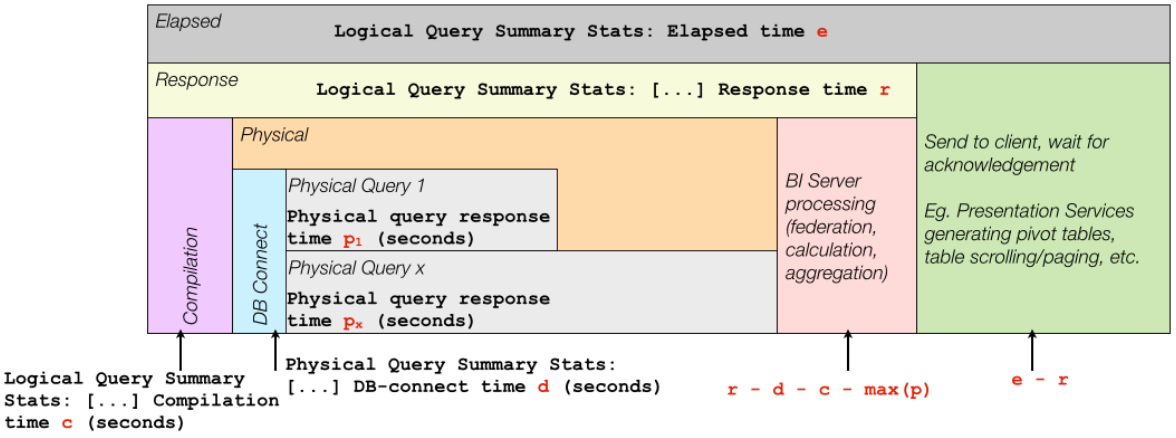
The concept of a **time profile** is key to identifying areas of performance that can be improved, and is detailed at length in Cary Millsap's paper, Thinking Clearly About Peformance[1]. Before "tuning" is done on a system, there must be an understanding of–for a given process such as running a report–*where* time is being spent.

Having identified where time is spent in a system, performance improvement work can be pinpointed in problem areas. This has three key benefits:

1. The limit to which performance can theoretically be improved is known in advance. For example, if a report takes 30 seconds to execute, 29 seconds of which is spent in the database, making changes to network configuration between the user and the web server is, at the very most, only going to be able to reduce the response time of the report to 29 seconds.
2. Work can be prioritised based on the feasibility and impact of performance changes with the constraints of time and financial budgets.
3. Without a time profile, work can only be targeted on a gut-instinct basis–a performance approach sometimes known as the "*drunk man anti-method*"[2]

The analysis in this report combines the general approach of a time profile based on Usage Tracking data with the specific and detailed understanding of the internal process flow within OBIEE.

[1] Millsap, Cary. "Thinking clearly about performance." Queue 8, no. 9 (2010): 10.
[2] Gregg, Brendan. "Monitorama 2015 Netflix Instance Analysis"



Note that the scope of an OBIEE report's execution is not covered in entirity by Usage Tracking data, but for the purposes of a general assessment it is more than sufficient to identify the areas for improvement and analysis. A comprehensive analysis of end-to-end reponse time is usually only undertaken for individual reports with performance issues that are not clearly identified from the Usage Tracking data alone (for example; network problems between the web server and user).

Only performance of reports in the `/shared` area of the Presentation Catalog are included in this report. It is generally a good idea for all reports to live in the `/shared` area in order to aid support and maintanance of the report catalogue.

An extended discussion of OBIEE performance analysis techniques can be found at http://ritt.md/obiee-performance.

Terminology

- A "Report" in the context of this document refers to an **OBIEE Dashboard Page**. A Dashboard Page is made up of one or more **Analyses** with optional prompts to filter the data displayed. We term this a Report because it is the grain at which most will interact with OBIEE.
- This is a bit of an oversimplification, since the values provided in filters, selection steps, etc can drastically change the data retrieved and the path taken, all within the "same" dashboard page.

Data

Data analysed is from OBIEE's Usage Tracking, which logs all requests that the BI Server processes.

The source data covers 2752 days, from 13 Aug 2007 to 24 Feb 2015.

Excluded Data

For the purposes of this analysis we have *excluded* the following data:

- 0 Non-dashboard (i.e. Answers) requests
- 489,576 requests that failed to complete
- 3,300,534 User Dashboard page (i.e. not `/shared`) requests

In identifying performance optimisation candiates it is important to consider the relevancy of the dashboards in terms of age. Dashboards that have not been run for a long time may be slow, but applying tuning efforts to them is most likely not time best spent – since no-one uses the dashboard any more. Therefore we have applied a **"stale threshold"** of 180 days, which of the 1264 dashboard pages in total has excluded 67% of them (853), leaving 411 dashboard pages to analyse. In addition, *only executions more recently than the stale threshold are included in the analysis*.

In order to focus performance analysis on reports that will give the greatest benefit we also ignore reports that are **only** used by less than 1% (6.99) of the active user population (699 in the last 180 days). Figure 1 shows the number of dashboard pages excluded under these criteria in the red-bounded box, and a list of them can be found in the appendix:

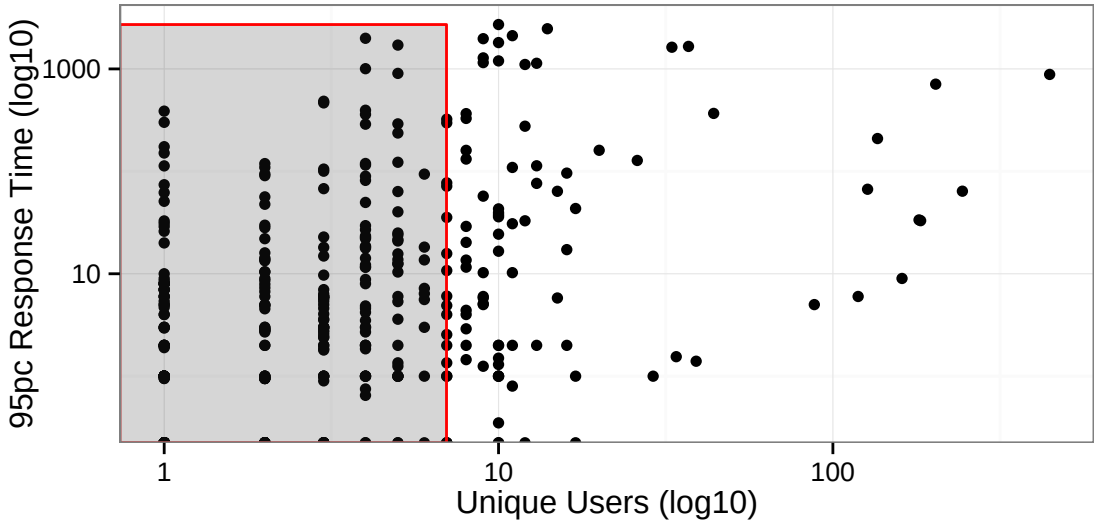


Figure 1: Per Dashboard Page, Unique Users/Response time distribution

Performance Summary

Within the analysed data (covering the last 180 days of 2752 available) there were:

- 101 different dashboard pages used
- 699 unique users
- 12,576 dashboard page executions in total.

Summarised By Execution

Having summarised and selected the dashboard pages that are to be analysed in detail, let us first look at the response time profile. Looking across all dashboard pages that were run, **half (i.e. median) of all the requests completed in 2 seconds or less**, and all but 5% (i.e. 95th percentile) completed in 375 seconds or less.

The full distribution of response times per dashboard page execution are shown in Figure 2.

Please see the appendix for a discussion of which measures are most appropriate to use for looking at performance; note that we specifically do not include the average (mean) here.

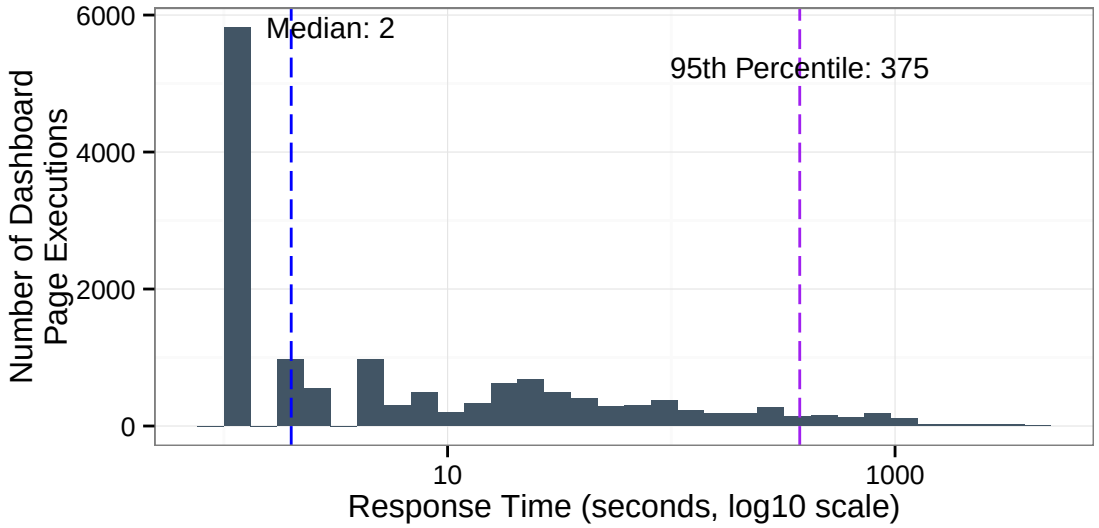


Figure 2: Response Time Distribution of All Analysed Dashboard Page Executions

Summarised by Dashboard Page

To understand how each dashboard page performs, and identify those with scope for performance optimisation, we now take the 95th percentile response time for each dashboard page. This gives us a measure by which to assess how each per dashboard page performs for most users.

Taking the summarised peformance of each dashboard page, we can say that :

- The median **95th percentile response time** for dashboard pages is 15.7 seconds. That is, of all the dashboard pages, half (50) run in this time, 95% of the time.
- Almost all (95th percentile) dashboard pages run in 1658.3 seconds or less, 95% of the time.

This distribution of the 95th percentile response time can be seen in Figure 3.

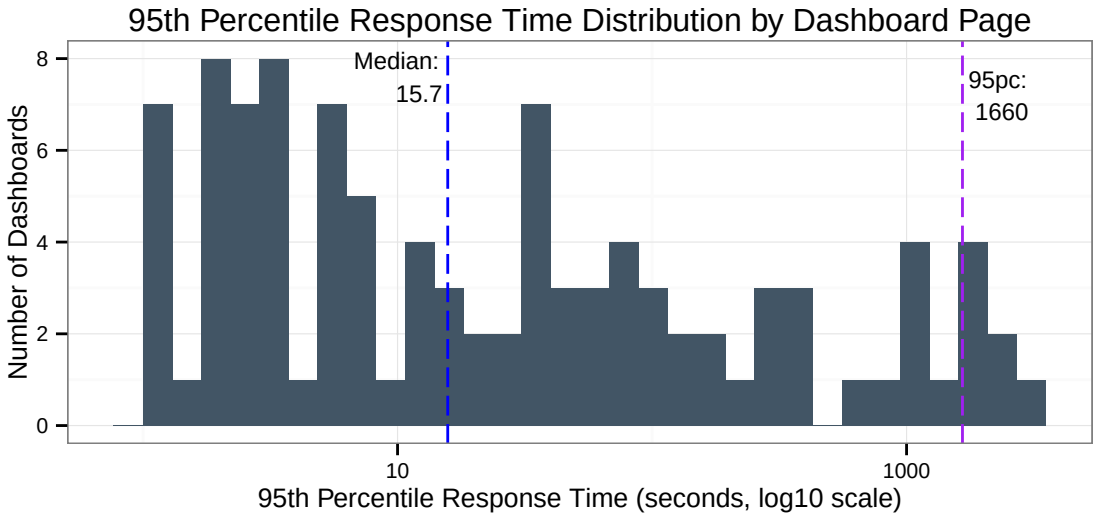


Figure 3: Dashboard Page - Summarised Response Time (95th percentile) distribution

Identifying Performance Optimisation Candidates

As a way of determining the biggest impact of a performance improvement we will consider the data in terms of the cumulative response time of each dashboard page by multiplying the number of executions by the mean response time.

Figure 4 shows the top 5 dashboard pages in terms of approxmiate total execution time, along with an overlaid indication of the number of unique users, executions, and 95th percentile response time. The data for this can be seen in Table 1. A full list of dashboard pages can be found in the appendix.

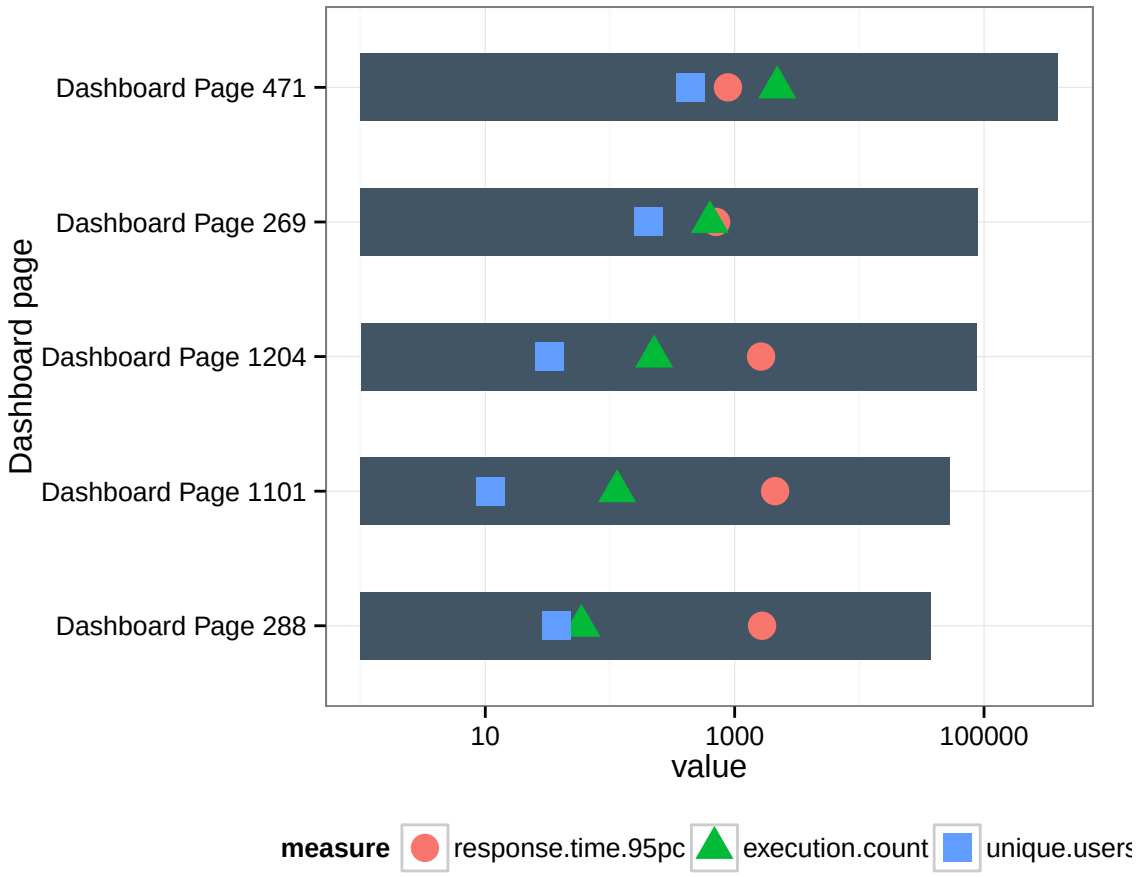


Figure 4: Top 5 dashboard pages by cumulative response time, with 95th percentile response time overlaid

Table 1: Top 5 dashboard pages by cumulative response time

Dashboard Page	Cumulative response time	Execution count	Unique users	95th pc time	Median time
Dashboard Page 471	394514	2195	445	883.30	32.0
Dashboard Page 269	89829	633	203	709.80	2.0
Dashboard Page 1204	86865	226	33	1628.25	93.5
Dashboard Page 1101	53287	114	11	2110.50	176.5

Dashboard Page	Cumulative response time	Execution count	Unique users	95th pc time	Median time
Dashboard Page 288	37393	59	37	1658.30	626.0

Figure 5 shows the top 5 dashboards in the context of all dashboards, in terms of unique user and 95th percentile response time:

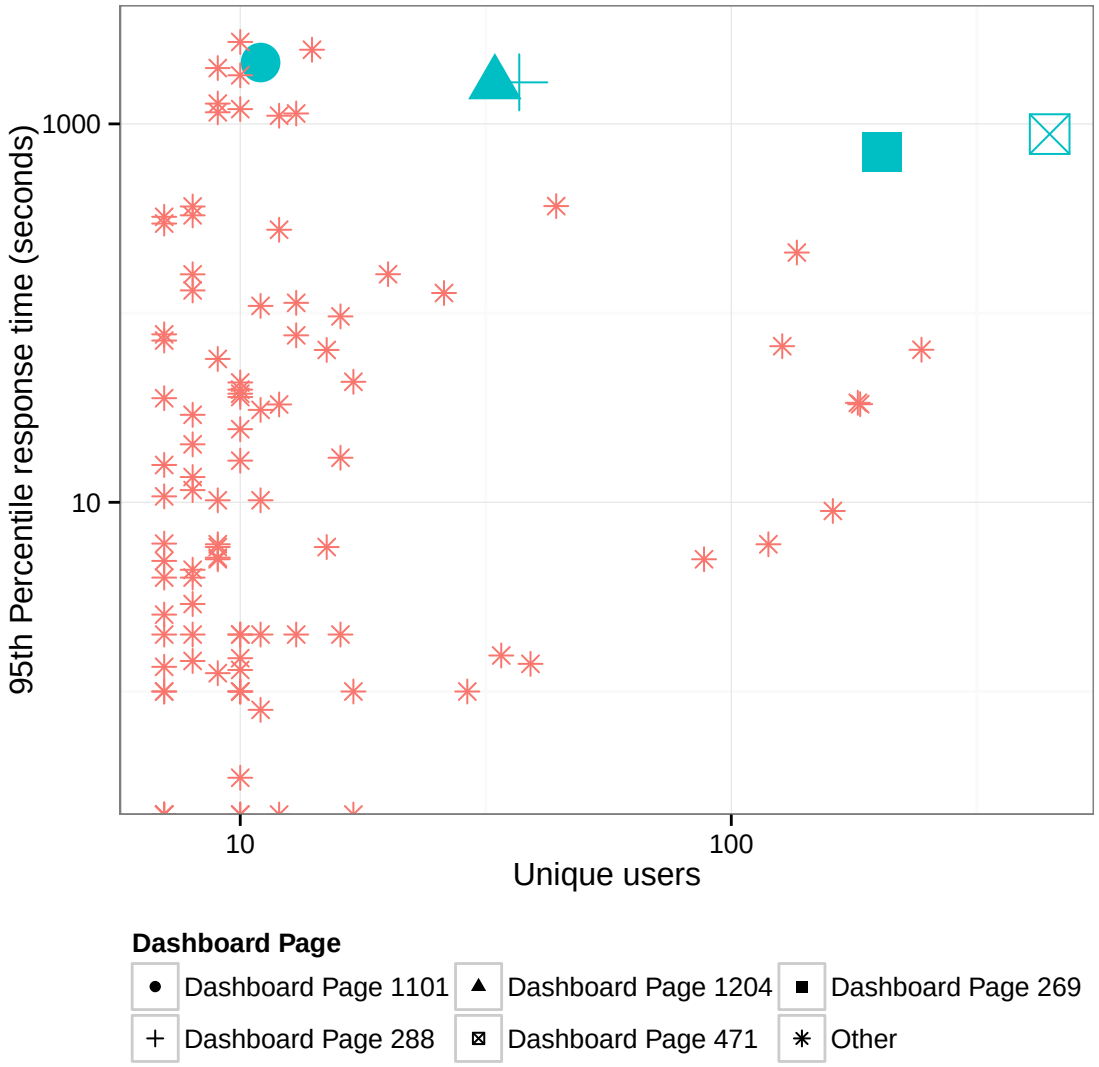


Figure 5: Dashboard Page users/response time, highlighting top-five by cumulative response time

Figure 6 shows the response time of each dashboard page execution plotted over time, so it can be seen if there has been a recent change in the performance profile:

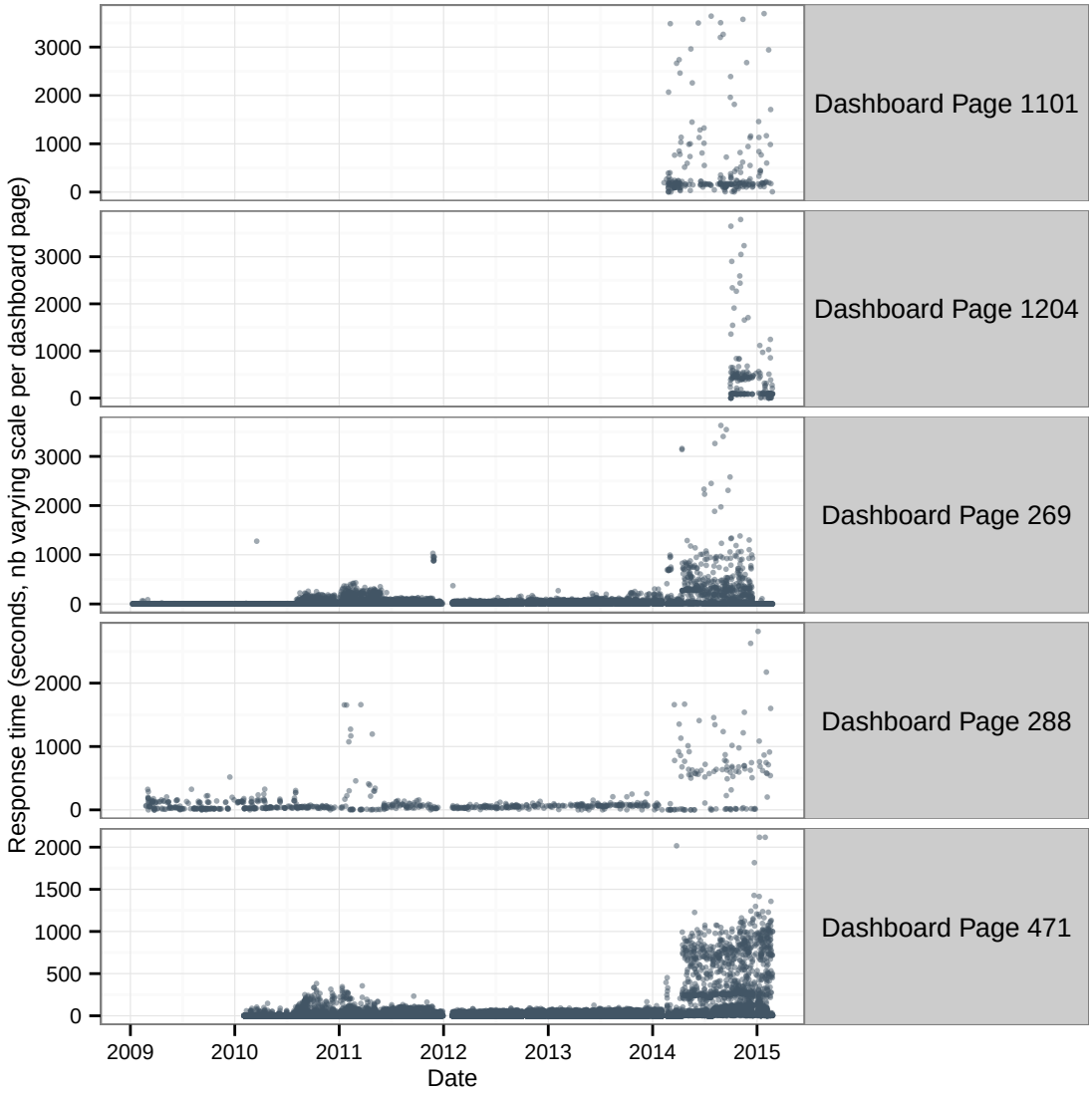


Figure 6: Dashboard Page performance over time

Analysing the Performance of a Dashboard Page

The next step in performance analysis is to analyse performance of the **analyses** within the individual **dashboard pages** that have been identified above.

Here we present an illustration of the analysis approach to take. In the Appendix of this report you can find a detailed breakdown of all 5 dashboards identified as candidates for optimisation.

General performance considerations for an analysis include:

- Database
- Is the datamodel implementation optimal (indexes, partitioning, etc)?
- Is the query being sent to the database optimal (eg are there joins or key columns missing that should be included in the RPD)?
- Is excessive data being returned from the database? Could additional filters (predicates) be added to the user query?
- BI Server (NQ)
- Is it having to filter or aggregate many rows from the database to few returned to the user?
- Is it having to stich together multiple physical resultsets?

Dashboard Page Performance Analysis: Dashboard Page 471

Totalling up the response times of each analysis gives us a picture of which analyses overall are taking the longest, shown here on a log scale in Figure 7, and the data in Table 2.

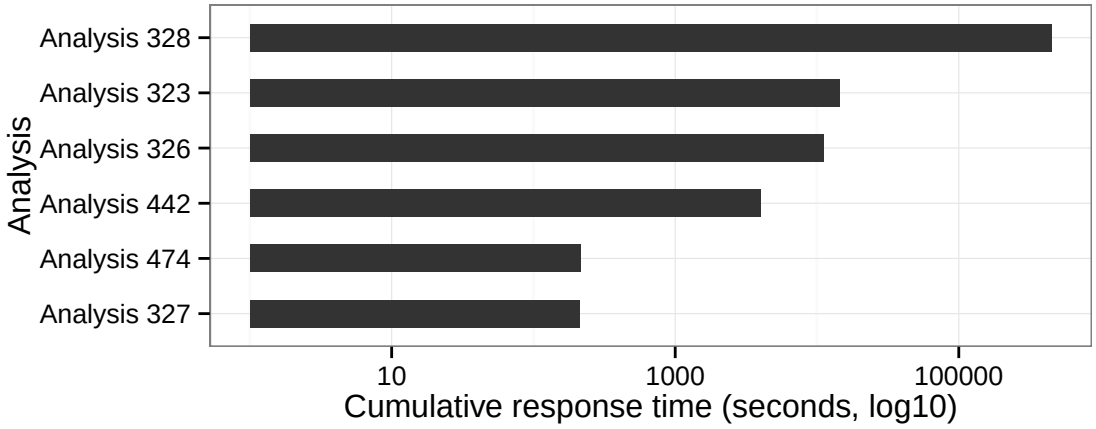


Figure 7: Cumulative response time of analyses in dashboard page `Dashboard Page 471`

Table 2: Analysis Cumulative Response Time

Analysis	Seconds
Analysis 328	453193
Analysis 323	14407
Analysis 326	11091
Analysis 442	4014
Analysis 474	216
Analysis 327	212

In subsequent analysis of these analyses, only the top five in terms of cumulative response time are examined.

The historical runtime trend for each of the top 5 analyses can be seen in Figure 8

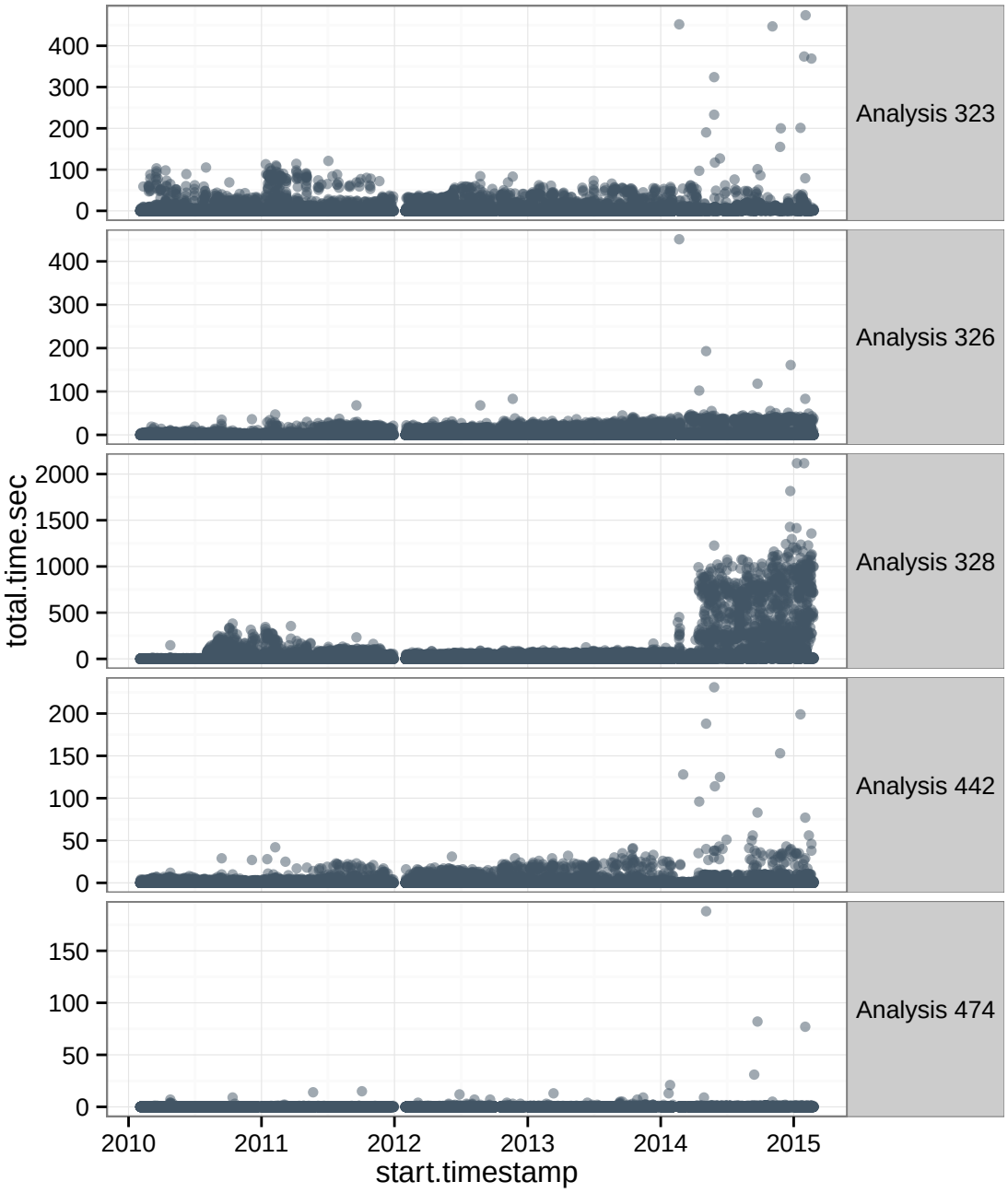


Figure 8: Top 5 Analyses - Response time, over time

Examining these analyses further, it is useful to see the distribution of reponse times for each analysis, shown here in Figure 9:

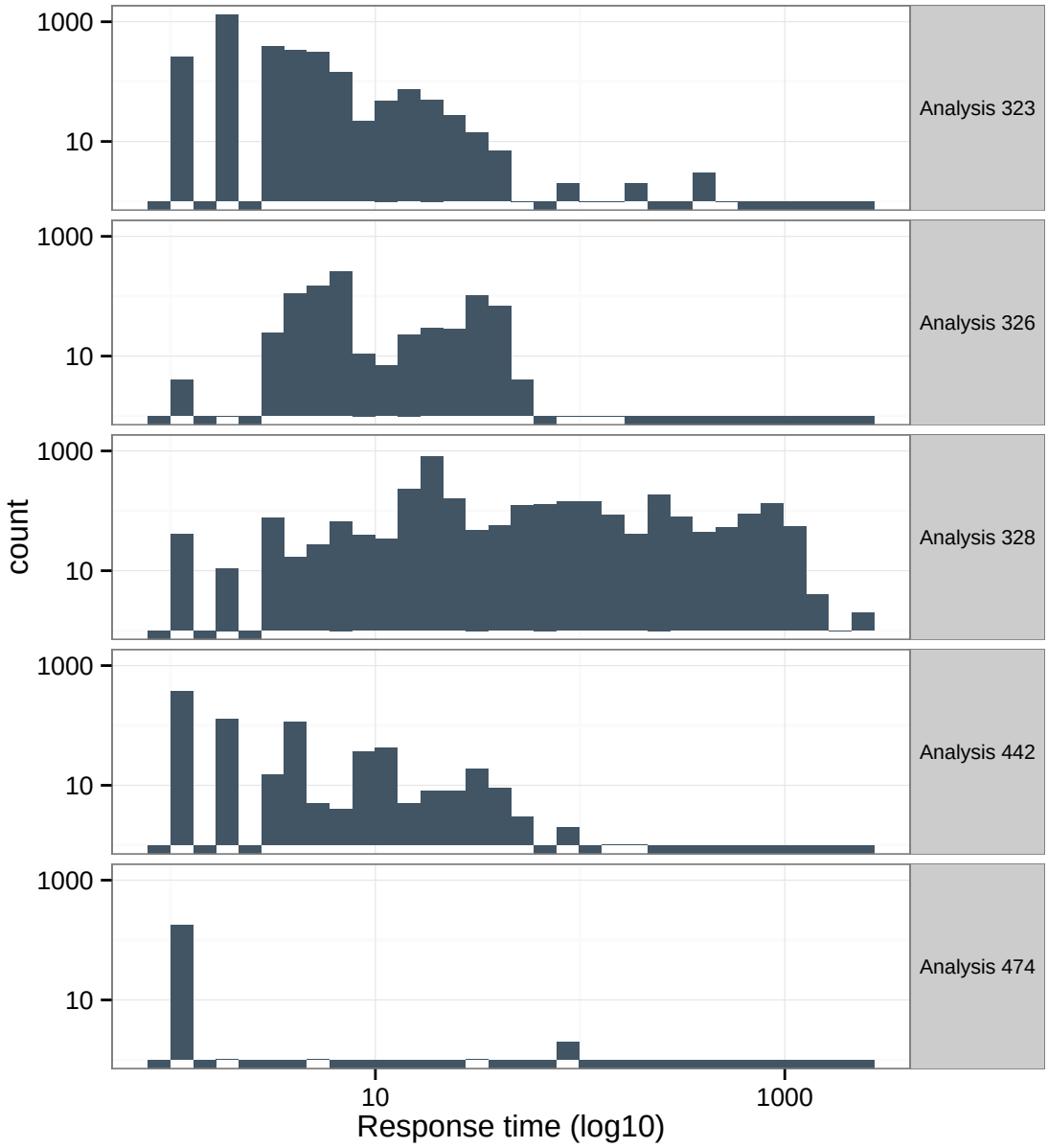


Figure 9: Response Time Profile of Top 5 Analyses in `Dashboard Page 471`

Of interest are (i) analyses taking a longer time to run than others, as well as (ii) analyses with greater varience in their response time. Response time varience can be accounted for by different predicates used when running a report (for example, selecting one year's worth of data vs one day's), so is not necessarily a 'problem'.

Within each analysis we can look at the **time profile**, showing where the time is being spent since this can often point to where performance can be optimised. Within the execution of an analysis on the BI Server the primary components of a time profile are as follows:

1. **Compilation** - *The 'logical' query as received from the client (usually Presentation Services) is compiled through the RPD to generate one or more 'physical' queries that will retrieve the data required from the database, and an execution plan of how the BI Server will process the data from the database into the logical resultset required by the client*
2. **DB time** - *This is the time taken executing a physical query on the database. Where an analysis generates multiple physical queries, the assumption is made here that they execute in parallel and therefore the maximum execution time of a single physical query is taken to represent physical execution time.*
3. **NQ time** - *After data is returned from the database the BI Server will process it into a single logical resultset to pass back to the client. In some cases there may be stiching, filtering, and/or aggregation that the BI Server has to do which can take significant time as a proportion of the overall response time.*

Note that this is the time profile within the BI Server component only; it does not include report rendering, transmission across the network to the user, etc

The distribution of these time components, along with the overall response time, is shown in Figure 10.

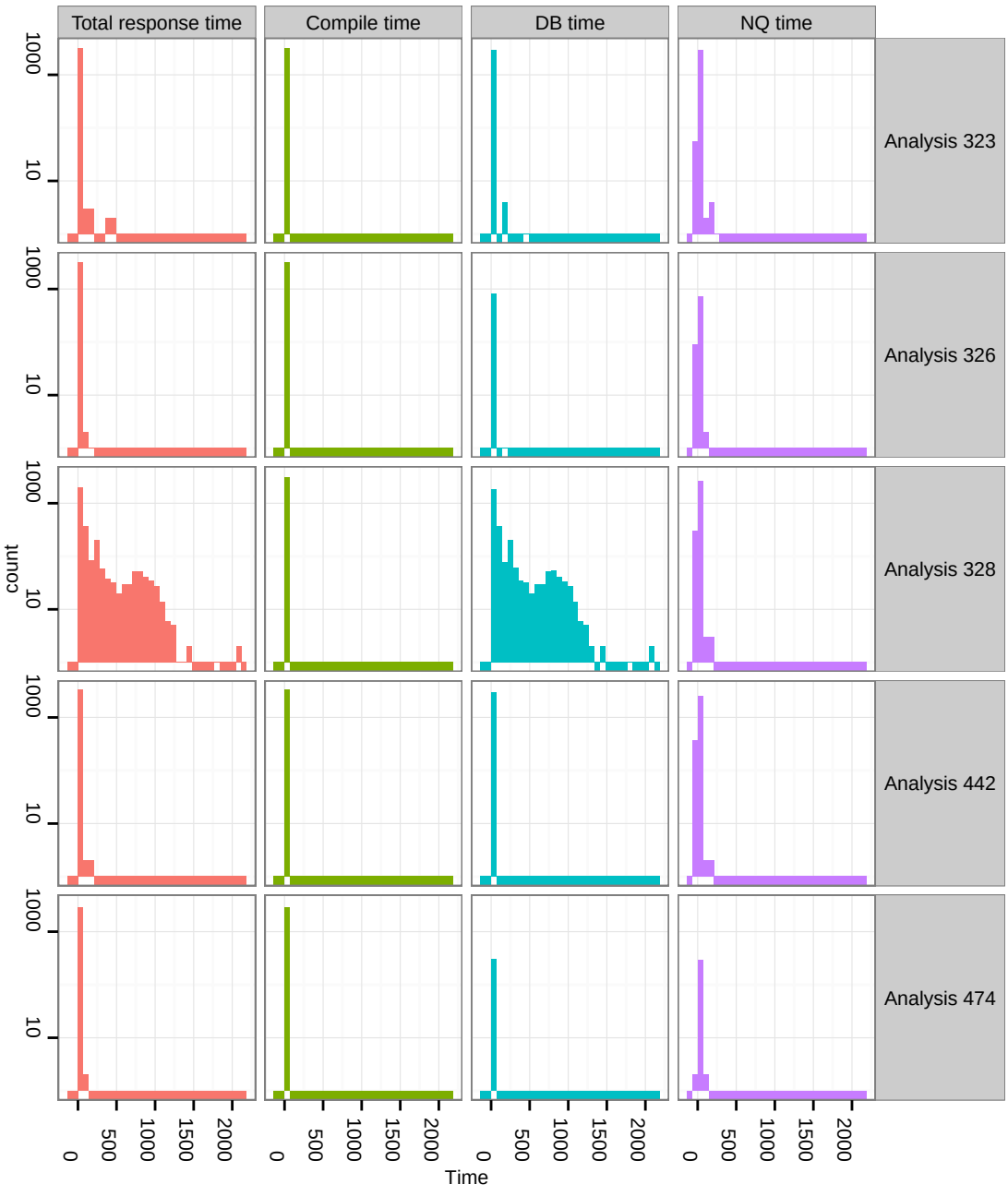


Figure 10: Time Profile Component Distribution by Top 5 Analysis

The median and 95th percentile of these distributions are shown here in Tables 3 and 4:

Table 3: Analysis Time Summary - Median

Analysis	Response	Compilation	Database	BI Server (NQ)
Analysis 328	23	0	22	0
Analysis 323	2	0	1	1
Analysis 326	6	0	5	0
Analysis 442	0	0	0	0
Analysis 474	0	0	0	0

Table 4: Analysis Time Summary - 95th percentile

Analysis	Response	Compilation	Database	BI Server (NQ)
Analysis 328	821	1	821.00	1
Analysis 323	14	1	12.95	1
Analysis 326	38	1	37.00	1
Analysis 442	4	1	4.00	1
Analysis 474	1	0	0.00	1

By summarising up each of the time components we can build a time profile for each analysis, shown in Table 5 and Figure 11. This indicates the general profile; it may mask variances within some executions of the analysis:

Table 5: Component Time as proportion of total response time

Analysis	Total Response (s)	Compilation %	Database %	BI Server (NQ) %
Analysis 328	453193	0.2	99.4	0.4
Analysis 323	14407	4.2	71.7	24.0
Analysis 326	11091	2.8	93.8	3.4
Analysis 442	4014	10.3	80.5	9.2
Analysis 474	216	2.3	0.0	97.7

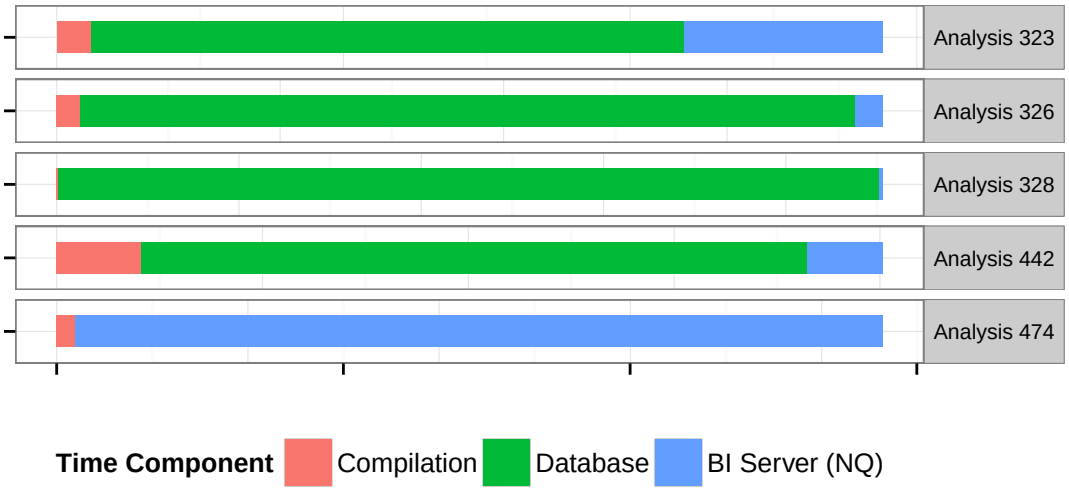


Figure 11: Time Components as proportion of total response time

Data fetch efficiency

Table 6 shows the number of rows returned from each database query compared to the number returned to the user, with the distribution of each execution shown in Figure 12. Generally these should be similar (i.e. a ratio close to 100%). If it is much less then it suggests possible inefficiency by returning data that is not needed or requires further aggregation.

Table 6: Rows of data handled

Analysis	Logical rows (total)	DB rows (total)	Ratio	SD
Analysis 328	5181358	5181678	100.0	4.9
Analysis 323	29427839	30028425	98.0	2.0
Analysis 326	12761	51204	24.9	0.5
Analysis 442	29361	212807	13.8	1.0
Analysis 474	35817	35817	100.0	0.0

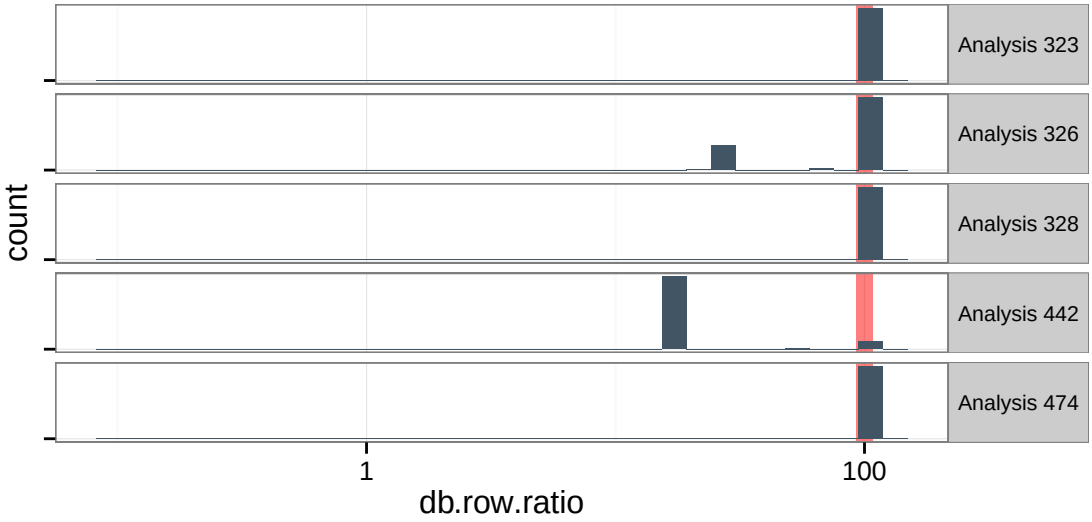


Figure 12: Row Ratio per Analysis Execution

Number of physical queries

The number of database queries that an analysis runs is shown summarised in Table 7 and shown as a distribution in Figure 13. Multiple database queries *can* (*but not always*) be inefficient as the BI Server has to combine the multiple resultsets into a single logical resultset. Sometimes multiple queries is unavoidable, for example when federating across multiple sources.

Table 7: Database queries per analysis execution

Analysis	Median	SD
Analysis 328	1	0.1
Analysis 323	1	0.1
Analysis 326	4	0.2
Analysis 442	1	0.1
Analysis 474	1	0.0

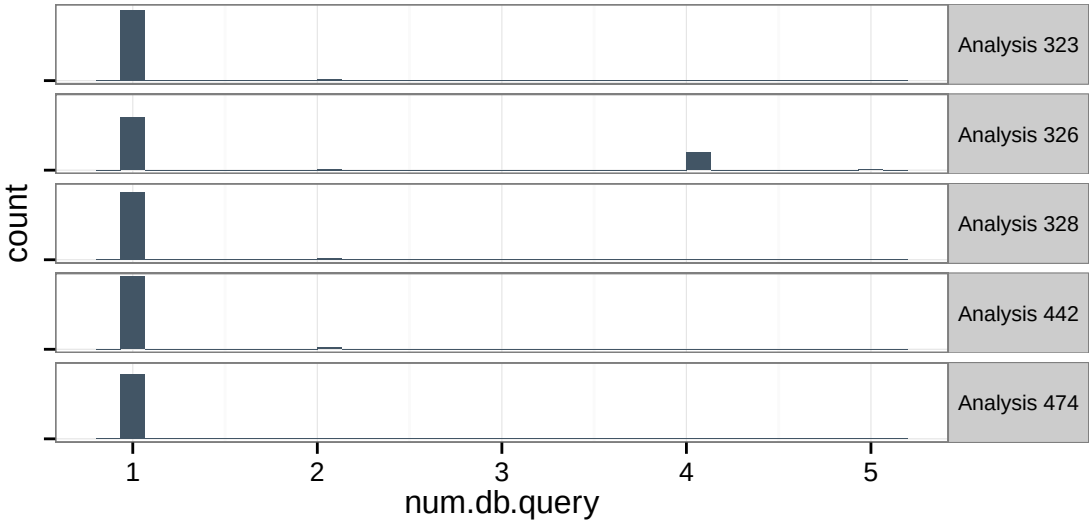


Figure 13: Database Queries per Analysis Execution

BI Server Cache usage

Table 8 shows the usage, if any, of the BI Server Cache, by each analysis. The distribution is shown in Figure 14. The BI Server cache can, used correctly, provide performance benefits. It is important to treat it as “the icing on the cake” rather than the first option to reach for in the event of performance problems lest it “paper over the cracks”.

Table 8: BI Server Cache hits per analysis excution

Analysis	Median	SD
Analysis 328	0	0.1
Analysis 323	0	0.1
Analysis 326	0	0.5
Analysis 442	0	0.1
Analysis 474	0	0.0

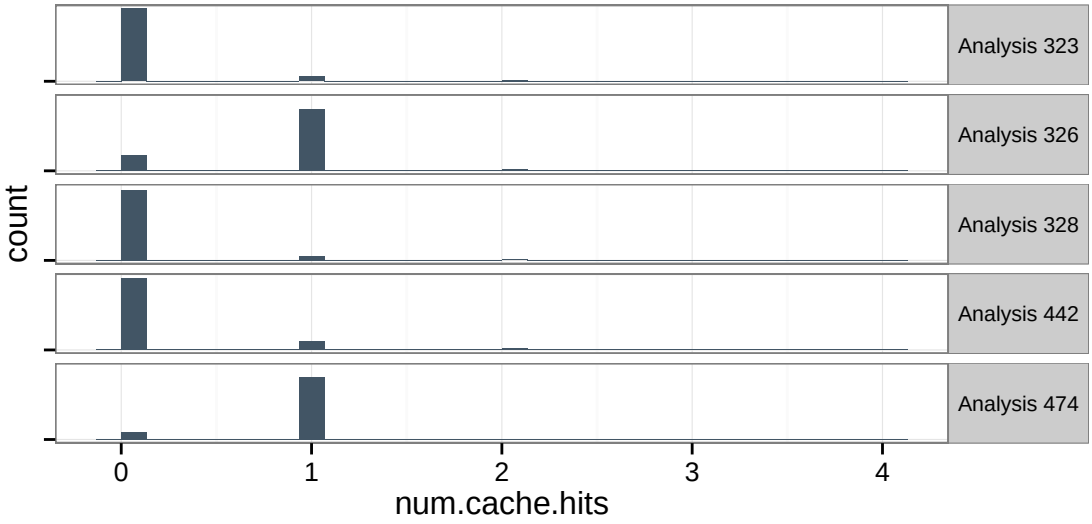


Figure 14: BI Server Cache hits

Report Health Check

In the previous section of this report the performance of analyses within the dashboard pages accounting for the longest cummulative response times was analysed. In this section section we consider the behaviour of analyses throughout the system, identifiying those that may need further examinination but are not yet key candidates for optimisation. *Analyses that have been run fewer times than the median (15) are excluded from this, as are those below the median 95th percentile response time (3.6 seconds).*

There are several patterns in OBIEE report execution performance that *generally* indicate sub-optimal performance. Inefficiencies such as these in a system may not be directly correlated to slow response times, but they will be consuming additional system resources that in time (or at periods of peak usage) may become exhausted. In each case the specific reason for the behaviour should be considered and if not justifiable then action taken to resolve or optimise it. Where the behaviour is directly causing poor response times these will be improved, and in general system resource will be freed up which will prolong the capacity of the system.

We should emphasise here that these are not absolute and in some cases they are difficult to practically avoid.

BI Server Time

In an optimal system the time spent in the BI Server should be minimal as *database pushdown* should occur, whereby all of the "heavy lifting" against the data is done by the database. However in some cases the BI Server may end up retrieving raw data from the database and performing additional processing (such as filtering, joining, and aggregation) against it itself. This is inefficient for several reasons including the volume of data transferred from the database, and the disk I/O and space usage required by the often-necessary work files that the BI Server writes.

Table 9 shows analyses with a high proportion of time spent in BI Server (NQ), and the distribution of this across all analyses is shown in Figure 15:

Table 9: Top 10 analyses by 95th Percentile NQ time

Analysis	Executions	95pc response time	95pc NQ time	NQ %
Analysis 2279	188	927.10	853.10	92.0
Analysis 2388	83	946.30	789.10	83.4
Analysis 1430	34	917.30	700.20	76.3
Analysis 2277	87	651.90	647.80	99.4
Analysis 2161	40	402.65	402.60	100.0
Analysis 1890	650	473.30	387.00	81.8
Analysis 2450	47	369.10	363.80	98.6
Analysis 1904	647	442.80	358.90	81.1
Analysis 2335	539	628.60	315.40	50.2
Analysis 2386	594	367.70	293.75	79.9

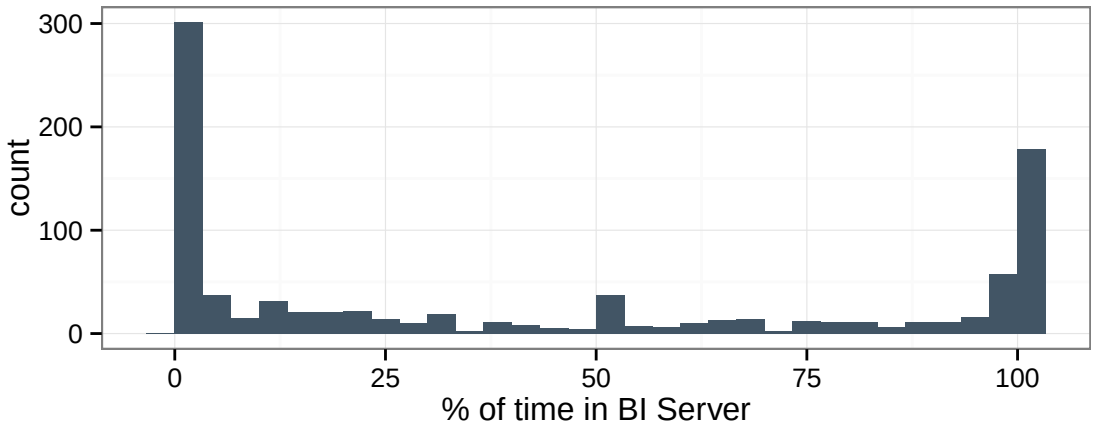


Figure 15: By analysis, 95th percentile of % of time spent in BI Server

High number of physical queries

OBIEE supports *query federation* whereby it issues multiple queries to one or more data sources and then combines the results together into a single logical resultset. This is a very powerful feature but when used unintentionally or unaware can lead to sub-optimal performance because the BI Server has to perform work on joining the data, as well as any aggregations and filtering only being able to take place after the join, necessitating the retrieval of a larger amount of lower-granularity data from the data sources than may otherwise be required.

If you have a single data source residing in a modern RDBMS (such as Oracle) then *in general* you should expect to see just a single query issued (there are notable exceptions to this including on Exalytics with TimesTen). For multiple data sources query federation will happen and by design. In cases where performance is unsatisfactory then caching *may* help, but for optimal performance the entire data set should reside in a single database.

Table 10 shows the top 10 analyses ranked by number of physical queries and response time. The distribution is shown in Figure 16.

Table 10: Top 10 analyses with high number of physical queries and response time

Analysis	Queries	NQ time %	Executions	95pc response time
Analysis 2048	4	1.6	72	61.9
Analysis 2062	5	37.8	54	61.7
Analysis 2167	4	76.4	144	255.0
Analysis 2383	9	13.2	293	98.4
Analysis 2389	4	2.8	471	36.0
Analysis 326	5	2.6	820	38.0
Analysis 2384	18	16.7	123	6.0
Analysis 2390	8	5.9	368	17.0
Analysis 473	4	16.7	1385	12.0
Analysis 676	4	16.7	443	24.0

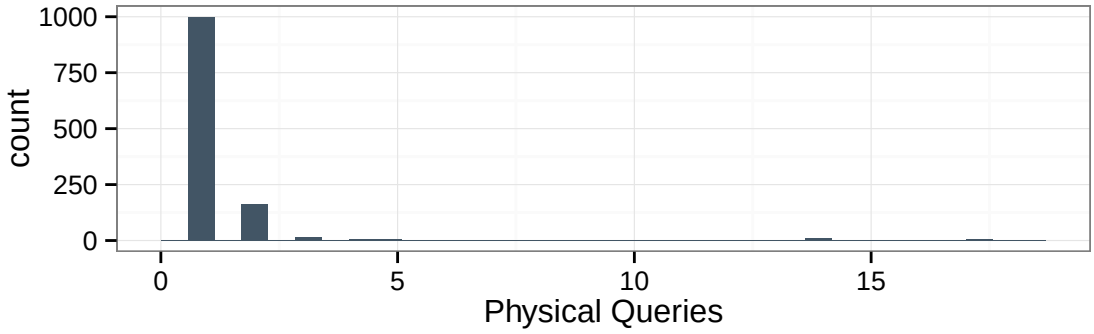


Figure 16: By analysis, maximum number of physical queries per execution

Large Amounts of Data Returned to User

A key principal of good performance is to minimise the amount of data being processed by the system. Table 11 shows the top ten analyses in terms of median rows returned to the user.

Table 11: Top 10 analyses by user row count

Analysis	Median rows	Max rows	Exec count	95pc response time
Analysis 1629	17644	21982	251	283.5
Analysis 2531	16038	16740	87	6.0
Analysis 1673	14235	45360	71	20.0
Analysis 1625	9645	17789	123	13.0
Analysis 323	9296	1000202	3002	14.0
Analysis 2120	8990	9293	71	4.0
Analysis 1911	8063	8579	48	60.0
Analysis 2062	7207	543355	54	61.7
Analysis 2048	6955	432864	72	61.9
Analysis 1715	6657	42744	34	4.7

Where analyses are returning large amounts of data it can be caused by several factors:

1. Users omit to include appropriate filters
2. Dashboard prompts are not given default values, so reports execute without data restriction before users then specify the required data.
3. OBIEE is sometimes used as a means of simply extacting large volumes of data for use in another tool (commonly Excel). Done incorrectly this can put unnecessary strain on the system.

Where OBIEE is being used as a data extraction tool there is a document from Oracle (doc ID 1558070.1[3]) that details optimal methods to follow, of which export from OBIEE through Answers/Dashboards is not recommended except for minimal volumes.

[3] My Oracle Support, "OBIEE 11.1.1.7 - New Features, Export Guidance And Recommendations For Working With Microsoft Office (Doc ID 1558070.1)"

Figure 17 shows the distribution of the number of rows returned to the user (note that this is a log10 scale), and Figure 18 compares the response time of an execution with the number of rows returned to the user.

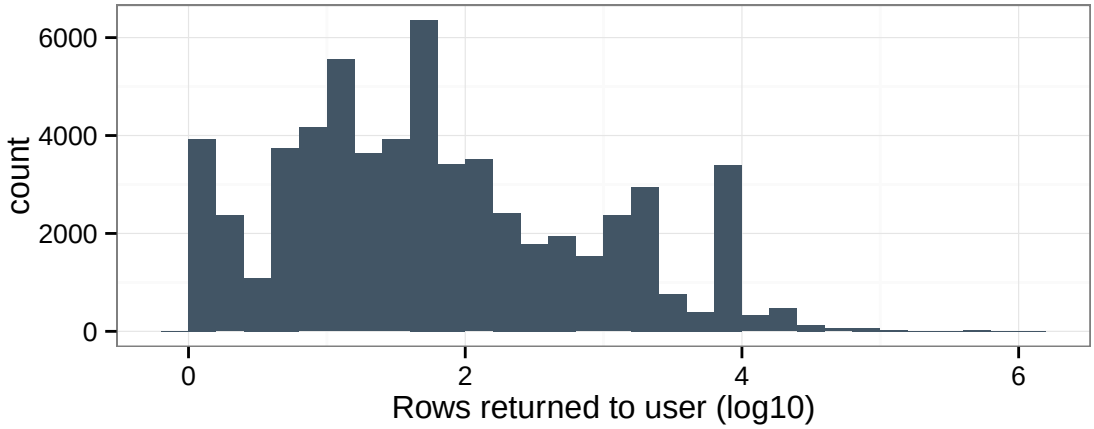


Figure 17: Distribution of rows returned to user per analysis execution

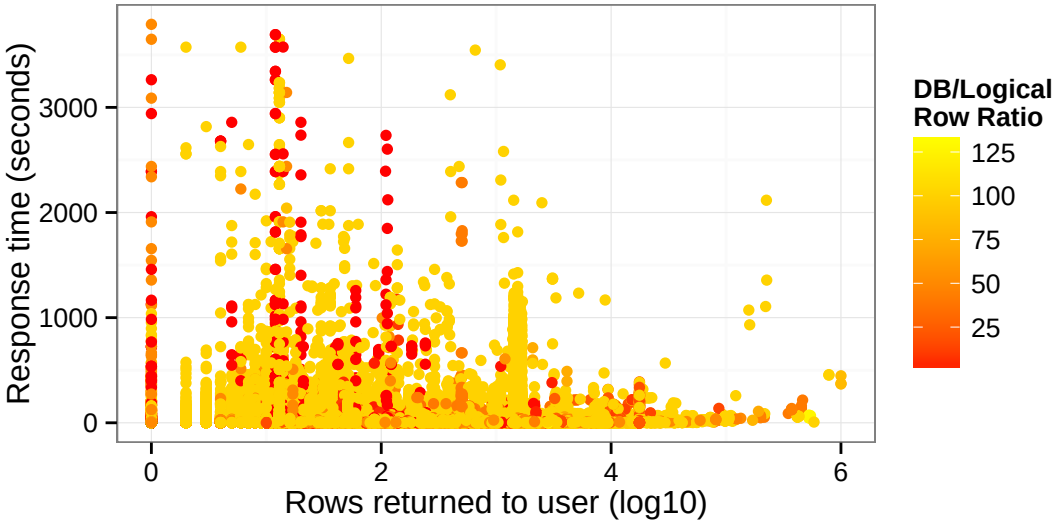


Figure 18: Response time vs Number of rows returned to the user

Inefficient Data Retrieval

In an optimal system the least amount of data needed is retrieved from the database by the BI Server. When this doesn't happen additional load is placed on the database, the network, and the BI Server. This behaviour is commonly seen in conjunction with high BI Server time, since the BI Server has to then process and filter/summarise the extra data. Often large temporary files will be seen on the BI Server when this happens.

One method for analysing the efficiency of the data retrieval process is to look at the ratio of rows returned from the database to those returned to the client. If all the rows fetched from the database are passed to the client (ratio of 100%) this suggests minimal inefficiencies.

Table 12 shows the top 10 analyses with a poor ratio of data returned from the database vs that returned to the user.

Table 12: Top 10 analyses by Row Ratio (Logical:Physical)

Analysis	Row Ratio	Logical rows	Physical rows	95pc response time	Executions
Analysis 2135	0	13	168012	21.00	91
Analysis 2279	0	20	505710	927.10	188
Analysis 2285	0	95	285290	219.50	103
Analysis 2287	0	95	285290	219.70	103
Analysis 2298	0	40	273197	89.00	135
Analysis 2299	0	40	532785	96.95	188
Analysis 2320	0	183	526198	218.60	104
Analysis 2335	0	12	583021	628.60	539
Analysis 2388	0	60	6601492	946.30	83
Analysis 2435	0	40	265533	91.00	145

The ratio for analysis executions vs the reponse time is shown in Figure 19.

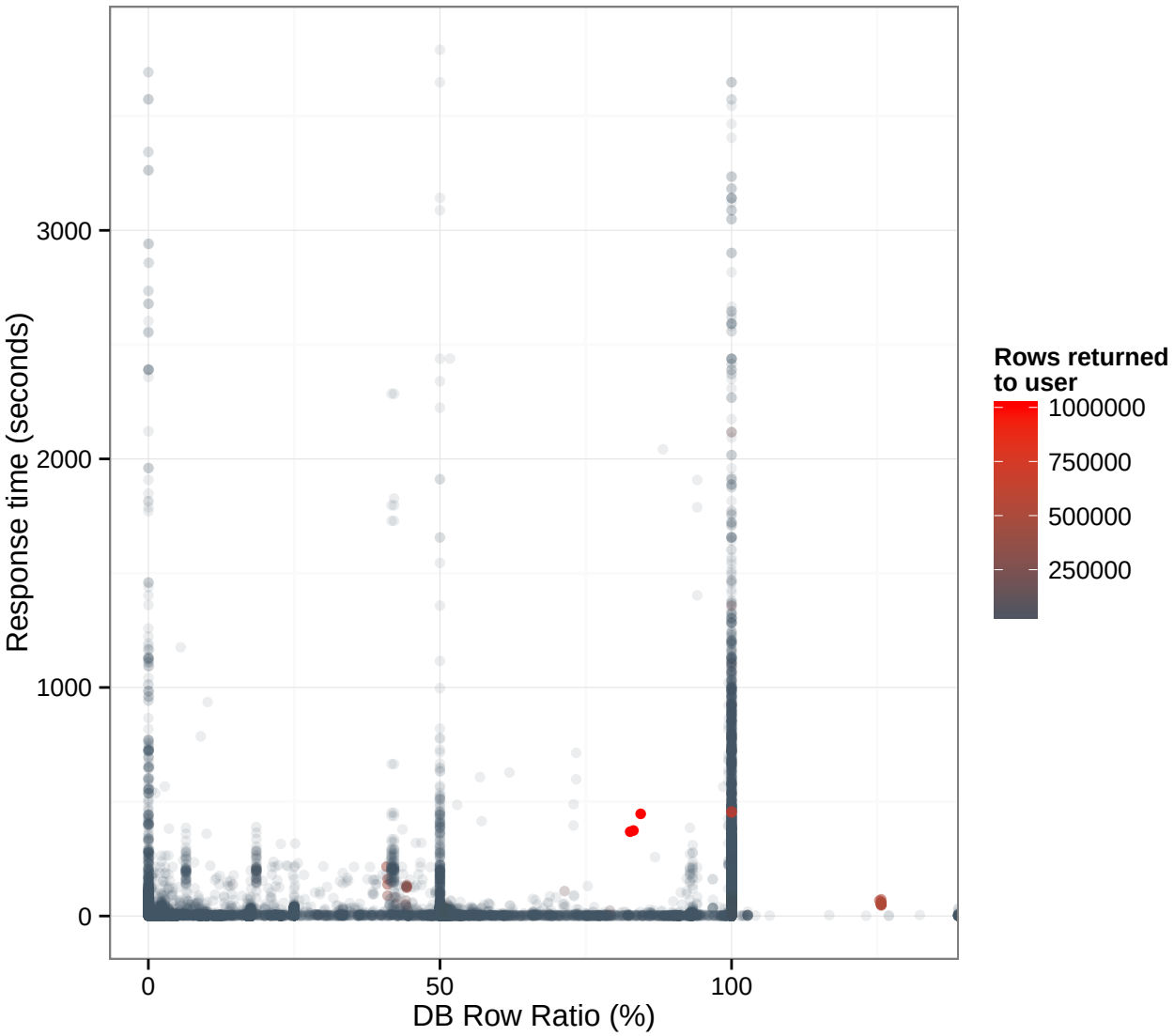


Figure 19: How is response time per execution affected by the DB/logical row ratio?

Dashboards with high number of analyses

A well designed dashboard page should present the user with the data in such a way that the user is not overwhelmed nor lost in what they are looking for. As well as aiding user experience with the data, it is a good idea to remove extratraneous analyses from a dashboard so that they are not executing and redundantly consuming system resource.

Table 13 shows the top ten dashboard pages with the most analyses on them. They should be reviewed for design and user experience:

Table 13: Top ten dashboard pages by analysis count

dashboard.page	analysis.count
Dashboard Page 1182	83
Dashboard Page 1002	54
Dashboard Page 1097	51
Dashboard Page 685	30
Dashboard Page 1191	27
Dashboard Page 696	27
Dashboard Page 1206	24
Dashboard Page 1199	23
Dashboard Page 1200	20
Dashboard Page 1207	20

Recommendations

This report provides the starting point for identifying performance improvement opportunities in your OBIEE deployment. We recommend working with a performance specialist from Rittman Mead to investigate in detail the following :

1. Dashboards accounting for the greatest cummulative runtime, listed in Table 14:

Table 14: Top 5 dashboard pages by cumulative response time

Dashboard Page	Cumulative response time
Dashboard Page 471	394514
Dashboard Page 269	89829
Dashboard Page 1204	86865
Dashboard Page 1101	53287
Dashboard Page 288	37393

2. The analyses listed in the **Report Health Check** section.

In addition, feedback should be solicited from users directly as to any performance concerns that they may have so that they can be included in the prioritisation activity.

Appendix A - Miscellaneous

Which Response Time Metric is Representative?

When examining performance it is vital to use an accurate measure of response time, one that is representative of user's *experience*.

When summarising up a set of response times the **95th percentile response time** is commonly used, since it represents the maximum time that nineteen out of twenty users will have experienced. Using an average (**mean**) is notoriously unrepresentative because extremes at either end of the distribution can skew it.

Using a log scale to show the data profile better (given the large range of response times), Figure 20 shows the distribution of the response times:

In terms of the response times seen, the *overall median response time* was 2 seconds and *95th percentile response time* was 375 seconds.

What we can see here is that 95th percentile (375) is indeed a representative measure to be using. If we were to use the Average (mean) of 68.4 seconds it's clear from the graph that this is unrepresentative of

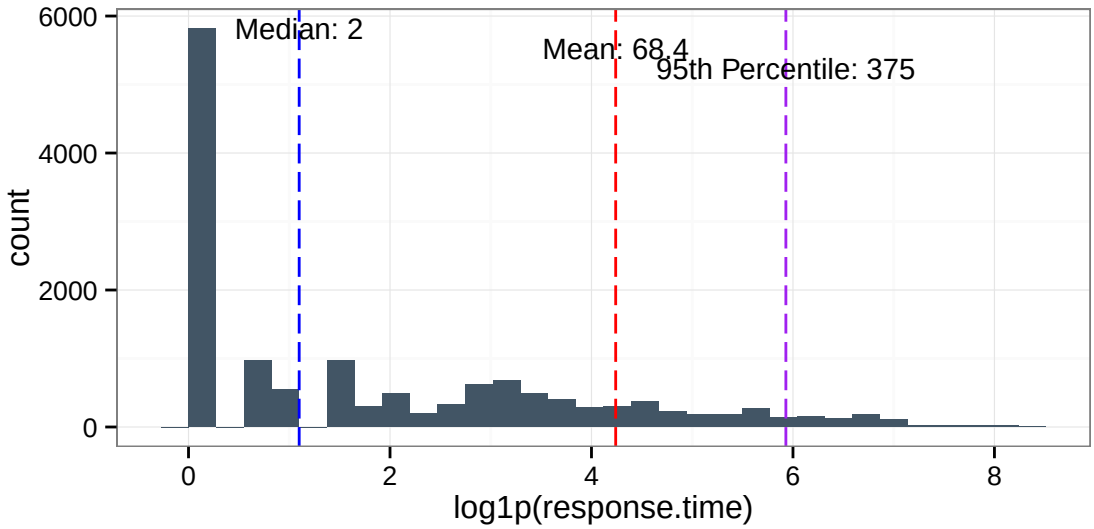


Figure 20: Median, Mean, and 95th Percentile Response Times

a lot of the response times experienced by users, whilst the median (2) excludes (*by definition*) half of all user experiences.